# Design and Implementation of 3D Education Platform for Humanoid Robot Based on Unity3D

**Kailin Zheng, Baoling Han, Qingqiang Wang, Xinda Wang**

Beijing Institute of Technology, School of Mechanical Engineering, Beijing, China

Email: 18810772915@163.com

**Keywords:** robots education; software platform; part model assembly; 3D visual programming

**Abstract:** In this paper, I design a multi-functional universal software platform suitable for robot education teaching, striving to make breakthroughs in interactivity and training young people's capacities of logical thinking and spatial imagination. Unity3D engine is used for control software and function platform development, enhancing the effect of 3D visualization. 19-DOF humanoid robot developed by our laboratory is used as teaching aid to realize part model assembly and control of the robot in 3D visualization environment. The robot software platform realizes the part assembly and motion control of the robot well. Experiment tests suggest that the educational robot software platform designed in this paper has good interactivity, stability and easy operation, which greatly enhances the cognitive level to robots of young people.

## 1. Introduction

With the rapid development of robotics and the continuous advancement of robot education, it is becoming more and more important to strengthen the cultivation of backup innovative talents who master robotics [1]. The platform of robot education plays an extremely important role. The educational robots of the Harbin Institute of Technology Robotics Group are biased into physical assembly; the educational robotic product technology of Shenzhen Ubtech Company is completely encapsulated. The degree of integration with the robot education system is not high, and it is biased towards entertainment. I designed a software platform suitable for robot education to realize the assembly and control of robots in the 3D visualization environment, to enhance the cognition, logical and spatial thinking ability of students. In this paper, I mainly study the architecture design of the educational software platform, communication module design, 3D model assembly module design and 3D visual programming design.

## 2. Analysis of functional requirements

### 2.1 Functional requirements analysis of 3D model virtual assembly module

In this paper, the three-dimensional virtual assembly module of robots will be developed, so that students can learn the knowledge of robotic mechanical structure system, parts assembly and robotic control through the three-dimensional virtual assembly method of software. As shown in Fig.1, robot virtual assembly function module includes five branches.

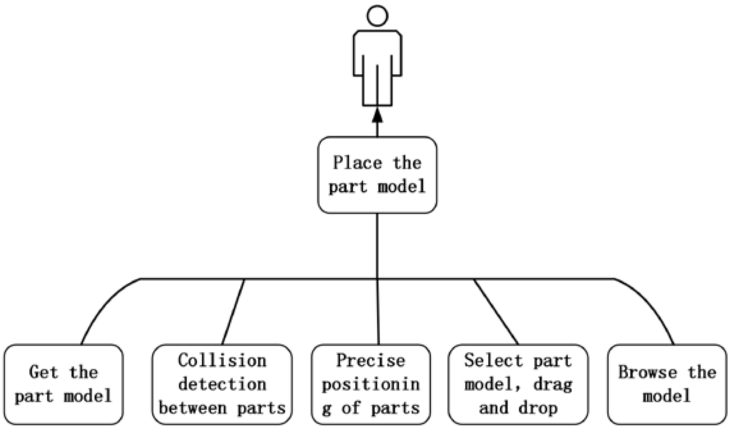

Place the part model
Get the part model
Collision detection between parts
Precise positionin g of parts
Select part model, drag and drop
Browse the model

Fig. 1 Robot virtual assembly function module

The general steps for robots model assembly are as follows:

Step1: Open the software and select "Assembly" .

Step2: Click the part library on the interface to load the part model, 360 degree rotation and zoom; Click "Delete" to delete the unsuitable model.

Step3: Select the assembled part model, drag and drop it.

Step4: Drag and drop the part to the corresponding position to perform collision detection and detect whether there is positional intervention.

Step5: The part model is automatically positioned to the correct position to complete the part placement.

Step6: Repeat in turn until the robot model is fully assembled.

### 2.2 Analysis of 3D visual programming module functional requirements

At present, most educational robot control software adopts the method of graphical control, which is to abstract the data structure and function through the primitives, and to express the flow of the program by using the connection, drag-and-drop programming, and user operation is convenient [2], so there is no need to have strong expertise in computer programming to meet the education of primary and secondary school students. However, most of them use 2D primitives, which are poorly interactive and not intuitive enough. Therefore, designing a control software in this paper that can control the joint motion of solid robots by directly changing the angle of the 3D model. The 3D control function module mainly includes changing the pose of the model, transmitting data and saving data.

The general steps for 3D visual programming module are as follows:

Step1: Select "3D Control" on the software homepage;

Step2: The student clicks on the corresponding joint and selects the corresponding joint;

Step3: Input the angle to be changed and change the selected joint;

Step4: Save the changed joint angle data;

Step5: Click the “Send” button to send the data to the interface window and send data to the slave computer to enable the robot to perform the corresponding actions.

## 3. Design of software platform

The software system design process is roughly divided into the following parts: the development engine selection, software architecture design, communication module design, 3D model virtual assembly module design, 3D visualization programming module design. Unity 3D is selected as main engine to develop the robot software platform.

### 3.1 Design of software platform architecture

According to the MVC mode (Model View Controller), the software system is divided into four layers, namely user layer, view layer, controller layer and model layer, from top to bottom [3].

The main body of the user layer are users, interacting with the view layer, and triggering the response of view layer.

The application view layer, which manages the user interface. The main specific functions include: layout, user input and reception, animation playback, and processing of object texture. In Unity development, the third-party plug-in NGUI is used for UI layout and 3D model displaying in the scene [4].

ViewPresenter is a collection of operations of the view layer which is equivalent to the properties of the view layer. It can be called by the GameObject. GetComponent in the controller, such as Label and Sprite in the set button.

The controller layer can store the state of the view layer and change the state of view based on the trigger event, which is used to connect the Model layer and the View layer. The main functions are: handling the events triggered by the user in the View layer, controlling the process of the view layer, and loading or birthing the resource model.

The Model layer refers to the data model. The main features are: It can only be accessed to by Controller or other Model layer, does not save any data or state of the View layer and notify external systems by triggers events for processing and changes.

### 3.2 Communication module design

The serial communication based on C# language is mainly applied to the communication among the 3D visualization control module, the application module and the robot slave computer control.

In the 3D visualization control module, the state of the robot to be moved is obtained by changing the angle of joint model in the software platform, and then the state data (19 steering angle values) is sent to the robot slave computer control system.

The code classes for serial communication are OpenSerialPortScript.cs and DealSendDatas.cs. The former class is in charge of the explicit serial port connection and the open serial communication, and the latter is in charge of data acquisition, parsing and sending. The main process includes setting serial port parameters, opening the serial port, obtaining data/instructions to be sent, and accepting and displaying the transmitted data.

The communication protocol is showed as follows: Status data is sent in the 3D visualization module. Each transmission of data is equivalent to controlling a single action frame, sending the status of 19 joint servos at a time, and each servo state is represented by three data: the ID of the corresponding servo, the angular position, and the rotation speed. Referring to the AX-12A servo manual, the servo ID can range from 0 to 254, the range of motion is 0 to 1023, and the range of motion speed is 0 to 1023. Therefore, each servo status parameter can be set to “XX_XXX_XXX”. "The entire data frame format including 19 servos status is "XX_XXX_XXX#XX_XXX_XXX#...#XX_XXX_XXX", and each servo state parameter is separated by "#".

### 3.3 Design of robot 3D model assembly module

The robot 3D model assembly module mainly help students to be able to assemble robots online using 3D model parts [5]. The following is a detailed exploded view of the 3D model assembly software: including model icon interface, model assembly system and model editing toolbar. The model icon interface is a way to obtain the corresponding model. The model is loaded into the assembly scene by clicking the corresponding interface of the icon interface. The model assembly system contains a series of functional operations to assemble the model, including picking model, dragging model, collision detection, precisely positioning of model and browsing in all directions. The class figure frame involved in the 3D model assembly module is shown in Fig.2.

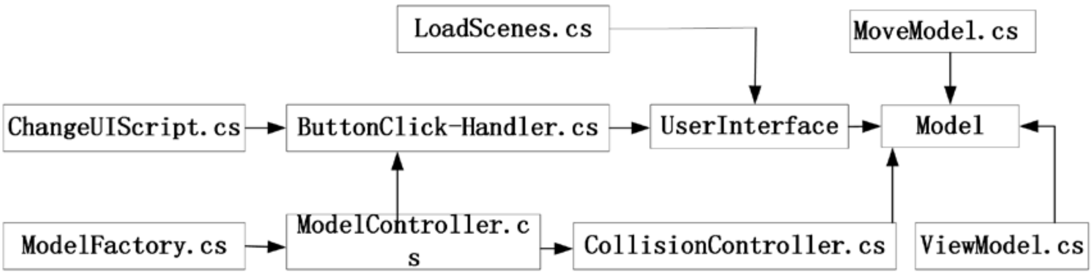


Fig.2 Class design involved in 3D model assembly module

The detailed explanation of class name is shown in Table 1.

Table 1 The explanation of class

| ID | class | Function description |
|---|---|---|
| 1 | LoadScenes.cs | In charge of scene switching. |
| 2 | ButtonClickHandler.cs | Assigning model loading events to model icon buttons and calling modelController class. |
| 3 | ChangeUIScript.cs | Dividing many model icons into three Widget interfaces, switching the model icon interface. |
| 4 | ModelController.cs | Managing the status of the model after loading, and calling the ModelFactory class. |
| 5 | ModelFactory.cs | Loading the model in the scene According to the given model address. |
| 6 | CollsionController.cs | Accurate positioning of the model after correct collision. |
| 7 | ViewModel.cs | Users can rotate the view and shrink angle from all directions to see the specific location and structure of the model. |
| 8 | MoveModel.cs | Carrying out picking, dragging, and deleting operations. |

### 3.4 Design of 3D visual programming module for the robot

The robot 3D visual programming module means that the student can control the corresponding motion effect of the robot by changing the joint angle of the 3D model of the upper computer. The following are the functions of the robot 3D visualization programming software divided into model joint drive, drive data preview and drive data transfer. Model joint drive is a function to change the joint angle of the 3D model. By selecting the corresponding model, input the angle that needs to be changed to change the joint angle. The drive data preview refers to displaying the current pose of each joint of the robot in the interface, that is, the angle of rotation of each joint. Drive data transmission refers to the transmission of the joint data by means of serial communication to the robot's lower position machine, so that the robot completes the corresponding joint drive.

## 4. Implementation of software platform

This chapter mainly introduces the realization of communication module, part assembly module and visual programming module.

### 4.1 Realization of main interface of software platform

The interface can be divided into two parts: UI icon interaction interface and 3D model scenario interface. [6] Users can enter the part assembly of robots, three-dimensional visual programming control and virtual world by clicking on the "assembly", "control" or "virtualization" buttons in the main interface individually. The robot preview model in the main interface can be switched by clicking on the "preview mode" button. Main function interface is  shown in Fig.3.

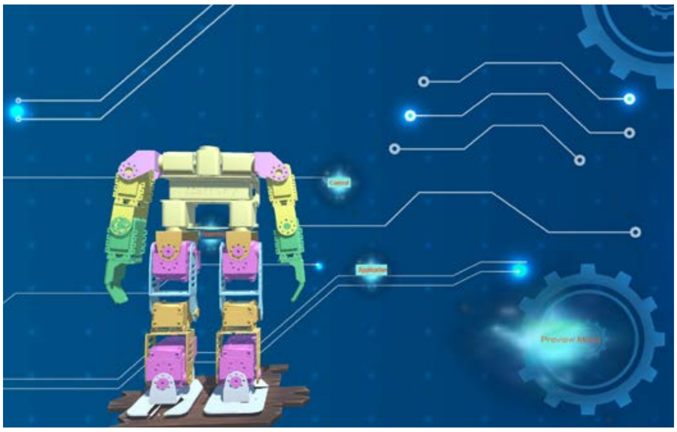


Fig. 3 Main function interface

### 4.2 Implementation of 3D model assembly module

The virtual three-dimensional model assembly module scene is showed by triggering the "assembly" button. Firstly, the user clicks on the corresponding part on the left interface, and the scene loads the corresponding model; then clicks on the selected part to be assembled for dragging; drags to the corresponding model location, and collision detection occurs; If the collision detection is correct, then the part model is automatically and precisely positioned to the corresponding location to complete the parts placement as shown in Fig.4 and Fig. 5.

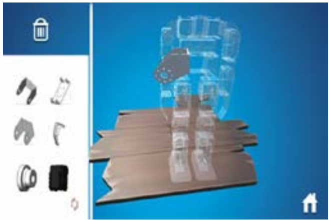


Fig. 4 Place the part

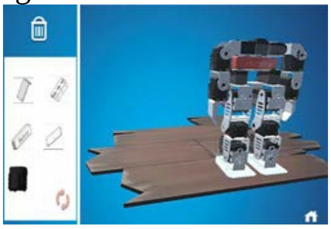


Fig. 5 Complete assembly

Enter the 3D visualization programming module scene by triggering the "Control" button. First, the user clicks on the corresponding joint, selects the joint; Then the user inputs the angle to be changed, changes the selected joint, saves the joint angle data; clicks the send button to send the data to the interface display window, and simultaneously sends it to the slave computer, so that the robot performs the corresponding action. Fig.6 shows that users should Open serial port communication before execute control task. Fig.7 shows that inputting angle and speed to make the robot walk. Figure.8 shows that Joint data is sent to the slave computer successfully.

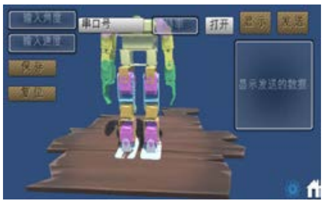


Fig. 6 Open serial port

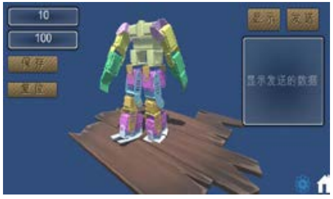
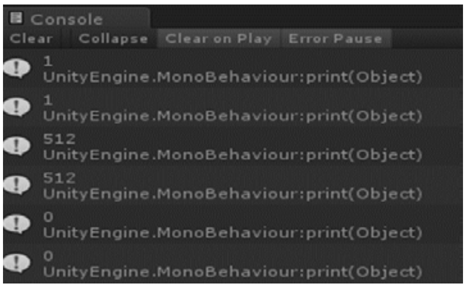


Fig. 7 Input angle and speed



```
1
UnityEngine.MonoBehaviour:print(Object)
1
UnityEngine.MonoBehaviour:print(Object)
512
UnityEngine.MonoBehaviour:print(Object)
512
UnityEngine.MonoBehaviour:print(Object)
0
UnityEngine.MonoBehaviour:print(Object)
0
UnityEngine.MonoBehaviour:print(Object)
```

Fig.8 Data of driving joint is sent

### 4.3 3D visual programming implementation

According to design of the robot 3D visual programming module, the whole operation function can be divided into six parts: picking joints, inputting angles, changing joints, acquiring data, organizing data, observing data, and sending data.

The process of forming each joint is as follows: The first step is to merge the parts models that are always relatively static in motion into one. This part first merges the corresponding models into STL format part models in Solid works, and then introduces the STL files into 3DMAX to get FBX format, finally poured into Unity3D; The second step is to ensure that the rotation axis of each joint in Unity3D coincides with the joint model's own coordinates (this article is the y-axis), which is convenient to drive the joint rotation according to its own coordinates. This part uses 3DMAX to modify the joint. The model's own coordinates can be used; the third step is to ensure the linkage between the relevant joints, such as the elbow joint of the hand will move with the shoulder joint movement, this part is achieved by the parent-child linkage relationship in Unity3D. The formed joint model is shown in Fig.9.

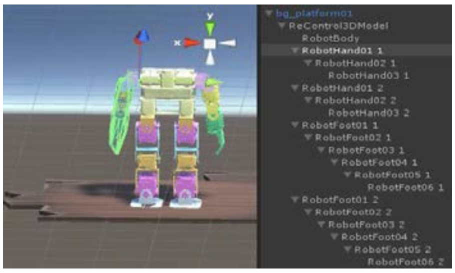


Fig.9 Joint model

Input angle: Create a user input box for the Input Angle directly by directly calling the Simple Input Field in the Prefab Toolbar in NGUI.

Change the joint: Create a "Save" button. After clicking,  get the number of angles entered and drive the selected joint. The main implementation code is as follows:

```
currentTargetAngle = float.Parse (InPutAngle. text);//Get the angle of the uses’ input.
motorDriver.ChangeModelAngle(currentTargetName, currentTargetAngle); // Call joint driver function
```

In order to prevent the user from selecting the corresponding joint and causing the program to be abnormal, the following main code is used:

```
If (targetModel != null) / / if not empty, executing the corresponding instruction.
{
…//Omitting corresponding execution code.
}
Else// If empty, the user is prompt "Click Model!" in the "Input Angle" box.
{
      InPutAngle.text = "Click Model!";
}
```

In order to prevent the user from inputting an wrong angle and causing the program to be abnormal, the same principle is implemented:

```
If (int.TryParse(InPutAngle.text, out a) == true)//Judging whether the users’ input is a numeric string.
```

Create a "Reset" button, when clicked, returns the joint model to its original pose state. The main implementation code is as follows:

```
For (int i = 0; i < 18; i++) //Traversing subset of a set.
{
     If (objStructs[i].AngleMark != 0)//If the joint state
     {                                        angle is not 0.
        motorDriver.ChangeModelAngle(objStructs[i].Obj.name
      }
```

```
 }
```

Get data: In order to get the current driving data of the model, we need to create a variable to detect the rotation angle of each joint. In this paper, we create a constructor to add an attribute variable “AngelMark” of the model. The joint rotation angle data can be directly obtained by code. The main implementation code is as follows:

```
Public ModelStruct(GameObject obj,float angelMark)//Create a  constructor
{
        this.Obj = obj;//Define the model.
      this.Anglemark= angleMark;//Define the rotation angle property.
}
```

Organize the data: Create a “display” button, triggering the send event by clicking the button. That is, the joint angle data is sent to the “display send data” window according to a certain communication protocol. The main code is as follows:

```
sendDate=objStructs[0].AngelMark+"#"+objStructs[1].AngelMark+"#"+...+"#"
+ objStructs[18].AngleMark;
```

Data observation: directly create the display window of "display sent data" by directly calling the Simple Text Box in the Prefab Toolbar in NGUI. The main code that implements the display of sending data is:

```
ShowSendDate.text = sendDate;
```

Data transmission: Create a “send” button. By clicking this button, the sending event is triggered. The joint angle data is sent to the robot lower computer control system through serial communication according to a certain communication protocol. The main code is as follows:

```
dynamixel.write2ByteTxRx();
```

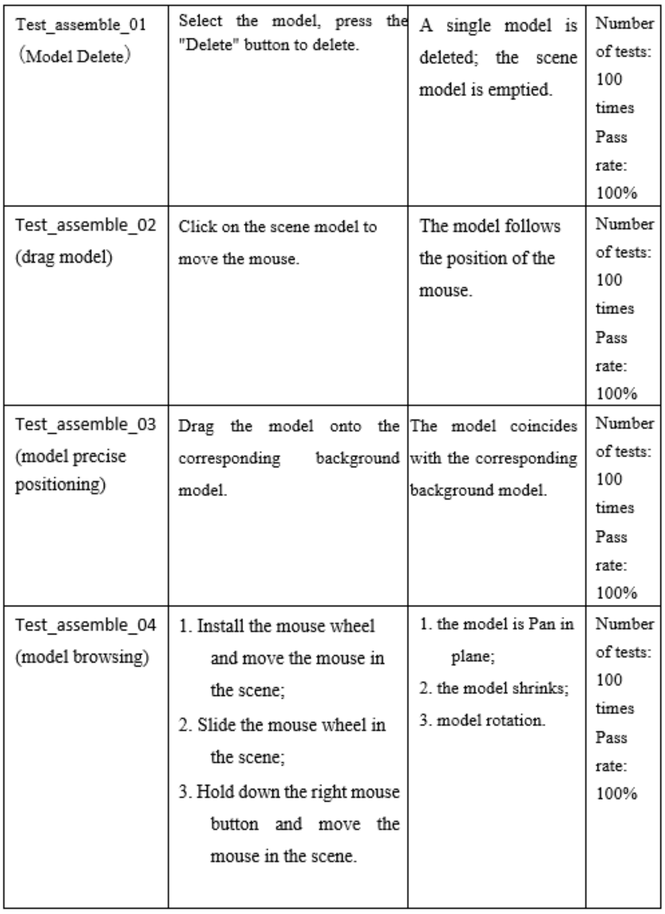
## 5. Software experiment test

There are many kinds of software testing methods, which can be divided into two types according to whether or not to check the internal structure of the software: black box testing and white box testing [7]. Black-box testing refers to the method of testing software functions according to the specifications of program components, so it is also called "function testing". This method focuses on results of input and output. Black box testing method is used in this paper.

The Unity3D engine can develop software across platform. This software test is based on 64-bit Windows 10 platform. The display resolution is 1920*1080.

After testing the 3D model assembly module according to the 3D model assembly module test case table, the results are shown in Table 2.

Table 2 3D model assembly module test case

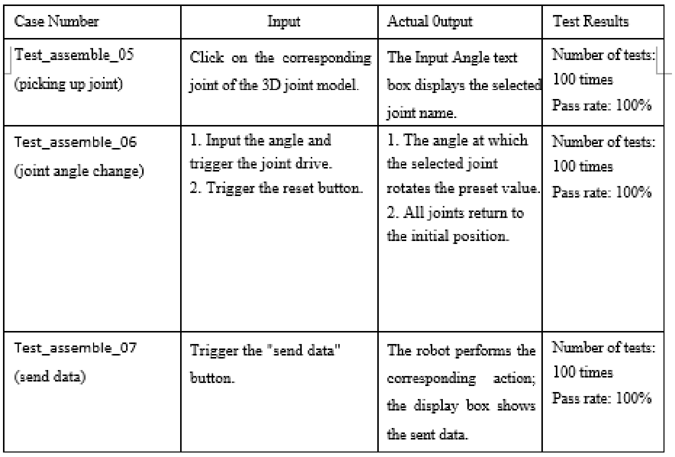


| | | | |
|---|---|---|---|
| Test_assemble_01<br>（Model Delete） | Select the model, press the "Delete" button to delete. | A single model is deleted; the scene model is emptied. | Number of tests: 100 times<br>Pass rate: 100% |
| Test_assemble_02<br>(drag model) | Click on the scene model to move the mouse. | The model follows the position of the mouse. | Number of tests: 100 times<br>Pass rate: 100% |
| Test_assemble_03<br>(model precise positioning) | Drag the model onto the corresponding background model. | The model coincides with the corresponding background model. | Number of tests: 100 times<br>Pass rate: 100% |
| Test_assemble_04<br>(model browsing) | 1. Install the mouse wheel and move the mouse in the scene;<br>2. Slide the mouse wheel in the scene;<br>3. Hold down the right mouse button and move the mouse in the scene. | 1. the model is Pan in plane;<br>2. the model shrinks;<br>3. model rotation. | Number of tests: 100 times<br>Pass rate: 100% |

From Table 2, it can be concluded that the three-dimensional model assembly module meets the needs and can enable students to assemble a complete robot online well.

After testing the 3D visual programming module according to the 3D visual programming module test case, the results are shown in Table 3.

Table 3 3D visual programming test case



| Case Number | Input | Actual 0utput | Test Results |
|---|---|---|---|
| Test_assemble_05<br>(picking up joint) | Click on the corresponding joint of the 3D joint model. | The Input Angle text box displays the selected joint name. | Number of tests: 100 times<br>Pass rate: 100% |
| Test_assemble_06<br>(joint angle change) | 1. Input the angle and trigger the joint drive.<br>2. Trigger the reset button. | 1. The angle at which the selected joint rotates the preset value.<br>2. All joints return to the initial position. | Number of tests: 100 times<br>Pass rate: 100% |
| Test_assemble_07<br>(send data) | Trigger the "send data" button. | The robot performs the corresponding action; the display box shows the sent data. | Number of tests: 100 times<br>Pass rate: 100% |

The 3D visual programming module meets the needs and can be used to control the robot graphically online. Tests results show that the software platform works well.

## 6. Conclusions

In this paper, a complete robot education software platform with excellent interactivity based on Unity3D engine is developed. The development work of platform mainly includes analysis of functional requirement, design of software platform architecture, design of communication module, design of 3D part assembly module of the robot, design of 3D visual programming module for robot. The software platform realizes assembly and control of the robot. Experiment tests suggest that the software platform works well. It helps students to understand and master robot knowledge fully, even to stimulate great interests in robotics research. The robot education software has good interactivity and achieves the desired goals.

## References

[1] Zhong Baichang. The Core Theory of Robot Education in Primary and Secondary Schools: New Classification of Robot Teaching Model [J]. Research on Audio-visual Education, 2016 (12): 87-92. (In Chinese)

[2] Wang Siyu. Research on Graphical Programming System of Educational Robots_Wang Siyu[D], 2017. (In Chinese)

[3] Shao Gang. Optimization and improvement of MVC controller based on Spring framework [D]. Shandong University, 2011. (In Chinese)

[4] Yang D H . Apparatus and method for executing application [J]. 2014.

[5] Wang Xinwei. Design of virtual robot assembly system based on Kinect [D]: Harbin Institute of Technology, 2015. (In Chinese)

[6] Yun, Sang-Seok, Seoul, Korea Kim, Munsang, International Journal of Social Robotics, v5, n3, pp 335-343, August 2013.

[7] Zhang Xinhua, Ho Yong. Overview of software testing methods [J]. Science and technology horizons, 2012 (4): 35-37. (In Chinese)