Construction of industrial Robot Virtual Assemble Training System with Unity3D

Jun AN[1], Zhuo-Tao LIU[1], and Jin-Song FAN[2,*]

[1]School of Mechatronics Engineering, Foshan University, 528000, Foshan, China

[2]School of Industrial Design and Ceramic Art, Foshan University, 528000, Foshan, China

[1]annyfan@yeah.net, [1]1789142317@qq.com, [2]jackfan68@hotmail.com

*Corresponding author: Jin-Song FAN

Keywords: Virtual assemble, Industrial robot, Virtual reality, Unity3D.

Abstract. By deeply analyzed virtual reality technology, industrial robot virtual assemble system and related software and hardware platform, the overall structure and fundamental configuration of virtual assemble training system applying to general industrial robot was put forward. The virtual scene and the UI of the system was developed with Unity3D, and the Leap Motion was used to indentify hand gesture and carry out human-machine interactions. The solution for assemble models bundling and uploading to database, and dynamically loading from database was given. Finally, a system prototype of industrial robot virtual assembles training system based on Unity3D and Leap Motion was developed, and fundamental feature of the system was achieved.

Introduction

Virtual Reality (VR) is a computer-generated virtual environment that can give stimulations human's multiple senses (sight, hearing, touch, etc.), with which people can interact in natural ways and get the immersive feeling of exposure to corresponding real environment. Its feature include: Immersion, Interaction and Imagination[1]. Intelligent Manufacturing is recognized as an important direction of modern manufacturing by Germany's "Industry 4.0", the United States "re-industrialization" and China's "Made in China 2025". As an important part of Intelligent Manufacturing, the application of Virtual Reality technology in manufacturing is getting more and more concern and attention. Virtual assembly technology is a combination of product assembly and virtual reality technology, in which special software is used to create a virtual assembly environment consistent with the actual assembly environment. By using appropriate equipment, the Assembly personnel enter virtual assembly environment and carry out simulation assembly through fit the shape and positional relationship between the parts of the product according to pre-defined rules. At the same time, the operation process and date were record to determine its assembly or disassembly sequence and path, relevant assessment reports and video were generated for future analysis. [2-7]

Software and Hardware Platform of the System

Configuration of System Hardware

A general virtual reality system are made up of professional graphics processing computer, application software systems, input and output devices and database(Figure 1).

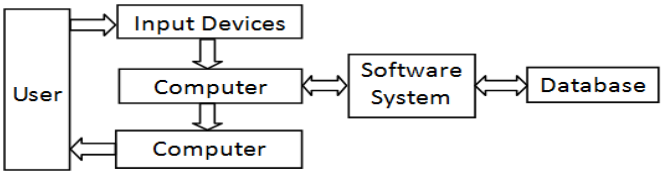


Figure 1. Configuration of Virtual Reality

Depending on different degree of "immersion" and degree of interaction of virtual reality technology, the existing virtual reality systems can be classified as four typical types: Desktop virtual reality system, Immersive virtual reality system (Helmet, Cave, etc.), augmented reality systems and Distributed virtual reality system. According to the situation of laboratory conditions and applications, we choose to use the desktop virtual reality system.

In desktop virtual reality system, common virtual reality interactive equipment include: Location Tracker, Data grove, 3DMouse, Gesture recognition device (LeapMotion), Somatosensory equipment(Kinect, Wii Remote, PlayStation Move, etc.), eye tracker and Force feedback devices. Besides, mouse and keyboard are also common equipments. In order to meet the natural way of interaction with real world, virtual reality interactive equipment were adopted in the system, minimizing the use of traditional mouse and keyboard operation. Considering the price and accuracy, gesture recognition and somatosensory equipment which were widely used in many virtual systems and computer gaming were adopted in our system.

For virtual assembly, the operations of the hand are mostly involved. LeapMotion, a very small gesture recognition device, can track every part of human hand, including fingers and palm, with high accuracy (1/100mm) and low latency. It also feedback hand position and velocity with low cost, so Leap Motion was chosen as the primary virtual interaction devices.

Selection of Software

By comparing the existing virtual reality software and the game engine, Unity 3D was chosen as system developing software. Unity 3D is a professional cross-platform game and virtual reality engine, released by Unity Technologies. Its biggest feature is support multi-platform. Apart from the mainstream platform of Windows, iOS, Android, etc., it also supports PSVITA, XBOXONE, Wii and other platforms, which allow one development to be deployed to multiple platforms. It has a complete set of development toolkit, and the SDK package can be used directly in Unity3D, which can greatly reduce the workload of program development and shorten the product development cycle.

Systems Structure and Framework

Industrial Robot Assembly Sequence

In general, industrial robot assembly can be divided into two parts: Machinery section and auxiliary section. Machinery section consists of three parts: Main body (Column), Arm and hand. After all parts are ready, sub-assembly of base, body, arm, forearm and wrist should be complete first, then the assembly from base to arm should be carried out. Finally, drive mechanism, balancing cylinder and connected wires are installed, and the assembly of an industrial robot is completed[8]. The assembly sequence is shown as Figure 2.

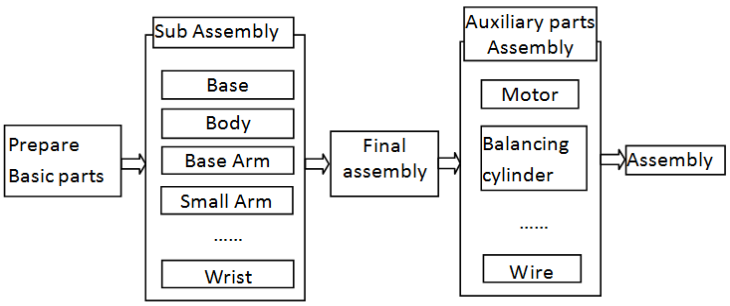


Figure 2. Manufacturing Process of Industrial Robots

Basic Components of the System

After a start screen(Figure 3), introduction page and instructions for use, the main assembly interface is entered. According to the above assembly sequence of industrial robot, the main assembly can be

divided into three modules: Sub-components assembly (including base, body, arm, forearm and wrist), main assembly, and auxiliary assembly (installing of motor, balance cylinder). Apart from these, the assembly animations of correct assembly sequence were produced to be used as tutorials and demonstration. The user can choose to do parts assembly, main assembly or check corresponding animations.

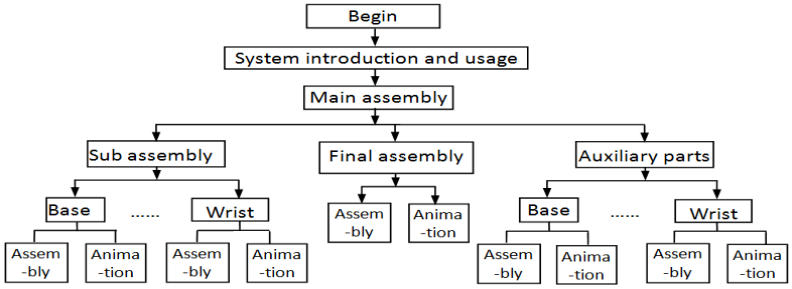


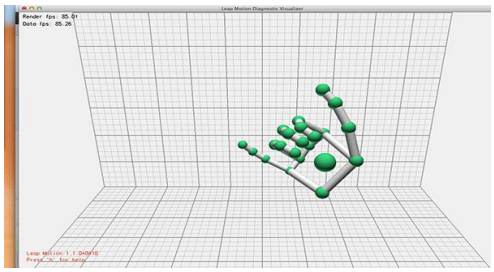
Figure 3. Basic structure of the system Design of Interaction programming

Design of Interaction Programming

Interactive Principle

There are two cameras in Leap Motion(Figure 4), through which several images can be captured in a certain period of time. These images are called “frames”, the information in each frame can be analyzed and manipulated.

When the operator stretch up a hand, Leap Motion can determine what hand gesture is according to certain algorithms: All 5 fingers are open or some them are still bent. It cans "discrimination" actual position of bones of our hands, and reconstruct hands with the same gesture in the virtual scene. At the same time, it can feed back the position and rotation, even velocity information of each hand bone in each frame to computer, and various functions can be realized by programming with these information. In figure 4(b), we can see that thirty control points are defined by Leap Motion.



(a)Internal structure of Leap Motion (b) The mapping image in computer of the hand that Leap Motion “see”

Figure 4. Leap Motion identification mode

Principle of Object Grasping

To realize object grasping, the following two conditions need to be met (Figure 5):

(1) Calculate the distance between thumb and nodes of any two bones of each finger, and find the minimum value. When the minimum distance is less than a given distance, system is told that the current "hand" in the Grab Status.

(2) Since Leap Motion can capture the position of each finger in each frame, the spatial coordinate index finger and thumb can be obtained and utilized. Set the middle point of line between these two points as the center point, and generate a sphere at a given radius distance (The ball is a virtual invisible, hereinafter referred to as "detection ball"). Which a collision object is detected within the ball, then collision object is marked as grabbed object. If several objects are inside the detected ball, the closest object will be marked.

When the above two conditions are satisfied, the pivot point of grabbed object will followed the middle point between the index finger and thumb, thus the grabbing of a object is achieved.

Start, Calculate all the distance between thumb and nodes of each finger, find the minimum value, minimum distance is less than a given distance, a collision object is within the detection ball, a object is crawled.

The release action is triggered when the lengths of the farthest part of thumb and the distance between the tips of index finger and thumb are larger than given threshold value. The coordinates of the object is the coordinates at the moment of triggering, and will not be affected by hand any more.

Based on the above principle, the functions of move, rotate and crawl are realized with C# programming. In this system, various gestures of grabbing object is real life are simplified and only a case is considered: index finger and thumb of right hand can grab object without consideration of the participation of left hand.

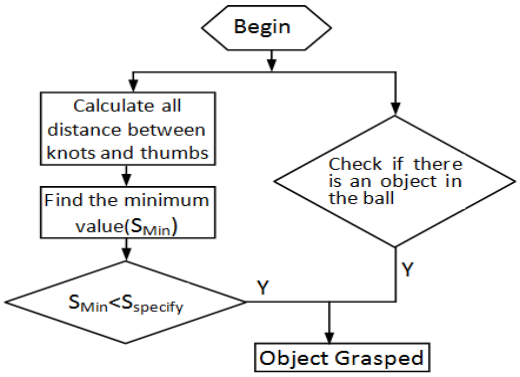


Figure 5. Program flow chart of grasping object

System Realizations in Unity3D

Model Importing and Bundling

The model of industrial robot we used is a SolidWorks' model. Since the 3D model from engineering software such as SolidWorks, Pro/E, UG, cannot be imported into Unity3D directly, they must be exported into 3DS MAX or MAYA first, preprocessing the shape, color and texture, and then converted to .fbx files before imported into Unity3D.

These imported models are then handled with their material and lighting. AssetBundle is used to pack these models up and the packed files are uploaded to database so that they can be downloaded later for dynamic loading as well as distributed publishing.

Definition of Assembly

How to define the assembly is the core issue of the whole system design. In virtual reality system, collision detection is an important part of information exchange of physical object. The physical system in Unity 3D provide two ways of detecting physical interaction, one is using collision object, the other is using trigger. Here, we adopt the way of trigger to handle the issue whether assembly is in place and assembly sequence is right.

A "trigger" should be place on each part to be assembled. When other part enter the range of this trigger, it will be determined whether it is the next assembly parts, and color will be used to prompt if it is the correct parts. If the selected part is the right part, the color of the part will be blue (Figure 6), and the system will connect to database and download the model and relevant information of that part. The assembly is done when the operator move it to the correct location. If the operator chooses the wrong part, the color of the part will be red, and the message of "assembly error" will be display on the interface (Figure 7).

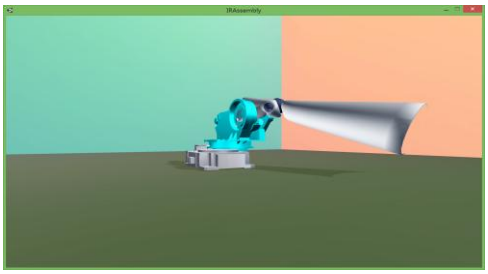


Figure 6. Assembly correct

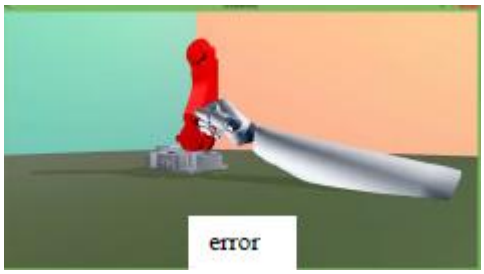


Figure 7. Assembly wrong

System Realization

Based on the above ideas and specifications of Leap Motion and Unity3D programming, following basic classes were designed, which are all programmed with C #.

1) Interaction Class: These classes, inherited from ButtonReaction, are designed mainly to complete the definition some gestures and interactions of LeapMotion, and control the corresponding interface operation respectively.

2) Assembly Class: These classes, inherited from assembeTrigger, are mainly deal with issue related to triggers and collisions.

3) Database processing Class: They deal with the operation related to database processing, upload bundling and download data.

4) Other Class: Carrying out the operation of events handling, animation, etc.

The interaction and assembly part will be described below.

Realization of LeapMotion Interaction

The basic interaction mode of the system is to control cursor movement with Leap Motion, the event of a button is triggered when a cursor has stayed on the button for a certain period of time. There are many buttons of the UI in the system, but some actions of these button are the same (the progress bar will animated when the cursor touch button). In order to avoid code duplication, the object-oriented inheritance is used. the class of all buttons is inherited from ButtonReaction class, which defines all the same part of the code of buttons and defines a virtual function to facilitate the subclass overrides. The basic codes of ButtonReaction class are as follow:

```
void OnTriggerStay2D (Collider2D other)
{
  if (other.name.Equals ("Cursor"))
  {
      gameObject.GetComponent<Image>().color = highlightedColor;
      stayedTime= Time.time - enterTime;
      other.gameObject.GetComponentInChildren <Slider>().value = stayedTime;
      if (stayedTime == 1.0f)
      {
            timeIsUp();
            stayedTime = 0;
           other.gameObject.GetComponentInChildren <Slider>().value = 0;
           gameObject.GetComponent<Image>().color= Color.white;
      }
  }
}
protected virtual void timeIsUp()
{

}
```

Realization of Virtual Assembly

The base class of assembly is AssembleTrigger, which inherits from the MonoBehaviour class of Unity3D. All objects that need to assemble must load the class. The DetailPanelControl class is utilized by event mechanism to monitor. When user needs to load parts into the scene, the respected button for different parts on the panel were selected, and Detail Panel Control class receives this message and corresponding treatments are done. The models in database are then loaded in assembly system, and the relevant message is shown on the panel at the same time. The main code of Assemble Trigger class are as follow:

```
protected void OnTriggerEnter(Collider other)
{
    if (other.CompareTag (assembleTag))
   {
         Color currentAssembledComponentColor = currentAssembledComponent.GetComponent
         <MeshRenderer> ().material.color;
     }
     if(!getTheColor && currentAssembledComponentColor!= assembleCorrectColor &&
        currentAssembledComponentColor!= assembleErrorColor)
     {
         orgColor = currentAssembledComponentColor;
         getTheColor = true;
     }
     ComponentIsNerby (currentAssembledComponent, gameObject,
      specialComponentCorrectPosition);
    }
}
```

Summary

With the principles and methods discussed above, a prototype system of virtual assembly training of industrial robot are developed, which interface and assembly process are shown in Figure 8 and Figure 9. The system generally realize the interaction of grabbing, moving, rotating, and menu selection with Leap Motion, and can upload and download bundling model, its related information, parts, sub-assembly and demonstration animation to and from database. This system has basic function of a virtual assembly training system, and its design of interface and the assembly authenticity need to be further improved.

Figure 8. System interface

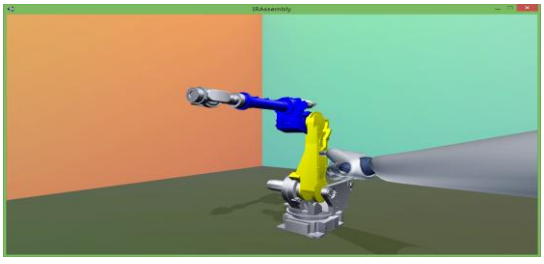


Figure 9. Example of assembly

Acknowledgment

This work is supported by The High-level Expert Project of University from Department of Education of Guangdong, China (No. Yue Cai Jiao [2013]246), and the 2016 Foshan Science and

Technology Innovation Platform Project (Industrial Design Innovation Public Service Platform, No. FoCaiGong[2016]76)

Reference

1. Huan Hai. Virtual Reality. Beijing University of Posts and Telecommunications Press, Beijing, 2014 (In Chinese).

2. P. Horejsi. Procedia Engineering, 100 (2015)699-706.

3. A. Syberfeldt, O. Danielsson, M. Holm, L. h. Wang. Procedia Manufacturing, 1(2015) 98-109.

4. S. Wiedenmaier, O. Oehme, L. Schmidt, H. Luczak. International Journal of Human-Computer Interaction, 16(3) (2003)497-514.

5. G. Lawson, D. Salanitri, B. Waterfield. Applied Ergonomics, 53 (2016) 323–330.

6. P. Fillatreau, J.-Y. Fourquet, R. Le Bolloc' h, S. Cailhol, A. Datas, B. Puel. Computers in Industry, 64 (2013) 1253–1262.

7. S. Jayaram, U. Jayaram, Y. Kim, C. De Chenne, K. Lyons, C. Palmer, T. Mitsui. Virtual Reality, 11 (4) (2007) 217-228.

8. LU You-Bin, ZHAO Cong-Hu, ZHANG Lin, LIU Jia-Lei. Development & Innovation of Machinery & Electrical Products, 24(5)(2011)19-20 (In Chinese).