Proceedings of 2016 IEEE
International Conference on Mechatronics and Automation
August 7 - 10, Harbin, China

The Convenient Assembly Method for the Virtual Robot Kit

Haitao Gao, Youxiong Xu, Yali Han, Songqing Zhu and Yinglu Zhou

School of Mechanical Engineering
Nanjing Institute of Technology
Nanjing, Jiangsu Province, China

{ ght & zdhxxyx } @njit.edu.cn

***Abstract* - A convenient assembly method for the virtual modular robot is put forward, proposed assembly method does not need the designers to specify explicitly constraint relation of components but rather establish them automatically according to the defined assembly information and the matching algorithm of connectors. The key technology of the assembly method is studied, a hierarchical representation of robot component is proposed to express necessary assembly and simulation information, the 3D picking method of model is realized by introducing a picking ray which takes the viewpoint as a starting point and goes through the clicked point to compensate lack of depth information, as well as the fast matching algorithm based on square area comparison is set up to quickly determine assembly point. Finally, this assembly method is realized with the key technology solved and an application example is given to show its effect.**

Index Terms - Robot kit; 3D simulation; Assembly method.

I. Introduction

Robot simulation is always useful in teaching robotics. On one hand, it enables the evaluation of different robot design. On the other hand, it offers the possibility of directly debugging and testing robot system. However, existing robot kit that is special robots like building block toys is still physical robot provided by manufacturers and lacks appropriate simulation platform. Therefore, it is important to research a simulator for modular robot kit.

In the past, several robot simulators have been developed with different attention. Stage[1] is the simulator that simulates a two-dimensional world where multiple robots can act concurrently. But, Stage is an intrinsically two-dimensional tool. Gazebo[2] is a general 3D multi-robot simulator with graphical interface and dynamics simulation, the simulated environment is described in XML files using several predefined elements and robots. Webots[3,4] is a commercial robotic simulator developed by the Cyberbotics Ltd, which has an ODE-based physics simulation and offers support also for fast prototyping and simulation of mobile robots. Microsoft Robotics Developer Studio (MRDS) [5,6] is a Windows-based environment for robot control and simulation developed by Microsoft in association with the community. It includes a visual programming tool, Microsoft Visual Programming Language for creating and debugging robot applications. The other robot simulator and package, such as SimRobot[7], USARSim[8], ARGOS[9], etc., have also been developed and available.

Several robot simulators mentioned above are different in the possibility of creating and integrating own robot models and virtual environments. Some of the simulators such as stage are restricted to the two dimensional environment or high cost of commercial software, and some of the simulators mainly focus more on complex kinematics and dynamic analysis, robot planning, but focuses less on assembly process and simplicity of operation. Those robot simulators cannot meet the requirements of the robot kit because the user of robot kit is the primary user and the robot training should involve the complete design process of the robot. For this reason, a three-dimensional robot simulator integrating robot assembly, program design and animation simulation is put forward. Purpose of this paper is to establish a fast assembly method for robot kit in order to build above simulator..

II. Proposition of Assembly Methods

In the modular robots, the building block principle is used. As is shown in Fig.1, components of modular robot such as mechanical components, sensors, controller, etc. are designed in the modular structure like building blocks. Multiple male and female connectors are uniformly distributed on the surface of components with special size and structure. Designers can freely and conveniently set up different robot configurations by connecting male and female connectors.

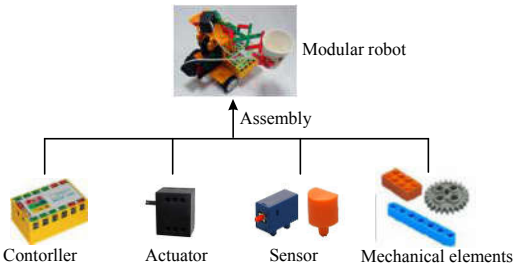


Fig.1 Components of Modular Robot Kit

How to simulate assembly rules of modular robot in the computer is an important and difficult task due to the fact that intelligence and interaction level of computer is still far lower than that of human. According to popular human-computer interaction and assembly rules of physical robot, we develop a convenient assembly method for modular robot kit in the 3D environment. The main process of proposed method is shown in Fig.2, which includes selection of assembly reference face, adjustment of component position and direction, searching and

matching of assembly point and determination of connection type. Using this assembly method, constraint relations between components do not need to be interactively and explicitly specified, rather, automatically established.

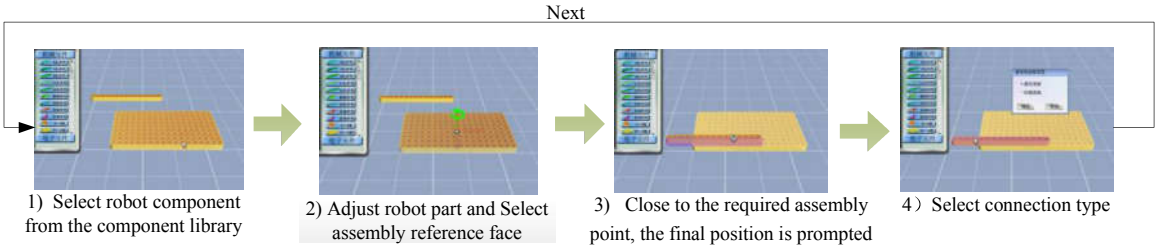


Fig. 2 Assembly Process of Robot Component

The key technical problems of this assembly method such as the representation of robot component, the picking of component and assembly surface and the automatic searching and matching algorithm of possible assembly point will be solved in detail as follows.

III. Representation of Robot Component

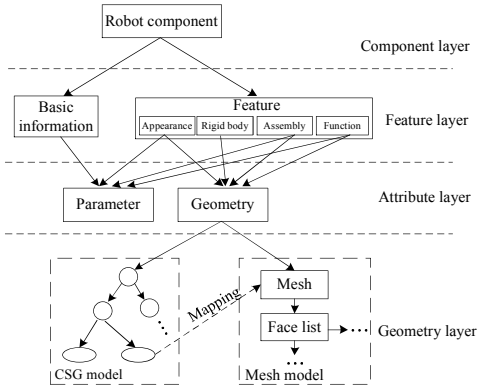


Fig.3 Hierarchical Representation of Robot Component

In the integrated robot simulator, the 3D model of robot has not only a geometric shape, but also other required information such as physical properties, bounding box, assembly knowledge, and so on. In order to make a unified representation, this information is packaged as feature and expressed in the hierarchical structure [10]. The 3D model of component is mainly divided into component layer, feature layer which includes appearance feature, rigid body feature and assembly feature, function feature etc., attribute layer and geometry layer according to the different levels of abstraction (as shown in Fig.3), different layer are related through data or mapping. At the lowest geometry layer, the hybrid geometric model based on constructive solid geometry (CSG) model and mesh model is used for the need of assembly and collision detection. In the feature layer, appearance feature defines appearance information of the model, the rigid body feature which includes some bounding box such as Axis Aligned Bounding Boxes, Sphere, Oriented Bounding Box, etc. is used to replace the complex geometry and improve the efficiency of intersecting test, the assembly feature defines the information of assembly process such as matching parameters, assembly action, assembly sequence and the functional feature.

We also establish the special component designer, so that related attribute and feature of components can be defined in designer. XML language is used to express 3D model since XML documents is supported by a variety of editors and its cascaded structure is also quite suitable to reflect the structure of robot component[11]. A XML schema called ER-XML is defined to organize the structure of educational robot component and robot[10].

Ⅳ. 3D Picking Based on Ray Detection

The mouse picking is the most popular picking method in the 2D environment, but it is still difficult to use in the 3D environment due to the lack of depth information. In order to overcome this problem, ray detection algorithm is introduced. A picking ray which takes the viewpoint as a starting point and goes through the clicked point is constructed and the 3D models which intersect with the ray can be obtained by ray detection algorithm. According to rules, the top part model is selected as picked object. Since ray detection algorithm can only be used when the picking ray and model is in the same space, the picking ray needs a series of transformations to get into the same space with picked model.

A. Space Transformation of Clicked Point

According to the theory of computer graphics, the clicked point is firstly transformed from the screen space to the projection space. Usually, origin of screen space does not coincide with origin of the projection plane and transformation needs to be done. As shown in the Fig.4(a), let P_s (X_s,Y_s) denote clicked position in the screen space, W_s and H_s denote the width and height of the program window respectively. the coordinate of the clicked point in the projection plane can be obtained according to the coordinate relation:

$$X_p=(X_s-W_s/2)/W_s\times 2, \qquad Y_p=(Y_s-H_s/2)/H_s\times 2 \tag{1}$$

Projection space a normalized space containing both far-clipping surface and near-clipping surface in the range of -1.0-1.0 and is shown in the Fig.4 (b). The far projective point P_p $(X_p, Y_p, 1)$ of the clicked point on the far-cutting surface is selected as representation of the clicked point in the 3D environment, and then the clicked point can be transformed from the projection space to the view frustum space through left multiplying the inverse of projection matrix:

$$P_v=(M_p)^{-1}\times P_p \tag{2}$$

In formula, $M_p = \begin{bmatrix} \frac{2n}{l} & 0 & \frac{r-1}{r+1} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & \frac{-(f+n)}{f-n} & \frac{-2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix}$, ($l$, b, $-n$) and (r, t, $-n$) are the coordinate values of the upper left and lower right corners on the near-clipping surface and the far-clipping surface, n represent the distance between viewpoint and the near-clipping surface and f represents the distance between viewpoint and the far-cutting surface.

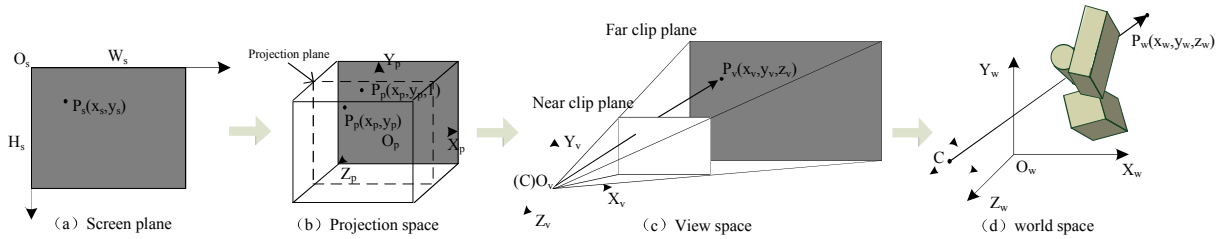


Fig.4 Space Transformation of Clicked Point

Secondly, a ray which starts from the viewpoint and goes through P_v in the view frustum is constructed (in Fig.4(c)):

$$R_v(t)=C+Dt \tag{3}$$

Where C is the viewpoint, D is the direction vector of the ray and t is the distance vector.

Next, the constructed ray is further transformed to the world space, and the transformation can be realized through the ray equation left multiplying the inverse matrix of observing transformation matrix (in Fig.4 (d)). The transformation matrix is as follows:

$$R_W(t)=(M_v)^{-1}C+(M_{v1})^{-1}Dt \tag{4}$$

In formula,

$$M_v = \begin{bmatrix} u_x & v_x & n_n & 0 \\ u_y & v_y & n_y & 0 \\ u_z & v_z & n_z & 0 \\ -\boldsymbol{u}\cdot\boldsymbol{c} & -\boldsymbol{v}\cdot\boldsymbol{c} & -\boldsymbol{n}\cdot\boldsymbol{c} & 1 \end{bmatrix}, M_{v1} = \begin{bmatrix} u_x & v_x & n_n & 0 \\ u_y & v_y & n_y & 0 \\ u_z & v_z & n_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Where $\boldsymbol{u}(u_x,u_y,u_z)$, $\boldsymbol{v}(v_x,v_y,v_z)$, $\boldsymbol{n}(n_x,n_y,n_z)$ are the direction vectors in the world coordinate system respectively.

The ray expressed by equation (4) is the final picking ray, which will change with the movement of the mouse. The object is picked up by this picking ray and the detection algorithm. In order to improve the efficiency of intersecting test, detection algorithm is divided into two steps. The first rough ray detection with bounding box of the model helps to quickly exclude some non-intersect models, and then further detailed test with the model is done according to the results.

B. Pick-up of Geometric Elements and Assembly Body

The geometric elements such as geometric face and assembly body are often picked up in the assembly process of modular robot. Since the fact that the 3D model of robot component is mainly established by 3dmax and expressed through triangular mesh, the geometric elements cannot be directly chosen. But according to the mapping of triangle patch and geometric elements in the definition of robot component model, an indirect picking method of geometric elements can be realized; as is shown in Fig.5, this method is to detect the intersection of the picking ray and triangles patches firstly and then pick up geometric elements by mapping of triangle patch and geometric elements. Similarly, the pick of assembly can also be indirectly achieved through the constraint relation of the picked part.

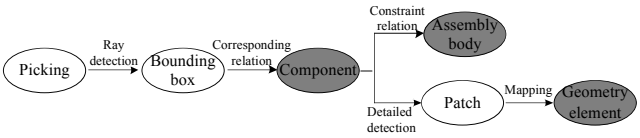


Fig. 5 Picking Process of Geometry Element and Assembly Body

In the indirect picking method of geometric elements, the usually-used algorithm to detect the intersection of the ray and triangle patches is based on the computation of the weight coordinates of the triangle's vertices. Assuming the three vertices of the triangle are: P_1, P_2, P_3, the triangle and all of its internal points can be expressed as a weighted average of the vertices based on the principle of analytic geometry, namely:

$$P = e_1P_1 + e_2P_2 + e_3P_3 \tag{5}$$

Where e_1, e_2 and e_3 are the weights of three vertices respectively, and meet the conditions: e_1>=0; e_2>=0, e_3>=0, and $e_1+e_2+e_3$=1. To gain the intersection point of the ray and the triangle just need to solve the following equation:

$$C + Dt = (1-e_2-e_3)P_1 + e_2P_2 + e_3P_3 \tag{6}$$

t, e_2, e_3 are extracted as unknown, after transposition and collation, the following linear equation (7) is obtained :

$$\begin{bmatrix} -D & P_2-P_1 & P_3-P_1 \end{bmatrix} \begin{bmatrix} t \\ e_2 \\ e_3 \end{bmatrix} = C - P_1 \tag{7}$$

Cramer's rule is used to solve the equation (7) and the result is expressed by the form of vector mixed product.

$$\begin{bmatrix} t \\ e_2 \\ e_3 \end{bmatrix} = \frac{1}{[D\times V_2 \cdot V_1]} \begin{bmatrix} T\times V_1 \cdot V_2 \\ D\times V_2 \cdot T \\ T\times V_1 \cdot D \end{bmatrix} \tag{8}$$

Where $V_1 = P_2 - P_1$, $V_2 = P_3 - P_1$, $T = C - P_1$. The expression

(8) is achieved by program and then the intersection of the ray and triangle patches can be quickly judged.

Ⅴ. Fast Search of Possible Assembly Point

In the automatically assembly of modular robot, the designers do not need to specify interactively and explicitly the constraint relations of components, but rather, the system automatically recognizes the assembly intention and then matches the assembly point. The designer always expects the assembly units to be assembled get close to each other. Naturally, judgment of relative distance between the assembly points can be used as right method of recognizing the designer intention and determining possible assembly point. But this method still has the shortage of low judgment speed in the practical application due to a large number of assembly point and time-consuming distance calculation. In order to improve the judgment speed, a fast algorithm based on coordinate comparison is given.

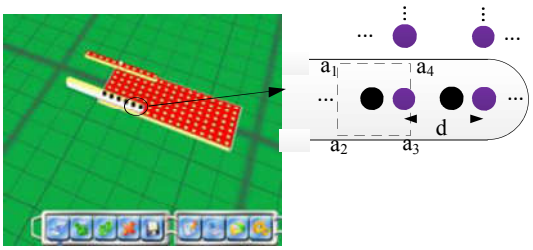


Fig. 6 Fast Search of Possible Assembly Point

Algorithm 1: Fast search of assembly unit

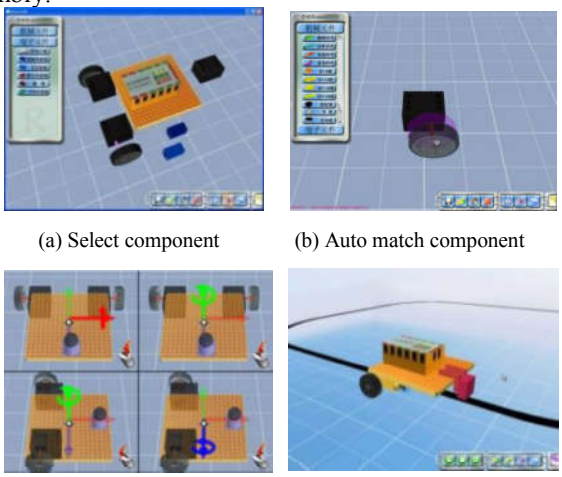
Input: two assembled robot component
Output: assembled Connector List, search flags
1：**initialize parameters:** component, component position, connector list, reference plane, Isfind.
2: **For** i=1 **to** ComponentA. ConnectorList.Count **do**
3: Create_JudgeSquare (Connector[i].Position, Connector_ Distance, ReferencePlane),
3: **For** j=1 **to** ComponentB. ConnectorList.Count **do**
4: **if** (Connector [j]. Position<= Connector [i]. JudgeSquare)
5: AssembledConnectorPairlist.add(ComponentA. ConnectorList[i], ComponentB.ConnectorList[j]);
6: Isfind=true;
7: **else**
8: Isfind=false;
9: **End if**
10: **End for**
11: **End for**

Fig. 7 Pseudo-code of the Fast Search Algorithm

Just as the example shown in Fig.6, there are two assembly components: PlateA and BeamB. PlateA is the assembly base and its surface is selected as a datum of this assembly, accordingly, the corresponding parallel faces of the rest components in the scene become the faces to be assembled. When BeamB is dragged near to PlateA, the chosen assembly faces of two components automatically coincide, and the current component only move inside the datum face of the assembly base. A square area for intention recognition which aligns the world coordinate is set up at the center of connectors and its length is equal to distance between two connectors. As a result, the intention of the designers can be determined by judging whether the connector near the component falls into this square area, which can be realized by comparing coordinates value of connector and square vertices without calculating the distance between the connectors. The specific pseudo-code of the algorithm is given and shown in the Fig.7.

Ⅵ. Application

In the end, we also put the fast assembly method into application in the robotic integrated simulation platform. Take the self-tracing car as an example. The main assembly interface is shown in Fig.8 below and operation of select components, adjustment of components and automatic search and match for assembly point is displayed. Practical application shows that this assembly method is rapid and simple to operate and the algorithm of capturing the assembly intention can capture the intention of the assembler quickly, which can satisfy the requirement of the modular robot assembly.



(a) Select component (b) Auto match component

(c) Adjust component (d) Simulaltion

Fig. 8 Application of Assembly Method

Ⅶ. CONCLUSION

In the paper, assembly feature of robot kit has been analyzed and then an assembly method which simulates assembly process of modular robot has been developed. This method makes virtual robot's assembly more convenient and fast, as a result, designer pays more attention to the design of robot, rather than cumbersome assembly operation. At last, we realize this assembly method with the key technology solved. The practical application shows that proposed method has better effect.

In order to build proposed 3D robot simulator, there are many key technologies to be solved. Our future work will include continuing to improve assembly modeling method and study animation simulation and graphic programming.

Acknowledgment

This work was partially supported by the Natural Science Foundation of Jiangsu province (BK20130743), the Natural

Science Foundation of Higher Education of Jiangsu province (14KJD460001, 15KJA460007), the Innovation Foundation of NJIT (CKJB201202, CKJA201501), the Science and Technology Support Program of Jiangsu province (BE2014142).

REFERENCES

[1] R.Vaughan, "Massively multi-robot simulation in stage," *Swarm Intelligence*, vol.2, pp.189-208, 2008.

[2] N. Koenig , A. Howard, "Design and use paradigms for gazebo-an open-source multi-robot simulator," *2004 IEEE RSJ International Conference on Intelligent Robots and Systems*, vol. 3, pp. 2149-2154, 2004.

[3] O. Michel, "Webots: Professional Mobile Robot Simulation," *International Journal of Advanced Robotic Systems*, vol. 1, pp. 39-42, 2004.

[4] J. Craighead, R, Murphy, J. Burke, et al. "A Survey of Commercial & Open Source Unmanned Vehicle Simulators," *International Conference on Robotics and Automation*, vol. 1, pp. 852-857, 2007.

[5] J. Jackson, "Microsoft robotics studio: A technical introduction," *IEEE robotics & automation magazine*, vol. 14, no. 4 pp. 82-87, 2007.

[6] J. Cepeda, L. Chaimowicz, R. Soto, "Exploring Microsoft Robotics Studio as a Mechanism for Service-Oriented Robotics," *American Robotics Symposium and Intelligent Robotics Meeting*, vol.1, pp. 7-12,2010.

[7] T. Laue, S. Kai, T. Rofer, "SimRobot–A General Physical Robot Simulator and its Application in RoboCup," *Robot Soccer World Cup IX*, vol.1, pp. 173-183, 2005.

[8] S. Carpin, M. Lewis, J. Wang, et al., "USARSim: a robot simulator for research and education," I*EEE International Conference on Robotics & Automation*, vol.1, pp. 1400-1405, 2007.

[9] L. Gambardella, M. Dorigo, "ARGOS: A modular, multi-engine simulator for heterogeneous swarm robotics," *IEEE International Conference on Intelligent Robots and Systems*, vol.1, pp. 5027-5034, 2011.

[10] G. Haitao, *Key Technologies for 3D Simulation Platform of Educational Robots*, South-east University Press, 2010, pp. 23-29.

[11] S. Kadrya, J.Clavera, "XML Parser GUI using .NET Technology," *Ieri Procedia*, vol.2, pp. 554-560, 2012.