Int J Adv Manuf Technol (2013) 64:1765–1779
DOI 10.1007/s00170-012-4140-1

ORIGINAL ARTICLE

VR-based wheeled mobile robot in application of remote real-time assembly

Janus S. Liang · Kuo-Ming Chao · Paul Ivey

Received: 26 November 2011 / Accepted: 4 April 2012 / Published online: 24 May 2012

Abstract The objective of this paper is to develop an appropriate methodology with open architecture for real-time monitoring and remote control of networked mobile robot. An integrated product data model and the related configuration management methods are generated. Within the context, this paper presents a new enabling technology to bring traditional robotic tools on-line with combined monitoring and control capability. Under a hybrid architecture of Web browser/server and client/server, an application system is presented to manipulate the wheeled mobile robot data and to carry out a variety of assembly functions. Issues such as architecture design, methodology development, and prototype implementation are addressed through a component assembly case study. Finally, a CORBA stands-based integration framework is proposed to achieve interoperability among multiple data and application objects over the Internet.

Keywords Wheeled mobile robot · Virtual reality · Remote control · Component assembly · Configuration management

1 Introduction

Due to the pressure of international competition and market globalization in the twenty-first century, there continues to be strong driving forces in industry to compete effectively by reducing time-to-market and cost while assuring high-quality product and service. Meanwhile, manufacturing companies have been plagued by depletion of natural resources, diversified customer requirements, reduced product lifecycles, and increased production complexity. These challenges call for the manufacturing companies that want to succeed in the international market to respond to market agilely and shorten the lead-time. Quick response to business opportunity has been considered as one of the important factors to ensure company competitiveness [1]. To get collective competitive advantages in the changing markets, manufacturing companies are compelled to form various alliances, collaboratively providing customer-demanded products and services. However, a highly efficient infrastructure that can integrate the pieces of automated equipment together and link them directly to the virtual reality (VR) assembly is still lacking. A new enabling technology is therefore urgently required to bring traditional robotic tools on-line with combined monitoring and control capability. Targeting this problem, this paper is to serve as the medium in factory automation and to bring intelligent robots and sensors together through an integrated (including the Web, VR, object-oriented program, Internet, and wireless protocol) collaborative environment. In this environment, production engineers or managers can share real-time data with each other while performing their own duties through this wireless sensor-driven VR-based collaborative environment.

The goal of this paper is to propose an appropriate methodology with open architecture for real-time monitoring and remote control of networked mobile robot. Within the context, this paper presents a new enabling technology to bring traditional robotic tools on-line with combined monitoring and control capability. The remaining of this paper is arranged as follows. Section 2 presents a brief

J. S. Liang (✉)
Department of Vehicle Engineering,
Yung-Ta Institute of Technology and Commerce,
316, Chung Shan Road,
Linlo, Ping Tung 909, Taiwan, Republic of China
e-mail: janus@mail.ytit.edu.tw

K.-M. Chao · P. Ivey
Faculty of Engineering & Computing, Coventry University,
Coventry, UK

literature review of related work. Section 3 proposes the architecture of VR-based-integrated wireless sensor-driven assembly (Vbiws-Assembly). In addition, the author proposes an integration framework based on Common Object Request Broker Architecture (CORBA) standards to achieve interoperability among multiple data and application objects in the heterogeneous environments. Section 4 shows the remote real-time assembly. Prototype implementation relevant to remote assembly is addressed through a wheeled mobile robot case study in Section 5. Base on the case study, Section 6 summarizes this paper.

2 Brief literature review of related work

Since 1993 shortly after the debut of the Web, a number of methods and frameworks have been proposed for building Web-based collaborative systems. Most of the frameworks were developed for collaborative design, project management, and conflict resolution during collaboration [2–4]. In terms of technologies used in the existing systems, Java applets, HTML, and ActiveX are widely adopted for developing the client-side user interfaces. At the server side, technologies including JavaServer Pages, Java servlets, and XML are quickly obtaining attentions for new system development. Meanwhile, VR technology [5, 6] has been widely used in industry, especially in Web-based virtual design and machining simulation system that has become popular because of its cooperation facility and good platform independence [7–10]. The virtual Reality Modeling Language (VRML) is a file format for describing interactive 3D objects, having the character of being simple to use, open, and interactive which make it a powerful tool for setting up the Web-based-simulating systems [10–12]. Although VRML has a strong ability for graphics display and good Web running, it lacks the functions for file reading and writing, as well as its inferior two-dimensional word display effect. This means that make VRML cannot meet the special demand of virtual assembly of a wheeled mobile robot. For these causes, we propose a combination of VRML, Java applet, and JavaScript to solve this problem.

In the design, assembly, and manufacturing area, there has been a lot of research and development and a numbers of software tools have been developed in this field. Lee et al. [7] proposed a Web-enabled approach to feature-based modeling in a distributed computing environment. In this work, several key technologies of the Internet-based computer-aided design (CAD) systems, such as shape abstraction and client–server communication, were studied. Li et al. [13] developed an Internet-based integrated system which enabled designers to design products collaboratively and supported product preview and evaluation of design parts using Web-based virtual reality technology. Choi and Chan [14] presented a virtual prototype system for rapid product development. This system incorporates the dexel-based and the layer-based fabrication approaches to simulate the powder-based and laminated sheet-based RP process, respectively. Pang and Wittebrink [15] developed a shared viewing environment called Cspray to support distributed users to interactively view their data from independent or shared camera positions. Yeung et al. [16] proposed a comprehensive virtual simulation model of realistic and modular computer numeric controlled (CNC) system which can be sued to predict realistic CNC system performance. Liang [17] presented several Web-based interfaces for user interaction and a task-oriented decision approach to build a tasks schedule model and then display the manufacturing process in a virtual environment, which is created by a geometric virtual reality-based visualization technology for screw threads generation.

In the assembly filed, much work has been done since the early 1980s on the assembly sequence generation (or assembly sequences planning). These researches focused on sequences generating and sequences representation [18–21]. As for the Internet-based framework, Shyamsundra and Gadh [8] developed a new compact assembly representation, called AREP, for Internet-based collaborative assembly design involving clients and subcontractors. Jayaram et al. [22] explored the potential and technical challenges by using virtual assembly technologies in design and manufacturing. Wang et al. [23] proposed a CAD-linked virtual assembly environment. It is designed as client/server architecture and integrates virtual assembly applications with an external CAD system through automation interfaces. Generally, most virtual assembly environments are originally generated from CAD models, which were constructed by commercial CAD software such as SolidWorks and Mechanical Desktop and so on. In contrast, VR applications focus on a more intuitive and immediate understanding of simulated objects. And the performances of real-time rendering, interactive, and immersive visualization take more priority for VR applications.

Literature documents that robotic assembly planning system can be broadly classified as stand-alone systems and Web-based systems [24]. Graphical robot interactive programmer [25] is an off-line programming system and was built to serve as an interactive path planner for pick and place operations. The robot is taught the path using graphically simulated teach pendant which is automatically converted into the robot program as per the format of the controller. Intelligent assembly modeling and simulation [26] was generated at CMU for assembly simulation and visualization to detect assembly-related problems without going through physical mock-ups. Here, the user can specify a high-level assembly plan which is automatically converted into low-level tool and part motions. All the above system target on specific robot configurations and assembly

operations and are of the stand-alone types. In the Web-based category, a Web-based telerobot system was reported [27] wherein the user submits the request by filling up the fields of the HTML form and receives the textual and graphical information of the current state of the robot at the remote site. A model-based telerobotics system with vision for assembly-like operations wherein the user specifies the task interactively using a mouse and receives the image of the current state of the robot as the feedback [28] was implemented for a single user only and did not incorporate anytime, anywhere philosophy.

In this paper, we address the remote real-time assembly problems by means of wheeled mobile robot in an assembly company. The objective of this paper is to develop an appropriate methodology with open architecture for real-time monitoring and remote control of networked mobile robot. An integrated product data model and the related configuration management methods are generated. Within the context, this paper presents a new enabling technology to bring traditional robotic tools on-line with combined monitoring and control capability. Under a hybrid architecture of Web browser/server and client/server, an application system is presented to manipulate the wheeled mobile robot data and to carry out a variety of assembly functions. Advanced and distributed shop floor monitoring and remote assembly control remain impractical as Web-based applications due to the real-time constraints. To facilitate a viable VR-based environment on the Web, application servers must engage users in a 3D graphical interaction in addition to the dialog-like data sharing because remote users need active and visual supports to coordinate their efforts in a distributed environment. To be practical, a Web-based monitoring and control system should also be efficient and adaptive and can address dynamic issues such as job delay, product changeover, etc. that happen unpredictably in manufacturing shop floors.

3 Architecture of application system

From the point view of users, there are two basic types of information flows concerned in the remote assembly environment. In forward information process, assembly data from the centralized database are distributed to users, e.g., operators, engineers, and managers. In the backward information process, the controlled data are collected from users and entered into the integrated database after further processing. During the remote assembly processes, an integrated information application system is required to validate and control access to the assembly database, to generate and record various data and files. The application system is also acquired to facilitate a variety of planning, managing, and controlling functions and to create specific reports in the wheeled mobile robotic assembly cycle. Many commercial systems, e.g., product information management, technical data management, product data management, etc., are available to efficiently manage product data and to control change [29–31]. As for many business systems, such as SAP's R/3 system [32, 33], can carry out sophisticated manufacturing planning activities on the basis of effectively managing product data. However, these commercial systems are found not suitable for the specific remote assembly cases. These systems are too large and complicated, offering many redundant and requiring long implementation times. On the other hand, these systems are deficient of functions uniquely required in the remote wheeled mobile robot assembly, such as remote data access and collecting through Web browsers. Thus, a specifically tailored remote wheeled mobile robot assembly system is developed to manage and control these information flows.

3.1 Concept of Vbiws-Assembly

The Vbiws-Assembly is designed to provide users with a VR-based and wireless sensor-driven intuitive assembly environment where real-time monitoring and remote are undertaken. It applies the Java technologies, including Java servlets and Java 3D, as enabling technologies for system implementation. A physical apparatus of interest (e.g., a robotic arm or mobile carrier) is generated by a scene graph-based Java 3D model, i.e., virtual model, in an applet with behavioral control nodes embedded instead of camera images. The virtual model can work on behalf of its remote correspondent showing real behavior for visualization at a client side when it is downloaded from an application server. It remains alive by connecting with the physical device through message passing (user commands and sensor data). By combining virtual reality models with real apparatus through synchronized real-time data communications, the Vbiws-Assembly allows engineers and managers to assure normal shop floor operations and enabled Web-based troubleshooting particularly helpful when they are off-site. Figure 1 illustrates how it is connected to a real apparatus in a shop floor. An off-the-shelf Web-ready camera can easily be shifted on remotely to catch unpredictable real scenes for diagnostic intentions, whenever it is demanded. Besides, the real-time monitoring and control, the framework can also be extended and applied to many phases, e.g., design verification, virtual machining, remote diagnostics, and so on.

3.2 Framework of Vbiws-Assembly

The framework is generated to use the client/server architecture and model–view–controller design pattern [34–37] with built-in secure session control. The framework is

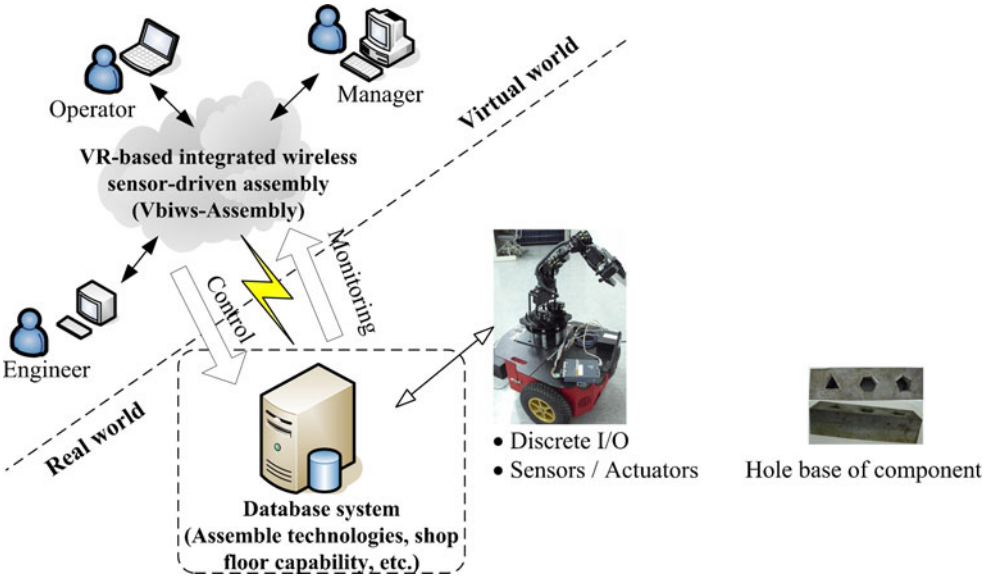


Fig. 1 The concept of Vbiws-Assembly

shown in Fig. 2. Targeting the real-time monitoring and remote assembly of wheeled mobile robot, data communication, and security issues are examined. The proposed solutions for meeting the user requirements of rich visual data sharing and the real-time constraints are listed below: (a) provide users with VR-based graphical user interface for assembly navigation in shop floor; (b) apply interactive virtual models instead of camera images for assembly visualization; (c) transmit the data (sensor and user commands) between virtual models and real device controllers for

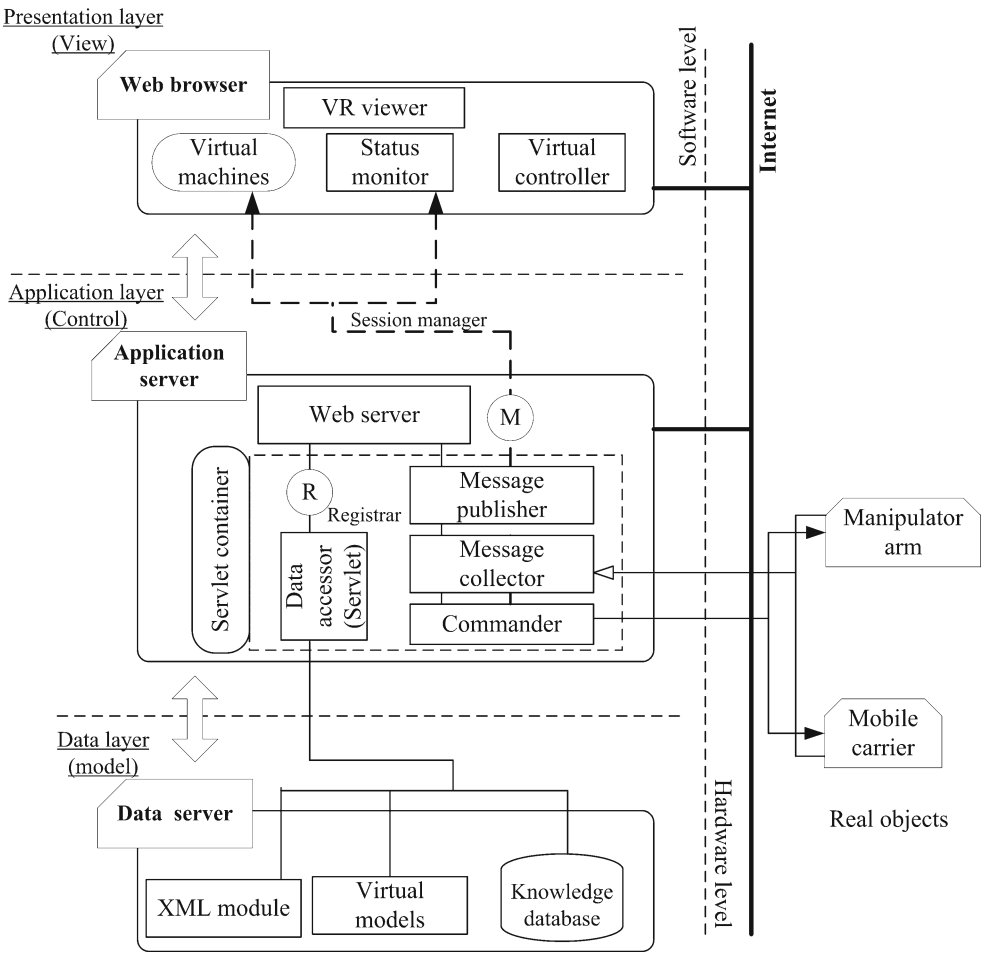


Fig. 2 Framework of application system—Vbiws-Assembly

remote monitoring and control; and (d) deploy major control logics in a secure application server.

In client tier, the virtual models are operated by the server based on real-time sensor messages, but users still have the flexibility of monitoring the models from different perspectives, e.g., changing viewport, picking different virtual models, and adjusting view scale, through VR viewer at client side. Authorized users can submit commands through Virtual controller to the application server. The commander at the server side then takes over the handle for real apparatus manipulations. Furthermore, the module Status monitor can offer users with a view of run-time status of the controlled apparatus.

In server tier, the application layer manages important security items, such as session control, viewer registration, real apparatus manipulation, data collection/distribution, and so on. A Session manager is created to look after the issues of user authentication, session control, session synchronization, and data logging. Constrained by network security, a virtual model residing in an applet is not allowed to communicate directly with a real apparatus through socket communication. Is it also not efficient to have multiple clients who share the same model talking with the same apparatus at the same time. Thus, publish–subscribe design pattern [38–41] is used to collect and distribute sensor data at the right time to the right client efficiently. The Message collector is responsible for sensor data collection from networked physical apparatus. The data are then passed to another module Message publisher that in turn multicasts the data to the registered subscribers through applet–servlet communication. The Registrar is generated to keep a list of subscribers with the requested sensor data. The virtual model thus can communicate indirectly with sensors wherever the client is, inside a firewall or outside. An HTTP networking protocol is chosen for the combination between applets and servlets. For the sake of security reasons, a real apparatus is controllable only by the Commander that resides in the application layer. Furthermore, Data accessor is built to divide the physical and logical types of data. It encapsulates Java Database Connectivity and SQL queries and offers standard methods for accessing data (virtual models, knowledge base of the physical apparatus, or XML documents).

3.3 Information integration

The application system has been generated to support object-oriented concepts and reconfigurability. For the moment, the assembly database is implemented with the relational database management system MySQL. Remote external users download from and/or upload various data to the assembly database. The data sources at the remote users are separated by the neutral front-ends (i.e., the browsers). A CORBA-based integration framework is proposed to achieve the dynamic interoperability among the data sources distributed. The CORBA-based architecture provides the ideal means to support distributed transactions involving multiple application objects and spanning multiple data sources over the Internet. The information integration framework is based on Web, Java/RMI, and CORBA standards [42–46], and the diagram is shown in Fig. 3. In the integrated framework, data sources are encapsulated into information objects and are defined and registered by the brokers. The application systems are encapsulated as application objects, which are connected through adapters to CORBA-compliant object request broker bus across the Internet and company Intranet. Remote users create http connections with the Web server containing data sources and download the necessary Java applet and html documents over the Internet by using browsers. Afterward, the remote users run the Java applets on the virtual machines, to

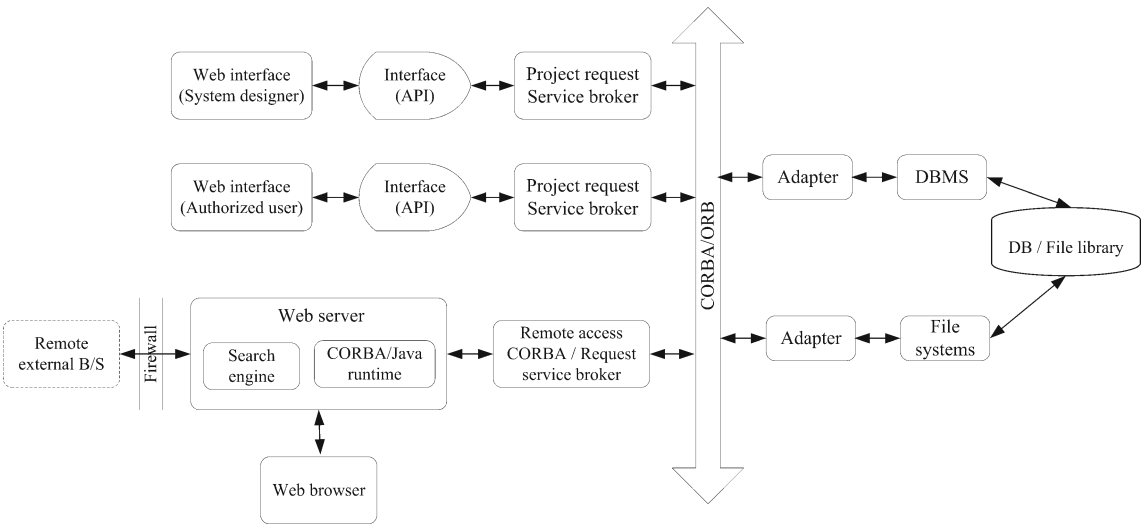


Fig. 3 The diagram of information integration framework

build connection with CORBA request/service brokers embedded in the Web server. Remote users can have access to various data sources linking to the CORBA bus by invoking the request/service brokers. Thus, users can inquire, modify, commit, and transmit related assembly data.

The emphasized essence of the framework is that all the information transactions are conducted with the help of the brokers. The CORBA object request broker offers the information infrastructure to link the remote data and application objects. Meanwhile, the brokers keep the information transparency in the environments, separating application objects from the details of data access. The brokers will find the right target data object when an application object issues a request, then invoke it, deliver it the message, and return the response messages to the application object.

4 Remote real-time assembly

4.1 Configuration of test platform

The test platform used as a case study in this paper is a wheeled mobile robot. It is composed of two parts—a mobile carrier (PIONEER 3) and a 7-degree of freedom manipulator arm with a gripper end effector (CYTON ALPHA 7D 1G). As illustrated in Fig. 4, it is equipped with a PC-based open architecture controller that serves as a gateway between wheeled mobile robot and the application system. The wheeled mobile robot is controlled by a user interface, implemented as java application, running on the same open architecture controller PC, which in turn, beyond a local operator, allows a remote user to send assembly commands to the open architecture controller from a remote Web browser and have them executed locally. Remote users can monitor and/or control the wheeled mobile robot indirectly through the server-side modules (as shown in Fig. 2) that implement data collection/distribution and assist remote component assembly. Meanwhile, two different communication protocols are adopted for data transmissions at different tiers: (1) transmission control protocol for hardware protection and (2) HTTP streaming protocol that is firewall-transparent. Based on this configuration, it allows remote user to monitor the motions of all axes of a manipulator arm and mobile carrier as well as to control the gripper speed and force for assembly task.

4.2 Assembly task specification and robot program synthesis

The primary objective of this application system is to facilitate user to specify assembly operations in the virtual at the task level. Vbiws-Assembly will convert this task level assembly commands into a detailed wheeled mobile robot path using inverse kinematics [47–51]. Various issues related to task level program synthesis in the Vbiws-Assembly are discussed below.

4.2.1 Assembly tasks

At the task level, assembly process is typically characterized by the following motion: (a) component placement (*Move*) in assembly region; (b) component (C_i) *On* hole base (H_j); (c) component (C_i) *In* hole base (H_j)—peg-in-hole. Besides, parts in the assembly environment are classified into stationary objects and movable objects. Movable object (i.e., component) can be interactively "picked" up by the user and "assembled" onto the stationary object (i.e., hole base). Figure 5 illustrates a typical task specification dialog in Vbiws-Assembly wherein the assembly of different geometric shape components on a base with linear array of holes is being specified by the user. Parts in the virtual assembly world are classified into fixed objects and movable objects.

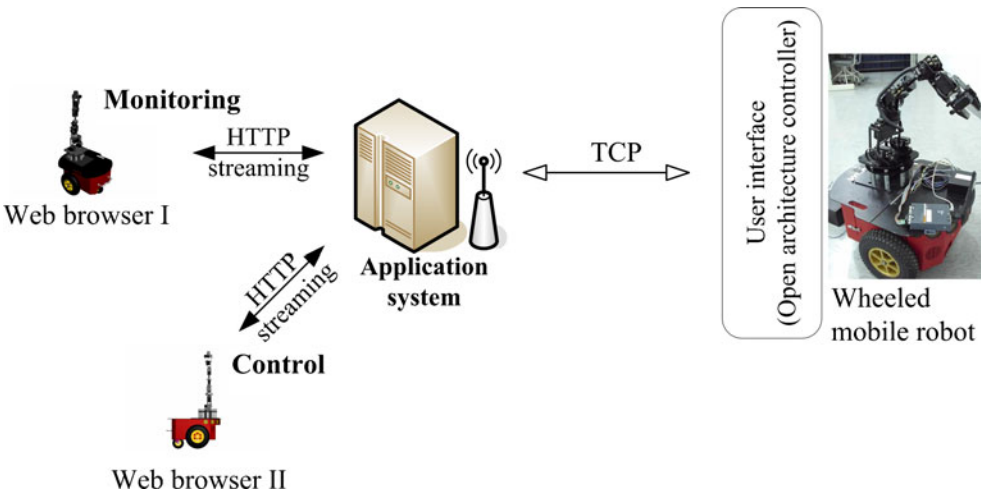


Fig. 4 Configuration of test platform

Movable objects (i.e., the components on the component platform) can be interactively picked up by the user, clicked the virtual model directly, and then assembled onto the corresponding fixed objects (i.e., the holes on the hole base). The aligned point is a mark between the component and its corresponding hole for locating assemble orientation. Furthermore, the graphical dialog also shows the top view of the virtual assembly world indicating the positions of the wheeled mobile robot, the fixed as well as movable objects. The contents are intelligently updated as the assembly specification proceeds step by step. Several validation checks have also been incorporated while specifying the task. Important among them are as under: (a) assembly task specified by the user for the features in the fixed part is feasible. For instance, *In* task for a peg must have a Hole feature with proper geometric constraints of size and orientation; (b) the parts specified in the task are already modeled with requisite geometry, location, and orientation; and (c) the parts once placed (and the corresponding features on fixed object) cannot be reused. The Vbiws-Assembly intelligently does this by automatically disabling the specified part (movable) and its related features in fixed object from the dialog. Meanwhile, the assembly tasks entered by the user are stored a text file containing records as per the following format:

Format

{*Task*}:{ID_Source_Part}:{*Task_Type*}:{ID_Destination_Part}:{ID_Location_Feature}

Example

Place:C_1 *In*:HB *At*: H_1

The assembly task planner thus enables the user to specify assembly tasks at a higher conceptual level in a very elegant manner. The assembly procedure will be synthesized by Vbiws-Assembly.

4.2.2 Robot program synthesis

This module essentially processes the assembly modeling and the assembly task specification files generated as a result of user interaction with Vbiws-Assembly. It carries out the following functional tasks: (a) interpreting the assembly model and task files; (b) planning the motion of wheel mobile robot; and (c) creating the program of wheel mobile robot. The assembly task files are interpreted to decide the proper sequence of assembly for the specified task (*Move*, *On*, and *In*) and to check the feasibility of task in terms of the motion of wheeled mobile robot. For the task specified, it internally checks whether there is sufficient space around the object so that the wheeled mobile robot can grip the object during pick and placement without interfering with the surrounding objects. The motion of the wheeled mobile robot for the assembly is planned and verified by checking that manipulator arm motions do not come closer to limits of joint angles and link lockups [47, 48]. Once the task is feasible, the program of wheel mobile robot is automatically created and post-processed to suit the specific wheel mobile robot controllers (Robai Cyton and ARCOS microcontroller). Vbiws-Assembly creates assembly procedure form the task file by intelligently doing the motion planning.

Planning the motion of wheeled mobile robot A proper procedure of assembly is evolved among the tasks specified in the task file, keeping in view the space around the parts in virtual assembly world. For this task, objects are approximated to convex polygonal shapes, leaving some safety envelope around them. Dimension of the safety clearance is chosen as a function of convex hull of the object. Meanwhile, there are two portions confirmed in this stage —grasp planning and robot motion planning. Grasp planning primarily generates a grasp method for assembly plan which is further used for computing data for sequential robot positions, coordinate values, joint angles, etc. In the present work, axis *Z* is considered as the direction of object access,

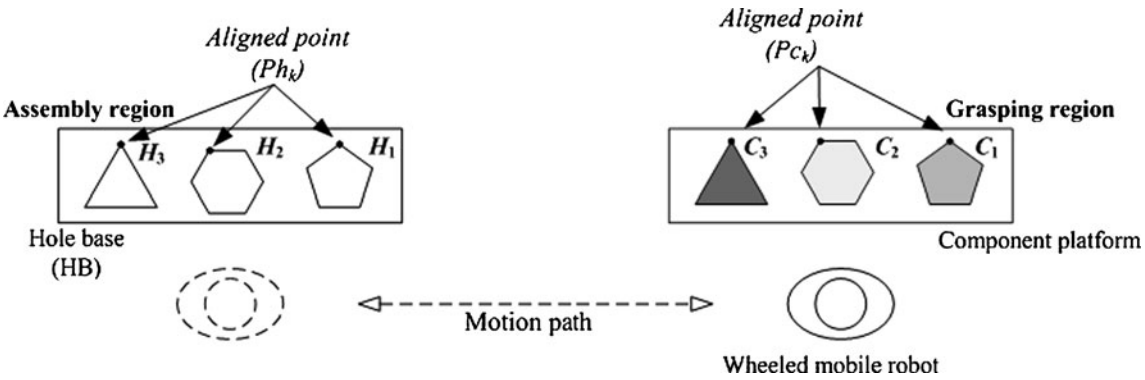


Fig. 5 Diagram of task specification for component assembly

gripping, and assembly. Geometric algorithms are written to extract and process the following information from the virtual model file: object dimensions, position, orientation, convex hull, free space around objects, faces for gripping, access directions of gripping faces, and so on. The purpose of the grasp planning is to select the suitable initial and final grasp configuration (position and orientation). It is confirmed based on the following situations: (a) object geometry and assemble direction; (b) object stability during gripping and transfer; (c) collision-free grasp configuration; and (d) pair of parallel surfaces of the target object since the gripper is parallel jaw type. Single part placement strategy is discussed as an example. Several steps are listed:

1. The data of movable object are extracted from assembly model file, such as shapes, dimension, location, orientation, etc.
2. The initial position of the gripper depends on the height of the movable object under consideration (as shown in Fig. 6a).
3. Because the gripping is along axis Z, the gripper can take any orientation about vertical axis while grasping the component. In addition, it is a parallel jaw gripper, so gripping is done normal to the pair of parallel surfaces of the component. The pair of parallel surfaces is chosen considering the minimization of the torque about the axis between the gripper and the position of aligned point. Besides, collision-free grasping has to be considered.
4. The final position of the movable object depends on the goal feature (i.e., the depth of hole) in the fixed object. In case of Part C_i *On* Part H_j, the final position depends on the Part C_i height (as shown in Fig. 6a).
5. The final orientation of the movable object depends on the position of aligned point of the corresponding hole. Considering the linear array of holes in the hole base (as shown in Fig. 6b), three gripping orientations are evaluated: along axis X, along diagonal, and along axis Y.

Scope required for gripping is computed considering the gripper width and the component sizes. Method to grape along X, diagonal, or Y is confirmed based on the available space as indicated in Fig. 6b. The orientation of the gripper is decided by its coordinate system with respect to the robot coordinate system (Rcs). Therefore, the orientation of the gripper is computed by calculating the direction cosines (d.c.'s) of each axis of gripper coordinate system (Gcs). The position and orientation of the gripper are together represented in a Denavit-Hartenburg (D–H) transformation matrix [46, 47]. The format is shown below:

$$\begin{bmatrix} T_{11} & T_{12} & T_{13} & X \\ T_{21} & T_{22} & T_{23} & Y \\ T_{31} & T_{32} & T_{33} & Z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

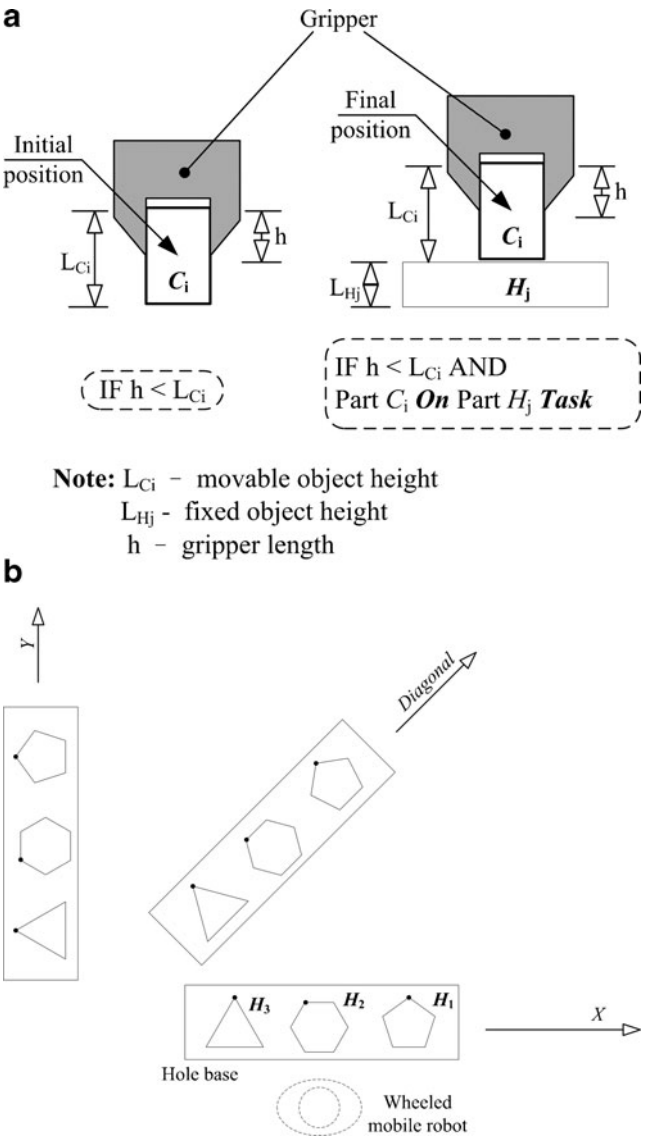


Fig. 6 **a** Initial and final position of gripper. **b** A linear array of holes in the hole base

Here, T_{11}, T_{21}, and T_{31}: d.c.'s of axis X of Gcs with respect to Rcs; T_{12}, T_{22}, and T_{32}: d.c.'s of axis Y of Gcs with respect to Rcs; T_{13}, T_{23}, and T_{33}: d.c.'s of axis Z of Gcs with respect to Rcs; X, Y, and Z: coordinate of Gcs with respect to Rcs.

After finding the initial position and orientation, they are represented in the above matrix form and the joint angles of the robot arm manipulator are calculated suing the inverse kinematics analysis. The joint angles of the final position and orientation of the gripper are also calculated.

As for the robot motion planning, a rough flowchart of robot plan is shown in Fig. 7. The motion plan is worked out by assigning the Z-value of the middle positions at sufficient

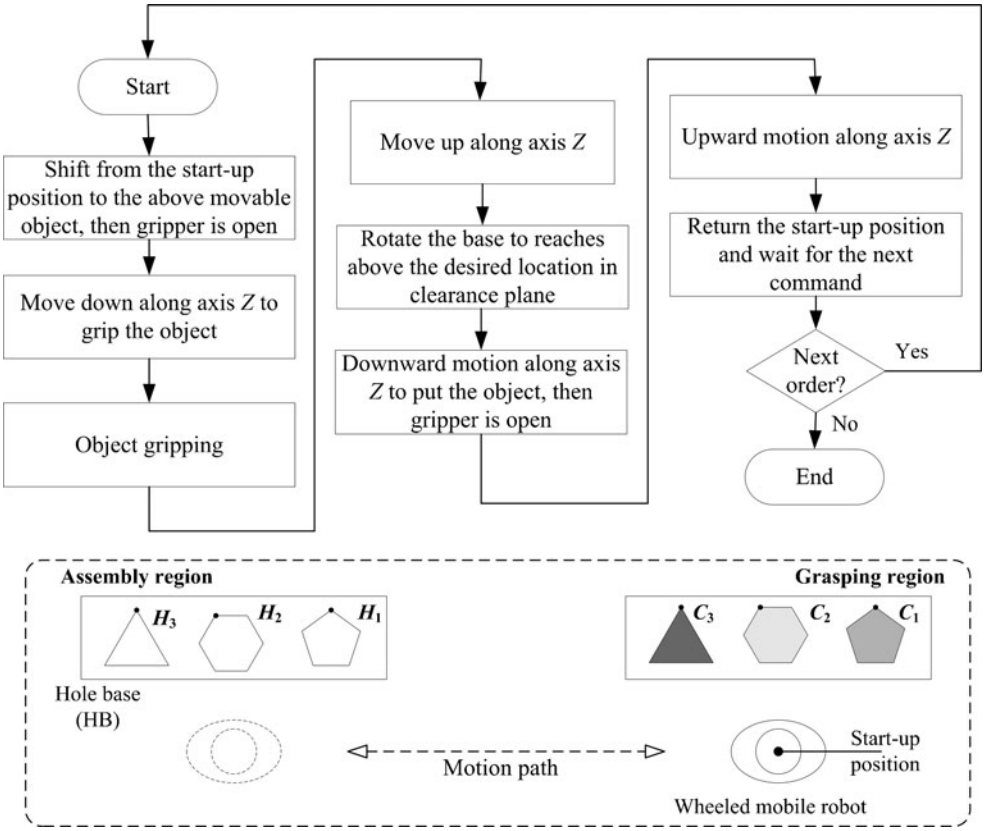


Fig. 7 A rough flowchart of robot plan

height so as to avoid collision with other objects in the assembly world during the robot motion. At the middle origin position just after grasping the target object, the orientation of the gripper is taken as the initial grasp orientation which is computed in grasp plan. Similarly at the middle target position just before the destination, the orientation of the gripper is taken as the final orientation. The D–H transformation matrix is used for representing the position and orientation of gripper at middle origin position. In addition, this matrix is adopted to calculate the joint angles through inverse kinematics. Moreover, the joint angles at middle target position are also calculated.

Creation the program of wheeled mobile robot As the above mentioned, the joint angle positions corresponding to the middle and the final positions are calculated for the grasp and robot motion plan. The program of wheeled mobile robot is created automatically from this data and further post-processing is performed to generate an executable file to match the format required by the specific wheel mobile robot controllers. Figure 8 illustrates wheeled mobile robot program generated by OOP as well as in XML format. It can be transferred to the server for the direct execution of the wheeled mobile robot. Meanwhile, the application server is requested to set up the real assembly world as communicated by client in terms of wheeled mobile robot position, assembled components, etc. The application server checks up the program files and further transfers it to the controllers through wireless serial Ethernet and was extensively tested from various sites.

4.3 Data collection and distribution

As mentioned in Section 3, the task of sensor data collection and distribution is handled by a set of server-side modules, i.e., Message collector and Message publisher. For every client registered with the Registrar, there is one assigned Subscriber that is knowledgeable of the user's request and responsible of maintaining an active channel between the client and the server. Multi-threading techniques are applied to the module implementations. A comprehensive data flow are given Fig. 9. The reason of choosing HTTP streaming as the protocol for data transmission is because it uses standard port 80 and can pass through firewalls. As a result, a wheeled mobile robot user can receive real-time data anywhere if network connection is provided.

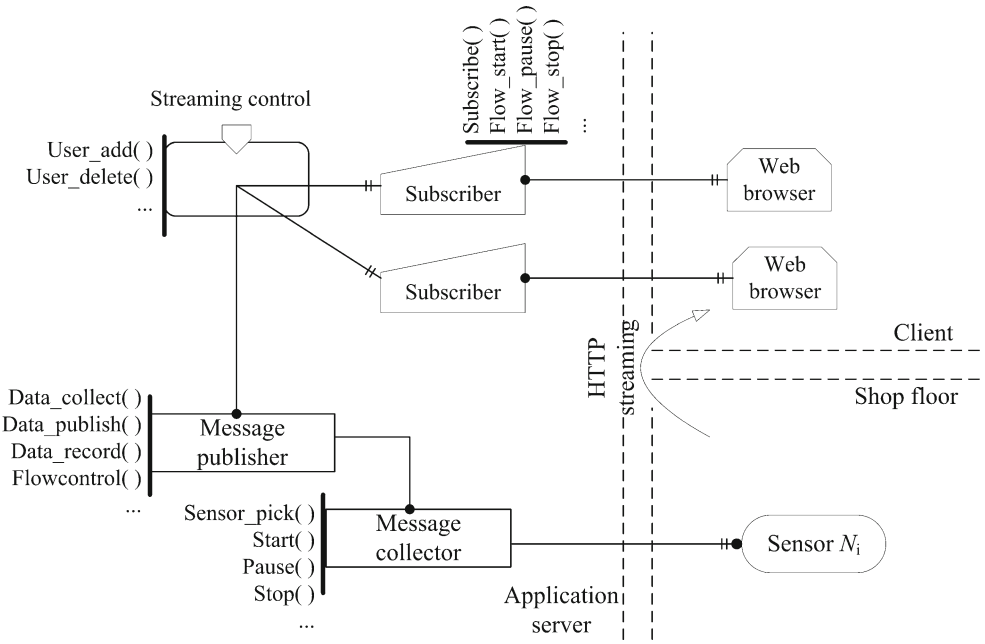
Meanwhile, the modules help the clients to transfer the generated files, i.e., virtual models, task, and program files, to the server tier. The Web interface has an option of file transfer wherein the user can browse to select the files and finally transfer to the server tier. The user is sent an acknowledgement in the form a message dialog box when the server receives the file.

Fig. 8 XML format of wheeled mobile robot program

```
<?xml version="1.0" encoding="ISO-8859-1: ?>
<statePath Xmlns:xsi="http://www3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.me.ytit.edu.tw/nanmespace/sm al.smxsd"
       xmlns="http://www.me.ytit.edu.tw/namespace/sm" version="2.0.0">
<ct:stateVector xmlns:ct="http://www.me.ytit.edu.tw/namespace/ct" size="215">
<ct:element>
<mn:positionStates xmlns:mn="http://www.me.ytit.edu.tw/namespace/mn" size="1">
<mn:elemnt>
<mn:coordinateSystemTransform>
  <mn:orientation q0="1" q1="0" q2="0" q3="0"/>
  <mn:translation x="0" y="0" z="0" />
  </mn:coordinateSystemTransform>
<mn:jointPositions size="9">
  <mn:group>=1.6690552960439478 -0.010944476041613107 -0.54576368896538952
      0.05600000000000005 0.0700000000000062 0.25179685249929462
      -0.63465309928604408 -0.012550044217380308
      0.012550044217380308</mn:group>
  </mn:jointPositions>
  </mn:element>
  </mn:positionStates>
  <mn:time xmlns:mn="http://www.me.ytit.edu.tw/namespace/mn">0</mn:time>
<mn:velocityStates xmlns:mn="http://www.me.ytit.edu.tw/namespace/mn" size="1">
<mn:element>
<mn:generalVeloctiy>
  <mn:angular x="0" y="0" z="0" />
  <mn:linear x="0" y="0" z="0" />
  </mn:generalVelocity>
<mn:jointVelocities size="9">
  <mn:group>-0.095493139254595391 -0.066695542576247116 0.24369013498409434
      0.036956259663191132 -0.013500057000541466 0.0035728114804006728
      -4.4408920985006262e-016 1.7229510632999979e-008
      -1.7229510632999979e-008</mn:group>
  </mn:jointVelocities>
  </mn:element>
  </mn:velocityStates>
  </ct:element>
...
```

4.4 Visualization

For the sake of network bandwidth conservation, Java 3D is used as an alternative of camera-based solution for Web-based visualization [52–55]. Enabled by the scene-graph architecture, Java 3D provides an abstract, interactive imaging model for behavior control of 3D objects. Utilizing the Java 3D technology, a machine of interest (wheeled mobile robot in this case) can be modeled as a scene graph, representing its physical counterpart in the virtual assembly environment. The data transmitting through the network are limited to the data showing run-time status of the machine (position, orientation, rated speed, etc.). Driven by the real sensor data, the virtual model can demonstrate the true behavior of the real machine with largely improve network performance. Figure 10 illustrates the Java 3D scene graph architecture of the wheeled mobile robot. The scene graph involves a complete description of the entire scene. It contains the geometry data, the attribute information, and the viewing information needed to render the scene from a particular point of view. The Virtual Universe object offers grounding for the entire scene. A Branch Group node serves as the root of a sub-graph, or branch graph, of the scene graph. The Transform Group nodes inside of a branch graph specify the position, the orientation, and the scale of the geometric objects in the virtual universe. Each geometric object comprises a Geometry object, an Appearance object, or both. The former depicts the shape of an object; the latter records the appearance of the geometry, e.g., color, texture, light, etc. As for the behavior of the virtual model, the Behavior node is assigned and it contains user-defined control codes and state variables. Sensor data processing can be embedded into the codes for remote monitoring and control. In this case study,



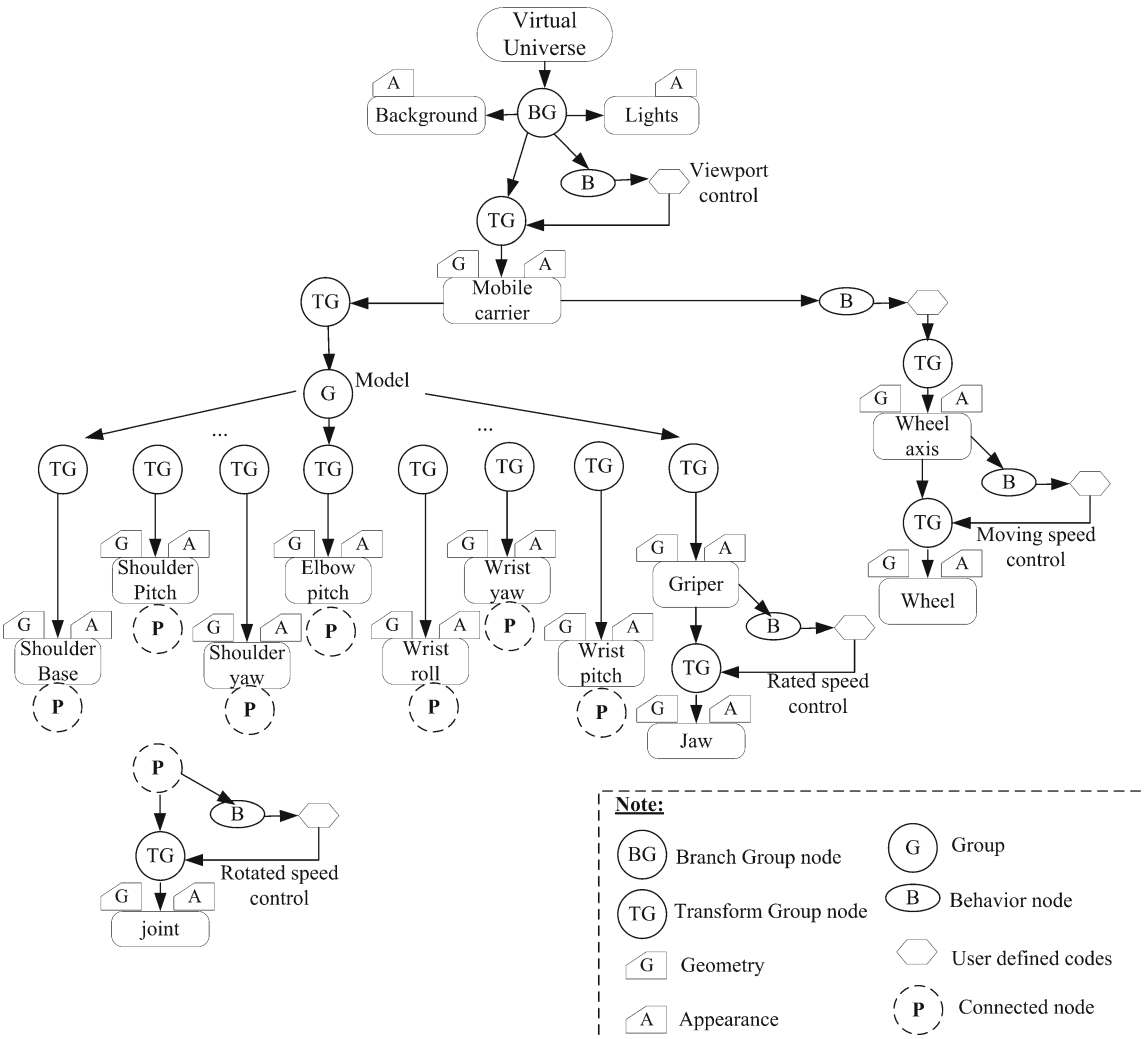


Fig. 9 Data collection and distribution through HTTP streaming mode

Fig. 10 Scene graph architecture of wheeled mobile robot in Java 3D

the motions of wheeled mobile robot (wheels, shoulder base joint, shoulder pitch joint, shoulder yaw joint, elbow pitch joint, wrist roll joint, wrist yaw joint, wrist pitch joint, and gripper jaw) are controlled by their corresponding behavior control nodes, for on-line monitoring/control and off-line simulation. As the virtual model is connected with its physical fellow through the control nodes by low-volume message passing (e.g., real-time sensor signals and control commands), it becomes possible to remotely manipulate the real wheeled mobile robot by its virtual model.

5 Implementation

A proof-of-concept prototype is developed on top of the framework to demonstrate its application in remote monitoring and control. Figure 11 illustrates one snapshot of the Web user interface of the prototype. A more detailed discussion is provided through a wheeled mobile robot case study. Meanwhile, Vbiws-Assembly has been implemented in the modular fashion using object-oriented programming and extensively tested from different locations connected through Internet. Various assembly tasks such as component placement (*Move*) in assembly region, component (C_i) *On* hole base (H_j), and component (C_i) *In* hole base (H_j)—peg-in-hole are modeled in the virtual assembly world and assembly tasks specified to automatically generate and transfer wheeled mobile robot programs.

The programs are tested by setting up the real assembly environment in the CIM laboratory. A wheeled mobile robot is linked to a server using wireless serial Ethernet to test out the assembly tasks. Positioning accuracy of the wheeled mobile robot is tested for different paths in assembly world. This positional error is accounted by calculating the error at the required position and adding it to the ideal position. Using calibration data, the wheeled mobile robot programs are automatically corrected by the Vbiws-Assembly to improve the positioning accuracy. Figure 12 shows the photograph showing a typical assembly being carried out by the wheeled mobile robot for placing a hexagon component into a linear array of holes in the hole base.

6 Conclusions

This paper proposes an approach to remote real-time component assembly. It is implemented as a Web-based system

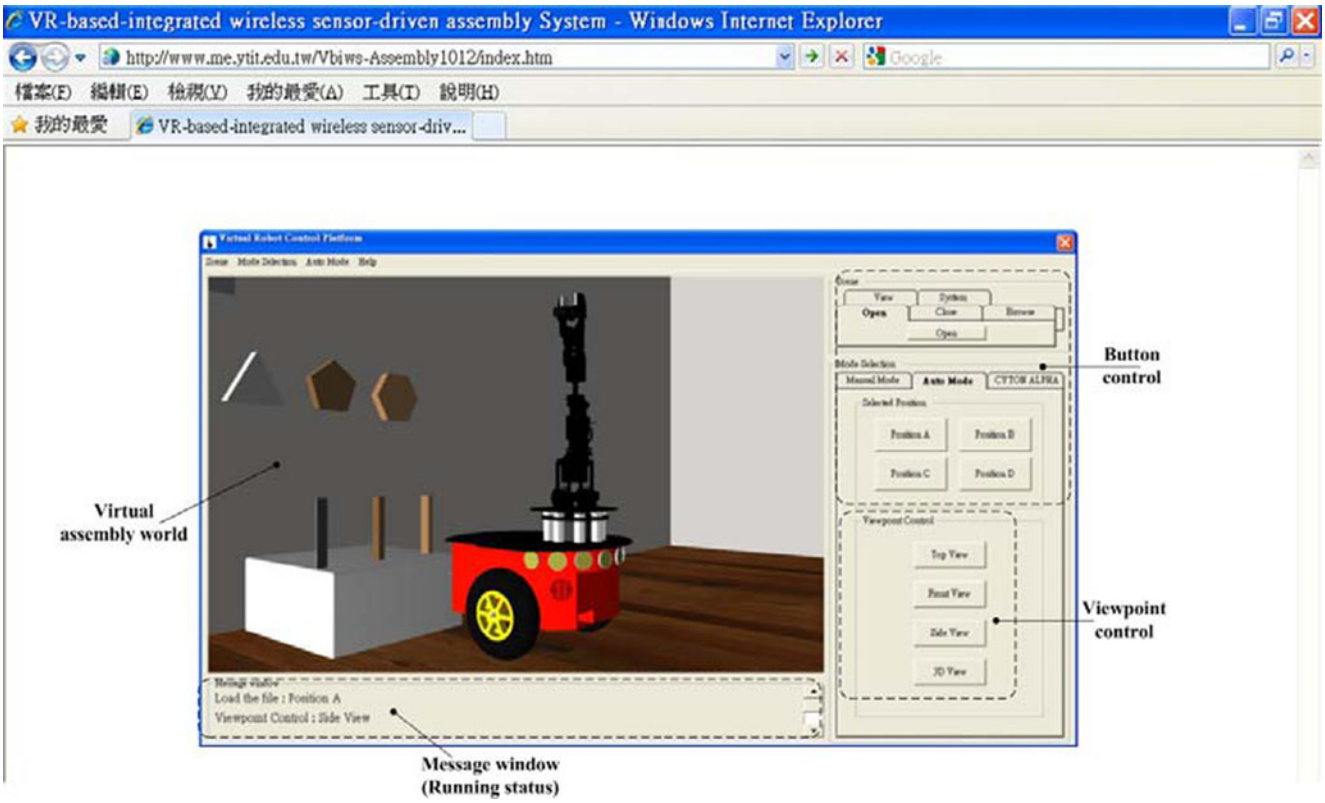


Fig. 11 Web user interface to Vbiws-Assembly. *Step I*—Move down to grip the component. *Step II*—Component gripping. *Step III*—Move to the assembly region. *Step IV*—Place a hexagon component in the hole base. *Step V*—Gripper is open and upward motion

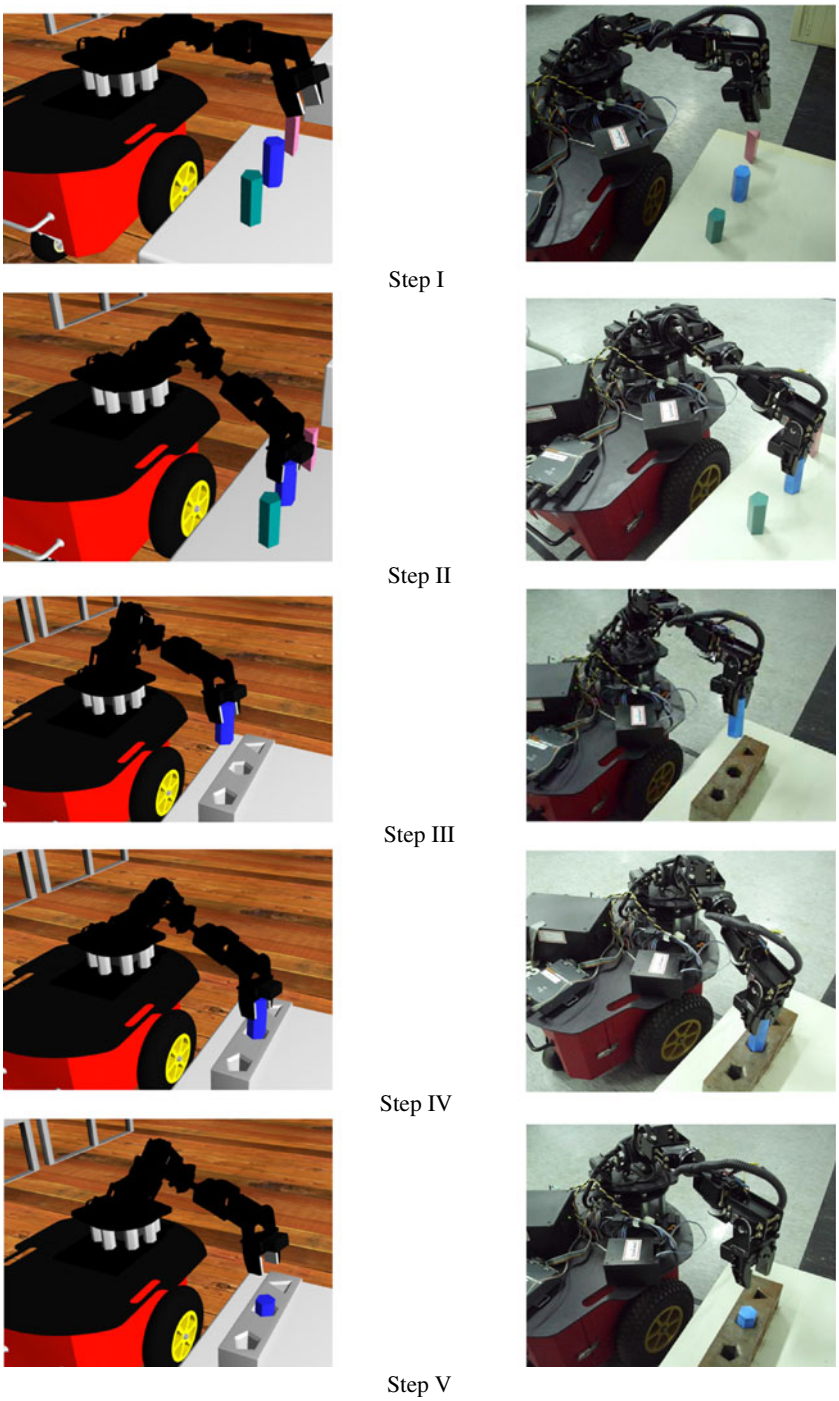


Fig. 12 Snapshot showing the assembly procedure on the Vbiws-Assembly and its corresponding real scene

on top of the Vbiws-Assembly framework with three-layer architecture. The main goal of our combined Web-based and sensor-driven approach is to reduce network traffic by using virtual models, while still providing remote users with intuitive environments. Participating in the virtual model-enabled Vbiws-Assembly, users not only can feel largely reduced network traffic by real-time interaction, but can get more flexible control of the real apparatus. A wheeled mobile robot case study demonstrates its feasibility and shows the promise of this approach to the emerging distributed assembly criterion. Meanwhile, remote assembly implementations inevitably depend on the development of a comprehensive assembly data model and the enabling information infrastructures over the Internet.

In the remote assembly environments, many planning, design, and fabrication applications need to operate on multiple heterogeneous data sources. The information infrastructures are very critical to achieve the interoperation among the dispersed multiple data sources. With implementations of remote wheeled mobile robotic assembly

techniques, the company realizes the following competitive advantages: (1) Application management is smoothly streamlined, enabling more prompt response to user's requirements; (2) Web-based techniques allow easy access and distribution of assembly data among users over the Internet; (3) WebPages coded in hypertext and Java applets are used for coordinating activities among the cooperation partners, and for providing user support services. In addition, dynamically maintained WebPages allow users to participate to some degree in the assembly processes.

Acknowledgments This paper is supported in part by the National Science Council in Taiwan for the financial support and encouragement under grant nos. NSC 101-2918-I-132-001, NSC 100-2511-S-132-001, and NSC 100-2511-S-132-002-MY2. Meanwhile, important parts of the system, especially the user interface and programming, were developed by the student Lv Qing-Wen.

References

1. Yao Y, Li J, Lee WB, Cheung CF, Zhejun Y (2002) VMMC: a testbed for machining. Comput Ind 47(3):255–268
2. Caldwell NHM, Rodgers PA (1998) WebCADET: facilitating distributed design support. IEE Colloquium on Web-based knowledge servers UK, 9(1): 1–4
3. Smith CS, Wright PK (1996) CyberCut: a world wide web based design-to-fabrication tool. J Manuf Syst 15(6):432–442
4. Adelson B (1999) Developing strategic alliances: a framework for collaborative negotiation in design. Res Eng Des 11(3):133–144
5. Jayaram S, Connacher HI, Lyons KW (1997) Virtual assembly using virtual reality techniques. Comput Aided Des 29(8):575–584
6. Beier KP (2000) Web-based virtual reality in design and manufacturing application. COMPIT'2000, March 29–April 4, Potsdam, Germany
7. Lee JY, Kim H, Kim K (2001) A web-enabled approach to feature-based modeling in a distributed and collaborative design environment. Concurr Eng Res Appl 9(1):74–87
8. Shyamsundar N, Gadh R (2001) Internet-based collaborative product design with assembly feature and virtual design spaces. Comput Aided Des 33(9):637–651
9. Suh SH, Seo Y, Lee SM, Choi TH, Jeong GS, Kim DY (2003) Modeling and implementation of internet-based virtual machine tools. Int J Adv Manuf Technol 21(7):516–522
10. Lihui W, Orban P, Cunningham A, Lang S (2004) Remote real-time CNC machining for web-based manufacturing. Robot Comput Integr Manuf 20(6):562–571
11. Kamat VR, Lipman RR (2007) Evaluation of standard product models for supporting automated erection of structural steelwork. Autom Constr 16(2):232–241
12. Ong SK, Jiang L, Nee AYC (2000) An Internet-based virtual CNC milling system. Int J Adv Manuf Technol 20(1):20–30
13. Li WD, Fuh JYH, Wong YS (2004) An Internet-enabled integrated system for co-design and concurrent engineering. Comput Ind 55 (1):87–103
14. Choi SH, Chan AMM (2004) A virtual prototyping system for rapid product development. Comput Aided Des 36(5):401–412
15. Pang A, Wittenbrink C (1997) Collaborative 3D visualization with Cspray. IEEE Comput Graph Appl 17(2):32–41
16. Yeung CH, Altintas Y, Erkorkmaz K (2006) Virtual CNC system—part I system architecture. Int J Mach Tool Manuf 46(10):1107–1123
17. Liang SJ (2010) An approach for generating a tasks schedule model in web-based virtual manufacturing system of screw threads. Int J Adv Manuf Technol 46(5):737–755
18. Gottipolua RB, Ghosh K (2003) A simplified and efficient representation for evaluation and selection of assembly sequences. Comput Ind 50(3):251–264
19. Jones RE, Wilson RH, Calton TL (1997) Constraint-based interactive assembly planning. IEEE International Conference on Robotics and Automation, 20–25 April, Albuquerque, New Mexico, 913–920
20. Barnes CJ, Jared GEM, Swift KG (2003) A pragmatic approach to interactive assembly sequence evaluation. Proc Inst Mech Eng B J Eng Manuf 217(4):541–550
21. Van Holland W, Bronsvoort F (2000) Assembly features in modeling and planning. Robot Comput Integr Manuf 16(4):277–294
22. Jayaram S, Wang Y, Jayaram U (1999) A virtual assembly design environment. In: Proceedings of the IEEE Virtual Reality 1999 Conference, 13–17 March, Houston, TX, USA, 172–179
23. Wang QH, Li JR, Gong HQ (2006) A CAD-linked virtual assembly environment. Int J Prod Res 44(3):467–486
24. Sunil VB, Pande SS (2004) WebROBOT: Internet based robotic assembly planning system. Comput Ind 54(2):191–207
25. Friedrich H, Holle J, Dillmann R (1998) Interactive generation of flexible robot programs. IEEE International Conference on Robotics and Automation, 16–20 May, Leuven, Belgium, 538–543
26. Gupta SK, Paredis CJJ, Sinha R, Wang C, Brown PF (1998) An intelligent environment for simulating mechanical assembly operations. Proceedings of DETC 1998, ASME Design Engineering Technical Conferences, 13–16 September, Atlanta, Georgia, USA
27. Doulgeri Z, Matiakis T (2006) A web telerobotic system to teach industrial robot path planning and control. IEEE Trans Educ 49 (2):263–270
28. Lloyd JE, Beis JS, Pai DK, Lowe DG (1997) Model-based telerobotics with vision. Proceedings 1997 IEEE International Conference on Robotics and Automation, 20–25 April, Albuquerque, NM, USA
29. Kropsu-Vehkapera H, Haapasalo H, Harkonen J, Silvola R (2009) Product data management practices in high tech companies. Ind Manag Data Syst 109(6):758–774
30. Crow K (2011) Product data management/product information management, DRM Associates [online]. Available from: http://www.npd-solutions.com/pdm.html. Accessed 15 Aug 2011
31. Cmkovic L, Asklund U, Dahlqvist AP (2003) Implementing and integrating product data management and software configuration management. Artech House, Norwood
32. Esteves J, Pastor J (2001) Enterprise resource planning systems research: an annotated bibliography. Communications of the AIS, 7(1): 1–52
33. Mandal P, Gunasekaran A (2002) Application of SAP R/3 in online inventory control. Int J Prod Econ 75(1):47–55
34. Sauter P, Vögler G, Specht G, Flor T (2004) Extending the MVC design pattern towards a task-oriented development approach for pervasive computing applications. In: Proc. Int. Conf. on Architecture of Computing Systems—Organic and Pervasive Computing, Augsburg, 23–26 March 2004, Spinger-Verlag, LNCS 2981: 309–321
35. Sengupta S, Sengupta A, Bhattacharya S (2007) Requirements to components: a model-view-controller architecture. 14th Monterey Workshop, 10–13 September, Monterey, CA, USA, 167–184
36. Gupta P, Govil MC (2010) MVC design pattern for the multi framework distributed applications using XML, spring and struts framework. Int J Comput Sci Eng 2(4):1047–1051
37. Shelest O (2011) Model view controller, model view presenter, and model view—ViewModel design patterns [online], Available from: http://www.codeproject.com/KB/architecture/MVC_MVP_MVVM_design.aspx. Accessed 10 Aug 2011

38. Parzyjegla H, Graff D, Schröter A, Richling J, Mühl G (2010) Design and implementation of the Rebeca pubish/subscribe middleware. Lect Notes in Comput Sci 6462:124–140
39. EventHelix.com (2011) Publish-Subscribe design pattern [online]. Available from: http://www.eventhelix.com/realtimemantra/patterns/publish_subscribe_patterns.htm. Accessed 10 Aug 2011
40. Mitchell JD, Siegel ML, Schiefelbein MCN, Babikyan AP (2007) Applying publish-subscribe to communications-on-the move node control. Lincoln Lab J 16(2):413–430
41. Rouvoy R, Merle P (2007) Using microcomponents and design patterns to build evolutionary transaction services. Electron Notes Theor Comput Sci 166:111–125
42. Curtis D (1997) Java, RMI and CORBA, Object Management Group (1997) [online]. Available from: http://www.omg.org/library/wpjava.html. Accessed 15 Aug 2011
43. OMG (Object Management Group) (2009) Corba basics. Available from: http://www.omg.org/gettingstarted/corbafaq.htm. Accessed 15 Aug 2011
44. Web 3D Consortium (2011) X3D FAQ—What is the status of the X3D specification? [online]. Available from: http://www.web3d.org/about/faq/#process-3. Accessed 15 Aug 2011
45. Liang SJ (2010) A web-based automotive refrigeration troubleshooting system applying knowledge engineering approach. Comput Ind 61(1):29–43
46. Liang SJ (2010) A web-based collaborative design architecture for developing immersive VR driving platform. Int J Comput Integr Manuf 23(10):876–892
47. Siciliano B, Sciavicco L, Villani L, Oriolo G (2011) Robotics: modeling, planning and control, 2nd edn. Springer, London
48. Jazar RN (2010) Theory of applied robotics: kinematics, dynamics, and control (2/e). Springer, New York
49. Mester G (2010) Intelligent mobile robot motion control in unstructured environments. Acta Polytech Hung 7(1):153–165
50. Szépe T (2009) Sensor-based control of an autonomous wheeled mobile robot. Proceedings from PROSENSE 3rd Seminar Presentations, 17–21 November, Ljubljana
51. Mester G (2010) Sensor-based control of autonomous wheeled mobile robots. Ipsi J 6(1):29–34
52. Barrilleaux J (2001) 3D user interfaces with Java 3D. Manning, Greenwich
53. Selman D (2002) Java 3D programming. Manning, Greenwich
54. Liang SJ (2007) A web based 3D virtual technologies for developing a product information framework. Int J Adv Manuf Technol 34(5–6):617–630
55. Xu HT, Lin JY, Chan CC (2003) A web-based product modeling tool-a preliminary development. Int J Adv Manuf Technol 21 (9):669–677