Robot Programming using Augmented Reality

H.C. Fang, S.K. Ong and A.Y.C. Nee
Mechanical Engineering Department, Faculty of Engineering,
National University of Singapore, 9 Engineering Drive 1, Singapore 117576
{g0600952 | mpeongsk | mpeneeyc}@nus.edu.sg

Abstract

Human-robot interaction issues, especially for industrial robots, have largely been confined to finding better ways to reconfigure or program the robots. In this paper, an Augmented Reality based Robot Programming (RPAR-II) system is proposed. A virtual robot, which is a replicate of a real robot, is used in a real environment to perform and simulate the robot trajectory planning process. The RPAR-II system assists the users during the robot programming process from task planning to execution. Stereo vision-based methodologies for virtual objects registration as well as interactive device position tracking are employed in this system. Practical issues concerning the system implementation are discussed.

Key word*: Robot Programming, Augmented Reality, Robot Registration, Human-Robot Interaction.*

1. Introduction

Robots are primarily employed for carrying out programmed, repetitious tasks in order to promote productivity and efficiency. Advancements and developments in engineering, artificial intelligence, human-computer interaction (HCI) and Human-Robot Interaction (HRI) in recent years have collectively provided strong technical supports that allow the design of more advanced robotic systems for a wider range of applications, *e.g.*, complex industrial robots in production lines, robots for underwater tasks, robots for security and defense, and robots for research and teaching purposes.

In a robotic system, hardware components, such as mechanical parts, actuators, controller, *etc.*, would affect the level of accuracy in task execution. The software platform, on the other hand, is an interface that enables HRI and conveys the human intention on how certain tasks should be executed by the robot. HRI issues are largely confined to identifying new ways to achieve robot programming. Traditional programming approaches are either unintuitive or time-consuming [1-2]. In lead-through programming, a teaching pendant is used as a HRI tool. This is only applicable for certain groups of applications, *e.g.*, pick and place operations (PPO), where there are lower accuracy requirements. Walk-through programming requires the operator to be present within the working envelope when the robot is in operations, posing considerable safety concerns.

HRI researchers are constantly looking for better user interfaces for robot programming as more enabling technologies are made available [3]. For examples, Virtual reality (VR) has been proven useful in tele-robotics [4-6], where tasks are completed intuitively as if the operator is present at the remote working environment. The main constraint in VR-based robot programming is the need to completely construct the virtual environment (VE), which requires full *a priori* knowledge of the working area and thus more computational resources. This is not suitable particularly for semi-structured or loosely structured working conditions, where there may be constant changes to the environments.

The Programming by Demonstration (PbD) approach provides the users with an intuitive and fast way for robot programming through manually performing the tasks and leaving the robot to observe, follow and learn in real-time [7]. It has successfully shifted the burden of robot programming from robot experts to task experts. There are constraints in this approach, *e.g.*, jerks and jitters in the human demonstrations [8], sub-optimalities during the multiple demonstrations [9-10], *etc.*

Recent research on robot programming is moving towards multimodal interfaces. Gestures, voices, *etc.*, are used as high-level symbolic inputs to control the robotic system [11-13]. However, methods using these forms of inputs are not robust, leading to possible improper or even incorrect task interpretations.

978-0-7695-3791-7/09 $26.00 © 2009 IEEE
DOI 10.1109/CW.2009.14

Augmented Reality (AR) is a splicing of virtual elements, mainly computer-generated graphics, onto the real world so that both can be perceived by the users at the same time. When applied in robot programming, AR offers the possibility to visualize the motions and trajectories of a robot overlaid on the real environment, enabling the users to intuitively interact with the spatial information [14]. One of the first AR robotic applications is in tele-robotics [15]. The AR Viewer [16] provides the operators with instantaneous visual feedback and various visualization and simulation options. From the visual inputs, the robot motions and paths can be simulated before executing the tasks using the real robot. The AR Viewer provides two methods for robot tracking, marker-based optical tracking and mechanical tracking. Another innovative prototype has been developed for programming surface-related processing tasks, such as robotic laser welding [14, 17].

In this paper, an AR-based Robot Programming (RPAR-II) system is proposed to assist users in robot programming for PPO through providing an intuitive human-robot interface to enrich the interactions between the operators and the robot. In this system, a virtual robot model is augmented onto the working environment and overlaid on the real robot. An interaction device, which can be tracked through tracking the attached marker-cube, is employed to guide the virtual robot for trajectory planning. The rest of the paper is organized as follows: Section 2 presents the architecture and components of the RPAR-II system. Section 3 introduces the vision-based methodologies for robot registration in the augmented environment. Section 4 presents a method of using a marker-cube used to guide the virtual robot for virtual operations within the work place. Section 5 discusses the system implementation issues; and Section 6 gives the conclusion.

2. RPAR-II System Architecture

The hardware components of RPAR-II are shown in Figure 1, and include a Scorbot-ER VII type manipulator arm, an electrical gripper (end-effector (EE)), a robot controller, a desktop-PC, a desktop-based display, and a stereo camera. The system architecture is shown in Figure 2.

2.1. Augmented Environment

In the RPAP-II system, the augmented environment consists of the physical entities that exist in the robot working area (or within the robot operating range), such as the robot manipulator, the working platform, tools, work pieces, *etc.*, a virtual robot model to replicate the real robot, and the virtual models of the objects to be "grasped" and "released" for the PPO. An interactive tool is used to facilitate the interaction between the user and the virtual robot. A monitor-based display scheme is employed to provide a broader field of view (FOV), to allow the users to be able to visualize the entire operating range of the robot.

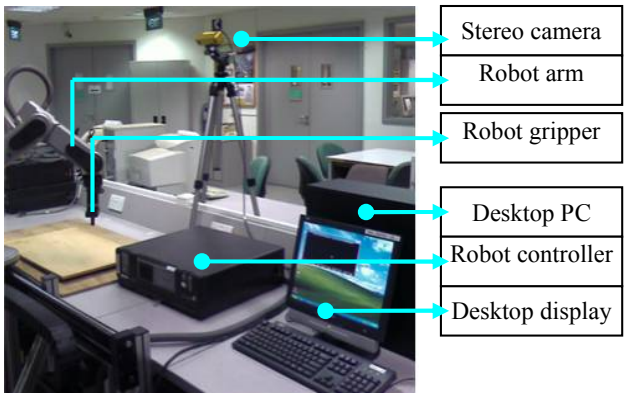


Figure 1. RPAR-II setup

A virtual model that replicates the configuration of the Scorbot arm in its "parked home" position is shown in Figure 3. The geometry of each arm is modeled separately and linked together through aligning the coordinate system of each link. The virtual robot can be scaled up or down for different robot configurations. The links are built with both functional and geometric features so as to enhance the visualization effects. In addition, different types of EEs have different swept models for collision detection during the robot programming process.

2.2. Operating Space and Solution Space

The operating space of a robot is the space that the EE could reach with a full stretch of each link of the robot, and it would be changed when different EEs are used. The solution space, however, is task-dependent and can be defined as a sub-set of the operating space. In the RPAR-II system, an interaction device is used to guide the virtual robot, so as to demonstrate a path for performing a task to the virtual robot and generate a collision-free volume (CFV) for the virtual robot. The steps to generate a solution space can be achieved through precise registration and tracking of the interaction device and other physical entities in the real environment. Registration and tracking are crucial so that the virtual robot and virtual assembly components

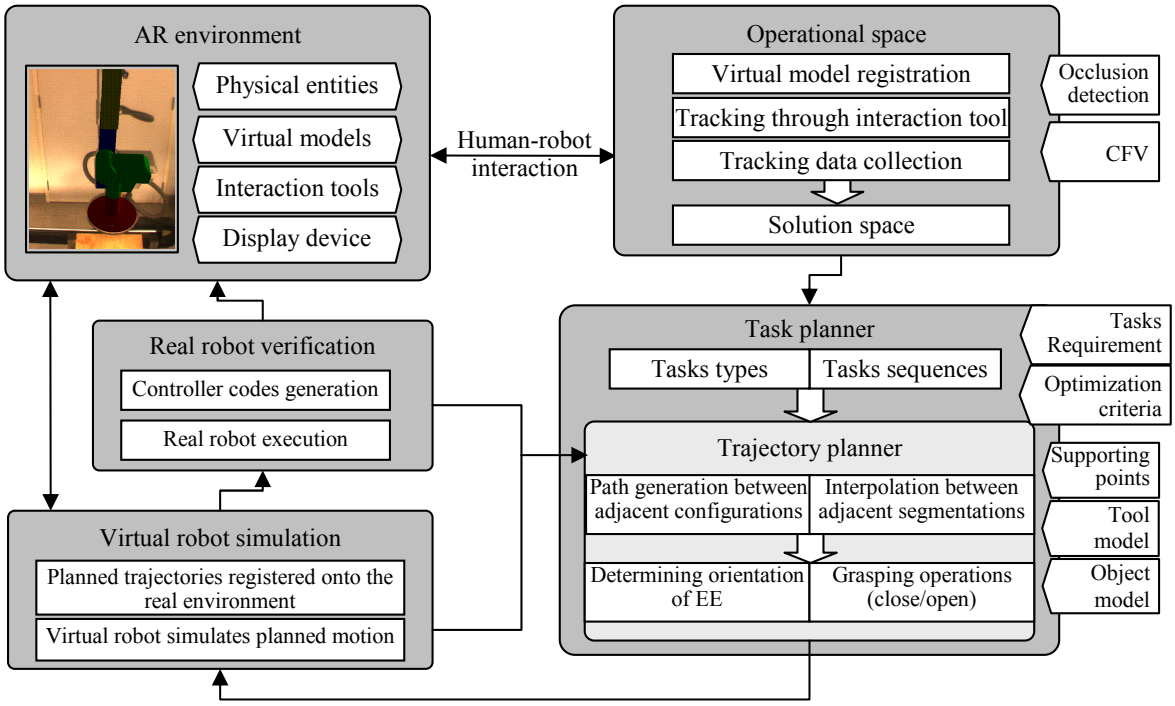
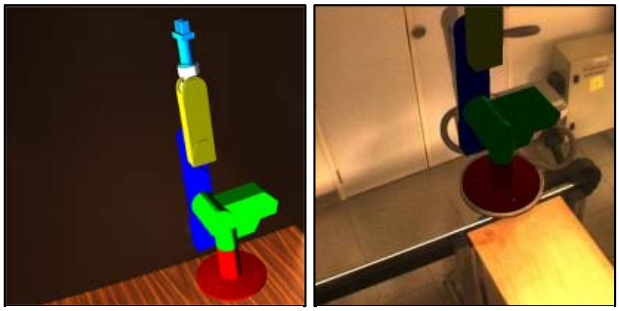


Figure 2. Architecture of the RPAR-II system

can be correctly augmented onto the real work cell. The outputs from registration and tracking are inputs to the robot trajectory planning process.



a b

Figure 3. Virtual model of SCORBOT ER-VII: (a) scaled-down model; and (b) full-scaled model

2.3. Tasks Planning

It is common for small and medium enterprises to own a robotic system with multiple types of EEs for various tasks. In our system, different robotic tasks are represented as different task-level descriptions. Given such descriptions, the task requirements can be retrieved from the database, and the task optimization criteria are hereby determined, *e.g.*, shortest path, minimum actuator effort, minimum time, *etc*.

The solution space for a task can be obtained from user demonstrations. As a user can see the actual environment where the robot will operate when he demonstrates a robot path, the path that is generated from the user demonstrations will be collision-free within the operating space of the robot. However, this CFV sub-space generated from demonstrations may not contain the globally optimal path for a given task. Hence, optimization is required to obtain an optimal solution from the feasible solutions within the CFV.

During path planning, the operating time, the acceleration, and the working load should comply with the operating specifications of the robot. The output from the robot trajectory planning is a recorded profile that contains the positions of the discrete points along the trajectory and the orientation and state of the EE. The path can be refined based on certain criteria.

2.4. Path Simulation and Verification

Once an optimal path has been determined, it will be translated into robot controller codes to operate the real robot for the planning verification. Hence, the real robot will "learn" and be able to perform the same task and path as demonstrated by the user. A simulation of this path prior to the translation would be carried out using a virtual robot. This simulated path can be correctly registered in the real environment, superimposed over the real robot, and viewed using a

monitor-based display as the working range of the real robot is within the FOV of the camera.

3. Virtual Robot Registration

Registration and tracking are the two essential components in an AR-based system. In the RPAR-II system, a vision-based method is adopted to register a virtual robot onto the real robot, as shown in Figure 4. The camera is maintained at a fixed position and orientation. Thus, the transformation between the camera and the base of the robot is invariant. Hence, robot registration is only required to be performed once using a chessboard. The transformation matrix between the camera and the base of robot can be stored, and retrieved when the chessboard is removed from the FOV of the camera. Figure 5 shows a virtual robot registered and augmented over the real robot in the real environment.

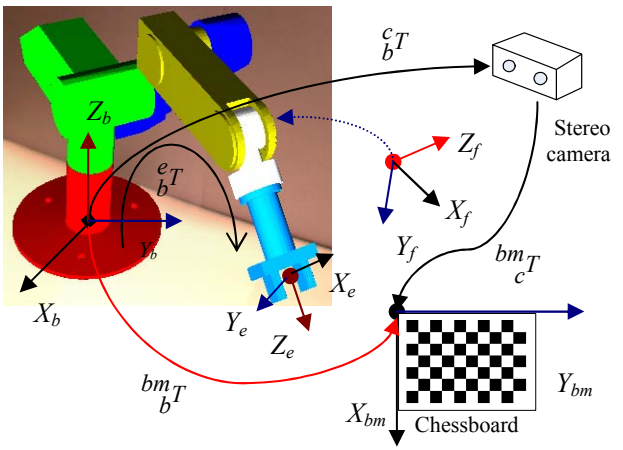


Figure 4. Relationships between the different coordinate systems

A critical issue in registration is occlusion. It occurs when a real object overlaps with a virtual object in the FOV of the camera. A straightforward way to deal with this problem is to use the depth information of the real and virtual objects. Zhu and Pan [18] reported a fast occlusion registration method where two monocular cameras are used for depth estimation.

In the RPAR-II system, a stereo camera is used for tracking and registration. Camera calibration is carried out to obtain the intrinsic parameters of the camera before it is used in the system. The camera is located at a fixed position with respect to the base of the robot. To enhance the visualization of the complete robot programming process, the camera should be able to cover the operating space of the robot, or at least the work space for a given task.

4. Human-virtual Robot Interaction

4.1. Marker-cube

In order to guide the virtual robot in the RPAR-II system, a marker-cube is used as the human-virtual robot interaction device, as shown in Figure 5. The use of a marker-cube to facilitate robot planning has been reported by Bischoff and Kurth [16], where a marker-cube with six planar markers is attached to the EE of a real robot. It provides users with instantaneous visual feedback of the movements of the EE. For the marker-cube used in the RPAR-II system, four different planar markers are attached to the faces of the cube. The advantage of using multiple markers over a single marker is that the cube can be tracked even if it has undertaken a relatively large rotation, *e.g.*, 180°, as there is a known relationship between each planar marker and the coordinate frame of the cube.

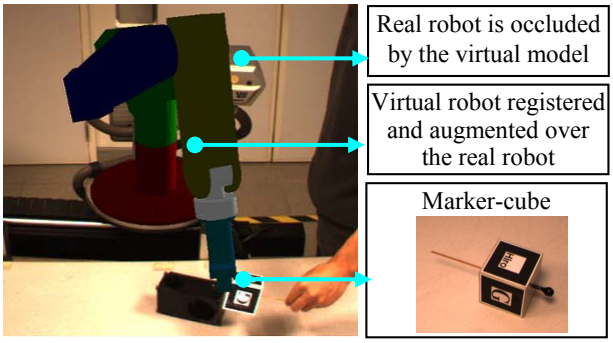


Figure 5. A virtual robot is augmented over a real robot, and guided by the marker-cube

With proper manipulation of the marker-cube, the camera is able to capture at least one marker in each frame, which is sufficient to obtain the position and orientation of the cube. The marker with the largest confidence ratio, a parameter reflecting how well the recognized marker matches with its pattern, is selected to define the coordinate system of the marker-cube. Figure 5 illustrates using a marker-cube to guide a virtual robot in the RPAR-II system.

Each planar marker on the marker-cube defines a single virtual world (corresponding to one coordinate system). Thus, to define the coordinate system of the cube, calibration is required. The procedure can be summarized as follows: (1) select a dominant marker, and define the coordinate system of the marker-cube

with reference to this marker; (2) obtain the transformation from a recognized 2D marker to the dominant marker, so as to obtain the coordinate system of the cube. After the calibration, the pose of the cube can be smoothly tracked. The jerks in the reference frame of the cube, which may be caused by the alternation of the recognized markers, are reduced.

4.2. CFV Generation and Intermediate Poses Selection

In the trajectory planning process, the marker-cube can be used to lead the virtual robot model through moving it within the work space of the real robot to generate a CFV. Figure 6(a) illustrates the CFV generation process where a virtual sphere is attached to the marker-cube. By recording the position of the centre of the sphere, which has a predefined radius, the CFV can be constructed. This concept has been reported by Chong *et al.* [19], where a single planar marker is used. By carefully dragging the marker-cube within the work space of the real robot, the sequential states of the virtual sphere can be recorded to generate a CFV. The representation of a CFV could be a set of N_0 virtual spheres [19]:

$$\mathrm{CFV} = \{S_i\,(c_i,\, r_0);\; i=1,\ldots,N_0\} \qquad (1)$$

where the c_i gives the centre of the sphere, r_0 is the predefined radius. Here the coordinate system of the base of the robot is used as the world coordinate system.

For PPO, the marker-cube can be used to assist the users to select the intermediate poses during trajectory planning, as shown in Figure 6(b). The intermediate poses obtained in the Cartesian space should meet two requirements: (1) they are reachable by the EE of the robot; (2) at each intermediate pose, both the grasped object and the EE should be within the CFV. As the virtual object is assumed to be rigidly attached with the gripper, the coordinate systems of the swept model and of the EE are related by a rigid transformation.

A swept model which envelops both the EE and the object would be appropriate to facilitate the CFV check, which is to determine whether the virtual object and the gripper are within the volume and hence collision-free. A bounding cylinder is adopted as a swept model in the RPAR-II system. In the Cartesian space of the robot, a bounding cylinder (*BC*) can be characterized by four parameters, namely, the radius (r), height (h), origin (o) and axis (z) of the cylinder,

$$\mathrm{BC} = BC\,(o, z, r, h) \in \mathrm{R}^3 \qquad (2)$$

Once the CFV is generated, the next step is to select intermediate points, so the bounding cylinders corresponding to these points can be defined as below:

$$\mathrm{BC} = \{BC_j\,(o_j, z_j, r, h, t_j);\; j=1,\ldots,N_p\} \in \mathrm{R}^3 \qquad (3)$$

Where r and h are the radius and height which are constant for a given bounding cylinder; o_j and z_j are the origin and axis of the cylinder corresponding to the jth intermediate pose; N_p refers to the number of intermediate poses. A newly introduced parameter t_j indicates the sequence of the selection. After the selection is done, this parameter can be reassigned to an exact time that indicates when the jth intermediate pose should be reached by the EE.

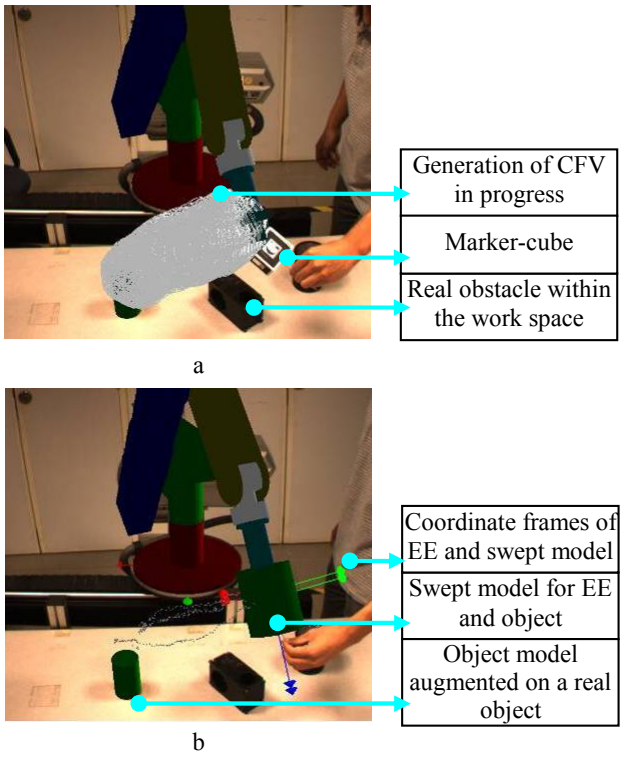


Figure 6. Trajectory path planning; (a) CFV generation using the marker-cube, and (b) selection of intermediate points

4.3. CFV check

In the current RPAR-II system, a cubic-spline interpolation method is used to generate a trajectory which passes through the selected intermediate points [20]. After the trajectory has been obtained, the following steps will be performed: (1) collision check is performed in the Cartesian space for the planned

trajectory, as the initial, final and intermediate poses in a PPO are defined in the Cartesian space and the planning process is conducted in the joint space; and (2) the maximum velocity, acceleration and torque required for each joint will be verified to be within the specifications of the robot.

The cubic interpolation implemented in the joint space can only guarantee that the Cartesian trajectory prescribed at the initial, final and intermediate poses are met. Hence although the bounding cylinder at intermediate poses are within the CFV, it is possible that the cylinder might encounter collision with the boundary of the volume when it moves along the trajectory, or even a segment of the trajectory between two prescribed points could be outside of CFV. This gives rise to the need for a thorough CFV check. Virtual robot simulation provides an intuitive interface by which the planned trajectories can be augmented on the Cartesian space and visualized by the users. These augmentations could be used as cues to guide the users to modify the intermediate poses adjacent to the collision, or to insert new pose which is able to "drag" the trajectory back into the CFV. The trajectory planning will be executed again with the updated intermediate poses. The process will be repeated until the planned trajectories are entirely within the CFV and no collision occurs between the swept model and the CFV.

5. Related Issues

5.1. Dynamics Modeling in Trajectory Planning

To plan a smooth trajectory for a real robot, dynamic constraints of the real robot should be considered during the planning process, such as the joint velocity and acceleration constraints, maximum load specifications, *etc.*, which would require a dynamics model of the robot. A flowchart for trajectory planning with consideration of dynamics modeling is illustrated in Figure 7.

Based on the procedures in Section 4, a CFV is genrerated, the initial, intermediate and end points are selected, and collision checks are performed during the simulation. When a planned trajectory is not satisfactory, the entire process will be repeated. During trajectory verification, a check is carried out to determine whether the real robot is able to follow the planned path in order to to adjust and fine tune the kinematics and dynamics models of the virtual robot if the planning and simulation process needs to be repeated.

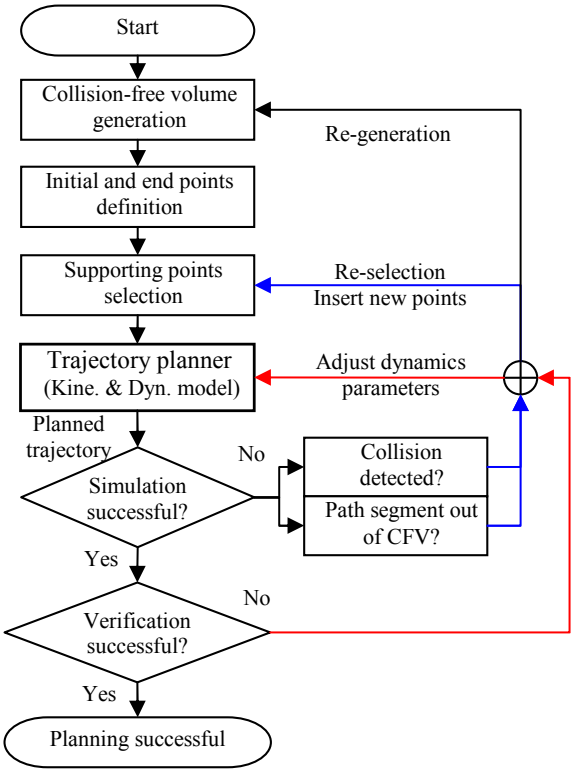


Figure 7. Flowchart of trajectory planning with consideration of dynamics modeling

Even though the robot dynamic characteristics are taken into account during the trajectory planning process, the performance of the controller might still fall short of the desired requirements, such as overshooting of the arm from a desired trajectory [21]. The trajectory will be translated into controller codes to operate the real robot. If the movement of the real robot does not coincide with the simulation performed, the dynamics model of the robot will be refined through adjusting the dynamics parameters, and the trajectory planning process will be performed again considering these updated parameters. In the RPAR-II system, there will be iterations between the planning, simulation and real robot verification, which are used to tune the parameters of the dynamics model (*e.g.*, motor gear ratio, frictional coefficients, *etc.*). The system will provide an interface that enables intuitive visualization options and facilitates the interactions (alternations) between the planning process and simulation.

5.2. Orientation Selection of EE

In the RPAR-II system, the user leads a virtual robot to demonstrate a path from an initial position to a

target position, For PPO, the procedure can be divided into five steps: (1) move the EE of the virtual robot along a path to the grasping position, (2) adjust the orientation of the EE to ensure that the gripper is able to grasp an object, (3) grasp a virtual object model, (4) move the virtual object to the releasing position, and (5) release the virtual object in a certain orientation. The EE may have to be oriented at different angles in each of these steps. In step (1), the EE is set at a fixed angle to the robot base plane to avoid collision with the obstacles along the path. In step (2), the operator guides the EE during the grasping operation to pick up a virtual object. In step (3), the EE has to be maintained in a fixed orientation. If the grasping operation is successful, the virtual object will be attached to the gripper rigidly. In the collision detection stage, normally a simple swept volumes, such as cuboids, cylinders, *etc*., will be adequate for representing the boundaries of the objects. If the swept volume of a virtual object occupies a larger space than that of the EE, the orientations of the EE and the virtual object have to be adjusted along the trajectory from the grasping position to the releasing position. By repeating these steps, the PPO can be implemented.

5.3. System Accuracy and Robot Calibration

The accuracy of the RPAR-II system is largely affected by the tracking methods used in the system. Currently a marker-based approach is used for real-time registration and tracking. Although a stereo camera is used, there are still errors in the determination of the depth information of the marker. The tracking accuracy achievable currently is roughly 10-15 mm depending on how far the camera is installed from the robot. This means that the current system can only be used in general "pick and place" operations with relatively lower accuracy requirement.

Another source of error that affects the level of accuracy of this system is caused by the hardware of the robot system. From this point of view, the robot should be able to compensate these intrinsic errors prior to task execution through robot calibration. In the RPAR-II system, the positioning accuracy is most crucial in order to achieve good performance. Thus, robot calibration has to be performed to identify the parameters that influence the static positioning characteristics of the manipulator. The most commonly used methods are vision-based methods [22, 23]. A more accurate calibration method is to use a laser tracker [24]. The calibration method adopted in the RPAR-II system is vision-based where a calibration device is rigidly attached to the EE. Together they will be mounted onto the flange of the robot. The calibration device consists of a stereo camera and an Inertial Measurement Unit (IMU) for tracking the position, orientation and movements of the robot arms. A chessboard calibration device is used to represent the exact positions of the point clusters used for the calibration.

6. Conclusions and Future Works

In this paper, an AR-based robot programming system is proposed to assist a user for both on-site and remote robot programming, as well as robot selection and installation. In this system, a virtual robot, which is a replicate of a real robot, is used to perform and simulate the task planning process. An interaction device is used to allow the users to interact with the virtual robot during robot programming process. Related issues concerning system implementation are discussed and an iterative approach for dynamic modeling of the robot is being developed to facilitate the trajectory planning in the RPAR-II system.

For remote operations, two or more cameras will need to be installed at the remote site to acquire information of the remote working environment. An alternative interaction device (such as a joystick or a PHANToM) will be needed to replace the marker-cube to enable the operators to manipulate the virtual robot that is augmented on the remote site to carry out the planning tasks. This system can be used to assist the operator in the selection of an optimal location of a robot prior to its final installation, and determination of a suitable robot configuration among a number of available robots.

To enhance the accuracy of this system, a more accurate and flexible tracking method will be explored to enhance the HRI and improve the system performance.

References

[1] J.N. Pires, A., Loureiro, T. Godinho, P. Ferreira, B. Fernando and F. Morgado, "Object Oriented and Distributed Software Applied to Industrial Robotic Welding", *Industrial Robot: An International Journal,* 2002, 29(2): 149-161.

[2] J.N. Pires, T. Godinho and P. Ferreira, "CAD Interface for Automatic Robot Welding Programming", *Industrial Robot: An International Journal,* 2004, 31(1): 71-76.

[3] S. Thrun, "Toward a framework for human-robot interaction", *Human-Computer Interaction*, 2004, 19(1/2): 9-24.

[4] E. Freund and J. Rossmann, "Projective virtual reality: Bridging the gap between virtual reality and robotics", *IEEE Transactions on Robotics and Automation*, 1999, 15(3): 411-422.

[5] E. Freund and J. Rossmann, "Projective virtual reality as a basis for on-line control of complex systems-not only-over the Internet", *Journal of Robotic Systems*, 2005, 22(3): 147-155.

[6] E. Freund, K. Hoffmann and J. Rossmann, "Application of automatic action planning for several work cells to the German ETS-VII space robotics experiments", *Proceedings of IEEE International Conference on Robotics and Automation*, 2000, **2**, pp. 1239-1244.

[7] A. Billard, S. Calinon, R.Dillmann and S. Schaal, "Robot Programming by Demonstration", In: B. Siciliano and O. Khatib, (Eds.) *Springer Handbook of Robotics*, Berlin, London: Springer, 2007, pp.1371-1394.

[8] C.G. Atkeson and S. Schaal, "Learning Tasks from a Single Demonstration", *Proceedings of IEEE International Conference on Robotics and Automation*, 1997, **2**, pp.1706-1712.

[9] M. Kaiser, H. Friedrich and R. Dillmann, "Obtaining Good Performance from a Bad Teacher", *Proceedings of International Conference on Machine Learning, Workshop on Programming by Demonstration*, Tahoe City, California, USA, Jul. 1995.

[10] J. Chen and A. Zelinsky, "Programming by Demonstration: Coping with Suboptimal Teaching Actions", *The International Journal of Robotics Research*, 2003, 22(5): 299-319.

[11] S. Iba, C.J.J. Paredis and P.K. Khosla, "Interactive multimodal robot programming", *The International Journal of Robotics and Research*, 2005, 24(1): 83-104.

[12] R. Martin, P.J. Sanz, P. Nebot and R. Wirz, "A multimodal interface to control a robot arm via the web: a case study on remote programming*", IEEE Transactions on Industrial Electronics*, 2005, 52(6): 1506-1520.

[13] J.N. Pires, "Robot-by-voice: experiments on commanding an industrial robot using the human voice", *Industrial Robot: An International Journal*, 2005, 32(6): 505-511.

[14] M.F. Zaeh and W. Vogl, "Interactive laser-projection for programming industrial robots", *Proceedings of the IEEE and ACM International Symposium on Mixed and Augmented Reality*. 2006, pp.125-128.

[15] A. Rastogi, P. Milgram and D. Drascic, "Tele-robotic control with stereoscopic augmented reality", *SPIE Stereoscopic Displays and Virtual Reality Systems III*, 1996, **2653**, pp.115-122.

[16] R. Bischoff and J. Kurth, "Invited talk: Concepts, Tools and Devices for Facilitating Human-Robot Interaction with Industrial Robots through Augmented Reality", *ISMAR Workshop on Industrial Augmented Reality*, 2006.

[17] G. Reinhart, U. Munzert and W. Vogl, "A programming system for robot-based remote-laser-welding with conventional optics", *CIRP Annals - Manufacturing Technology*. 2008, 57(1): 37-40.

[18] J.J. Zhu and Z.G. Pan, "Occlusion Registration in Video-based Augmented Reality", *Virtual-Reality Continuum and its Applications in Industry 2008*, Singapore, December 8-9, 2008.

[19] J.W.S. Chong, S.K. Ong and A.Y.C. Nee, "Methodologies for immersive robot programming in an augmented reality environment", *International Journal of Virtual Reality*, 2007, 6(1): 69-79.

[20] J. Angeles, Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms. 3rd ed. Springer, 2007.

[21] J.J. Craig, Introduction to Robotics, Mechanics and Control. NY: Pearson Education Inc, 2005.

[22] J.M.S.T. Motta, G.C. de Carvalho and R.S. McMaster, "Robot Calibration Using a 3D Vision-based Measurement System with a Single Camera", *Robotics and Computer-Integrated Manufacturing*. 2001, 17(6): 487-497.

[23] H. Malm and A. Heyden, "Extensions of Plane-Based Calibration to the Case of Translational Motion in a Robot Vision Setting", *IEEE Transactions on Robotics*, 2006, 22(2): 322-333.

[24] Y. Bai, H.Q. Zhuang and Z.S. Roth, "Experiment Study of PUMA Robot Calibration Using a Laser Tracking System", *Proceedings of IEEE International Workshop on Soft Computing in Industrial Applications*, Binghamton, NY, Jun. 23-25, 2003, pp.139-144.