ISBN 978-1-84626-xxx-x
Proceedings of 2011 International Conference on Optimization of the Robots and Manipulators (OPTIROB 2011)
Sinaia, Romania, 26-28 Mai, 2011, pp. xxx-xxx

Off-line programming of industrial robots using co-located environments

Florin Gîrbacia, Mihai Duguleană and Adrian Stavăr

Transilvania University of Brasov

Abstract. This paper presents a methodology and a prototype system for off-line programming of an industrial robot using augmented reality technology. The system allows to control a virtual model of the industrial robot co-located in the real environment, planning for collision-free paths, generate robot program and simulate the robot actions before the real robot perform the task. The advantage of this system is use of inexpensive equipment for intuitive off-line programming of an industrial robot.

Keywords: Augmented Reality, Robot Programming, Co-location.

1. Introduction

As the computer technology is providing a crucial advantage on the industrial world competition more than ever, the complexity of the industrial process simulations increases dramatically. Industrial robots are important elements in automated manufacturing systems used for carry out manufacturing operations. Traditionally a robot program is generate by teaching the robot one point at a time using a teach pendant. This process is time consuming and also errors can come out during the teaching process.

In order to overcome these limitations, off-line programming systems can be used. These kinds of systems are mainly based on Virtual Reality (VR) technologies [2], [9]. The restrictions encountered require the necessity to create prior simulation of a virtual world which has to integrate all the information related to the real working environment. Therefore, the applications and operational environment are limited to those in which a virtual model of the working environment can be reconstructed.

Augmented Reality (AR) is a relative new research direction that allows creation of an interactive virtual space embedded into the physical word. Unlike Virtual Reality (VR) systems, in which users are completely immersed in the virtual environment, AR users see the virtual objects and the real world coexisting in the same space (co-located). It is the goal of AR to supplement reality rather than completely replace it as in conventional VR applications[1].The AR technology provides useful information about the environment, enhancing perception and interaction with the real world. The user is able to manipulate both real objects, and the virtual ones.

AR technologies are now used in various application fields like medicine, manufacturing, military, education and entertainment etc. [1]. However, irrespective of the application, an augmented reality system uses the same components like: computer graphics, image processing, and computer visualization. One of the main fields of application for augmented reality technologies is robotics. By using augmented reality technologies for the programming of robots the user get a better visual feedback that will improve efficiency of the process and the necessity to generate the virtual representation of the working environment is eliminated.

Nowadays, there are several applications of using augmented reality in robot programming [3], [4], [5], [6], [7], [8]. In [8] a typical application is described in which augmented reality technologies are used for control of a virtual robot overlaid in the real environment, plan the robot‘s actions and predicts the effects of

the manipulation. Also in [6] a standalone augmented reality pilot system is presented allowing an operator to program robot waypoints and process specific events related to painting applications.

This paper present ongoing research carried out at Transilvania University of Brasov with the aim to develop a methodology and an operator interface based on augmented reality technology that enables off-line programming of an industrial robot. The prototype application presented allows to a user to control a virtual model of an industrial robot co-located in the real environment, generate robot program and simulate the robot actions before the real robot perform the tasks.

2. Methodology for industrial robots off-line programming using co-located environments

The methodology to generate robot programs using co-located AR environments involves the next steps:

- Modeling of the virtual robot 3D model: consist in generation of the virtual robot model using specialized Computer Aided Design software (for example Solidworks).
- 3D virtual robot model data conversion: the virtual model can't be loaded in the AR software because there isn't standard interoperability procedure. Therefore this step consists in extracting the entire geometric data of CAD model and conversion of standard CAD file to an appropriate common exchange file format (for example 3ds, VRML , X3D etc.) that can be loaded by general AR dedicated framework.
- Development of the inverse kinematics robot model. There are 2 different types of methods used for synthesizing the motion of linked bodies Kinematics: Forward Kinematics (FK) and Inverse Kinematics. From the AR point of view the Inverse Kinematics is more suited to be used because provides direct control over the position of the end-effector by solving each of the joint angles from the kinematic chain [10]. There are 2 different approaches to IK problem: analytical and numerical. Analytical methods attempt to give an exact solution by directly inverting the FK equations. This is only possible on relatively simple chains. Numerical methods use approximation and iteration to converge to a solution. These tend to have a more general purpose but require most of the times computational resources [11].
- Virtual robot model integration in AR software consists in the generation of a configuration file that contain the marker tracking setup and the 3D scene file.
- Generation of a unique marker consists in the generation of a unique fiducial marker for each 3D CAD model and storage the marker shape data in the AR software.
- Installation of a video camera device in the real robot working environment and positioning of the AR marker in the place where virtual robot will be rendered.
- Starting of the AR software and registration of the co-located virtual robot model corresponding to the real robot characteristics.
- Simulation of the virtual robot actions by using interaction devices (for example a keyboard) and generation of robot instruction by storage of robot control points.

3. Architecture of the developed AR prototype system

A prototype AR system was developed to demonstrate the methodology presented above. The system is designed to offer an intuitive interface for programming of an industrial robot. In the developed system, a virtual robot model is co-located with a real industrial robot. Interaction devices are used to allow the users to interact with the virtual robot during robot programming process. The main structure of our application using the augmented reality technologies for industrial robots programming is shown on figure 1. The following hardware components are used: an ABB IRR 1600 robot with 6 DOF integrated in a manufacturing robotic cell, robot controller, wearable computer, interaction devices (joystick and keyboard), visualization devices (monitor or HMD), video camera. Components of the system (interaction devices, head-mounted display and robot system) are connected to a wearable computer running algorithms for detection of markers, 3D position and orientation determination, rendering of virtual robot, as well as generation of the robot programs. In order to augment human's visual sense, a physical display device is used allowing combining real and virtual images and present them to the user. Many forms of video display

can be used: Head Mounted Displays (HMD), portable displays (like PDA), monitors and projectors. HMD is a common choice for AR because it is portable, and it is placed directly on the users' visual range. In this research, in a first phase it was used a Trivisio stereo video see-through AR display. The problem occurred when using this device was the necessity to register virtual robot onto the real robot when the user moves, because the transformation matrix between the HMD camera and the base of the robot was changed. From this reason it was used a video camera with a fixed position for generate the video stream and a LCD monitor for visualisation of co-located environment.

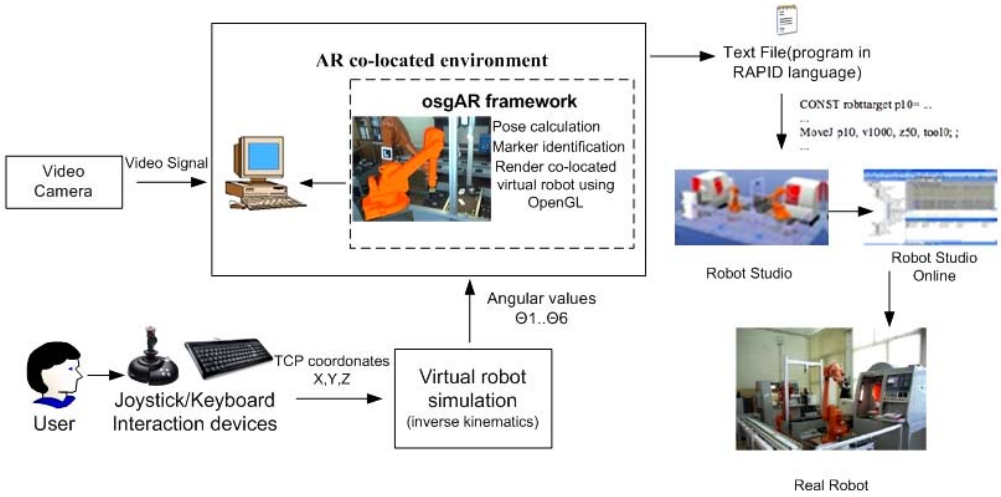

Fig. 1: Overall architecture of the AR prototype system

The software architecture is discomposed in an AR co-location software module, an interaction device module, a simulation of virtual robot module and a module for generation of robot program commands using RAPID language. The prototype system has been developed using OOP with Visual C++ compiler for Windows.

The AR software module allow identification of the square markers, determine 3D position and orientation of identified markers in order to align the virtual robot onto the real one, register the 3D virtual robot in the real environment, and simulate the movement commands of the real robot (fig. 3). The code written for the AR software module is based on an open source library called osgAR [12]. The interaction device module allows the user to send motion command of the TCP by using a keyboard device or a Microsoft Sidewinder Joystick device. The simulation of virtual robot module enables calculation of the inverse kinematics of the ABB virtual robot.

The virtual robot model can be rendered in the co-located environment using OpenSceneGraph framework integrated in osgAR. The advantage of using this library is possibility to integrate various neutral graphical formats of virtual objects. In this research the .3ds graphics format was used to render the virtual robot model. The virtual robot model was generated using Solidworks CAD Software and converted into the .3ds neutral format.

With the purpose of registration of virtual robot model onto the real robot a square marker was attached in the real environment. The osgAR API includes computer vision functions that allow analyzing each video frame and identifying markers. The video frames provided by the video camera are then processed by several osgAR routines which identify every visible marker through a template matching algorithm, computes 3D position and orientation for every visible marker and overlay virtual objects on the correct position computed in relation with the markers. Each marker in the system has a black border and a unique symbol inside the black frame. The osgAR framework uses the unique symbol to identify the markers. Before using a marker with osgAR, the osgAR marker training application needs to store the marker's shape data through an individual identification process for each marker. This process generates a template used by osgAR as a reference for comparison. Each marker from the video stream is processed in order to get a standard normalized symbol to be compared against the list of templates. In this way osgAR gets the camera

transformation matrix as well as the marker's id. The 3D position and orientation are used to overlay virtual objects in the real environment. The user can modify scale, 3D position and orientation relative to the camera transformation matrix by using keyboard. In this way the registration of the virtual robot onto the real robot (fig 2) can be adjusted.

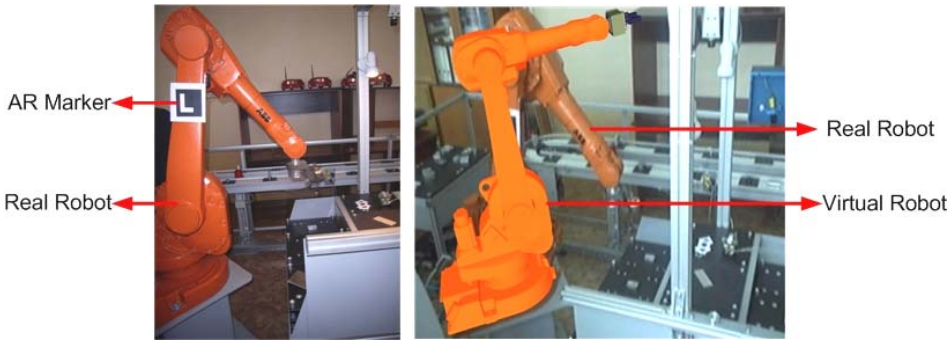

Fig. 2: The co-located ABB IR 1600 virtual robot

4. Preliminary Results

To evaluate the system behaviour an experiment was conducted with the purpose to simulate a typical pick-and-place operation. In a fist step, the virtual robot was registered onto the real ABB IR1600 robot. The user controls the co-located virtual robot arm by pressing keyboard functions or a moving the joystick which increase with an increment the position/orientation of the TCP or send a grasp command. By using these interaction devices, the operator will jog the robot in the good position to grasp the part on its work-plane. When the robot reaches the expected position, the user presses a button to save robot configuration in the text file containing the program. The application also calculates the quaternion and other data needed to write the corresponding program. To be understandable by the ABB controllers, the generated text file needs to be written using a special language called RAPID (specific for ABB robots). Using the module for generation of robot program, data from the text file is converted in the RAPID robot language. The RAPID program is composed by a module, containing several layouts (position, quaternion, quadrants and constants) and several procedures. Each procedure is a linear following of procedures or actions. The most common action is the MoveJ non-linear jogging command. The generated program (fig. 3) can also be tested in the RobotStudio offline environment, before being uploaded on the real controller and used.

5. Conclusions

In this paper, it was presented a methodology and prototype software system which provides a promising tool for off-line programming of an industrial robot based on co-located environments. This approach proved to be a powerful visualization tool that offer important advantages to the traditional programming methods. Another advantage is the ease and rapidity of creation, quicker revision and analysis of the robot path. Working with this system can be made without the need for expensive equipment, because only a video device and a computer are needed. However, out of these positive results the system has his limitations, since problems can come out when the intensity of the light it is weak, which make the identification of markers difficult and affects the level of tracking accuracy. Future work is focused on improve tracking accuracy using a Microsoft Kinect sensor, and detection of collision between the real working environment and virtual robot.

```
MODULE MainModule
    CONST robtarget p10:=[[-422.947,-796.732,698.407],[0.5,0.5,0.5,-0.5],[0,0,-1,0],[9E+09,9E+09,9E+0
    CONST robtarget p20:=[[-422.612,-791.812,700.151],[0.5,0.5,0.5,-0.5],[0,0,-1,0],[9E+09,9E+09,9E+0
    CONST robtarget p30:=[[-423.642,-790.241,699.229],[0.5,0.5,0.5,-0.5],[0,0,-1,0],[9E+09,9E+09,9E+0
    CONST robtarget p40:=[[500.011,504.535,679.428],[0.5,0.5,0.5,-0.5],[-1,0,0,0],[9E+09,9E+09,9E+09,
    CONST robtarget p50:=[[495.112,500.058,678.661],[0.5,0.5,0.5,-0.5],[-1,0,0,0],[9E+09,9E+09,9E+09,
    CONST robtarget p90:=[[176.442,6.81055,1226.15],[0.5,0.5,0.5,-0.5],[-1,0,0,0],[9E+09,9E+09,9E+09,
    PROC main()
        MoveJ p10, v1000, z50, tool0;
        MoveJ p20, v1000, z50, tool0;
        MoveJ p30, v1000, z50, tool0;
        WaitTime 2;
        Closegrip;
        WaitTime 2;
        MoveJ p40, v1000, z50, tool0;
        MoveJ p50, v1000, z50, tool0;
        WaitTime 2;
        Opengrip;
        WaitTime 2;
        MoveJ p90, v1000, z50, tool0;
    ENDPROC

    PROC Opengrip()
        WaitTime 0.2;
        Set DoGripOn;
        WaitTime 0.2;
    ENDPROC

    PROC Closegrip()
        WaitTime 0.2;
        Reset DoGripOn;
        WaitTime 0.2;
    ENDPROC

ENDMODULE
```

Fig. 3: Simulation of a pick-and-place operation using AR interface and the generated RAPID program

6. Acknowledgements

This paper is supported by the Sectoral Operational Programme Human Resources Development (SOP HRD), financed from the European Social Fund and by the Romanian Government under the contract number POSDRU 89/1.5/S/59323.

7. References

[1] R. Azuma. A survey of augmented reality. *Presence: Teleoperators and Virtual Environments.*1997, **6**(4): 355-385.

[2] C.J. Chen, S.K. Ong, A.Y.C. Nee, Y.Q. Zhou. Interactive Path Planning of a Haptic-based Virtual Robot Arm. *International Journal of Interactive Design and Manufacturing.*2010, **4**(2):113-123.

[3] H. Fang, S. K. Ong, A. Y.-C. Nee. Robot Programming Using Augmented Reality. In: *Proceedings of the 2009 International Conference on CyberWorlds(CW '09).*2009, pp. 13-20.

[4] C.A Jara., F.A. Candelas, P. Gil, M. Fernandez, F. Torres. An Augmented reality Interface for Training Robotics Through the Web. In Basanez L., Suarez R., Rosell J., (eds). *Proceedings of the 40th International Symposium on Robotics*. AER-ATP. 2009, pp. 189-194.

[5] S. K. Ong, J. Chong, A. Nee. A novel AR-based robot programming and path planning methodology. In: *Robotics and Computer-Integrated Manufacturing.* 2010, 26: 240–249.

[6] P. Milgram, S. Zhai, D. Drascic, J. Grodski. Applications of Augmented Reality for Human-Robot Communication. In: *Proc. of IROS'93*. 1993, pp. 1467-1476.

[7] A. Nawab, K. Chintamani, D. Ellis, G. Auner, A. Pandya. Joystick mapped Augmented Reality Cues for End-Effector controlled Tele-operated Robots. In: *Proc. of Virtual Reality Conference 200(VR '07).* IEEE 2007. pp. 263-266.

[8] T. Pettersen, J. Pretlove, C. Skourup, T. Engedal, T. Lokstad. Augmented Reality for Programming Industrial Robots. In: *Proc. of the Second IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR '03)*. 2003, pp. 319-321.

[9] N. Rodriguez, J. Jessel, P. Torguet. A Virtual Reality Tool for Teleoperation Research.In: *Virtual Reality*. 2002, 6, pp. 57-62.

[10] T. Kang. Solving Inverse Kinematics Constraint Problems for Highly Articulated Models. *Master thesis*, University of Waterloo, Ontario, Canada, 2000. Available at:http://www.cs.uwaterloo.ca/research/tr/

[11] S. Rotenberg. Inverse Kinematics. *UCSD, 2005*. PowerPoint presentation. http://graphics.ucsd.edu/courses/cse169_w05/

[12] Augmented Reality cross-platform development library. http://osgart.org