Int. J. Engng Ed. Vol. 25, No. 2, pp. 261–271, 2009
Printed in Great Britain.

0949-149X/91 $3.00+0.00

Development, Integration and Evaluation of a Web-based Virtual Robot Task Simulator in the Teaching of Robotics*

ZOE DOULGERI, NIKOS ZIKOS
Department of Electrical and Computer Engineering, Aristotle University of Thessaloniki, Greece. Email: doulgeri@eng.auth.gr, nzikos@auth.gr

A web-based virtual robot task simulator has been developed and used in the teaching of robotics through lab demonstrations and a set of lab assignments. The simulator allows a high degree of real time interaction with a virtual robot that can be commanded to perform a number of pick and place operations on virtual objects replicating an existing industrial robot manipulator. A unique characteristic of the system is that, apart from using high level robot motion commands like the real robot, it allows a simplified graphical input of target positions and orientations to define a gripper path. It also encourages interaction and collaborative problem solving. Empirical evidence shows that the effectiveness of the learning program is increased by the enhanced motivation and interest of the students and through their improved learning capabilities.

Keywords: Virtual reality, robot task simulator, teaching robotics, web-based virtual lab.

INTRODUCTION

IN ENGINEERING EDUCATION it is important to expose students to practical issues in order to develop engineering intuition and bridge the gap between theory and practice [1]. However, many difficulties exist in exposing students to sufficient practice in robotics because of class size, the limited available course time and the expensive equipment and complicated experimental setups, particularly of industrial robot manipulators. The solution to this problem relies on the adaptation of the new technologies in education. Computer-based educational tools can be used in any stage of the teaching process [2]. They are increasingly used in science and engineering; reported benefits include higher student interest rates, increased participation in course work and improved educational outcomes [3]. Furthermore, with the rapid development of the Web, distance learning and e-learning activities have been developed and promoted [4].

Web-based teaching and learning in lab-based engineering courses adopts two main approaches. The first is to provide remote operation of a physical lab setup through the internet and the second is to replace the lab's hardware by simulation software [5]. Mixed approaches have also been proposed replacing some of the lab's hardware by simulation, e.g. hardware in loop control systems may use a real controller connected to a simulated plant. Each option has its benefits and disadvantages. The web teleoperation of lab set ups provides students with the experience of operating even remotely a real device but involves issues of safety and security of the real system and the people working in the real site and issues of network limitations in relation to the data flow required by the experiment; such limitations include transmission time delays and limited network resources that affect operator telepresence and system performance or stability particularly in real time interactions. In our previous work [6, 7], a web telerobotic system has been used as a supplement to the teaching of robotics and was shown that the practical experience acquired by the students is equivalent to hands-on experience given enough familiarization time. However, the limited and static views of the real site that hamper the fast 3D perception of the real scene, the bandwidth constraints that affect the quality of the vision feedback and the need for setting up the real site appropriately for remote access and operation for successive use as well as the wearing of expensive equipment, led to the proposed virtual reality approach. In general, the use of simulated systems allows student training on simulated devices before their exposure to costly hardware; furthermore, the web-based aspect allows student access to the lab from different locations and at their own times (including nights and weekends) that would not be possible if a real device needing set up and monitoring was involved. The use of virtual reality simulations greatly enhances the realism of the experiments.

The virtual robot task simulator (VRTS) is an internet-based application for the kinematic simulation of a specific robotic manipulator interacting with its virtual environment following user

* Accepted 5 December 2008.

commands. VRTS has been integrated and used for supervised, unsupervised learning and student assessment in a robotics course taught to undergraduate electrical engineers. The objective is to provide an enhanced knowledge and understanding of the basic concepts and technologies that are involved in robotic arms and their use in handling applications. Attitude questionnaires were used to measure student reaction to the tool regarding its technical characteristics and learning objective. Student perception on the latter was compared with the results of a formal testing on a composite robot handling project using VRTS and VRTS's evaluation capabilities.

System description.

The VRTS represents virtually a real robotic site consisting of a six degrees of freedom Puma 761 equipped with a simple on/off two finger pneumatic gripper and virtual object settings consisting of a working table supporting a number of virtual objects. The system is an internet -based application that can be remotely accessed by authenticated users. The VRTS system allows motion commands to be issued to the virtual robot to perform a number of tasks that are subsequently executed by the virtual robot in real time. These tasks are mostly pick and place operations on the virtual objects of the setting. VRTS architecture is designed to communicate with more than one user over the internet. Users are classified in three role types, the operator, the viewer and the supervisor. The viewer receives a stream of position data that are delivered from the simulation engine in a fashion similar to video frames and are represented within a visualization environment. The operator is a viewer that is additionally authorized to send motion commands to the simulation engine and must be unique in a network session; more than one viewer may be connected simultaneously. The supervisor is a viewer and/or operator that has extra rights with respect to the system. It is in charge of the normal operation of the network, can change the virtual object setting and can hand operator's rights to a connected user bypassing the automatic allocation procedure.

A unique characteristic of the system is that, apart from using high level robot motion commands like the real robot, it allows a simplified graphical input of target positions and orientations to define a gripper path; it further allows both Cartesian straight line and joint interpolated motion mode between path segments as the real robot does, as well as real time motion control through a 'teach pendant'-like function.

System design and implementation

VRTS was developed following a methodology that allows implementing a virtual robot scene and various forms of interfaces for issuing steering commands and acquiring real time information regarding its position in a generic modular fashion. The methodology is generic in the sense that is independent with respect to the robot geometry, adopted kinematic model, syntax of high level robot command language, software development platforms and client server components so that it decouples modelling from implementation issues. The modular structure of the entire system, the functions performed by each module and its interconnections as well as the particular implementation choices in each one is described in the sequel.

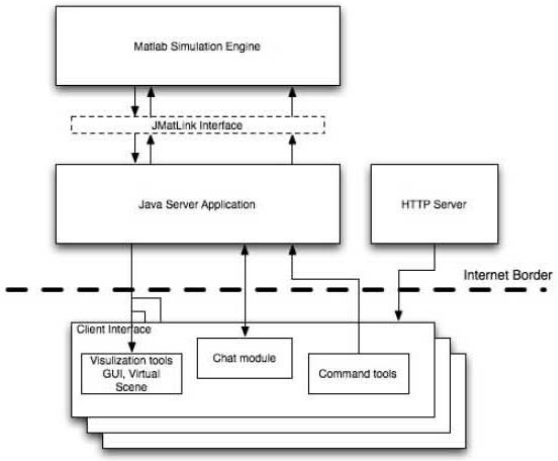

Fig. 1. General structure.

The entire software code for this work has the general structure that is shown in Figure 1.

At the heart of the software is the simulation engine of the virtual robot. This is developed in MATLAB (by Mathworks) and resides in the application server. The simulation engine takes as inputs the robot motion commands and exits a time series of robotic arm positions. The networked capabilities of the VRTS are realized by using JAVA, which is able to interact with Matlab through the 'JmatLink' package (an open source software library that interfaces JAVA with Matlab). The Java server application plays the mediator role between the clients and the simulation engine. The client on the other side runs on the client's web browser using JAVA applet technology from the http server, negotiates the connection, handles the chat facility, creates the graphical user interface, loads the virtual scene and sends the high level motion commands issued in the user interface.

- Simulation engine. The backend of this is developed in Matlab, a standard control design tool by Mathworks, in order to benefit from its ability to handle complex mathematical computations. The simulation engine is event driven rather than time driven; it consists of the trajectory generation module and the current position calculation module. Its structure is shown in Figure 2. The kinematic model adopted for the construction of these modules is fully compatible to a PUMA robotic arm with six rotary joints available in the automation and robotics lab in our department.

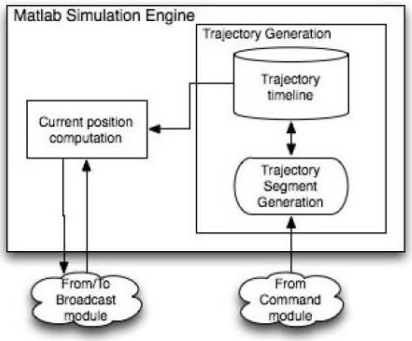

Fig. 2. Structure of the simulation engine.

- Trajectory generation module consists of a recursive trajectory segment generation function and a storage area, 'the trajectory timeline' that contains trajectory segments which have been generated under all the motion commands that have been issued so far in the simulation session. The generation function is triggered each time a new motion command issued by the operator arrives at the module and affects the contents of the trajectory timeline. It contains all the kinematic and kinetic robot parameters that are invariant under the simulation procedure. Such parameters include, for example, link lengths, maximum joint velocities and accelerations etc. The function uses the desired position and orientation of the robot's end-effector, and/or the desired motion type (motion in straight line or joint interpolated motion) and the command's time stamp (issuing time) in order to modify old trajectory segments and add one or more new segments. Trajectory segments are polynomials associated with a time interval and are represented by the vector of the polynomial coefficients as well the start and end time of the time interval the trajectory segment is valid.
- The current position computation module is a function returning the robot position at a specific time stamp. The robot position contains joint angles and tool position and orientation (Euler angles). The calculation is performed by first finding the trajectory segment that is valid at the required time using the trajectory timeline and then using this segment's polynomial coefficients and the robot kinematics.
- The JAVA server is capable to accept multiple user sessions. Its structure is shown in Figure 3 and consists of the following modules:

  a) The Broadcast module invokes the current position calculation module every 20 ms and broadcasts the received stream of position data to all the connected users. The period of 20 ms corresponds to 50 frames per second that are satisfactory when visualizing a dynamic scene.
  b) The command module receives the robot task commands issued by the operator, checks their integrity and invokes the trajectory generation module. For compatibility reasons VAL-II has been used as a reference model for the high-level syntax of motion commands and position specification. The command module also receives other type of network commands issued by the users-like connection and refresh requests, network transmission rate commands, etc.
  c) The chat facility module allows the synchronous exchange of messages among connected users that can be used for collaboration of remotely allocated teems.
  d) The administration module is used by the supervisor for functions that concern the network (e.g. determine user rights, monitoring the list of connected users, switch operator rights and other standard administration tasks).
  e) The evaluation module is based on a log module that traces each operator's commands in a simulation session in order to be used by the supervisor for evaluation purposes.

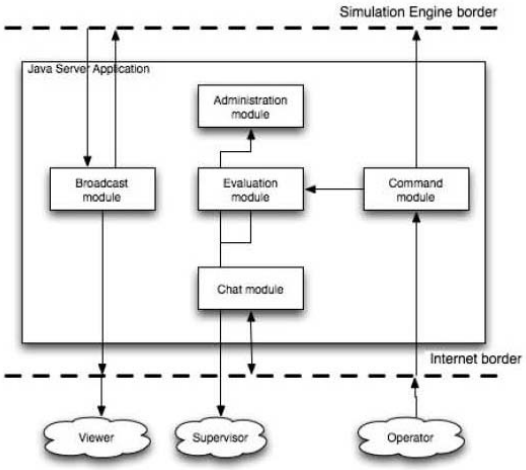

Fig. 3. Structure of the JAVA server.

- The client module runs on the client's web browser using JAVA applet technology from the http server; it acts as the front end interface to the user. It is graphics enabled in the sense that position information and/or position commands (the latter for the case of operator only) are represented within the virtual scene contained in the graphical user interface (GUI). In general, the design of a virtual scene may be implemented using any three-dimensional (3D) design applications but in this work, VRML (virtual reality model language) is used. The virtual scene includes the virtual robot, the virtual object setting, preset viewpoints and virtual lights. Its structure follows the classic structure of a 3D environment. Virtual objects are placed in the world with respect to the robot's base frame. Each robotic link is built as a separate 3D object but is located relative to its neighbour in order to

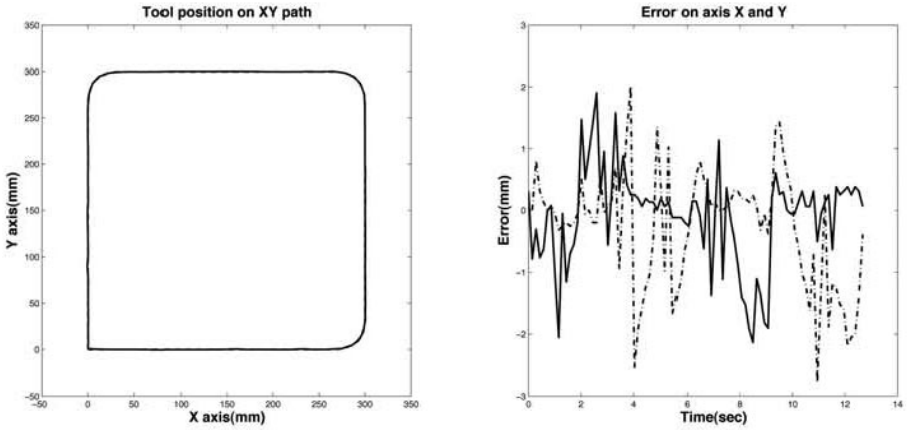

Fig. 4. Gripper path and XY coordinate errors.

decouple the 3D robot modelling from the robot simulation. Although the virtual link objects are not exact copies of the real, their kinematic values accurately copy those of the real robot. Last, the dynamic representation of the virtual scene is handled by the Java3D package, a Java plug-in provided by SUN.

- Communication protocols. Data exchange between clients and server are based on the TCP/IP protocol that ensure connection reliability. Maximum connection reliability is particularly required by the high level motion commands or motion control data. These commands make a minimum use of network resources as compared to the feedback position data from the broadcast module that is robust against limited data losses but is network resource demanding. In fact, position 'frame' packet is small (< 200 bytes) and 'frame' delivery rates of 40 fps (frames per second) have been tested without serious network overload. In any case, different delivery rates of position 'frames' are possible and can be chosen by the viewer.

System operation over the internet : the real time mode.

Because the robot simulator is used as a synchronized copy of an actual robot its time line is aligned to a real time clock, i.e. the viewer observes the evolution of a virtual robot motion as he would the real robot motion. Furthermore, causality restrictions apply, i.e. motion commands are executed only after their arrival at the simulation engine. Note, however, that the simulation engine is flexible since its trajectory timeline is independent from any clock and therefore it can be employed in other type of applications (e.g. backward simulation, fast forward, etc.). As the system is operated over the internet, network transmission time delays may occur, which in relation to the robot task speed, may adversely affect timely observation of the robot task evolu-

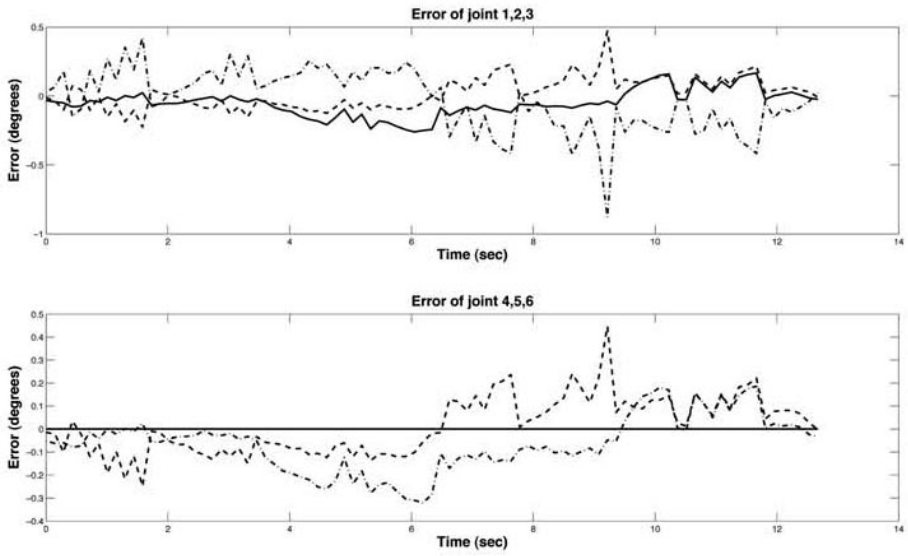

Fig. 5. Joint errors.

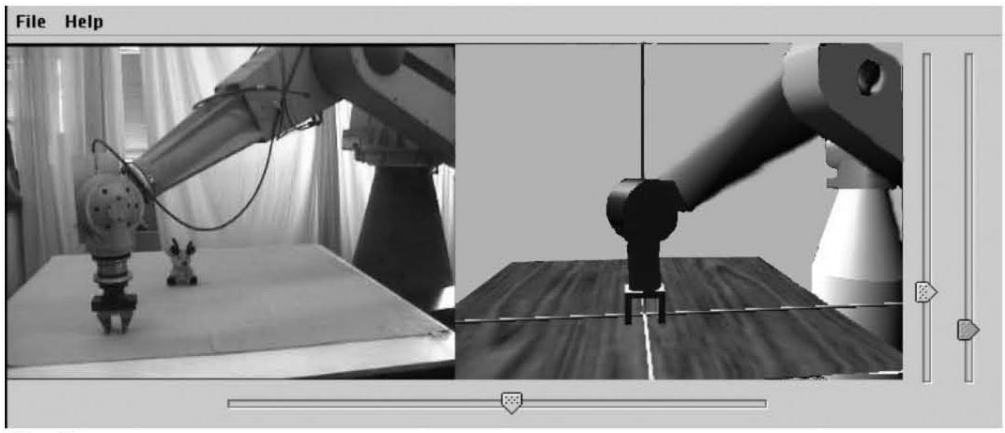

Fig. 6. Visual validation (VVT).

tion and the timely user response. In a virtual environment, most problems arise when the operator is streaming motion-control data based on the visual feedback obtained (such a motion-control mode emulates a real robot's control by the teach pendant). Then collision with virtual objects or undesired final gripper positions is possible, caused by lack of instantaneous information on the robot position and by the lack of instantaneous response from the robot operator. In the case of high level motion commands, transmission delays may result in the delayed execution of a motion command that would affect the outcome of the trajectory modification.

System validation

Validation of the VRTS was made regarding the correspondence of the virtual robot kinematic behaviour with respect to that of its real counterpart, i.e. the industrial PUMA manipulator. A motion scenario consisting of various high level motion commands was given to both the virtual and real robot in order to follow a closed path in the xy plane via intermediate positions. Position data from the resulting end effector and joint motion trajectories were collected and compared. Data collection in both cases involved the sampling of position data associated with a time stamp of the sampling moment. Figures 4 and 5 show the closed path of both the virtual and real robot, the XY coordinate error in mm and the joint errors in degrees; the mean square error is 0.61 and 0.94 mm for the X and Y-coordinate respectively and does not exceed 2.5 mm while joint errors are in the range of 10^{-2} degrees.

Furthermore, a visual validation tool (VVT) for confirmation of the virtual robot's replicated operation equivalent to the real one has been

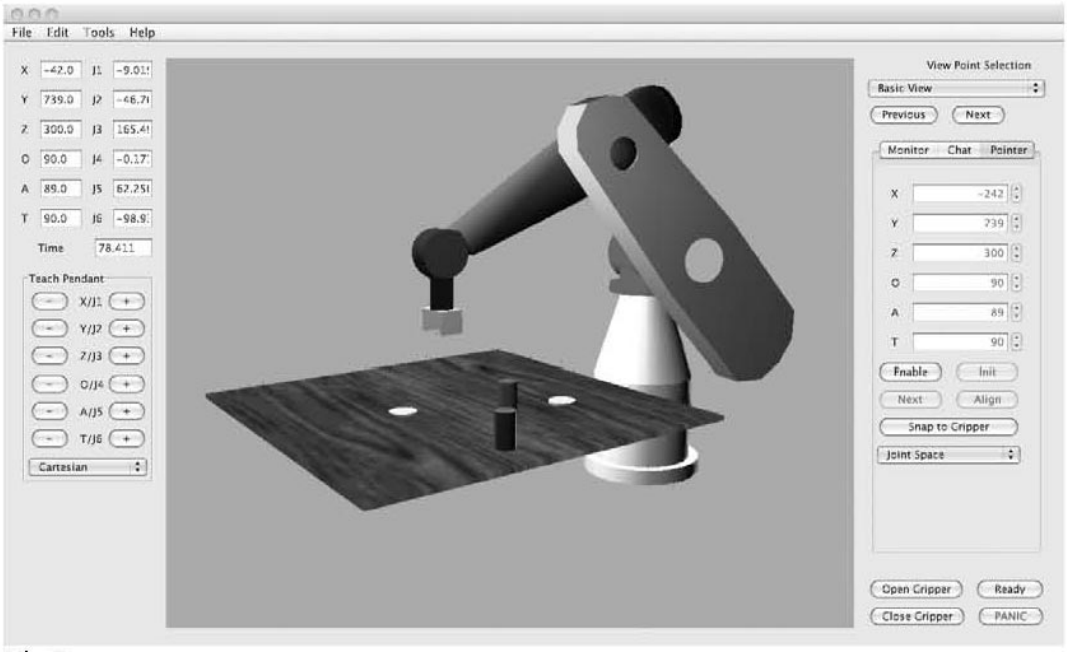

Fig. 7. Operator's interface.

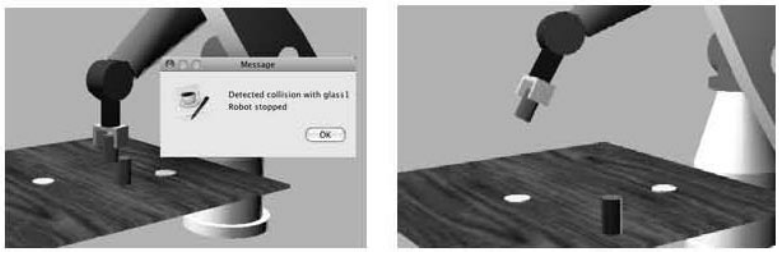

Fig. 8. Collision and grasping.

developed. The tool is based on a simple concept; it provides visual feedback from common view points of both the real and virtual robot's motion that is evoked by a common high-level motion command spawned to both the robots. VVT replaces the visualization panel of the user's interface by a new one that provides real time visual feedback of the real robot through a pan-tilt-zoom static camera and a visual feedback of the simulation through the 3D scene (Figure 6).

The pan-tilt-zoom camera's position in the robot work space (found by applying TSAI algorithm [8]) was used to place the virtual camera in the virtual scene so that both have the same position with respect to the robot and thus provide the same viewpoint to the user. The pan, tilt and zoom parameters of the camera can be controlled by the user using the panel's scroll bars (Figure 6). This action simultaneously updates the respective parameters of the virtual camera so that the two viewpoints remain identical. The user's motion commands are forwarded instantly to both the simulation engine and the real robot's terminal. Thus the motion of the two robots is synchronous and provides a visual validation of the virtual robot simulation.

Operator interface for virtual robot task control

The most important part of this interface is the virtual world panel that enables the graphical input of target robot gripper positions and orientations through a virtual pointer and displays the robot task in the three dimensional space. The interface also includes the monitor panel that is a high level robot motion command area and a virtual 'teach pendant' for real time motion control of the robot in joint or end effector coordinates. Last, the user interface includes a real time status display panel where current joint or end effector positions are displayed during the motion of the robot (Figure 7).

- 3D virtual world panel, the main part of the interface, is the 3D virtual scene of the virtual robot and object setting which is shown through a preset viewpoint (or a preset virtual camera position). The user is able to change the viewpoint through the upper right panel either by choosing one of a number of preset viewpoints or by dragging in the virtual word with a pointing device (e.g. mouse) in order to modify the virtual camera's current position. Hence, the user is able to choose any viewpoint without any constraints that may exist in the real world in order to comprehend the robot and objects relative location. The virtual object setting is chosen by the supervisor for the current teaching purposes out of a set of virtual scenes.
- Collision detection algorithms are used to detect collisions between virtual objects and the robot. A collision report informs the user instantly for the collision existence and the collided objects (Figure 8). Collision algorithms evaluate the virtual scene every time that there is a motion. Complex collision algorithms demands large processing power and hence smooth visualization (requiring 25–30 fps) may be affected. Collision detection can be turned on and off.
- Grasping visualization. When a close gripper command is issued from the panel (button at the right down corner of the GUI) and there is an object between the gripper fingers then the

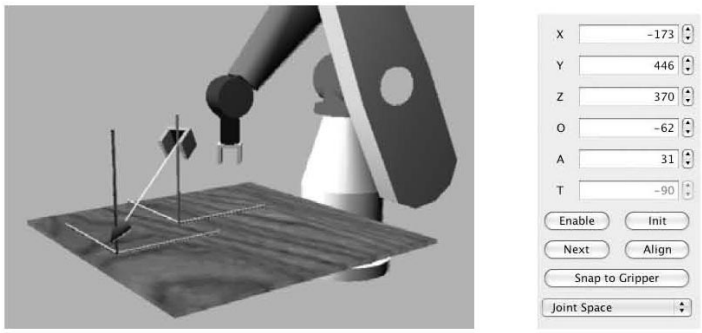

Fig. 9. Virtual pointer.

object's frame is attached to the gripper's frame so that the object is moving with the gripper from then on (Figure 8). When an open gripper command is issued the object is detached from the gripper. If the gripper releases an object in the air a simple gravity algorithm enforces the object to appear in the table's support level below in static equilibrium. Incorporation of physical laws for the simulation of dynamic phenomena like bouncing slipping rolling etc. are not supported in the present version.

- Virtual pointer is a virtual object that appears in the 3D environment in a parking position when enabled by the user and can be easily manipulated in order to define intuitively desired target positions and orientations (Figure 9). Cartesian coordinates and Euler angles or equivalent axis-angle representations are the usual parameters for an object's position and orientation in space; their numeric specification is however counter intuitive as compared to the virtual pointer that is adopted in this work which works well with spherical robot wrists. Its basic idea is the graphical determination of the desired gripper axis (or wrist approach direction line) and the desired gripper rotation around it. Thus, the visual representation of the pointer is an arrow (a segment of a directed straight line in the 3D space) whose base is attached to a ghost gripper virtual object. The placement of this line segment in the virtual world defines the desired gripper position as denoted by the ghost gripper position and the desired gripper approach vector that corresponds to two independent orientation parameters. Dragging the Cartesian frames associated with each of the arrow edges allows their xy positioning while dragging the arrow's edges (i.e. the ghost gripper and the arrow tip) along the corresponding vertical Cartesian axis allows z-positioning. Thus, the desired target position coordinates and two orientation parameters have been specified. Activating the next stage by pushing the 'Next' button the user can last define the desired gripper rotation around the approach vector by dragging and so finalize the desired gripper's orientation. On the right of the virtual 3D panel, numerical values for the virtual pointer position coordinates and OAT (Orientation Altitude Tool) Euler orientation angles are displayed; they can be fine tuned by the user through up and down button arrows or by directly importing the data (Figure 9). Extra features of the virtual pointer include first, the possibility to drag the ghost gripper along the approach vector in order to allow repositioning at a given orientation (by clicking and dragging the arrow line) second, the fast vertical orientation of the virtual pointer by using the 'Align' button and last the 'snap to gripper' button that places the virtual pointer at the current gripper position and orientation (Figure 9). After setting the desired target through the visual pointer pressing the next/go button issues a motion command to the virtual robot to go to the desired target choosing either a 'joint space' interpolation motion or a 'Cartesian space' linear motion.
- The monitor panel consists of a command line and a display area (Figure 7). High level robot motion commands can be issued through the command line. These commands adopt a VAL-II like syntax. VAL-II is the PUMA programming language. Motion commands may specify either a desired joint position (six desired joint angles) or a desired gripper position and orientation (Cartesian position and OAT Euler orientation angles used by PUMA manipulators). The motion mode can be a joint interpolation motion or a straight line motion and is determined by the syntax of the command.
- The special motion control panel is the button panel below the monitor which includes open and close gripper buttons, the panic button that stops robot motion and the ready button that is a motion command that brings the robot in its parking position (straight up)
- The virtual teach pendant is a button panel similar to the teach pendant provided by most industrial robotic arms. Through this virtual teach pendant the user can only induce motion of one robot joint or one generalized coordinate at a time. The discrete motion steps that correspond to each button click can been refined according to the user requirements
- Real time status display panel is in the left upper part of the GUI. It provides the viewer with a numerical perspective of the current position of the robot (Figure 7). Data are updated directly by the broadcasting stream sent by the simulation engine. The data packets include the Cartesian position, the joint angles and a time stamp.
- A chat facility area in the GUI implements a synchronous communication channel between the teacher and the students as well as among the students. This is provided by an instant messaging module in which the users are able to chat.

Pedagogical aspects and instructional objectives and approaches

VRTS was developed and integrated into an undergraduate robotics course following a general reconstruction of the curriculum of the electrical and computer engineering department aimed at the increase of use and student exposure to information technologies. The development of VRTS and the way it is used has made some fundamental assumptions about the pedagogical approach; first, teaching is based on constructivist learning theory and second every pedagogical approach follows the generic pedagogical model developed by Hubka and Eder [9] for the design engineering discipline. VRTS embeds the broad principles of constructivism that emphasize the individuality of knowledge representation and the importance of learner directed discovery of knowledge and social

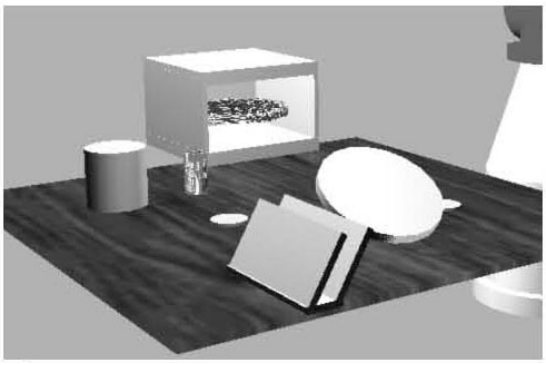

Fig. 10. The project object setting.

interaction through its features of individualized access and interactive exploration of a realistic context and chatting/communication facility. Taking a moderate constructivist view, VRTS has not replaced formal instruction on the related teaching material but was used in parallel to increase attractiveness as well as effectiveness of the learning program. Students were encouraged to use VRTS from the beginning of the course following a certain instructional sequence or as a discovery learning resource. Its early introduction in the course was made possible by the VRTS operator interface features like the virtual pointer that allow its use without a prior robot kinematics and programming knowledge or skill, and from this aspect VRTS is unique. Thus, students are given the opportunity to carry out large realistic tasks without needing to learn all of the subtasks and calculations involved at an early stage; they may later on attempt to carry out the authentic task using the monitor commands.

As reported in [10] student exposure to practical issues through lab demonstrations and particularly lab assignments produce effective learning outcomes. VRTS has thus been used for supervised, unsupervised learning and student assessment.

- In supervised learning, the tutor used the tool to demonstrate basic robotic concepts, instead of using static images and figures or instead of using the real robot site where, if student numbers are high, a clear view may be restricted or obstructed and choice of preferred viewpoint may be limited. At an early stage of the course the visual validation tool was used to demonstrate the simultaneous response of the real and the virtual robot under the same motion commands in order to increase the students' trust and enthusiasm for the virtual robot.
- In unsupervised learning, lab exercises are assigned by the tutor to individual students or student groups for collaborative work at specific times during the course. The didactic concept relies on problem based learning that can be achieved through the accomplishment of different robot handling tasks with increasing aspiration level. In particular, robot handling tasks involve initially pick and place exercises of virtual objects that involve target position definitions and point to point paths (cans on bases), then pick and place tasks that require gripper reorientation (dish on declined stand) and pick and place tasks requiring careful path planning (intermediate points and straight line paths) for obstacle avoidance. Lab assignment is implemented in the VRTS server by changing the virtual object setting. The change of the setting is performed by the system supervisor. Then students can connect at their own time to perform the lab exercise. The alternative ways for motion command definitions (high level programming, teach pendant, virtual pointer) and executions embedded in VRTS allows the accommodation of different types of learners ranging from the practically inclined to the analytical one. For each task the student has to define and execute a motion plan consisting of robot motion commands in order to achieve the task goal and observe the response to their commands on the virtual robot scene. During the assignments the student may have to experiment with various approaching and grasping strategies, modify or redesign their motion plan and repeat the task. The specific learning target is to acquire knowledge and skills regarding industrial robot motion planning and control. Synchronous communication using the chat facility with other connected students and the tutor (who connects at specific times) reinforces learning through exchange of information and opinions.
- A robot object handling and manipulation project is used at the end of the course to assess the student performance based on a number of measurable performance indices. The project object setting is a clustered environment consisting of cans, dish, bases, oven, and pizza (Figure 10) lying on the working table for which a number of collision free handling tasks are required to be performed. Judging the quality of the solution achieved by the number of collisions if any, the target achievement, the intermediate path positions and the total task duration an overall mark was given for each student in the scale 0–10.

Empirical evidence and assessment of VRTS effectiveness

As the interest, attention and motivation of the students are critical to the success of any teaching tool, questionnaires were passed to measure student's reaction to the tool regarding its technical characteristics after the first assignments. Students can connect to VRTS from the University or their own home computer and familiarize themselves with the tool since the beginning of the course. The student sample size was 23. Three VRTS technical characteristics were measured, namely user friendliness, functionality and interactivity. For the latter characteristic a recently

Table 1. VRTS technical characteristics

	Average	Standard Deviation	Weighted average
Usability	3.89	0.43	3.91
Functionality	3.55	0.46	3.55
Interactivity	3.89	0.42	3.92

Table 2. Student interest and perceived learning outcome

	Average	Standard Deviation
Learning	3.57	0.54
Interest and motivation	3.95	0.53

developed scale of perceived interactivity is used which emphasize control, responsiveness and personalization issues [11]. For each characteristic, student opinion was asked through a number of questions and for each question the student had to indicate its level of importance. A one to five scale was used in both cases.

Table 1 shows the average and standard deviation for each characteristic as well as the weighted average. Notice that usability and interactivity scored approximately a 3.9 weighted average while functionality stayed at 3.55. The latter mark was probably the result of program bugs at this beta version of the simulator. Improvements of VRTS in future versions can benefit from these results.

In order to assess the influence of the teaching activities with VRTS on student's learning, a second questionnaire was used after the end of the VRTS lab using a number of questions regarding two characteristics; perceived learning and motivation. In fact, the collected data are student opinion on whether the comprehension, knowledge and skill acquisition and theory to practice connection has been enhanced as well as student interest and motivation for robotics. A one to five scale was used and the average and standard deviation is shown in Table 2.

Results indicate that VRTS may have raised their interest and motivation for robotics relatively more than their perceived learning outcome. The latter was compared with the marks of formal testing produced by the student performance on the composite robot handling project using VRTS and VRTS's evaluation capabilities. A regression test between the variables of perceived learning (independent variable) and the formally acquired lab exam marks (dependent variable) was performed using SPSS. The most important statistical results are shown in Table 3 and Figure 11.

The linear regression graph is shown in Figure 11 and the regression line model coefficients are given in Table 3; the regression line has constant – 4.896 and slope 2.720. It is clear that the two variables are positively correlated. The Beta variable (0.720) in Table 3 indicates the degree of correlation between the two variables and is equal to the R coefficient whose square known as the coefficient of determination appears in the scatter diagram (Figure 11); in fact, R square = 0.518 which means that approximately 52% of the observed values can be explained by the regression model. The curves at both sides of the straight line (Figure 11) represent the confidence band within which the "real" trend line lies with a 95% probability. Other statistical results include the mean and standard deviation of the dependent variable (the lab exam marks) that are equal to 4.8188 and 1.95 respectively. Furthermore, the standard error of the estimate according to the model is given

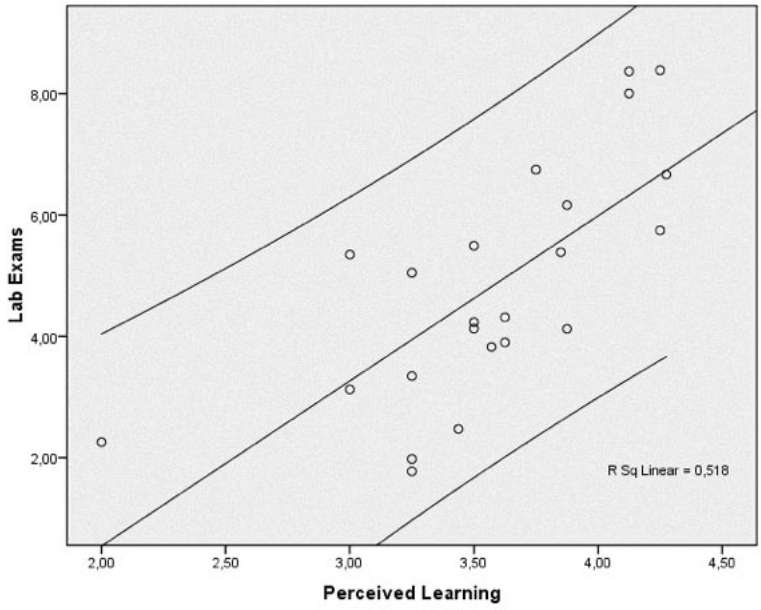

Fig. 11. Scatter diagram of student perceived learning and project mark.

Table 3. Coefficients*

Model	Unstandardized Coefficients		Standardized Coefficients	t	Sig.	95% Confidence Interval for B	
	B	Std. Error	Beta			Lower Bound	Upper Bound
1 (Constant)	–4,896	2,064		–2,372	,027	–9,189	–,603
Perceived Learning	2,720	,572	,720	4,753	,000	1,530	3,911

* Dependent Variable: Lab Exams

equal to 1.38 that is less than the standard deviation of the dependent variable (1.95) and hence the regression model predicts the dependent variable better from its mean (4.8188). Another statistical result is the F ANOVA significance level value of 0.0005 which means that the probability of the type I error is less than 0.0005 and consequently the variability of the dependant variable is explained satisfactory by the independent one.

CONCLUSIONS

This work describes a web-based virtual robot arm simulator and its use for laboratory experiments on industrial robot planning and control through a user- friendly interface that allows a non-expert user to perform basic industrial robot tasks. The integration of this system in the robotics teaching has stimulated student interest and motivation and their perceived learning was positively correlated to lab exam marks. Students can benefit from the increased time and space flexibility offered by the tool's network aspect while the acquired 'practical' experience helps them in bridging the gap between theory and practice. However, during the laboratory experiments students' questions were about technical issues concerning the real robot's function rather than the theoretical basis of robotics. Thus, despite the students' enthusiasm, the instructor's perception is that the use of VRTS did not affect significantly depth of learning. Nevertheless, it is the instructor's belief that a robotics course should expose students to practical issues. To this aim and based on the initial findings of VRTS use, it is planned to incorporate VRTS at the initial phase of a course project, as a robot programming learning and debugging tool, in order to produce a robot task program that will eventually be dispatched to the real robot. Although the simulator will allow the students to initially complete the task using VRTS graphical command tools and to perform typical mistakes and learn from them, a real robot motion command program will be produced in the end and tested in the real robot.

Acknowledgment—The project is co-funded by the European Union—European Social Fund and National Resources—(EPEAEK II).

REFERENCES

1. P. Antsaklis, T. Basar, R. DeCarlo, N. H. Clamroch, M. Spong, and S. Yurkovich, Report on the NSF/CSS workshop on new directions in control engineering education, *IEEE Contr. Syst. Mag.*, **19**, 1999, pp. 53–58.
2. D. Jonassen, *Computers in the classroom: Mindtools for critical thinking*. Englewood Cliffs, NJ: Prentice Hall. (1999).
3. A. C. Croft, M. Danson, B. R. Dawson & J. P. Ward, Experiments using computer assisted assessment in engineering mathematics, *Computers & Education*, **3**, 2001, pp. 53–66.
4. D. Dolan, B. Holmes, D. Leahy, P. Lynch, T. Ward and Y. Amghar, European E-tutors—inductive models for on-line learning in synchronous collaborative environments, *Third International Conference on Multimedia and Information and Communication Technologies in Education,* m-ICTE 2005, Caceres, Spain, (June 7–10 2005).
5. A. Valera, J. L. Diez, M. Valles, and P. Albertos, Virtual and Remote Control Laboratory Development, *IEEE Contr. Syst. Mag*, February, 2005, pp. 35–39.
6. Z. Doulgeri, T. Matiakis, A web telerobotic system to teach industrial robot planning and control, *IEEE Trans. Educ.* **49**(2), 2006, pp. 263–270.
7. Z. Doulgeri, S. Tzortzidou, G. Hassapis, Teaching robotics through an e-learning environment that allows the application of e-tutoring guidelines and supports telelab operations, *WSEAS Transactions on Information Science and Applications* **3**(11), 2006, pp. 2107–2113.
8. R. Y. Tsai, A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses, *IEEE J. Robotics and Automation*, **RA–3**(4), 1987, pp. 323–344.
9. V. Hubka and W. E. Eder, Pedagogies of design education, *Int. J. Eng., Educ.* **19**(6), 2003, pp 799–809.
10. J. Self, From constructionism to deconstructionism: Anticipating trends in educational styles, *Eur. J. Eng. Educ.*, **22**, 1997, pp 295–307.

11. G. Wu, Perceived interactivity and attitude towards website In M.S. Roberts (Ed.) *Proc. 1999 Annual Conf. American Academy of Advertising*, Gainesville, FL: University of Florida, (1999). pp. 254–262.

Zoe Doulgeri is currently an Associate Professor in Robotics and Control of Manufacturing Systems at the Dept. of Electrical and Computer Engineering of the Aristotle University of Thessaloniki, Greece. She received a diploma in Electrical Engineering in 1980 from the Aristotle University, an M.Sc. in Control Systems in 1982, an M.Sc. in Social and Economic Studies in 1983 and a Ph.D. in Mechanical Engineering in 1987 from Imperial College, London, UK. Her current research interests are in the object grasping and manipulation by robot fingers, the control of robot contact tasks under kinematic uncertainties and the use of web telerobotics and virtual robotic environments in the teaching of robotics.

Nikos Zikos received a diploma in Electrical and Computer Engineering from the Aristotle University of Thessaloniki, Greece in 2005. He is currently an M.Sc. student in cognitive systems at the same department. His interests include robotics, automatic control and cognitive systems.