Proceedings of the 2002 IEEE/RSJ
Intl. Conference on Intelligent Robots and Systems
EPFL, Lausanne, Switzerland • October 2002

A Predictive Interface Based on Virtual and Augmented Reality for Task Specification in a Web Telerobotic System

R. Marín P. J. Sanz A. P. del Pobil

Jaume I University, Robotic Lab, Castellón, Spain, {rmarin, sanzp, pobil}@uji.es

Keywords: Automatic Speech Recognition, Telerobotics, Web technologies, Human-Robot Interaction.

Abstract

A very interesting robotics area is the high-level tasks' specification, and the way a user-robot interface can facilitate programming complex actions to be executed on real environments. This paper shows a novel contribution in the user-robot interaction domain, and particularly in the web robotics field. The user, who is in the control loop, is able to specify the robot actions over a simulated scenario (running off-line), assisted with 3D virtual and augmented reality. Then, once the task is defined, it can be easilly executed over the real scenario (running on-line). To get its main goal (i.e. task specification) working in real life scenarios, the system incorporates automatic object recognition and visually-guided grasping among others capabilities.

1. Introduction

Only a few years ago, researchers presented the first telerobotic systems with a web-based interface at the University of Southern California (USC) and the University of Western Australia (UWA). The Mercury Project [2], carried out at the USC, led to the development of a system in which the manipulator was a robotic arm equipped with a camera and a compressed air jet, and the interface consisted of a web page that could be accessed using any standard browser. The interface allowed the user to move the robot to a point in the workspace and blow a burst of compressed air into the sand directly below the camera. All robot controls were available via the mouse interaction.

The telerobotic system developed at the UWA [9] lets the user control an industrial robot to manipulate objects distributed on a table. The user interface allows an operator to specify the coordinates of the desired position of the arm, the opening of its gripper and other multiple parameters by filling forms and clicking images through an HTML interface. Recently, [1], a more advanced user interface for the UWA telerobot has been presented, where some information about the gripper position is added to the images coming from the server in order to provide some augmented reality feeling.

As described above, this kind of systems let the user send simple commands to the robot such as moving to a specific location in its workspace and performing an action with its attached tool. Very little attention has been paid to the use of more natural ways of interaction like natural language, or even virtual reality.

On the other hand, there are two important problems related to the web-based telerobotics. The first one, the *cognitive fatigue* [6], is a consequence of being always attending and controlling every robot movement (teleoperation). The solution is to give more intelligence to the robot and let the operator specify tasks at a higher level of interaction (e.g. "Grasp the scissors"), that determines a supervised human-robot interaction. The second problem, the *Internet latency*, implies that the user must wait some unpredictable time to get the real results of his operations on the real robot. To avoid such a boring interaction, some alternatives like *predictive displays* can be applied [10].

In our case, the predictive capability through virtual and augmented reality scenarios, and the supervisory control by means of some advanced features like visually-guided grasping, and processing very high level actions, are both available.

Summarizing, included in this paper are the integration of automatic object recognition [5], visually-guided grasping [7] and advanced user communication (i.e. voice, text and mouse) capabilities in a multimedia interface. Therefore, the system is able to respond to commands like "pick up the pen" in on-line connection. Finally, the user can also specify tasks in off-line mode, and save them into a text file. He can thereafter execute them in the real scenario once the robot is accessible (on-line connection).

2. Experimental setup

In Figure 1, the robot scenario is shown, where three cameras are presented: one taking images from the top of the scene, a second camera from the side, and a third camera from the front. The first camera is calibrated, and used as input to the automatic object recognition module and 3D-model construction. The other two cameras give different viewpoints to the user when a teleoperation mode is necessary in order to accomplish a difficult task (e.g. manipulating overlaping objects). Besides this, they allow the operator to check the proper execution of high level commands like "*Grasp the cylinder*".

Second, in order to perform a proper 3D model construction from the top camera input, it is necessary to know the mean height of every object in the scene. A distance sensor installed on top of the gripper is used when this calculation is necessary (e.g., an unknown object is detected). Besides this, it allows the implementation of some collision detection algorithms, giving more autonomy to the robot. Several experiments with distance sensors have been performed, which give an idea of the different situations where this setup works successfully.

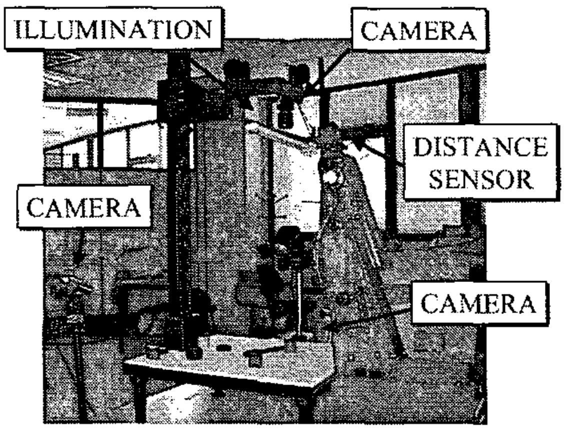

***Figure 1:** Server side experimental setup*

3. The user-robot interaction

As we have introduced, the telerobotic system allows the manipulation of objects over a board by means of mouse interactions on the 3D virtual reality environment and also by using a simplification of natural language. Thus, the system can respond to commands like "pick up the scissors" or "drop it at the left side of the screwdriver". This kind of interaction is possible thanks to an optimized object recognition CORBA module that processes the camera images and returns object names. As the program is able to learn new object characteristics through the user interaction, the system becomes more robust as time goes by. As introduced above, such a capability has not been reported for a telerobotic system yet [2][9][5].

As shown in Figure 2, once the user takes control of the robot, the manipulator goes to the initial position, so the scene information is accessible from the top camera. Then, the camera image is captured and transferred to the *computer vision* module [5], which identifies every isolated object in the scene, and calculates a set of mathematical features that uniquely distinguish every object on the board (moments, contours, area, etc).

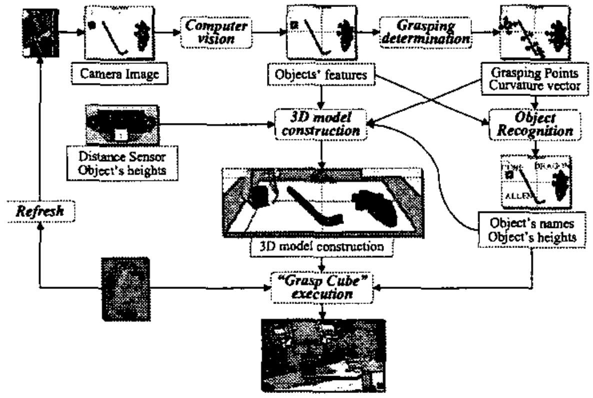

***Figure 2:** User-Robot interaction flow-chart*

Then, the contour information and the moments calculated previously are used to determine every stable grasp associated to each object. This process is crucial to allow high level tasks specification (e.g. "grasp object 1"). For more details, refer to [7]. Another important output of the grasping determination procedure is the *k*-torsion vector, which defines the external curvature representation of every object. This information can be very useful in order to accomplish the following procedure, which is the *automatic object recognition* process.

By using the computer vision output (moments, area, etc.) and the grasping determination information (curvature, grasping points, etc.), the *object recognition* module calculates a set of invariant descriptors associated to every object (thinness ratio, elongatedness, Hu moments, etc.). Then, it compares this information with the already learnt representation of known classes stored into a database. By applying the object recognition algorithms described at [5], an object name (e.g. allen key, screw, etc.) is associated to every object in the scene. Now, interaction with the robot can be accomplished by using commands like "Grasp allen", instead of "Grasp object 1".

At this point, we shall use the output from the grasping determination and the computer vision module in order to *construct a 3D model* of the robot scenario. Thus, the user will be able to interact with the model in a predictive manner, and then, once the operator confirms the task, the real robot will execute the action. The 2D camera input is not sufficient to construct the 3D model. In order to calculate the object's heights the distance sensor and the object recognition output are used. Once the 3D model is constructed, the user is able to program the robot in a predictive manner. Thus, for example, the operator can speak directly to the microphone and command the action "Grasp Cube". The task is first executed on the virtual 3D model, and then, once confirmed by the user, performed by the real robot on the remote scenario.

3.1 Multiple ways of Interaction

One of the main important features of our System is the possibility of interacting at different levels depending on the actual state of the robot (busy or available) and the current task to be performed (supervised or teleoperated).

First of all, the user interface enables two interaction modes:

Simulated (Off-line): For those situations where the robot is busy, the user can still interact with the robot in a simulated manner. Thus, the only requirement is having to provide the user interface with a top camera image (it can be the real working environment) and then the 3D model is constructed with the different objects. The important point is that the user can specify a whole assembling task by using this initial state (camera image), which can be applied later on in a single step over the real robot (task specification).

Real Robot (On-line): Moreover, when the robot is available, the operator can interact with it in a very similar way to the simulated one. The only difference is that once a robot operation has been confirmed, apart from storing it into the task specification field, the operation is launched on both, the real and the simulated robot. Moreover, in order to sort out the time delay problem over the Internet, the *on-line* interaction is accomplished by default using a predictive display. This means that the user can specify, for example, a grasping task over the simulated interface and then deciding whether applying this operation or not to the real robot. This predictive configuration can be deactivated and then work directly with the real robot in a *Move&Wait* strategy [10].

As presented in Figure 3, for every user interface mode (*off-line, on-line and predictive, on-line and direct*), we can control the robot at different interaction levels, depending whether the task is to be *supervised* (intelligence at the robot) or *teleoperated* (intelligence at the operator).

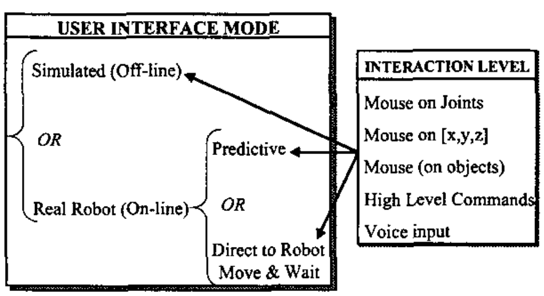

Figure 3: User interface modes (simulated, real, etc.) and their possibilities of interaction with the robot (Mouse, Voice, etc.)

An interesting example is shown in Figure 4, where a complex situation needs to be solved. Because the objects are overlaping, it is convenient that the user himself provides the intelligence (teleoperated). The user can select a given grasp among those presented, separating the objects before assembling them. Otherwise, when the objects appear isolated and hence are recognized by the system, the user can operate in a semiautonomous manner (supervised interaction) with commands like "pick up the cube", as is observed in Figure 5.

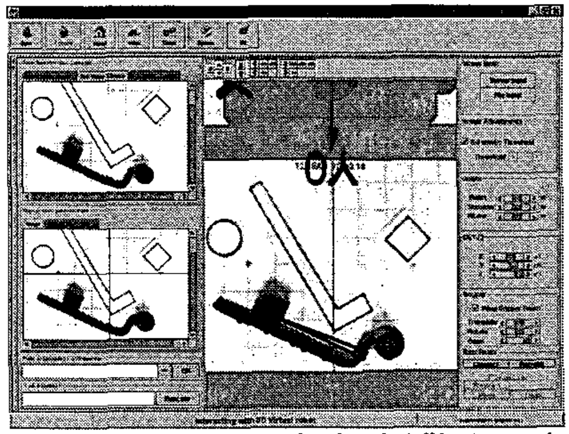

Figure 4: User interface in simulated mode (off-line) to resolve and assembling task where objects are overlapping.

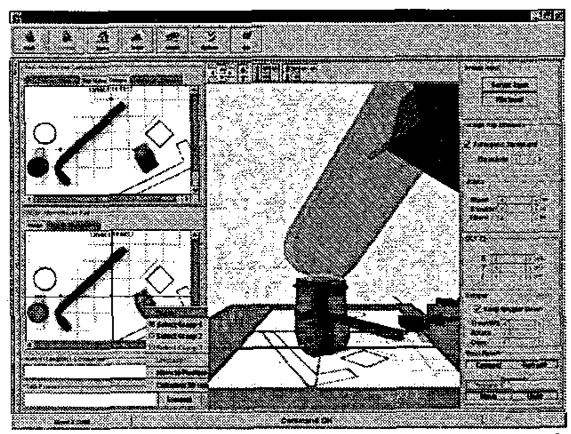

Figure 5: User interface in simulated option predicting the grasping execution of a cube. As the robot position is predicted the manipulator is represented in transparent color.

3.2 Available commands

In order to control the robot by using the high level commands or the voice input, the system defines a complex set of commands. Some of them are summarized in Table 1.

In summary, the user interface offers a great flexibility to the user, in such a way that it allows selecting the interaction level required at a particular moment. For those situations where the robot intelligence is insufficient to accomplish the task (e.g. objects overlapping), the operator can interact at a lower level by selecting directly grasping options or even moving the robot in world and joint coordinates (*teleoperation*). In addition, when normal situations are presented (e.g. well isolated objects), the robot will be able to respond properly to high level commands like "grasp the scissors" (*supervised telerobotics*).

***Table 1:** Some of the available commands*

Command	Description
Grasp the {object name}	Given a scene with objects it executes the most stable grasp alternative on the object {object name}.
Grasp the object {number}	It executes the most stable grasping alternative on object {number}, numbered from top to down and from left to right.
Move {ahead / back /left /right / up / down}	It moves the TCP {ahead / down /left /right}x cm, taking the manipulation pad as the reference
Move to position {x} {y} {z}	It moves the TCP to the world coordinates position [x, y, z]
Pick the {object name} up	It executes the grasping operation over the object labelled as the {object name}, and then elevates it x cm over the board
Place it over the {object name}	Having an object grasped by the gripper it places it on top of the {object name}
Refresh	It brings the robot to the initial position, captures the scene image, recognizes the objects, and constructs the 3D model.

4. Virtual & Augmented reality

Virtual reality has been a subject of great interest, and increasing attention is being paid to the related field of *Augmented Reality*, due to its similar potential. The difference between Virtual Reality and Augmented Reality is in their treatment of the real world. Virtual Reality immerses a user inside a virtual world that completely replaces the real world outside. In contrast, Augmented Reality lets the user see the real world around him and augment the user's view of the real world by overlaying or composing three-dimensional virtual objects with their real world counterparts. Ideally, it would seem to the user that the virtual and real objects coexist.

As it can be seen in Figure 6, our user interface presents a non-immersive VR 3D model implemented with Java3D that enables the user to vary his viewpoint by means of mouse interactions. Besides this, in order to facilitate the robot programming when the camera input is partially occluded by the robot, an augmented reality feature is added to the "Object Manipulation Pad" panel. Hence, in our system, the augmented reality capability has been used to improve the user interaction with the robot. For example, for those situations where the real robot is below the camera (objects' occlusion), it is necessary to generate some kind of virtual information in order to let the user program the task. Other features, like object measurements etc., can be easily added to the system by using the same technique.

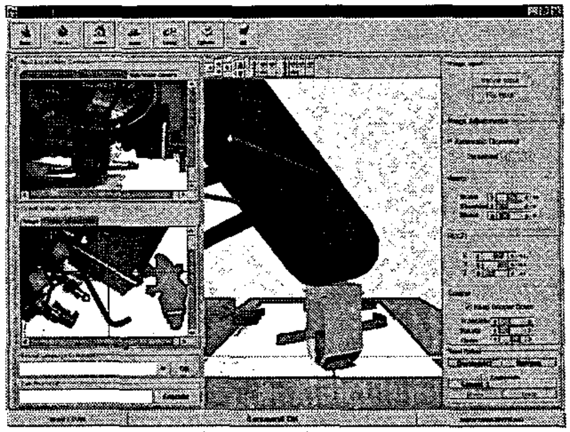

***Figure 6:** Augmented reality and non-immersive virtual reality are put together in a 3D multimedia interface*

5. The task specification

A very challenging field from the very beginning has been to develop a programming language directed to the task, and being natural to the user. Some extensive research was conducted in that direction during the 80's, such as RAPT (University of Edinburgh) or AUTOPASS (IBM), but with limited success. Nowadays, after successful research has been developed, advanced information technologies are available which facilitate the improvement of these techniques.

In our case, note that for those situations where the robot is being accessed by someone else on the Internet, the system allows users to program the manipulation activities in an off-line manner, by using the 3D model representation of the robot scenario as a task simulation tool. Then, once the task is programmed, its specification is stored into a set of high level commands that can be saved and executed afterwards on the real robot. The idea is providing users with a tool that enables interacting with the robot without depending on the actual accessibility. This is another project contribution to the web telerobotics domain, where the majority of available systems make users wait until the robot is free to be used (e.g., Telegarden and Australian Telerobot). This off-line programming has application on many environments. For example, in an industrial domain this kind of technique is very convenient, due to the fact that while the robot is being programmed to perform a given task it can be accomplishing an important activity on the production line. The robot programming does not mean stopping the production activity. Moreover, in the education and training environment (like ours), letting fifty students wait until one of them frees the robot control is obviously not very convenient.

5.1 A case study: Pick & Place task operation

At this point we are going to see an example of programming a simple task: picking an object and placing it on a selected board position. The problem can be observed in Figure 7, where the cylinder has to be placed on its corresponding position.

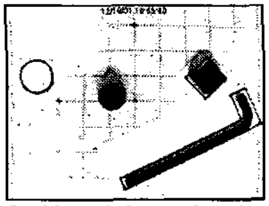

Figure 7: Problem of placing the cylinder into its corresponding position (circle)

Once specified into the telerobotic system, the initial scenario can be seen in Figure 8.

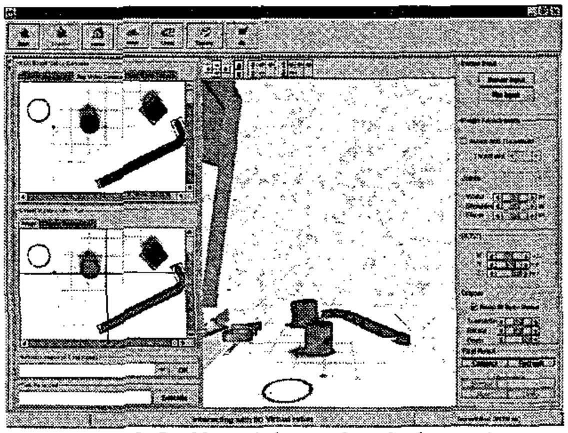

Figure 8: Problem of placing the cylinder into its corresponding position (circle)

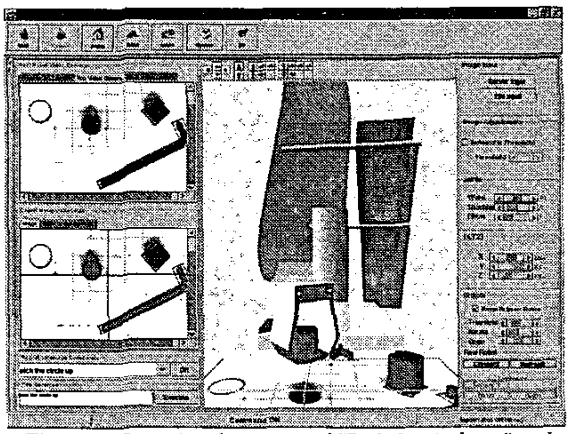

Figure 9: Executing the command "Pick the circle up" and saving into the task specification field

Next step is using any of the robot commands (mouse, text, etc.) to begin the task execution. A good alternative would be picking up the circle by using the command "Pick the circle up". The result is observed in Figure 9.

As the robot already knew the grasping points associated to the circle object, it just executed the only grasping alternative and elevated the object along the Z-axis. Look how the command has been stored into the Task specification field once confirmed by the user by pressing the "Move" button.

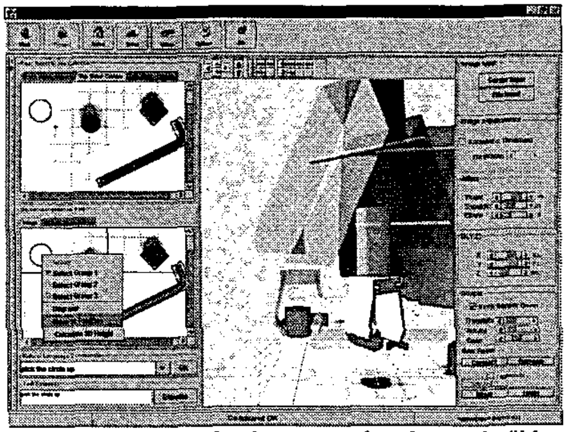

Figure 10: Specifying the placing point by selecting the "Move To Position" option

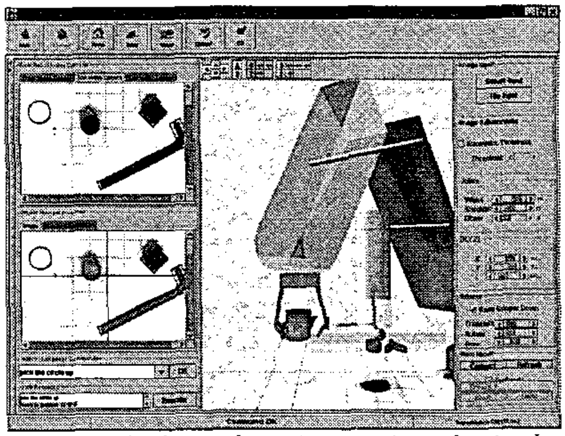

Figure 11: Confirming the moving operation and saving the action on the task field specification field

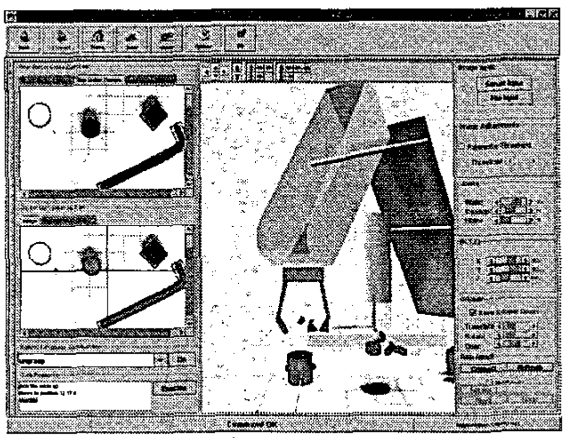

Figure 12: Selecting the ungrasp option

As observed in Figure 10, the following action is to select the exact point where the object is to be placed. A possible alternative could be, for example, clicking the placing point in the manipulation pad, and then selecting the "Move to Position" option. Then, once the user confirms the selected position, the results are shown in Figure 11. The opaque robot is moved to the corresponding position and the action is saved into the task specification file. And finally, as shown in Figure 12, by selecting the "ungrasp" option on the manipulation pad the robot places the circle on the board. Note how the action has been recorded into the task file too. Finally the task programmed is the following:

pick the circle up
move to position 12 17 5
ungrasp

Now, the task could be executed again and again over the off-line environment and on the real robot once the user takes control of it.

6. Conclusions

This paper presents a web-based on line robot system, which uses virtual & augmented reality in order to desing a predictive display that enables specifying high level tasks on both, off-line and off-line modes. The system is being used as an educational product for teaching-learning basic concepts in robotics to every user connected to the web. The idea is allowing the student to have a tool that helps to learn the difficult robotics concepts and complements traditional lectures [3][4]. See the web-site: (http://ciclop.act.uji.es/rmarin/). In fact, our experience has demonstrated that at students can be very motivated programming into a virtual environment if the user interface is designed using proper techniques (virtual and augmented reality, 3D model construction, task prediction, etc.). Then, they are able to execute their programs (i.e. robot tasks) on the real robot.

The training system has been implemented using multimedia technology and is able to be run over the Internet by means of a standard browser. This has been possible thanks to the use of Java and CORBA technologies. Besides this, note that the multimedia programming with Java is possible thanks to the newly launched Java2D and Java3D API's [8]. Moreover, Java3D technology has made possible the implementation of such a system that offers an interaction based on virtual and augmented reality at the same time. The result is a tool that enormously facilitates the manipulation of objects over a board.

Related to the task specification procedure, it is remarkable that, in order to execute a programming task over the real robot, it should be taken into account if the initial state is the same on both, the real and the simulated robot. Depending on the way the task has been defined, it will be dependent or independent of the initial object's position. Finally, as introduced in the above section, this kind of user interface is very appropriate because it allows the manipulation of a robot arm even when there is no physical connection to the robot. The 3D virtual environment takes care of these situations.

Acknowledgements

This paper describes research carried out at the Robotic Intelligence Laboratory of Universitat Jaume-I. Support for this research is provided in part by the CICYT under project CICYT-DPI2001-3801, by the Conselleria de Cultura i Educació (Generalitat Valenciana) under projects inf01-27, GV01-244, by the Fundació Caixa-Castelló under project P1-1B2001-28.

References

[1] Behnke R. (2001): A User Interface for telecontrol of a robot over the Internet: First IFAC Conference on Telematics Applications in Automation and Robotics TA 2001, Weingarten, Germany

[2] Goldberg K. (2000): The Robot in the Garden: Telerobotics and Telepistemology in the Age of the Internet: The MIT Press Cambridge, Massachussets, 2000. ISBN 0-262-07203-3.

[3] Jona M., Bell B., and Birnbaum L. et al. *Button Theory : A taxonomy of student-teacher communication for interface design in computer-based learning environments.* Proceedings of the Thirteenth Cognitive Science Conference, pp. 765-769. Chicago, IL, 1991.

[4] Marin R, Sanz P.J., Coltell O., et al. *Student-teacher communication directed to computer-based learning environments*. Displays, Elsevier Science. Special Issue on Displays for Multimedia (17) pp. 167-178. 1997. Abstract available on (http://www.system-concepts.com/displays/)

[5] Marin R., Sanz, P.J., Jimeno, A.J, Iñesta, J.M: Advances in Pattern Recognition: Lecture Notes in Computer Science 1876: Design of a Telerobotic Interface System by using Object Recognition Techniques. Springer-Verlag, Berlin, 2000.

[6] Murphy R.R., Introduction to AI Robotics. The MIT Press, 2000

[7] Sanz PJ, del Pobil AP, Iñesta JM, Recatalá G. "Vision-Guided Grasping of Unknown Objects for Service Robots". In *Proc. IEEE Intl. Conf. on Robotics and Automation*, 3018-3025, Leuven, Bélgica, 1998.

[8] Spell, B. *Professional Java Programming. Wrox Press Ltd, United States, 2000. ISBN 186100382X.*

[9] Taylor K., Dalton B.: Internet Robots: A New Robotics Niche: IEEE Robotics & Automation Magazine, Vol. 7, No. 1: Global Remote Control Through Internet Teleoperation: U.S.A, 2000.

[10] T. Sherindan, Telerobotics, Automation, and Human Supervisory Control. MIT Press, 1992.