VIREC, A VIRTUAL ROBOT CENTER FOR E-LEARNING

Ciprian Comşa*, Robert Mitrică*, Mircea Nițulescu, Gabriel Vladuț***

** S.C. IPA S.A. CIFATT Craiova*
***University of Craiova, Faculty of Automation, Computers and Electronics*

Abstract: In the technological field, virtual reality technology has been widely proposed and recognized as a major technological advance for supporting life-long education to individuals along with a flexible workforce. One of the unique capabilities of the VR technology is the successful translation of abstract concepts into visualized events and the interaction of students with them, that in real life could be limited due to distance, time, and safety factors. This paper presents a virtual laboratory ViReC accessible through the Internet. The architecture uses open standards, such as World Wide Web and its related technologies: HTTP, HTML, Java, Macromedia Flash, PHP, MySQL and so on.

Keywords: Virtual robotics, Virtual laboratory, E-learning, Internet solutions.

1. INTRODUCTION

The wide expansion of the World Wide Web and the Internet has formed all the necessary preconditions for adopting this powerful means for purposes such as delivery of E-learning content, collaboration and distance learning, both in the industrial and the educational field.

E-learning can also be very useful in the case of a potential public for learning that could be described as "itinerant or that is widely scattered over a vast area" (Hamburg *et al.*, 2003a). E-learning may be a way of overcoming geographical isolation and physical distances from resources or learning canters. In these situations, the use of the Internet and networking takes on its full meaning. Robotics is well suited as a problem domain because it intrinsically requires multidisciplinary knowledge. Robotics encompasses subjects such as mechanical engineering, electronics, control, communication, vision, real-time parallel computing and systems design. By choosing a problem domain with a high multi-disciplinary requirement, we reduce the chance that any one person on the team will "cheat" and do all the work.

As the literature supporting the positive educational effects of robotics accumulates, it becomes increasingly clear that:

- Robotics is very absorbing and enjoyable for many children and adults
- Robotics events and competitions motivate the study of technical subjects
- Robotics provides hands on examples for teaching science, technology, engineering
- Robotics creates excellent possibilities for learning team working, especially through competitions.

Cumulative experience suggests that robotics is an excellent domain for introducing students to a range of technological and scientific disciplines. Robotics provides leverage on two key pedagogic aims of E-learning: problem-based learning and engaging students in discourse (i.e. creating a community of learning). So, robotics possesses a number of specifics advantages that make it particularly well suited to unsupervised project work at a distance. Distance learners often have little opportunity for face-to-face contact, although some courses offer up to two hour-long monthly tutorials or one-off day

schools. However, attendance at such tutorials is not compulsory and there are many reasons why distance education students may be unable to attend meetings.

2. MOTIVATIONS AND DESIGN

The main motivation factors that inspired the design and the implementation of the ViReC and the goal that should be achieved by this work is threefold:

- To provide an E-learning environment for students as well as to overcome communication difficulties among them
- To overcome limitations of current E-learning applications
- To avoid limitations met in most Virtual Robot applications so far.

At a first level we plan to support the community of Virtual Laboratory users as follows:

- To overcome the problem of interaction and communication among them
- To support the students to construct their knowledge in a "learning by doing" situation
- To provide remote support to students by mentors.

The ViRec source of inspiration was a real robotized application for didactical purpose (Nițulescu *et al.*, 2002), developed into the Robotics laboratory of the University of Craiova, Faculty of Automation, Computers and Electronics (Figure 1). The principally components of the overall structure are an ABB IRB 1400 robot (1) and a multifunction didactical platform (2), containing five stations.

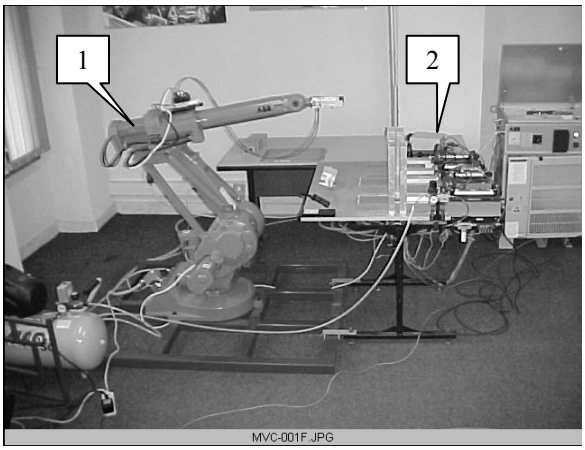

Fig. 1. The real platform for robotized application.

3. ARCHITECTURE AND IMPLEMENTATION

This section is dedicated to describing the structure of the system that supports the ViReC. The term structure is used for referring to the logical view of the static structure of the architecture in terms of its components, their interconnections and the interfaces and operations offered by these components.

The step that follows the definition of the main design principles of the Virtual Robot is the assessment of the technologies that will facilitate the implementation of the system. The system should be an easily accessible web-based system, used by a worldwide community, taking into account bandwidth, and client-side system constraints. Therefore one of the major targets apart from implementing the Virtual Robot with the appropriate functionality is to satisfy the following prerequisites. First, there is a need to minimize the client-side system requirements and the cost of client-side system set-up. This means that the end-user should be able to access the laboratory using a typical personal computer without excessive requirements either in hardware or in software. Second, the system should have the ability to support a maximum number of simultaneous users. This requirement should in particular be considered in the multi-user applications. Finally, using technologies, which do not require excessive hardware requirements, should also minimize the cost of the server side set-up. As far as it concerns the web server, it is used for storing the client-side files of the Virtual Robot, voice chat and text chat. Furthermore, the web server stores and executes the PHP scripts to obtain the users' data from a database. The web server that satisfies these prerequisites is the Apache web server, which is free of charge, runs on almost all operating systems, supports PHP scripts, can host Director and Flash movies and can interoperate with MySQL, Macromedia Flash Communication Server MX and Java. The internal structure of the Virtual Robot Center is presented in Figure 2.

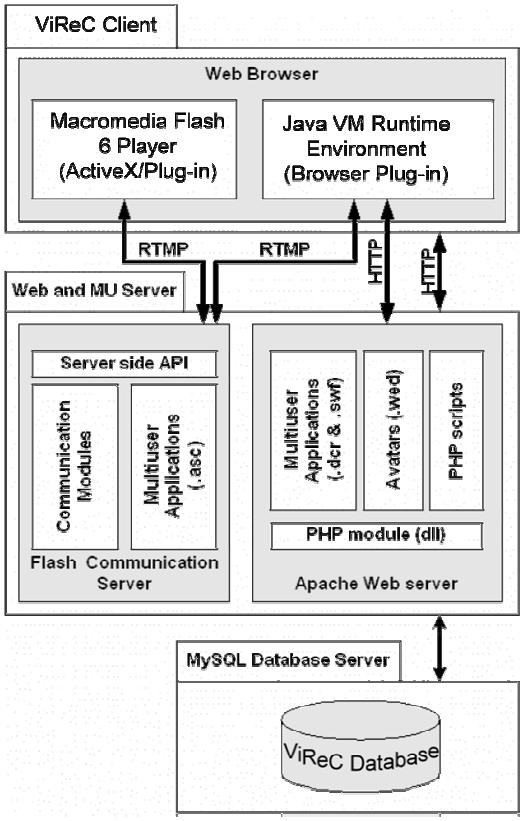

Fig. 2. The structure of the Virtual Robot Center.

Fig. 3. The initial starting GUI interface of ViReC.

As Figure 3 shows, the basic graphical user interface GUI of ViReC contain 5 topics (links): Docs, Tutorial, Exercises, Online Support and Plugins. It can be downloaded from our site http://www.ipacv.ro /proiecte/robotstudio/index.

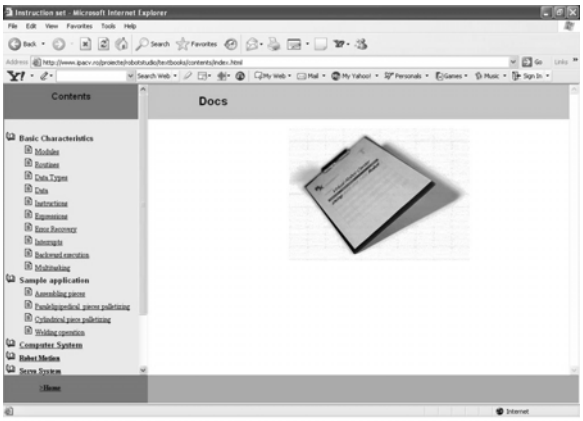

Fig. 4. The GUI interface for the topics “Docs”.

The “Docs“ topics presents basics information for the user. Before using the compiler, it is necessary for any user to enrich his knowledge by reading about the robot controller, the robot programming language and communication protocols, how to make a module or a routine, how to combine them into a program, how to read and analyze the programming instructtions and the complete examples presented in this section. The information is enough in order to offer support for a user with medium skills in this domain. Because the use and the programming is limited by the real robot operational space, the user has to know more information about: the coordinates of robot movement, the position of the table with pneumatic devices which are placed facing the robot, the pieces type and their position on the table, the buffers where the raw pieces are gathered and the way the pneumatic extractors act. This section also presents the most useful instructions of programming (for moving, for activating and deactivating sensors, for logical, arithmetic, assigning, conditioning, looping, selection etc.), together with examples of programs. All these information (Figure 4) are gruped in the folowing 10 chapters: Basic Characteristics (including the sections: Modules, Routines, Data Types, Data, Instructions, Expressions, Error Recovery, Interrupts, Backward Execution and Multitasking), Sample Applications (including sections: Assembling pieces, Paralelipipedical pieces palletizing, Cylindrical pieces palletizing and Welding operation), Robot Motion, Servo System, Robot Structure, Manipulator, Teach Pendant, Controller, Communication Protocols and Application Environment.

For example, the chapter “Sample Applications“ presents simulated and interactive applications with basic operations: palletizing of two types of pieces (a parallelepipedical piece and a cylindrical piece), the assembling of these two types of pieces and the welding-line and welding-point operations in an attractive and explicit form for the reader. The user can analyze the instructions in the program during its execution and the way of activating and deactivating the sensors used in the run process. So, the user can stop the execution and even run step by step the execution in order to understand how the program is created. In these tutorials we can also find interactive applications with the user where he can choose the type of the piece that needs to be transformed and the place on the pallet where the piece is going to be placed. He can also analyze the instructions in the program and the activating and deactivating the sensors during its execution (Comsa *et al.*, 2003).

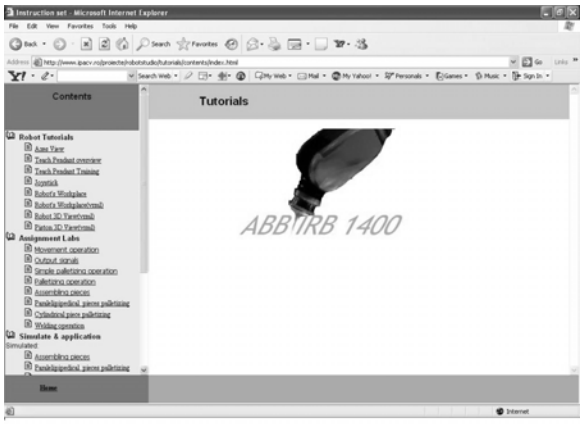

Fig. 5. The GUI interface for the topics “Tutorial”.

The “Tutorial“ topics (Figure 5) presents 5 chapters: Robot Tutorials, Assignment Labs, Simulate & Interactive application, Robot Movies and Help. Each of these chapters contains other sections. For example, the chapter Robot Tutorials presents dynamic on-line tutorials for robot axes in 3D graphic, a tutorial dealing with the buttons of the Teach Pendant for programming the robot in an attractive and explicit form, and an interactive presentation with the Teach Pendant joystick for the user. To offer useful information, this topics presents so a film with the execution in the real lab (Figure 1) and with information about the application components. We can also learn about how to integrate a module into a program, about the set of useful instructions in an explicit presentation with sound, images and exam-

ples of execution of linear and circular movement instructions in a 3D form and so, the assignment of values in a variable. The lab assignments are presented explicitly with sound and images for the user to understand the theme. The lab assignments are:

- Movement operation (by moving the tool of robot from Home position to a specified point)
- Output signals (set and reset output signals using RAPID instructions)
- Simple palletizing operation (by grabbing a piece and move it to a specified position)
- Palletizing operation (by grabbing the pieces and move its on specified positions)
- The assembling operations (by assembling these two types of pieces we will obtain a new product. Flowing from the parallelepipedical pieces is supplied by the first two workstations and flowing of the cylindrical pieces from the last two workstations. All the pieces must be placed in specific positions, knowns by the robot. This two types of pieces are taken one by one by the robot and placed into the assembling place. Because of position's errors, it's necessary for the robot to do two supplementary movements to arrange all the pieces on the assembling place. After that, the robot puts the new product in a matricial storehouse with four locations.
- The palletizing operation of parallelepipedical pieces (all the pieces must be placed successively in a matricial storehouse with four locations).
- Palletizing cylindrical pieces (flowing of the cylindrical pieces is supplied by the workstation number three and workstation number four. All the pieces must be placed in a matricial storehouse with eight locations.
- The welding operation (to do this operation we use storehouses (or Station) 1 and 2, which are supplied with parallelepipedical pieces. The robot take these pieces one by one and puts them together in the assembling place. After that, with a welding device that is taken from the special support created for the welding device, the robot simulates welding-line and welding-point operations. When the operation is finished, the robot puts in the support the welding device and the piece in the matricial storehouse with four locations (in order 1, 2, 3, 4).

There are also presented programming examples for the applications of palletizing, assembling and welding. They differ from the lab assignments by the positioning of the pieces in another order.

Activating the section Robot Movies, the user can see a set of 4 movies with the real robot applications (Cylindrical Palletizing, Paralelipipedical Palletizing, Welding, Assembling).

In the chapter "Help", the section Robot's Workplace is intended to provide a simulation tool in which the user can preview his written programs. After successfully loading the operating environment and the program to execute, just pushing the RUN button (Figure 6) will start the program simulation, current instruction being shown on screen. For greater customization, the user can define it's own environment space, containing various objects defined inside the editor. Those objects can have different properties, like a piston that makes a push operation or a parallelepiped piece with/without holes.

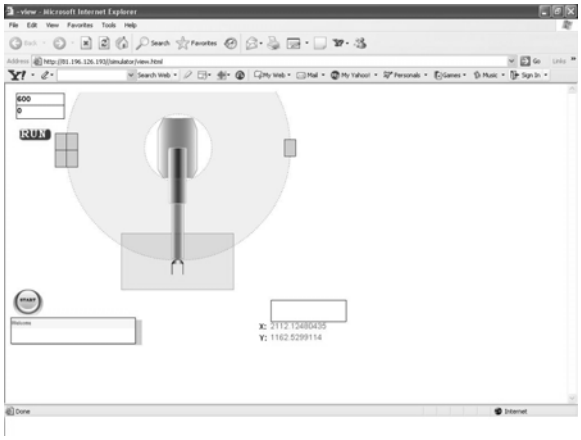

Fig. 6. The GUI interface for the topics "Simulation".

The third topics presented in the ViReC GUI is "Exercises". The user opens this window in the same time with the compiler and can edit a program. Once the program edited, the user opens the option "Run" of the compiler (Figure 7) to see if the program has or has not errors.

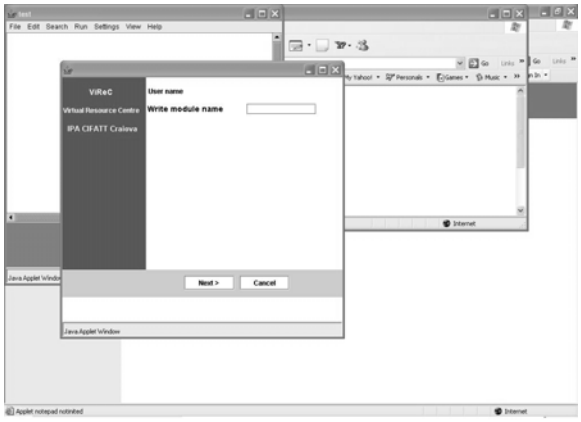

Fig. 7. The GUI interface for the compiler.

If errors are observed in the program, the compiler shows in a bottom window the number of the line in the program where the error is and helping explanations to make it right. The running of the program can be made slowly or step by step. The user can stop the execution and can pass to the next line, before the initial line, or can continue the execution from the initial line, if wanted. During the running of the program the user can visualize the stocked values in registers**.** It can be monitoring the activating and deactivating of some inputs and outputs pressing the number of the output that needs to be visualized. After the program being checked for errors, the execution can be seen by pressing the soft button

"View Result". After pushing this button, a window in which the execution is visualized in 2D graphics is opened. The execution can be visualize from any angle by a simple rotating of the mouse.

The last two topics presented in the GUI interface, "Online Support" and "Plugins", are introduced to offer auxiliary E-facilities for a user: chat, links to on-line appropriate courses or software.

4. TECHNICAL DESCRIPTION OF ViReC

Concerning the software support using to create the site ViReC, we can note in brief:

- DataBase. MySQL is an open source relational database management system (RDBMS) that uses Structured Query Language (SQL), the most popular language for adding, accessing, and processing data in a database.
- Compiler. The compiler is developed using Java technology. An applet is a small Internet-based program written in Java, a programming language for the Web, which can be downloaded by any computer. The applet is also able to run in HTML. The applet is usually embedded in an HTML page on a Web site and can be executed from within a browser. Each user can save his work in database with public and private access. Software offer possibility to load components in environment to develop more applications.

5. SOME TECHNICAL DETAILS

The code behind simulation is made as follows:

- initialization code
- run-time code
- clean-up code

In the initialization phase, there are two different operations. First, the corresponding operating environment is read from the robot database. This is done through a service based on Apache Web server and PHP scripting language with MySQl Support. Below is shown how the operating environment is read using a sample script (getenv.php):

```
//Make the database connection using $databaseserver,
$databaseuser and $databasepass variables
$conn =
@mysql_connect($databaseserver,$databaseuser,$database
pass);
// Select the database
$database = mysql_select_db($database,$conn);
// The query text (Select all from the products table in our
database)
$select = "SELECT * FROM environment where
user='$_POST[user]' and env='$_POST[env]'";
// Make the query
$result = mysql_query($select);
// Count the rows (total result)
$rows = mysql_num_rows($result);
// For each result
while($list = mysql_fetch_array($result)){
   // Set some variables, using the info get from the
database
   // $list is the array that contains the Db values
   // For example $list[&amp;quot;id&amp;quot;] is the value of an
'id' field in the Db
   $id = $list["id"];
   $object = $list["object"];
   $x = $list["x"];
   $y = $list["y"];
   $z = $list["z"];
   //Print in the browser window a string, in a format that
Flash can read
   print("Total=$rows&Object$id=$object|$x|$y|$z&");
}
mysql_free_result($result);
```

In this way the objects from the environment are passed to the simulator, using the requested environment "$env" defined by user "$user". Then, the simulator decodes object data and interprets it for use. This is done in the _root.onLoad function, which is called when the simulator loads.

```
env = new LoadVars();
env.load("http://localhost/proiecte/robotstudio/getenv.php")
;
 env.onLoad = function (success)
             {
  for (this.a = 1; this.a <= this.total; this.a++)
                {
 this["object" + this.a] = this["Object" + this.a].split("|");
_root.robot.attachMovie(this["object" + this.a][0], "object"
+ this.a, this.a);
 _root.robot.attachMovie("piston", "piston" + this.a,
this.a+50);
 eval("_root.robot.object" + this.a)._x =
parseFloat(this["object" + this.a][1] / aspect);
  eval("_root.robot.object" + this.a)._y =
parseFloat(this["object" + this.a][2] / aspect);
eval("_root.robot.object" + this.a)._rotation = -
Math.asin((eval("_root.robot.object" + this.a)._x) /
Math.sqrt((eval("_root.robot.object" + this.a)._x) *
(eval("_root.robot.object" + this.a)._x) +
(eval("_root.robot.object" + this.a)._y) *
(eval("_root.robot.object" + this.a)._y))) * 180 / 3.141593;
  eval("_root.robot.piston" + this.a)._x =
parseFloat(this["object" + this.a][1] / aspect);
  eval("_root.robot.piston" + this.a)._y =
parseFloat(this["object" + this.a][2] / aspect)+30;
  eval("_root.robot.piston" + this.a)._rotation = -
Math.asin((eval("_root.robot.piston" + this.a)._x) /
Math.sqrt((eval("_root.robot.piston" + this.a)._x) *
(eval("_root.robot.piston" + this.a)._x) +
(eval("_root.robot.piston" + this.a)._y) *
(eval("_root.robot.piston" + this.a)._y))) * 180 / 3.141593;
 this["object" + this.a].splice(4, 1);
} // end of for
};
```

The program is read in a similar way, with according modifications. When the user presses the "run" button, the simulator will take the first instruction and execute it. Depending on the instruction mnemonic, the simulator runs the specific function in order to complete the program. Considering a "Movel"

instruction, the simulator will call the "Movel" function. When the function ends its opera-tion, the simulator extracts the next instruction and fetches it, and so on. After the last instruction is executed, the simulator stops and waits for another simulation to be run. Below is shown how the instructions are fetched and the specific functions are called:

```
with (_root)
{
 if (go)
 {
 if (move == false && i < program.total)
                {
                        i++;
                        message.scroll = i + 1;
                        instruction = program["object" + i][0];
 if (instruction.substr(0, 4).toLowerCase() == "move")
                        {
  x1 = parseFloat(program["object" + i][1]); y1 =
parseFloat(program["object" + i][2]);
                           x = x0;
                           y = y0;
 trace("Moving to [" + x1 + "," + y1 + "]...");
                           move = true;
                        }
                        else if (instruction.substr(0,
3).toLowerCase() == "set")
                        {
 trace("Setting [" + program["object" + i][1] + "]");
   _root.robot.close_gripper();
                        }
                        else if (instruction.substr(0,
5).toLowerCase() == "reset")
                        {
 trace("Resetting " + program["object" + i][4]);
 _root.robot.open_gripper();
 } // end if
 else if (instruction.substr(0, 4).toLowerCase() ==
"waittime")
 {
 trace("Waiting for " + program["object" + i][1]+"
miliseconds")
 stop(); _root.myTimer = setInterval(_root.wait,
program["object" + i][1]);
                        } // end if
                           //.....
//various other instruction definitions and/or functions
              //.....
                   } // end if
                } // end if
              } // End of with
```

6. CONCLUSIONS

Virtual laboratories represent now an important and modern educational tools and solutions that bring together geographically distant research groups, allowing them to share data, documents, video and audio presentations, while integrating their computational and laboratory resources. Among the many benefits of virtual laboratories, the following are particularly very important:

- Resource sharing becomes a reality, improving the utilization of costly equipment
- Easier access to educational and research material is provided to students and professional training courses
- Scientific investigation standard can be established in areas where practical experimentation is a required part of research
- Reduction in travel time leads to productivity enhancement.

The recent created virtual laboratories, as ViReC, are implemented as a new generation communication service, not as a simply Worldwide Web application. So, they employ a sophisticated access framework, a communication infrastructure able to support multimedia flows and a component-based software construction. As the Internet is turning into a truly multi-service network with a steady increase in bandwidth and decrease in response time, the environment becomes more suitable for implementtations such as Virtual Laboratories.

Furthermore, it offers various communication channels such as gestures, voice, and text chat, that help learners to interact and cooperate with each other.

REFERENCES

Comşa C., M. Nițulescu, R. Mitrică and G. Vlăduț (2003). E-Laboratory solution for training in virtual robotics, *Annals of the University of Craiova – Electrical and control engineering series*, 27/1, pp. 276-283.

Hamburg I. (2003). Verteilt und doch gemeinsam lernen. In: *Wissenschaftszentrum Nordrhein-Westfalen: Das Magazin*, **14**, H. 1, S. 34.

Hamburg I., O. Cernian and T. Herbert (2003a). Blended learning and distributed learning environments. In: *Proceedings of 5th International Conference on New Educational Environments*, pp. 197-202, Lucerne, Switzerland.

Hamburg I., O. Cernian and T. Herbert (2003b). Lernen und Kooperieren in verteilten Umgebungen: die Chance für die betriebliche Weiterbildung! In: IT-basierte Lernformen für die betriebliche Weiterbildung. *Gelsenkirchen: Inst. Arbeit und Technik*, S. 45-55.

Nițulescu M., A. Căpitănescu and C. Comșa (2002). Integrarea robotului ABB IRB 1400 într-o celulă flexibilă de fabricație cu scop didactic. In: *Proc. of Conferința Națională de Robotică*, pp. 155-160.

http://news.bbc.co.uk/1/hi/sci/tech/1111654.stm.BBC News Article – Virtual Lab brings science to life.

http://physicsweb.org/resources/Education/Interactive_experiments/ physicsweb.org. Virtual Interaktive Experiments.

www.sci.brooklyn.cuny.edu/-marciano.Virtual Multi-Media Internet Laboratories.

Waller J.C. and N. Foster (2000). Training via the web: a virtual instrument. In: *Computers and Education*, **35**, pp. 161-167.