Web-based Remote and Virtual Programming Console of the V^+ Robotic System

Manthos Alifragis, Andreas Mantelos and Costas S. Tzafestas
National Technical University of Athens
School of Electrical and Computer Engineering
Division of Signals, Control, and Robotics
Zographou Campus, 15773 Athens, Greece
Email: manthos@central.ntua.gr; amantelos@ece.ntua.gr; ktzaf@softlab.ntua.gr

***Abstract*—The objective of practical training is a major issue in students education, in many engineering disciplines. The access to specialized technological equipment for education is often limited by specific time restriction, or not provided at all. Therefore, the benefits by providing a web-based platform for remote experimentation via LAN or Internet are evident. This paper describes the development of an e-laboratory platform intending to be used as a distance training system in the field of robotic task planning (e.g. programming of a robotic *pick and place* task). In prior work [10], this platform was evaluated by training students remotely to implement robotic tasks, using the robot's Teach Pendant. This paper is focusing on the design of a training platform, aiming to make students familiar with the V^+ robotic operating system. The proposed platform intends to remotely provide the students with the ability of programming robotic manipulation tasks using directly V^+ scripts. An evaluation protocol, presented in [11], [12], is considered to be employed in the near future, in order to assess the performance of the proposed e-laboratory platform, with respect to the level of students learning and assimilating of the robot's programming language (V^+).**

I. Introduction

Remote teaching in many aspects of technology and engineering has been the focus of considerable research during the last decade. Many teleoperation platforms have been developed in order to implement remote laboratories. The development of such applications has led engineers to remotely operate in realistic way complicated systems such as a robotic manipulator. Thus, tasks like remotely exploring a hostile environment or attending a class lecture from a distance, nowadays are feasible goals. The study presented in this paper emphasizes on a web-based platform that supports a distance learning scenario for robotic *pick & place* application.

Robot teleoperation technologies have been constantly advancing and evolving for more than two decades now [13], [9]. Initial teleoperation systems were deployed in dangerous and hostile environments (e.g. in the nuclear industry for the telemanipulation of radioactive material). The rapid development of communication and networking technologies, together with the advent of new human-machine interactive simulation media (such as virtual reality systems [2]), influenced considerably the field of telerobotics. Research on this field led to the evolution of new concepts and techniques, which have been proposed and demonstrated with great success, such as *predictive displays* [7], *shared-autonomy* teleoperation control [1], or the "hidden-robot" concept [6]. At the development stage of the platform presented in this paper, an effort was made in the direction of adapting such advanced teleoperation concepts, exploring their implementation in remote laboratory settings, such as the one considered in this paper.

A virtual and remote robotics laboratory platform was presented by Tzafestas et al. in [10] and [11], where one of the learning objectives was to enable remote training and familiarization of the students with the basic concepts of a robotic manipulator kinematic structure, which in that case was an AdeptOne SCARA-type robot. The platform enabled students to interact with 2D panels on the graphical user interface (representing top and side views of the robot), to issue a set of *move* commands (each single *move* command consecutively), as part of a general robot manipulation task program. Virtual (3D) representations of the simulated and real (remote) robot were also integrated in this web-enabled platform, as well as live video streaming from the remote robot site. An additional goal of the platform was to support remote training of students in robot task planning using the robot's teach pendant. The training was conducted according to an evaluation protocol [11] [12], using three groups of students : group-I trained the "classical way" on the real robot, group-II trained by controlling remotely the robot through a graphical web interface, and group-III trained by the "virtual robot" of the web based platform, with no remote connection to the real robot. That study described a quantitative way to evaluate learning performance, in relation with mid and high level skills of the students in the field of robot task planning. One significant result was related to the comparative assessment between groups II (remote) and III (virtual), showing that a realistic virtual environment can provide adequate learning elements, compensating the lack of physical presence on the real experimental site.

Additional application examples towards the development of "virtual laboratory" platforms include: (a) the RELAX project [5], (b) the CyberLab network [3], and (c) the eMersion project [4]. The main goal of the above research efforts is to support teaching of the fundamental concepts of engineering through teleoperation.

978-1-4244-1766-7/08/$25.00

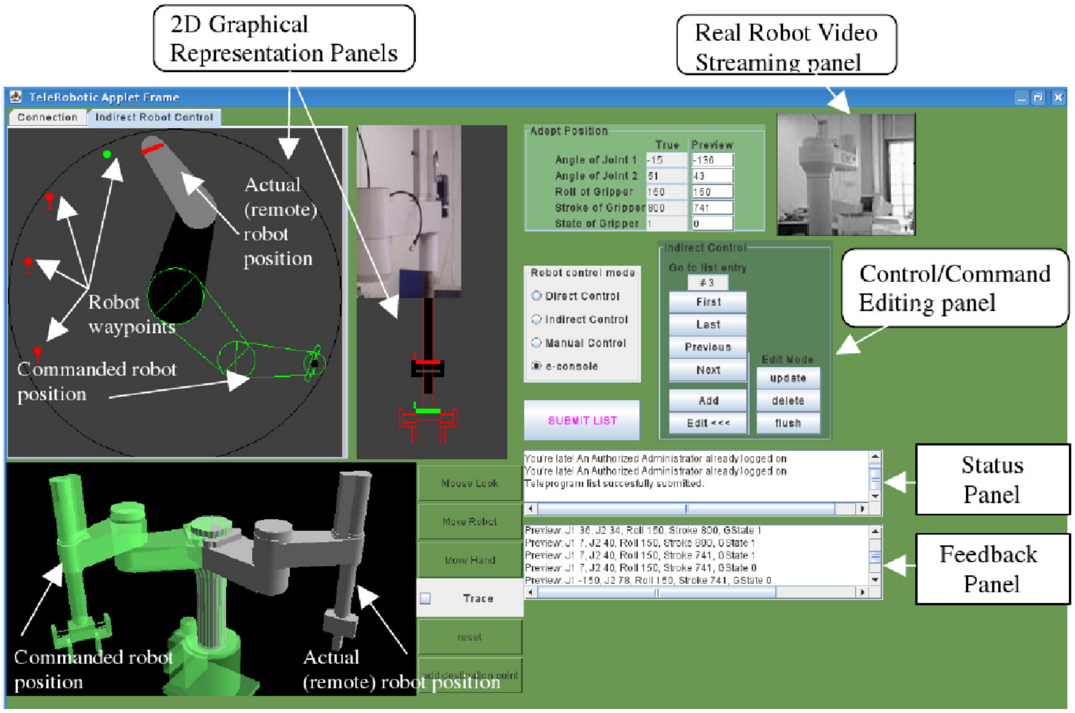

Fig. 1. The graphical user interface of the virtual robotic platform, extended by an additional robot control mode, the e-console functionality.

II. Virtual and Remote Laboratory Platform

The existing web-based platforms for e-laboratory implementation have multiple aims. Firstly, they enable every student to be trained in complicated control systems, such as a robotic manipulator, from a remote place, without the restricting need of real physical access to expensive and/or sensitive equipment. Moreover, such a platform can provide 2D and/or 3D kinematic and/or dynamic motion simulation of the robotic arm, potentially including video streaming feedback from the distant laboratory site, concurrently.

Taking into account these considerations, the focus of the work presented in this paper was to extend the functionality and learning scope of the web-based platform previously developed and presented in [11]. This platform has been further developed to integrate remote script programming modalities, enabling students to train on how to program a SCARA-type Adept manipulator, located at the premises of our robotics and automation laboratory, using the V+ robot programming language, as described in this paper. A brief description of the main components of the platform follows. The development of the graphical user interface was based on Java Technologies, leading to a system that is portable in both Linux and Windows operating systems. The application integrates the following panels (see Fig.1):

- 2D graphical representation panels (top-view and side-view), visualizing both actual and commanded robot's joints configuration.
- a real-time video streaming panel, which is based on RTP protocol and was implemented using JMF, showing the motion of the real robot manipulator.
- an interactive panel providing an exact emulation of the robot's teach pendant.
- status and feedback panels providing real-time textual information on current robot state and robot's joint coordinates.
- a virtual robot panel, implemented using Java 3D API. The panel is providing the 3D visualization of both the commanded (wireframe view) and the current robot configuration.

The architecture of the remote laboratory platform (top view of Fig.2) is based on a server-client model, using multi-thread technologies of Java SDK. The robot server communicates with the robot controller through a serial port protocol. Additionally, it transmits, using a TCP/IP socket connection, the actual joint coordinates data of the real robot, to be viewed by the human operators that are online and authenticated. A specific protocol for real time robot data exchange and multi-user robot manipulation was employed, which has been described more in detail in [10] and [11].

Four remote control modes were developed and are supported by the robot server: (i) direct teleoperation control, with a single *move* command issued by the user (human operator) to the robot, (ii) indirect control, providing robot task planning by utilizing 2D/3D panels, with multiple *move* commands issued by the user, providing the ability of off-line positions sequence editing, (iii) manual control, which is an emulation of the robot's manual control pendant, and was named Virtual Control Pendant. In this paper, an enrichment of the remote robot control modes is proposed, integrating a new telerobot script programming mode, called *e-console* mode. This mode provides local compilation of V^+ script commands and remote operation of the Adept robot manipulator.

The e-console application emulates the functionality of the script programming operations that are provided by the real robotic system, in our case a SCARA-type Adept manipulator with four degrees of freedom. The application is embedded in the *Remote Robotic Laboratory Platform* [10], as an alternative Robot Control mode. Specifically, the system emulates robot programming using directly interpreted commands of the V^+ robotic language, enabling the human operator to directly program the robot to perform simple manipulation tasks. The key idea behind the development of the e-console mode is to train the students for text editing of robot programming code directly, and to use their own code created on the e-console platform to remotely perform robot control tasks with the Adept manipulator. Based on these considerations, the development of this robot control mode and its integration on the platform was designed to resemble as closely as possible a real-world scenario, as a human operator would have to cope with in physical interaction with the real robot.

III. E-CONSOLE: DESIGN AND IMPLEMENTATION

The e-console was designed for training engineering students to gain robot manipulator programming. This paper presents the first version of the e-console, a teaching tool for robot motion programming in forward and inverse kinematic mode. The application development was based on Java technologies and the in-built *RegEx* package [8] of the JDK version 1.7. The architecture of the e-console (see bottom part of Fig.2) consists of the *KeyControl* class, the *EditorGui* class, the *Compiler* class, the *MonitorCommands* class, the *AdeptCom* class and the *Interface* class.

The *KeyControl* class monitors continuously the keyboard input, issued by the operator. The *EditorGui* class, which is the core of the application, interacts with the human operator and serves its requests by accessing the appropriate subclass. The script that the user creates through the e-console application is compiled by the *Compiler* class, according to the V^+ regulations and syntax rules. In the event of a successful compilation, the script can be sent to the real Adept robot interacting with the main *Remote Robotic Laboratory Platform*. In case of a syntax error, the e-console reports to the students indicating the line of the script that generated that error. The *MonitorCommands* class simulates the real Adept V^+ monitor commands that the user provides to the e-console for the creation, editing, or deletion of the script file. The e-console application as an embedded module of the *Remote Robotic Laboratory Platform*, has a continuous robot position feedback in terms of joints and end-effector coordinates. The *AdeptCom* class codes the V^+ script commands according to the established communication protocol between the *Remote Robotic Laboratory Platform* and the Adept manipulator. The *Interface* class provides a set of methods, which, based on their functionality, are implemented by the *Compiler*, the *MonitorCommands* and the *AdeptCom* classes. It was noticed during experimentation that this architectural approach reduces computational execution time and can be easily expanded to integrate more V^+ commands.

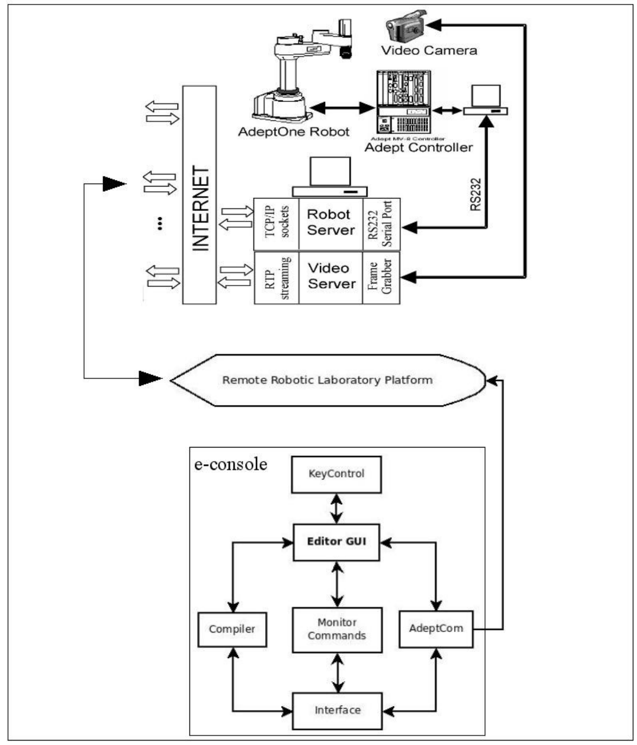

Fig. 2. Overall system architecture: the e-console application is embedded in the Remote Robotic Laboratory Platform.

The e-console interacts with the 2D/3D panels of the *Remote Robotic Laboratory Platform*, in terms of robot position feedback. This creates a seamless integration of the e-console functionalities within this platform, enabling the human operator to evaluate script code by interacting with the 2D/3D panels, while a video streaming feedback of the actual robot is supplied by the platform. Complete integration of the e-console within the *Remote Robotic Laboratory Platform* also offers to the human operator the option of using the virtual pendant from the main platform, in order to supply the joint coordinates and the gripper state to the e-console application.

IV. E-CONSOLE: USER INTERFACE

The e-console application, as mentioned before, is based on Java technologies and integrates the following Java GUI components controls (see Fig.3):

- an input text field for the monitor commands provided by the user.
- an output text field displaying the last issued V^+ command.
- a status bar indicating the current operating status of the application
- an output text field reporting the current joints angles or end-effector coordinates including gripper state.
- the main input text field, which emulates the embedded *SEE* editor of the V^+ real time operating system.

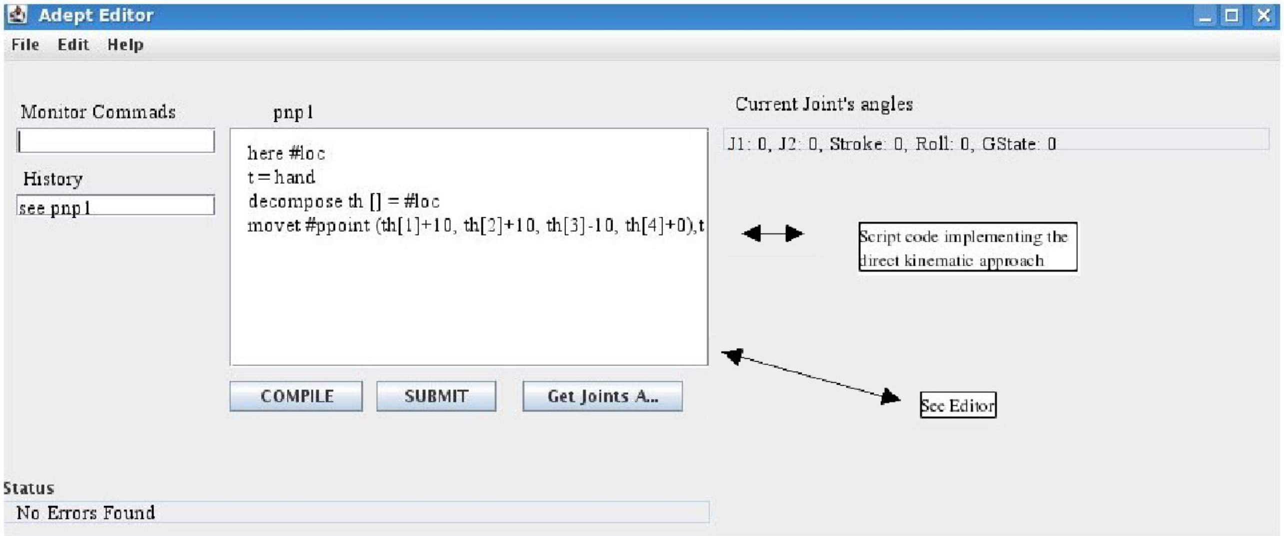

Fig. 3. The e-console emulates the V^+ computer based control system and computer language. The above script implements the direct kinematic approach executing the appropriate commands.

- the *Compile* and the *Submit* button, and
- the *Get Joint Attributes* button, that polls through the graphical representation of the robot, the current joints angles or end-effector coordinates.

The graphical user interface emulates the V^+ computer-based control system and programming language, designed specifically for the Adept Technology Industrial robots. At the current stage of the e-console platform development, a set of monitor and robot motion commands have been implemented. As in the real V^+ control system, specific keyboard shortcuts are embedded to facilitate training of the user in realistic conditions. A walkthrough of a typical training session follows.

The user provides the appropriate monitor command, *SEE*, in order to create a new script file. In the main input text field, the user creates a script file to program the desired robot manipulation task. According to the requirements of the training task, the application supports two groups of instructions for robot motion control. The first group contains the instructions intended to move the robot in a direct kinematic mode (see Fig.3, script code), while the second one employs the robot's inverse kinematic model. The direct kinematic motion control mode accepts as an input the joint angles, in order to move the robot at the specified position (see top Fig.4). The user provides values for the coordinates of all the robot joints. The *movet #ppoint* instruction drives each joint by updating its angular displacement with the new value, provided by the user together with an optional offset. The inverse kinematic mode, on the other hand, accepts as input the coordinates of the target position of the end-effector (see bottom part of Fig.4). The user provides the end-effector position coordinates, and through the *TRANS* instruction, the robotic system transforms the given values into joint angles.

The *Get Joints Attributes* button must be accessed prior to any script code compilation, in order to update the values of the joint angles. Both of the above robot control modes need a set of robot parameters to be defined. Towards this direction, the *here #loc* instruction assigns the current joints angles into a variable named *loc*, and the *t = hand* command assigns the gripper state into a variable named *t*. Finally, the *decompose* instruction maps the joint angles to an array.

After completing the script code editing, the user can locally compile this code; the status bar then provides information about the result of this compilation. A successful compilation allows the user to submit the program through the *Remote Robotic Laboratory Platform* for virtual preview (2D/3D graphics representation), in case a simulation mode is selected, or to the real robot for actual execution, in case of direct teleoperation. Multiple *move* commands are supported at the current version, in order for the student to accomplish a simple robotic manipulation task, such as a pick & place task. It is important to point out here that this e-console robot script editing mode is integrated within the indirect robot control mode of the *Remote Robotic Laboratory Platform*, thus providing additional support for off-line editing of the intermediate control points, before the sequence of *move* instructions is actually submitted to the robot. The student is thus able to access dynamically the joints and the end-effector coordinates during programming process, by means of the virtual pendant or through graphical interaction, which provides augmented capabilities than real programming on the robot site.

The e-console application is capable of teaching/training the students on the basic programming and control features and the enhanced operations (task planning) of an (Adept) industrial robot. The students become familiar with the syntax

```
here #loc
t = hand
decompose th [] = #loc
movet #ppoint (th[1]+10, th[2]+10, th[3]-10, th[4]+0),t
```

```
here #loc
t = hand
decompose th [] = #loc
SET target  = TRANS (+400,-200,+830,0,180,0)
 MOVE target
```

Fig. 4. Initially both scripts read the current joint positions of the robot. Top: It is implementing the Direct Kinematic approach and causes joint interpolation motion with an adding offset value at each joint. Bottom: This script code implements the Inverse Kinematic approach. The *TRANS* instruction returns a transformation value computed from the given X, Y, Z end-effector position and y, p, r orientation.

and the logical rules of the V^+ computer-based real-time control system. The application is able to perform a virtual representation of the user code, within the graphical user interface context of the *Remote Robotic Laboratory Platform*. The virtual representation of the robot program created by the user, which is a process taking place locally, allows the operator to tune the code according to the graphical motion modelling of the robot. From a technical point of view, the e-console platform, embedded within the overall framework of the *Remote Robotic Laboratory Platform*, offers students the ability to familiarize themselves with the programming process of a robotic manipulator, without actually being in physical contact with a real robot. This is achieved realistically by emulating how actual robot programming operations take place on the Adept SCARA-type robotic system.

V. Conclusion

This paper describes the development of an e-laboratory framework in the field of robotics. The platform in its current configuration is designed to offer distance training, including programming remote robot manipulation tasks with a Virtual Teach Pendant (see [12]) and creating remote robot command scripts in the V^+ programming language in a cooperative mode, planning and implementing robotic tasks like a pick & place operation. The type of the evaluation protocol is currently under investigation, where the goal will be to measure (both quantitatively and qualitatively) the effective knowledge transfer from the e-laboratory platform to the students, regarding basic robot programming principles. The framework intents to prove, in the near future, that distance learning with the interaction of new media, 2D/3D graphics representation of the manipulator, live video streaming and the e-console, can compensate for the lack of direct physical presence on the real robot site. This framework can potentially be a useful tool to remotely teach the students how to perform robotic operations usually employed in the industrial area. In addition, the students become familiar with the constraints of a real robot operation, particularly as related to designing and implementing robotic manipulation task planning operations. The integration of the e-console within the remote laboratory platform constitutes a valuable add-on component for the training system, in the sense that it enables the students to obtain an intutive visual representation of the geometric robot constraints, directly linked to the operations performed during robot task planning and scripting of a motion program. All these learning elements, complemented of course by a series of lectures on the theoretical foundations of robotics, can constitute a complete teaching module forming an introductory course on robotic manipulation.

Acknowledgment

This work was supported by grant ΠΕΝΕΔ-2003-ΕΔ865 [co-financed by E.U.-European Social Fund (80%) and the Greek Ministry of Development-GSRT (20%)].

References

[1] B. Brunner, G. Hirzinger, K. Landzettel, J. Heindl, “Multisensory shared autonomy and tele-sensor-programming - key issues in the space robot technology experiment ROTEX,” *In Proc:IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS93)*, Yokohama, July 26-30, 1993.

[2] G. Burdea and P. Coiffet. “Virtual Reality Technology. John Wiley,” 1994.

[3] T. I. Eikaas, B. A. Foss, “CYBERLAB - A Global Remote Laboratory Experimentation Network,” *AlChe 2003 Annual Meeting, Session 501: New Technologies For Experimentatation Over The Internet*, November 17, 2003, San Francisco, CA.

[4] D. Gillet and G. Fakas, “eMersion: a new paradigm for web-based training in mechanical engineering education,” *in Proc. Int. Conf. Engineering Education*, Oslo, Norway, Aug. 2001,pp. 8B4-10-8B4-14, Session 8B4.

[5] B. A. Foss, K. E Malvig, and T. I. Eikaas, “Remote experimentation-New content in distance learning,” *in Int. Conf. Engineering Education (ICEE’2001)*, Oslo, Norway, Aug. 6-10, 2001, pp.8D1-1-8D1-6, Session 8D1.

[6] A. Kheddar, C. Tzafestas, P. Coiffet, T. Kotoku, K. Tanie, “Robot Teleoperation Using Direct Human Hand Actions,” *International Journal of Advanced Robotics*, Vol. 11, No. 8, pp.799-825, 1997.

[7] A. K. Bejczy, W. S. Kim, S. Venema, “The Phantom Robot: Predictive Displays for Teleoperation with Time Delay,” *IEEE Int. Conf. on Robotics and Automation (ICRA90)*, 1990.

[8] Mehran Habibi, *Java Regular Expressions: Taming the java.util.regex Engine*, Apress, 2004

[9] T. B. Sheridan, “Telerobotics, Automation and Human Supervisory Control. The MIT Press,” Cambridge, USA, 1992

[10] C. S. Tzafestas, M. Alifragis, N. Palaiologou, S. C. Thomopoulos, M. Brahman, and A. E. Exarchou, “Development and experimental evaluation of a remote labaratory platform for teaching robot manipulator programming,” *in Proc. Int. Conf. Engineering Education (ICEEE’2004)*, Florida, Oct. 16-21, 2004.

[11] C. S. Tzafestas, N. Palaiologou, M. Alifragis, “Experimental evaluation and pilot assessment study of a virtual and remote laboratory on robotic manipulation,” *in Proc. IEEE Int. Symp. Industrial Electronics (ISIE’2005), Dubrovnik, Croatia, Jun. 20-23, 2005, pp. 1667-1683*, Special Session on: Science and Engineering Distance Learning.

[12] C. S. Tzafestas, N. Palaiologou, M. Alifragis, “Virtual and Remote Robotic Laboratory: Comparative Experimental Evaluation,” *IEEE Transactions on Education*, vol. 49, no. 3, p.360-369, August 2006.

[13] J. Vertut and P. Coiffet, “Les Robots: Tlopration. Tome 3A: Evolution des technologies. Tome 3B: Tlopration assiste par ordinateur.” Edition Hermes, Paris, 1984