ROCON - A Virtual Construction Kit, Visualization Tool and Remote Control System for Mechatronic Devices

J. Kaiser and Thomas Fries
Department of Computer Structures
The University of Ulm, Ulm, Germany

kaiser@informatik.uni-ulm.de

Abstract: ROCON (RObot visualization and CONtrol system) is an integrated virtual construction, visualization and control tool for complex mechatronic devices ROCON allows to built virtual robots from geometric elements connected by rotational and linear actuators. It also includes the facility to define sequences of motion patterns to explore the complex mechanical constructions. Additionally, ROCON enables the control of a physical equivalent by generating the necessary control signals derived from the simulation. This can be exploited for remote robot control. Sensor information from a real robot which is fed back to the visualization system supports the presentation of a realistic view of the robot, particularly concerning orientation in space which can not be derived from the visualization only.

Keywords: 3-D visualization, augmented reality, remote robot control.

Introduction

Complex mechatronic devices like legged robots, robot arms and sophisticated gripper devices are usually composed from multiple geometric elements connected via flexible joints [1, Paul], [2, Featherstone] . These joints may be implemented as mechanical actuators like electric, pneumatic or hydraulic servos. Rapid prototyping and testing of the general movement of these devices is difficult because of the complex mechanical design. Additionally, due to the many degrees of freedom, the movement of these devices are difficult to control. The high mechanical forces generated by the servos may lead to a certain probability of mechanical destruction and therefore it may be desirable to test the complex control algorithms and motion patterns in virtual reality in which it is possible to observe the behaviour and detect collisions of the moving parts. Once tested, these sequences could later be used to control the real actuator.

When controlling the actuators of a robot, a visualization of the actuation is highly beneficial. Firstly, it can be used to support the operator which can directly manipulate the 3-D image by an adequate input device and derive the control signals from the animation. This is, of course, particularly important for remote control where the robot can not be observed directly. In such an application, a 3-D animation can show the status and the actions of the remote robot where a video camera would cause problems, e.g. if the bandwidth of the control channel is very low or the robot operates in opaque fluids or dark environments. Also it is not always possible to position an camera adequately outside of the robot. Finally, in a 3D-representation, the perspective can be chosen freely and any angle of view can be generated which is also not possible for a camera. However, a visualization needs feedback from the real robot to match the calculated and the real conditions. Particularly, orientation in the real world can not be derived reliably from a simulation. Therefore, to obtain the real orientation and actuator positions, appropriate sensor feedback from the physical device is needed like inclination and direction in some system of coordinates. The ROCON system addresses these needs. It provides:

- a construction and visualization tool to design a virtual robot,
- the management of an extensible construction kit of reusable elements
- the facility to define motion sequences and detecting collisions,
- the ability to control a physical robot remotely by exploiting sensor feedback.

ROCON is completely written in Java except the low level parts of the communication and the instrumentation interface.

ROCON Architecture

Fig 1 depicts the components of the ROCON architecture. The core of ROCON is the internal

object model which represents all elements and their relationships. The internal model will be further described in chapter 3. The graphical user interface (GUI) visualizes the internal model. There is a rich set of functions to construct and manipulate the geometric representation of the robot device. Fig 2 shows a screenshot of ROCON in the manipulation mode, where it is possible to pick geometric blocks of the virtual device and move them (currently with an ordinary mouse). As is described in the next chapter, all blocks which are connected to a selected one are also moved accordingly.

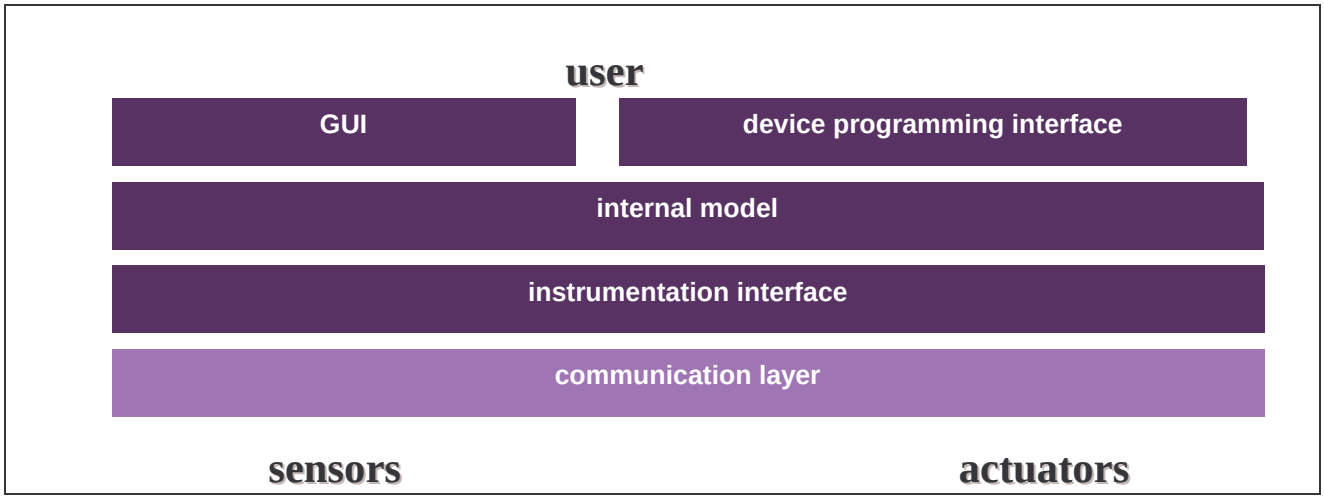

Fig. 1 The components of the ROCON architecture

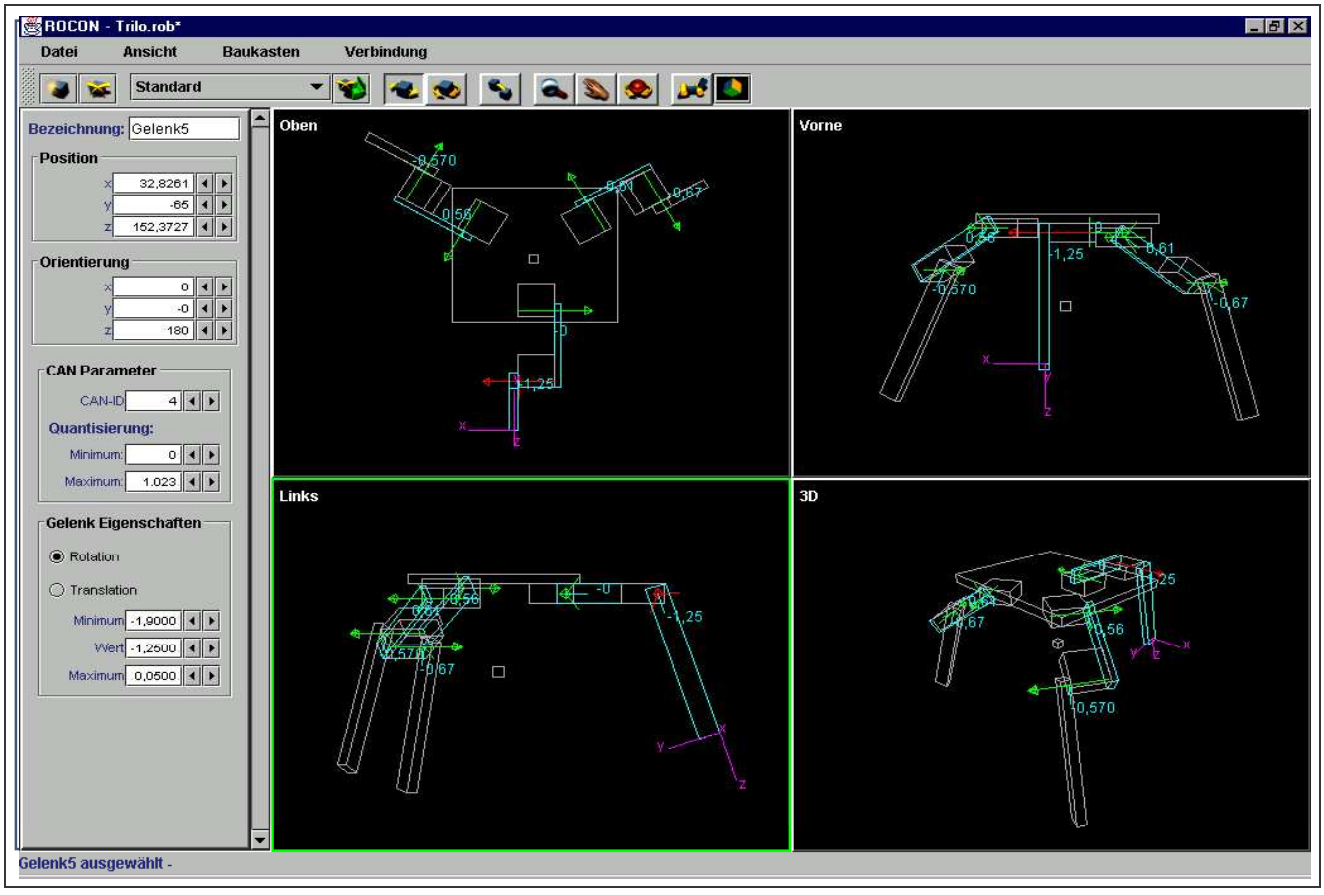

Fig. 2. Operator Screen of ROCON

It is also possible to point to joints in which case the characteristics of the joints are displayed, e.g. whether it is a rotational or a linear joint, the constraints of movement and the current position.

The device programming interface allows to define motion sequences for the internally represented mechanical device. Instead of using a special script language, this can be done in Java. The programmer can define arbitrary movements within the constraints of geometric rules and the joints which connect the blocks. When the movements are executed, the internal model could be used for collision detection.

From the geometric representation of the internal model the instrumentation interface layer calculates the respective control signal for the real robot. Some of the basic parameters like the granularity of movement are also adjustable on-the-fly from the GUI. In case the real device is not directly attached to the operator host, a communication layer will pack the control signals in respective messages and forward it to the real device. This is detailed in chapter 5.

The Virtual Construction Kit

When considering how to model a complex robot, we have to distinguish two aspects. Firstly, we have to visualize the robot. This requires the model to provide information about the shape of the robot which at least to some extend should reflect the real appearance, although details can be omitted. Examples for such models are the Java3D- Scene Graph [3] and VRML [4]. Secondly, we have to describe the robot in terms of geometric elements related by joints which have positions and certain degrees of freedom of motion in a system of coordinates. This models are often expressed in Denavit-Hartenberg [5] coordinates or equivalent representations. The model which we present to visualize the robot has to integrate both aspects of modelling. The construction tool allows to built a virtual device from rectangular 3-D geometric elements as shown in Fig. 3. We distinguish three basic types of Elements: (geometric) ***element***, ***joint*** and ***anchor***. Element abstracts the physical building blocks and stores attributes necessary for the visualization like point which describe the surfaces and lists of these surfaces to make up a 3-D geometric body. At the moment we only use rectangular building blocks. A point where a connection to another block is possible is marked as an anchor. These anchors provide means to build up a set of reusable parts.

Anchor points represent the relative positions of a part, where other parts can be plugged in. Two parts of the construction kit can be combined by connecting a joint of one part to an anchor of the parent part.

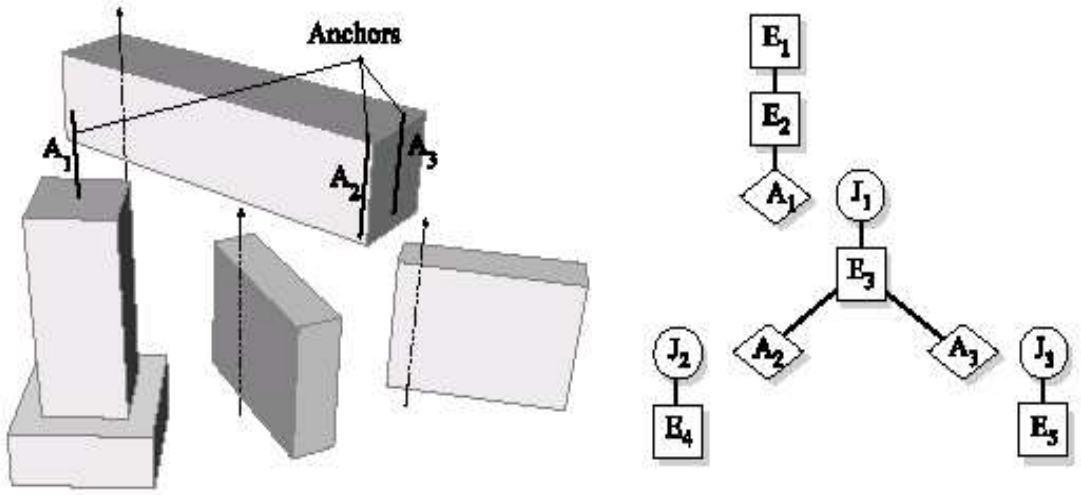

Fig. 3 Elements, joints and anchors: building blocks in ROCON

The relation between the building blocks can be specified as trees (cf. Fig. 3.). A joint has one degree of freedom and may have a circular (rotation) or linear (translation) characteristic. In Fig. 3. only rotational joints are used and their axes depicted. The relations between the construction elements are described in a graph, shown at the right hand side in Fig. 3. The static pillar on which the robot arm will be placed is made of two elements E_1 and E_2 and an anchor A_1. Similarly the building block representing the arm comprises the joint J_1 which will be connected to A_1, an element E_3 and two anchors to connect the gripper elements. Fig. 4 displays the respective robot arm and the related graph.

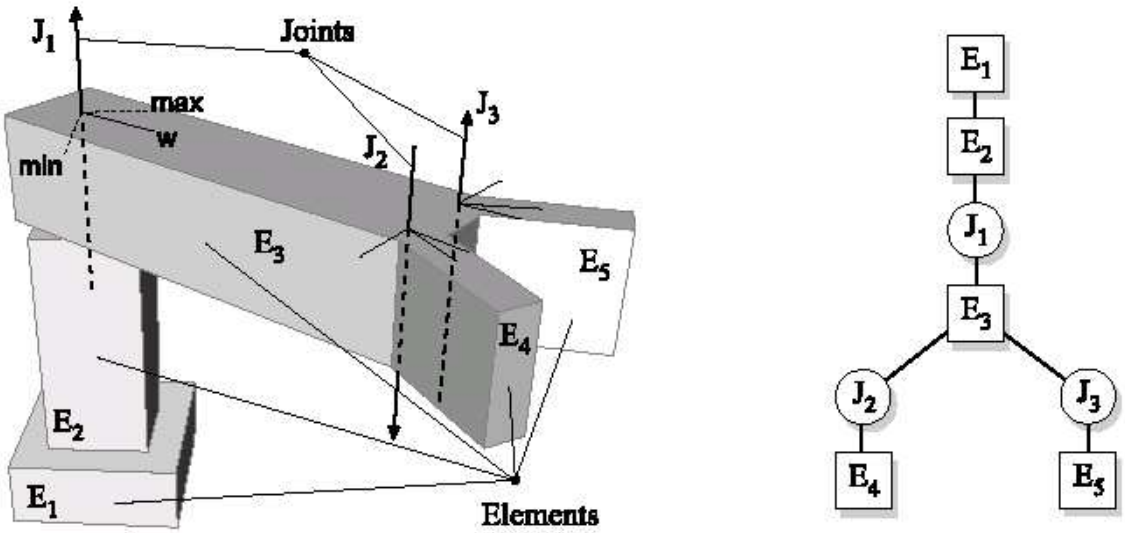

Fig. 4 An example of a virtual robot arm and the related connection graph

As can be seen from the figures, the anchors only define the coordinates where a joint can be attached and are obsolete when the connection is defined. Elements, anchors and joints are represented as Java classes.

There are two systems of coordinates used, world coordinates which refer to an absolute reference in the universe while the object coordinates describes the positions of the building blocks relative to each other. The first is necessary because a device may have an arbitrary position and orientation in space and there may be many independent objects for which it is necessary to display their absolute positions. Object coordinates are used to calculate the position of a geometric block relative to its predecessor in the graph. They also support a hierarchy of movements. When a component down the hierarchy is moved, its movement does not affect components at a higher level, however, vice versa, a motion of a higher level component will obviously affect all components down in the hierarchy.

Animating the Virtual Device

In addition to a direct manipulation by appropriate input devices, the virtual model of the mechatronic device can be animated. It is possible to define sequences of movements for each block according to the specifications of the joints. The model includes for each joint the specification of angle or linear ranges. Each joint can be manipulated within their ranges by an arbitrary java program that accesses the internal ROCON model by an API (application programming interface). A simple java class that implements a specific interface can be loaded dynamically to control the joints of the actual construction. Fig. 5 shows an example of a Java program defining the synchronous movement of the joints J2 and J3 of the example in Fig. 4 which results in a symmetric gripper property. As can be seen from the code, we use an event-based programming model which is particularly suited for control systems. As an extension, the geometric

information of the model can be used to detect collisions along the movement. ROCON does not provide a complete physical simulation of the mechanical devices, taking into consideration masses, acceleration, etc. However, when the physical device is available, ROCON includes the possibility to use sensor feedback from the real device to reflect the physical conditions.

```
1  ...
2  Public class MyControl extends Control {
3
4    Class MyListener implements ValueChangedListener {
5      Joint otherJoint;
6      Public MyListener(Joint otherJoint) {
7        This.otherJoint = otherJoint;
8      }
9      public void valueChanged(ValueChangedEvent thisJointEvent) {
10       otherJoint.setValue(thisJointEvent.getValue());
11     }
12   }                    // init() overrides a method of class Control
13                        // It is called by ROCON after loading.
14   public void init()   // getJoint(x) retrieves the Joint object with
   name x
15     getJoint("J2").addValueChangedListener(new
   MyListener(getJoint("J3")));
16   }
17 }
```

Fig. 5 Example of a motion script

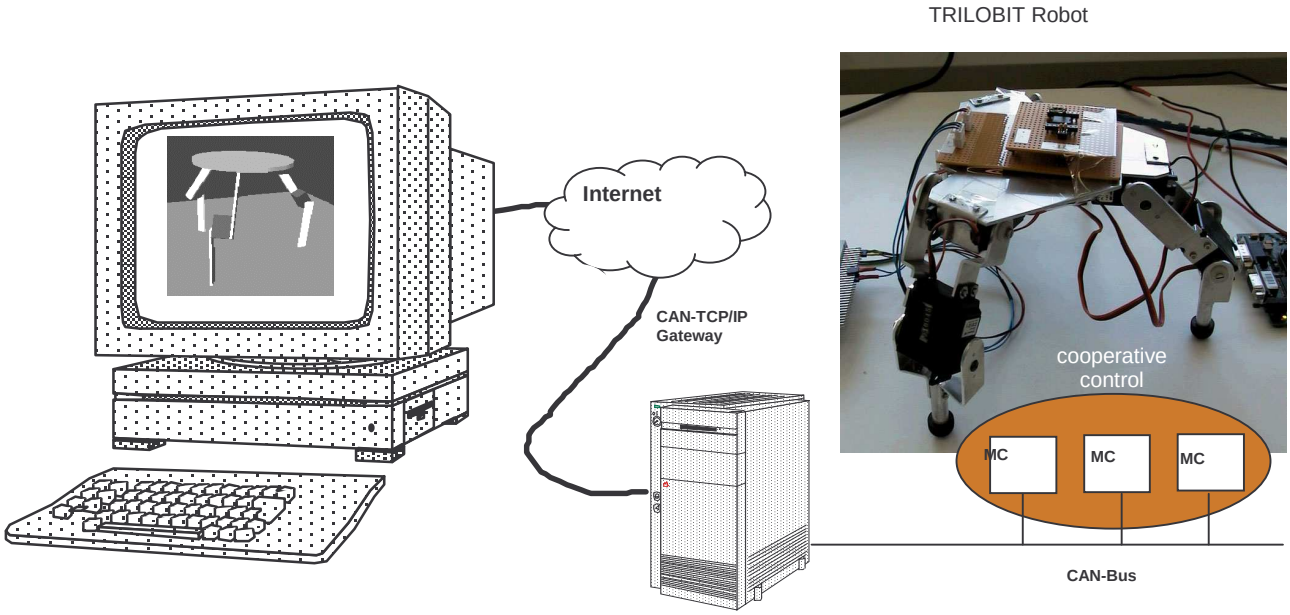

Fig. 6 Remote Control Scheme

Controlling the Robot Remotely

One of the most important features is the remote control capability provided by ROCON. Fig. 6 depicts the overall system. On the local control system the virtual representation of the real device is visualized. In this example, we modelled an active tripot. The operator can e.g. move the table by just pointing to a specific joint and drag it to a certain position. The new values for the joint will be calculated and sent via the internet to the remote host. Here a gateway will feed it into a fieldbus network (CAN-Bus [6]) to which the micro-controllers for the real device are connected.

Because the orientation of the real device in space (i.e. the orientation in the world coordination system), and particularly its orientation relative to gravitation cannot be derived from the simulation, we use a smart acceleration sensor on the real robot and sensor feed back to visualize its actual inclination in a plane. Fig. 2 depicts the operator interface screen with four selected views on the active tripot. The left side includes fields for the position, orientation, joint characteristics and network parameters.

Discussion and Related Work

ROCON combines a simple 3D visualization and virtual construction kit with the ability to derive control signals for a real equivalent of the virtual device directly from manipulating the virtual image and to integrate sensor feedback into the visualization. For each of the specific functions of ROCON there exist similar or better solutions in the CAD, simulation and control area. However, the combination is hardly found in other systems. Driven by the internet, during the last years there appeared a large number of systems which allow the 3-D visualization of complex mechanical devices. VRML97 [4], Java 3D [3] or X3D [12] encourage the modelling and animation of virtual robots. However, neither allow these systems to control a physical robot nor do they support the on-the fly construction of virtual robots similar to a LEGO construction kit. There are also systems which allow to test data developed for a real robot by a simulator before they are used to control the physical device, e.g. [7] [8]. Other systems allow the control of robots via the internet by special control panels [9]. However, ROCON takes another way, it directly derives the control information from the motion of the visualization, not from the motion of the input device or some control panel. Additionally, it is also possible to specify algorithms for robot control and motion patterns in Java to be visualized.

Future Work

Visualization combined with the appropriate sensor feedback can replace a camera controlled remote operation in many cases with a fraction of required bandwidth. Additionally, a virtual representation of a robot, a complex tool or gripper device gives the advantage to take any view point and perspective to observe operation. ROCON was designed specifically with applications in mind which have to cope with low bandwidth communication channels where the control information and the feedback from the sensors do not require large volume data. At the moment, our experimental system has an inclination sensor for its absolute orientation in space and position sensors in the joints. We use smart sensors connected to the CAN-Bus as shown in Fig. 6 and will further integrate tactile and distance sensors to allow a better perception of the environment. The low communication requirements enable a remote control over narrow, loaded and unreliable channels. However, there are applications in which latency may become the major problem. Long latencies can be encountered either when multi-hop connections are required with long delays in the routing nodes, the robot is far away e.g. in space missions, or the communication medium is slow, e.g. for AUVs (Autonomous Underwater Vehicles) where we have acoustic networks with propagation delays of 0.67 sec/km, signal ranges of 10-90 km and a bandwidth of 8-15 Khz [10], [11]. In these applications we firstly need local autonomy of the controlled entities, i.e. they must incorporate some goals, planning and adaptive algorithms. Secondly, we need learning or predictive filters to match the visualization with the (predicted) remote situation to mask the latency of the channel. Future work will include these properties in ROCON.

Acknowledgments

This work was partly funded by the EU in the CORTEX Project under the contract No. IST-2000-26031. (CORTEX: **CO**operating **R**eal-**T**ime s**E**ntient objects: architecture and e**X**perimental evaluation.

References

1. Paul. Paul R.P.: Robot manipulators: Mathematics, Programming and Control, MIT Series in Artificial Intelligence, The MIT Press, 1981

2. Featherstone. R. Featherstone: Robotic dynamics algorithms", The Kluwer International Series in Engineering and Computer Science, SECS 22, Robotics : Vision, Manipulation and Sensors, Kluwer 1987

3. Deering, M. Sowizra H.: Java3D Specification, Version 1.1., Sun Microsystems, 2550 Garcia Av. Mountain View, CA, USA, 1998

4. VRML97 International Standard, ISO/IEC 14772-1:1997

5. Denavit j., Hartenberg R.S. : A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices, Journal on Applied Mechanics, June 1955, 22:215-221, 1955

6. Robert Bosch GmbH: CAN Specification Version 2.0, 1991

7. Speck A.and Klaeren H.: RoboSiM: Java 3D Robot Visualization'', Proceedings of the IECON'99, The 26th Annual Conference of the IEEE Industrial Electronics Society, IES., San Jose, CA, 1999, 821 - 826

8. EASY-ROB: http://www.easy-rob.de/, May 2002

9. Internet Robotics: http://www.keldysh.ru/i-robotics/home.html

10. A networking protocol for underwater acoustic networks, Technical report, TR-CS-00-02, Department of Computer Science, Naval Postgraduate School, December 2000

11. Xie G.G. , Gibson J. : A network layer protocol for UANs to address propagation delay Induced performance limitations, Proceedings of MTS/IEEE Oceans 2001 Conference, Honolulu, HI, November 2001.

12 X3D: web 3D consortium: Extensible 3D, www.web3d.org