Int. j. simul. model. 4 (2005) 3, 105-117
ISSN 1726-4529 Professional paper

3D SIMULATION IN THE ADVANCED ROBOTIC DESIGN, TEST AND CONTROL

Vajta, L. & Juhasz, T.
Budapest University of Technology and Economics, Faculty of Electrical Eng. and Informatics,
Department of Control Engineering and Information Technology,
XI. Magyar tudosok korutja 2, H-1117 Budapest, Hungary
E-mail: vajta@iit.bme.hu

Abstract
In the paper an overview about modern visualization approaches in the field of telerobotics and interactive robotics simulation is presented. The role of 3D simulation in the advanced robotic design, test and control is highlighted.
We present an overview about the advanced graphics techniques in semi- or true 3D. We have made a comparison presenting the advantages and disadvantages between the (binocular) anaglyph and the (monocular) motion stereo methods in our robot simulator application.
(Accepted by previous Editorial Team.)

Key Words: Mobile Robot Simulation, 3D Visualization Methods, RobotMAX Simulator

1. INTRODUCTION

The intelligent mobile robots are widely used in applications such as general indoor and outdoor operations, emergency rescue operations, underwater and space exploration, pipe- and duct inspection in nuclear power plants, construction environments and so on.

Mobile robots use locomotion that generates traction, negotiates terrain and carries payload. Some robotic locomotion also stabilizes the robot’s frame, smoothes the motion of sensors and accommodates the deployment and manipulation of work tools. The locomotion system is the physical interface between the robot and its environment. It is the means by it reacts to gravitational, inertial and work loads.

Generally the mobile robots have some amount of autonomy. This means that they can do programmed tasks while they are responding to the occurring environmental effects automatically. A large proportion of the mobile robots are remotely operated platforms that usually have local autonomy as well. The interactive human control of these telerobots needs advanced 3D visualization using novel graphics techniques.

Typically the remote controlling of a robot requires a virtual model of the given platform that describes its behavior correctly. We can state that telerobotics and robot simulation are usually interconnected research areas. Concerning the current field of telerobotics and robot-simulation in general, one key factor in the human-robot interface is realistic visualization. Although the traditional computer graphics algorithms produce fairly good quality virtual environments, there is no doubt about the need for more realistic (“near true”) 3D impressions. Of course this can lead to interactive, entertaining applications [1] as well. In this paper we will present an overview about the advanced graphics techniques in semi- or true 3D, which could be applied in a modern mobile robot simulation solution.

Although our final aim is real robotics, it is often very useful to perform simulations prior to investigations with real robots. Designing mobile robots (either teleoperated or fully-autonomous platforms) requires a step-by-step approach including the accompanying testing processes as well. The sum of the development, testing and running charges or any other expenses demands utilizing a simulator to cut down overall costs. Simulations are easier to setup, less expensive, faster and more convenient to use. Building up new robot models and setting up experiments usually takes only a few hours. Simulated robotics setups are less expensive than real robots and real world setups, thus allowing a better design exploration. Virtual prototyping through behavioral simulation enables continuous iterative design, allowing the developer of a robotics system to find inconsistencies earlier in the design phase. If we have validated simulation results, we can transfer our procedures onto the real robots. In the second part of this paper we will present our robot simulator project from the visualization side and the design, test and control side as well.

Finally it should be noted, that people in the field of interactive computer graphics use the terms “3-D” in the meaning of a “realistic”-looking image which may be perceived with one eye. This is a confusing nomenclature because most people use the term “3-D,” when they mean a *stereoscopic* image. Such an image requires the beholder to use both eyes. We will use the term “3-D” with both meanings, too.

2. MOBILE ROBOT SIMULATION APPLICATIONS

In general the ideas and techniques developed during the simulation process yield to ideal conditions to raise synergy: a catalytic effect for discovering new and simpler solutions to traditionally complex problems. Using virtual prototyping by means of behavioral simulation reduces the time-to-market, as it shows the inconsistencies early in design phase. Abstract modeling with CAD tools, enhanced conceptual design and moving up life-cycle assessments by virtual prototypes allow devising the optimal architecture of the system.

Today there is a wide variety of simulators on the market, but most of them are trade, model- or application specific ones. A summary was created by Yiannis Gatsoulis (University of Leeds) who was claimed by the CLAWAR community to analyze the state-of-the-art of this field a few years ago. It contains a snapshot of the available software (environment editors, image processing- and control libraries, system simulators, etc.) that are in connection with this research area. It assesses the cost, usability, expandability and rapid development abilities of the given applications. Unfortunately since the creation of this document a lot of projects were suspended or cancelled – as their authors became to deal with other tasks. There are many “bid fair” applications with tall feature lists, almost without any usable, implemented functionality. Considering the amount of available system simulators today – that are really worth trying to use – let us emphasize only the Webots® software system that is still developing continually (as a commercial product of the Cyberbotics Ltd.). Reviewing this software can describe well the state-of-the-art in mobile robot simulation in these days.

2.1 Cyberbotics Ltd.

The Cyberbotics Ltd. is one of the leading companies in the mobile robot prototyping and simulation software area. The company was founded in 1998 by Olivier Michel, as a spin off company from the MicroComputing and Interface Lab (LAMI) of the Swiss Federal Institute of Technology, Lausanne (EPFL). Cyberbotics develops custom simulation tools like the Sony Aibo robot simulator which was developed for Sony Corporation, but it also develops and markets Webots, the award winning fast prototyping and simulation software for mobile robotics. Developed since 1996, Webots became a reference software used by over 200 universities and research centers worldwide.

2.2 The Webots® Simulator – Overview

The Webots simulator package is now the main commercial product available from Cyberbotics Ltd (Cyberbotics). The provided robot libraries enable you to transfer your control programs to several commercially available real mobile robots. Webots lets you define and modify a complete mobile robotics setup, even several different robots sharing the same environment. For each object, you can define a number of properties: such as shape, color, texture, mass, friction, etc. Each robot can be equipped with a large number of available sensors and actuators. You can program these robots in many languages using your favorite development environment, simulate them and optionally transfer the resulting programs onto your real robots. Webots has been developed in collaboration with the Swiss Federal Institute of Technology in Lausanne, thoroughly tested, well documented and continuously maintained for over 7 years.

2.3 The Webots® Simulator – Features

Webots has a number of essential features intended to make this simulation tool both easy to use and powerful:

- Models and simulates many kinds of mobile robots, including wheeled, legged and flying ones.
- Includes a complete library of sensors and actuators.
- Lets you program the robots in C, C++ and Java, or from third party software through TCP/IP.
- Transfers controllers to real mobile robots (including Aibo®, Lego® Mindstorms®, Khepera®, Koala® and Hemisson®).
- Uses the ODE (Open Dynamics Engine) library for accurate physics simulation.
- Creates AVI or MPEG simulation movies for web and public presentations.
- Includes many examples with controller source code and models of commercially available robots.
- Lets you simulate multi-agent systems, with global and local communication facilities.

2.4 The Webots® Simulator – Robot and World Editor

A library of sensors is provided so that you can plug a sensor in your robot model and tune it individually (range, noise, response, field of view, etc.). This sensor library includes distance sensors (infra-red and ultrasonic), range finders, light sensors, touch sensors, global positioning sensor (GPS), inclinometers, compass, cameras (1D, 2D, color, black and white), receivers (radio and infra-red), position sensors for servos, incremental encoders for wheels. Similarly, an actuator library is provided. It includes differential wheel motor unit, independent wheel motors, servos (for legs, arms, etc.), LEDs, emitters (radio and infra-red) and grippers.

With Webots, you can create complex environments for your mobile robot simulations, using advanced hardware accelerated OpenGL technologies, including lighting, smooth shading, texture mapping, fog, etc. Moreover, Webots allows you to import 3D models in its scene tree from most 3D modeling software through the VRML97 standard. You can create worlds as large as you need and Webots will optimize them to enable fast simulations.

Complex robots can be built by assembling chains of servo nodes. This allows you to easily create legged robots with several joints per leg, robot arms, pan / tilt camera systems, and so on.

2.5 The Webots® Simulator – Realistic Simulation

The simulation system used in Webots uses virtual time, making it possible to run simulations much faster than it would take on a real robot. Depending on the complexity of the setup and the power of your computer, simulations can run up to 300 times faster than the real robot when using the fast simulation mode. The basic simulation time step can be adjusted to suit your needs (precision versus speed). A step-by-step mode is available to study in detail how your robots behave. Simulating complex robotic devices including articulated mechanical parts requires precise physics simulation. Webots relies on ODE (Open Dynamics Engine) to perform accurate physics simulation wherever it is necessary. For each component of a robot, you can specify a mass distribution matrix (or use primitives for simple geometries), static and kinematical friction coefficients, bounciness, etc. Moreover each component is associated with a bounding object used for collision detection.

Servo devices can be controlled by your program in torque, position or velocity. The control parameters for the servo can be individually adjusted from your controller program.

The graphical user interface of Webots allows you to easily interact with the simulation while it is running. By dragging the mouse, you can change the viewpoint position, orientation and zoom using the mouse wheel. Pressing the shift key while dragging the mouse allows you to move or rotate objects. This feature facilitates interactive testing.

2.6 The Webots® Simulator – Programming Interface

Programming your robot using C or Java language is quite simple. Any Webots controller can be connected to a third party software program, such as Matlab®, LabView®, Lisp®, etc. through a TCP/IP interface. Research experiments often need to interact automatically with the simulation. The supervisor capability allows you to write a program responsible for supervising the experiment. Such a program can dynamically move objects, send messages to robots, record robot trajectories, add new objects or robots, etc. The supervisor capability can be used in computationally expensive simulations where a large number of robot configurations and control parameters have to be evaluated, as in genetic evolution, neural networking, machine learning, etc.

2.7 The Webots® Simulator – Transfer to Real Robots

Once tested in simulation your robot controllers can be transferred to real robots:

- Khepera® and Koala®: cross-compilation of C Webots controllers and remote control with any programming language.
- Hemisson®: Finite state automaton graphical programming with remote control and autonomous execution modes.
- LEGO® Mindstorms®: cross-compilation for RCX of Java Webots controllers based on LeJOS.
- Aibo®: cross-compilation of C/C++ Webots controller programs based on the Open-R SDK.
- Your own robot: The Webots user guide explains how to build your own Webots cross-compilation system for your very own robot.

2.8 The Webots® Simulator – Summary

Webots has been used by more than one hundred universities and research centers worldwide since 1998 for both education and research purposes. Education applications allow the students to get started with robotics, 3D modeling, programming, artificial intelligence, computer vision, artificial life, etc. with an integrated tool. It is easy to implement a virtual robot contest, like a soccer contest or a humanoid locomotion contest, based on Webots, which is highly motivating for the students.

3. METHODS FOR 3D VISUALIZATION

The human sensing of the depth is a complex sensation. It has at least three different components, the so called extraretinal (not discussed here), the monocular and the binocular ones.

The monocular impression is the mixture of more, well-known phenomena as the perspective, shadowing, atmospheric distortion, a priori expected size of the objects, texture distortion, etc. Even in case of one-eye (monocular) sensing, the human brain interprets some kind of depth information. Although the capability of the monocular sense of depth is limited and it is sometimes inaccurate, it has essential role in the global sensing procedure.

Binocular sensation (frequently called as stereo vision) is the most trivial component of the depth sensing. Due to the 4-6 cm separation between the eyes, each eye has a slightly different viewpoint. The images from the two different viewpoints are sent to the brain and their difference – which is termed parallax – is interpreted as depth.

3.1 Advanced Visualization Techniques

As we mentioned before, the artificially generated visual impression is an inherent part of the man-machine interface. The use of visualization methods in this field, especially in case of remote controlling of mobile systems (teleoperation) needs the possible maximum level of reality. There are plenty of displaying methods under development and in use for this purpose. In the following we give a short overview on this topic.

Stereoscopic systems need two different images from the same scene. The human eyes must sense the two images totally separated. We can classify the stereo displays based on the method of the separation:

- passive separation,
- active separation.

Stereoscopic systems, which use *passive spectacles*, separate the two – in space overlapped – images by some kind of filters. Anaglyphs use color code separation and the used spectacles are having red and cyan optical filters in fact. Other solutions are using polarized lights, where the two images have perpendicular or circular polarization, which is aligned with the direction to the polarization filters of the glasses. A new method, called Infitec™ (acronym for <u>in</u>terfering <u>fi</u>lter <u>tec</u>hnique, that is a trademark from Daimler-Chrysler Research and Technology – Ulm, and is licensed to Barco) has superior stereo separation, which is perfectly suited for applications with high-contrast, non-saturated images. Two optical filters split the color spectrum in 2 parts: one for the left and one for the right eye information. It delivers superior stereo separation without ghosting, with full freedom of motion, independent of head tilt. Based on (Barco) – compared to polarization-based passive stereo – it achieves better separation, is independent of screen technology and therefore results in better uniformity.

Stereoscopic systems with *active glasses* usually use time-overlapped images. In fact the active spectacles are programmed shutters, which direct the images intermittently to the right and left eyes. Active stereo method is a flicker-sensitive solution – the smallest time-shift between the image display and the glass control “kills” the stereo feeling. There is no doubt about the need of glasses is the more difficult aspect of the implementation of stereoscopic systems. Other methods, which eliminate the need of glasses, have growing importance.

LCD displays with lenticular lens in front are projecting the adjacent columns of the screen in two slightly different directions – hopefully into the left and right eye of the viewer respectively. The screen displays two, vertically interlaced images in the same time, which construct a stereo pair. The solution is sensitive on the movement of the viewer. There are displays on the market, which are able to track the movement of the user’s eye, and modify the displayed images accordingly. This solution produces true depth sense, and allows relative free movement of the user – without the need for glasses. Auto-stereoscopic 3D imaging becomes popular for example on laptop computers.

We may not forget that monocular feeling plays an important role in our visual sensation. Our experiments proved that by using monocular sensing only – for example the transmitted images of an onboard camera – we can even drive a mobile platform remotely with high safety. How is this possible?

The stereo depth sense is produced through evaluation of the horizontal shift of corresponding points on two images. Even in case of random dot structure the human brain can find the corresponding pairs and represent them as in depth distributed information.

But there are other image features, which are heavily influenced by the depth, too. In our understanding, the key issue for the monocular depth sensing phenomenon is the depth sense from the motion.

The projected image of a true 3D scene, which contains objects in different distances, will change the arrangement of the objects on the image in case of any change in the viewer’s position. Objects on the image are covering each other, whereas this coverage varies according to the viewing angle. Continuous movement of the viewer causes continuous change in this coverage. By displaying the image of the moving camera on a monocular screen impressive depths feeling can be achieved. The solution is popular in some telerobotic application, where the movement is an inherent part of the task in any case.

On the software side there are many stereo solutions for CAD systems. Quad-buffered stereo is a routine within OpenGL that can be employed by 3D applications in order to show stereo images. For CAD applications such as SolidWorks and Solid Edge, there are plugins to generate stereo pairs. There are solutions which allow images and movies formatted in common stereo formats – such as above and below, side by side and interlaced – to be displayed in any available viewing format including anaglyph, page-flipping, sync-doubling stereo display modes, or auto-stereoscopic display. Supported file formats include .jpg, .jps, .bmp, .tga, .gif, .wma, .wmv, .asf, .avi and .mpg. In addition some software allows a stereo 3D effect to be applied to non-stereo movies and images allowing them to be viewed stereoscopically.

Unfortunately the usage of stereo visualization of robot simulators is not a widespread solution. We will introduce two different approaches on this field: the use of anaglyphs and the 3D visualization by motion.

4. THE ROBOTMAX MOBILE ROBOT SIMULATOR

Now there is an ongoing mobile robot simulator project (called: RobotMAX) at the Mobile- and Microrobotics Laboratory of the Department of Control Engineering and Information Technology in Budapest University of Technology and Economics.

The predecessors of this application are discussed in [2-4]. RobotMAX is written totally from scratch (with code translating / reusing from the previous versions) using the new C# language [5] that was introduced in the Microsoft .NET Framework. The framework is meant to be platform independent (Windows and Linux versions are available) and has a native support to advanced software technologies such as XML, SOAP or COM+.

4.1 Objectives in the Development of RobotMAX

In these days the key objectives for developing large software systems are modularity (concerning all devices and aspects) and interoperability (using the output from- or serve input to other related applications). We are trying to give general solutions for the upcoming robot simulation problems staying apart from model- or application-specific constraints.

One of our most important objectives is to develop a simulator that supports the testing of individual robotics components (vision system, navigation system, etc.) by means of hardware devices as well (so to support hardware-in-the-loop testing). For this purpose the main components of the simulated system are coupled through a standard interface and the simulation core and they are designed in a way that they could be replaced by an analogous physical hardware, transparently. Previously verified hardware devices can be useful if you want to focus on testing only one separate software component's behavior.

Another important objective is incorporating the X3D (Web3D standard) format to describe the virtual environment. This is a powerful and extensible open file format for 3D visual effects, behavioral modeling and interaction. It can be considered as a successor of the well-known VRML format. By providing an XML-encoded scene graph and a language-neutral Scene Authorizing Interface, it makes scene verification much easier and allows 3D content to be easily integrated into a broad range of applications. Its base XML language lets incorporating 3D into distributed environments, and facilitates moving 3D data between X3D-aware softwares (a.k.a. cross-platform or inter-application data transfer).

4.2 Visualization in RobotMAX

Our 3D visualization engine is based on the open-source Axiom 3D engine, which is used as our rendering "middleware". Axiom is a fully object oriented 3D graphics engine developed using C# and the Microsoft.NET v1.1 platform to create an easy to use, flexible, extendable, and powerful engine that allows for rapid development of games and other graphical applications (Axiom3D). By using the .NET framework [6] as the target platform, developers can focus more on core functionality and logic, rather than dealing with the complexities of languages like C++. The core of Axiom is a port of the very popular OGRE graphics engine, which was chosen based on its clean object-oriented design, powerful features, and flexibility. The main features of our extended 3D engine that is used in the robotics simulation are:

- Extensible Hierarchical Scene Graph with support for serializing standard X3D (successor of VRML standard) format scenes.
- Up to 4 reconfigurable rendering views (orthogonal or perspective cameras) with wireframe, solid or anaglyph rendering modes. We are using Managed DirectX 9 target render system (the OpenGL implementation using Tao. OpenGL wrapper is not working correctly, yet).
- Support for Ogre .material files, allowing the flexibility for controlling fixed function or programmed render state on object basis. For realistic dynamic behavior the objects' materials are containing their individual physical parameters such as density, friction, bounciness and so on. The dynamic physics simulation will be carried out via using a C# wrapper package for the also open source Open Dynamics Engine (ODE) using these parameters.
- Vertex and fragment programs can be the part of the material files. There is a full support for low level shaders written in assembler, as well as all current high level shader language implementations (Cg/DirectX HLSL/GLSL) for stunning visualization effects. Supported profiles are: arbvp1, vp_1_1…vp30 for vertex shaders; and arbfp1, ps_1_1…ps_2_0, fp20, fp30 for pixel shaders.
- Easy to use Render to Texture functionality (used in anaglyph render mode).
- Keyframe animation support. Currently allows animations to be assigned to nodes in the scene graph, allowing objects to move along predefined spline paths. Can be used to define moving obstacles in the virtual environment.

- Extendable controller support, allowing a wide variety of automated effects at runtime. Can be used to do time-specific motions and parameter changes during the simulation.
- Skeletal animations with an Ogre .skeleton file loader. Features include multiple bone assignments per vertex, smooth frame rate scaled blending, and multiple animations can be blended together to allow for seamless animation transitions. Skeletons are the best technique for describing the complex movement of the vertices of a walking humanoid platform.

4.3 User Interface in RobotMAX

Our mobile robot simulator solution uses a CAD interface that is similar to the well-known 3DStudio MAX® software (proprietary of Discreet™) to let you easily design the desired virtual environment and mobile platforms. Many primitive object types are available (boxes, planes, cylinders, cones and spheres) and general 3D mesh objects can be imported from standard X3D or Ogre .mesh files. All 3D objects (even cameras and lights) are nodes in a tree structure that are handled and stored hierarchically in the X3D scene description file. Thus one object can be a parent of another, where each object passes its transformation to its children. By construction all objects are the child of an unyielding root object (the one that has no parent at all), and they can be re-linked to their new parent (with the Link tool) as desired. The nodes that are representing the individual robot platforms themselves do not differ much from other general nodes. You can mark “anything” (3D objects, joints, cameras, etc.) as a part of an individual mobile platform during the assembly phase (in the Robot Construction Dialog). Generally there is a base chassis that holds (i.e. it is the parent of) the wheels, onboard cameras, sonars, etc. in each platform. These parts can be referenced and manipulated independently within the user-defined high-level simulation program.

After finishing up with the environment construction, you can switch to simulation mode where you can load, edit and execute your own high-level program (i.e. a precompiled .NET library) that will control the behavior of your robots. Of course the data of the onboard sensors (CCD cameras, sonars, tilt sensors, etc.) are at your disposal during the simulation.

4.4 The Architecture of the RobotMAX Simulator

We can say that mobile robot systems (both real and simulated ones) have three separate main components – sensing, processing and locomotion – that are communicating with each other. If we are in the software world, it is an understandable imagination that these components are treated as separate modules in the application. These modules are connected through a standard communication interface. The modular architecture of the simulator and this interface make possible the simulation of the behavior of a remotely controlled platform where the sensing and locomotion components are spatially separated from the intelligent control component (running on different computers that are connected via the internet or similar communication channels). The communication model of a remote controlled mobile robot is presented in Fig. 2.

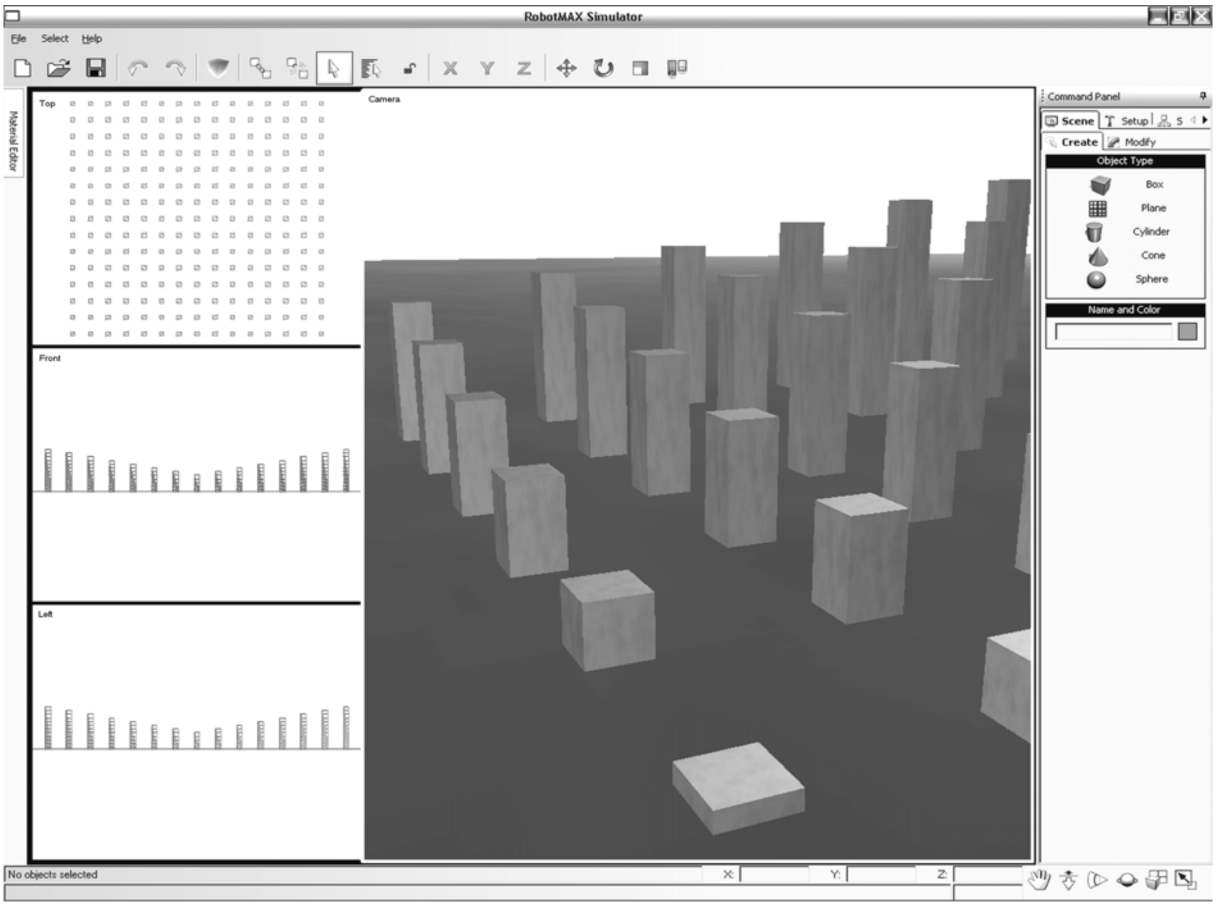

Figure 1: The user interface of the RobotMAX simulator.

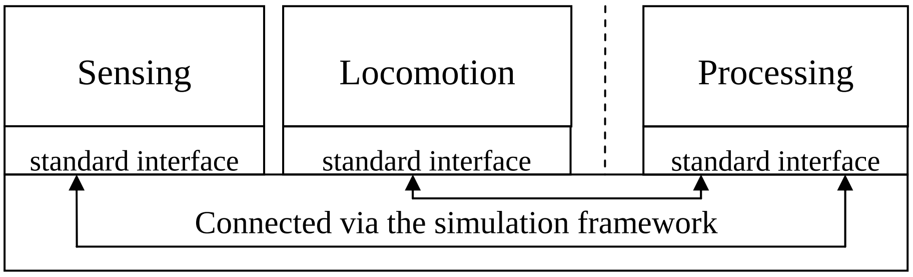

Figure 2: The modular architecture of a remote controlled robot (processing is on the local side).

The simulator will offer the following kinds of locomotion models:

- Differential drive with caster,
- Ackermann steering,
- Synchronous drive,
- Tricycle,
- Omni-directional (a.k.a. Swedish wheel).

The user can manually select and configure the desired driving subsystem in the previously mentioned Robot Construction Dialog. The simulator incorporates the dynamic model of these platforms and it uses the open source ODE (Open Dynamics Engine) physics engine to approximate the behavior of the physical platform that is simulated. If we use the previously discussed standard communication interfaces between the implemented modules then the core simulator framework can treat virtual and physical components the same way. By using hardware implementation in a given module, we can talk about hardware-in-the-loop testing. Using analogous hardware devices in a test loop lets you focus only on a specific part of the system that will be under investigation. This methodology plays an important role in virtual prototyping scenarios. Let's assume the following: we have a virtual representation of the real scene (with acceptable accuracy) where our real robot will be operating. We place this physical robot into an empty room with phantom obstacles (e.g.: sketch drawings on the floor: to avoid any unpredicted collisions) and control it according to the virtual environment's visual information. Thus we can verify our dynamic model being used whether the physical platform reacts the same way as the virtual one upon the same command sequence (for example a common navigation algorithm). If we got satisfactory results we can take our platform out of the room and put it to the real environment.

4.5 Advanced 3D Visualization in RobotMAX

The simulator supports advanced 3D visualization using the anaglyph technique. You can see good quality colored anaglyphs at (Anachrome 3D). In the followings we will discuss the used algorithm in our application, and compare it against the motion-based monocular 3D technique.

Generally when we are looking at an object in the distance we perceive distinct images with our left and right eyes. The closer the object is the bigger is the deviation (the so called parallax) between these images. Our brain "calculates" the depth from this difference: this is the basis of stereo imaging that can lead to true 3D perception.

We have implemented anaglyph rendering mode in the RobotMAX simulator. Anaglyph imaging produces a stereoscopic motion- or still picture in which the left component of a composite image is usually red in color. This is superposed on the right component in a contrasting color (usually cyan or blue) to produce a three-dimensional effect when viewed through correspondingly colored filters in the form of spectacles.

In our application this technique can be used to enhance 3D perception of the user in the virtual reality. Let's assume we have a monocular perspective camera (named "CameraM"): the parameters of which (position, orientation, field-of-view, etc.) are all known. We clone this camera twice: getting "CameraL" and "CameraR" as results that are matching their parent exactly. The previous one will present our left eye's view and the subsequent one is for the right eye's view.

According to the anatomy of an adult human being, the center-to-center distance between the eyes of the viewer is assumed to be 6 cm, thus we shift the left ("CameraL") and right ("CameraR") cameras by 3-3 cm along their local X axes (horizontal) in opposite ways.

We can "mark" the information of the given images using RGB filters: the left camera should have a red filter (RGB: [1,0,0]) whilst the right one should have a cyan filter (RGB: [0,1,1]). The anaglyph image contains the combined visual information that can be separated if we are using a red-cyan anaglyph spectacle. Fortunately red and cyan are complementary colors (their sum produces pure white or gray if they are mixed in the right proportions), so we don't lose brightness (luminance) information in anaglyph images. If we combine these images by simply adding them together, we get the desired anaglyph image.

All of these operations can be carried out in our rendering engine. The engine is using texture render targets for "CameraL" and "CameraR", and is multi-texturing the given images in each frame over the same invisible 2D plane using texture-blending operations, then this plane is made visible to the user. This can be done many times per seconds, so this rendering mode can be used in anaglyph motion pictures as well.

Although anaglyph images look black and white viewed through the spectacles, this feature isn't disadvantageous in case of RobotMAX, because we don't need color rendering in this mode of the simulator. Color information in fact is not really necessary for designing and testing. Nevertheless in case of telerobotic applications there is a limited need for color information as well (depth information is more important).

The previous technique is able to create and enhance the depth cue of stereoscopic vision at a price. It requires specialized viewing equipment, and therefore is not a realistic solution for three-dimensional viewing for the masses. It also is limited to a single viewing perspective. Therefore, it is unable to provide the important depth cue of motion parallax.

On the other hand the generation of anaglyph images is more effective than the calculation of continuous camera motion, which is necessary for both the motion-based monocular stereo effect and motion parallax. Even in case of well-configured motion stereo sequences without binocular sensation the resulting depth feeling is strongly limited. However we implemented motion stereo reproduction of still images in RobotMAX (Fig. 3 shows an example) as a compromise to eliminate the need for additional special equipment (glasses).

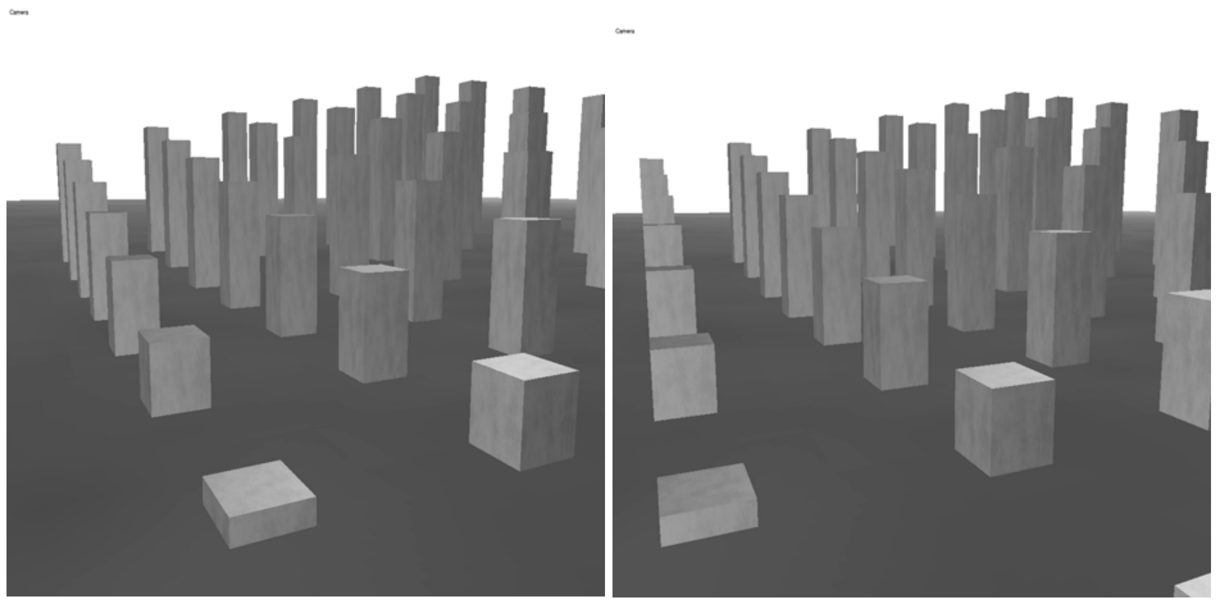

Figure 3: The first and last frames of a motion-stereo image sequence.

On Fig. 3 you can see two frames of a motion-stereo image sequence rendered using RobotMAX. The X coordinate (in world coordinate system) of the viewpoint of the monocular camera was slowly changing during the time. You can see that the corresponding perspective projected objects are overlapping differently on these frames. Those objects that are in the distance are almost static whilst the closer ones are seemingly moving a lot more. Our brain is using this changing occlusion information during interpreting the different 2D images from time-to-time, perceiving a depth sensation. This is an example how monocular, 2D perception can produce 3D information.

It is incontrovertible, that the use of a special auto-stereoscopic display is the most advanced solution (to set aside its price), because lenticular displays are able to solve both of the problems and provide three-dimensional stereo images over a range of viewing angles without the need for special viewing glasses (or a head mounted unit).

A lenticular display consists of a horizontal array of cylindrical lenses placed in front of interleaved pictures of a given object from different viewing angles. The device is designed so that at a given viewing angle, only one of the set of interleaved images can be seen. Therefore, as the viewer moves, the lenticuar sheet picks out a different image, and the percieved image rotates to provide the proper perspective for the viewer. This creates the experience of motion parallax. Fig. 4 illustrates the method with which a lenticular display creates this effect [7]. Since there is a difference between the eyes in both distance and viewing angle, each eye will see a different image, and stereo vision is achieved as well (see Fig. 5).

This technique is able to provide stereoscopic vision and motion parallax without the need for specialized viewing equipment. However, the achievable resolution of such a device is limited by large files sizes, and the available monitor resolution due to the need for many interleaved images. Another drawback is that despite its incorporation of many depth cues, it is still a two dimensional representation, and it is incapable of exploiting accomodation. Only truly spatial three-dimensional displays will be able to provide this.

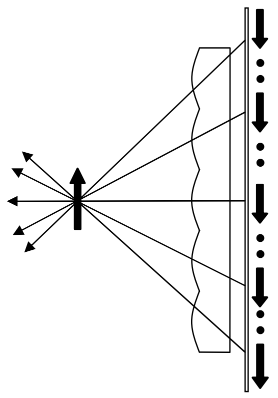
Figure 4: Lenticular display.

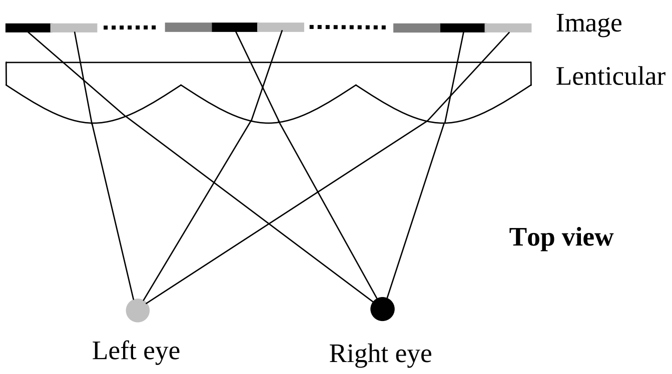

Figure 5: Stereo sense with motion parallax.

In fact all of the three-dimensional imaging techniques described up to this point has been two-dimensional representations with enhanced depth cues to create the illusion of a third spatial dimension. However, no matter how complex they are, they are still two-dimensional and ultimately flawed. This is because they lack the ability to appease the eyes need for ocular accommodation in experiencing a three-dimensional world. Spatial three-dimensional displays will allow the eye to focus at different depths throughout the display and truly experience all three dimensions. The volumetric display technique is still in its infancy, but it is worth further investigation because it is rich not only in fundamental science from different fields but also in skilful, detailed engineering. Already their application in the robotics lies far away. The auto-stereoscopic plug-in module for RobotMAX is currently under development.

5. CONCLUSIONS AND FUTURE PLANS

In this paper we presented an overview about modern visualization approaches in the field of telerobotics and interactive robotics simulation. We have made a comparison presenting the advantages and disadvantages between the (binocular) anaglyph and the (monocular) motion stereo methods in our robot simulator application.

At this time, our team in the Mobile- and Microrobotics Laboratory is in the very center of experimenting with some of the great amount of available devices supporting advanced monocular and binocular 3D techniques to find one that is the nearest to our requirements.

From this aspect we presented the RobotMAX simulator, which is currently under development at our laboratory: it will integrate robotics design (advanced visualization, driving mechanisms- and kinematical structure planning, sensor layout optimalization, etc.) and interactive control with the effective hardware-in-the-loop testing methodology into a modern simulation framework. Our mobile robot simulation project has another objective also: RobotMAX is aimed be a new research and education platform for the next generation of students in the field of mobile robotics and 3D imaging in our department at BUTE.

6. ACKNOWLEDGEMENTS

We would like to acknowledge the support of the Hungarian Scientific Research Fund (OTKA) grant No.: T-042634-OTKA, and the CLAWAR (Climbing And Walking Robots) Research Training Network – Mobile Robotic Demonstrators and Applications: G1RT-CT-2002-05080.

REFERENCES

[1] Vajta, L.; Juhasz, T. (2004). New challenges in mobile robot simulation systems, *Proceedings of 1st CLAWAR/EURON Workshop on Robots in Entertainment, Leisure and Hobby*, 2-4 December 2004, Vienna, Austria
[2] Juhasz, T. (2003a). OpenGL powered mobile robot simulator supporting research on landmark-based positioning, *Proceedings of MicroCAD'03 Conference*, 2003, Vol. N., 85-91
[3] Juhasz, T. (2003b). Graphics acceleration techniques for a mobile robot simulator, *Proceedings of CESCG'03: Central European Seminar on Computer Graphics*, Section 5: Computer Vision, 22-24th April 2003, Budmerice, Slovakia
[4] Juhasz, T. (2004). Surveying telerobotics and identification of dynamic model parameters (in Hungarian), *Proceedings of MicroCAD'04 Conference*, 2004, Vol. K., 211-216
[5] O'Reilly & Associates (2003). *Programming C#*, ISBN 0-596-00489-3
[6] Thai, T.; Lam, H. Q. (2001). *.NET framework essentials*, ISBN 0-596-00165-7
[7] Okoshi, T. (1976). *Three-Dimensional Imaging Techniques*, Academic Press, New York
[8] Anachrome 3D. http://www.anachrome.com/index.htm
[9] Axiom3D. http://www.axiom3d.org
[10] Barco. http://www.barco.com/VirtualReality/en/stereoscopic/infitec.asp
[11] Cyberbotics. http://www.cyberbotics.com
[12] Discreet-3D Studio MAX. http://www4.discreet.com/3dsmax/
[13] Web3D consortium. *X3D overview*, from http://www.web3d.org/x3d/overview.html