Wolfgang Strasser, Friedrich Wahl (editors):

Graphics & Robotics

Dagstuhl-Seminar-Report; 61
19.04.-22.04.93 (9316)

ISSN 0940-1121

Copyright © 1993 by IBFI GmbH, Schloß Dagstuhl, 66687 Wadern, Germany
Tel.: +49-6871 - 2458
Fax: +49-6871 - 5942

Das Internationale Begegnungs- und Forschungszentrum für Informatik (IBFI) ist eine gemeinnützige GmbH. Sie veranstaltet regelmäßig wissenschaftliche Seminare, welche nach Antrag der Tagungsleiter und Begutachtung durch das wissenschaftliche Direktorium mit persönlich eingeladenen Gästen durchgeführt werden.

Verantwortlich für das Programm ist das Wissenschaftliche Direktorium:
Prof. Dr. Thomas Beth.,
Prof. Dr.-Ing. José Encarnaçao,
Prof. Dr. Hans Hagen,
Dr. Michael Laska,
Prof. Dr. Thomas Lengauer,
Prof. Dr. Wolfgang Thomas,
Prof. Dr. Reinhard Wilhelm (wissenschaftlicher Direktor)

Gesellschafter: Universität des Saarlandes,
Universität Kaiserslautern,
Universität Karlsruhe,
Gesellschaft für Informatik e.V., Bonn

Träger: Die Bundesländer Saarland und Rheinland-Pfalz

Bezugsadresse: Geschäftsstelle Schloß Dagstuhl
Universität des Saarlandes
Postfach 1150
66041 Saarbrücken
Germany
Tel.: +49 -681 - 302 - 4396
Fax: +49 -681 - 302 - 4397
e-mail: office@dag.uni-sb.de

International workshop on

Graphics & Robotics

April, 19-22, 1993
Schloß Dagstuhl, FRG

Abstracts

Organizers: **Prof. W. Strasser, WSI/GRIS, University of Tübingen, FRG**
Prof. F. Wahl, Institute for Robotics and Computer Control,
Technical University of Braunschweig, FRG

Contents

Graphics Simulation for Robotics

Collision Detection/Trajectory Planning

Visualization & Virtual Reality

Geometry for Graphics & Robotics

Assembly Planning

Model-Based Robot Vision

Principles of Robot Simulation and their Application in a PC-based Educational Robot Simulation System

Dipl.-Inform. C. Laloni

Technical University of Braunschweig
Institute for Robotics and Computer Control
Hamburger Str. 267, W-3300 Braunschweig
Tel.: (49) 531 391-7460
Fax.: (49) 531 391-5696
E-Mail: cla@rob.cs.tu-bs.de

Abstract

Modeling and simulation of robots and robot work cells is one of the main aspects in the field of robotics research. The main task of simulation systems, which are used e.g. in CAD systems, is the visualization of a virtual environment (e.g. a planned building in an architectural application), with the aim of giving an illusion to the spectator, as close as possible to reality. These visual simulation is achieved by different techniques, which have been introduced by the computer graphics community, e.g. 3d views, multiple light sources, true color displays, ray tracing etc.

In contrast to such systems, a robot simulation system is used for off-line robot program development, and therefore has to satisfy additional constraints such as *kinematic* and *dynamic* models to achieve a useful simulation. The kinematic model of a robot arm describes the geometric structure of the robot arm links, including joint angles and joint velocities. For a description of the robot kinematics, coordinate systems are affixed to each link; their positions and orientations can be described by homogeneous transformation matrices. The *Denavit-Hartenberg* notation is used to specify the relationship between two adjacent links using a set of four parameters, including the free joint variable.

To achieve a 3d-graphical simulation of such a kinematically specified robot, previously modeled bodies can be attached to each link coordinate system. These relations are specified by *affixments*, which are also used to describe the connections of different objects. With these affixments object relations such as *'object A is mounted to object B'*, or *'object C is gripped by the robots hand'* can be modeled.

In addition to the robot arm kinematics it is very useful during off-line robot program development, to provide the relation between the cartesian coordinates within the robot work space, an the corresponding joint variables. This relation is known as the inverse robot arm kinematics and offers the possibility to specify robot movements relative to a user defined cartesian coordinate system.

With the additional use of a dynamic robot arm model, the robot simulation can be improved and even critical effects such as overshots can be simulated correctly.

Finally, the term *time* plays an important role in robot simulation, because the relation of the different objects within the simulated world can vary during the course of time. Especially, if the simulation computation can only be done off-line, and the *simulation time* is not identically with the *simulated time*, it is necessary to get information about the elapsed time to execute the simulated actions in the real world environment.

Advanced Techniques in Robot Simulation

Dipl.-Ing. Christoph Woenckhaus
Dipl.-Ing. Rainer Stetter

Institut for Tool Machines and Industrial Engineering
Prof. Dr. J. Milberg
Technical University of Munich

Introduction

The number of robot installations in industry has increased significantly over the past several years. To support planning of new complex robot applications, several offline programming and simulation systems have been developed.

The Institute for Tool Machines and Industrial Engineering (iwb) of the Technical University of Munich is developing its own offline programming and simulation system, called **USIS** (**U**niversal **Si**mulation **S**ystem) [1,2]. The goal of the development is to create a system which, one the one hand, contains a precise model description of the work cell to simulate robot behaviour accurately. On the other hand, new programming and planning techniques will be implemented to reduce programming efforts significantly. This paper presents some results from this research work [1].

USIS in General

A basic feature of the USIS is the design of a robot work cell. The user is able to select different cell components from several libraries, for example about 30 different robot types, containing up to six different control languages. The geometrical description of parts, which are not stored in this libraries can be passed over from CAD-systems using different interfaces, like VDAFS, IGES or SLA.The components have to be placed interactively after being choosen. This task is done by the use of several interactive functions to change the locations of components.

In the next step programs are generated offline in the robot specific control language. USIS offers different interactive functions for graphical robot programming.

Another feature is the execution of these programs in real time, which allows the user to simulate operations of different robots simultaneously. Different actions can be synchronized by input and output signals between different simulated robot control systems. While simulating offline programs it is possible to check for collisions in real time. The application of USIS is not limited to robot simulation, also the programming and simulation of tool machines [3] and the simulation of manual work systems [4] is feasible.

To meet the requirements for a quick validification of new assembly concepts and to increase planning efficiency, it is important to apply sophisticated computer graphics to the problem of graphic robot simulation. USIS uses the capabilities of powerful UNIX-workstations and the graphic standards PHIGS (+) and MOTIF.

Advanced Model Description

In order to get usable simulation results exact model descriptions are necessary. This does not only affect the geometrical model description but the real component behaviour. Some examples for this

---

[1] The work on which this report is based is sponsored by the Deutsche Forschungsgemeinschaft (DFG) within the framework of the special research projects SFB 331 and SFB 336

extended model description are physical effects, like gravity, flexibility or friction, which are implemented into the USIS [5]. E.g. under the effect of simulated gravity a part, which is released from a gripper, is forced to fall down. Coupled with fast algorithms for collision detection it is possible to calculate the final position of impact. With the development of faster and lighter manipulators robot dynamics becomes increasingly important. The determination and solution of the system of nonlinear differential equations is done by integrated dynamic simulation packages [6]. Another topic is the simulation of flexible material behaviour, for example a flexible pipe between the robot and the gripper. In addition to this different sensor systems, like a laser sensor [7], a laser scanner or a CCD-camera [2] are integrated into the USIS system, too.

Graphical Robot Programming

An advantage of offline programming is the possibility to develope robot programs without using the real robot. The most common method of graphic robot programming is the graphic teach-in. To generate a whole path the user has to identify certain robot positions. Teaching a position, a new command line is automatically inserted into the robot program. In order to do this the programming system automatically computes the new robot coordinates and warns the user if a desired position cannot be reached. This method is very time consuming and does not use all available information, which the simulation model contains.

A more advanced method of graphical robot programming is realized by using the so called frame mode. Positions are no longer stored in absolute coordinate values but with respect to a reference coordinate system linked to a component. If this component is placed to another position during the planning process, the program positions are adjusted automatically and the program will run, using the new location of the part.

The most comfortable method of offline robot programming is the automatical path planning. In USIS this is done by a self developed algorithm, which searches a collision free path between an user defined starting point and a destination point [8]. To find this robot track, it is necessary to have exact informations about the assembly cell environment. These informations are obtained from several sensor systems, which are also implemented in USIS. The result of path planning is a complete optimized robot program, which is written in the specific robot control language.

In addition to the path planning algorithm grasp operations are planned automatically by the system, too. The planning process is done in two steps. First an optimized arrangement between gripper and part is choosen with respect to geometric restrictions. In the second step the generation of the robot program for the complete grasp sequence is automatically done. Both, path planning and grasp planning, can be used in combination. So the planner only has to identify the part to grasp and the start and destination location. USIS then automatically generates the necessary program statements for a collision free grasp sequence, the transfering motion and the droping motion.

Numerical Optimization of a Layout

Planning robot work cells using three-dimensional simulation systems is an effectiv way to minimize planning efforts. While creating a layout of a work cell contradictory parameters must be considered. In most cases the context between these parameters is very complex and a manual change will not lead to the optimal solution. For example there is still a great potential to optimize the component arrangement in order to get the shortest possible cycle time.

The basic idea is, to link the three-dimensional simulation system to a numerical optimization package [9,10]. An optimization algorithm will suggest changing locations of different cell components. After this the robot programs will be automatically updated to the new locations using the so called frame mode. Subsequently this layout configuration is estimated by the execution of the modified robot programs. The result from a special assessment function is a so called quality value, which describes the quality of the current layout configuration. This value is given back to the optimization algorithm. Now this algorithm is able to suggest new component locations, and the next optimization loop starts.

With this system the planner is able to find optimized locations for different cell components, robots or sensor systems. Some possible optimization criterions are for example the cycle time, the sum of angle to cover and the dynamic load while executing a robot program.

Conclusion

Advanced three-dimensional planning and simulation systems should enable the simulation of complex work cells as realistic as possible. In addition to this different planning tools should support the planner with automatic functions, like path planning algorithms or numerical layout optimization packages. USIS embodies the discussed features of advanced graphic robot simulation systems. USIS has been applied to solve various problems and has proved its functionality in several industrial projects.

References:

[1] Wrba, P.: Simulation als Werkzeug in der Handhabungstechnik, Springer Verlag, Berlin 1990.

[2] Woenckhaus,C.; Stetter, R.; Rockland, M.: Optimierung flexibler Montagezellen durch die 3D-Simulation. Tagungsbericht: Simulation und Fabrikbetrieb, Arbeitskreis für Simulation in der Fertigungstechnik, Aachen (1993) 188 - 211.

[3] Schrüfer, N.: Erstellung eines 3D-Simulationssystems zur Reduzierung von Rüstzeiten bei der NC-Bearbeitung. Springer-Verlag, Berlin (1992).

[4] Kummetsteiner, G.: Planung manueller Arbeitssysteme mit der 3D-Simulation. Produktionsautomatisierung pa 2/92 (1992) 34-37.

[5] Stetter, R.: Einbindung physikalischer Effekte in die 3D-Simulation eines Handhabungsprozesses, Robotersysteme 8, 134 - 138 (1992).

[6] Woenckhaus,C.: ADAMS in robot simulation. 6th ADAMS NewsConference, Wiesbaden (1990), Proceedings.

[7] Milberg, J.; Schuster, G.: Modellierung von Sensoren für die Montagesimulation. ZwF 84 11, 650-654 (1989).

[8] Stetter, R.: Bahn nach Plan, Roboter, mi-Verlag (1990).

[9] Woenckhaus, C.: Konzeption eines Systems zur automatischen 3D-Layoutoptimierung. Robotersysteme 8, 239-244 (1992).

[10] Milberg, J.; Woenckhaus, C.: Automatic Layout Optimization Using 3D-Simulation. Proceedings of the European Simulation Symposium, Dresden, 290-294 (1992).

Graphical Robot Simulation within the Framework of an Intelligent Telesensorprogramming System

B. Brunner, G. Hirzinger

Institute for Robotics and System Dynamics

DLR Oberpfaffenhofen

Keywords: extended robot offline programming, 3D stereo graphics, virtual reality, shared autonomy, telerobotics, task oriented robot programming, multisensory robot, sensor simulation, local sensory feedback loops, sensor fusion, man-machine-interface, uncertain environment.

Today the most significant problems in robot programming arise in the *uncertainties* given by robot and environment. Mechanical inaccuracies on the manipulator level and uncertainties in the environment cannot be compensated by offline programming systems. Local adaptations on path generation level have to be done via teach-in commands by human operator. Even small deviations in position, orientation and form of all the objects to be handled are not allowed because the task execution will fail in an uncertain environment. Graphical simulation of robot tasks and downloading the generated commands to the real robot is limited to the joint or cartesian motion level. This approach is only useful if geometrical consistency of real environment and the simulated one can be guaranteed. This is a demand that cannot be met with available programming systems.

Similar problems as in robot offline programming arise in the field of *telerobotics* with time delay especially in space and subsea applications. Direct visual feedback in a few seconds time delayed control loop is not feasible for the human operator to handle the robot movements in a suitable way.

The uncertainty problem in both application areas can be solved by increasing *autonomy* on the execution level. To achieve this goal there are a lot of things to provide. The most important requirement is the ability of *sensory perception* and sensor data processing. Without accurate information about the actual environment successful task execution can be considered as impossible in an uncertain environment. High level planning facilities for task scheduling or intelligent error handling mechanisms are required for full autonomy but state-of-the-art techniques are insufficient to provide adequate tools.

Therefore we favour a *shared autonomy* concept that distributes intelligence to man and machine. Local sensory feedback loops are executed by the robot system, global task level jobs have to be specified interactively from a human operator. Coarse planning activities have to be done on a task-oriented level by human intelligence, fine path planning on manipulator level takes place on a sensor based control level with predefined artificial intelligence.

For this shared control approach we have coined the term *telesensorprogramming*. This means teaching by showing with the aid of sensory refinement in a completely simulated world on a task-oriented level. The graphical offline programming concept is extended by the processing of simulated sensory data. Not only joint and cartesian information is gathered by graphically guiding the robot through the task, but also simulated sensory information to store them offline as nominal patterns for subtask execution on a local feedback loop level. Besides this sensory perception simulation the shared autonomy concept has to provide interactive tools to describe various tasks by different parameters. The operator has only to decide what kind of sensors and control algorithms should be used at each local sensor controlled feedback loop. The fine motion control to handle uncertainties occur independent of any human influence both at the simulation side and the real one.

In the field of telerobotics the time delay problem can be solved with the same approach of telesensorprogramming. *Predictive simulation* -- graphical and functional --- is the medium for the operator to telemanipulate the remote system online. He only has to guide the robot in a rough manner through the task space and to activate specific sensor control phases. After sending these gross commands to the remote system the real robot will be able to execute the desired task with his own local autonomy after the delay time has elapsed. On the basis of the online requirement realtime power is an important aspect in spite of simulation facilities. The main feature of our telerobotic concept is to replace the time-delayed visual feedback by predictive stereo graphics with sensor simulation providing a supervisory control technique that will allow to shift more and more autonomy and intelligence to the robot system.

The *main focus* of this paper lies on the model based simulation of robot, workcell and a set of typical sensors. We are able to emulate the behaviour of laser distance and force-torque sensors under realtime considerations. Stereo vision is simulated by the tools of graphical animation as the framework for a model based vision approach. These different kinds of sensors can be used to develop and test new types of sensor fusion algorithms to provide efficient control schemes and to verify the proposed telesensorprogramming approach.

Graphics Simulation of Dynamic Vision Based Robotic Workcell

- Extended Abstract *-

Dr.-Ing. Fan Dai
Fraunhofer-Institute for Computer Graphics (FhG-IGD)
Wilhelminenstrasse 7, 6100 Darmstadt, Germany
Tel.: +49-6151-155-207
Telefax: +49-6151-155-199
e-mail: dai@igd.fhg.de

Prof. Dr. Ping Dai
Department of Automatic Control Engineering
Central South University of Technology
Changsha, Hunan
410083, P.R. China
Tel: +86-731-83921-628
Fax: +86-731-82817

February 1, 1993

1 Introduction

Interactive computer graphics allows now to view and to manipulate different kinds of information interactively so one can simulate the behavior of complex systems in real-time to analyse and optimize the systems. There are several commercial and research systems for robot simulation available [4]. However, in existing robot simulation systems, vision systems especially dynamic vision systems are scarcely taken into account. In fact, graphic simulation of vision based robotic workcells is not only needed for the off-line planning and programming of an advanced robot, but also useful for the investigation of its real-time control strategies and algorithms.

As well known, artificial vision is of primary importance for an intelligent robot. It extends the ability of robots to cope with uncertainties and to work adaptively in changing environments.

At present, the vision systems in industrial use remain in general monochrome, two-dimensional (2D), static and most often two-valued. Being at such rudimentary level, they are limited to some simple robotic tasks, especially pick-and -place type manipulations. Many potential applications call for more advanced robot vision, prossessing characteristics such as multi-gray-levels, 3D, dynamics and colour. The so-called *dynamic visual control* of industrial robots implies that the robot control is based on computer vision which should offer the changing world information all along the robot manipulative process. This involves on one hand a continuous processing of time varying imagery, on the other hand the integration of vision into the different control loops in a robot system.

This paper presents our preliminary work for establishing a graphic simulation support for dynamic vision based robots, by extending a graphic interactive robot simulation system, the Robotics Kernel System (RKS) [3, 4].

2 Dynamic Vision based Robotic System

Dynamic vision is capable of sensing in process the environmental situation and the manipulator position, thus permitting visual servo control and on-line decision making of a robot.

In [2], a theoretical analysis on dynamic vision based robotic systems and an experimental realization were presented. We use now the model described there (figure 1).

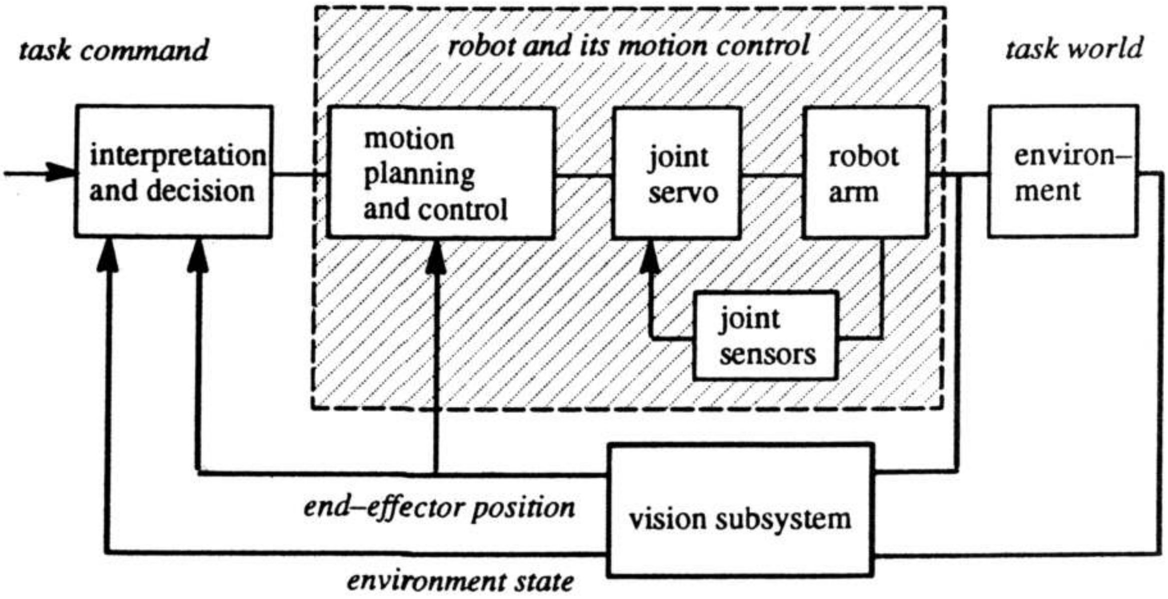


Figure 1: Dynamic Vision Based Robot Systems

Additional to a conventional robotic system consisting joints and joint servo control, the vision subsystem allows to control the robot arm according to the actual external situations. This vision subsystem in its turn has two basic kinds of roles: proprioceptive and exteroceptive ones, corresponding often to "end-effector position measurement" and "environment sensing". Robot vision in either static or dynamic mode serves primarily as exteroceptive sensor. However, in static mode, the vision subsystem recognizes the environment which supposed to be stationary while the robot system works. It is only in dynamic mode, that vision forms a feedback in a closed control loop.

3 Graphics Simulation

Concerning the robot and the environments, the geometric or kinematic model is usually adopted in existing robotic simulation systems. For vision based systems, an additional model is needed which represents the transformation from the observed scene to the digital images in the frame buffer and further to the information needed by the decision and control component of the robotic system. This is done by a viewing transformation to the vision systems coordinate system [1] and by a filtering process which emulates the image preprocessing (e.g. [5]). The viewing transformation is based on the object hierachy (e.g Figure 2).

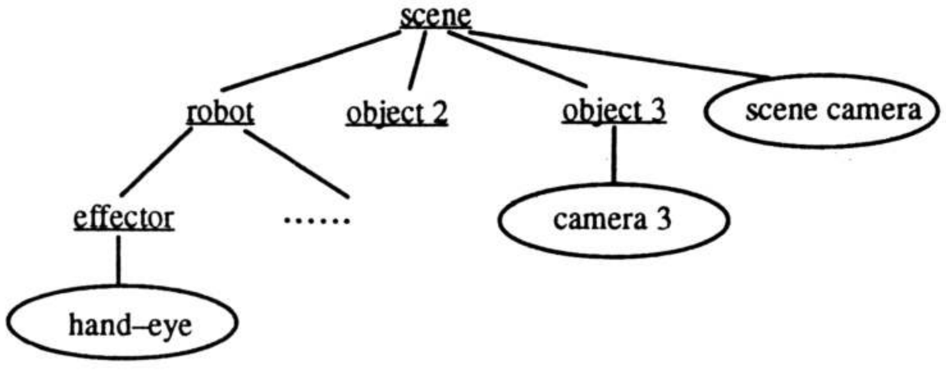


Figure 2: Example of the Object Hierachy

Additionally for dynamic vision, dynamics in the vision process have to be considered, thus, the motion related problems, such as the bluzzing caused by the relative movements between the camera and the observed objects, and also the dynamic behavior of the robot. For the first step, the time delays due to image processing, control calculations and robot inertia are essential, as they will influence greatly the system control performance. The overal simulation model is as depicted in Figure 3.

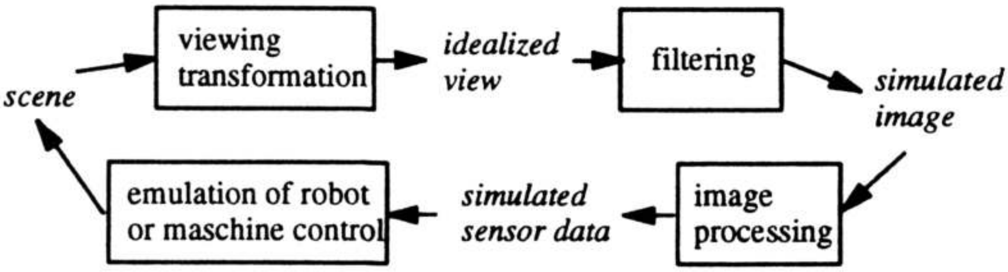


Figure 3: Simulation Model of Dynamic Vision Based Robot Systems

In the graphics simulation, it is important to present the modeled information in a suitable manner and to allow easy manipulation of them. At first, multiple user-controlled windows allow to present all information individually. These are for example: *users view of the environment, vision systems views, images after preprocessing* and *final vision information.*

Second, vision system parameters, for instance the image resolutions of the cameras and the frame buffer, the placement of the lights and the cameras, the acquisition frequence, the target tracking windows, and the position of the cameras can be changed interactively using powerful graphic interaction features. Of course, the whole work environment of the robots can be conveniently defined too.

Other than the uses mentioned above, this work is also helpful for vision system calibration and evaluation, model based vision and model based control of the vision based robots.

4 A Simulation Example

Our preliminary realization is based on RKS, which offers powerful interactive modeling and simulation tools. Using the existing features of RKS, we defined an additional object called "eye", that can be put onto individual objects like robot hand, car or room parts. To simulate the vision system, a procedure for image processing is extended to RKS. A module emulating the robots dynamics is implemented preliminary considering only the time delays. This module is to be extended later.

In the simulation example, the robotic system under consideration consists mainly of the following three subsystems: a 6 DOF manipulator and its controller, a dynamic vision system, and the working environment including the workpieces to be manipulated. The dynamic vision system considered here is composed of a hand-held CCD camera, a fixed camera, an image preprocessing board and a micro computer.

5 Conclusions

Advanced robot systems will be dynamic vision controlled. Interactive graphics simulation helps to analyse the effectiveness of sensor placement, characteristics and supporting environment for the robot. Considering the dynamic behaivior is important for vision systems calibration and evaluation. If the control strategies and algorithms are considered in the model, simulation will also help to investigate and to develop, optimize the whole robot system.

Using powerful graphics interaction techniques and multiple windows the user can view all information and define the parameters conveniently. Our work discussed the model for the graphics simulation of dynamic vision-based robot systems and has shown an example realized using RKS. It shows also, that RKS can be easily extended to simulate new applications. The next step will be to integrate multi-media functions into RKS. This will allow to display real and simulated vision data simultanously.

Further works will be also, the improvements of the system modeling and simulation, and on the other hand, study of novel methods for the planning and control of the dynamic vision based robot, utilizing the simulation support thus developped.

References

[1] Sakane, S., Sato, T. and Kakikura, M.: Model-based planning of visual sensors using a hand-eye action simulator system: HEAVEN. ICAR'87.

[2] Dai, P.: *On Dynamic Visual Control of Industrial Robots.* IEEE International Symposium on Industrial Electronics, May 1992, Xian.

[3] Dai, F.: *Integrated Planning of Robotic Applications in a Graphic-Interactive Environment.* Robotics and Autonomous Systems 8 (1991) 311-322. North-Holland, 1991.

[4] Wloka, D.W. (ed.): The International Handbook of Robot Simulation Systems. (To be published by John Wiley & Sons, Ltd.)

[5] Raczkowsky, J., Mittenbuehler, K.H.: Simulation of Cameras in Robot Applications. IEEE Computer Graphics and Applications, January 1989. pp. 16-25.

Using Robot Simulation Systems in Research and Education

Peter Stöhr
Institut für Informatik
Lehrstuhl für Echtzeitsysteme und Robotik
Technische Universität München
email: stoehr@informatik.tu-muenchen.de

During their studies, students of the Technische Universität München can attend a tutorial on robot programming. One aim of this tutorial is to present to the students the state-of the-art technology of robot programming, and therefore some work has been done to integrate exercises which use a robot simulation system.

The purpose of these exercises is to save time in learning to work with a robot simulation system. This is why the system must have an easy-to-learn user interface and its program operated intuitively.

Some characteristics of robots are very difficult to recognize in reality. For example, the difference in robot motion between straight movements and those with joint interpolation. Using a real robot one can see these motions only sequentially, whereas a simulation system can show them at the same time. Another of these characteristics is the so called continuous path, which is only visible in high speed motions of the robot. Unfortunately, this type of motion is an exertion for the robot and should therefore be avoided. All these effects can easily be shown with the help of a simulation system.

In a further step, the simulation system will be used to verify the correctness and to test the reliability of programs that run in the real robots. The main purpose of this step is to increase the safety of our robots.

The increased safety is one of the main reasons for the industry to work with robot simulation systems. This, together with the fact that a real robot is much more expensive than a simulation system, led our group to work predominantly with simulation systems.

One aim of our research activities is closing the gap between a real environment with real robots and control units, and simulated reality. The first step is to design a common interface for the simulation system and the real world, which at the moment is our main work. But this can only be the first step in making an interchangeable context for both, the real world and the simulated one.

Our next efforts will deal with the simulation of optical sensors such as CCD cameras, one- and two-dimensional laser scanners and a three-dimensional laser scanner integrated in a robot gripper. First, we want to simulate mathical exact sensor — which means that these simulated sensors will give undistorted results — and then extend these concepts to design sensors producing data which is almost the same as that obtained using real sensor systems.

In a last step we want to change the simulation of the robots and the control units. As with optical sensors, the simulation of a perfect robot can today be considered resolved and the implementation of such a complex system is only a matter of time and money. But even two real robots of the same type and from the manufacturer act differently with a given program. Therefore, our goal is to reproduce this uncertainty in the simulated robot. But the modelling of such robots is one of the unsolved problems of robot simulation systems.

TOROS: Graphical Toolbox for Robot Simulation

C.Balaguer, F.J.Rodríguez, C.F.Rodríguez, A.García, A.Barrientos, R. Aracil, E.A.Puente

Div. de Ingeniería de Sistemas y Automática (DISAM)
Polytechnical University of Madrid, SPAIN

EXTENDED ABSTRACT

Graphical environments for task and path planning, simulation and off-line programming are basic tools for design and analysis of robotized systems. Integrating graphics and interactive features is specially suitable for dealing with problems like generation and validation of tasks sequences, planning grasping conditions and collision free path generation in autonomous or assisted way. Simulation tools provide a safe way to foresee robot's capabilities, drawbacks and proficiency, during design or application stages. A great advantage is acquired, if experimenting on the real system is impossible because it is not constructed yet, or if system's environment conditions are hard to reproduce.

In this paper we introduce TOROS (TOol-box for RObot Simulation), a software tool developed by DISAM during the last three years. TOROS is an integrated environment for design, path planning and simulation of complex robot systems, such as redundant and multirobot systems. The system is graphically displayed by the integrated graphical simulator. There are other modules for robot parameter adjusting, automatic path planning and collision-free verification.

The graphical simulator is the central module of the tool. It provides an easy way to represent geometric models of the robot(s) and its environment. The representation is fully three dimensional, including wire and solid modelling and is displayed using one or three mobile or static views. It is also possible to get specific sections of the 3D model, for example, a plane section by the work plane of a robot which may be very useful in collision-free verification and robots kinematic design.

The simulator is commanded by an internal clock. Animation speed can be increased or decreased adjusting the clock period. The clock minimum period is limited by the calculation time. Anyway applications for ASEA IRB-2000, PUMA 560, and AMR2 robots, confirm that simulation is always fast enough for the real system. During each clock iteration the following sequence must be completed: a) transformation of mobile objects, b) verification (if selected) of collisions between robots themselves, robots-environment, robot-workpieces and workpieces-environment, c) control of input devices, and d) composition and redraw of scenes.

Robot and environment modelling

Geometrical models of robot(s) and its environment are constructed using rectangular polyhedra, generated by polygon linear swept. Although, boundary representation is used internally. There are three types of objects: active objects (robots), passive objects (workpieces) and fixed objects (obstacles). Objects modification is performed easely editing specific files.

An active object is an object able to produce movement by itself. There are two types of active objects: robots and vehicles. Robots are represented by solid links, and rotation or translation joints. The relationship between links and joints is based in homogeneous transformation according to Denavit-Hartenberg parameters. Any type of gripper or tool can be added at the tip of the robot's last link, defining its solid model and its fingers' movement type (discrete or continuous). The robot's TCP is defined for each gripper or tool.

Vehicles are represented by a solid's set without articulations. Their movements are specified by their position and orientation, constrained to stay on the defined floor. Slight modifications have to be made to describe the behaviour of other possible active objects, like belts, transportation devices, etc. in order to include their models into the environment.

Passive objects cannot move by themselves, but they can be held and moved by an active object. The passive object's position and orientation may be changed only by the action of an active object. In this sense they can be seen like workpieces. Finally, some passive objects in the environment can be considered fixed obstacles like floor, walls, etc.

Two objects may be related by a grasping or holding relationship. In such case if one is moved, also does the other one, so they are said to be "linked". Although only active objects can "generate" movement by themselves, for some tasks it is easier to move a passive object following a desired path and automatically produce the appropriated movement for any object linked to it (either active or passive).

Collision-free verification

Automatic collision verification between objects can be activated if necessary. Every solid is verified to not intersect any other which is not related to. For example, if a robot is grasping a bar, the bar is verified with every robot link, except the grasping gripper, which is obviously collided with the bar.

Due to the fact that exhaustive intersection verification between every side and face of both solids is time consuming, a set of very fast checks are performed before doing that. This set of sufficient or necessary conditions for intersection or not-intersection are applied in increasing complexity order.

Another important feature of TOROS system is the collision-free verification in manual mode. If an object (robot, workpiece) is moved and collision-free verification is activated during the movement, it is not possible to collide the object, i.e. only guarded movements are produced.

Path planning module

The path planning module permits the manual or automatic generation of robot trajectories. For manual operation a simple line editor and an instruction interpreter have been implemented.

Automatic collision-free trajectory generation is based in path searching in C-spaces. Two methods have been implemented: a global method and a local one. In the global approach, several complete C-spaces are computed and combined. The computation of C-spaces is not quite fast, but it is possible to store them for search many other paths. Trajectories are computed very fast and they have good quality.

The local method builds the C-space and searches the trajectory at the same time. In both methods (global and local) path searching is performed using the heuristic searching algorithm A^*. The local method takes advantage of the local information for auto-adjusting and adapting the algorithm parameters on-line. Normally, only a sub-region of the free space is handled. Local C-space generation is fast but path planning is slower than in the global method.

Robot parameter adjusting module

As mentioned before, the environment can be used for design robot's kinematic structure, especially when reachability, configuration or accessibility are important features. This module was developed to show the effects of kinematic parameter changes in robot structure. Joint limits or Denavit-Hartenberg parameters can be interactively modified and consequent effects are displayed immediately in the simulator. Links shape and size can be easily modified as a new input of the simulation environment.

Conclusions

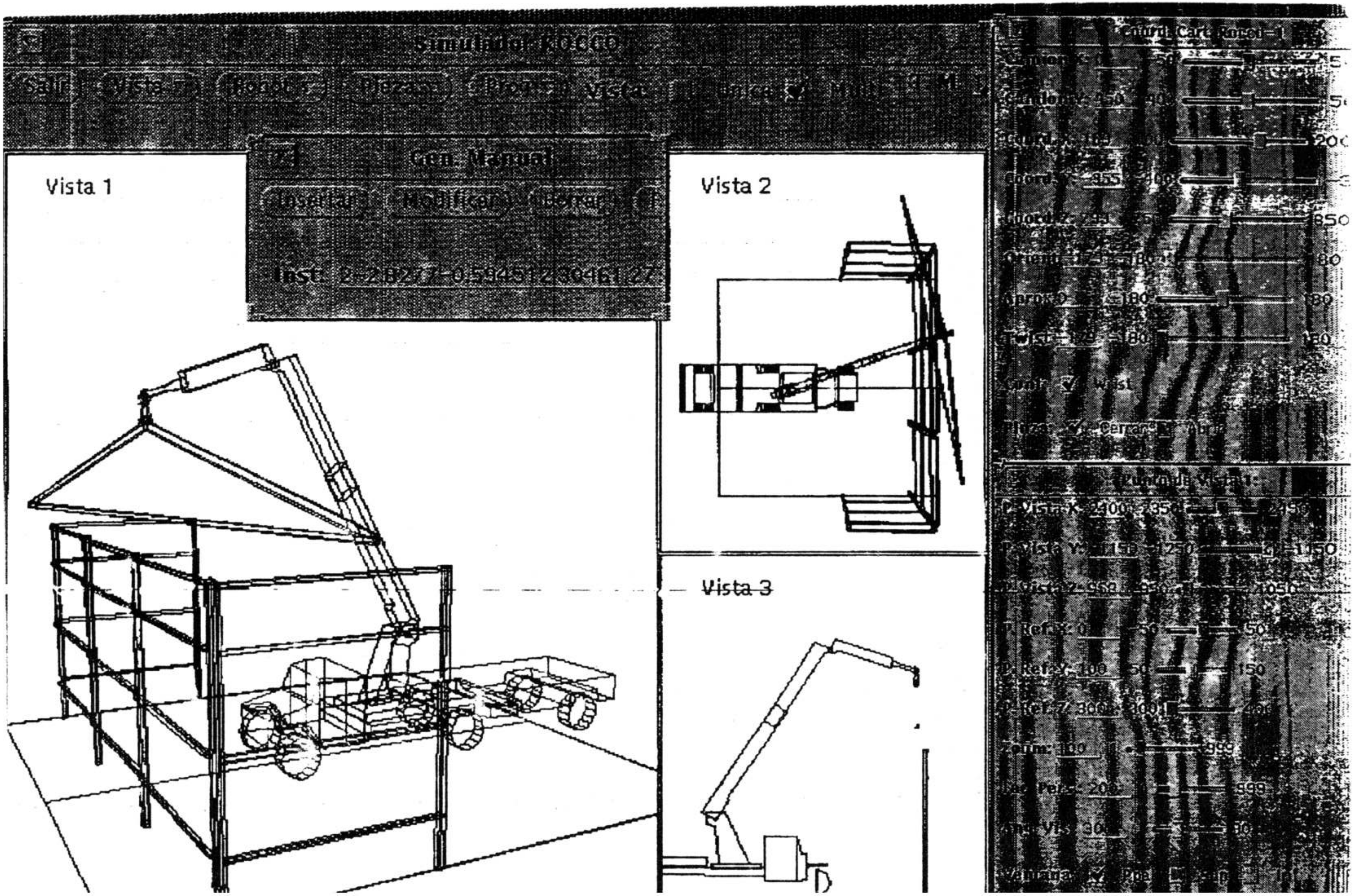
The "TOROS" environment has been successfully applied and checked in several R&D projects, in which DISAM has been involved:

- For collision-free path planning in complex environments in the EUREKA-EU-18 project "Advanced Mobile Robots for Public Safety" (AMR).
- For path planning and simulation of a multirobot system in CICYT ROB89-0174 project "Coordinated Control of Multiarm Systems" (COCO).

- For design and path planning of a robotized system for electrical hot lines maintenance in the IBERDROLA S.A. project "Robot for hot line maintenance" (ROBTET).

- For design and path planning of a construction robot in the ESPRIT 6450 project "Assembly Robot System for Computer Integrated Construction" (ROCCO).

An example of path planning application in bay construction of industrial buildings using a 12 meters long 7DOF robot, is presented in Fig. 1.



Inst: 8-2.8277-0.594512.30461.27

Camion X: [illegible] -50 5
Camion Y: 450 -400 5
Coord. X: 169 -100 200
Coord. Y: -355 -400
Coord. Z: 791 850
Orient: 173 -180 180
Aprox D: [illegible] -180 180
Twist: 179 -180 180
Cont: ☑ Wrist
Pinza: ☑ Cerrar ☐ Abrir

P. Vista X: 2400 2350 2450
P. Vista Y: -1150 -1250 -1150
P. Vista Z: 952 950 1050
P. Ref. X: 0 -50 50
P. Ref. Y: 100 50 150
P. Ref. Z: 300 300 400
Zoom: 100 1 999
Dist. Pers.: 200 999
Ang. Vis: 30 80
Ventana: ☑ Ppal ☐ Sup

Priv. Doz. Dr.-Ing. Dieter Wloka, M.Eng.
Forschungsgruppe für Robotertechnik
Lehrstuhl für Systemtheorie
Geb.13, Im Stadtwald
Universität des Saarlandes
6600 Saarbrücken
Deutschland - Germany

Tel. (49) 681-302-3993
Fax. (49) 681-302-2678
email wloka&uni-sb.ee.de

Robot simulation - an overview

Robot simulation techniques have found their way into commercial products. This contribution gives an overview about commercial and academic systems. It tries further to show areas of applications, problems of use and further developments.

A.o. Univ.-Prof. Dr. W. Purgathofer
Institut für Computergraphik
Technische Universität Wien

CSG based Collision Detection

A method for detecting collisions among complex objects modelled with constructive solid geometry will be presented. The algorithm is performed in three steps: bounding volumes reduce the possible colliding primitives to a small number, spatial subdivision detects pairs of primitives to be tested, and the tests are performed on an analytical basis. The goal of the method is to provide a powerful tool for simulating realistic motion in computer animation.

Using Graphics Algorithms as Subroutines in Collision Detection Simulation

Heinrich Müller, Fachbereich Informatik, Lehrstuhl VII (Graphische Systeme),
Universität Dortmund, Postfach 500500, 4600 Dortmund 50, Germany

Collision detection is the central task of motion planning, in on-line planning simulations as well as in precalculating the configuration space for searching optimal paths. Collision detection can be formulated as follows:

Input: A geometric environment, a geometric object.

Output: For an arbitrary given location of the geometric object in the environment, the information whether the object intersects the geometric environment.

A closer look at this problem shows that it is related to the visibility problem of computer graphics. This observation has led e.g. to the use of raytracing for collision detection [Pellegrini, 1992]. Raytracing is a well know technique of photorealistic rendering in computer graphics.

One approach for analyzing configuration spaces is to carry out a regular cell decomposition. This is in fact a task of rasterization or scan conversion which is central in raster graphics. This observation was used by [Lengyel et al., 1990].

In our contribution we focus on the possibilities of using further graphics algorithms for collision detection purposes, in particular the z-buffer-algorithm.

References

M. Pellegrini, Incidence and nearest-neighbor problems for lines in 3-space, 8th ACM Symposium on Computational Geometry, 1992, 130–137

J. Lengyel, M. Reichert, B.R. Donald, D.P. Greenberg, Real-time robot motion planning using rasterized computer graphics hardware, Computer Graphics 24(4) 1990, 327–336

Efficient Path Planning Strategies for Cooperating Robots in Environments With Obstacles

Max Fischer
Institut für Informatik
Lehrstuhl für Echtzeitsysteme und Robotik
Technische Universität München
email: fischema@informatik.tu-muenchen.de

31.Januar 1993

In recent times the field of cooperating robots has been gaining more and more interest. However, the task of programming cooperating robots is very complex, and there is still a lot of research to be done. One subproblem to be treated in this paper is planning collision-free paths for cooperating robots with dependent tool center points (TCPs) in environments with obstacles. Collision means robots colliding with each other or with their environment, as well as reaching their joint limits.

A planner for these tasks has to plan a pair of time-synchronous paths with one TCP moving along a given path relative to the other robots TCP frame. These planning tasks are subdivided into two groups:

- transport tasks, in which the points where the synchronous paths begin and end are given, and the relative path between the TCPs is reduced to a constant transformation, and
- manufacturing and assembly tasks, with one robot holding a workpiece in a continuously changing position so that the other robot can perform its task on it. Here, the beginning and end point of the synchronous paths and the chosen synchronous paths themselves are more or less arbitrary, while the relative path between the TCPs has to be executed without collision or reaching the robot's joint limits.

This paper describes three planners. The first solves the problem of finding a pair of synchronous paths for transporting objects in an environment with obstacles. The second and third are designed to find a pair of synchronous paths that realize a given relative path between two TCPs.

The planners avoid the immense complexity of a complete search and instead use heuristics as goal-directed search and sliding along obstacles. Escaping from a local minimum is done by stochastic strategies. Although these search strategies are incomplete, a planner using them for a single robot has been shown to be extremely efficient and has proved its ability to find a solution in many difficult realistic environments.

1. The first planner designed for transport tasks tries to find a path by following the straight six-dimensional path from the starting point to the goal. Whenever an obstacle is hit, the planner tries to find a path by sliding along the obstacle as long as the distance to the goal decreases. As soon as there is no possible collision-free step leading closer to the goal, this position is stored as a local minimum. Then an arbitrary position is defined as a subgoal and paths are searched from it to the starting and end positions and all the stored local minima. Thus a net of possible connections is constructed. As soon as there exists a path through this net from the start to the goal position this path is returned as a solution of the planning task.

   As mentioned above for manufacturing tasks (like welding a large workpiece), there is no need for the workpieces to follow a certain path. Only the given relative TCP path has to be followed. Thus, to obtain the next incremental step different optimization criteria can be integrated.

   Examples for these are:

   - minimizing the sum of all joint movements,
   - minimizing the movement of the workpieces or
   - keeping the joints around the middle of their movement range in order to prevent them from reaching their limits.

2. The second planner is designed for manufacturing and assembly tasks. It tries to find a pair of paths that fulfil the given relative path between the robot's TCPs. A time is defined for the relative path. Time **0** means the starting point, time **T** means the end point. If a local minimum is encountered, the planner stores time **t** of this local minimum. Then it tries to find several pairs of configurations that fulfill the relative TCP path at time t and goes on planning a pair of paths for each until time T is reached or the next local minimum is encountered. The task of connecting two configurations having the same time along the relative TCP path is a transporting task and is solved by the first planner.

   The second planner constructs a pair of paths by generating parts of paths and then connecting their starting and end points. It thus realizes more complicated manufacturing tasks by stopping execution, changing to another configuration and continuing there.

3. Planner 3 is designed for cases demanding a continuous execution of the manipulation path.

   It searches a path by 'growing branches from the already found tree of paths'. Like the second planner, it moves through free space and slides along obstacles while proceeding on the relative TCP path. When a local minimum is encountered, an arbitrary point on the already existing path (or tree of paths) and an arbitrary six-dimensional direction for the robots to move in are chosen. From there the search goes on until the end of the relative TCP's path is reached or another local minimum is encountered. By this strategy a large area of the configuration space is sequentially searched.

Dr. Hans-Heino Ehricke
Wilhelm-Schickard-Institut
für Informatik
Universität Tübingen

Visual Navigation within Volume Data

Recently, the availability of new interaction devices from the Virtual Reality area as well as fast graphics hardware have triggered the need for techniques allowing navigation within three-dimensional data spaces. In this context, one of the main problems consists of the identification and visualization of relevant objects within the volume. Volume visualization techniques which have been proposed in the literature rely on a pre-segmentation or pre-classification of the data. In most situations this is hardly achievable, or only by tedious editing tasks. A new approach will be presented allowing interactive volume exploration by walkthrough simulations. Here, object recognition is performed by integration of computer vision techniques into the visualization pipeline. When the volume data represent a real 3D space, as for example in medical imaging, the optimal path (e.g. minimal risk of destructing vital organs in neurosurgery) can be found, thus visually programming and navigating a robot.

RoboVis – a Scenario for Using Virtual Reality Techniques in Learning Robot Development

Berthold Kirsch [†], Uwe Schnepf [‡], Ipke Wachsmuth [†]
Christian-A. Bohn [†], Axel Dirksen [†], Ulrich Licht [‡]
[†] Scientific Visualization Dept. of HLRZ
[‡] AI Research Division
GMD
Sankt Augustin
Germany

January 28, 1993

Abstract

The context of this paper is the design of an appropriate working environment for learning robots. We are interested in robots which are able to structure incoming sensor data on the basis of internal reference schemes and robot tasks. For efficiency reasons, we want to perform the learning cycles in simulated environments explored by the robot. In this paper, we outline how advanced visualization and interaction techniques as developed in the field of Virtual Reality could be employed to study the development and properties of the internal data of a semi-autonomous robot, as well as the learning process itself.

1 The scenario

Robots of the forthcoming generation are envisioned to have greater autonomy in getting organized in their environment, and they will be able to make task-dependent decisions on the basis of available world data. These robots will be learning systems which gain information about their environments individually, and will develop and maintain their own world models of the environment on the basis of their sensorimotor capabilities. This way, they create their own "Merkwelten", i.e. simple internal representations of corresponding entities in the real world.

In our scenario, a physical robot serves as a template for a simulated robot model: a virtual robot in a virtual environment. A robot controller is able to control either the physical or the virtual robot without any further modification. The virtual environment is intended to show nearly the same characteristics as the natural environment of the physical robot.

The virtual environment will be an artificial room similar to a maze, and the virtual robot will navigate through this room. In this room, there are one or several users which participate in this scenario through the use of headphones and data gloves, or simplified versions of a Virtual Reality-type of access. The user is able to inspect the robot's internal representations and to switch between different robot views. Also, it is possible for the user to interact with the robot in their shared environment, either physical or simulated.

The distributed and modular flow of control in the overall system allows to specify individual changes in the communication channels of sensors and effectors of the robot. So, arbitrary variations in the interaction between physical and virtual sensors and effectors are possible, e.g., a virtual sensor might trigger the action of a physical effector and vice versa.

2 The working environment

Our goal is to support the developer of learning robots with advanced methods for controlling, inspecting and manipulating the learning processes of these robots. To this end, we want to enable the developer to actually experience the sensorimotor capabilities of the robot in terms of both raw data as well as learned data abstractions performed by the robot controller (the *robot data view*). This robot sensor and control data has to be translated into representations accessible by human senses. In addition, we want the user to be able to experience and manipulate internal states and processes of the robot controller (the *Merkwelt view*). Finally, we want to equip the user with advanced visualization and interaction techniques in order to provide natural ways of inspecting and testing the robot's behaviour (the *user view*).

2.1 The physical robot

The physical robot is equipped with four wheels driven by two separated stepping motors, and one or several of the following sensors: ultrasonic sensors, touch sensors, simple light intensity sensors, odometric sensors, laser range finder, compass. We have started with using ultrasonic and touch sensors because of their simplicity and robustness as well as the low complexity of returned data.

2.2 The virtual robot

The virtual robot has similar sensorimotor characteristics as the physical robot plus additional abstract sensors which only exist and function in the virtual versions of the robot and its environment. These abstract sensors can be thought of as, e.g., a data glove sensor which is able to transmit various commands given by a human instructor.

3 The approach

In this section we describe how we want to realize different aspects of the working environment mentioned above. As a basis, we use a physical robot as described before. First we have to model the robot, its physical sensorimotor capabilities and its environment to realize the simulation aspect of our work. Second, we need interface communication channels between the different modules such as robot controller, world simulation and visualization. Third, we describe three different views, the *robot data view*, the *Merkwelt view*, and the *user view*, which enable the user to obtain taylored visualizations of the robot and its activities.

3.1 Modelling

Modelling can be divided into two categories, (1) geometric and (2) physical modelling of the robot, its sensors and motors, and the robot's environment. For geometric modelling, we use standard techniques from computer graphics to create the data model to be visualized. This data model is the basic reference model for the visualization taking place when adopting the user view.

The physical model of the robot and its environment will be developed to realistically simulate sensorimotor behaviour of the physical robot as well as changes in the simulated environment due to interactions of the user with the working environment. For sensorimotor simulation, we apply specific techniques for the simulation of the physical effects of various robot components such as ultrasonic, light and touch sensors, as well as motors. We make use of (precalculated) radiosity for the simulation of light, and raytracing for the simulation of touch and ultrasonic sensing. For imposing changes on the scene, we use Virtual Reality interaction techniques.

3.2 Communication

Process interface The main modules of the system are the physical robot, the robot controller, the virtual robot in its virtual environment, and the visualization. All these modules can be linked together through a parallel message passing-based programming system based on distributed and cooperating Unix and transputer processes. The robot controller communicates with the physical robot in the same way as with the virtual robot. For the robot's controlling behaviour there is no difference between communicating with the physical or the virtual robot. The learning process which is part of the robot controller is independent of the actual robot (virtual or physical) used. The visualization module receives its data either from the robot controller only (when using the physical robot) or from the robot controller and the environment (when using the simulated robot). Then these data is processed and displayed depending on the selected view.

User interface Since the working environment is adaptable to different parameters (physical or virtual robot, physical or virtual sensors used) the interface to the user interacting with the robot and the environment must be configurable as well. This may be done via keyboard commands, mouse or user interface tools which consist of buttons, sliders, browsers, or input fields. Using Virtual Reality equipment such as the data glove allow a more straight-forward approach to interact with the robot and the environment. A head-mounted display allows to explore sensor data in a more direct way. For instance, a visual grasp of robot sensor data could support the system-user cooperation in solving a given problem, e.g., obstacle detection, or maneuvering.

3.3 Visualization

The robot and the environment are modelled as different sets of simple planar surfaces. The model can be visualized as a wireframe picture as well as a more elaborated picture including static, precalculated light intensities based on radiosity. The robot currently is visualized by a cube. Later on, this simple representation will be replaced by a more realistic model of the physical robot.

We want to develop methods for inspecting internal properties and activities of the robot controller. For this, we consider two different views, the *robot data view* and the *Merkwelt view*.

The robot data view The robot data view requires new visualization techniques to translate and display non-human sensory information into a form understandable to humans. For instance, it is necessary to display the dynamic characteristics of ultrasonic beams. The robot data view is used to visualize the data from the different (physical or simulated) sensors of the (physical or simulated) robot. For this purpose, we have to transfer the non-visual quality of the (physical or simulated) sensors into visual representations understandable to humans. Adopting the robot data view, the user can examine and interpret the data which form the basis of the robot's current activities.

The Merkwelt view The Merkwelt view expresses the internal robot "knowledge" gained by ongoing learning processes. The idea to provide the Merkwelt view originates from the incorporation of machine learning techniques into the robot controller. The learning process and the knowledge state of the robot, as it is based on the robot's sensing and acting capabilities, is difficult to understand by the user and therefore will be translated into visual categories.

How to represent the Merkwelt view will be a new research topic to be dealt with in cooperation between robotics and graphics. The main focus herein will be the visualization of the gradually growing knowledge of the robot controller about its (physical or simulated) environment formed on the basis of ultrasonic sensor readings.

The user view Apart from developing methods for inspecting internal properties and activities of the robot controller, we need a visual way to inspect the overall scene. When using the robot simulation, the user view is to provide similar external viewing and interaction capabilities as in a physical testing environment. In this view, the user underlies no restrictions with respect to physical size, position, and movements. Further simulated devices such as abstract sensors and manipulation abilities for robot control algorithms may enrich the physical interaction capabilities.

4 State of the work - proposed work

Work is divided into several work packages some of which have been realized at the time of this writing, and others which still have to be implemented (based on our existing robot, its sensorimotor capabilities, and the message passing-based operating system as described before):

Geometric modelling of:

- the robot shape.
- the environment.

Physical modelling of:

- the motor behaviour of the robot.
- the ultrasonic sensors.
- the light sensors.
- the touch sensors.
- the odometric sensors.

Development of:

- learning robot control algorithms.
- user interface for:
  - user robot interaction.
  - visualization of sonar sensor data.
  - visualization of robot control data.
  - user robot communication.

The geometric modelling activities have begun using the `wavefront` data file format used in VR visualization. We are able to directly feed in the geometric model of the robot and the environment into the VR visualization system and can make use of different views and perspectives. Currently, the robot is shown as a simple cube, and the environment has simple perpendicular walls with homogeneous surface and reflection properties. We are also able to make use of different modellers, e.g., softimage, in order to develop modified or new environments.

The physical modelling aspects have been solved partly on the basis of raytracing-like simulation of sound distribution and obstacle detection (for the simulation of sonar and touch sensors). The light sensor physics will be modelled by the use of radiosity techniques for the simulation of light distribution in a scene (for the simulation of light sensors). Current radiosity techniques calculate light distributions of a particular scene offline. Powerful workstations could perform the scene rendering on the basis of the data generated by the radiosity algorithm online. This way, a particular scene having a static light distribution can be displayed from various perspectives. One disadvantage, however, is the long compute time of radiosity data as well as the lack of dynamics in case the scene is changing. These aspects limit the applicability of radiosity techniques in Virtual Reality, where real-time performance is requested. Hence, one focus will be on improving radiosity techniques with respect to efficiency and dynamics. A fast parallel algorithm for radiosity computation on the Connection Machine CM2 has been implemented and is used to prepare input for scene rendering on Silicon Graphics workstations. It would be helpful to have incremental radiosity techniques to allow real-time computation of dynamic light distributions. Future work will include the use of more advanced obstacle detection techniques developed on the CM2 in the context of VR obstacle detection.

The development of learning control algorithms is underway on the basis of a behaviour-based control approach. In this approach, we make simple reflexes available to the overall robot controller, so that raw sensor data can be transformed into motor commands quickly and reliably. A modularisation is planned for different behavioural aspects such as wall following, crossing and room traversing, turning, and obstacle avoidance.

Virtual Environments and Situated Agents

Ipke Wachsmuth
Dept. of Technology
University of Bielefeld
P.O. Box 100131
4800 Bielefeld, Germany
ipke@techfak.uni-bielefeld.de

A context for this work is the increasingly important use of computer graphics in all areas of object and scene design. An example is the visualization of virtual environments such as office rooms or buildings prior to their physical realization. The aim is to obtain realistic impressions of a construction while it is evolving and to give free way to imagination at the same time. Working with the graphics computer allows the designer to explore, and to interact with, a manipulable environment without wasting physical matter and with the ability to readily change the immaterial model. Due to the need of having to communicate ideas of complex form to a technical device, though, the designer may face crucial obstacles in the process of designing. Hence, a comfortable user interface is of special importance to keep the designer free from technical considerations such as planning of geometric details, etc.

In the paper a scenario for some of the main themes in the new research program on "Artificial Intelligence and Computer Graphics" at the University of Bielefeld is presented for discussion. One aim is to provide ways of intelligent communication with a technical system for designing and generating 3D computer graphics. To do so new AI methods and techniques are going to be applied that build on ideas of situated agents (e.g., Rodney Brooks at MIT), graphics agents (e.g., Norman Badler at U of Pennsylvania), and of interface agents (e.g., Pattie Maes at MIT Media Lab). The concept of an "intelligent mediator" is proposed – a system which integrates certain abilities to perceive, act, and communicate and which is able to exploit these abilities in the fulfilment of a particular task and adapted to the actual situation. As one feature a synthetic agent graphically visualized is used to place the designer's eye in the virtual environment and to allow the use of situated language and of gestures in interactive modelling.

Virtual Reality as a novel man machine interface for a multirobot control system

E. Freund, J. Roßmann
Institute of Robotics Research
Otto-Hahn-Str. 8
4600 Dortmund 50
Email: rossmann@damon.irf.uni-dortmund.de

The design of a man machine interface for a multirobot control system that enables users to specify processing tasks after a short training period is difficult to accomplish. Conventional programming techniques, such as teach-in of seperate movements and CAD-based offline programming systems, require a basic knowledge of robot programming and vocabulary like working area, singularities, pathtypes etc. These techniques allow the writing of simple programs. But they are dedicated for single-robot-workcells, therefore they make poor use of a multirobot system´s flexibility. For example, it is not possible to program a task where the task's parts can run parallely or sequentially depending on the availability of the robots at runtime. Furthermore, the conventional programming techniques do not allow object handling with more than one robot, which becomes neceessary if the object cannot be handled with one robot due to its size or weight.

The multirobot control system IRCS developed at the Institute of Robotics Research is used for the CIROS-testbed. This is an exact scale model of a space laboratory, that contains two 7-link-robots with overlapping working areas executing experiment servicing tasks. The IRCS provides the functionality that is needed to handle objects with several coordinated robots and it has the feature of automatic action planning, which greatly simplifies the programming. For example, a valid task description is "put sample x into heater-slot y". The automatic action planning decomposes this high-level task description using an environment model into elementary operations, which can be executed by the robot control system. If both robots are available, the elementary operations are executed parallely, i.e. one robot gets the sample from the sample stock, while the other one opens the heater door. If it is neccessary to exchange grippers, then this operation is generated automatically.

The project VITAL (Virtual Reality for telerobotics) has the goal to improve the IRCS by adding a Virtual Reality component as an intuitively operatable man machine interface. The advantage of the VR consists in the further minimization of the training time for the multirobot control system´s programming, which has been improved greatly by employment of the automatic action planning. The ease of operation is given by the fact that the user does his handling tasks inside the VR environment, where he doesn´t have to consider robot specific problems, and the robot system executes these tasks in the real working cell.

The presentation explains why only in special instances it makes sense to imitate the movements of the data-glove inside the VR with the robots. As an alternative a second mode of operation is proposed, which uses the automatic action planning´s abilities for optimal execution of tasks described within the VR. Furthermore, a method for avoiding collisions between the robots or between the robots and their environment in both modes of operation is briefly presented.

Virtual Manufacturing

Presented by: Philip Willis
Project colleagues: Adrian Bowyer, Richard Taylor, Gary Bayliss

University of Bath
Bath BA2 7AY, UK

Abstract

There is a growing interest in virtual reality techniques for many applications, ranging from mass-market entertainment to scientific visualization. The Virtual Manufacturing (VMAN) project at the University of Bath is exploring what desk-top VR techniques can do to assist the computer-aided design of engineering components. Our aim is to supply a virtual mechanical engineering workshop which can be used to make prototypes using computer-modelling techniques. This safe, interactive environment also captures enough information to allow the subsequent creation of the actual components with a numerically-controlled milling machine. The project is newly-established and we describe here our hopes and expectations for its progress.

---

1 Introduction

Computer-aided design of three-dimensional components has come a long way since the very early days of computer graphics. It is however still the case that it is difficult to bring together all the aspects of the design process needed to give a full manufacturing specification of a component. Modellers are very good at representing geometry and topology in a way which allows rendered images to be produced. To be useful, the basic modelling process has to be extended to incorporate attributes, such as material; tolerances; and blends. Even then, we still lack the information needed to manufacture the component by machine. In particular, determining a tool-path which will correctly and safely make the component is a major difficulty. A further problem centres on the user interface. Human beings are notably poor at visualizing in three dimensions, though well-used to manipulating objects in 3-space. Computers are good at calculating in any number of dimensions but less easy to interact with above two spatial dimensions.

2 Virtual reality and the design process

Virtual reality promises a ready ability to interact in three dimensional space. In particular it is possible to provide a visual simulation of familiar real-world environments, and to make changes within such an environment. We have accordingly set ourselves the task of creating a virtual mechanical engineering workshop with the aim of allowing the user to design by direct creation of the component, using virtual machinery, capturing what the user does, and later 'replaying' the actions on a real, numerically-controlled machine. This approach is certainly open to criticism: this is not how designers design, construction is a separate process to design, some practical aspects are not relevant during the creative part of design, design should be imagination-limited not machinery-limited and so forth. While we accept that such a case has its strengths, we also believe that there is much to be gained by exploring modelling-by-machining. An engineer may well machine an approximate form as part of the design experimentation, so this is a legitimate part of the design process which is currently not addressed by computer systems. Further, and most importantly, using virtual tools is a good way of ensuring that the component can really be made and indeed of verifying toolpaths before a real machine and a real piece of material collide, with possibly dangerous and certainly expensive results. Verification is an important part of the design process. There are other virtues too, especially of using an involving medium like VR.

3 Desk-top or immersive VR?

¿From the start, we wanted the viewer to be able to work in stereo, in order to be able to inspect the workpiece and to judge its overall shape and balance. We have tried the fully-immersive, headset VR systems and it was apparent that they were not suitable for our needs. Firstly, they were tiring to use for extended lengths of time. Secondly the image quality was poor. Thirdly there was a lag in picture update which made precise work rather difficult. Fourthly, the user can only see the virtual world, making it impossible to consult real working drawings, to share the design process with colleagues or even to type. Fifthly, certain forms of interaction, such as clicking on a menu, are much harder (to the point of irritation) in an immersive virtual world. We therefore concluded that desktop VR, the use of a stereo visual system in an otherwise familiar computer workstation environment, was a better approach. The system we chose has liquid crystal spectacles, synchronized by an infra-red beam with an alternating left-right view on the sharp workstation monitor. It is possible to see the real world through these spectacles (slightly lowered in brightness) and of course several users can each have a pair. The weight and bulk are both low, so they are not tiring to use, and they can be put on and taken off immediately with no set-up procedure.

We have installed a Silicon Graphics Reality Engine for the main demonstrator on this project, as this supports the real-time texture that we intend to use later. We also have an SG Elan with the stereo option, for visual development, and two 24 bit Indigo 4000s for software development.

4 Technical progress

To date, we have relied on existing software as we patched together an early prototype in order to get a feel for the possibilities. The solid modeller is one which has had extensive development here at Bath, called DORA (Divided Object Raytrace Algorithm). Traditionally this has been used with a descriptive input language, SID, which converts into a structure which DORA ray-traces. For the current project we have introduced two changes. Firstly, the output is now converted into polygons so that the SG display hardware can render it directly (and in real time). Secondly, we accept as input the G-codes which are normally used to drive a real numerically-controlled milling machine. We use these to drive a virtual tool across a block of virtual material, cutting a path which the in fact is used to remove the corresponding shape from our computer model. In the early system this is done in near real-time in the sense that we show the tool moving continuously but update the model only when the tool changes direction. The visual effect is thus seen correctly only at these instances. This is not intended to remain the case and we expect full real-time updating soon. At the time of writing the project has been active for only four months, so we are quite satisfied with current results.

5 Planned improvements

There are many experiments that could be tried, some relating to the model-construction and some to the human interface. As a base, we expect to put together a system which allows the user to drive the virtual milling machine directly, with the display updating in real time, and with the model at all times essentially up to date. We expect this to generate output which can be used directly to control a real milling machine, and indeed our largest computer is installed in an engineering workshop a few metres from an actual numerically-controlled milling machine for precisely that reason.

Texture mapping is likely to be important to give a greater illusion of reality, both for the material surface and also to give greater visual realism to the rotating tool and the surrounding workshop area (necessary to give a sense of scale and depth)

There are also non-realistic effects which can be put to good use. For example, as we intend to model the overall shape of the machinery and the tools, it is quite possible to check the toolpath and to highlight or prohibit cutting of the machine rather than the workpiece. We can also check the tool speed and known material properties and show material and tool temperature, colour-coded to highlight hot-spots. The human interface can be enriched in non-realistic ways. For example, pop-up instructions can be made to float in space near to the object to which they refer. Similarly arrows can be made to point to features of interest. In visual simulation it is common to reduce levels of detail to ensure adequate updating in important parts of the scene. In our kind of VR world there is an argument for using level-of-detail management to reduce the visual clutter to allow the user to concentrate on the important task. This can be done either by removing the extra detail or by 'lowlighting' it so that it only appears dimly.

One overall interest is the extent to which we can use non-realistic operations initially, then impose realistic ones later. For example, there is no need to respect actual milling speeds when creating a virtual prototype, so all constraints on speed, tool-material temperature, clogging etc can be removed. Holes can be drilled instantly if we are only interested in the resulting shape.

Finally, the user is able to see the real world and so there is an argument for making familiar machine controls available as a physical mock-up, rather than as a virtual presentation. It is our current belief that this will be more acceptable and indeed more reliable as a source of data.

6 Summary

We are studying the use of desk-top virtual reality as an aid to manufacturing design. The project is newly established, though we have demonstrated an early prototype. We aim to use the system to generate real objects on a numerically-controlled milling machine, using data created from our virtual milling machine. Our experiments are intended to find the best way of presenting this technology to actual designers.

Parallel Processing for Interactive Photorealistic Building Walkthroughs

Alan Chalmers
Department of Computer Science
University of Bristol
Bristol, BS8 1TR
United Kingdom

Abstract

Accurate visualisation of a design is an important aspect of an engineering process. This is especially true in those disciplines involving building design and urban planning. Computer simulations that allow a user to ''walk through'' a proposed building extension or modification enable the proposal to be visualised and thus evaluated before the expensive construction phase is commenced. The more realistic the proposed building model is, the better the evaluation will be, but such a model may be extremely complex consisting of thousands of objects. A walk-through which required the user to wait for possibly hours for each image to be rendered would be of little use. Test have shown that at about 6 frames per second the virtual building illusion ''works'' and the user is able to comfortably navigate the building.

This paper discusses PING (Parallel processing for INteractive Graphics), which will implement the particle tracing method of computing global illumination on a minimum path parallel processing system in order to achieve the required interactive photorealism for building walk-throughs.

Animating Geometric Algorithms

David Dobkin

Department of Computer Science
Princeton University
Princeton, NJ 08544
(dpd@cs.princeton.edu)

As modeling problems in graphics and motion planning problems in robotics grow in complexity, the needs arises for sophisticated algorithms for doing geometric computations. Many such algorithms have been developed by computational geometers. However, there has been limited technology transfer between the theoreticians and practitioners. The difficulty seems to arise because of the complex nature of many theoretical algorithms. This makes it difficult for practitioners to implement these algorithms or even to understand key aspects of the algorithm which might be applied to simplify computational tasks. Our work is aimed as helping with this technology transfer. We do so by implementing some of these algorithms and producing animations which aid the viewer to understand the essence of the algorithm. As a part of our work, we are developing tools that make it easier to implement, debug and animate geometric algorithms. We describe here the algorithm animation and designs of a tool kit to ultimately make this task easier. These animations describe fast algorithms for computing all intersections of n line segments in the plane and for detecting intersections of suitably preprocessed convex polyhedra in sublinear time. We will describe these algorithms and their animations.

A longstanding open problem in computational geometry was the problem of determining in optimal time all intersecting pairs among a collection of n planar line segments. This problem lies at the heart of object space algorithms for hidden surface removal. Many other intersection algorithms relating to ray shooting and shadow casting have similar structure so that methods which apply here are likely to be worthwhile for various other problems. Ideally, an algorithm to solve this problem would run fast when there are few intersections, slowing down to report all intersections if there are many. Indeed, it is known that no algorithm can have a running time asymptotically smaller than $n \log n + k$ where k is the number of intersections reported. Initial algorithms for the problem used sweepline techniques similar to scanline algorithms for hidden surface removal. They were able to achieve the bound $n \log n + k \log n$. It could be shown that a limitation of the sweepline paradigm was its inability to break through this barrier and the problem of finding an optimal algorithm remained open. By combining a number of ideas and data structures, Chazelle and Edelsbrunner achieved this optimal time in 1988. Their algorithm is complex, though the ideas on which it is built might find application elsewhere. We believe our video makes this algorithm more accessible to potential users.

Many tasks in computer graphics and robotics involve the detection of intersections between convex polyhedra. This arises as a subroutine in motion planning tasks as well as in tasks where occlusion or interference between objects must be determined. Most often, objects to be tested do not intersect. Thus, a fast test which detects intersection or separation but computes no portion of an intersection is desirable. In joint work with Chazelle we developed algorithms satisfying this criterion. In later work with Kirkpatrick we have refined these algorithms. A result of this is a data structure of independent interest for organizing convex polyhedra so that they can be rapidly searched. This structure can be used to give algorithms of running time $O(log\ n)$ for determining whether a plane intersects such a polyhedron. An algorithm of running time $O(\log^2 n)$ which can be used to detect if two such polyhedra intersect. This data structure and algorithm are best described by the video we show.

The tools we are developing to aid in creating our videos are of independent interest. Our animations differ from those typical in the computer graphics community. We seldom are near the edge of the art of image synthesis techniques. Indeed, our models tend to be polyhedral and we want to highlight their rough edges rather than smooth them. On the other hand, our algorithm tend to require sophisticated data structures and it is often desirable to show multiple views of an algorithm as it progresses.

Surfaces in an object-oriented geometric modeler

Reinhard Klein

Universität Tübingen, WSI/GRIS
Auf der Morgenstelle 10, C9
W-7400 Tübingen
Germany
Email: reinhard@gris.informatik.uni-tuebingen.de
Phone: 0049-(0)7071-295462
Fax: 0049-(0)7071-295466

To use the advantages of different surface representations, such as tensor-product surfaces, surfaces over triangular regions, implicitly defined surfaces, etc., it is necessary to integrate these surface types together with a large variety of algorithms into one programming environment.

Inheritance and polymorphism of object-oriented languages offer the opportunity of an implementation, which is not bounded to a certain representation of the surface. The most reasonable way to benefit from such an object-oriented approach is to use existing and if necessary even to develop new algorithms that are based on the functionality provided by as many surface representations as possible.

An object-oriented design for surfaces is presented and the advantages of such a design is illustrated by examples.

Curves and Surfaces With Minimal Curvature for Graphics and Robotics

Günther Greiner
IMMD IX . Universität Erlangen
Am Weichselgarten 9. 8520 Erlangen

Abstract

Both in Graphics and Robotics one often faces the problem to construct "optimal" curves and surfaces subject to certain constraints. Here optimal typically means that the curve or surface minimizes length, area, curvature or other geometric properties.

In Graphics for example, scattered data interpolation as well as blending surfaces lead to such problems. In scattered data interpolation one has to construct a surface which does not oscillate too rapidely and goes through a sampled set of points in the space. Finding a blending surface (i.e. a rather smooth transition between primary surfaces) amounts to construct a (nice looking) surface which satisfies certain boundary conditions determined by the primary surfaces.

In Robotics, path planning can be considered as a problem of the above type. One has to find a curve which connects two points and optimizes a mixture of length, curvature and the distance to obstacles.

All these problems can be solved by finding a surface and curve respectively, which satisfies two conditions:
— it has to fulfill certain constraints and
— it has to minimize an appropriate functional, which measures the
total curvature and surface area (arc length).

This optimization problem is highly nonlinear, a direct solution (if possible at all) will be very involved. We present an iterative procedure. Instead of investigating the nonlinear problem, we consider a sequence of quadratic variational problems. Thus, in each step of the iteration, one has to solve a linear system. In each step one obtains a curve or surface which satisfies the constraints. The smoothness of the curve/surface will increase progressively.

For practical purposes, the problem is considered in a (sufficiently large) space of quadratic or cubic spline curves. Or in a space of tensor spline surfaces. Thus the solution we obtain can be visualized easily.

We describe the general procedure in detail and explain how one can apply it to generate blending functions, to interpolate scattered data and find optimal paths. We also compare it to existing methods and outline possible further applicatons.

Triangular B-Splines for Modeling in Graphics and Robotics

Hans-Peter Seidel
Universität Erlangen
IMMD IX - Graphische Datenverarbeitung
Am Weichselgarten 9, D-8520 Erlangen, Germany

Abstract

Triangular B-splines are a new tool for the modeling of complex objects with non-rectangular topology. The new B-spline scheme is based on blending functions and control points and allows to model piecewise polynomial surfaces of degree n that are C^{n-1}-continuous throughout. A first test implementation of the new scheme has succeeded in demonstrating the practical feasibility of the fundamental algorithms underlying the new scheme.

In this talk we discusses applications of triangular B-splines for modeling in graphics and robotics.

One of the features that make triangular B-splines attractive for applications in graphics and robotics is their low degree: It is possible, e.g., to construct C^1-continuous surfaces with piecewise quadratics (total degree $d = 2$) (in contrast to the more standard bi-quadratic tensor-product surfaces having total degree $d = 4$).

Furthermore, it is possible to represent any piecewise polynomial surface (rectangular or non-rectangular topology) as a linear combination of triangular B-splines. Thus, triangular B-splines provide a unified data format.

Finally, triangular B-splines are ideally suited for blending applications. Using triangular B-splines, it becomes possible, e.g., to achieve smooth C^1-blends with piecewise quadratics. Again, this compares favourably with other approaches, where the parametric degree of the blending surfaces is typically exceedingly high.

Symbolic Relationships as Basis for Assembly Planning

R. Gutsche, F. Röhrdanz and F. M. Wahl
Institute for Robotics and Computer Control
Technical University of Braunschweig
Hamburger Str. 267, D-W-3300 Braunschweig, F.R.G.

Abstract

In modern and flexible manufacturing processes the automatic generation of assembly plans and their subsequent execution by a robot will be one of the key technologies. Our assembly planning system consists of two consecutive steps. During the first one all constraints are taken into account which are caused directly by the assembly itself (internal constraints), like *stability*. During the second step the possible solutions are further restricted, because constraints concerning the robot and its environment are taken into account. Examples for these external constraints are the shape of the robot's work space and the kind of the gripper. In this paper only the first step of the whole system will be described.

An overview about several techniques dealing with mechanical assembly planning can be found in [1]. Most of the systems are working with an ideal geometric model and with homogeneous transformations to describe the assembly. Thus the resulting plan consists often of a set of ideal frames without any additional information. As the environment can't be modeled exactly and as a real robot has always a position uncertainty, sensor information has to be added later during the execution of the plan. But this method has the following drawbacks: the description of an assembly using numerical homogeneous transformations is more difficult and more faulty than using *natural* symbolic descriptions and the automatic adding of sensor information to a plan consisting only of a numerical description is very difficult.

Our assembly planning system works with rigid parts, modeled in two levels. In one level parts look like bodiless objects consisting only of a set of mathematical frames [3]: one base frame and several feature frames, like *face* or *hole*, describing one special feature of the part in relation to the base frame. In the other level the parts are represented as polyhedrons and optionally as CSGs. Often some faces of the polyhedron correspond to infinite features

frames and vice versa, but that isn't necessary.

The geometrical relations of the whole assembly consisting of several parts is described only with symbolic spatial relationships between the feature coordinate systems, like *face against face*. A cycle finder computes automatically the numerical homogeneous transformation between the parts. The principle of the cycle finder like published in [3] and our extensions will be shown in the paper. The results of our cycle finder are the ideal numerical homogeneous transformations between the parts and the corresponding symbolical relations, which can be used during a latter step to generate additional sensor information for the execution of the plan. Thus the end position can be defined by a special sensor event instead of using the ideal frame from the exact model.

Decomposing a mechanical assembly into two subassemblies a lot of constraints have to be taken into account, e.g. *stability* of the resulting two subassemblies, *geometric-feasibility* and *mechanical-feasibility* of the assembly process of these two subassemblies [2]. In the full paper, we will describe our assembly planning system including the modeled constraints. Some ideas of our system are similar to the system introduced in [2]. But in addition, we have integrated a symbolic description of the assembly which is used to compute automatically the numerical values of the homogeneous transformations between the parts (s. above). Consequently it isn't necessary to specify manually homogeneous transformations between components.

Our assembly planning system was designed as base framework with the symbolic spatial description of the assembly, a cut-set method and AND/OR graph plan representation to deliver a good environment for procedures which should check additional constraints, like *stability*. Each of these procedures delivers automatically *no failure*, if the appropriate constraint is fulfilled, *failure*, if the constraint isn't fulfilled and *unknown*, if the procedure can't decide between the other two cases. In the last case the user will be asked to answer the question.

References

[1] Luiz S. Homen de Mello and Sukhan Lee, editors. *Computer-Aided Mechanical Assembly Planning*. Kluwer Academic Publishers, 1991.

[2] Luiz S. Homen de Mello and Arthur C. Sanderson. A correct and complete algorithm for the generation of mechanical assembly sequences. *IEEE Transactions on Robotics and Automation*, 7(2):228–240, 1991.

[3] R. J. Poppelstone. Specifying manipulation in terms of spatial relationships. Technical Report 117, Department of Artificial Intelligence, University of Edinburgh, 1979.

From CAD models to Assembly planning

Caracciolo R. - Ceresole E.
Dipartimento di Innovazione Meccanica e Gestionale
Università di Padova - Italy

De Martino T. - Falcidieno B. - Giannini F
Istituto per la Matematica Applicata- C.N.R.
Genova - Italy

The realization of an automatic assembly planner is strictly related to the knowledge about the assembled components and the workcell. Geometric knowledge of components, stored in CAD models, are not sufficient for the assembly planning solution which, in general, corresponds to a sequence of elementary actions. Most advanced assembly planners are described by graphs, where nodes represent assembly states and connecting arcs represent actions.
Thus, assembly planners work on a higher level of information than geometric modelling, so they cannot refer to the same structures and data types defined in CAD models. The required data include adjacencies between components, assembly directions, mating operations referred either to a component or to a pair of components or to the whole assembly.
Some of these data can be extracted from the low level geometric description, but it is more powerful and easier to extract or to associate them to models which include also functional and technical information. These are the so called Feature Based Models.

In this paper, we present a module developed for the National Project on Robotics, sub project P.R.O.R.A., regarding Automatic Programming of Assembly Robot by Feature Recognition.
The presented module is characterised by some fundamental aspects:

- adjacencies identification between components;

- integration of the adjacency information with the feature based description in a data structure which enables the manipulation at different levels of detail;
- identification of the assembly operations and their execution directions. performed by a rule based module

We have adopted graphical facilities for the classification of the assembly features. The problem of feature classification is well known in the automation environment, in fact it is quite impossible to give a "universal" definition of what a feature is. It strictly depends on the context, i.e. manufacturing, assembling, design,.... and also in the same context it may change depending on the available tools. Thus, in order to realise a system suitable to satisfy different needs, we have provide an easy system for the interactive definition of the features of interest. A deep knowledge neither of the internal description nor of the system code are needed by the user. In fact, he has only to create the CAD representation of an example of the feature to be recognised Moreover he can associate functional parameters to the feature through the graphical user interface.

The structure resulting from the CAD model analysis associates the adjacencies information to the feature description. This is very useful to determine the required operations, the access directions and the knowledge of the available space.

Starting from this structure, a rule based module performs the search of all possible mating operations associated with each component. The compatibility of these operations with the selected robot is also verified. Moreover, the feature based model and all the identified operations are used to make fine-motion interference checks. In this way, geometric constraints and consequent precedence relationships are determined

We have also taken advantages from graphical facilities to have a feedback of the resulting assembly plan. Thus, we have integrated our sequence generator into a commercial CAD system oriented to robotics. For the assembling simulation we have used both facilities of the system for the optimisation of the graphical representation and newly implemented ones. In this way, assembly sequences are automatically represented and collision detection is performed. In order to avoid collisions, a module has been developed to automatically update the gross-motion trajectories.

Frederik W. Jansen

Faculty of Technical Mathematics and Informatics
Delft University of Technology
Julianalaan 132
2628 BL Delft
The Netherlands

Interactive Feature Modeling

Feature modelling offers new opportunities to add functional information related to design, analysis, manufacturing and assembly, to geometrically defined product models. A widely used approach is to recognize features from the geometric model. For different applications, different classes of features are applied and therefore different feature recognition passes have to be performed. Changing the geometric model will therefore require a completely new evaluation of the derived feature models. In this presentation an integrated approach will be presented that combines interactive modeling with semi-automatic feature conversion capabilities. Each view will be represented by a window showing the feature representation of the model for that particular aspect or application. Proposed modifications will be propagated to the other views by means of an constraint graph. One of the applications that is considered in the near future ist the use of features for automatic assembly planning for the Delft Intelligent Assembly Cell (DIAC). This assembly cell consists of two robots equiped with vision and sensor systems that can handle fairly complex assemblies. Use of assembly features will increase the possibilities of, among other things, semi-automat sequence and grip planning.

Projective Geometry for Robot Vision

K. Voss, S. Abraham, R. Neubauer, M. Schubert
Friedrich Schiller University Jena / Germany
Department of Mathematics and Informatics, UHH, 17.OG, D-O 6900 Jena

In this paper, an overview is given about the extensive use of projective geometry and vector algebra for problems in 3D scene analysis. The model of the perspective mapping of space points $\mathbf{r}=(x,y,z)$ to image points (ξ,η) is the only mathematical object under investigation:

$$\xi = \frac{a_{10} + \langle \mathbf{a}_1 , \mathbf{r} \rangle}{1 + \langle \mathbf{a}_0 , \mathbf{r} \rangle} \quad , \quad \eta = \frac{a_{20} + \langle \mathbf{a}_2 , \mathbf{r} \rangle}{1 + \langle \mathbf{a}_0 , \mathbf{r} \rangle} \ .$$

The projective camera parameters a_{ik} are determined by a least-square method of camera calibration. The residual errors are of order ± 1mm in a robotic scene of about 300^3 mm^3 and a distance of about 1000 mm between scene and camera. From the projective camera parameter, one can derive in a very simple manner motion invariants which are directly related to the physical camera parameters (e.g. principal point of the camera). Also location and orientation of the camera (e.g. direction of the optical axis) follow from the a_{ik}.

In the second part of the paper, the epipolar geometry of more-camera experiments will be investigated, and it is shown that the vector algebra yields relatively simple formulas using the camera vectors $\mathbf{a}_1,\mathbf{a}_2,\mathbf{a}_3$. The epipolar-rectification of more-camera images are investigated, and applications are given for point matching in correspondence problems.

Finally, basic problems of monocular robot vision are discussed, also in the projective-geometry / vector-analysis approach. It is shown that using CAD information, not only the 3D reconstruction for spheres, cones and cylinders can be solved in a short way, but also the full reconstruction of planar n-point configurations with $n>3$ is described in terms of projective geometry: simple systems of linear equations give the location and the orientation of the planar CAD object in the robot coordinate system.

The framework given here is theoretically closed and numerically effective. Therefore only a small set of procedures for least-square estimations and vector algebra is needed to handle the problems of camera calibration, epipolar matching, and 3D reconstruction.

Depths Data Acquisition

R. Sasse, TU Berlin

Abstract

Subject of the talk is the range measurement with computer vision systems, in this case the so-called *active stereo*, which involves an active source of light and a passive stereo set-up. This measurement principle and a special application of it, the *active stereo with continuous colour stripes*, will be introduced.

The measurement principle uses the additional light source to project an easily identifiable pattern onto the scene. In the special application of active colour stereo, the pattern is composed of coloured vertical stripes. The difference in colour between adjacent stripes is so small that the pattern may be called continuous. Thus the correspondence search is reduced to the search for points of the same colour.

This technique sports the following advantages:

- The correspondence search (matching) is heavily simplified.
- Unique marking a set of points avoids ambiguities, which results in very dense range maps because "negative false" matches, such as familiar from passive stereo, are unlikely.
- The method is insensitive to colour changes in the pattern caused by scene objects because they affect both cameras in the same way.
- Position and orientation of the light source are irrelevant because they do not influence the range computation.
- Correspondences of all points visible in both cameras can be found from a single pair of images (snap-shot). This enables the range image generation of fast moving objects.
- The usage of a continuous pattern enables high resolution and accuracy. This is only limited by the resolution of the cameras.
- The simplification of correspondence analysis enables the employment of simple algorithms and parallel computation to decrease evaluation time.

Disadvantages of the technique are reduced to problems inherited from the underlying measurement principles. These are namely the missing part problem and problems caused by absorbed or reflected energy.

We present quantitative and qualitative results of the experimental set-up and will finally discuss methods to improve and extend the measurement method.

Dr. André Gagalowicz

Vision-Driven Home Robots

The purpose of this work is to study the vision task of a home robot. We want the robot to "understand" perfectly the workspace in which he is evolving so that he can perform all kinds of different tasks, such as bring coffee, close a door, switch a light on and off,... We shall not consider speech analysis/synthesis useful for the interface with the robot, neither robotics nor specialised hardware which are fundamental for the solution of the problem. We restrict ourselves to the vision task.

We first propose an interactive technique in order to create the data base modelling the robot knowledge. It is here a pure geometric decription of the robot workspace. We then propose a vision task where, from a given stereo pair of the scene observed, we produce a complete description of the scene including a polyedral model and a photometric one. We finally design a synthesis program using this complete description as input. With this program, we compute a stereo pair related to the same focal planes as those of the cameras from which the initial stereo pair was taken. The idea is then to use the difference between the natural and the synthetic stereo pair on order to improve the scene description. This, in its turn, implies an improvement of the vision and the synthesis procedures.

Automatic Model-Generation for Image Analysis

Abstract

A. Winzen
Lehrstuhl für Informatik 5 (Mustererkennung)
Universität Erlangen-Nürnberg
Martenstr. 3
D-8520 Erlangen

A model-based vision system consists of components performing tasks like preprocessing and segmentation, image analysis and a knowledge base containing models of objects and scenes which should be analysed. Models suited for image analysis have to provide information different from correct geometric data. Information represented by such models depends on segmentation results, because only those features which are stable, i.e., the detection of them by the segmentation process is reliable, should be included in the model. Using additional unstable features unecessarily enlarges search space for matching models and segmentation data during image analysis, because objects are recognized by comparing features in the models and features found in the image. Statistical information about numerical data, such as lengths of line segments and positions of vertices, is also useful for measuring the quality of a matching during image analysis. These are reasons for an automatic generation of models from samples of image data. Threedimensional models are generated by integrating samples of range data from multiple views of an object. Models are represented using a semantic network, with a structure similar to CAD-models. The structure of such models, automatic model-generation and experimental results will be described.

Dagstuhl-Seminar 9316: List of Participants

17.06.93

Norbert **Ahlbehrendt**
FFT e.V. Berlin
Gebäude 8.17
Rudower Chaussee 5
O-1199 Berlin
Germany
tel.: +49-30-6704 5297

Bernhard **Brunner**
DLR - Oberpfaffenhofen
Institut für Robotik und Systemdynamik
W-8031 Oberpfaffenhofen
Germany
bernhard@donau.df.op.dlr.de
tel.: +49-8153-28-488

Roberto **Caracciolo**
Padua University
Dept. of Innovation in Mechanics
and Management
Via Venezia1
Padova Italy
tel.: +39-49-8286 820

Alan **Chalmers**
University of Bristol
Dept. of Computer Science
Queen's Bld.
University Walk
Bristol BS8 1TR
Great Britain
alan@compsci.bristol.ac.uk
tel.: +44-272-30 31 09

Fan **Dai**
TH Darmstadt
Fraunhofer-Institut für
Graphische Datenverarbeitung
Wilhelminenstr. 7
W-6100 Darmstadt
Germany
dai@igd.fhg.de
tel.: +49-6151-155-2 07

Teresa **De Martino**
CNR / Instituto per la Matematica
Applicata
Via De Marini 6
I-16149 Genova
Italy
demartino@image.ge.cnr.it
tel.: +39-10-6475-662

Mingyue **Ding**
TU Braunschweig
Institut für Robotik u. Prozeßinformatik
Hamburger Str. 267
W-3300 Braunschweig
Germany
tel.: +49-531-391-74 50

Hans-Heino **Ehricke**
Universität Tübingen
Wilhelm-Schickard-Institut für Informatik
Auf der Morgenstelle 10 C9
W-7400 Tübingen
Germany
ehricke@gris.informatik.uni-tuebingen.de
tel.: +49-7071-29-5464

Max **Fischer**
TU - München
Arcisstr. 21
W-8000 München
Germany

André **Gagalowicz**
INRIA
Domaine de Voluceau
Rocquencourt
BP 105
F-78153 Le Chesnay Cedex
France
ag@bora.inria.fr
tel.: +33-1-39 63 54 08

Günther **Greiner**
Universität Erlangen-Nürnberg
IMMD IX
Graphische Datenverarbeitung
Am Weichselgarten 9
W-8520 Erlangen
Germany
greiner@informatik.uni-erlangen.de
tel.: +49-9131-699-145

Ralf **Gutsche**
TU Braunschweig
Institut für Robotik u. Prozeßinformatik
Hamburger Str. 267
W-3300 Braunschweig
Germany
rgu@rob.cs.tu-bs.de
tel.: +49-531-391-7452

Gerhard **Hirzinger**
DLR - Oberpfaffenhofen
Institut für Robotik und Systemdynamik
W-8031 Oberpfaffenhofen
Germany
df2n@vm.op.dlr.de
tel.: +49-8153-28-401

Frederik W. **Jansen**
Delft University of Technology
Fac. Technical Math. and Informatics
P.O. Box 356
NL-2600 AJ Delft
The Netherlands
fwj@duticg.twi.tudelft.nl
tel.: +31-15-795-517

Berthold **Kirsch**
GMD Schloß Birlinghoven
HLRZ.VIS
Postfach 1316
W-5205 St. Augustin 1
Germany
kirsch@gmd.de
tel.: +49-2241-14-2706

Reinhard **Klein**
Universität Tübingen
Wilhelm-Schickard-Institut für Informatik
An der Morgenstelle 10
W-7400 Tübingen 1
Germany
reinhard@gris.informatik.uni-tuebingen.de
tel.: +49-7071-295 462

Claudio **Laloni**
TU Braunschweig
Institut für Robotik und Prozessinformatik
Hamburger Str. 267
W-3300 Braunschweig
Germany
cla@rob.cs.tu-bs.de
tel.: +49-531-391-7460

Heinrich **Müller**
Universität Dortmund
Fachbereich Informatik Lehrstuhl VII
Postfach 50 05 00
W-4600 Dortmund 50
Germany
mueller@ls7.informatik.uni-dortmund.de
tel.: +49-231-755-6324

Ralf **Neubauer**
Friedrich-Schiller-Universität
Institut für angewandte u. technische Informatik
Lentragraben 1
O-6900 Jena
Germany
tel.: +49-3641-82-25433

Werner **Purgathofer**
Technische Universität Wien
Institut für Computergraphik
Karlsplatz 13/186
A-1040 Wien
Austria
purgathofer@eigvs4.una.ac.at
tel.: +43-1-58801-4548

Frank **Röhrdanz**
TU Braunschweig
Institut für Robotik u. Prozeßinformatik
Hamburger Str. 267
W-3300 Braunschweig
Germany
fro@rob.cs.tu-bs.de

Francisco José **Rodriguez**
Universidad Politecnica de Madrid
Dep. Automatica Ingenieria
José Gutiérrez Abascal 2
28006 Madrid
Spain
uzbaho@disam.upm.cs
tel.: +34-1-561- 69 89

Ralph **Sasse**
TU Berlin
Institut Technische Informatik
FR 2-2/FB 20
Franklinstr. 28-29
W-1000 Berlin 10
Germany
sasse@cs.tu-berlin.de
tel.: +49-30-314-73483

Uwe **Schnepf**
GMD Schloß Birlinghoven
13. KI
Postfach 1316
W-5205 St. Augustin 1
Germany
uwe.schnepf@gmd.de
tel.: +49-2241-14-2704

Michael **Schubert**
Friedrich-Schiller-Universität
Institut für angewandte u. technische
Informatik
Lentragraben 1
O-6900 Jena
Germany
tel.: +49-6341-82-26091

Hans-Peter **Seidel**
Friedrich Alexander Universität
Fakultät für Informatik
W-8520 Erlangen
Germany
seidel@informatik.uni-erlangen.de
tel.: +49-9131-699-1 42

Peter **Stöhr**
TU München
Institut für Informatik
Arcisstraße 21
W-8000 München 2
Germany

Wolfgang **Straßer**
Universität Tübingen
Wilhelm-Schickard-Institut für Informatik
Sand13
W-7400 Tübingen
Germany
tel.: +49-7071-296356

K. **Voß**
Friedrich-Schiller-Universität
Institut für angewandte u. technische
Informatik
Friedrich-Schiller-Str. 2
O-6900 Jena
Germany
tel.: +49-3641-82-25431

Roland **Vollmar**
Lehrstuhl Informatik für Ingenieure
u. Naturwissenschaftler
Am Fasanengarten 5
W-7500 Karlsruhe 1
Germany
vollmar@ira.uka.de
tel.: +49-721-608-43 12

Ipke **Wachsmuth**
Universität Bielefeld
Technische Fakultät
Postfach 100 131
W-4800 Bielefeld 1
tel.: +49-521-106-2910

Friedrich **Wahl**
TU Braunschweig
Institut für Robotik u. Prozeßinformatik
Hamburger Str. 267
W-3300 Braunschweig
Germany
tel.: +49-531-391-7450

Philip **Willis**
University of Bath
School of Mathematical Sciences
Computing Group
Claverton Down
Bath BA2 7AY
Great Britain
p.j.willis@bath.ac.uk
tel.: +44-225-82 69 64

Andreas **Winzen**
Univ. Erlangen-Nürnberg
Lehrstuhl für Informatik 5
Mustererkennung
Martensstr. 3
W-8520 Erlangen
Germany
winzen@informatik.uni-erlangen.de
tel.: +49-9131-85-7891

Dieter W. **Wloka**
Universität des Saarlandes
Systemtheorie der Elektrotechnik
Postfach 1150
D-66041 Saarbrücken
wloka@uni-sb.ee.de
tel.: +681-302-3993

Christoph **Woenckhaus**
TU München
Institut för Werkzeugmaschinen
und Betriebswissenschaft
Außeninstitut Dornach
Karl-Hammerschmidt-Str. 39
W-8011 Dornach
Germany
tel.: +49-89-90 99 4 157

Zuletzt erschienene und geplante Titel:

K. Compton, J.E. Pin , W. Thomas (editors):
Automata Theory: Infinite Computations, Dagstuhl-Seminar-Report; 28, 6.-10.1.92 (9202)

H. Langmaack, E. Neuhold, M. Paul (editors):
Software Construction - Foundation and Application, Dagstuhl-Seminar-Report; 29, 13.-17.1.92 (9203)

K. Ambos-Spies, S. Homer, U. Schöning (editors):
Structure and Complexity Theory, Dagstuhl-Seminar-Report; 30, 3.-7.2.92 (9206)

B. Booß, W. Coy, J.-M. Pflüger (editors):
Limits of Information-technological Models, Dagstuhl-Seminar-Report; 31, 10.-14.2.92 (9207)

N. Habermann, W.F. Tichy (editors):
Future Directions in Software Engineering, Dagstuhl-Seminar-Report; 32; 17.2.-21.2.92 (9208)

R. Cole, E.W. Mayr, F. Meyer auf der Heide (editors):
Parallel and Distributed Algorithms; Dagstuhl-Seminar-Report; 33; 2.3.-6.3.92 (9210)

P. Klint, T. Reps, G. Snelting (editors):
Programming Environments; Dagstuhl-Seminar-Report; 34; 9.3.-13.3.92 (9211)

H.-D. Ehrich, J.A. Goguen, A. Sernadas (editors):
Foundations of Information Systems Specification and Design; Dagstuhl-Seminar-Report; 35; 16.3.-19.3.9 (9212)

W. Damm, Ch. Hankin, J. Hughes (editors):
Functional Languages:
Compiler Technology and Parallelism; Dagstuhl-Seminar-Report; 36; 23.3.-27.3.92 (9213)

Th. Beth, W. Diffie, G.J. Simmons (editors):
System Security; Dagstuhl-Seminar-Report; 37; 30.3.-3.4.92 (9214)

C.A. Ellis, M. Jarke (editors):
Distributed Cooperation in Integrated Information Systems; Dagstuhl-Seminar-Report; 38; 5.4.-9.4.92 (9215)

J. Buchmann, H. Niederreiter, A.M. Odlyzko, H.G. Zimmer (editors):
Algorithms and Number Theory, Dagstuhl-Seminar-Report; 39; 22.06.-26.06.92 (9226)

E. Börger, Y. Gurevich, H. Kleine-Büning, M.M. Richter (editors):
Computer Science Logic, Dagstuhl-Seminar-Report; 40; 13.07.-17.07.92 (9229)

J. von zur Gathen, M. Karpinski, D. Kozen (editors):
Algebraic Complexity and Parallelism, Dagstuhl-Seminar-Report; 41; 20.07.-24.07.92 (9230)

F. Baader, J. Siekmann, W. Snyder (editors):
6th International Workshop on Unification, Dagstuhl-Seminar-Report; 42; 29.07.-31.07.92 (9231)

J.W. Davenport, F. Krückeberg, R.E. Moore, S. Rump (editors):
Symbolic, algebraic and validated numerical Computation, Dagstuhl-Seminar-Report; 43; 03.08.-07.08.92 (9232)

R. Cohen, R. Kass, C. Paris, W. Wahlster (editors):
Third International Workshop on User Modeling (UM'92), Dagstuhl-Seminar-Report; 44; 10.-13.8.92 (9233)

R. Reischuk, D. Uhlig (editors):
Complexity and Realization of Boolean Functions, Dagstuhl-Seminar-Report; 45; 24.08.-28.08.92 (9235)

Th. Lengauer, D. Schomburg, M.S. Waterman (editors):
Molecular Bioinformatics, Dagstuhl-Seminar-Report; 46; 07.09.-11.09.92 (9237)

V.R. Basili, H.D. Rombach, R.W. Selby (editors):
Experimental Software Engineering Issues, Dagstuhl-Seminar-Report; 47; 14.-18.09.92 (9238)

Y. Dittrich, H. Hastedt, P. Schefe (editors):
Computer Science and Philosophy, Dagstuhl-Seminar-Report; 48; 21.09.-25.09.92 (9239)

R.P. Daley, U. Furbach, K.P. Jantke (editors):
Analogical and Inductive Inference 1992 , Dagstuhl-Seminar-Report; 49; 05.10.-09.10.92 (9241)

E. Novak, St. Smale, J.F. Traub (editors):
Algorithms and Complexity for Continuous Problems, Dagstuhl-Seminar-Report; 50; 12.10.-16.10.92 (9242)

J. Encarnação, J. Foley (editors):
Multimedia - System Architectures and Applications, Dagstuhl-Seminar-Report; 51; 02.11.-06.11.92 (9245)

F.J. Rammig, J. Staunstrup, G. Zimmermann (editors):
Self-Timed Design, Dagstuhl-Seminar-Report; 52; 30.11.-04.12.92 (9249)

B. Courcelle, H. Ehrig, G. Rozenberg, H.J. Schneider (editors):
Graph-Transformations in Computer Science, Dagstuhl-Seminar-Report; 53; 04.01.-08.01.93 (9301)

A. Arnold, L. Priese, R. Vollmar (editors):
Automata Theory: Distributed Models, Dagstuhl-Seminar-Report; 54; 11.01.-15.01.93 (9302)

W. Cellary, K. Vidyasankar , G. Vossen (editors):
Versioning in Database Management Systems, Dagstuhl-Seminar-Report; 55; 01.02.-05.02.93 (9305)

B. Becker, R. Bryant, Ch. Meinel (editors):
Computer Aided Design and Test , Dagstuhl-Seminar-Report; 56; 15.02.-19.02.93 (9307)

M. Pinkal, R. Scha, L. Schubert (editors):
Semantic Formalisms in Natural Language Processing, Dagstuhl-Seminar-Report; 57; 23.02.-26.02.93 (9308)

W. Bibel, K. Furukawa, M. Stickel (editors):
Deduction , Dagstuhl-Seminar-Report; 58; 08.03.-12.03.93 (9310)

H. Alt, B. Chazelle, E. Welzl (editors):
Computational Geometry, Dagstuhl-Seminar-Report; 59; 22.03.-26.03.93 (9312)

H. Kamp, J. Pustejovsky (editors):
Universals in the Lexicon: At the Intersection of Lexical Semantic Theories, Dagstuhl-Seminar-Report; 60; 29.03.-02.04.93 (9313)

W. Straßer, F. Wahl (editors):
Graphics & Robotics, Dagstuhl-Seminar-Report; 61; 19.04.-22.04.93 (9316)

C. Beeri, A. Heuer, G. Saake, S.D. Urban (editors):
Formal Aspects of Object Base Dynamics , Dagstuhl-Seminar-Report; 62; 26.04.-30.04.93 (9317)

R. Book, E.P.D. Pednault, D. Wotschke (editors):
Descriptional Complexity: A Multidisciplinary Perspective , Dagstuhl-Seminar-Report; 63; 03.05.-07.05.93 (9318)

H.-D. Ehrig, F. von Henke, J. Meseguer, M. Wirsing (editors):
Specification and Semantics, Dagstuhl-Seminar-Report; 64; 24.05.-28.05.93 (9321)

M. Droste, Y. Gurevich (editors):
Semantics of Programming Languages and Algebra, Dagstuhl-Seminar-Report; 65; 07.06.-11.06.93 (9323)

Ch. Lengauer, P. Quinton, Y. Robert, L. Thiele (editors):
Parallelization Techniques for Uniform Algorithms, Dagstuhl-Seminar-Report; 66; 21.06.-25.06.93 (9325)

G. Farin, H. Hagen, H. Noltemeier (editors):
Geometric Modelling, Dagstuhl-Seminar-Report; 67; 28.06.-02.07.93 (9326)