PLC-PROGRAMMING BY DEMONSTRATION USING GRASPABLE MODELS

Kai Schäfer, Willi Bruns

University of Bremen
Research Center Work – Environment – Technology (artec)
Enrique Schmidt Str. 7 (SFG)
D-28359 Bremen
Fon: +49 (0)421 218 4206
Fax: +49 (0)421 218 4449
e-mail: schaefer@artec.uni-bremen.de
www.artec.uni-bremen.de/field1/rugams/index.html

Abstract: PLC-programming for production plants is time and money consuming. It takes place in the critical period shortly before start of production which is a very time critical period. It should be started as early as possible, using concurrent engineering, interdisciplinary cooperation, efficient programming tools and simulation. Programming by demonstration using graspable models brings these techniques together and represents a crucial new approach. Graspable models of solid bricks are well suited for factory planning. Using these graspable models also for programming may improve cooperation and understanding of interdisciplinary experts. The Research Center *artec* features a new concept called *Real Reality* to synchronize real and virtual worlds dynamically. A magnetic tracking system serves as input device together with a data glove or a sensor ring. These devices provide input data that allow to keep track of the changes of the real world model. An automatic system for input abstraction generates PLC programs automatically from demonstrated behavior. The benefits of this concept are shorter time to production, lower costs, better control program validity and better cooperation in interdisciplinary teams.

Keywords: Automation, Gesture recognition, Interactive approaches, Interdisciplinary design, Interfaces, Programmable logic controllers, Programming, Programming by demonstration, User interfaces, Virtual reality

1. INTRODUCTION

Several projects with industrial partners as well as our own practice in the design of simulation models indicated that physical models play an important role for cognition and communication. They are used as prototypes for new products, design studies, and for the illustration of complex tasks and processes by making use of their medial qualities (Brauer 96). Especially in heterogeneously qualified teams physical models allow to work in a problem oriented way, without the need to concentrate on user interfaces and software functionality as it is the case with purely computer based tools (Fischer 00). Nevertheless the advantages of abstract digital systems are their capabilities of quantitative analysis, modification and automatic variation of symbolically represented virtual models and most important, they are necessary to control real processes by computers. Therefore, the idea arose to combine both previously separated model worlds the physical and the virtual one, thus preserving all their advantages.

In 1993 the idea of synchronizing graspable real models and virtual worlds with a sensorized was published for the first time by Bruns (93). Many ideas, applications and prototypes based on this concept have been investigated and implemented since then. The name *Real Reality* has been introduced as a term for this concept. Main research topics have been

1. Real world interactions and cooperation
2. Sensor equipment and configuration
3. Maintenance of dynamic - maybe distributed - models by automatic abstraction

This article introduces a factory layout planning scenario with simulation support at first. Then the technology of sensor data interpretation and automatic abstraction is dis-

cussed. Finally the generation of PLC programs out of the virtual model base is introduced.
The idea of coupling simulation modeling with PLC-programming is motivated by the discovery of redundant work. Already for simulation the plant control must be programmed in a simulator specific manner. Especially for complex models this is a big afford. After debugging the control algorithms and experiences are documented. After this documentation the programs are written again for the PLC controls - conventionally. We introduce a system to gather these tasks with a physical model. Planning, simulation and plant programming can be performed on the planning desk, which avoids redundant and expensive work.

1.1. Application in Factory Layout

In our project RUGAMS (part of the DFG Research Programme 'Modeling of Production') we applied the *Real Reality* Technology to design factory layouts and to programme plants with conveyor systems and robots (Fig. 1.).

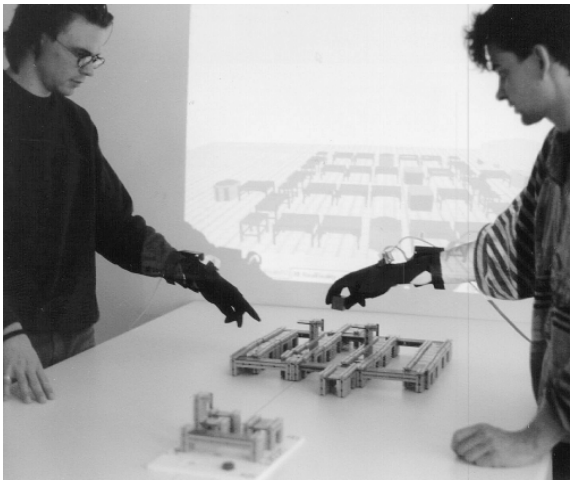

Fig. 1. Co-operative Modeling and Programming of an Industrial Plant

We configured the modeling environment for interdisciplinary teams to plan layout and material flow in complex factory scenarios. Therefore, we constructed an appropriate set of factory models for conveyors, tool machines, buffers and junctions presented in an object box (foreground of Fig. 1.). Corresponding virtual objects have been defined to represent geometry and simulative behavior of the machinery elements. The group plans and discusses the layout in the modeling session. The computer in the background keeps track of the modeling process, by generating a virtual model of the plant. This enables a simulation applying the behavior stored in the virtual components. A video beamer projects the activity of this digital factory model in the background to provide a feedback to the planning group. In later versions the simulated dynamics are projected directly onto the objects in the scene using *Augmented Reality* Technologies (Fig. 2.). This helps to provide a better context to the model.

We will describe later how a predefined behavior of the objects can be influenced by using programming by demonstration techniques. These demonstration techniques allow the specification of the material flow which optimizes the system performance and generates PLC programs. The environment may support a substantial part of the planning tasks within a group of decision-makers in just one session. This reduces costs and the time to market significantly.

2. TECHNOLOGY

2.1. Model Synchronization

Grasp Recognition; In 1993 we laid the foundation for a new class of user interfaces in shop floor and handicraft working (Bruns 1993). The main characteristic of the "*Real Reality* User Interface", as we called it, is the application of the user's hand as a manipulator of physical objects in a real environment. Appropriate interface devices like data gloves and tracking systems capture the user's hand movements and finger flexions. Gesture recognition algorithms analyze the raw interface data and recognize gestures, grasps or user commands in real time. Working with physical objects while being linked to a computer has a certain analogy to the well-known Drag & Drop principle of GUIs. When the object is grasped all following data of the Drag-Phase is recorded. This phase terminates when the user puts the object to another location and releases it (Drop). Now, the physical object has a new position and due to this the virtual computer model of the physical environment has been updated.

The system will trigger a grasp event, if a grasp gesture together with a collision between index fingertip and the boundary box of a virtual object is detected. Stopping a grasp gesture triggers a release event. Giving the user an acoustic feedback in the moment of grasping and releasing, the graphic output on a display becomes obsolete. So the user can work independently of the encumbering aura of the monitor, the keyboard and the mouse.

The term *Real Reality* emphasizes the difference to the term *Virtual Reality*. In a Virtual Reality environment the user immerses and is surrounded by the interface. *Real Reality* means to remain in the real world and to experience it. All human senses are stimulated and communication within groups is encouraged. The interface becomes a passive observer and is ideally not noticed by its users. We achieve this by linking physical objects to their virtual counterparts. As our physical objects always have a virtual counterpart they are called „*Twin Objects*". In this way the description of actions effecting two model representations becomes much easier.

2.2. Preparing Real Reality Modeling Sessions

The *Twin Objects* are one of the basic elements of the *Real Reality* concept. For both kinds of object representations a number of instances must be available. This means to cre-

ate a virtual representation consisting of the object's geometry and attributes describing the dynamic behavior. The geometric description contains the object's size

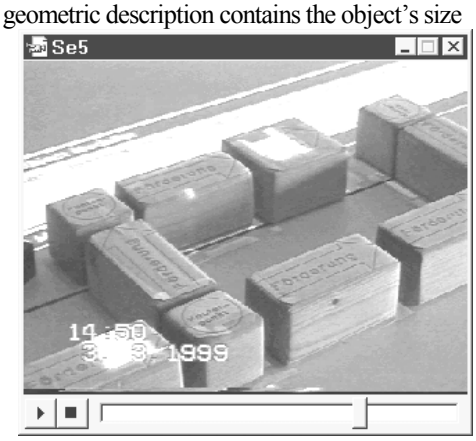

Fig. 2. Augmentation of a concrete Model to visualize the Dynamics of the Simulation

(length, width, height) and its surface shape. On the other hand, the physical objects may be constructed by using technical construction kits, wood or other materials. The object representations may vary in shape, size and level of detail. In the initial state, the objects are located in an object box, which has a predefined position on the tabletop (Fig. 3.). Thus, for each object in the box the position can be computed.

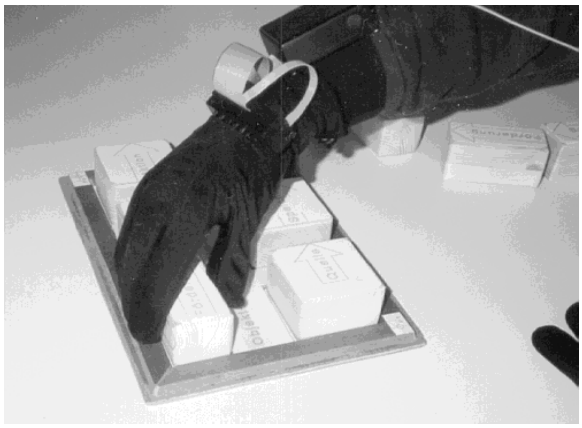

Fig. 3. Object Box

Using the Real Reality Modeling System; After the preparation of the modeling elements and the data glove, the *Real Reality* software is initialized and ready for use. A model is created gradually by taking *Twin Objects* out of the object box and putting them on the model ground. As two models are handled synchronously, the *Real Reality* system provides two views on the same model. With the help of 3d visualization software, the virtual representation is displayed and animated on a monitor screen, projected directly onto the model table or observed remotely via a network. Although the visual feedback is not necessary for persons, who work with the physical objects, it is used for replaying actions recorded during a modeling session.

Fig. 4. shows the virtual representation of a hand reaching for a *Twin Object* contained in the object box, in this case a conveyor. It is taken out of the box and placed in a location near another conveyor, which was inserted into the model during a preceding step (Fig. 5.).
Some other forms of interaction are provided. By pointing at a specific object the user gets access to information about it, while creating a model. The information is shown on the display. For the future, voice in- and output for the computer are planned.

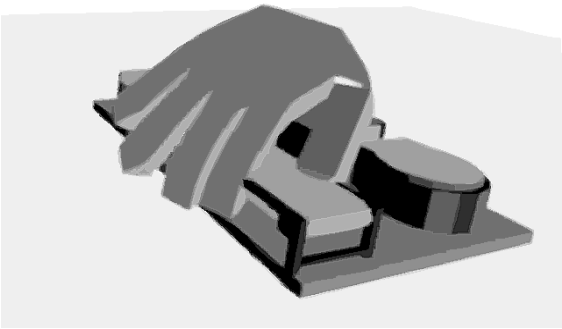

Fig. 4. The Virtual Hand grasping a Twin Object

The virtual model is represented on a scene graph in VRML2 Format. This is a dynamic representation of the real scene. Real-time analysis (a step of abstraction) permits the filtering of relevant events like collision, connection and separation, pile up elements and characteristic object motions. Saving the scene graph in a VRML2 file permits subsequently analysis, and serves as a documentation of the modeling process that can be recalled and graphically animated later.

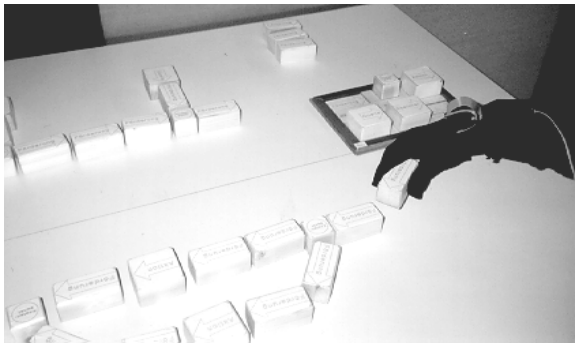

Fig. 5. Building a Model with Twin Objects

Special Hardware DevicesM; We use some other input devices, which may need additional explanations:
The *Grasp Tracker* is a new device that can replace the data glove. It is a ring, worn at the index finger tip (Fig. 6.). We have observed that in most of the natural grasps for model manipulation the index finger is in contact with the object. We have decided to mount the sensor to this finger and achieve well satisfying results. The tracking system is mounted on top of the ring, so its relative position to the grasped object is fixed. The underside of the ring is covered with a pressure-sensitive foil for triggering grasp events. Grasp and pointing gestures can be distinguished by inclination. If more different gestures are desired, the *Grasp Tracker* can be combined with a data glove. Several persons can wear a *Magic Ring* and modify the model si-

multaneously. Parallel plant activities may also to be demonstrated cooperatively.

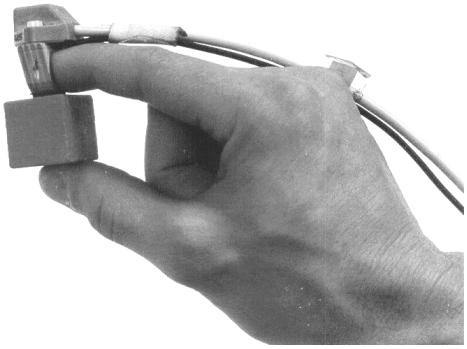

Fig. 6. The "Magic Ring" Grasp Sensor

3. PROGRAMMING BY DEMONSTRATION

One of the major aspects of simulation models is dynamic behavior. When using the grasp detection technology it is possible to recognize dynamics of the users performed with model objects. Interpreting this demonstration as programming-language enables us to program system behavior for simulations and plant control. To derive programs from what the user has previously demonstrated is an old issue in human-computer interaction. The approaches however have been focused on the conventional WIMP (Windows Icons, Menus and Pointers) interface styles (Cypher 94). The 3d interface provided by the *Real Reality* concept offers new opportunities for specifying dynamic model behavior by demonstration. The investigation of this potential is one of our main goals of research. In this section we discuss an approach to this issue.

3.1. Programming Material Flow in Conveyor Systems by Demonstration

One interesting application application of the *Real Reality* concept is for event based simulation systems for material flow control and plant layout. A typical scenario in this area of application is a conveyor system supplying several machines in a factory with raw materials or partially finished parts for further processing. It is the task of a control system to ensure an optimal utilization of the production capacities. Especially the order of machine utilization is an important factor for the productivity of such systems. In practice, this logistical problem is hard to solve, even for smaller plants. Therefore, simulation technology, particularly event-based simulators are a widely used tool.

In order to support this kind of simulation task a construction kit consisting of conveyors, machines and work pieces has been built (see Fig. 1. and Fig. 5. for different levels of abstraction. In Fig. 5. Rectangular solids represent conveyors, the quadric solids are the machines, where specific operations are performed. The arrow on each element indicates the direction of the material flow. If a single element has two exits, which means that there are two possible directions available to continue material flow, this constellation will be called a branch. On the other hand, if a single element contains two inputs, the material may flow together again. Such elements may cause conflicts if material is delivered from more than one direction at the same time.

At a branch a decision is necessary to determine which direction has to be followed. In our examples, blank work pieces are delivered and put in a circuitry where they are buffered and moved around, until a series of machine operations is performed on them. Afterwards the work pieces leave the circuitry via a branch. This simple system allows the investigation of important aspects regarding the flow of work pieces through the plant. The main issue discussed here is the question how to derive rules for branching decisions from the input observed with the *Real Reality* modeling system. Furthermore, these rules must be represented in a formal notation for the utilization in subsequent simulation experiments as well as for the transformation into PLC control programs.

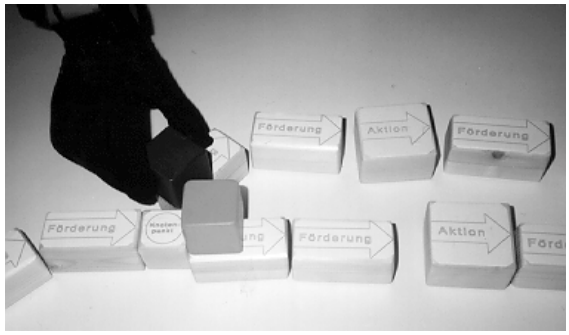

Fig. 7. Demonstrating a Branching Decision depending on Object Properties

Already our first program version presented at the Hanover-Fair ´96 was able to generate rules depending on work piece attributes coded by color. Of course a more sophisticated control mechanism is needed for real industry problems. Fig. 7. shows a situation in which one specific rule has been demonstrated: „move all light containers straight and branch dark ones out". This rule is extracted, transferred to the simulator and the participants can evaluate their program behavior immediately in this system configuration.

In a different situation, the user may desire a decision depending on the current state of the plant. Each resource (machine or conveyor) of the plant is either free or occupied. These two states determine whether a resource is available for processing or not. Placing work pieces, represented by colored plates, on the resources indicates this state. In a branching decision just a few resources are relevant. The context of a decision rule must be specified by placing tags on these resources (Fig. 8.). This situation shows that the state of the two machines determines the decision of branching, which is indicated by their tags (see the small discs). One of the machines is occupied whereas the other one is free for processing. The user moves the light-colored work piece towards this machine. From this

action the following rule can be derived: *„if machine A is occupied and machine B is free then route pallets to B“*. From now on, the simulator will apply this rule each time the specified condition occurs.

These activities of demonstration can be processed as a programming language. This allows the recognition of user intentions and the generation of formal specifications serving as programs for simulation or plant control. The major advantages compared to textual as well as to mainly graphical programming languages are:

- It's easy to learn
- Context of the program and the plant model is kept permanently

Fig. 8. A Branching Decision depending on the Plant's State

- Immediate feedback through simulation is possible
- Simultaneous and cooperative work of participants is being featured

Machine understanding of demonstrated rules is the topic of the following passage.

3.2. A Stage Model of Demonstration Analysis

A system consisting of seven stages has been developed to model the process of plant layouts and material flow from input to PLC program output. From stage to stage an abstraction of the raw input data to formal programs takes place.

Fig. 9. illustrates the stages. Above the square process stages the characteristic input information provided by the system is shown. This information partially depends on the application context and is therefore named context knowledge. Below the process stages the feedback provided to the users is represented. This feedback either supports the user interactions at the model directly or offers relevant information
about the actual model. This computer generated information helps to refine the model and optimize the planning results.

3.3. Program simulation, Extraction and Export

As shown before, the abstraction model in 7 stages can filter logical rules out of gestural user inputs. These rules are applied in stage 6 for simulation and in step 7 for PLC program export. Simulation supports iterative program specification. Each material flow component comes with a predefined programmed behavior. Without any demonstration this can be simulated. In all situations where the predefined behavior is unsatisfying it can be changed by particular demonstration. The user may watch the consequences in simulation runs.

The model contains the control logic of the elements in an object-orientated structure as Petri-Net representation. Each class has a default behavior. Each instance allocates free I/O resources of a PLC in case of creation. A PLC object has to be part of the simulation model. The Petri-Net of the instances controls behavior in the professional logistics simulator named SIMPLE++ (E-MPlant) in our implementation. The simulator provides also capabilities to modify the controlling Petri-Nets graphically where it goes beyond the capabilities of programming by demonstration. After successful simulation Petri-Nets can be exported as "Anweisungsliste" in SIEMENS S5 or S7 format. We have proven that these programs can be loaded into PLC-programming environments, can be translated, copied to the PLC and control a plant without further modifications. We can demonstrate the process chain between programming by demonstration up to running PLC-controlled manufacturing systems with our prototype at artec.

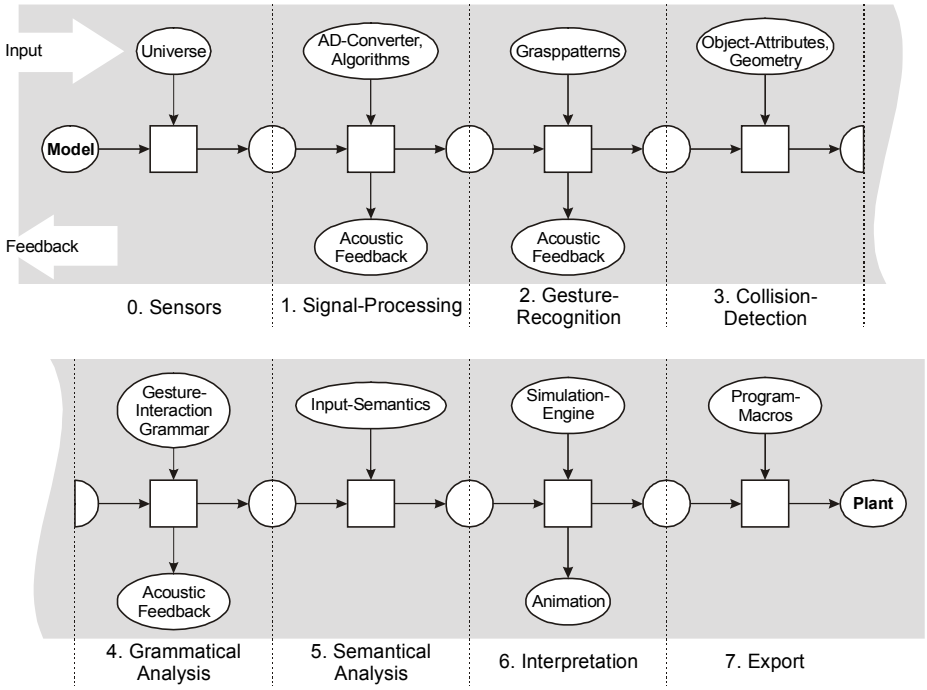

Fig. 9. Processing User Inputs on a concrete Model in 7 Stages

4. CONCLUSIONS

Graspable models have proven to play an important role in cooperative factory layout planning (Ireson 52; Scheel 94). artec's *Real Reality* concept couples graspable models dynamically with a virtual model. We have shown that graspable models together with programming by demonstration also may result in better control programs.

Implementation of several prototypes have demonstrated that the Real Reality principle of updating a virtual model dynamically with data from a glove or "Grasp Tracker" is a complex task to implement and several problems have to be solved. We have started and have implemented several functional prototypes. Other techniques like image recognition are also promising for static layouts. Dynamic 3d object tracking in real-time is still to manage (Ehrmann 00). It only works under particular conditions that exclude coverage. Coverage is a normal circumstance because small objects are partially or totally hidden when they are in a grasp for manipulation. In consequence grasp tracking is the most promising technology for Programming by Demonstration.

For planning and programming tasks simulation plays an important role. Simulation allows foreseeing (with some limitations) the performance of production systems and is capable to validate PLC programs. The system introduced in this article considers this important aspect.

Costs and time for taking up operation of production plants can be lowered significantly with the *Real Reality* approach. This is to be demonstrated in pilot projects, which may serve as reference projects later.

REFERENCES

Brauer, V. (1996). Simulation Model Design in Physical Environments. ACM SIGGRAPH, Computer Graphics, Vol. 30, No. 4, November 1996, pp. 55-56.

Cypher, E. (Eds.) (1994). Watch What I Do - Programming by Demonstration. MIT Press, Cambridge, Massachusetts

Bruns, F. W. (1993). Zur Rückgewinnung von Sinnlichkeit. Technische Rundschau, Heft 29/30, Juli 1993, S. 14-18.

Ehrenmann, M.; Ambela, D.; Steinhaus, P.; Dillmann, R. (2000). A Comparison of Four Fast Vision Based Object Recognition Methods for Programming by Demonstration. Applications Proceedings of the 2000 International Conference on Robotics and Automation (ICRA); pp. 1862-1867; San Francisco, California, USA; 2000.

Fischer. G. (2000). Shared Understanding, Informed Participation, and Social Creativity - Objectives for the Next Generation of Collaborative Systems. In Proceedings of COOP'2000, Sophia Antipolis, France, May 2000.

Ireson, William Grant (1952). Factory Planning and Plant Layout. Prentice-Hall, New York.

J. Scheel, K. Hacker, K Henning (1994): Fabrikorganisation neu beGreifen. Köln , TÜV Rheinland. 155 ff.