Proceedings of the 2002 Winter Simulation Conference
E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, eds.

SIMULATING REALITY USING AUTOMOD

Matthew W. Rohrer
Ian W. M^{c}Gregor

Brooks-PRI Automation
Planning and Logistics Solutions
5245 Yeager Road
Salt Lake City, Utah 84116, U.S.A.

ABSTRACT

Decision making in industry has become more complicated in recent years. Customers are more demanding, competition is more fierce, and costs for labor and raw materials continue to rise. Managers need state-of-the-art tools to help in planning, design, and operations of their facilities. Simulation provides a virtual factory where ideas can be tested and performance improved. The AutoMod product suite from Brooks-PRI Automation has been used on thousands of projects to help engineers and managers make the best decisions possible. With the release of AutoMod 11.0 in 2002, AutoMod now supports hierarchical model construction. This new architecture allows users to reuse model objects in other models, decreasing the time required to build a model. Composite models are just one of the latest advances that make AutoMod one of the most widely used simulation software packages.

1 INTRODUCTION

AutoMod is unique in the world of simulation tools. It allows users to construct models of any size and complexity that can be used not only for planning and design but for day to day operations analysis and controls development and testing. AutoMod combines concurrent 3-D graphics, shown in figure 1, with the most comprehensive set of templates and objects for modeling many different applications. Because of AutoMod's capacity for detail, it has become the tool of choice for serious simulation practitioners who need to maximize their return on the model building investment. The architecture of AutoMod also allows for reuse of model objects, which reduces the time it takes to build a model. AutoMod also can be linked to other software through ActiveX to help create simulation tools that can be used by a wide range of individuals, from plant manager to the area supervisor. Simulation technology is being applied to more than just planning and design today, and AutoMod has met the challenge with functionality to support a wider range of applications.

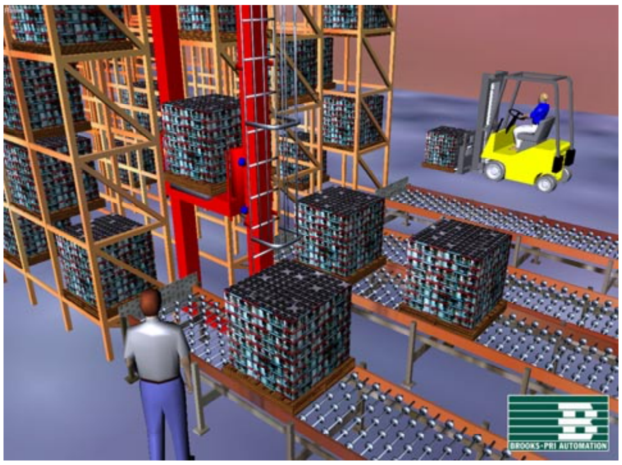

Figure 1: AutoMod's Concurrent 3-D Graphics

The AutoMod Product Suite includes:

- AutoMod – model build and simulation execution environment
- AutoStat – statistical analysis including optimization
- AutoView – dynamic walk-though of animation with AVI support
- Model Communications Module (MCM) – protocols for linking to third party software including OLE for Process Control (OPC) – for example control systems for testing.

AutoMod contains templates for material movement, which are called movement systems. AutoMod's movement systems aid users in defining the movement of material either manually or by automated equipment.

Movement systems include:

- Path mover (path/vehicle systems, such as lift trucks, AGVS, and human movers)
- Conveyors (including belt and roller types)
- Automated storage and retrieval systems (ASRS)
- Robots
- Bridge cranes
- Power and free chain conveyors
- Tanks and pipes.

To define movement systems, the user simply defines movement elements, such as paths and stations, and then fills in the operating parameters, such as velocity and acceleration. AutoMod then automatically creates the corresponding control logic for the devices. Statistical performance reports and 3-D animation are created automatically as well, providing a realistic view of how a facility will look and operate. Model animation can be viewed from any angle or perspective in real time, providing visualization capabilities unmatched in other simulation tools.

AutoMod's key strengths are:

- Flexibility to model complex systems accurately
- Unlimited model size
- High performance simulation engine
- Best in class statistical analysis features
- Graphic environment for geometry creation.
- Hierarchical model construction
- True to scale import from CAD tools.

The animation provided in AutoMod models is concurrent, meaning that the graphic pictures are running real-time with the simulation model. The model execution environment is very interactive, allowing the user to stop and start the simulation, run without animation in an accelerated time scale, and select objects from the animation screen to gather detailed statistics about the system being modeled. Statistics can be viewed at any time during a simulation run. These model execution features make it easier to verify and validate models of complex systems and provide a forum for communication about system performance among project team members.

2 AUTOMOD INTERFACE

An AutoMod model consists of one or more systems organized in one or more sub-models. A system can either be a process system, in which flow and control logic are defined, or a material movement system. Each model must contain at least one process system and may contain any number of movement systems. Since version 11.0 supports hierarchical model construction, users can integrate pretested sub-models into new models. Processes can contain complex logic to control the flow of either manufacturing materials or control messages, to contend for resources, or to wait for user-specified times. Loads can move between processes with or without using movement systems.

All inter-arrival and event times can be represented by deterministic values or be derived randomly from one of many statistical distributions. AutoMod's interface is windows-based, using pop-up and pull-down menus, dialog boxes, selection lists, and an editor for developing process logic.

3 AUTOMOD'S WORLD VIEW

Any number of movement systems can be defined in an AutoMod model, and process systems connect the movement systems to the logical flow of products. In a process system, loads (products, parts, etc.) move between processes (locations) and compete for resources (equipment, operators, and queues). The load is the active entity, executing action statements that are connected to the processes. Typical action statements give users the ability to:

- Use machines/operators
- Move into queues
- Clone new loads
- Change load types
- Wait on user-defined delay lists
- Increment/decrement counters
- Set variable values
- Read from data files
- Send to other processes
- Make conditional tests.

Figure 2 shows the process system pallet for AutoMod. The pallet is organized in a "top-down" manner with the most commonly defined elements at the top.

3.1 Processes

The process system is the backbone of an AutoMod model, providing the general-purpose simulation features required for modeling a wide range of real-world problems. While material movement is important, it is not always critical in manufacturing. No value is added to product while it is being moved around a manufacturing facility. Machines and people perform value-added operations. The AutoMod process system is where the value-added operations and control logic are defined.

AutoMod's process system includes a simulation language based on action statements. Action statements combine the power of a structured language with the ease of use of English-like, manufacturing-oriented syntax. Auto

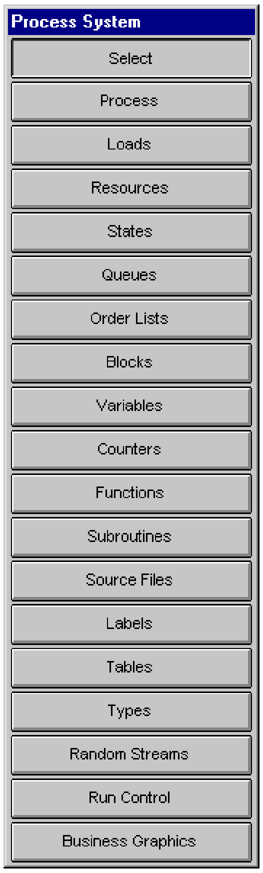

Figure 2: AutoMod's Process System Pallet

Mod models are not limited in any way, so model logic can be of any size and complexity. The power and flexibility of the AutoMod language makes it easy to model almost any real-world situation.

Processes in AutoMod are the places where actions are performed or decisions are made. For example, an inspection operation could be represented as a process in AutoMod. From the final assembly process, parts would be "sent" to the inspection process, where an inspector (resource) would look at the parts. Processes can have a physical location, but it is not a requirement.

3.2 Loads

Loads are the active entities in AutoMod and can be created in many ways, including deterministic or probabilistic generation. The inter-arrival rate for loads can be read from an external data file or attached to a statistical distribution. AutoMod has predefined random distributions that can be used to fit most real-world random events. Loads are given types, such as "RedCar" or "PartA," and they can have attributes such as color, stock keeping unit (SKU), priority, and time in the system. Attributes can be accessed and modified in AutoMod's action statements. Loads have 3-D shapes and dimensions like most other entities in AutoMod. Loads can also be changed on the fly in AutoMod through actions like:

```
scale boxA by x 4
              /*scales in x by 4 times */

set widget color to brown
```

3.3 Resources

Resources in AutoMod are used to represent machines, operators, fixtures, containers, and any other finite capacity objects. There are two default categories for a resource's state. The first is the working category (busy or idle), and the second is the availability category (up or down). During the animation, state colors indicate the status of each resource. Statistics are automatically collected for every resource in a model.

Loads use resources for specified processing times. These times can be deterministic or probabilistic, using AutoMod's built-in statistical distributions to simulate randomness. Processing times are either part specific or are applied to all parts using a particular resource. Resources can also be preempted if a higher priority load needs immediate attention.

Resources can have downtimes, which are defined by MTTF (mean time to fail) and MTTR (mean time to repair). These times can use AutoMod's statistical distributions to represent random failures. Though MTTF is calculated by model-simulated time by default, it can also be based on parts processed or machine running time.

Resource cycles can be created and attached to specific resources to indicate when or how often events such as random failures or PMs occur for a resource.

3.4 States

The user can define states other than the default states. These states might be used to represent conditions such as blocked, starved, PM, offline, etc. The state of a resource can then be changed using AutoMod actions so that statistics can be tracked for each state. In addition, state monitors can be defined and used to track states for entities other then resources, such as vehicles, conveyors, or particular areas of a facility.

3.5 Queues and Order Lists

Queues in AutoMod are both graphical and statistical elements, and they can have any user defined capacity. When a queue has reached its capacity, the next load trying to enter that queue must wait until there is space available. Queue contents can be shown dynamically in the animation, and loads can be stacked in any direction in the queue.

Loads that are at a queue or process may be sorted or delayed until they are explicitly ordered to leave. To determine which action should take place next, the loads must place themselves on order lists. An order list is not a physical entity like a queue, but a logical element that provides a way to sort loads that have been delayed for any reason. To remove a load from an order list, another load must execute an order action. Loads can be ordered to move to another process or order list, or to simply continue where they left off in their processing. Order lists can be sorted by load priority or other load attributes, either in ascending or descending order.

3.6 Blocks

Blocks control the number of entities occupying a physical space, making them very useful for controlling path mover vehicles. Blocks may have any capacity from one to infinity. Path mover vehicles (lift trucks, AGVs, electrified monorails, etc.) and loads increment blocks automatically when moving through the physical space defined by the block. Loads can also claim blocks in process logic, as directed by the user. Blocks can have any shape, including combinations of the AutoMod-supported shapes (discussed in section 5).

3.7 Variables and Counters

Data may be stored in an AutoMod model using variables. Variable values are changed using the "set" action as follows:

```
set Setup_time to 123.456
```

Variables can be used in calculations or can be compared to other variables. In addition to integer, real, and string types, variables can also be used to store references to other process system entities, such as processes, queues, resources, order lists, counters, loads, and locations. Storing these references gives the users more power and flexibility to expand and extend models to match modifications of the actual system. AutoMod also supports the notion of arrayed entities, making it easier to model real-world systems that have some dimensionality, for example a warehouse with similar operations on two floors.

Counters are similar to variables except they can have finite capacity and can only be positive integer values. Counters have a maximum capacity, making them useful for traffic control. When a load tries to increment a counter that is at its capacity, the load will be delayed until another load decrements the counter. Statistics for counters are collected automatically.

3.8 Functions, Subroutines, and Source Files

Functions and subroutines are used to create modular models. This allows the models to be extended more easily. Users can define their own functions in the AutoMod or C languages, and these functions can be called from anywhere in the AutoMod model. Subroutines help eliminate duplication of action statements in the model.

Source files contain the logic for the model. Any number of source files may be defined, and they are processed differently depending on their file extension. For example, ".m" files contain AutoMod logic and will be checked by AutoMod for correctness when edited. Files with a ".c" extension should contain C language and will be compiled with the model. Users may also define other source files to contain input data or documentation for the model.

3.9 Labels

AutoMod provides the ability to use text labels to enhance model understanding. Labels may be added to any location in the model's physical space, and they can either be static or dynamic during the simulation. Labels can rotate when the animation view changes or can be attached to a fixed position on the screen.

3.10 Tables

Tables in AutoMod supply the user with the means to collect statistics on any model entity and to classify those statistics for better understanding of their distribution. Tables automatically provide average, standard deviation, maximum, and minimum for all values entered in the table. Users can define the number of "bins" or categories and AutoMod adds values to the appropriate bin when directed by the user in the tabulate action statement.

3.11 Types and Random Streams

Types in AutoMod provide users with a powerful tool for creating, among other things, lists of entities such as resources or stations, and then making complex decisions based on the contents of the lists. Some of the default types are integer, real, resource, location, etc. But with user-defined types, it is possible to create, for example, a variable of type ResourceList, whose data can then be manipulated in any number of ways to facilitate complex decision-making.

Random streams may be defined to give each element of randomness independence from other elements. The number of different random streams used by AutoMod models is without limit. AutoMod uses the Combined Multiple Recursive Generator (L'Ecuyer and Touzin 2000).

3.12 Run Control

Run control in AutoMod allows users to define the warm-up and steady-state periods for the model by resetting time-persistent statistics. Reports can be printed for any run control period, or “snap.” Business graph output can be automatically created. Post-processed animation periods used by AutoView, an animation extension (discussed in section 4), are also defined in the run control. Run control also provides an entity tracing capability that gives the model builder an event-by-event account of the model run. This information is useful in verifying and validating a model.

3.13 Business Graphics

Graphs in AutoMod are easy to define and they update in real time with the animation. Graph types include bar charts, pie charts, and timelines. Figure 3 shows a typical timeline business graph. Any model entity can be attached to a graph, including transporter vehicle velocity, number of loads on a conveyor section, or average utilization of a machine. Graphs can be printed or plotted to a variety of supported output devices, and graph displays can be controlled using the AutoMod language.

4 AUTOVIEW

AutoMod’s post-processed animation extension can be used to view the animation records created by running a model. The AutoView product allows the user to pre-define all views and time periods in the model animation, and then play them back to generate presentation-quality animation files.

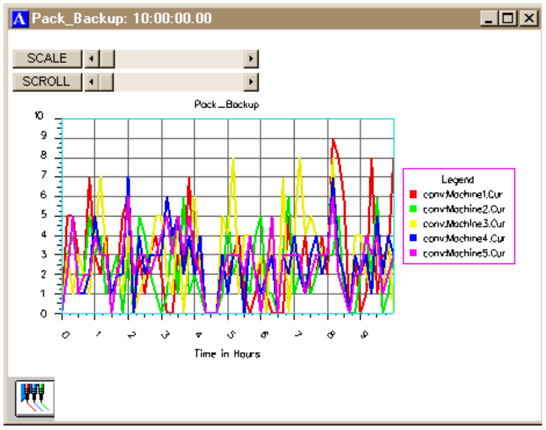

Figure 3: AutoMod Timeline Business Graph

AutoView is a free program, so users can create detailed animation walk through graphics and can distribute the AutoView files freely to their customers.

AutoView also has single-frame capture capability to AVI and MPEG format files. This feature allows users to create animations that can be shared with others without the need for any additional software. Also, the single-frame capability helps smooth the animation on larger models where hardware constraints may affect animation performance.

5 GRAPHICS IN 3-D

Both dynamic and static objects can be displayed during model execution. Figure 1 shows a screen shot of a typical AutoMod model during a model run. Dynamic objects represent loads, vehicles, resources, queues, and statistics. The static layout is the background graphics of the plant, such as columns, aisle markings, and walls. Labels can identify specific areas in the facility.

There are several ways to create a layout of the system to be modeled. AutoMod comes with a three-dimensional graphics editor that allows the user to construct objects from standard graphics primitives. Cone, Box, Hemisphere, Trapezoid, Frustum, Cylinder, Arc, Vector (list), Set, Text, and Triad are primitives that can be selected, placed, and scaled to create any static entity in the facility.

AutoMod supports the true-to-scale import of CAD format files under the IGES standard, and Virtual Reality Modeling Language (VRML) and Open Inventor graphic format files can be imported directly.

Most CAD systems and other solid geometry creation programs support VRML, and there are many translators available that can convert from other file formats, like IGES, STEP, and DXF, to VRML. VRML format files can be included as either static or dynamic shapes in the AutoMod model.

Also, since the release of version 11.0, AutoMod allows the import of Data Exchange Files (DXF) for entity graphics.

6 RUN-TIME ENVIRONMENT

In keeping with AutoMod’s interactive features, the user has complete control of the model in the run-time environment. The model can be viewed with the animation on or run with the animation off. AutoMod uses concurrent animation; the simulation progresses as the animation picture is being updated. With animation off, the simulation does not render the animation, but it performs all simulation calculations. The user can suspend the simulation at any instant to review statistics through pop-up windows, to take resources down, to set break points or alarms, or to control the view of the animation without constraint.

6.1 View Control

AutoMod provides a comprehensive, flexible, and easy-to-use method of interacting with a model during model execution. If the simulation project is in the experimentation phase where only parameter changes are made and the model needs to be re-run several times, AutoMod provides the ability to run in batch mode without animation.

6.2 User Interaction

AutoMod provides advanced debugging and trace facilities. A model can be single-stepped at any time during the animation. Also, the ability to set breakpoints and alarms allows the user to suspend the simulation when a certain event occurs or when a specific clock time is reached.

AutoMod also provides comprehensive reports. The reports can be displayed on request at any time during the simulation. Printed versions of the reports can also be specified in the Model Development Environment. Reports are configurable by the user, who can choose between standard, full, and no report for all AutoMod entities.

AutoMod automatically keeps track of many statistics. These automatic reports are linked to specific entity types, such as:

- Movement systems
- Processes
- Queues
- Resources
- Order lists.

Vehicle states are tracked during the entire model run, and reports are generated automatically. Reports can be sorted alphabetically or numerically for easier analysis. The user can also develop and generate custom reports from within process procedures.

7 AUTOMOD'S SIMULATOR INTERFACE

In addition to AutoMod's standard interface already described, AutoMod also includes a manufacturing template called the Simulator. The Simulator can be used to model problems where:

- Machines are grouped by capability. For example, in a machine shop one might have sets of lathes, mills, and drill presses that can perform the same operation.
- Parts have complex routings, including alternate steps, setups, and batching.
- Product is manufactured to customer order.

Figure 4 shows the main menu for the Simulator.

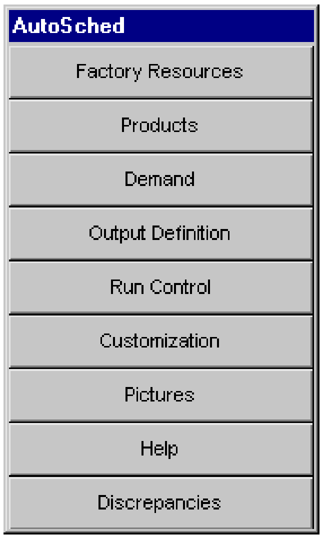

Figure 4: Simulator Main Menu

7.1 Factory Resources

Factory resources include the equipment, operators, fixtures, and any other item required for the manufacturing of products. Machines are grouped into families that can perform similar operations. When a machine finishes the current operation, it looks at all the products available to process, and makes a decision about the next thing to do. This decision is called a scheduling rule. The Simulator allows the users to select from among many standard rules, like first-in-first-out, highest priority, or same setup.

7.2 Products

Products are the items that are being produced in the Simulator model. Products have a routing that defines the operations required to manufacture them. The routing defines families of machines to visit and can also include alternated steps, setup requirements, processing time, and batching requirements. A bill of materials can also be defined for subassemblies, which can be either produced or purchased.

7.3 Demand

Demand in the Simulator represents customer orders. In manufacturing, many decisions need to be made, including how to allocate resources, what each operation should do next, and when to "launch" new orders into the facility. The demand file in the Simulator defines each customer order, start date and time, and expected due date. Reports from the Simulator indicate which orders met their due dates and which were late.

The Simulator in AutoMod provides a quick and easy way to define models of a certain class of manufacturing problem. Most Simulator models can be created without writing any code. The spreadsheet interface to the simulator is intuitive and easy to use. Data can be imported directly from other data sources, like an MRP or ERP system, to facilitate even faster model building.

8 AUTOSTAT

The AutoStat product is the statistical analysis tool in the AutoMod product family. AutoStat does extensive output analysis on an AutoMod model, including:

- Confidence intervals
- Warm-up determination
- Sensitivity analysis
- Factor-response analysis
- Design of experiments
- Optimization using evolution strategies.

AutoStat also has support for running scenarios across a network of machines. Support for multiple machines and CPUs give users the ability to make many more runs of the simulation than was possible before. Figure 5 shows an example graph from AutoStat.

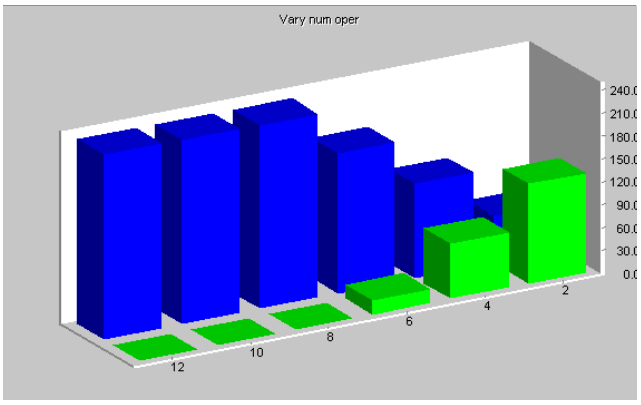

Figure 5: Example Graph from AutoStat

The evolution strategies algorithm used by AutoStat is well suited to finding the optimum solution without getting "trapped" at a local optimal value. Figure 6 shows the output from an optimization run. Across the X axis are the number of generations required to find the optimum, and the Y axis gives the values for the user-defined fitness function. The fitness function can be defined as any combination of model inputs and outputs, with user-defined weighting factors applied to each value.

9 MODEL COMMUNICATIONS MODULE

The Model Communications Module (MCM) is an extension to AutoMod that allows simulation models to

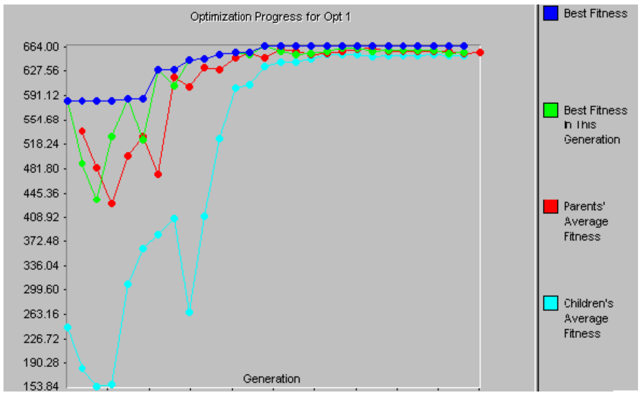

Figure 6: Optimization Run Graph

communicate with other programs locally or on a network of machines. Some of the applications of MCM include communications between:

- Two or more AutoMod models
- Control systems and AutoMod models
- Other applications and AutoMod models.

Communication between two or more models allows for parallel and distributed simulation. MCM also includes Multi-Model Sync (MMS), which keeps the event lists of the linked models synchronized.

In order to facilitate communication, MCM enables sockets, Object Linking and Embedding for Process Control (OPC) and Dynamic Data Exchange (DDE) technologies. Connecting across different platforms is achieved using sockets technology, which allows communication through strings or C structures. Models may be easily linked to other applications using DDE. Examples of this include Excel spreadsheets, Access databases, ergonomics programs, or control programs.

9.1 Emulation with OPC

When MCM was added to the AutoMod family, it quickly became apparent that AutoMod users were applying the concept to a wide range of industrial applications. Although sockets technology is robust and works efficiently across different computing platforms, there are other technologies that may provide better solutions for certain specific areas of application. One of these is OPC, or Object Linking and Embedding (OLE) for Process Control, which is a *de facto* industrial standard adhered to by all major control systems hardware manufacturers, including Rockwell Software, Siemens, Mitsubishi, GE, Schneider, Modicon, and many more. This well-defined standard makes it simple for users to link AutoMod emulation models to Programmable Logic Controllers (PLC) control systems from almost any supplier. This enables the testing of real control programs using an AutoMod model to provide the same responses as the physical system. The benefits of this are considerable:

- Controls departments no longer have to wait for the system to be installed on site before testing can begin during the commissioning period.
- Testing can be more complete and is much more convenient.
- Tests can be run in parallel.
- Operators can be trained offline on the real control system.
- Operators can see the whole system respond to their input.

The growing use of emulation is creating more opportunities for simulation departments to build models in collaboration with controls departments and enlarging the area of application for simulation technology within materials handling, production, and automation companies.

9.2 How Emulation Differs from Simulation

Emulation is a powerful and flexible way to test the operation of industrial control programs before they are installed on-site. While a simulation model is used in the design and development phase of a project to produce a better solution, the role of an emulation model is much more precisely defined as the verification of the controls system. The emulation model is used to provide the control system with the same responses it would get from the physical system. A simulation model is used to generate quantities of results from different sets of operational parameters, random number generators are employed to closely match the uncertainties of the real situation, and analysis programs are used to calculate trends and identify anomalies. Emulation models are used to step through a series of pre-defined sets of parameters that are unlikely to contain random elements; a "checklist" of situations is verified and the response of the control system is followed to detect any malfunctions.

9.3 The Advantages of Emulation

Figure 7 shows the emulation of a paint shop. This replacement of the real system by a virtual one offers several advantages:

- It can be ready for use before the real system.
- It can be duplicated at minimal cost.
- It does not disrupt existing production.
- It exists in a completely controlled environment.
- It can provide an overall view that is often impossible in the real situation.

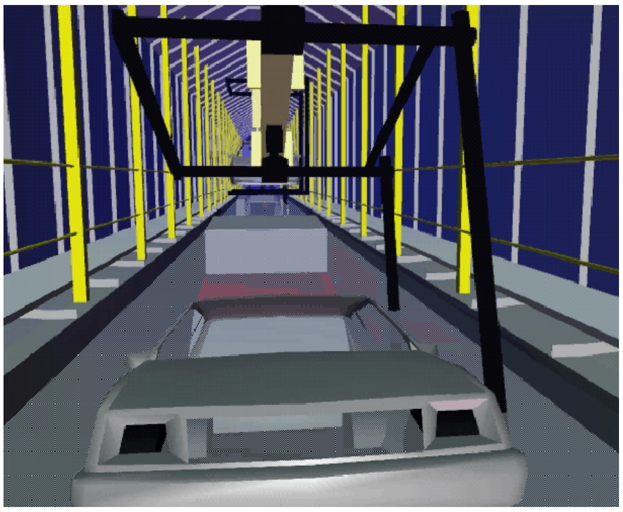
Figure 7: Emulation exhaustively tests control systems

An emulation model driven by a real control system provides a convenient and safe way to perform a series of standard operational tests that are currently carried out either on-site or at test sites.

9.4 Controlling the Commissioning Environment

As automated systems are designed to operate under loading conditions projected 5 or more years into the future, a common commissioning problem is a lack of suitable or sufficient loads with which to test the system. As a result, full testing on-site may be difficult, expensive, time-consuming, or impossible. The commissioning and start-up phases of most automation projects are tightly scheduled, a situation which is not helped by the tendency of many projects to accumulate lateness without moving the start-up date by a corresponding amount. Commissioning problems may be hard to trace and diagnose because commissioning is often the first time that the hardware system has been brought together with the control system—and a missing signal could be the result of a control system error or a misaligned detector, for example.

The most profitable area of application for emulation concerns savings to be made during the commissioning phase. The use of emulation allows standard control system verification to be done outside of the commissioning window, leaving more time available for other steps. Savings are likely as on-site costs are reduced. As verification can be carried out on several identical models simultaneously rather than on the single real system, the opportunity exists to carry out more complete system tests under a wider range of conditions than would be possible during commissioning. The result of more complete testing is obviously fewer problems during ramp-up and normal operation, which in turn leads to further savings on maintenance, staff turnover, and unforeseen interventions.

10 TEACHING AUTOMOD

Teaching AutoMod has been greatly facilitated with the publication of the text *Getting Started with AutoMod* (Banks 2000). PowerPoint slides have been prepared to accompany the text. Also, the electronic files are available for all of the example models in the text. These are all downloadable from <www.automod.com> as is the Student Version of AutoMod 9.1. Additionally, professors can receive a file with the solutions to all of the exercises in the text.

11 SUMMARY

AutoMod is a state of the art simulation system that provides the ability to define the physical elements of a system using true-to-scale CAD-like graphics and the logical portion of the system using a powerful procedural language. The results are that a typical user can be three to ten times more productive using AutoMod in comparison to using any other simulation tool. The accuracy and degree of detail with respect to model development is unequalled.

AutoMod allows the construction of very large, complex models. In fact, AutoMod provides an architecture that has proven that the larger the project, the more benefits AutoMod provides over other simulation systems.

AutoMod provides realistic, 3-D visualization. There are no limits to the views or the size of the picture to be shown. The degree of animation realism is also unmatched, as AutoMod provides animation that facilitates faster model building and better communication of model results.

The AutoMod product suite, including the AutoStat, AutoView, and MCM extensions, provides a comprehensive solution to the needs of the simulation user. Through the use of Brooks-PRI Automation's technologies, models can become more than design tools. Models can be used to test controls, operate the facility, and plan for expansion.

REFERENCES

Banks, J. 2000. *Getting Started with AutoMod.* Bountiful, UT: Brooks-PRI Automation.

L'Ecuyer, P. and R. Touzin. 2000. Fast combined multiple recursive generators with multipliers of the form a = $\underline{+}2^q$ $\underline{+}2^r$. In *Proceedings of the 2000 Winter Simulation Conference*, ed. J.A. Joines, R.R. Barton, K Kang, and P.A. Fishwick, 683-689. Piscataway, New Jersey: Institute of Electrical and Electronic Engineers.

AUTHOR BIOGRAPHIES

MATTHEW W. ROHRER is Director of Simulation Products and Services at Brooks-PRI Automation in the AutoMod group. He directs the development and support of the AutoMod product suite as well as the application of AutoMod by Brooks-PRI Automation's simulation consulting group. Mr. Rohrer is also actively involved in the simulation community. He served as business chair for the Winter Simulation Conference 1998, and General Chair for WSC 2002. His email and web address are <matt.rohrer@brooks-pri.com> and <www.automod.com>.

IAN W. MCGREGOR has worked as Simulation Business Development Manager for Brooks-PRI Automation since 1996. He has been posted in Singapore, Japan, and Utah. He has an MSc in Computer Integrated Manufacturing from Cranfield Institute of Technology in England, a Diplome d'Ingenieur from the Universite de Technologie de Compiegne in France, and a BSc in Production Engineering from Kingston Polytechnic in England. He is a Chartered Member of the Institute of Mechanical Engineers and served as Registration Chair for WSC 2001. His email is <ian.mcgregor@brooks-pri.com>.