1-1-1998

# The VR Factory : discrete event simulation implemented in a virtual environment

Jason John Kelsick
*Iowa State University*

# The VR Factory: Discrete event simulation implemented in a virtual environment

by

Jason John Kelsick

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Mechanical Engineering

Major Professor: Judy M. Vance

Iowa State University

Ames, Iowa

1998

Graduate College

Iowa State University

This is to certify that the Master's thesis of

Jason John Kelsick

has met the thesis requirements of Iowa State University

Signatures have been redacted for privacy

## TABLE OF CONTENTS

## LIST OF FIGURES

## LIST OF TABLES

## ACKNOWLEDGEMENTS

I would like to thank my major professor, Dr. Judy M. Vance, for the opportunity to participate in her research project. Her insight, direction, and encouragement have been invaluable over the course of my research.

I would also like to extend my thanks to Dr. Cheryl Moller-Wong and Dr. Greg Luecke for participating on my committee and offering their assistance and knowledge. In addition, I thank Lori Melaas, a masters student in industrial engineering, for her contributions to the research project.

I would also like to thank my fellow ICEMT workers for the helpful advice and friendship they have given me.

I also give special thanks to my friends and family who have given me support throughout my education.

Further, I would like to express my sincere thanks to my fiancée, Stephanie, for her unending patience, support and encouragement throughout my graduate work.

# CHAPTER 1. INTRODUCTION

Virtual reality (VR) refers to an immersive, interactive, multi-sensory, viewer-centered, three-dimensional (3D) computer generated environment and the combination of technologies required to build such an environment [1]. It provides a human-computer interface that allows the computer-generated information to be viewed and manipulated in an intuitive manner. This intuitive manner is based on the concept of using natural head and hand movements to interact with the virtual environment instead of the medium of the monitor, keyboard, and mouse. Thus, the person perceives the virtual world in the same manner as the real world because of the way in which the information is received through a person's senses.

Most people consider virtual reality just another form of entertainment. However, virtual reality is much more than just movies and video games. Today it is used in many different areas, as well as entertainment. Such areas include product design and development, flight simulation, and training. NASA used VR to view the local environment analyzed by Pathfinder on Mars [2]. A 3D model of the area was created with a stereo camera and scanning system on Pathfinder and explored using VR software [2].

Research is being done to investigate the effectiveness of VR in the area of psychotherapy. The use of VR to help people overcome their phobias is a developing area in psychotherapy. Max North, Sarah North, and Joseph Coble did a study to determine the effectiveness of VR in overcoming agoraphobia (the fear of being in places or situations from which escape might be difficult or embarrassing) [3]. People were placed in virtual environments such as a balcony scene, a dark barn, and an open elevator without walls. The results of the study revealed that VR could decrease the anxiety people encounter when faced with their phobias.

The VR Solutions company is an example of a company developing a VR training tool. Children are placed in a virtual home where they are trained to secure a house "by closing windows, doors, fixing smoke alarms, obscuring hi-fi equipment, locking up, removing ladders from walls, and changing the size and location of burglar-friendly trees and hedges" [4]. Another example of training in the virtual environment is of the research done by James Bliss, Philip Tidwell, and Michael Guest. The purpose of the research was to determine whether VR could train firefighters to navigate through burning buildings [5]. This research showed VR as an alternative way of preparing firefighters for potentially dangerous situations.

In terms of engineering, VR represents an additional way to design, develop, and visualize new technology. Industries are looking to use the visualization aspects of VR to help reduce the time and cost involved in the creation of a product. Virtual prototypes of products can be designed or constructed in the virtual environment, improved upon, and analyzed for problems such as assembly, manufacturability, and also ergonomic factors. Products have been designed using computer technology for many years. These designs exist as computer data files. VR allows advanced analysis such as assembly to be performed on these computer data files. This eliminates the more traditional methods of building physical prototypes, tooling, etc., which can be very costly.

VR is also becoming a part of the manufacturing/planning process for a product. Manufacturing plan flaws could be discovered through the creation of a virtual factory facility instead of finding the flaw after the actual facility has been built. One such flaw could be creating a layout of the equipment that is not ergonomically correct. The virtual factory facility proves promising because changes can easily be made to computer models but can be costly when made to physical prototypes.

VR can also be used as an effective communication tool. The development of a

product includes not only engineering but also marketing and accounting. Communication is required between these departments in order to insure a profitable product. VR can be a useful part of this communication. In both virtual prototyping and VR manufacturing, VR can be used to clarify not only virtual objects' shapes but also the functionality of virtual objects to non-technical members involved in the engineering process. The visualization and intuitive interaction capabilities of a virtual environment make possible this clarification. For example, VR can demonstrate a new product's capabilities or the benefits of a new manufacturing method to non-technical members of the development team that make the decisions on budgeting. The development of a new manufacturing work cell is a case where the use of VR can be very cost effective because of the large amount of capital required. VR allows a person to intuitively analyze and improve upon a computer-generated model of manufacturing work cell instead of analyzing and improving upon a multi-million dollar manufacturing work cell.

This thesis describes the process of constructing a virtual manufacturing work cell and implementing the results of a discrete event simulation of the work cell in the virtual environment. This implementation provides a visualization tool for viewing and analyzing the working processes and problems of the actual manufacturing work cell. The virtual manufacturing work cell was designed with the capability to investigate how various changes to the manufacturing cell part mixture affect part production. In order to show the effectiveness of the virtual environment in identifying the working processes and problems, navigation through the facility and interaction with individual parts was incorporated into the virtual manufacturing work cell.

Chapter 2 of this thesis presents a discussion of current simulation software packages and a description of the components of virtual environments. Chapter 3 discusses the operation of the virtual manufacturing work cell and its hardware and software components.

Chapter 4 describes the construction of the virtual environment and the issues faced by the developer. In Chapter 5, the creation and implementation of the discrete event simulation into the virtual environment is explained. The conclusions of this research are included in Chapter 6 along with suggestions for continuing this research.

# CHAPTER 2. DEVELOPMENT OF COMPUTER SIMULATION

Defined by Pritsker, "in its broadest sense, computer simulation is the process of designing a mathematical-logical model of a real system and experimenting with this model on a computer" [6]. This model is used to represent the behavior of the real system that may or may not already exist. The purpose of the simulation is to analyze and understand the system's behavior under various situations.

There are two general categories of simulations: continuous and discrete. In a continuous simulation, the dependent variables can change at any point in time. An example of where this type of simulation is aircraft simulation. Because variables like wind speed and direction are continuously changing, the controls on the aircraft must be able handle this continuous change. In discrete event simulation, the dependant variables must change at distinct times, thus forming events. In other words, the state of the simulated system can only change at event times. This method of simulation is more commonly used in the simulation of manufacturing systems and computer networks. In use with manufacturing, discrete event simulations model part flow through a manufacturing process. The part flow is divided into a series of events with event times. The simulation can determine bottlenecks, machine tool usage, and material handling problems, etc. before they occur.

## 2.1 History of Simulation

Computer simulation was realized as a potentially useful tool for industry in the late 1950's and early 1960's. It allowed industries to test configurations of manufacturing

systems before purchasing and implementing the actual equipment. Since then, many simulation-programming languages (SPLs) have been created and improved. Two SPLs, GPSS (General Purpose Simulation System) and GASP (General Activity Simulation Program), were developed during the late 1950's and early 1960's and have been the basis for most SPLs used today. Philip J. Kiviat developed GASP in 1961 at the Applied Research Laboratory [7]. GASP had two descendents that have played a major role in simulations used today. They are SLAM II (Simulation Language for Alternative Modeling), produced by Pritsker and Associates, Inc. [6], and SIMAN (SIMulation ANalysis) developed by C. Dennis Pegden. Both SPLs were developed in the late 1980's and have become major components of analysis and research for many industries.

Original SPLs had no way to show a visual representation of the simulations they created. This was due to the limitations of computer technology during the late 1950's and early 1960's. The computer simulations were analyzed through output not related to the layout or geometry of the facility being simulated. This output was in the form of tables, charts, and graphs. Because the results were produced in this manner, people familiar with how the simulation was created and the simulation software were the only ones able to analyze these results. In other words, only simulation specialists could accurately interpret the simulation results. This posed a drawback for incorporating the creation of a simulation as a part of the design and development of a system. People who were not as familiar with the simulation software but who were a major part of the development of a system could not utilize the capabilities of the computer simulation very readily.

The past few years have seen computer simulation development take a new direction. Previous versions of simulation software produced only text-based output because of the limitations of computers, but today, visualization of the simulation is now possible because of the increased graphics capabilities of computers. A graphical representation of the

simulation provides a more efficient way of communicating between the developers of the simulation and the rest of the members involved in the creation or development of a system. This is because "visual model descriptions use the brain's visual information processing capabilities, which are older and more highly developed than our verbal information processing capabilities, to quickly describe the model" [10]. A graphical representation of the simulation helps confirm the validity of the results of the simulation. Take for example the simulation of a manufacturing process. Things such as bottlenecks, a place in the simulation where there is a build up of entities, and deadlocks, where two or more entities are occupying the same resource, are easily seen through a graphical representation.

Because of the benefits of a visual representation of a simulation, Pritsker and Associates, Inc. have created AweSim, an interface tool that creates a graphical user interface for SLAM II, previously mentioned, and also allows for the integration of external programs and databases. Included in this version is the capability to create two-dimensional graphical animations. Although it is limited to a 2D display, this graphical animation helps the user visualize the simulation. Importing a CAD file converted into a picture file, known as a bitmap file, creates the 2D background display for the animation. The entities that move throughout the simulation are represented as icons that move according to defined simulation events and paths in the animation (see Figure 1). Another simulation software designed for manufacturing, ProModel, also includes a 2D graphic animation [8]. The 2D display is created in the same manner as AweSim with a CAD file portraying the layout and icons representing changing entities.

While simulation software like AweSim and ProModel offer a visual representation of the simulation, as mentioned before, they are limited to two dimensions. AutoSimulations, Incorporated has developed a simulator also designed for manufacturing, AutoMod, that has the capabilities for true 3D graphics [8]. The graphics are created in a similar manner in

a CAD-like utility is used. This capability for 3D graphics extends the visualization aspect of the simulation. Taylor II, developed by F&H Simulations, provides a 3D environment like AutoMod, but the creation of the geometric models is performed before the simulation is modeled.

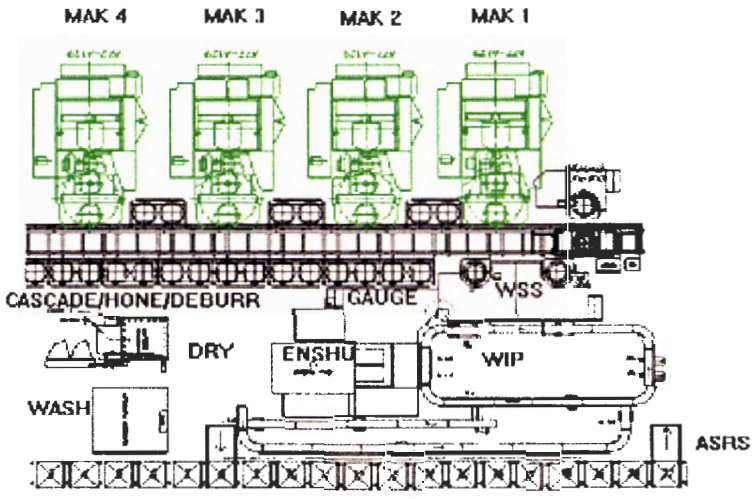



**Figure 1 - 2D Animation of a Manufacturing Simulation**

The previously mentioned simulation software packages have been created and developed first as simulations and then the graphical aspects have been added. This process of development is not the only way in which simulators with graphical aspects have originated. Three-dimensional manufacturing simulation companies have seen the benefits of event simulation and have started to include this capability in their software. RobCAD, of Tecnomatix Technologies Ltd., originated as a robotic programming tool and has been improved to include discrete event manufacturing simulation with 3D graphic capabilities [9].

Another computer simulation package that has similar origins to RobCAD is called QUEST® (QUeuing Event Simulation Tool) [11]. QUEST®, developed by Deneb Robotics, is a simulation software package with its own simulation language, Simulation Control

Language (SCL). A simulation is created in a similar way to AutoMod in that the geometric models are made first followed by the animation of the mathematical-logical model. QUEST® includes 3D graphical animation that allows for a more immersive environment than the two-dimensional animation provided by simulators like AweSim.

Deneb has also created ENVISION® to provide a virtual environment for creating geometric models used in QUEST®. The mathematical-logical model of QUEST® is still used to create the simulation components. These models can then be used in other software packages Deneb has created for things such as virtual prototyping, human factors engineering, and simulation based design. Through ENVISION®, QUEST® can transform its simulation into a virtual environment. This technology is still relatively new and because of this, the capabilities of virtual simulation through Deneb's software are still limited.

There are evident benefits of incorporating VR into simulation software. VR provides an immersive, more realistic environment that requires less intuitive visualization ability to see this environment. However, this technology has not been designed for only one application. Its potential to become a part of many applications has enabled VR to be a separate entity from a particular application. Thus, how to merge the VR technology with the simulation technology to obtain the most improvement in simulation is foremost in the development of immersive visualization of simulation.

## 2.2 Necessary Components of a Virtual Reality Environment

To create a virtual environment, certain hardware and software elements must be present. Referring back to the definition, Virtual reality (VR) refers to an immersive, interactive, multi-sensory, viewer-centered, three-dimensional (3D) computer generated environment and the combination of technologies required to build such an environment [1].

The combination of technologies is made up of these hardware and software elements. To create an immersive environment, there needs to be a display device that creates the illusion of three-dimensional space. Interaction with the virtual environment needs to be similar to the way in which a person would interact with the real environment. This could be through manual and/or auditory signals given by the user. In order to have a viewer-centered environment, the 3D stereo display is dependent on the position of the user. With these elements properly implemented, a VR environment can be achieved.

There are different ways in which a 3D stereo display can be shown. Some examples of this are: a monitor, a head-mounted display (HMD), a mechanically-mounted display known as a BOOM, a stereoscopic projection screen, or a cave-like setup like Iowa State University's C2. The mounted displays have two different displays, one for each eye. To produce the stereo image, the display for each eye is dependent on the viewpoint of that particular eye. The human mind sees the two different images as one and when merged together they form the illusion of 3D.

Showing two displays corresponding to the viewpoint of each eye, also forms the stereo image for the projection screen examples. However, if two viewpoints were projected on a single projection screen, the result would be a blurred view. To overcome this problem, a person wears glasses that allow only one viewpoint to be seen through each eye. The glasses could accomplish this by colored lenses, polarized lenses, or through shutter glasses. Shutter glasses alternate visibility in each lens faster than the human eye can detect while the computer alternates the viewpoint display shown on the screen in sync with the glasses. CrystalEyes stereo shutter glasses are a commercial product designed to produce a stereo image in the alternating display example. A transmitter located on the projection screen sends or monitor out a signal to signify whether the left eye viewpoint or right eye viewpoint image is being displayed. A sensor on the shutter glasses enables the

glasses to respond to the signal and allow visibility through the corresponding eye lens.

Iowa State University's C2 is an immersive, stereo projection room complete with 3D interaction capabilities. It is comprised of three stereoscopic walls and floor, and a three-dimensional sound system (see Figure 2). The floor is 12x12 feet and the walls are 9 feet high. CrystalEyes stereo shutter glasses enable the participants to view the 3D display on each wall and speakers are placed in each corner to create surround sound.

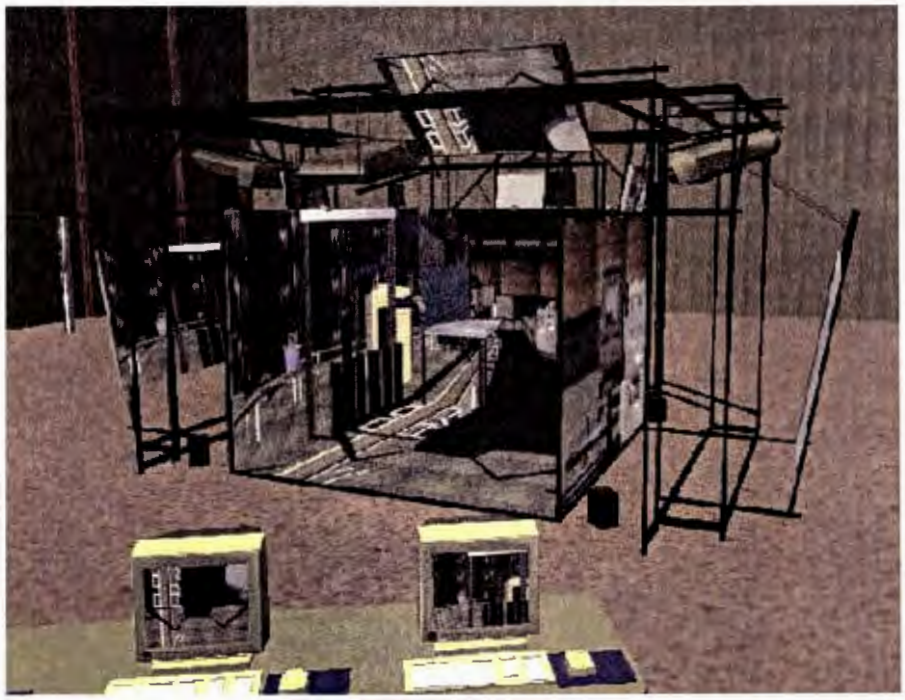


**Figure 2 - Iowa State University's C2**

The HMD provides a totally isolated virtual view (see Figure 3). The user can only see the computer-generated images. This results in a very immersive environment that does not allow the user to see the real world and lose the illusion of being in the virtual environment. Although the HMD provides the most immersive environment, there are some limitations of using the HMD over the projection screen or C2. One such limitation is that only one person can use the HMD at a time. The C2 environment and projection screen

facilitates multiple users interacting with the same virtual environment since several people can be in the C2 at the same time. Another limitation of using the HMD over the C2 is the limited field of view. Peripheral vision in the HMD is limited because the field of view ends at the outer edge of the lens for each eye. Because of the reduced field of view, the realism of the experience is somewhat affected. The C2 environment overcomes the problem of peripheral vision because images are being displayed around the user independently of where the user is looking. Only the frames of the CrystalEyes glasses obstruct the field of view.

**Figure 3 - n-Vision's Head Mounted Display (HMD)**

Besides field of view, stereo vision greatly affects the user's senses of immersion in the environment. Stereo vision in the HMD is straightforward in concept. Since each eye sees a different view in real life, the display for each lens on the HMD is different and based on corresponding eye. When viewed on the projection screen or in the C2, stereo vision is obtained through CrystalEyes stereo shutter glasses. As mentioned before, the shutter glasses alternate visibility in each lens faster than the human eye can tell and the display

shown on the screen is alternated in sync with the glasses to create the illusion of three-dimensional space. Because of this, viewing the application is not limited to one person but to how many pairs of shutter glasses are available. Even though multiple users can view the stereo images, the correct perspective can only be drawn for one user; i.e. is on the user who wears the shutter glasses with the tracker attached.

Interaction with the virtual environment can be fulfilled in numerous ways. It is most commonly done with hand gestures and movements. Three methods of hand controlled interaction or input are a glove with sensor's along the fingers and thumb calibrated to fit a person's hand, a glove with electrical contact points at the fingertips and palm, and a wand held in the hand.

The Cyber-Glove, by Virtual Technologies, is an example of a calibrated glove. This device can measure the position of each finger and thumb as the user moves his/her fingers to create gestures. Because of this, it must be calibrated for each user's hand size. The Fakespace PINCH™ Glove has electrical contact patches at the fingertips and the palm as well as on the back of the hand. Contact of these patches completes a circuit and this information is given to the computer to be interpreted as gestures. It is less expensive than the Cyber-Glove and it does not have to be calibrated for each user. A wand is a joystick with buttons and is held in the hand. Pressing the buttons registers commands to the virtual environment. In addition to registering commands, the input devices also need to be equipped with a position tracker so that a person can intersect or "touch" virtual objects.

Position trackers are used to track both input devices and the user's viewpoint. The person's viewpoint needs to be tracked to enable the displays to project the correct stereo image. Ascension's Flock of Birds is a magnetic position tracking system which consists of a fixed transmitter and a small receiver that can be attached to either the hand of the user or the input device. The interference of the magnetic fields produced by the transmitter and

the receiver enable the identification of the position and orientation of the mobile receiver relative to the transmitter. The receiver can be attached to almost anything, whether it be a helmet, a pair of glasses, a glove, or a wand and multiple receivers can be used in the same VR application.

One of the most important components of VR is the software that creates the virtual environment. The VR software generally reads in a 3D-model database of the geometry and also manages the input and display of the application. To manage the input and display, the software needs provide an interface to the interaction/peripheral devices mentioned earlier. The software does not necessarily have to have the software drivers for each device built in to the software, but it must have the capability for implementation of the desired devices.

WorldToolKit® [12], created by Sense8, is an example of software designed for use in VR. WorldToolKit® is a "cross-platform software development system for building high-performance, real time, integrated 3D applications for scientific and commercial use"[12]. It allows for the creation or importation of realistic models in a scene hierarchy that enables proper animation in real time. One benefit of using WorldToolKit® is the support it has for several interaction/peripheral devices to be used in the virtual environment. Another VR software package called dVISE, created by Division, also has the required components to create a virtual environment.

This thesis work looks at the creation of a virtual factory, called the VR Factory, using the VR software WorldToolKit® on a Unix-based platform. The creation of the VR Factory also includes the implementation of a discrete event simulation using the software AweSim mentioned in section 2.1. The next chapter describes how the VR Factory presents a discrete event simulation of a manufacturing process in VR and the interaction with this virtual environment.

# CHAPTER 3. THE VR FACTORY

The VR Factory is a virtual reality simulation program developed at Iowa State University. The VR Factory uses the results of a simulation developed in AweSim to provide the numerical drivers of the simulation. The visualization of the simulation is accomplished through the creation of a virtual environment. This environment was designed as a tool for interpreting and analyzing the results of the simulation through interaction with the VR Factory. The intent of the VR Factory is to provide an intuitive interface to be used in analyzing a simulation of product flow through a factory.

## 3.1 General Aspects of the VR Factory

The VR Factory is a 3D, animated model of the Machining Automation Center (MAC) Cell at Sauer-Sundstrand in Ames, Iowa that can be viewed in a virtual environment. The present MAC Cell is a flexible machining system which contains five Machining Centers (four Makinos and one ENSHU), one Rail Guided Vehicle (RGV), a Work In Process (WIP) conveyor, Cascade deburr, Wash and Hone Center, and an Automatic Service and Retrieval System (ASRS) [13]. It is approximately 12000 square feet. The MAC Cell is the initial station in the process of building hydrostatic transmissions. Its purpose is to machine the housings from raw materials which are stored on the ASRS, and to place the finished housings back on the ASRS after all processes are complete. Figure 4 and Figure 5 show the arrangement of the equipment mentioned above.

The process of manufacturing the part as shown in the VR Factory begins at the ASRS as raw material. A part is brought down off the ASRS in a batch and then individually loaded onto the WIP conveyor. When a Work Set Station (WSS) is free, the part is taken off the WIP conveyor and put on a fixture at the WSS station. From the WSS, the part goes

into one of the Makinos through the RGV where it is machined. Once the part is done in the Makino, it is taken out of the Makino by the RGV and put back on the WSS where it is inspected and loaded onto the WIP conveyor. The part exits the WIP and, depending on the part, goes to the ENSHU. It next goes to the Cascade deburr and then to the Wash and Hone Center. After the part is through the Cascade deburr and the Wash and Hone Center, the part is sent back to storage on the ASRS.

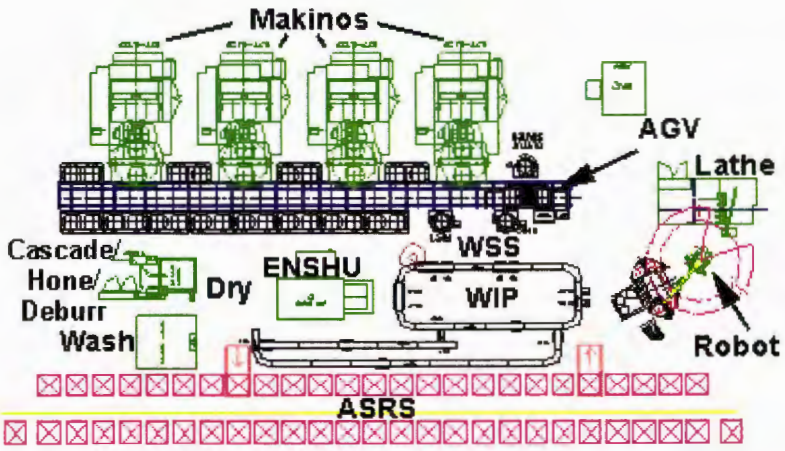



**Figure 4 - 2D Layout of Factory Floor**

**Figure 5 - Overview of the VR Factory**

Interaction with the VR Factory includes navigation throughout the work cell floor using a Fakespace PINCH™ Glove and part identification with a 3D-menu system. The VR Factory is written in C/C++ and utilizes Sense8's VR software WorldToolKit® library to enable a 3D, animated environment and the WTDX (WorldToolKit® Display Extensions) library, developed at Iowa State University as an extension of WorldToolKit®, for an easier way to control the display and motion tracking equipment. The 3D-menu system and the virtual animated hand representing the PINCH™ Glove were implemented using additional libraries also developed at Iowa State University. The following figure (see Figure 6) offers a visual representation of the VR Factory's software structure.

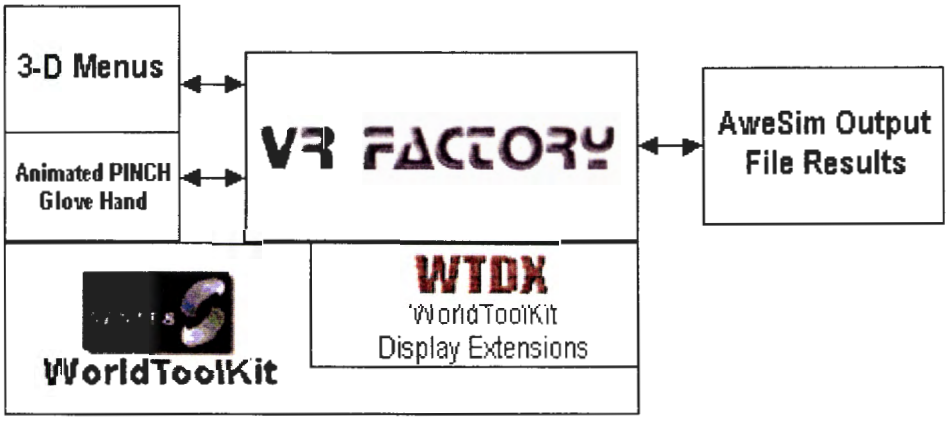



**Figure 6 - Program Structure**

### 3.2 Peripherals

The VR Factory can be viewed in several different environments: in a head-mounted display (HMD), on a stereoscopic projection screen, or in Iowa State University's C2. While any of these environments can be considered VR, each provides a different amount of immersion and this can lead to different levels of understanding the VR Factory.

Interaction with the virtual environment (created by any of the display devices

mentioned above) is achieved through a Fakespace PINCH™ Glove. The Fakespace PINCH™ Glove records contact between a user's fingers so various hand gestures can be used to control movement and virtual menu selections. The user's viewpoint and hand position are tracked with Ascension Flock of Birds™ magnetic trackers. In the case of the HMD, a magnetic tracker is attached to the HMD for viewpoint tracking. For the projection screen settings, the viewpoint is tracked by attaching a tracker to a pair of shutter glasses. In both applications, a magnetic tracker is attached to the PINCH™ Glove for tracking hand positions. When VR peripherals are not available, the user can interact with the program using a standard monitor and mouse.

### 3.3 Navigation and Interaction

Since the factory floor space is larger than the workspace of the virtual environment, tracking the viewpoint position with the Flock of Birds™ alone does not allow the user to examine the entire work cell. Extra navigation is needed to move to a viewpoint outside the physical workspace of the C2 structure. The PINCH™ Glove gestures enable this navigation while implementing navigation independent of head tracking. The user can look off to one side, yet use the PINCH™ Glove to navigate in any direction.

To navigate, the user reaches out, touching the index finger to the thumb, and pulling toward his/her body while holding the finger and thumb together. A good analogy would be grabbing a rope and pulling yourself along. This procedure is illustrated in Figure 7. Step 1 shows the initial position of the hand. Step 2 represents the user reaching out. After reaching out the user touches the index finger and thumb (Step 3). Between Step 3 and Step 4 the user keeps the thumb and finger pinched together and draws the hand inward to move the world objects closer. To accomplish this on the computer, the position of the

must be stored while the finger and thumb are touching. Each time the virtual world is updated and the finger and thumb are touching, the previous position stored is subtracted from the current position. The virtual world is translated based on the subtraction results.

Other aspects of the navigation are rotating the virtual world clockwise and counter-clockwise. By touching the middle finger to the thumb, the user's viewpoint rotates clockwise and by touching the ring finger to the thumb the user's viewpoint rotates counter-clockwise. If the C2 included the capability to display on all for walls, the rotation function could be eliminated.

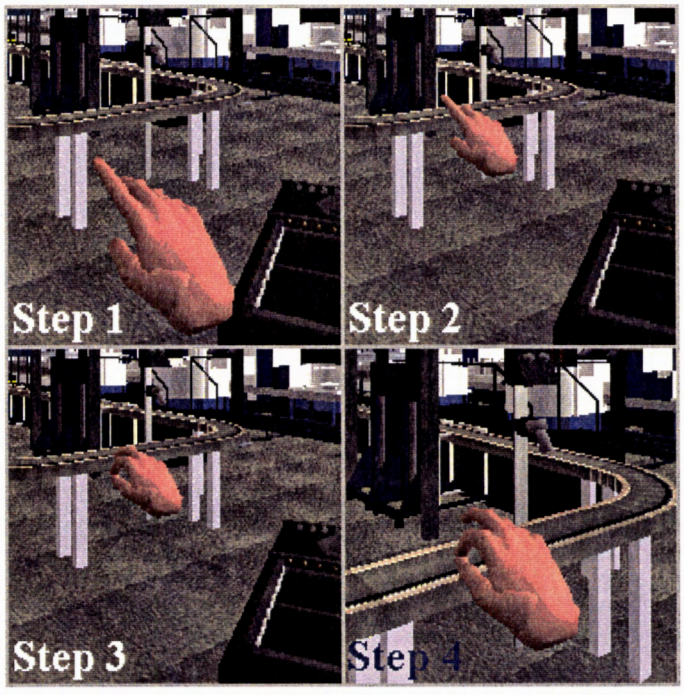



**Figure 7 - Navigation in the Virtual Environment**

Interaction with the VR Factory is accomplished through intersection with objects, a three-dimensional menu, and an information table. The three-dimensional menu can be positioned anywhere in the virtual space. To make the menu appear the user makes a fist making contact with the tips of all four fingers to the palm of the hand. The user can then

intersect the hand with the frame of the menu and, by gesturing with the index finger and thumb, and "grab" the menu. The menu then moves with the hand and can be placed anywhere in the virtual environment by releasing the index finger and thumb. The options on the menu list different possible simulations of the same factory work cell (see Figure 8). To choose a particular simulation, the user must intersect the virtual hand with the menu option and make a gesture with the index finger and thumb. When the hand intersects a virtual menu option, the option changes from white to green to verify the selection. Once the option has been chosen, the menu disappears and the simulation begins.


**Figure 8 - Virtual Menu**

Another aspect of the interaction with the virtual environment is the identification of each part and its characteristics. The user can open an identification table listing a part's type, its current station, the time left at the station, and the next station the part is headed to by touching the pinky finger to the thumb. This table will stay with the user while the user

navigates through the VR Factory. By intersecting the virtual hand with a part in question, the table is updated to show the current statistics of the part (see Figure 9). The table will not close until touching the pinky finger to the thumb again. With this in mind, the user uses the same table to identify any part within reach. Only the information in the table changes to represent the last part the virtual hand intersected.

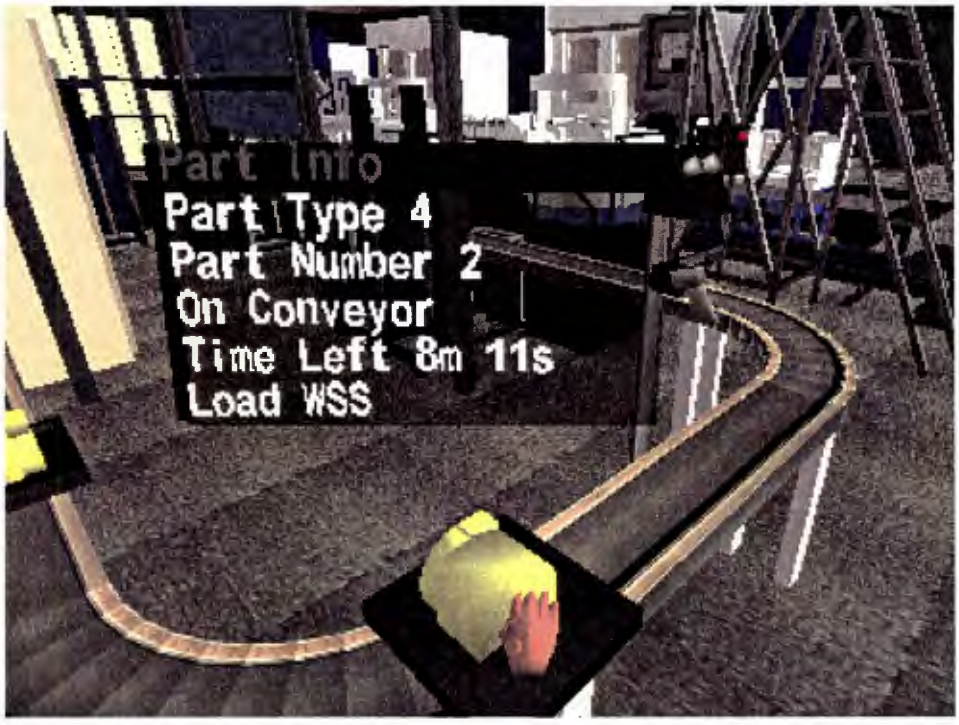



**Figure 9 - Information Table**

A very important feature of the VR Factory is the user's ability to manipulate time. To evaluate a factory work cell, the simulation of the factory work cell must run for the entire time it would in the real world. Since the VR Factory runs in real time, this means the VR Factory has to run for hours to show the entire process. With time manipulation, the user can jump forward and backward through time to evaluate the simulation at any time. Time can be adjusted by moving a slider in the virtual environment. When the user's hand

intersects (touches) the slider, a time is displayed floating in front of the user as seen in Figure 10. When moving the slider the time adjusts accordingly. By touching the middle finger to the thumb the user selects the desired simulation time and the world updates to that particular time.

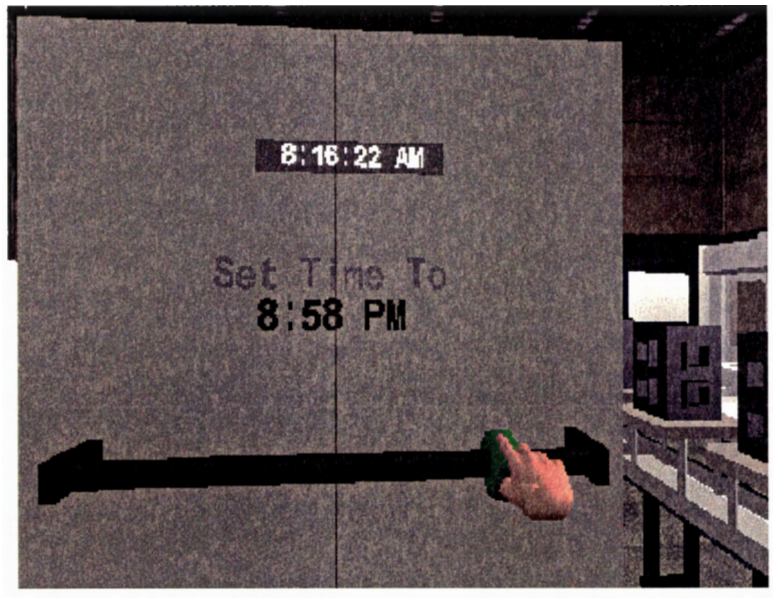



**Figure 10 - Time Manipulating Slider**

Through this type of interaction, the user can follow any part's progress through a simulation while also viewing the whole system as parts flow through the process.

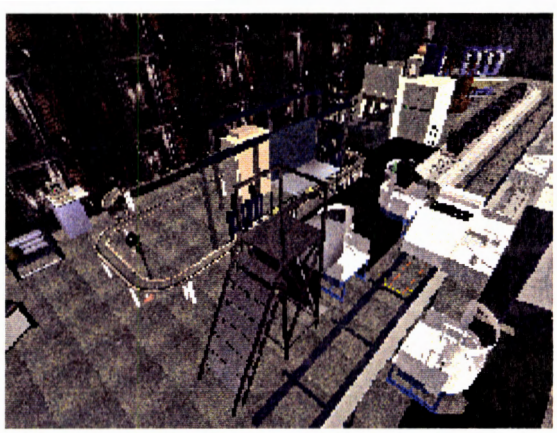
### 3.4 Execution of the VR Factory

An initial simulation is created using AweSim and the results are stored as text-based output. Upon execution of the VR Factory, the results are read into the program and stored in a two-dimensional array. When the program begins, the user is placed inside the synthetic environment composed of the factory floor. By using the navigation mentioned previously, the user is able to move about the VR Factory and inspect the components

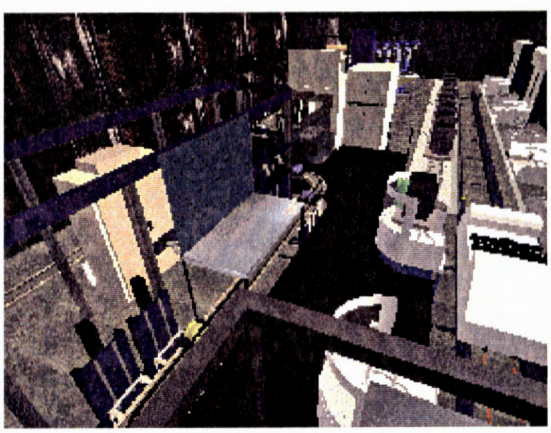
(machines, fixtures, etc.) of the work cell. This navigation, along with the tracking of the user's viewpoint, allows the user to get close to the models in the factory for inspection and view them from any angle, including from beneath the models.

To maintain a realistic feeling in the VR Factory, the user is restricted to viewing the VR Factory from an eye level viewpoint. The user can not "fly" around the factory and see it from above. In order to enable the user to get an overview of the entire factory work cell, a ladder (see Figure 11) was created. This ladder permits the user to view all aspects of the part flow at once. The user utilizes the ladder by walking up the ladder just as one would walk up a real ladder or by the forward navigation of the hand.

The user's ability to walk up the ladder is achieved by creating a type of collision detection in the VR Factory program. By checking to see if the user is in the same space as other objects in the virtual world, the program can disable the user's ability to move in the virtual environment. This creates the effect of being stopped by a physical object in the real world. This collision detection is limited to a few objects in the VR Factory. The ladder is included in the collision detection to enable the user to walk up the ladder and not go through it.



Arial View of Ladder



Viewpoint on Ladder

**Figure 11 - Virtual Inspection Ladder**

The main platform located between the machining centers and the WIP conveyor is also included in the program's collision detection. This allows the user to "stand on" the platform and view the parts along the conveyor and the RGV from the same viewpoint as in the real factory work cell. The rest of the objects in the VR Factory are not part of the program's collision detection. This lets the user inspect the parts even if they are inside other objects like the machining centers.

To start a manufacturing simulation, the user selects a particular simulation from the virtual menu and, once the data has been imported from the stored array, the manufacturing process begins at the simulation's initial time. At this point the user can not only inspect the machines and tools in operation, but also follow any part through a complete manufacturing process. When following the part through its process, the user can identify its characteristics by viewing the virtual identification table. As mentioned before, the user opens up the information table by touching the pinky finger to the thumb. This table appears directly in front of where the user is looking. While this table is open the user can "touch" different parts and the information in the table is updated to show the new part's information.

The user can at any time begin a new simulation by making a selection from the virtual menu and the process is started over. The user can also adjust the time by using the slider at any time while the VR Factory is running. The slider provides the user with the capability to repeat a part of the process as many times as necessary.

This chapter provided an overview of the functionality of the VR Factory. There are two main aspects of creating a virtual environment as a visualization tool for discrete event simulation. They are the creation of the virtual world and the implementation of the simulation results. The following chapter describes the process in which the geometric models for the VR Factory were created and animated, and Chapter 5 outlines the implementation of the simulation results.

# CHAPTER 4. CREATION OF THE VIRTUAL WORLD

The creation of the virtual world consists of creating the geometric models and programming the interaction and animation of these models. The VR Factory program loads in previously created models and places them in the VR Factory. This chapter explains the process of creating these models, implementing animation in the virtual environment, and implementing interaction in the virtual environment.

## 4.1 Creation of Factory Models

The first step in creating the geometric models was acquiring all of the necessary measurements and dimensions. Floor plans and dimensional drawings of the machines were used as references, but since the VR Factory is modeled after the Sauer Sundstrand Machining Automation Center (MAC) Cell located in the Ames area, dimensions were also acquired by measuring the actual objects.

CAD software (Pro/Engineer®) and modeling software (World Up™ Modeler, MultiGen® II) were used to build the models. Some of the geometric models were already created using Pro/Engineer® before this research began because of a previous research project. They were exported from Pro/Engineer® and imported into WorldToolKit®'s World Up™ Modeler where they were modified for efficiency and realism. The rest of the models were either built entirely with World Up™ Modeler or initially with MultiGen® II because of the developer's experience with MultiGen® II and converted for use in WorldToolKit® into the World Up™ Modeler's file format.

After the geometric aspects of the models were created, material properties such as color, shininess, and emissive properties were assigned to the models using WorldToolKit®'s World Up™ Modeler. By manipulating these properties, a model becomes more realistic. For example, if a particular machine is made of metal, the shininess property of the representative model is increased (see Figure 12).

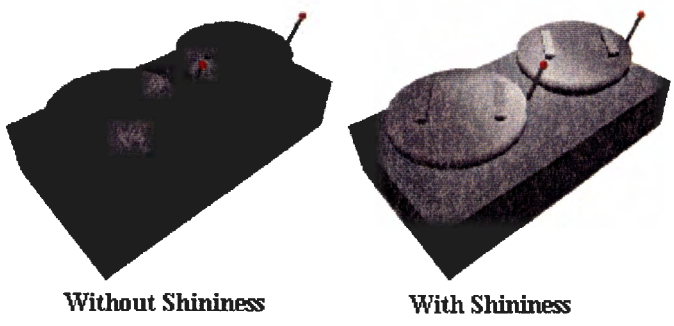



**Figure 12 - Material Properties Example**

Textures were also applied to the models in the modelers. The textures were created in two main ways. They were either gathered by taking pictures of the actual machines and tools or created with Adobe PhotoShop. Textures, like material properties, add realism to the virtual environment. In fact, they can add much more realism to the models than the material properties. Figure 13 shows how a few textures can add realism to a model of a computer monitor and its surroundings. However, textures require larger amounts of memory, unlike material properties, and therefore slow down the speed, or frame rate, of the computer. The frame rate is the rate of frames the computer can render in a set amount of time. A good analogy would be the flashes of light from a strobe light. If the flashes were fast enough, the strobe light would appear to never turn off. By slowing down the speed of the flashes, anything animate in the world would seem to jump from one position to the next.

This is what happens to the virtual environment when the frame rate is decreased. Realism is lost and interaction with the world becomes less intuitive. Thus, if too many textures are applied to the models, the program's frame rate will slow down causing the problems mentioned above. This problem, realism vs. the speed of the program, is not only a consideration when dealing with textures, but it also affects other aspects concerning the creation of the geometric models.

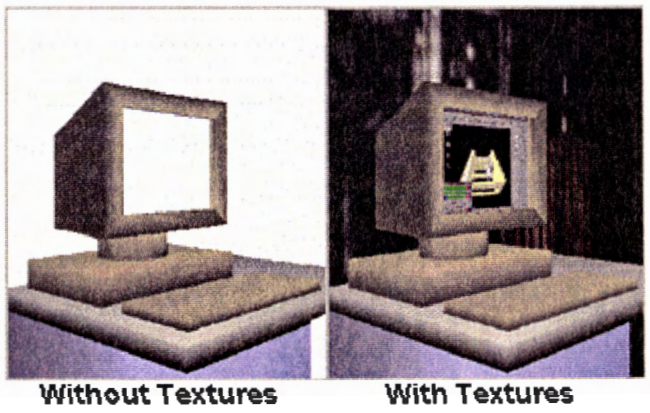



**Figure 13 - Texture Example**

One such aspect is the number of polygons that make up a model. Figure 14 shows two machining center doors. Both doors have the same dimensions yet the door on the left has 772 polygons and the one on the right has only 330 polygons. Since a computer stores the position of every vertex and its connectivity information, using the door on the left instead of the door shown on the right would be a dramatic increase in the amount of memory needed. Thus, creating a model with the smallest amount of polygons while also trying to keep the model realistic was a constant consideration. A common solution to reducing the number of polygons in the model was to decide if a certain feature of the model would be noticed or seen at all in the simulation. If it would not be noticed it was deleted from the model.

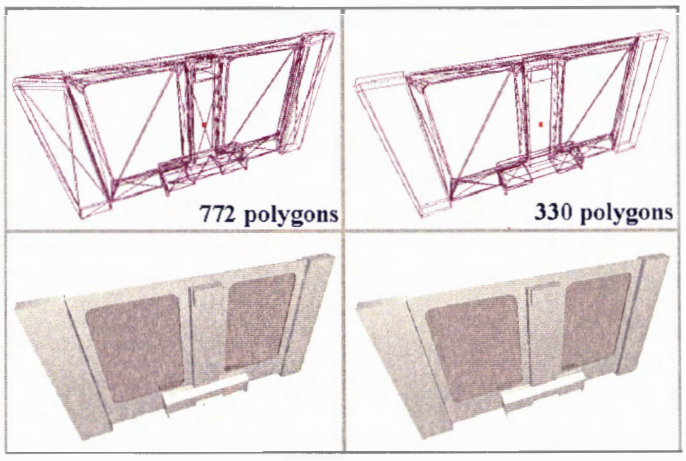



**Figure 14 - Number of Polygons Example**

Once the models were created, they were loaded into the program and translated to either the position defined by the floor plan or placed in a position relative to other objects in the virtual factory. As more and more models were loaded into the program it was becoming apparent that the models were still too complex (too many polygons). This in turn caused the program to slow down. To maintain speed, level of detail (LOD) models were introduced. LOD is "modeling the same object at different detail levels and the appropriate one is chosen for display based on some viewing criteria and system performance" [14]. In other words, if the viewpoint of the user is close to a particular object in the factory, a detailed version of the object is displayed. If the user is far enough away from the object so that the smaller features could not be distinguished, a simpler version of the model is displayed [15]. Figure 15 shows the complex model displayed when the user is close to the Machining center and the simple model displayed when the user is further away. With the combination of reduction of polygons, limited use of textures, and LOD, a more efficient model of the factory work cell was created.

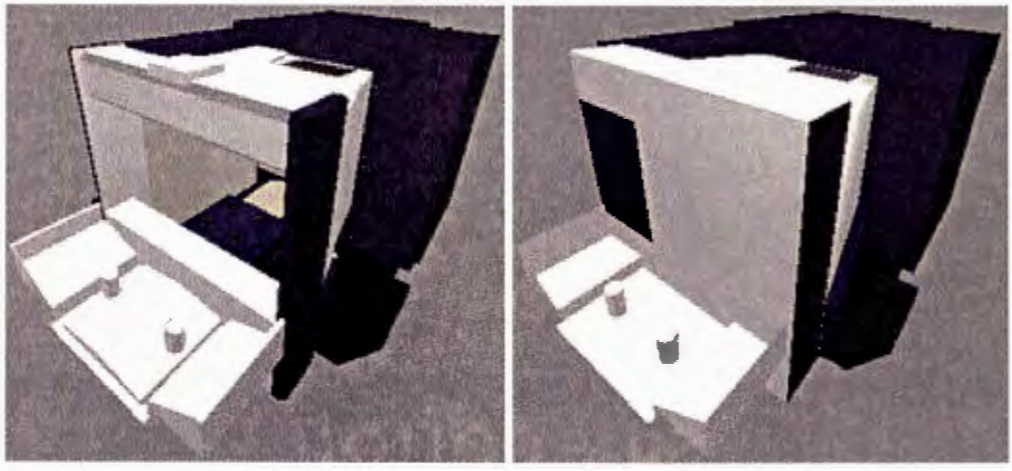



**Figure 15 - Level of Detail Models for a Machining Center**

### 4.2 Animation of the Virtual World

Once the geometric models in the VR Factory were constructed and placed in the virtual environment, the implementation of the simulation could be performed. Animation of the objects in the virtual environment required knowledge of how the actual machines and tools worked. Most of this knowledge came from observing the real factory work cell.

The program's structure was based chiefly on how each individual object in the virtual world would be animated. Each geometric model in the VR Factory is considered a node in the program and the structure of these nodes is called the node hierarchy. The node hierarchy's structure is similar to the structure of a family tree with each node having a parent node and possibly a child node. The child node would then inherit the motion of the parent node.

With the knowledge of how the machines and tools in the work cell worked, the models of these machines and tools were structured for proper animation. For example, a door on a

machining center was loaded into the program as a child node of the machining center node. This allowed for the door to be animated independently of the machining center, but if the location of the machining center were moved, the door would also move. Figure 16 represents the node hierarchy for this example.

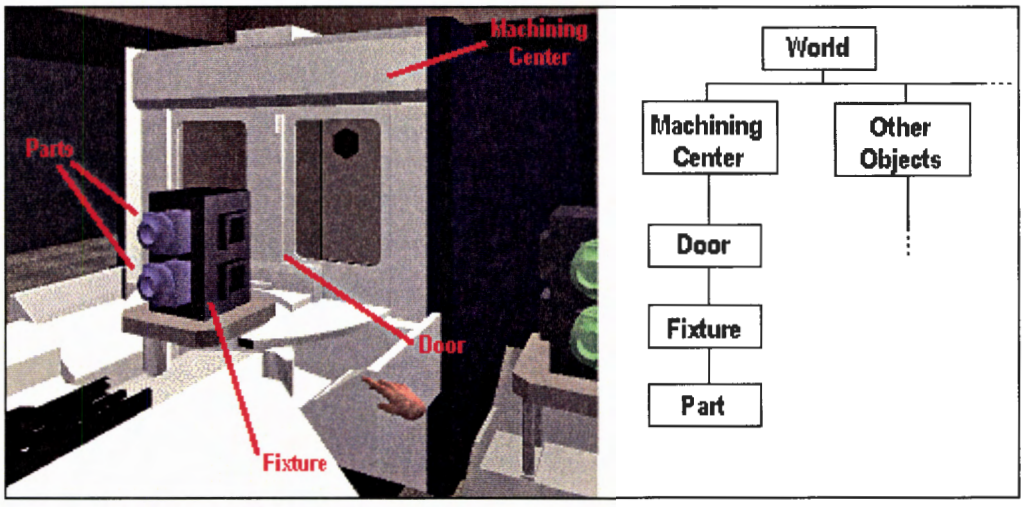



**Figure 16 - Node Hierarchy of Machining Center**

The models of the parts being manufactured in the VR Factory are initially not part of the node hierarchy. As the simulation runs, the parts are attached (become child nodes) and detached from other nodes. When the part is sitting on a particular pallet it is attached to the pallet. If the pallet were to move, the part would also move. Detaching and then attaching the part to another node would then make the part move with this node. In this manner, the parts are "carried" throughout the manufacturing process.

Collision of the geometric models was also a consideration in the animation of the VR Factory because of the need for realism. The VR Factory, not designed with collision detection, allows for geometric objects to occupy the same space. This collision detection had to be done visually and corrected manually because using the computer to analyze

collision detection would reduce the frame rate of the program. The decrease in the frame rate would cause the same speed vs. realism problems encountered when applying textures.

Time was a very important factor in the animation of the VR Factory. It is the major independent variable in the discrete event simulation of the manufacturing process. Therefore, it is the controlling variable in the virtual environment. For example, when a part is moved from one location to another location. This occurs at a specific time, which is defined by the results of the simulation. The program continually checks the time and when the clock arrives at the time given by the simulation, the part is translated. In this way, animation is activated and deactivated at specific points in time. By setting up the animation in this manner, the implementation of the discrete event simulation results is fluid.

### 4.3 Interaction Control in the VR Factory

The virtual model of the user's hand was created in the same way as the other geometric objects in the VR Factory (see Figure 17). It is loaded into the VR Factory as a node attached to the root node (the initial node all other nodes originate from). This node is tested for intersection with another node in the VR Factory to determine if the hand "touches" a certain object represented by its corresponding node. In this manner the computer identifies when the hand touches relevant objects such as parts, menus, and the slider.

Interaction in the VR Factory is divided into three modes: menu, navigation, and default. This arrangement eliminates unnecessary testing for intersections and allows the same gestures to be used for several tasks. Without these modes, a gesture could effect two or more aspects of the virtual environment at once. Take the following example. If the

hand intersects the time controlling slider and is in the default mode, a display for setting the clock to a new time is opened up in front of the user. If the user is navigating forward and the virtual hand intersects the slider, the display is not opened because the VR Factory is in its navigation mode. This eliminates the unexpected appearance and disappearance of the display.

**Figure 17 - Model of the Virtual Hand**

Another aspect of controlling the interaction in the VR Factory is the navigation through the virtual world. Navigation is disabled in certain modes of the program. One such mode is when the virtual menu is open. This allows the user to use the gestures previously defined for navigation to be used for choosing a menu option. The gesture of touching the middle finger to the thumb controls rotation in a clockwise manner in navigation but selects the menu option also. With this type of control over the interaction in the VR Factory, a single gesture can control many different aspects of the VR Factory.

# CHAPTER 5. IMPLEMENTATION OF DISCRETE EVENT SIMULATION RESULTS

Implementation of the discrete event simulation within the VR Factory involves two main tasks. The first task is to create the simulation and an output file of the simulation results. Lori Melaas carried out this task under the direction of Dr. Moller-Wong[13]. Visual SLAM II and AweSim 2.0 by Pritsker Corporation were the software packages used for the simulation. The second task involves structuring the VR Factory to correctly utilize the results of the discrete event simulation. This task can be divided into several parts: loading the results into the VR Factory, organizing the results once they are loaded, and controlling the VR Factory animations through the organized results. This particular approach does not allow for the alteration of the simulation in the VR Factory. The virtual environment was created with the intent to provide a visual representation of a simulation. The simulation was only to be altered in the simulation software. This means the simulation can only be altered when it is being created in visual SLAM II. Thus, the VR Factory serves as a viewer for the simulation in a virtual environment.

## 5.1 Creation of Simulation File

The software used to create the simulation, Visual SLAM II and AweSim 2.0, allows non-experienced programmers to program simulations because of the manner in which the programming is done. Instead of writing code, the programmer assembles symbols together to form the code. This also benefits the programmer in that the symbols show a visual representation of the process as the simulation is being developed. To better understand the MAC Cell's system and also prepare for the programming of the system, a

flow chart was constructed. This flow chart was created and improved upon to ensure an accurate representation of the MAC Cell before creating the program.

Also involved in the programming of the simulation was the creation of an output file. The VR Factory can read in a text file and AweSim can export a text file of user selected variables. Therefore, the output file was created as a text file. The elements in the text file were selected based on the following question: What input variables are needed in the virtual environment and what output variables can be supplied by the simulation software?

Time is the major independent variable in the simulation of the manufacturing process. Thus, at specific times the state of the part changes. For example, a part may be transported to a machining center, machined, transported to an inspection station, inspected, and then returned to storage. Each task is defined by a specific starting time. If a virtual environment were created to visualize this process, the necessary information about the part's characteristics and how it moves through the process should already be defined in the VR Factory. The time at which the part changes its state and the destination of the part are determined from the simulation. This is the information that needs to be loaded into the VR Factory from the results of the discrete event simulation.

To obtain the time and destination of the parts flowing through the process, the parts are marked and traced as the simulation is run. When a part changes state during the simulation, the time and location of the part is recorded as attributes of the part. Output simulation statements define the order in which the attributes are exported into the text file. The order in which the attributes are exported is defined in a manner best suited for the VR Factory to import. An abridged example of an output file is given in the appendix. The headings for the part attributes are not included in the actual file.

Since the VR Factory is a visual representation of the manufacturing process, more information than the time and destination of the part is needed. The additional information

must be determined independently of the simulation results. This information includes the physical characteristics of the parts and stations involved in the simulation and the manner in which the part would travel from one station to the next. These physical characteristics include not only the size and shape of the objects in the virtual world, but also the functionality of the objects. Take, for example, the operation of the Makino machining centers. The loading and unloading of the fixtures is through a rotating door. The rotation of the door is an example of the machining center's functionality.

The manner in which parts travel from one station to the next is composed of several factors. How and where a part is attached to a machine, fixture, pallet, etc. and also how a part is animated at a station define this manner in which a part travels. An example of this would be the process of loading a part attached to a fixture into a machining center. The output file defines what machining center the part is being loaded into and the time at which this takes place. To show a visual representation of this, the VR Factory also needs to know how the part is fastened onto the fixture, how the RGV attaches and detaches the fixture, and how the machining center loads the fixture.

The need for this additional information brings up another topic related to the contents of the output file. Without a visual representation of the simulation, the simulation program needs only enough detail to represent the real system numerically. The only information needed in the example given above is which machining center the part is being loaded into and the time at which this takes place. The times at which the fixture is attached to the RGV and when it is detached could also be included in the simulation. This would create a larger simulation program and the extra information is generally not of concern to the overall flow of the part through the manufacturing process. However, when there is a visual representation like the VR Factory the extra information could prove useful. Instead of the VR Factory taking the one given time and dividing up the animations before and after the

time, the VR Factory could have only one movement for each time given. This would in turn make the virtual world based off of the simulation results a more precise representation of the simulation.

### 5.2 VR Factory Structure for Loading of The Simulation

The actual reading of the simulation file into the VR Factory is an easy process because the program is written in C/C++. The data within the file is read into a two-dimensional array declared in the VR Factory. The first dimension identifies the part number. This part number is not the part type. It represents the part relative to the entire group of parts run through the simulation. The second dimension is made up of the attributes of each part. These attributes, mentioned earlier, include the time and location of the part. Table 1 is representative of a section of the actual output file. This table can also represent a matrix or two-dimensional array, which is how the VR Factory stores the output file. Each row designates a part number and each column denotes an attribute of the part.

**Table 1 - Partial Output File**

| Part Type | Begin ASRS | Begin WIP | WSS | Exit WIP | Exit WSS | Mill # | Exit Mill | Gauge | Return to WIP | End time |
|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 0 | 4.42 | 8 | 4.42 | 17.75 | 4 | 140.34 | 0 | 140.34 | 232.09 |
| 3 | 0 | 4.42 | 8 | 4.42 | 17.75 | 4 | 140.34 | 0 | 140.34 | 232.09 |
| 3 | 0 | 9.7 | 7 | 10.05 | 23.38 | 5 | 145.97 | 0 | 145.97 | 232.09 |
| 3 | 0 | 9.7 | 7 | 10.05 | 23.38 | 5 | 145.97 | 0 | 145.97 | 232.09 |
| 3 | 10 | 14.07 | 8 | 17.75 | 65.61 | 6 | 208.76 | 0 | 208.76 | 232.09 |
| 3 | 10 | 14.07 | 8 | 17.75 | 65.61 | 6 | 208.76 | 0 | 208.76 | 232.09 |
| 3 | 20 | 22.37 | 8 | 65.61 | 140.34 | 4 | 216.09 | 0 | 216.09 | 232.09 |
| 3 | 20 | 22.37 | 8 | 65.61 | 140.34 | 4 | 216.09 | 0 | 216.09 | 232.09 |
| 3 | 20 | 26.09 | 7 | 71.34 | 145.97 | 5 | 221.72 | 0 | 221.72 | 232.09 |
| 3 | 10 | 17.14 | 7 | 23.38 | 71.34 | 3 | 232.09 | 0 | 232.09 | 232.09 |

The physical characteristics of the parts are already defined in the VR Factory through a combination of a variable structure and an initialization function. The structure (PART[ ] ) is basically a group of variables that would define the characteristics of a part. The purpose of creating a structure for the part is that it allows the passing of an entire data set for a part through one variable name. Included in the structure is a node variable to store the part's geometry, floating and integer variables for storing the part's attributes, and also Boolean variables. The amount of variable structures the VR Factory allows for is determined by the size of the first dimension of the simulation array.

The VR Factory loads the data from the simulation array into PART[] through a function called create_part. This function is invoked for each part, creating the geometry and setting the values of the attributes defined in the structure based on the values in the simulation file. For example, the first part defined in the appendix simulation file is a part type 3. The create_part function creates a node consisting of a model of a part type 3 in the variable PART[0]. The entire row of locations and times are loaded into the variable PART[0] also. Once the data from the simulation file is loaded and compiled into useable information, the simulation begins.

### 5.3 VR Factory Structure for The Animation of The Simulation

In order to have a running simulation, the parts must move according to time. Thus, as mentioned in section 4.2, time is the controlling variable in the VR Factory. The simulation file has time starting at zero and all part times are incremented in minutes (see Appendix). The VR Factory contains a variable representing time (TIME) and this is set to zero once the simulation begins. This variable, TIME, is continually updated according to the computer's internal clock. For user interface, the time of zero is portrayed as 8:00 AM and displayed as a virtual digital clock.

To animate the virtual parts through their process, functions were created to continually update the positions of the parts and affected machines and fixtures. There were four particular part types so four functions were created to depict the process for each part type. Since the form of these functions is similar, they can be thought of as one function, called part_process, for the sake of explanation. The function part_process takes in the variable TIME along with the structure representing the part each time the computer iterates through the program. All possible animations of the part and the models interacting with the part are defined in this function and are activated and deactivated according to the variable TIME. These possible animations are activated and deactivated by being contained in conditional statements that check a part's attributes against the current time and also whether the part has already been activated or not (by the Boolean flags mentioned above). Figure 18 shows abridged pseudo code for the part_process function. The code uses the variable names given in the appendix simulation file.

```
If TIME > Begin_ASRS and < Begin_WIP and part not down from ASRS
        • Bring part down to ASRS exit
If TIME > Begin_WIP and < Exit_WIP and part not moving along WIP
        • Place part on pallet
        • Move pallet along WIP
If TIME > Exit_WIP and < Exit_WSS and part not attached to fixture
        • Put pallet on WIP lift
        • Raise lift
        • Transport part from pallet to fixture on WSS
If TIME > Exit_WSS and < Exit_Mill and part not in Mill
        • Load part/fixture into Mill
If TIME > Exit_Mill and < Return_to_WIP and part not moving along WIP
        • Put part/fixture on WSS
        • Transport part from WSS to WIP lift
        • Lower lift
        • Move pallet along WIP
If TIME > Return_to_WIP and < End_Time
        • Transport part from WIP to ASRS Entrance
```

**Figure 18 - Abridged Pseudo Code for part_process**

Take, for example, the first part of the simulation file in the appendix. The Exit Conveyor time is 4.42 minutes and Exit WSS time is 17.75. If the time was 10 minutes into the simulation, the conditional statement "If TIME is greater than the Exit WIP time and less than Exit WSS time and the part has not been attached to the fixture..." would be true and would execute the commands inside the loop. These commands tell the part to be transported off the conveyor and attached to the fixture at the WSS station and a Boolean flag referring to whether the part has been attached to the fixture or not is set to true.

### 5.4 Time Manipulation of The Simulation

Since the VR Factory runs in real time and a simulation can be hours long, the user needs the ability to jump through time. Manipulation of time is done with the slider mentioned in section 3.2. This slider is a geometric node that is linked with the movements of the virtual hand when the hand intersects the slider. The user sees another display of a digital clock affected by the position of the slider. When a new time is selected, the world updates accordingly.

To update the world accordingly, all of the objects in the virtual world, including the parts, are reset to their initial positions, time is set back to zero, and the parts' attributes are set to their original state. Another function is then called with the new adjusted time as input. This function, jump_to_time, is similar to the part_process function in that it is made up of conditional statements. It differs though in the animation of the parts. The jump_to_time function does not continually update the world but rather instantaneously changes the world's characteristics. Thus, it does not animate the objects in the world but deletes them from one station and attaches them to the next. However, the flags identifying

where a part is in its process are updated just as it is in part_process. Figure 19 shows the pseudo code for the function jump_to_time.

The following is an example of how time is adjusted in the VR Factory. If the time is set for twenty minutes into the simulation, the function jump_to_time is called for each part with twenty minutes as its input. The times given by the output file for each part can be considered checkpoints in the part's process. For a particular part there is a checkpoint at fifteen minutes to tell the part to move from the WIP to the WSS station. There is also a checkpoint at twenty-five minutes to tell the part to move from the WSS station to the machining centers. The function jump_to_time attaches the part to the WIP and then detaches it and attaches it to the WSS station based on the input of twenty minutes.

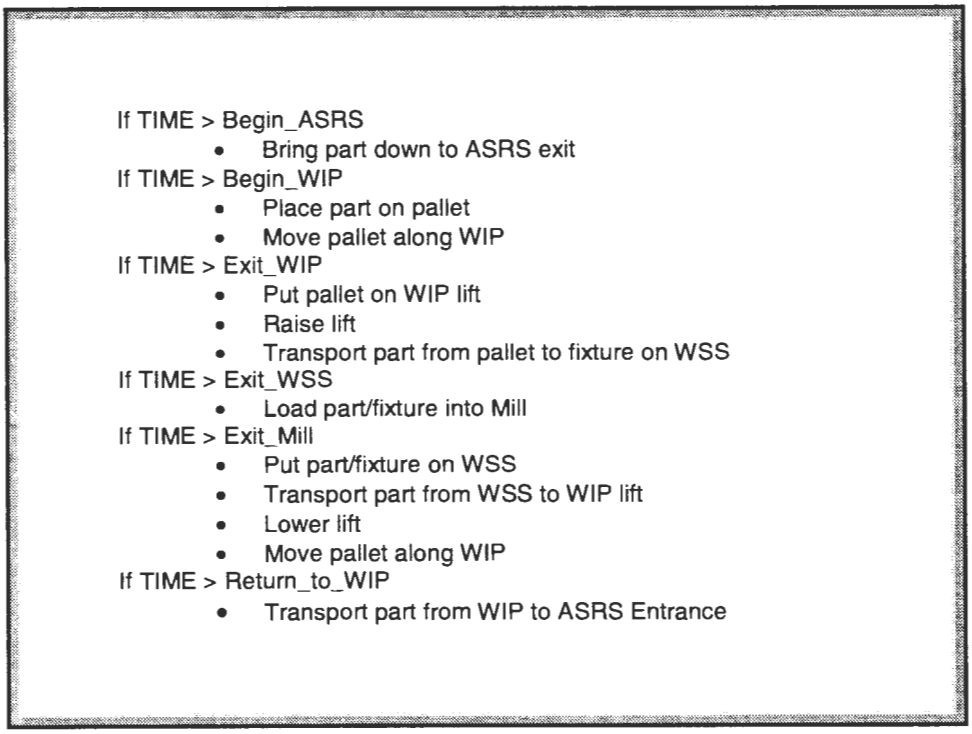



**Figure 19 - Abridged Pseudo Code for jump_to_time**

One final aspect of the VR Factory's creation is the capability to change to a new simulation through a menu selection. The process starts out the same as when the time is adjusted for simulation. Instead of calling a function like jump_to_time, the VR Factory resets all the values in the PART[] structure to null so that a simulation of any size can replace the existing one without old values remaining.

### 5.5 Evaluation of the Simulation Implementation

The VR Factory was demonstrated to many people throughout its development. The manner in which these people interacted with the simulation visualization tool identified the benefits of the VR Factory. A strong point of the VR Factory was the type of navigation. This navigation proved to be intuitive for many people. Through this intuitive method of navigation, the user of the VR Factory was able to follow a part through the manufacturing process and therefore get a better understanding of how it was created.

The user was able to view the results of four simulations each with different part mixtures. The four different part mixtures were created in AweSim. They allow the user to not only analyze one possible part mixture, but to also compare it to others. At a specific time the user was able to toggle between simulations to visually compare the number of parts at a particular station, the number of parts between stations, and the number of parts completed. This capability of the VR Factory demonstrates the possible benefits of implementing discrete event simulation results into a virtual environment.

# CHAPTER 6. CONCLUSIONS AND RECOMMENDATIONS

## 6.1 Conclusions

The use of Virtual Reality in manufacturing is growing due to the continually increasing capabilities of computers. Development of manufacturing simulation is now including 3D graphics because of the visualization benefits. With the immersion advantages of VR, interaction and analysis of the 3D simulations becomes a more intuitive and naturalistic process. The VR Factory demonstrates this more intuitive and naturalistic process and the possibilities that follow.

The VR Factory provides a visualization tool for viewing and analyzing the working processes and problems of the actual manufacturing work cell. It was designed with the capability to investigate how various changes to the manufacturing work cell part mixture affect the part production. Interaction and navigation were incorporated in the VR Factory in a manner that enables a person to effectively identify the working processes and the problems associated with the working process.

With these capabilities, the part flow of the manufacturing process was visualized. The user is able to instantly identify how many parts are on the conveyor, at a machining center, or completely through the process. Identifying how many parts are on the conveyor is helpful when analyzing the simulation for bottlenecks. Another benefit of the virtual environment is the time considerations involved in the process. Instead of seeing a five-minute wait on an output report, the user can wait five minutes and get a true feeling of how long five minutes is in relation to the simulation. The VR Factory also helped confirm the validity of the simulation results. It identified points in the simulation where a deadlock, two or more parts are occupying the same resource, occurred. This identification helped to verify the results of the simulation created in AweSim. Because of the mentioned

capabilities of the virtual reality program, the VR Factory was a successful representation of a visualization tool for discrete event manufacturing simulations.

### 6.2 Recommendations for Future Work

There are many aspects of the visualization of a simulation that the VR Factory has not fully explored and are currently being developed. One is implementing different scenarios developed through the use of SLAM II. Examples of the scenarios would include having a different number of machining centers, other machines, or operators of the machines. The user would then be allowed to interactively select the scenario of choice while in the virtual environment. This would enable the user to make comparisons between the scenarios to evaluate which is best.

The current version of the VR Factory incorporates only part of the entire manufacturing process of the Machining Automation Center (MAC) Cell at Sauer-Sundstrand. The inclusion of more of the process is another area for future work. The manner in which the VR Factory was created is not dependent on the size or intricacy of the simulation created in AweSim. Because of this, the simulation can be extended and also expanded to include more detail of the process.

Another recommendation for future work is enabling the VR Factory to speed up time. Instead of allowing the user to only jump to a time in the simulation and evaluate it in real time, it could be more beneficial show the VR Factory in a fast forward mode. This mode could show the parts moving through the process not in real time, but moving at a pace twice as fast or even faster. This would enable the user to see how a part moves through its entire process in a short amount of time.

The level of detail in the actual simulation is another aspect that could be improved upon. As mentioned in section 5.1, the detail of the simulation needs to reflect what is

needed to create and animate the necessary components of the manufacturing process. Communication between the designer of the simulation and the designer of the virtual world is required to clearly define these creations and animations before and during the development of a virtual environment for simulation.

Once additional aspects like the ones mentioned above are added to the VR Factory, a study will be performed to determine the benefits of VR in the visualization of manufacturing simulations. The VR Factory could be compared against a traditional workstation-based simulation. Results of this type of study could justify the use of VR as a visualization tool for simulations.

# APPENDIX. SIMULATION FILE

| Part Type | Begin ASRS | Begin WIP | WSS | Exit WIP | Exit WSS | Mill # | Exit Mill | Gauge | Return to WIP | End time |
|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 0 | 4.42 | 8 | 4.42 | 17.75 | 4 | 140.34 | 0 | 140.34 | 232.09 |
| 3 | 0 | 4.42 | 8 | 4.42 | 17.75 | 4 | 140.34 | 0 | 140.34 | 232.09 |
| 3 | 0 | 9.7 | 7 | 10.05 | 23.38 | 5 | 145.97 | 0 | 145.97 | 232.09 |
| 3 | 0 | 9.7 | 7 | 10.05 | 23.38 | 5 | 145.97 | 0 | 145.97 | 232.09 |
| 3 | 10 | 14.07 | 8 | 17.75 | 65.61 | 6 | 208.76 | 0 | 208.76 | 232.09 |
| 3 | 10 | 14.07 | 8 | 17.75 | 65.61 | 6 | 208.76 | 0 | 208.76 | 232.09 |
| 3 | 20 | 22.37 | 8 | 65.61 | 140.34 | 4 | 216.09 | 0 | 216.09 | 232.09 |
| 3 | 20 | 22.37 | 8 | 65.61 | 140.34 | 4 | 216.09 | 0 | 216.09 | 232.09 |
| 3 | 20 | 26.09 | 7 | 71.34 | 145.97 | 5 | 221.72 | 0 | 221.72 | 232.09 |
| 3 | 10 | 17.14 | 7 | 23.38 | 71.34 | 3 | 232.09 | 0 | 232.09 | 232.09 |
| 3 | 10 | 17.14 | 7 | 23.38 | 71.34 | 3 | 232.09 | 0 | 232.09 | 562.31 |
| 3 | 20 | 26.09 | 7 | 71.34 | 145.97 | 5 | 221.72 | 221.72 | 249.72 | 562.31 |
| 3 | 20 | 29.2 | 8 | 140.34 | 216.09 | 4 | 291.84 | 0 | 291.84 | 562.31 |
| 3 | 20 | 29.2 | 8 | 140.34 | 216.09 | 4 | 291.84 | 0 | 291.84 | 562.31 |
| 3 | 30 | 33.51 | 7 | 145.97 | 208.76 | 6 | 353.84 | 0 | 353.84 | 562.31 |
| 3 | 30 | 33.51 | 7 | 145.97 | 208.76 | 6 | 353.84 | 0 | 353.84 | 562.31 |
| 3 | 30 | 36.67 | 7 | 232.09 | 272 | 5 | 429.67 | 0 | 429.67 | 562.31 |
| 3 | 30 | 36.67 | 7 | 232.09 | 272 | 5 | 429.67 | 429.67 | 457.67 | 562.31 |
| 3 | 40 | 45.67 | 8 | 271.59 | 291.84 | 4 | 562.31 | 0 | 562.31 | 562.31 |
| 3 | 40 | 45.67 | 8 | 271.59 | 291.84 | 4 | 562.31 | 0 | 562.31 | 562.31 |
| 4 | 170 | 175.98 | 8 | 216.09 | 271.59 | 3 | 350.59 | 0 | 350.59 | 678.09 |
| 4 | 170 | 175.98 | 8 | 216.09 | 271.59 | 3 | 350.59 | 0 | 350.59 | 678.09 |
| 4 | 210 | 216.26 | 7 | 350.59 | 562.31 | 4 | 603.36 | 0 | 603.36 | 678.09 |
| 4 | 210 | 216.26 | 7 | 350.59 | 562.31 | 4 | 603.36 | 0 | 603.36 | 678.09 |
| 4 | 210 | 220.62 | 7 | 562.31 | 621.05 | 3 | 678.09 | 0 | 678.09 | 678.09 |
| 3 | 40 | 47.96 | 8 | 291.84 | 353.84 | 6 | 619.3 | 0 | 619.3 | 778.16 |
| 3 | 40 | 47.96 | 8 | 291.84 | 353.84 | 6 | 619.3 | 0 | 619.3 | 778.16 |
| 3 | 40 | 47.01 | 7 | 272 | 350.59 | 3 | 621.05 | 0 | 621.05 | 778.16 |
| 3 | 40 | 47.01 | 7 | 272 | 350.59 | 3 | 621.05 | 0 | 621.05 | 778.16 |
| 3 | 50 | 52.85 | 8 | 353.84 | 603.36 | 4 | 734.13 | 0 | 734.13 | 778.16 |
| 3 | 50 | 52.85 | 8 | 353.84 | 603.36 | 4 | 734.13 | 0 | 734.13 | 778.16 |
| 3 | 50 | 56.87 | 8 | 603.36 | 624.98 | 5 | 748.15 | 0 | 748.15 | 778.16 |
| 3 | 50 | 56.87 | 8 | 603.36 | 624.98 | 5 | 748.15 | 0 | 748.15 | 778.16 |
| 3 | 0 | 75.01 | 8 | 637.8 | 655.58 | 6 | 778.16 | 0 | 778.16 | 778.16 |
| 3 | 0 | 75.01 | 8 | 637.8 | 655.58 | 6 | 778.16 | 0 | 778.16 | 778.16 |
| 4 | 210 | 220.62 | 7 | 562.31 | 621.05 | 3 | 678.09 | 0 | 678.09 | 861.95 |
| 4 | 250 | 256.85 | 7 | 629.11 | 734.13 | 4 | 786.2 | 0 | 786.2 | 861.95 |
| 4 | 250 | 256.85 | 7 | 629.11 | 734.13 | 4 | 786.2 | 793.5 | 821.5 | 861.95 |
| 4 | 250 | 258.67 | 8 | 748.15 | 809.88 | 3 | 861.95 | 0 | 861.95 | 861.95 |
| 4 | 250 | 258.67 | 8 | 748.15 | 809.88 | 3 | 861.95 | 0 | 861.95 | 861.95 |
| 1 | 60 | 72.4 | 8 | 625 | 637.8 | 6 | 655.17 | 0 | 655.17 | 888.52 |
| 1 | 60 | 72.4 | 8 | 625 | 637.8 | 6 | 655.17 | 657.58 | 677.58 | 888.52 |
| 1 | 70 | 75.52 | 8 | 657.6 | 678.09 | 3 | 809.88 | 0 | 809.88 | 888.52 |
| 1 | 70 | 75.52 | 8 | 657.6 | 678.09 | 3 | 809.88 | 0 | 809.88 | 888.52 |
| 1 | 80 | 85.17 | 7 | 734.13 | 778.16 | 6 | 823.23 | 0 | 823.23 | 888.52 |
| 1 | 80 | 85.17 | 7 | 734.13 | 778.16 | 6 | 823.23 | 0 | 823.23 | 888.52 |
| 1 | 80 | 86.37 | 7 | 778.16 | 797.86 | 4 | 847.94 | 0 | 847.94 | 888.52 |
| 1 | 80 | 86.37 | 7 | 778.16 | 797.86 | 4 | 847.94 | 0 | 847.94 | 888.52 |
| 1 | 70 | 77.44 | 8 | 678.09 | 748.15 | 5 | 874.94 | 0 | 874.94 | 888.52 |
| 1 | 90 | 101.77 | 7 | 804.86 | 830.41 | 6 | 888.52 | 0 | 888.52 | 888.52 |

## BIBLIOGRAPHY

[1] Cruz-Neira, C., Virtual Reality Overview, *ACM SIGGRAPH '93 Notes: Applied Virtual Reality*, ACM SIGGRAPH '93 Conference, Anaheim, California, August 1-6, 1993.

[2] Industrial News, *VR News*, vol. 6/no. 6, 1997, 4.

[3] North, M., North, S., Coble, J., Effectiveness of Virtual Environment Desensitization in the Treatment of Agoraphobia, *Presence*, vol. 5/ no. 3, Summer 1996, 346-352.

[4] Industrial News, *VR News*, vol. 6/no. 5, 1997, 8.

[5] Bliss, J., Tidwell, P., Guest, M., The Effectiveness of Virtual Reality for Administrating Spatial Navigation to Firefighters, *Presence*, vol. 6/ no. 1, February 1997, 73-86.

[6] Pritsker, A. B., O'Reilly, J., and LaVal, D., *Simulation With Visual SLAM and AweSim,* Systems Publishing Corporation, West Lafayette, Indiana, 1997.

[7] Nance, R. E., Simulation Programming Languages: An Abridged History, *Proceedings of the 1995 Winter Simulation Conference,* 1995.

[8] Banks, J., Software Formulation, *Proceedings of the 1995 Winter Simulation Conference,* 1995.

[9] Jenkins, B., Virtual Manufacturing: The New Frontier, *Computer Graphics World*, March 1998, 23-24.

[10] Seila, A., Introduction to Simulation, *Proceedings of the 1995 Winter Simulation Conference,* 1995.

[11] QUEST simulation software, Deneb Robotics, Inc, Auburn Hills, Michigan, 1997.

[12] Sense8 Corporation, *WorldToolKit Reference Manual Release 6*, Mill Valley, CA, 1996.

[13] Melaas, L., Simulation and Analysis of a Flexible Manufacturing Cell, Master's Thesis, Iowa State University, Ames, IA, 1998.

[14] Chen, S. E., QuikTime® VR-An Image Based Approach to Virtual Environment Navigation, *Computer Graphics: Proceedings of SIGGRAPH '95*, Los Angeles, CA, August 1995.

[15] Fleischer, K., Laidlow, D., Currin, B., and Barr, A., Cellular Texture Generation, *Computer Graphics: Proceedings of SIGGRAPH '95*, Los Angeles, CA, August, 1995.