The Virtual Workshop: A Simulated Environment for Mechanical Design

by

John Warren Barrus

M.S., Mechanical Engineering
Massachusetts Institute of Technology (1987)

B.S., Mechanical Engineering
Massachusetts Institute of Technology (1987)

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 1993

© 1993 Massachusetts Institute of Technology.

Signature of Author: 

Department of Mechanical Engineering
July 1, 1993

Certified by: Woodie Flowers
School of Engineering Professor of Teaching Innovation
Department of Mechanical Engineering
Thesis Committee Chair

Accepted by: 

Ain Sonin
Chairman, Departmental Graduate Committee
The Virtual Workshop:  
A Simulated Environment for Mechanical Design  

by  
John W. Barrus  

Submitted to the Department of Mechanical Engineering  
on July 1, 1993 in partial fulfillment of the  
requirements for the degree of  
Doctor of Philosophy in Mechanical Engineering  

Abstract  

Often, after a part is designed by an engineer, it is redesigned by the manufacturer because the part is viewed differently from functional and manufacturing perspectives. The objective of this research is to begin to demonstrate that using a manufacturing-process-based CAD system in the initial design of parts improves the design path and reduces the number of design iterations. A process-based CAD system will bring a designer closer to a final design, without actually making the part, by simulating the processes used to build it. It will allow the designer to resolve both functional and manufacturing constraints during early design phases. This approach to design should uncover weaknesses and reduce the number of fundamental errors earlier in a design cycle. 

A Virtual Workshop was created on a computer system in MIT’s CADLab. For this thesis, a limited number of machines and processes (such as measuring, marking, fixtureing, cutting and assembly) were implemented. The extent of the process simulation was broad enough to reveal some manufacturing limitations, but is not complete. For example, geometric constraints were accounted for in the simulation, such as preventing intersections of solid parts, but elastic behavior of the designed parts was not simulated. 

After the implementation was completed, the workshop was demonstrated to both experienced and inexperienced designers, machinists, and mechanical engineers. The usefulness of the workshop metaphor as a design environment was investigated. Also, the usability of the computer interfaces of the machines and processes were explored. 

Thesis Supervisor:    Prof. Woodie Flowers  
Title:    School of Engineering Professor of Teaching Innovation
Thesis Committee

Chair

Prof. Woodie Flowers
School of Engineering Professor of Teaching Innovation
Department of Mechanical Engineering
Massachusetts Institute of Technology

Members

Prof. David C. Gossard
Professor
Department of Mechanical Engineering
Massachusetts Institute of Technology

Dr. Donald L. Grimes
Project Manager
Polaroid Corporation

Prof. Harry West
Associate Professor
Department of Mechanical Engineering
Massachusetts Institute of Technology

Dr. David Zeltzer
Principal Research Scientist
Research Lab of Electronics
Massachusetts Institute of Technology
Dedication

To Laura, my best friend.
Acknowledgments

Laura Macfarlane – Your generosity in taking care of Michael and Katie for countless weeks and months on end did not go unnoticed. Your kindness and incredible sacrifice will never be forgotten. Thank you for your willingness to listen to my struggles and for your always appropriate advice. The depth of your insight continues to amaze me.

Prof. Woodie Flowers – Thank you for your vision, funding, and long-suffering. It was an honor to be able to look to you as a mentor.

Prof. David Gossard – Thank you for reducing the estimated time to completion for my thesis by arranging for the geometric modeling software and also for your advice throughout my Ph.D. program.

Prof. Mark Jakiela – Thank you for providing the hardware necessary to make my gluttonous thesis software run at interactive speeds.

Doctoral Committee (Prof. Woodie Flowers, Prof. David Gossard, Dr. Donald Grimes, Prof. Harry West, and Dr. David Zeltzer) – Thank you for your encouragement, support and guidance throughout the entire process.

Don Grimes – Thank you for your example and leadership. You helped me to focus my work. Your unselfish encouragement and direction provided a much needed boost on a very long journey.

Barbara Balents and David Wallace – You were the constants in an ever-changing lab. Sorry to bail out on you, but not too sorry. It would have been a lot worse without you.

I also wish to acknowledge the Leaders for Manufacturing Program for its support of this work.
# Table of Contents

## Introduction
- Drafting, Designing, or Making? ................................................. 18
- Description of the Virtual Workshop (Illustration of the Environment) 20
- Interactions .................................................................................... 25
- Why a Virtual Workshop? ............................................................... 27
- Motivation ....................................................................................... 27
- Objectives ....................................................................................... 31

## Virtual Workshop Background and Description
- Background and Prior Art .............................................................. 36
  - Design by Process ....................................................................... 36
  - Swept Volumes ........................................................................... 37
  - Simulation .................................................................................... 38
  - Simultaneous Engineering ......................................................... 39
  - Direct Manipulation .................................................................. 39
  - Stereoscopics ............................................................................. 39
  - Solid Modeling ........................................................................... 39
  - Features ...................................................................................... 40
  - Tolerances .................................................................................. 40
- Overview ....................................................................................... 42
- Workshop Machines ...................................................................... 50

## Interaction and Vocabulary
- Material Selection ......................................................................... 55
  - Access ......................................................................................... 55
- Measuring and Marking .................................................................. 58
  - Overview .................................................................................... 58
  - Access ....................................................................................... 58
- Using the Virtual Machines .......................................................... 60
  - Machine Adjustments .................................................................. 60
  - Blade Selection ........................................................................... 61

## Representation and Implementation
- Virtual Workshop Objects ............................................................. 64
  - Overview .................................................................................... 64
  - Text-based Representation ......................................................... 65
- Solid Models .................................................................................. 68
  - Representation ........................................................................... 68
  - Manipulating the Models ............................................................ 70
  - Use of Solids in Virtual Workshop ........................................... 71
- Machine Representation and Control ............................................ 73
  - Data Structure ............................................................................ 73
  - Representing Machine Adjustments ........................................... 75
  - Constraints and Locking ............................................................ 78
  - Machine-Workpiece Interaction ................................................. 88
- Part Representation ......................................................................... 92
  - Data Structure ............................................................................ 92
- Lock Representation ....................................................................... 97
- Blade Representation ..................................................................... 99
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.7 Stock Representation</td>
<td>109</td>
</tr>
<tr>
<td>4.8 Workpiece Representation</td>
<td>110</td>
</tr>
<tr>
<td>Assembling Workpieces</td>
<td>113</td>
</tr>
<tr>
<td>4.9 Project Representation</td>
<td>114</td>
</tr>
<tr>
<td>4.10 Journals and Rollback</td>
<td>115</td>
</tr>
<tr>
<td>4.11 Mark Representation</td>
<td>116</td>
</tr>
<tr>
<td><strong>Evaluation</strong></td>
<td>119</td>
</tr>
<tr>
<td>5.1 Design Problem Survey</td>
<td>120</td>
</tr>
<tr>
<td>5.2 Demonstrations</td>
<td>124</td>
</tr>
<tr>
<td>Demonstration Objective</td>
<td>124</td>
</tr>
<tr>
<td>5.3 Response</td>
<td>125</td>
</tr>
<tr>
<td><strong>Conclusion</strong></td>
<td>127</td>
</tr>
<tr>
<td>6.1 Demonstration Results</td>
<td>128</td>
</tr>
<tr>
<td>Disadvantages</td>
<td>128</td>
</tr>
<tr>
<td>Advantages</td>
<td>129</td>
</tr>
<tr>
<td>6.2 Future Research</td>
<td>130</td>
</tr>
<tr>
<td>Journals and Playback</td>
<td>130</td>
</tr>
<tr>
<td>Workshop Shortcuts</td>
<td>130</td>
</tr>
<tr>
<td>Sheet Metal Brakes and Wire Jigs</td>
<td>130</td>
</tr>
<tr>
<td>Tolerances</td>
<td>130</td>
</tr>
<tr>
<td>Machine Configuration</td>
<td>131</td>
</tr>
<tr>
<td>Path Planning for Numerically Controlled Machines</td>
<td>131</td>
</tr>
<tr>
<td>Sound</td>
<td>131</td>
</tr>
<tr>
<td><strong>Example Input Files</strong></td>
<td>133</td>
</tr>
<tr>
<td>A.1 Machine Input File</td>
<td>134</td>
</tr>
<tr>
<td>A.2 Blade Input Files</td>
<td>140</td>
</tr>
<tr>
<td><strong>Hardware and Software Platform</strong></td>
<td>141</td>
</tr>
<tr>
<td><strong>Bibliography</strong></td>
<td>143</td>
</tr>
</tbody>
</table>
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Example of stock menu</td>
<td>20</td>
</tr>
<tr>
<td>1.2</td>
<td>Example bookshelf design</td>
<td>21</td>
</tr>
<tr>
<td>1.3</td>
<td>Ripping fence locking handle</td>
<td>23</td>
</tr>
<tr>
<td>1.4</td>
<td>Front and side of table saw</td>
<td>25</td>
</tr>
<tr>
<td>1.5</td>
<td>Example pocket design</td>
<td>27</td>
</tr>
<tr>
<td>1.6</td>
<td>Pocket with rounded corners</td>
<td>28</td>
</tr>
<tr>
<td>2.1</td>
<td>Main window of the Virtual Workshop</td>
<td>42</td>
</tr>
<tr>
<td>2.2</td>
<td>Drill Press with drill bit and stock</td>
<td>43</td>
</tr>
<tr>
<td>2.3</td>
<td>Drill press with locked (a) and unlocked (b) table</td>
<td>44</td>
</tr>
<tr>
<td>2.4</td>
<td>Drill press before (a) and during (b) the creation of a hole</td>
<td>44</td>
</tr>
<tr>
<td>2.5</td>
<td>Main menu of Virtual Workshop</td>
<td>45</td>
</tr>
<tr>
<td>2.6</td>
<td>Table saw cutting workpiece in half (setup)</td>
<td>46</td>
</tr>
<tr>
<td>2.7</td>
<td>Table saw cutting workpiece in half (partial cut)</td>
<td>47</td>
</tr>
<tr>
<td>2.8</td>
<td>Table saw cutting workpiece in half (finish)</td>
<td>48</td>
</tr>
<tr>
<td>2.9</td>
<td>Simple wood assembly (dado joint) on workbench</td>
<td>49</td>
</tr>
<tr>
<td>2.10</td>
<td>Band saw</td>
<td>50</td>
</tr>
<tr>
<td>2.11</td>
<td>Drill press</td>
<td>51</td>
</tr>
<tr>
<td>2.12</td>
<td>Milling machine</td>
<td>51</td>
</tr>
<tr>
<td>2.13</td>
<td>Radial arm saw</td>
<td>52</td>
</tr>
<tr>
<td>2.14</td>
<td>Table saw</td>
<td>52</td>
</tr>
<tr>
<td>3.1</td>
<td>Furniture wood dialog box</td>
<td>56</td>
</tr>
<tr>
<td>3.2</td>
<td>Motion of the table on the drill press</td>
<td>60</td>
</tr>
<tr>
<td>4.1</td>
<td>Directory layout</td>
<td>64</td>
</tr>
<tr>
<td>4.2</td>
<td>Example of polygonal shaft in polygonal hole</td>
<td>68</td>
</tr>
<tr>
<td>4.3</td>
<td>Body being separated by cutting operation</td>
<td>69</td>
</tr>
<tr>
<td>4.4</td>
<td>Diagram of a machine</td>
<td>76</td>
</tr>
<tr>
<td>4.5</td>
<td>Rotation of the band saw table</td>
<td>77</td>
</tr>
<tr>
<td>4.6</td>
<td>Detents (snapTo values) and snap resolution</td>
<td>80</td>
</tr>
<tr>
<td>4.7</td>
<td>Part with linear constraint being dragged with the mouse</td>
<td>81</td>
</tr>
<tr>
<td>4.8</td>
<td>Rotation plane for selected part with rotational constraint</td>
<td>83</td>
</tr>
<tr>
<td>4.9</td>
<td>Discrepancy between expected and actual rotation angles</td>
<td>86</td>
</tr>
<tr>
<td>4.10</td>
<td>Motion of a part with a planar constraint</td>
<td>87</td>
</tr>
<tr>
<td>4.11</td>
<td>Intersection (a) and bounding boxes (b) of blade and workpiece</td>
<td>90</td>
</tr>
<tr>
<td>4.12</td>
<td>Blade swept in arc intersects workpiece</td>
<td>90</td>
</tr>
<tr>
<td>4.13</td>
<td>Diagram showing blade geometry and profiles for flat end mill</td>
<td>103</td>
</tr>
<tr>
<td>5.1</td>
<td>Problem statement for design experiments</td>
<td>121</td>
</tr>
<tr>
<td>5.2</td>
<td>Inexperienced designers solution to card holder</td>
<td>122</td>
</tr>
<tr>
<td>5.3</td>
<td>Experienced designers sketch of card holder</td>
<td>122</td>
</tr>
</tbody>
</table>
# List of Tables

Table 1.1  Excerpt from surface finish table ........................................... 30  
Table 1.2  Design issues and solutions .................................................... 31-33  
Table 3.1  Types of marks that can be made on workpieces ....................... 58  
Table 3.2  Creation of marks in the Virtual Workshop ................................. 59  
Table 4.1  Contents of the Virtual Workshop directories .............................. 65  
Table 4.2  Information stored in the machine ............................................ 73  
Table 4.3  Constraint types and required data .......................................... 78  
Table 4.4  Symbols used in calculation of rotations ................................... 83  
Table 4.5  Information stored in the part .................................................. 92  
Table 4.6  Description of lock data .......................................................... 97  
Table 4.7  Information contained in a blade ............................................. 99  
Table 4.8  Rules for selecting profiles from blades .................................... 104  
Table 4.9  Example data from blade file .................................................. 106  
Table 4.10 Description of data found in blade input file .............................. 106  
Table 4.11 Description of information specific to stock in Virtual Workshop .... 109  
Table 4.12 Information stored with each workpiece ................................... 110  
Table 4.13 Information stored in a project ............................................... 114  
Table 5.1  Differences in approach between designers ................................. 123  
Table 5.2  Selected comments from survey .............................................. 125-126
As computer-aided drafting (CAD) systems were developed, they were designed to replace the traditional drafting board and automate the drawing production process. Computers allowed drawings to be modified easily and information could be reused without redrawing it.

Unfortunately, the drafting board metaphor only enhances a part of the design process because drawings are only a small part of the overall design activity. Drawings are used to communicate the designers' intent to the manufacturer. In order to communicate that intent, they incorporate geometric information, tolerance information, and notes about materials and processes. This information is generated and captured using processes unrelated to manufacturing.

Drawing is quite different from design. In an effort to bring the design process closer to manufacturing, a building-metaphor design system, the Virtual Workshop, has been created and evaluated. The Virtual Workshop is discussed in this document.
1.1 Drafting, Designing, or Making?

With the increased speed and graphics capability of computers and the development of new solid modeling and conceptual engineering systems, computers can now be used earlier in the design process. Engineers are recognizing the value of getting the design information into the computer at a much earlier stage of the design. Designers are using the computer not just for documentation, but for analysis, visualization, assembly testing and many other things. Conceptual design tools have been created for trying out several design alternatives in the early stages of design. There are many computer programs that allow a designer to specify the outer form of a design and assign textures and colors for rendering and displaying design alternatives.

This document describes a Virtual Workshop which has been created to allow designers to create parts and assemblies using a representation of a workshop. The Virtual Workshop is primarily graphical and allows designers to work as if they are in a real workshop, using the same tools and equipment as those found in a workshop.

The Virtual Workshop is used differently than a typical CAD program. Instead of drawing and sketching on an electronic drafting board, the designer works with parts and machines, as if in a real workshop. The designer selects stock from a list of available shapes and sizes of different materials. A machine and blade are then selected for machining the stock. The designer adjusts the equipment for the appropriate machining operation and then cuts or shapes the stock. The designer can work on several pieces of stock, or workpieces at once while working on a project and can assemble them when they are fabricated.

The machines that were created in this implementation of the Virtual Workshop include a milling machine, drill press, table saw, band saw and radial arm saw. As in a real workshop, blades that can be used in several different machines are interchangeable. For instance, a blade that fits in the table saw can also be used in the radial arm saw.

One of the motivations for using a workshop metaphor as a CAD system interface is to reduce the number of "languages" that must be used in a design environment. A designer familiar with a prototyping environment might be able to learn and use the Virtual Workshop using the same vocabulary of operations and tools as those with which she is already familiar.
The Virtual Workshop provides several advantages to the designer. It provides an environment to easily try out several design alternatives at a low cost. An undo feature allows the designer to try operations without fear of destroying the part.

Operations that are not possible in a real workshop are not available in the simulation of the workshop. Bringing the design constraints closer to the process constraints drives the designer to think intimately about the manufacturing ramifications of design decisions earlier in the design process. Also, an inexperienced designer can become familiar with operations and machines available for processing materials.

In design, there has been a long history of separation between design and manufacturing. Many products, even today, are designed and then "thrown over the wall" to the manufacturing department for production. This leads to longer product development cycles. Today, there is interest in shortening the product development cycle by bringing design and manufacturing together.

One of the effects of separating design and manufacture is to develop a method of communicating that specifies design intent without necessarily favoring one manufacturing operation over another. For instance, a part is designed and drawn. On the drawing of the part, not only are the overall part dimensions specified, but also the tolerances on those dimensions. The tolerances tell the person making the part how closely each of the surfaces must be to the ideal geometry and still function as requested.

Separating the design intent from the manufacturing process is useful in some circumstances, but also creates a communication barrier. It also brings the designers farther from the manufacturing process.
1.2 Description of the Virtual Workshop

Illustration of the Environment

If a designer was interested in building a bookshelf in the Virtual Workshop she would first select some wood. Wood is currently available through a pull-down menu entitled "Stock". The types and sizes of available wood are modeled after a local lumber yard. The dialog box that allows selection of the wood displays size (length, width, and thickness), weight (based on the current dimensions), and cost. The weight and cost of the wood are calculated from the density and cost information stored in the stock database. For this bookshelf project, the designer might choose some yellow pine, 4/4 thickness, 12 inch width. The length selected would depend on the design of the bookshelf.

![Figure 1.1 Example of stock menu.](image)

It should be apparent at this point that designers generally will not sit down at the Virtual Workshop with no idea about what they are trying to make. Some thought needs to go into the overall dimensions and appearance of the design. For instance, the gross dimensions of the bookshelf, its height, length, and depth must be decided before using the system.

However, since there is little cost in throwing away parts that have already been created (as opposed to the real world, where stock is quite expensive) it is not absolutely necessary that all of the details be decided in advance.
When the appropriate wood has been selected, the designer might begin by cutting the shelves to the right length using either the table saw or the radial arm saw.

When a machine like the table saw is selected, the part automatically moves to the workplanes associated with that machine. For instance, the table saw, with no jigs, fences, or fixtures, has a workplane with a normal in the upward vertical direction\(^1\) directly above the arbor and blade. The workpiece is free to move anywhere in the plane of the top of the table. If the ripping fence is in use, another workplane, fixed to the side of the ripping fence, constrains the workpiece. The combination of the two workplanes constrains the

---

\(^1\) In the Virtual Workshop, the positive Z direction is always pointing up so that the designer will never see a machine on its side or upside down.

Section 1.2 Description of the Virtual Workshop
part to move in a straight line along the ripping fence and the top of the table. The addition of a third constraining workplane would completely constrain the workpiece from independent movement. The workpiece could then only be moved by moving the fence or jig that contains the workplane.

Workpieces are cut by pushing them over the workplane past the cutting edge of the blade. As the blade intersects the workpiece, material is removed and if the blade passes completely through and cleaves the workpiece, two new workpieces are created. The material removed from the workpiece due to the thickness of the blade (called the kerf) is no longer part of either piece.

Often, when cutting shelves, it is far more important to cut them all the same width than it is to get them to be precisely the correct width. For example, assume that the shelves were supposed to be 10 inches deep. Depending on how the table saw and ripping fence are set up, the shelves may be cut with a tolerance of ±1/4 in. However, multiple cuts done in sequence with the same setup on the table saw will have a much tighter tolerance (measured in 1/100ths of an inch). Therefore, if the shelves are cut at different times using different table saw setups, the shelves are much more likely to differ in width. Cut at the same time, the shelves will be very close to the same size. In fact, model makers and wood workers generally cut same dimensions at the same time to avoid the problem of variations in critical dimensions. An interesting point to note is that the actual length of the shelves may not be very critical. In many situations, variations in the length of a bookshelf by as much as an inch may not be important. However, when it comes time to assemble the bookshelf, if each shelf dimension varied by more than an inch, it would be impossible to assemble it correctly.

After the designer has selected the table saw and chosen the fixtures and jigs appropriate for cutting the shelves of the bookshelf, a blade must be selected. The blades in the Virtual Workshop are not specific to a machine. Any machine that has an arbor of the same dimensions as the arbor on the table saw can use the same blades. For instance, a table saw, radial arm saw, miter saw, and circular saw that all have 5/8 inch arbors will accept blades that are made for a 5/8 inch arbor. The designer might select a combination blade for the table saw for cutting the shelves to the correct length, and then a 3/4 inch dado blade for cutting the dadoses to hold the shelves.

The combination blade would be used first for ripping the shelves to the correct width, cutting the shelves to the correct length and for cutting the sides of the bookcase.
When the blade is selected, the fences need to be adjusted to the correct position for cutting the shelves. The ripping fence should be adjusted so that the distance between the blade and the fence matches the width of the shelves. In the virtual workshop, the fence is adjusted simply by clicking with the left mouse button on the fence and dragging the fence into the correct position. The machine part location indicator (available from the view menu) displays the position of the machine part (in this case the ripping fence) on the screen as the part is dragged into position. This allows more accurate positioning of the fence than simple visual approximation. Once the ripping fence is located, it can be locked into place so that inadvertent mouse clicks will not reposition it accidentally. Locking is achieved by clicking with the left mouse button on the locking arm located on the front of the fence.

![Figure 1.3 Ripping fence locking handle (shown on the left in the unlocked position and on the right in the locked position).](image)

The lock on the fence moves from the unlocked position to the locked position when the lock is clicked. When the lock is in the locked position, the fence no longer moves when it is selected and dragged.

After the stock has been cut to the correct width, the crosscut fence can be used to cut the shelves to the proper length.

If the shelf length has been designed at 30 inches and the stock is 10 feet in length, four pieces can be cut from the stock. However, not all four pieces will be 30 inches long because of the blade kerf. With a 1/16 inch thick blade, the fourth piece will be only 29 and 13/16ths of an inch wide. For a naive user the difference may not be noticeable until assembly. An experienced designer would either use the last piece for a different part of
the bookshelf or adjust the position of the ripping fence to allow for the kerf when cutting up the board.

In this example, the bookshelf only requires 4 shelves, so the fourth piece will be used for one of the end pieces. The fourth shelf and a second end piece will be cut from another piece of stock.

When cutting the dadoses in the end pieces, a dado blade must be chosen that matches the thickness of the boards. A one inch thick dado blade might be selected for cutting the slots to support the shelves. However, stock that has already been dressed (cut to a more precise finish) is no longer one inch thick. A 4/4 (one inch) board is typically only 3/4 of an inch thick. Therefore, the appropriate dado blade would be 3/4 inch thick. The stock that is selected from the workshop menus matches the stock that you get from a lumber yard. Therefore, the designer can use the measuring tools in the Virtual Workshop to measure the thickness of the board before deciding on which blade to use.

After the overall dimensions of the shelves and end pieces have been cut, it is necessary to put the dadoses in the end pieces to support the shelf. If the designer wants 10 inches of space between each shelf and the boards are approximately 3/4 inch thick, the bottom of the lower shelf will be approximately 1 3/4 inches from the floor.

Once the new blade has been chosen, the blade must be lowered to the correct height for the dado cuts. The blade can be raised and lowered by turning a crank on the front of the base of the table saw (see Figure 1.4). Setting the cut depth once before creating all of the dadoses ensures accurate and consistent depths for all of the dadoses. One way to measure the height of the blade (or the depth of the cut) is simply to look at the machine part location indicator as the blade adjuster is moved. The machine part location indicator is a dialog box that indicates the position and limits of the current part being moved in a machine.
Figure 1.4 Front and side of table saw showing the cranks used for adjusting the height and angle of the blade.

After the dadoes have been cut, it is time to assemble the bookshelf. Finishing machines and finishing operations have not yet been built into the Virtual Workshop. Many finishing operations, like varnishing or painting, simply change the appearance or texture of a workpiece and add almost no thickness to the surface. Other operations actually remove or add material in significant amounts that they must be accounted for in the solid model. Although approaches for these operations and their representation have been devised, they are not currently implemented.

Minor assembly operations can be done in the Virtual Workshop. Faces can be aligned and, once positioned appropriately, the workpieces can be fixed to other workpieces. This allows visual interference and alignment checking.

**Interactions**

Almost all of the interaction with the Virtual Workshop is done using the left mouse button. Clicks activate menu items, buttons, and the locks on machine parts. Clicking and
dragging moves parts of the environment or makes marks on the workpieces, depending on which menu items have been selected.

One of the limitations of the Virtual Workshop as it is now implemented is that most of the operations and orientations are limited to 3D linear motions. Working in an unconstrained 6 DOF environment was found to be unmanageable. Workpieces would not align correctly. Moving the workpiece in a specific manner or direction was very difficult. The addition of motion constraints made it much easier to control the direction of motion and the placement of the workpieces in the machines.

All motions are limited by constraints. A machine part with a linear or rotational constraint is only allowed to move in the direction specified by the constraint and within the bounds of that constraint. For instance, the angle of the table on the bandsaw can be adjusted between 0 and 45 degrees for making angled cuts on workpieces (see Figure 4.5 on page 77). The constraint on the part specifies the axis of rotation (a normal and a point) and the limits to the rotation (0 to 45 degrees). A linear constraint, such as the ripping fence on the table saw, consists of a normal specifying the direction of motion and limits to the motion.

Allowable workpiece motions depend on which machine is being used and what fences and fixtures are selected. If only one workplane is active on a machine (like the surface of the table saw) then the workpiece can move anywhere in that plane. When the designer clicks and drags the workpiece, the workpiece is moved so that the initially selected point on the workpiece corresponds to the new mouse position. When a second workplane is activated (by turning on the ripping fence or the crosscut fence) the workpiece is constrained to move in a linear direction corresponding to the cross product of the normals of the two workplanes. Allowing motion only in this direction ensures that the workpiece will stay pressed up against the two constraining workplanes. Once a third workplane is added, only motion of the machine parts will drive the workpiece, because the workpiece is fully constrained.

Rotations of each workpiece are allowed around a single axis at a time and are independent of the machine workplanes. The workpiece can either be rotated 90 degrees at a time in either the X or Y axis (a combination of which would provide rotations around the Z axis) or it can be moved incrementally around an axis when attached to a machine with a rotational constraint on one of the machines parts.
1.3 Why a Virtual Workshop?

Motivation

Constraint Correlation

One of the characteristics of current CAD systems is that the operations in the CAD system often do not correspond to real life manufacturing processes. Some CAD programs have included feature-based menus for over 10 years, but the systems are still primarily additive, requiring the designer to slowly build up a design, perhaps starting with a profile that is then swept to create a part. Boolean operations of solids now allow subtractive CAD systems, but often the Boolean primitives are based on geometric primitives, like cubes, spheres and tori, instead of on real materials, tools, and blades. As a result, parts that are simple to create in a CAD system can be hard to create in a shop.

For example, if a designer wanted to make a block containing a pocket, the top view could be drawn using a rectangle to define the outside of the block, and another rectangle to define the inside. The depth of the pocket could be specified in writing, or the designer could quickly sketch a side view using two more rectangles (see Figure 1.5).

Unfortunately, making a block with a pocket containing very small radii on the corners is very difficult. Special machinery and tools are required (plunge EDM, for example) and the small radii may not even be an important design requirement.

Figure 1.5 Example pocket design.
A more easily manufactured design would have a pocket with rounded corners (see Figure 1.6). This pocket could easily be made using a milling machine which is found in most workshops.

In a CAD system using a manufacturing metaphor, the second design shown (with the rounded corners) is actually significantly easier to design because the design constraints are more closely matched to the manufacturing constraints. The designer does not have to obtain a second body of knowledge about manufacturing that is separate from understanding the interface to the CAD system. In fact, using a CAD system with the same "vocabulary" as the workshop where the part will be made will reduce the amount of learning required to master the design process. Knowledge gained from using the CAD system will support and reinforce knowledge obtained from experience in the workshop. Someone familiar with either the CAD program or the workshop will have a head start in learning about the other area.

**Coming Closer to the Final Design**

Another advantage to the workshop metaphor in a CAD system is the ability to test out designs before committing them to real hardware. For instance, solid CAD systems are significantly better for testing assemblies because of their ability to test for part intersections. In the same manner, a system that simulates manufacturing practices allows the designer to work out some of the process details and think about how a design will be made and assembled in advance.

Conversations with the director of the hobby shop at MIT and the model makers at Polaroid Corporation indicate that drawn or sketched designs brought in by engineers from MIT and Polaroid do not consider the processes necessary to produce them. The part must be redesigned with help from the people that actually make it in order to reduce the expense or
the amount of work required. Often, new engineers are so inexperienced with actually making things that their designs have to be reworked completely based on advice from an experienced machinist or model maker. More experienced designers have less trouble with their designs because of their knowledge of how things are made.

**Communication**

Communicating design intent is difficult with drawings. In fact, many 2-dimensional drawings and 3-dimensional wire frame models are almost impossible to understand and interpret because of the complexity of their geometry. Sending electronic versions of parts, including solid models and 2-D views, has allowed computerized workshops to interpret drawings using the computer to query and retrieve information about the design. More information is passed to the workshop so more information is available to the model maker.

Imagine not only passing information about the geometry of the model to the model maker but also suggested process information. If the model maker could not only retrieve geometric information but also find out which processes were requested to create each surface or feature the model maker would have a better understanding of the design requirements.

For instance, in order to specify surface quality and accuracy, the designer should understand process limitations and capabilities. Using a process metaphor CAD system, the designer could create parts and then query the surfaces and features about their accuracy and surface finish (assuming that the proper information was contained in the CAD system.) A designer using such a system would quickly become familiar with process capabilities and would be gaining an understanding of the processes that only comes from actually making things in a workshop. Arbitrary tolerances called out in drawings would not be necessary because the tolerance would be a side product of the process used to create the part.

Although the capability of specifying surface finishes and tolerances for a given process is not yet built into the Virtual Workshop, the process information necessary for attaching such information to the surfaces of designed parts is readily available and may be included in a future version of the workshop. An example of the information that could be available to the designer from the Virtual Workshop appears in [Jensen 79] on page 83, Figure 589. Table 1.1 is an excerpt from that figure.
<table>
<thead>
<tr>
<th>Rating</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>25.2</td>
<td>Rough, low grade surface resulting from sand casting, torch or saw cutting, chipping or rough forging. Machine operations are not required as appearance is not objectionable. This surface, rarely specified, is suitable for unmachined clearance areas on rough construction items.</td>
</tr>
<tr>
<td>6.3</td>
<td>Course production surfaces, for unimportant clearance and cleanup operations resulting from coarse surface grind, rough file, disc grind, rapid feeds in turning, milling, shaping, drilling, boring, grinding, etc., where tool marks are not objectionable. The natural surfaces of forgings, permanent mold castings, extrusions, and rolled surfaces also produce this roughness. It can be produced economically and is used on parts where stress requirements, appearance, and conditions of operations and designs permit.</td>
</tr>
<tr>
<td>1.6</td>
<td>A good machine finish produced under controlled conditions using relatively high speeds and fine feeds to take light cuts with sharp cutters. It may be specified for close fits and used for all stressed parts, except fast rotating shafts, axles, and parts subject to severe vibration or extreme tension. It is satisfactory for bearing surfaces when motion is slow and loads light or infrequent. It may also be obtained on extrusions, rolled surfaces, die castings and permanent mold castings when rigidly controlled.</td>
</tr>
<tr>
<td>0.2</td>
<td>A fine surface produced by honing, lapping, or buffing. It is specified where packings and rings must slide across the direction of the surface grain, maintaining or withstanding pressures, or for interior honed surfaces of hydraulic cylinders. It may also be required in precision gages and instrument work, or sensitive value surfaces, or on rapidly rotating shafts and on bearings where lubrication is not dependable.</td>
</tr>
<tr>
<td>0.05</td>
<td>Costly refined surfaces produced only by the finest of modern honing, lapping, buffing, and superfinishing equipment. These surfaces may have a satin or highly polished appearance depending on the finishing operation and material. These surfaces are specified only when design requirements make it mandatory. They are specified on fine or sensitive instrument parts or other laboratory items, and certain gage surfaces, such as on precision gage blocks.</td>
</tr>
<tr>
<td>0.025</td>
<td></td>
</tr>
</tbody>
</table>

Table 1.1 Excerpt from surface finish table in *Engineering Drawing and Design* by Jensen and Helsel [Jensen 79].

**Education**

A fourth important objective of creating a workshop metaphor CAD system concerns education and training. Many new mechanical engineers are not familiar with the shop and its equipment. Gaining experience is both time consuming and expensive. Even a small project can quickly become expensive, requiring money for materials, tools, and hardware. Having a simulation of the workshop available for use by designers would allow them to
try out design ideas and work out the difficulties in their projects before spending any money.

There is a danger of an inexperienced hobbyist or engineer removing parts of their bodies while using an extremely sharp, high-speed, spinning blade or bit. Designers new to the workshop can practice using the machines in a completely safe environment without danger to either themselves or those around them. Of course, experience in the Virtual Workshop will never compensate for rigorous safety training with the real machines, but it will allow the designer to become familiar with the machines and how they work before they experiment using the real equipment.

**Objectives**

The objective of this research project is to design and construct a Virtual Workshop that would bring a designer closer to the final design of an assembly on a computer and would be easy to learn and use. This workshop could be used as a research tool to determine how using a workshop metaphor as the interface to a CAD system changes the way design is accomplished.

**Design of the Workshop**

The Virtual Workshop was designed to be an extensible, easy to use design system. It was modeled after a real workshop at MIT with a focus on realistic interaction and visual appearance. Several of the issues around the design of the Virtual Workshop are listed below with a brief description of the implementation approach.

<table>
<thead>
<tr>
<th>Design Issue</th>
<th>Implementation Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workpiece representation</td>
<td>Material and workpieces are represented as solid models (using the ACIS solid modeling kernel), along with surface based attributes (material, etc.)</td>
</tr>
<tr>
<td>Stock representation</td>
<td>Stock is modeled after stock available in local hardware and lumber stores. For instance, a 2x4 is really 1 3/4 by 3 3/4.</td>
</tr>
<tr>
<td>Measurement</td>
<td>Points, edges, or surfaces are selected using the mouse and distances are calculated and displayed.</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Marking</td>
<td>Marking is accomplished by creating wires that are displayed as part of the solid model. Circles, lines, and polygons are available and can be created on the surface of any workpiece.</td>
</tr>
<tr>
<td>Machines</td>
<td>Machines are represented as a tree of constrained machine parts. For instance, a part with a linear constraint, such as the fence on a table saw, can only move in a straight line.</td>
</tr>
<tr>
<td>Machine parts</td>
<td>Each part can be constrained and locked or unlocked. A locked part cannot be moved, but when unlocked, it can move only according to and within the bounds of its constraint.</td>
</tr>
<tr>
<td>Constraints</td>
<td>Constraints can be linear, planar, rotational, or fixed. They also have bounds, so a linear constraint only allows motion in a straight line between two points.</td>
</tr>
<tr>
<td>Interaction with machine</td>
<td>Interaction is primarily with a mouse. If the pointer is clicked on a part and dragged, the part moves with the pointer. If the pointer is clicked on a part's lock, the lock is toggled either on or off.</td>
</tr>
<tr>
<td>Workpiece assembly</td>
<td>Workpieces can be assembled using face to face constraints.</td>
</tr>
<tr>
<td>Machine design flexibility</td>
<td>Machines, parts, and blades are all represented in text files. The text is readable and easily modified to change the way the machine works.</td>
</tr>
<tr>
<td>Jigs, fixtures, and fences</td>
<td>Jigs, fixtures, and fences are all represented as parts on the machine and can be turned on or off for different operations.</td>
</tr>
<tr>
<td>Blade, bit, and cutter representation</td>
<td>Blades are represented by a solid model and a series of constrained profiles. Each profile can be swept along a path to generate a cutting volume.</td>
</tr>
<tr>
<td>Blade constraints</td>
<td>Blade constraints are either linear, curvilinear or planar. A drill bit is an example of a linearly constrained blade. Milling bits are planar and linear, and band saws are curvilinear.</td>
</tr>
<tr>
<td>Automatic blade sweep generation</td>
<td>When a workpiece or a blade is moved and there is an intersection of the blade and workpiece, the profile from the blade is swept to create the cut geometry.</td>
</tr>
<tr>
<td>Solid model representation</td>
<td>The ACIS solid modeler kernel from Spatial Technology was used for representing the workpieces and machines</td>
</tr>
</tbody>
</table>
Workpiece/machine interaction  Workpieces are constrained to be attached to the workplanes of a machine, such as the table on the table saw.

Blade interchangeability  The blades contain information about how they can be mounted and their location and orientation. This allows blades to be interchangeable between machines. For instance, table saws and radial arm saws both use arbors as a blade mount. Any blade using the correct arbor size and type can be mounted on a machine that has an arbor.

<table>
<thead>
<tr>
<th>Table 1.2 Design issues and solutions in Virtual Workshop.</th>
</tr>
</thead>
<tbody>
<tr>
<td>One of the purposes of the Virtual Workshop was to have a very flexible environment. There are so many types of machines, tools, and blades available, even in a small workshop, that a single person would find it impossible to implement an entire workshop in a computer simulation. However, a system could be designed that would allow easy addition of more machines and tools. With that in mind, almost all of the input to the system is text-based and very flexible.</td>
</tr>
</tbody>
</table>

In order to limit the amount of work necessary to complete a usable subset of a workshop, only cutting machines were implemented. Sheet metal brakes, wire jigs, punches and other machines and tools were determined to be out of the scope of this effort. Future versions of the Virtual Workshop may contain additional types of machines, including lathes and sheet metal brakes.
Virtual Workshop Background and Description

Chapter 2

Design research covers a broad spectrum from feature-based design to automatic manufacturing planning. This chapter describes research related to the Virtual Workshop and also contains an overview of the Virtual Workshop.
2.1 Background and Prior Art

The move to solid modeling using Boolean operations in the 1970's [Braid 75] was one of the first steps towards a manufacturing metaphor for design. Up until that time, models in CAD systems were generated using lines and curves. Parts were built up a line or a vertex at a time. The ability to add a hole to a model by subtracting a cylinder instead of by drawing circles was a huge leap forward for designers.

More recently, feature-based modeling has taken a step toward the manufacturing metaphor by naming elements in a design to match features created using specific manufacturing processes. Invalid or impossible to manufacture designs can be created in a feature based modeler just as they can in any CAD system. The advantage of naming design elements after features is that they are easier to remember and understand because they have an analog in the real world.

The Virtual Workshop was envisioned as a method of making manufacturing constraints real to the designer. The designer is immersed in an environment that allows flexibility within the constraints of a real workshop. The elements of the design environment look and act like their counterparts in the manufacturing environment and design tasks are accomplished the same way as manufacturing tasks.

Development of the Virtual Workshop required an understanding of the disciplines listed in this section. For the Virtual Workshop, an effort was made to use the best available methods and algorithms from each of these areas.

Design by Process

One of the first systems to allow design using a manufacturing metaphor was a system devised by Gossard [Gossard 75]. His program allowed a designer to move a lathe tool into a piece of stock on a computer screen. Boolean operations between the swept tool path and the stock defined the part geometry. The focus of his work was graphical programming of NC tool paths. Since the advent of NC programming languages more than a decade earlier, most tool path programming was text-based rather than graphical. Gossard's analogic part programming method directly represented the geometry of the tool and the workpiece on the screen, allowing the machinist to generate NC code while viewing the final workpiece at the same time. This method of tool path programming was an effort to reduce the complexity of the programming task.
Robertson, Ulrich, and Filerman more recently created a system using a manufacturing metaphor for designing sheet metal parts [Robertson 91]. They reasoned that a CAD system "should utilize a production-like metaphor". Allowing the designer to design using processes should increase the producibility of parts and make the system easier to learn.

Cutkosky, at Stanford, has been developing a design system (originally called "First Cut") for over 5 years [Cutkosky 90]. His system allows the designer to work in "manufacturing modes" by adding process-based features to parts. Each feature, such as a through-hole or a slot, corresponds to both geometry and process. The user designs the process and the product concurrently. A set of programs called "manufacturing experts" assist the designer in making appropriate process decisions and guide the designer to create producible parts. Impressively, the output of his system is an NC machined part, created without the assistance of human hands. He has also worked on partially specified designs and his focus currently is on an extension of First Cut (called SHARE) that permits designers to work together across a computer network.

West has created a simulated manufacturing system for the design of submarine hulls [West 90]. Accurate process simulation feeds back essential information to the designer about cost, roundness, and quality based on design choices. His focus in developing the system was on design through manufacturing. His system allowed designers to "accelerate their learning experience" by making "mistakes with silicon instead of steel" [West 90]. Although the Virtual Workshop shares many of the goals of the previous systems (most notably that of Robertson) the focus has been on creating a design environment that simulates a workshop. The machines in the Virtual Workshop operate the same way as those in a real workshop. Also, the structure of the Virtual Workshop supports many different types of machines, cutting tools, and stock in a general fashion rather than focusing on a specific device or material. Stock is inserted into simulated machines and machined using bits, blades, and cutters like those found in a shop. Not all manufacturing constraints are enforced, but the user does become familiar with both the machines and the process through the use of the system.

**Swept Volumes**

One area of continuing research has been in the simulation and verification of NC machining programs. Simulators machine a simulated part based on the tool path specified by an NC program. This allows an automatic check of gross NC programming errors and deviation from the original design geometry. Verifiers are more complex programs that confirm geometry, monitor cutter collisions with fixtures, and evaluate cutting speeds.
Both of these types of programs require the ability to create complex sweeps of cutting tools, primarily ball and flat end mills. [Sungertken 86], [Wang 86a], and [Jerard 89] have all proposed methods for creating the swept volumes for material removal from stock, useful for both simulators and verifiers. Gossard inverted his analog part programming system to accept NC tape input and display final part geometry. [Gossard 78].

The Virtual Workshop creates generic swept volumes based on constrained blade geometry. The blades contain information about their profiles and cutting abilities that are converted to swept volumes as needed. For instance, drill bits contain information that allows sweeps only to be generated when the drill is moved against a workpiece in a direction that is in line with its axis. Mill cutters can cut material whether they are moved into the workpiece axially or from side to side.

**Simulation**

Other virtual environments have focused on flexibility and constraint management. Zeltzer and his colleagues have developed a system called Bolio that is a world in and of itself [Zeltzer 89]. Bolio is a graphical, distributed, constraint-based environment with objects which are free to interact with each other. In one configuration, the user of the system is allowed to play handball with a robot, incorporating inverse kinematics, dynamic simulation, 6 degree-of-freedom input and many other complicated interactions in a single simulation. Bolio succeeds in defining an extremely flexible simulation environment for testing input devices, simulation and animation methods, etc.

**Simultaneous Engineering**

The Virtual Workshop was also created to address the desire for simultaneous engineering. Many companies are focusing efforts on designing for manufacturing, recognizing that the cost and quality of a product are primarily determined during design [Evans 88]. One approach that helps a designer understand the impact of a design on manufacturing is to provide information about a design's impact earlier in the design process. Using this information, the designer can make informed choices and improve the final manufactured part based on real information. The Virtual Workshop helps the designer understand the impact of process by requiring the designer to create parts using those processes. It provides some immediate feedback, like cost, that the designer can refer to while developing the final product.
Direct Manipulation

One of the priorities during development of the Virtual Workshop was allowing a designer to directly manipulate tools and machinery. Direct manipulation removes a layer of abstraction in a human-computer interaction. "The computer becomes transparent and users can concentrate on their tasks." [Shneiderman 83]. Several methods of manipulating the machines and machine components were tried before settling on the methods described in Chapter 4. Users describe the Virtual Workshop as "approachable" and "easy to learn" as a result of the intuitive interface.

Stereoscopics

One of the options considered for implementation in the Virtual Workshop was the use of a real-time stereoscopic display. Although the addition of depth perception by using separate viewpoints for each eye is helpful, Sollenberger and Milgram [Sollenberger 89] found that the ability to rotate a monoscopic view improved depth perception substantially more than the addition of stereoscopic capabilities to a static view for certain applications. They also showed that the combination of stereoscopic view and image rotation was not a significant improvement over the monoscopic view. For this reason, the addition of stereoscopic viewing capabilities to the Virtual Workshop was not a priority in the initial implementation.

Solid Modeling

Over the past 20 years, several approaches to solid model representation have been developed. Boundary representations, constructive solid geometry, binary space partitioning, and octrees are all possible solid representation schemes. A comprehensive summary of solid representations and their advantages and disadvantages is contained in the book Computer Graphics: Principles and Practice [Foley 91].

Until a few years ago, CAD companies or researchers interested in representing solids had to develop their own modelers. Recently, several companies have produced solid modeling kernels. These kernels can be embedded into either commercial or research systems and provide a programmer with tools for representing and manipulating solids without burdening the programmer with specific, non-extensible user interfaces. One of those kernels comes from Spatial Technology and is called the ACIS solid modeling kernel [Sowar 91].

Originally, the Virtual Workshop used a solids representation based on the work of Mäntylä [Mäntylä 88]. A version of the workshop using binary space partitioning (BSP)
methods was also built and tested [Naylor 90c]. Both representations suffered from two primary limitations. Home grown or minimally supported solid modelers are often unoptimized (slow) and functionally incomplete. Also, half-edge solids and BSP trees can only represent polyhedra.

The Virtual Workshop eventually incorporated the ACIS kernel for all of the solid representations in the workshop. Numerous built-in functions and a robust, complete representation were benefits gained by using the kernel. Other kernels are available also, but are not discussed here.

Features

Today, many CAD companies use an interface based on adding "features" to a design. Pro/Engineer from Parametric Technology Corporation, and the ConceptStation modeler from Aries Technology allow designers to create models with slots, through-holes, and other manufacturing features. Features-based design is a way of using a familiar vocabulary to create geometry in a solid model.

Luby, et. al. created a system whereby a design is constructed using features from a features database [Luby 86a], [Luby 86b]. They claim that basing a design on features will assist in "manufacturability evaluation, process design, and process planning." Feature-based design also has the advantage of inherent parametric capability since features are described using dimensioned, prototype features. The focus of this work is similar to that of the "First Cut" system [Cutkosky 87].

In the Virtual Workshop, features are a side effect of the manufacturing operation. Allowing the designer to add features directly to a design might be a way to speed up geometry input in the workshop without sacrificing the workshop metaphor. Each feature would be associated with a machine and blade used to create that feature.

Tolerances

One of the most difficult tasks in design is deciding and specifying tolerances in the design. The complexity of the task arises from the multitude of ways an entity can be imperfect. For instance, a planar surface requires the following information for complete specification: surface roughness, distance from datum plane, and coplanarity to datum plane. As the type of surface becomes more complex, so do the ways of representing its accuracy and dimensions.
Gossard, Zuffante, and Sakurai explored tolerance and assembly representation [Gossard 88]. Requicha and Chan also proposed a solid modeler representation that incorporated tolerances and other attributes [Requicha 86].

As currently implemented, the Virtual Workshop does not store any tolerance information with the design. However, a knowledge-based system could be included in the workshop that determines surface positional accuracy, runout, and finish based on machining operations and setup. A database could be created that contains information about the effect of each type of process and setup. Information from that database would be attached to each surface of the solid during creation of that surface. The information could then be used during assembly to check fit and the information could also be queried by the designer to learn more about each operation.
2.2 Overview

The main window of the Virtual Workshop is shown in Figure 2.1. The window always displays one of the machines, the current workpiece, and the main menu. Dialog and information boxes are also available for some operations of the Virtual Workshop.

Figure 2.1 Main window of the Virtual Workshop.
The designer can change viewpoints in the Virtual Workshop by clicking on a part of the screen not occupied by the machine and dragging the mouse. When the mouse moves, the eyepoint moves in the corresponding direction and the view changes. To move to the other side of the machine, the designer clicks with the mouse and drags it either to the right or left until the machine has rotated 180 degrees. To move above the machine, the designer must drag the mouse upwards as if moving the eye viewing the machine.

The illusion created by the three dimensional environment is quite convincing. One user of the Virtual Workshop found himself moving his head in order to get a better view of part of the machine after adjusting the viewpoint.

![Figure 2.2 Drill Press with drill bit and stock.](image)

All of the elements on the machine that the user can manipulate are displayed as part of the machine. For instance, the drill press is shown in Figure 2.2. The little crank on the back part of the table will adjust the height of the table when rotated. In Figure 2.3a, the other side of the drill press is shown, displaying the locking handle for the table. When the designer clicks on that lock, it is redisplayed in its locked position (Figure 2.3b) and the crank and table are no longer allowed to move.
Figure 2.3 Drill press with locked (a) and unlocked (b) table.

The user can press the drill bit down into the stock by grabbing the press handle and rotating it. (See Figure 2.4).

Figure 2.4 Drill press before (a) and during (b) the creation of a hole.

The main menu of the Virtual Workshop is shown in Figure 2.5. The Project menu is used for saving and retrieving parts and also for exiting the application. The Stock menu allows the designer to select new pieces of material for cutting operations. The View menu allows
the user to change the viewpoint or display a dialog box for using multiple viewpoints. The View menu also has options to allow the designer to look more closely at the positions of the parts on the machines and the position of the workpiece.

Figure 2.5 Main menu of Virtual Workshop.

The Machine menu allows the designer to change machines or select a new blade, bit, or cutter for the current machine. The Mark menu allows the user to measure and mark the current workpiece, or erase the current marks. Finally, the Assembly menu allows the designer to assemble parts created in the workshop.

Figure 2.6 shows a workpiece on the table saw. The table saw fence has been adjusted to rip the wood 5 inches wide using a combination blade. Figure 2.7 shows the wood when it has been cut halfway along the length. Notice in Figure 2.8 that the workpiece has been completely cut, and the other piece remaining from the cut has disappeared. The new piece is still part of the project, but is a separate workpiece.
Figure 2.6 Table saw cutting workpiece in half (setup).
Figure 2.7 Table saw cutting workpiece in half (partial cut).
Figure 2.8 Table saw cutting workpiece in half (finish).
Figure 2.9 Simple wood assembly (dado joint) on workbench.

Figure 1.2 shows a bookshelf and Figure 2.7 shows a simple dado joint assembly on the workbench. Each of the pieces in the assemblies were created and assembled in the Virtual Workshop.
2.3 Workshop Machines

Figures 2.10 through 2.14 show the machines currently available in the Virtual Workshop.

Figure 2.10 Band saw.
Figure 2.11 Drill press.

Figure 2.12 Milling machine.
Figure 2.13 Radial arm saw.

Figure 2.14 Table saw.
This chapter describes interaction with the Virtual Workshop. Several tasks in the workshop are discussed in detail.
Many CAD systems are only accessible through a series of menus or icons. Although this is a familiar approach to computer users, it doesn't have an analog in the real world. One of the purposes of using a metaphor in a computer system is to make it more approachable for the user. Familiar operations are accomplished using familiar actions. For instance, adjusting the blade of a real table saw is achieved by rotating a hand crank on the side of the saw base. In the Virtual Workshop, the mouse is used to rotate the crank. An alternative approach might have been to select a series of menu items (e.g. Table Saw \| Blade \| Adjust \| Height). The menu approach might have some advantages, but directly manipulating the "hardware" corresponding to the adjustment mechanism on a real table saw is familiar and easy to remember.

Much human communication occurs through written or spoken language. The set of words and phrases used in that language is referred to as a vocabulary. The typical vocabulary of human-computer interaction consists of type-written commands, menu selections, and mouse clicks and drags. The Virtual Workshop, as with all virtual environments, has been created in such a way as to allow the vocabulary of the human-computer interaction to approach that of the original vocabulary of a familiar environment. By maintaining a consistent vocabulary with a real workshop, the VW allows its users to remember and reuse actions learned in both environments.
3.1 Material Selection

All operations in a manufacturing environment are performed on real materials using real equipment and processes. In fact, many of the details of a design are centered around what type of material and process are going to be used to achieve the goals of the design. However, design is often considered in terms of satisfying geometric and other constraints. Process and material decisions are left until the end. Occasionally this is the correct approach, especially where the performance and accuracy constraints are substantially more difficult to address than material and process selection. However, experienced designers seem to focus on material and construction very early in the design process (see Chapter 5). In fact, one of the big pushes in design currently is to coordinate manufacturing and design (known as "Concurrent Engineering") in order to substantially improve the final product. It has been argued that much better final products are produced by companies and individuals who consider all of the implications of each design decision, including constraint satisfaction and manufacturability, early in the design process.

A workshop metaphor requires the designer to select stock before beginning a design. The piece of stock selected defines several things about the design, including the processes and machines used and the maximum overall size of a single part designed using that piece.

Access

Currently, material selection is accomplished through a dialog box. All of the important information about the stock is displayed to the designer as the piece of stock is selected, including important dimensions, cost, etc. A visual representation of the stock is not displayed on the screen until it is added to the project and becomes a workpiece in the project. However, the combination of the material, name, and dimensions of the stock are usually sufficient to describe the stock. Future versions of the workshop might display a thumbnail sketch of the part before it is added to the project. This ability will be especially useful to people not familiar with material available in a stockroom.

Initially, some thought was given to representing the stockroom literally, the way the rest of the Virtual Workshop has been represented. However, a visit to the stockroom in a real workshop indicated that the stockroom is not the most convenient or efficient way to store and present material to the designer. One of the difficulties with selection of stock is the
number of variables that describe a particular piece of stock. For instance, wood comes in many shapes, styles, and sizes. Some of the variables available to the designer when selecting a piece of wood include thickness, length, width, type of wood, finish etc. With just those variables alone, the designer is asked to traverse a sparse 5 dimensional space to find stock. Trying to display that in a 3 dimensional virtual stockroom is problematic. However, a combination of menus and dialog boxes can be used effectively by allowing each dimension to be chosen separately. The menus narrow the choices to the designer before the dialog box is displayed. For instance, in the first level of menus, the designer selects between materials at the highest level. Selecting one of metals, glasses, wood, composites and specialty stock narrows the remaining number of choices significantly. The next level in the menu narrows the choices down again. For instance, if the designer selects wood from the stock menu, another menu appears allowing a choice of Framing lumber, Furniture wood, Plywood, or Specialty woods. If the designer chooses Furniture wood, a dialog box appears that presents a selection of wood to the designer (see Figure 3.1).

Figure 3.1 Furniture wood dialog box.

With the furniture wood dialog box showing, the designer is allowed to select a type of wood, its thickness, length, and width. Some types of wood are only available in certain thicknesses and the thickness option reflects the valid selections. Thicknesses are represented in the terminology of a lumber yard, reflecting the lumber yard environment. One inch thick boards (which are actually 3/4 inch thick after dressing) are offered as “4/4”
boards. Redwood only comes in “16/4” thickness or 3 and 3/4 inches (after dressing). When redwood is selected, all other thicknesses besides “16/4” are disabled in the thickness menu indicating that those thicknesses are unavailable. The cost, volume, and weight of the wood is always calculated and displayed.
3.2 Measuring and Marking

Essential to any workshop is the ability to measure and mark workpieces. Marks provide a reference for a cut when fixtures or jigs are not available for controlling the position of the workpiece with respect to the blade. Marks can also be used for simple cuts where the accuracy of the machining operation is not of paramount importance.

Overview

For the Virtual Workshop, several simple tools were chosen as representative of the types of marking and measuring done in a workshop. The ability to draw a circle (like a compass), a straight line, a freehand polygon, and an offset curve were selected to represent measuring and marking in the workshop. Also, since measuring and marking usually occur at the same time, measuring was built into marking in the interface. In other words, a measurement leaves a mark when it is complete. All of the marks can be erased from the workpiece using the “Erase Marks” item from the “Marks” menu.

Table 3.1 lists the types of marks that can be made on a workpiece.

<table>
<thead>
<tr>
<th>Mark Style</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Circle</td>
<td>Draws circle around center point through radius point.</td>
</tr>
<tr>
<td>Line segment</td>
<td>Line is drawn between two points.</td>
</tr>
<tr>
<td>Freehand polygon</td>
<td>Line is drawn along motion of mouse.</td>
</tr>
<tr>
<td>Edge offset</td>
<td>Line is drawn parallel to edge through point.</td>
</tr>
</tbody>
</table>

Table 3.1 Types of marks that can be made on workpieces.

Access

In order to measure or mark the workpiece, one of the types of marks must be chosen from the “Marks” menu. Table 3.2 specifies how each type of mark is created.
**Mark** | **Creation** |
---|---|
**Measurement information** | |
Circle | The designer selects the center point of the circle on the workpiece by pressing the left mouse button. The designer drags the mouse to a point on the circumference of the circle. While dragging the mouse, the workshop displays a circle based on the current position of the mouse. When the left mouse button is released, the final circle is created and added as a wire to the workpiece. Along with the circle, two orthogonal lines are created and added to the workpiece to mark the center of the circle. While the mouse is moving, the center point values, the current radius point values, and the current radius of the circle are displayed. |
Line | The designer selects the start point of the line by pressing the left mouse button. The designer drags the mouse to the endpoint of the line. While dragging the mouse, the workshop displays a line between the start point and the current mouse position. When the button is released, the line is created and added to the workpiece. During creation of the line, the start point, length, and x, y, and z displacements from the start point are displayed. |
Freehand polygon | The designer selects the start point of the polygon by pressing the left mouse button. As the designer drags the mouse, new points on the polygon are created and the current polygon is displayed. When the left mouse button is released, a line going through all of the polygon points is created and added to the workpiece. Nothing is displayed during creation of the polygon except the current polygon. |
Offset | The designer selects an edge on the workpiece by pressing the left mouse button. As the designer moves the mouse on the surface of the workpiece, an offset curve of the selected edge going through the current mouse point is created and displayed. When the mouse button is released, the offset curve is added to the workpiece. Offsets can be created from any edge in the workpiece (circular, straight, or spline.) During creation of an offset curve, the distance from the edge is displayed. Also, the start point and the x, y, and z values of the offset vector are displayed. However, the start point changes as the mouse is moved because the line between the edge and the new offset curve is always perpendicular to the edge and the offset. The start point is always the point on the original edge that is closest to the current mouse point. |

Table 3.2 Creation of marks in the Virtual Workshop.
3.3 Using the Virtual Machines

Most of the interaction with the machines in the workshop is directly through the visual interface. Machine parts can be adjusted, workpieces moved, rotated and cut, and marks applied. Other interaction takes place through dialog boxes and menu selections.

Machine Adjustments

Each machine in the workshop is listed in the “Machine” menu. The machines can be selected by name.

When a machine is selected, the stock is automatically inserted into the machine. Each machine has between one and three workplanes that constrain the position of the workpiece. See Section 4.3 for a description of the interaction between machines and workpieces.

Generally, machine parts can be adjusted by dragging the part itself or by moving a crank that drives the machine part. For instance, the table on a drill press can be moved up and down using the mouse (see Figure 3.2). The crank on the table will also move the table up and down (see Figure 2.2). This is similar to the way the machine works in a real workshop.

![Figure 3.2 Motion of the table on the drill press.](image-url)
The machines represented in a workshop are all cutting machines. Any motion of the blades and workpieces that causes an intersection removes material from the workpiece. A workpiece can be pushed into a blade (table saw) or the blade can be moved into the workpiece (drill press). The position of the blade can be adjusted by moving the part of the machine that holds the blade. For instance, the bit in the drill press is held by the chuck. When the chuck position is adjusted the blade moves with it.

Some of the parts in the machines have locks. A lock can either be on or off. When a lock is off, clicking on it with the mouse turns it on. Generally, locks change appearance to indicate their state. A lock that is on is displayed in one position and when it is turned off, it is displayed in a different position (see Figure 2.3).

**Blade Selection**

New blades can be selected for any of the machines. Any valid blade file can be used in the workshop. The files in the *Blades* directory underneath the main workshop directory that end in "blade" are valid blade files.

In order to be used in a machine, the type of mount that the blade fits in must be the same as the mount in the machine. For a detailed description of how the blades and mounts are handled in the workshop, see Section 4.6.

To change the blade, the designer must select "New blade, bit, or cutter..." from the "Machine" menu. A dialog box appears that allows the designer to select a file. If a file ending in "blade" is selected, the workshop reads in the file and tries to insert it in the machines blade mount. If successful, the new blade will define the shape and extent of the cutting operations. If the blade is not created, then the old blade remains in place in the machine.
Representation and Implementation

Chapter 4

One of the most difficult things about designing the Virtual Workshop was organizing the data in a flexible manner that would allow the representation of several types of machines, blades, and other objects. This chapter details the representation of each of the objects in the workshop and how the designer interacts with those objects.

Difficult concepts, such as how blade sweeps were generated, are covered in detail, but other common programming details, such as linked list implementations, are not expounded in this text.
4.1 Virtual Workshop Objects

Overview

The Virtual Workshop currently consists of machines, blades, and workpieces. Each of these objects have visual representations, solid models that underlie the visual elements, and other information. All of the information, including the solid model, are represented using text files.

At any given time, the Virtual Workshop displays one machine, the blade in that machine, and one workpiece assembly. The workpiece assembly may be a single workpiece, or several workpieces attached to each other. Machines are not required to have blades, because a machine may represent an object like a workbench which is just used for convenience in assembling workpieces.

All objects in the workshop have names. The names are used for dialog box information, selection, and associating the objects with their visual representations in the file system. In the future, the names could also be used for recording operations and processes. Each workshop object that is moved or adjusted can be referred to by name in the journal file.

Each object and the information about each object is stored in a directory. Figure 4.1 shows the organization and structure of the directories.

![Diagram of directory structure]

Figure 4.1 Directory layout.
<table>
<thead>
<tr>
<th>Directory</th>
<th>Contains:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workpieces</td>
<td>Each file in this directory contains a finished workpiece that was stored using the ‘Save’ command from the ‘File’ menu.</td>
</tr>
<tr>
<td>Machines</td>
<td>Files in this directory that end in “.machine” are the machines which are available for use in the Virtual Workshop.</td>
</tr>
<tr>
<td>Blades</td>
<td>All of the information necessary to completely describe a blade is contained in this directory. A “.blade” file contains information about how a blade can be used (cut direction, cut profile) and a “.sat” file contains a solid model and a visual representation of the blade.</td>
</tr>
<tr>
<td>PartGeometry</td>
<td>Each machine can contain one or more parts. The visual representation (or solid model) of those parts are contained in “.sat” files in this directory. Each file in this directory is named in a way that indicates which part it represents and the machine in which it is contained.</td>
</tr>
</tbody>
</table>

Table 4.1 Contents of the Virtual Workshop directories.

These directories and the way they are laid out are only for convenience and as a method of organization. They do not represent any internal organization of the program.

**Text-based Representation**

One of the goals in building the Virtual Workshop was to represent the individual elements of the workshop in a flexible way. This would allow quick development of many machines once the workshop was finished. The workshop should not be required to have specialized knowledge about any of the machines or other parts of the workshop. The representation had to be generic, yet rich enough to permit many different kinds of machines to exist using a single standard representation.

The data structures drive the workshop when a designer is using it. When someone clicks on a machine part, the workshop queries that part to find out what kind of motion, if any, is allowed. If motion is allowed, the workshop allows the part to move as the mouse is moved.

One of the many benefits of having a generic representation is that all of the objects in the workshop were represented using text files. Changes to the Virtual Workshop and the available machines and blades can be affected simply by editing text files using any text editor available to the designer.
To create and add new machines, they simply need to be defined in a text file using any word processor. The language for defining them is well defined and easy to read. Examples of machine and blade input files can be found in the appendices. This chapter will describe the elements of the Virtual Workshop and how the Workshop was constructed.
4.2 Solid Models

All of the solid models in the Virtual Workshop are based on and represented using the ACIS solid modeling kernel. This provides flexibility to the designer in that all workpieces created in the workshop can be used in any other system which reads ACIS format files. Any commercial solid modeler based on the ACIS kernel will be able to exchange workpieces with the Virtual Workshop.

Solid models are used for representing the visual elements in the Virtual Workshop, including machines, blades, workpieces, and the workshop environment. The machines, blades and environment are simply static models, while the workpieces may be modified through interaction with the blades.

Originally, two different types of polyhedral modelers were examined for solid representation in the Virtual Workshop. One modeler was based on work by Martti Mäntylä [Mäntylä 88] and used the half-edge data structure and Euler operators described in his book. The second modeler was developed by Bruce Naylor, John Amanatides, and William Thibault [Naylor 90b], [Naylor 90c]. This modeler was based on binary space partitioning trees (BSP trees) and had the advantage of being fairly quick. The primary drawback of both systems was their inability to represent curves. The implementation of the first version of the Virtual Workshop using a BSP tree library from AT&T showed that polyhedral models were not adequate. The appearance of the polyhedral models was unsatisfactory and did not lend realism to the simulation. Also, a serious difficulty arose when assembling parts. Imagine that a circular hole was created in a block. If a circular shaft was created slightly smaller than the hole, it should be able to fit in the hole without interference. However, if the shaft and hole were represented using different numbers of polygons around their circumferences, or if the shaft were rotated just slightly with respect to the hole, they could not be assembled due to interferences (see Figure 4.2). The interference is due to the inaccuracies implicit in polyhedral models and is unrelated to the actual geometry.
ACIS models contain a structure that can represent non-manifold solids using implicit or spline-based surfaces. A short description of an ACIS model is presented here to illustrate the representation of solid models in the Virtual Workshop. For a detailed overview of how ACIS models are stored, refer to the ACIS interface reference manuals.

All ACIS solids are stored in a structure called a body. A body consists of zero or more lumps and a transform that describes the current position and orientation of the body. A lump is a group of surfaces that are connected together and form a solid. A lump could be a sphere, cube, or any complex connected entity. Most bodies in the Virtual Workshop have a single lump. When a piece of stock is cut in such a way that the body is separated into two or more lumps, each lump is reinserted into a new body as a separate workpiece (see Figure 4.3).
The lumps consist of faces, edges, and vertices that describe the geometry and connectivity of the lump. There are also other entities within a lump for accounting purposes, but this simple description is adequate for understanding how the Virtual Workshop deals with solid models.

Transformations of complex solid models can be computationally expensive. However, recalculation and transformation all of the points, edges, and faces is not necessary when a model is transformed as long as the correct transformation is maintained in the model. The ACIS solid always keeps a transformation matrix as part of the body to track the position of the body. It only recalculates the geometry of the body when absolutely necessary, like when merging or intersecting two separate bodies with different transformation matrices.

A good example of saving computation time by maintaining a transformation matrix is during image redraw. When the workshop starts, each machine part is read into the workshop as an ACIS body. A faceted polyhedral representation of the machine part body is calculated and stored. When the designer moves the machine part and the part is transformed, it must be drawn in the new position on the screen. However, instead of recalculating the position of all of the points and normals of the polyhedra, the transformation matrix describing the change in position of the part is simply added to the display matrix. Thus, when the part is redrawn using the same list of polygons and points,
it appears in the new position on the screen. The transformation matrix is then removed from the display matrix and the next part is drawn.

Using this technique, none of the positions or orientations of any of the machine parts are ever actually changed within the Virtual Workshop. Only the transformation matrix in the body is modified. The faceted polyhedral representation of the body is displayed in the correct position without ever recalculating the position or orientation of the facets.

The polyhedra for each solid are created using a faceter included with the solid modeler. The faceter is general enough to create the facets for any model that can be represented in the modeler. The facets are created and stored in a object called a POLYGON_POINT_-MESH (PPM). The PPM contains a list of polygons that make up the polyhedron. Each polygon contains a list of points that outline the polygon and the normals at those points. Using the PPM, the Virtual Workshop creates an object called a pmesh that is used internally in the workshop. The pmesh contains all of the same information contained in the PPM and also has a transformation matrix to keep track of the location and orientation of the pmesh. The pmesh object also has drawing, transformation, and output routines that are not available in the PPM object.

A pmesh is created for each entity to be displayed, including workpieces, machine parts, and blades. The machine parts' and blades' pmesh are never recalculated, because their geometry is static. The workpieces' pmesh is recalculated after cutting operations because of changes in geometry.

**Manipulating the Models**

Bodies can be transformed, intersected, subtracted, united, and have a variety of other operations performed on them. The ACIS solid modeling kernel includes a set of interface routines called *API routines* that provide access to Boolean operations, body creation, and other essential manipulation routines.

Most manipulation of the bodies occurs through the API subroutines. However, some types of interactions with the solids are not available through the API and must be managed directly. For instance, marks are kept on the body as *wires* and those wires are kept with the solid part of the body. One limitation of the API is that the Boolean operations can not handle wires as part of a solid. Boolean operations like api_subtract() remove wires and delete them. In order to keep the wires on the body, they must be removed from the body.
before a Boolean operation. The wires are reattached to the body after the cut. They are also copied and attached to any new bodies that were created as a result of the cut.

**Use of Solids in Virtual Workshop**

The primary use of solids is to represent the visual elements of the workshop and to perform Boolean operations between the blades, their swept profiles, and the workpieces.

**Machines**

Each machine is made of several parts. A part is kept as an ACIS solid model in a file in the PartGeometry directory. The name of the file is based on the machine and the part path. All of the parts for one machine could be maintained and stored in one file, but it is more convenient to keep them separate during development.

Machine-machine and machine-workpiece intersections are not checked during the operation of the Virtual Workshop. Currently, only workpiece-blade intersections are calculated. The machine-machine intersections and machine-workpiece intersections can be calculated using the API, but intersection checking slows the workshop down considerably.

Most of the time, the constraints on the machine parts keep the machine from intersecting itself. However, with any complex mechanism there are some configurations that will cause machine-machine intersections. These intersections may have disastrous results in a real workshop, but in the Virtual Workshop, the intersections are inconsequential and they are allowed and ignored. Therefore, it is left to the designer using the workshop to keep the part from intersecting the machine and to keep from moving one part of the machine into another part. Because of the visual nature of the workshop, it is not difficult for the designer to recognize when these intersections occur.

**Workpieces**

Workpiece geometry is stored and manipulated as an ACIS solid model. The primary difference between workpieces and the other solids in the environment is that workpieces can contain marks drawn by the designer. Also, no material is ever removed from any other solid in the environment.

ACIS bodies can be solids or wireframes. Both a solid model and a wireframe can be stored in one body concurrently, even though they usually are not used at the same time.
In the Virtual Workshop, the wireframe portion of a workpiece body is used to store marks that are put on the body by the designer. When a workpiece is drawn, the wireframe portion of the body is drawn also, so any wires stored in the wireframe appear as if they were attached to the body. Of course, in order to appear on the body, the wires must be on or above the surface of the solid. Otherwise they will be obscured by the surface.

Boolean operations remove the marks, and so special care was taken when cutting the bodies to remove the marks and restore them after the cut.

**Blades**

Each blade contains at least two ACIS solids (remember that solids can contain either objects with surfaces or just wireframes or both). One solid represents the geometry of the blade. The other solids are really just wire frame profiles that represent the profiles of the blade. A detailed description of the blade and profile representation is discussed in the “Blade Representation” section of this chapter.
4.3 Machine Representation and Control

The machines are the most complex element in the Virtual Workshop. They must interact with the workpieces and the blades. They must also be adjustable. This complexity is compounded by the fact that the machines are represented in a flexible manner that allows them to be created automatically from a textual description.

Data Structure

The machine description data that is stored in the computer and read from the text file is listed in Table 4.2.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name, brand, style</td>
<td>Name, brand, style and name of the default blade.</td>
</tr>
<tr>
<td>Default blade</td>
<td>Name of default blade for this machine.</td>
</tr>
<tr>
<td>Parts list</td>
<td>Pointers to the parts on this machine.</td>
</tr>
<tr>
<td>Lock list</td>
<td>List of locks on the machine.</td>
</tr>
<tr>
<td>Blade mount information</td>
<td>Information about how the blade connects to the machine.</td>
</tr>
<tr>
<td>Blade pointer</td>
<td>Pointer to the current blade.</td>
</tr>
<tr>
<td>Linked list information</td>
<td>Linked list of machines currently available in the Virtual Workshop.</td>
</tr>
</tbody>
</table>

Table 4.2 Information stored in the machine.

Name, Brand, Style

The name, brand, and style of the machine are stored for the convenience of the designer. The name is also used to access parts from the PartGeometry directory because the filename of the machine parts is always preceded by the name of the machine. When the name is translated to a filename, the spaces in the name are converted to underscores ("_"). For instance, "Table Saw" becomes "Table_Saw". Brand and style are simply detailed descriptions of the machine.
Default Blade

The default blade name is kept as text so that when the machine is initially read into the Virtual Workshop and created, a blade can be automatically installed. The blade name is converted for use as a filename, similar to the way the machine name is converted, and then the blade is read in from disk. A complete description of the blades and blade input files is contained in the section entitled “Blade Representation.”

Parts

The visual elements of the machine are called parts and locks. Each part of the machine can be attached to the main machine, or attached to another part. The parts are attached using constraints which describe allowable motions of the parts. All of the parts attached directly to the machine are stored in a linked list in the machine object. Parts attached to other parts are called children of that part. A part that contains children is referred to as the parent of those parts. Parts that are children of other parts are stored in a linked list in the parent part.

Some machine parts can control the motion of other parts. Very complex machine structures can be built up using the part tree in the machine. More details about how the parts are connected and constrained together will be described later in this section and in section 4.4.

Locks

Locks can be attached to any part in a machine. A lock controls whether a part is free to move or fixed in position. When the lock is off, the part is free to move as described in the constraint connecting the part to its parent. When the lock is turned on, the part is fixed to its parent. An example of a lock is the lock handle on a table saw rip fence. The fence can move along a straight line on the table when the lock is off. As soon as the handle is locked, the fence can no longer move and stays fixed in place.

Blade Mount

The blade mount describes what types of blades can be used in the machine and how each blade must be oriented. Section 4.6, “Blade Representation”, gives a detailed overview of how the blades and blade mounts interact between the machine and the blades.
Blade Pointer

The blade pointer contains a pointer to the current blade in use by the machine. When no blade is in use, this pointer is empty, or NULL.

Linked Machine List

There are also pointers in the machine to keep track of all of the machines available in the workshop. This allows a machine to be selected by name because the workshop can search the entire list of machines using the linked list.

Representing Machine Adjustments

A primary focus of this research was creating a representation and format for machines that would allow easy use of each machine and an environment for quickly learning how to use the machines. Machines represented on the screen should mimic machines in a real workshop. Direct manipulation of parts, workpieces, and locks were essential to maintaining interactivity and ease of use in the workshop.

Some of the qualities of a real machine include the ability to adjust fences and fixtures, control adjustable parts on the machine, lock parts of the machine, and change blades and bits. Parts, locks and blades are all objects that assist in making the machine simulation more like a real machine. In the real world, many of those objects are manipulated by grasping them and pushing or twisting. In the Virtual Workshop, those objects can be grabbed and moved using the mouse.

As mentioned before, parts can be connected directly to the machine, or to other parts. Figure 4.4 displays an example of how parts, locks, and machines can be connected together.
Figure 4.4 displays the connectivity graph of a simple band saw like the one in the Virtual Workshop. The actual band saw is a bit more complex than this diagram shows, but this simple diagram will suffice for a quick explanation of how the machine elements interact. Not shown in this machine are parts that drive each other, where the motion of one of the parts drives the motion of another part. Driving parts are connected with a double arrow shaded line. An example of two driving parts would be the blade height crank and the arbor (blade mount) on the table saw. Motion of the blade height crank moves the arbor up and down in the saw, positioning the blade for a cross cut or a dado.

In the figure, the rounded rectangle represents the machine and the other rectangles are the parts in that machine. The ovals represent locks attached to the parts that can stop the movement of other parts. The black arrows connecting the parts represent constraints that define allowable motion between parts.

The pathname of a part is created from a combination of the machine name and the names of the parts above the part. The pathname is used for locating parts in a machine and also in naming the solid model input files for the machine. The pathname for the base in the band saw is “Band Saw.Base”. A period separates the name of each element. Spaces are
allowed and are converted to underscores ("_") when the pathname is used as a filename. A more complex pathname example is the path of the crosscut fence. It's pathname is "Band Saw.Base.Table.Crosscut Fence".

The reason that the names are so important is that they provide a method for reading the machine from the input file. When a lock is described in the machine input file, it must specify its parent part (the part on which it is mounted) and the part that it controls. The only way to specify the part name is by giving the full pathname to the part. Part pathnames also allow the recording of machine motion. As each part is adjusted, the name of the part that was adjusted and how it was adjusted can be recorded in a journal. The names can also be used for communicating with other workshops running concurrently on a network. The communication aspect of the Virtual Workshop will be discussed in more detail in the section entitled "Journals and Roll Back".

One reason for the tree hierarchy in the machine is to control how the machine parts are moved and adjusted. Just as in real life, when you move a part of a machine or mechanism, any part that is mounted on that part moves with it. The same is true of the Virtual Workshop machines. When the band saw table is moved to a different angle, all of the parts mounted on the table, including the crosscut fence and ripping fence, are moved with it (see Figure 4.5). The constraints that control respective motion between parts are also transformed, allowing the parts to continue to operate properly in their new orientation.

![Figure 4.5](image)

Figure 4.5 Rotation of the band saw table causes the crosscut fence to rotate.
Constraints and Locking

There are two kinds of constraints in the Virtual Workshop. One type is simply referred to as a constraint. This type of constraint manages the motion of one part with respect to its parent. The other type of constraint is called a cut constraint. A cut constraint is used in a blade to determine what types of cuts can be made with a blade. The cut constraints will be described in detail in section 4.6, “Blade Representation”.

A constraint is contained in a machine part and describes what kind of motion is allowed for that part. Every part has a constraint and different types of constraints require different information. The types of constraints are enumerated in Table 4.3. A part whose parent is the machine’s coordinate system. All other parts’ motions are with respect to the coordinate system of their parent part.

<table>
<thead>
<tr>
<th>Constraint Type</th>
<th>Description</th>
<th>Required information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed</td>
<td>Part should not move with respect to parent.</td>
<td>no extra information</td>
</tr>
<tr>
<td>Linear</td>
<td>Part can move in a one direction only.</td>
<td>direction vector interval (allowable linear motion)</td>
</tr>
<tr>
<td>Rotational</td>
<td>Part can rotate about an axis.</td>
<td>rotation axis (vector and point) interval (allowable rotation)</td>
</tr>
<tr>
<td>Planar</td>
<td>Part can move in a plane.</td>
<td>2 direction vectors 2 intervals (primary and secondary)</td>
</tr>
</tbody>
</table>

Table 4.3 Constraint types and required data.

Along with the required information, information can also be included about detent positions in selected constraints. Some parts on machines have mechanisms which make it easier for a part to be moved into a specific position. For instance, crosscut fences which can be adjusted to hold the workpiece at an angle other than perpendicular with the blade may have a detent at the perpendicular position. This detent makes it easier to return the crosscut fence to its original perpendicular orientation.

Each constraint, except for the fixed constraint, accepts a list of snap points, which correspond to detent positions. Additional snap points may also be implied by using an
ellipsis at the end of a list of two or more points. An example of each of the types of constraints will be illustrative.

**Fixed Constraint**

A fixed constraint does not allow any motion of the part and subsequently, no extra information is required.

The format of the constraint line is:

Constraint: *Fixed*

The fixed constraint is the default if no constraint is mentioned in the input file.

When a part with a fixed constraint is selected with the mouse, no motion of the part is allowed with respect to the parent’s coordinate frame.

**Linear Constraint**

Linear constraints allow a part to move along a straight line. No rotational motion or off-axis motion is allowed. The constraint is specified with a direction and an interval.

The format of the constraint line is:

Constraint: *Linear*: Direction: Interval: *SnapTo*: *SnapRes*: Offset

The components of a linear constraint are:

- Constraint Type: Linear
- Direction: (x, y, z)
- Interval: [min, max]
- SnapTo: [val1, val2, ...]
- SnapRes: resval
- Offset: offsetval

Type of constraint
Direction vector
Motion interval
Snap-to pts, ellipsis indicates cont’d
Snap when (abs(location - val1) < resval)
Added to interval for part indicator

The above constraint would appear in the machine file in the following format (with example values):

Constraint: Linear:(1,0,0):[-10, 36]:[5,12,18, ...]:0.1:10

When a solid model of a part is read in from the PartGeometry directory, it is assumed to be in the default (0.0) position of the interval. For this reason, a zero point must always be included in the interval. The constraint above indicates that the part is allowed to move 10.0 inches in the negative x axis direction and 36 inches in the positive x axis direction from it’s original position.
Interestingly enough, no information about the orientation or origin of the part is required by any constraint. The workshop simply keeps track of how far the part is moved in each direction and stops the part if it tries to move out of the allowed interval.

The snap-to values and snap resolution indicate detents in the constraint and how close the part must be to those detents before they are activated. The easiest way to think about constraint detents is by comparing them to a grid in a computer drawing program. When a part is near a detent, it is automatically moved to the detent position. When it is moved sufficiently far from the detent, it is free to follow the motion of the mouse.

![Diagram of detents and resolution superimposed on the constraint interval.](image)

Figure 4.6 Detents (snapTo values) and snap resolution superimposed on the constraint interval.

The ellipsis at the end of a snap-to value list indicates that the snap values should be continued to the end of the interval. For instance, if the interval is [0, 5] and the snap values are [0, 1, ...], the intervals are continued based on the difference between the last two values in the list. [0, 1, ...) is shorthand for [0, 1, 2, 3, 4, 5] when applied to the interval [0, 5].

Constraints that do not have detents can leave the SnapTo and SnapRes portion of the constraint line blank.

When parts are adjusted, the current position (also called location) of the part may be displayed in the Part Information indicator. The designer of the machines may not want the minimum and maximum values displayed in the indicator to include the zero point. However, the interval must contain zero, because the part is assumed to be at the zero position when it is retrieved from disk. The offset value at the end of a constraint description specifies an offset to add to the interval before displaying the interval in the
indicator. For instance, if there is a machine part whose minimum should be 45 and maximum should be 90, the interval can be set to [0, 45] and the offset can be set to 45.

When a machine part with a linear constraint is selected with the mouse, the intersection of the pointer and the selected part is calculated. A line is created that goes through the point on the selected machine part and in the direction of the linear constraint. As the pointer moves, the part is moved in a way that keeps the intersection point as close to the mouse pointer as possible.

Figure 4.7 Part with linear constraint being dragged with the mouse.

Keeping the intersection point close to the pointer requires calculating the nearest point between the line that the pointer makes through the Virtual Workshop\(^1\) and the line along which the part is being translated.

The equations for the two lines are:

\[ L_1(t) = P_1 + V_1 t \quad \text{and} \quad L_2(s) = P_2 + V_2 s \]  

(Eqn. 4.1 and 4.2)

\( P_1 \) represents the initial intersection point of the mouse pointer and the machine part. \( V_1 \) is the direction of the linear constraint. \( P_2 \) corresponds to a point lying under the mouse pointer in the Virtual Workshop. \( V_2 \) is the direction of the line from the designers eye, through the pointer, into the workshop (perpendicular to the screen).

---

\(^1\)Even though the mouse pointer indicates just a single pixel or point on the screen, that point corresponds to a line in the 3 dimensional space projected onto the screen.

Section 4.3 Machine Representation and Control 81
The intersection of the two lines occurs when \( L_1(t) = L_2(s) \). Substituting the equations above gives

\[
P_1 + V_1 t = P_2 + V_2 s
\]  
(Eqn. 4.3)

Subtracting \( P_1 \) from both sides and crossing with \( V_2 \) gives

\[
(V_1 \times V_2) t = (P_2 - P_1) \times V_2
\]  
(Eqn. 4.4)

Dotting with \( (V_1 \times V_2) \) and dividing by \( |V_1 \times V_2|^2 \) gives

\[
t = \frac{P_2 - P_1}{V_1 \times V_2}
\]  
(Eqn. 4.5)

If \( |V_1 \times V_2|^2 = 0 \), the lines are parallel and there is no intersection. If the lines are skew, \( t \) represents the parameter of the point of closest approach.

Substituting \( t \) back into Eqn. 4.1 returns the point to use when calculating how far to move the machine part.

**Rotational Constraint**

A rotational constraint allows the part to move around an axis, but not along that axis. The position and direction specify the axis and normal of the plane of rotation.

The format of the rotational constraint line is:

Constraint : Rotational : Position : Normal : Interval : SnapTo :
SnapRes : Offset

The components of a rotational constraint are:

- **Constraint Type:** Rotational
- **Position:** \((x, y, z)\)
- **Normal:** \((x, y, z)\)
- **Interval:** \([\text{min}, \text{max}]\)
- **SnapTo:** \([\text{val1, val2, ...}]\)
- **SnapRes:** \(\text{resval}\)
- **Offset:** \(\text{offsetval}\)

*Type of constraint*  
*Center of rotation*  
*Axis of rotation*  
*Amount of available rotation*  
*SnapTo locations.*  
*Snap when (abs(valx - location) < resval)*  
*Add this to location for part indicator.*
The above constraint would appear in the machine file in the following format (with example values):

Constraint: Rotational: (0, 0, 0): (1, 0, 0): [-10, 10]: {6, 12, ...}: 0.1:10

The interval can be specified in such a way that multiple rotations are allowed. For example, [-2π, 16π] allows up to 9 rotations, with the default (0 radians) 1 rotation from the minimum.

The system begins adjusting a rotationally constrained part by creating a plane defined by the selection point and the rotation axis. The intersection of the rotation axis with the new plane is calculated and this is used as the center of rotation for the part.

![Figure 4.8 Rotation plane for selected part with rotational constraint.](image)

The symbols defined in Table 4.4 will be used in the rotation calculation.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Represents</th>
</tr>
</thead>
<tbody>
<tr>
<td>α</td>
<td>angle between initial position and new position of part</td>
</tr>
<tr>
<td>L</td>
<td>intersection plane</td>
</tr>
<tr>
<td>n</td>
<td>rotational axis and unit normal to intersection plane</td>
</tr>
<tr>
<td>PA</td>
<td>point defining axis of rotation</td>
</tr>
<tr>
<td>PC</td>
<td>center of rotation (on L)</td>
</tr>
<tr>
<td>P1</td>
<td>initial mouse-part intersection point.</td>
</tr>
<tr>
<td>P2</td>
<td>current mouse-plane intersection point</td>
</tr>
</tbody>
</table>

Table 4.4 Symbols used in calculation of rotations for part with rotational constraint.

Section 4.3 Machine Representation and Control  83
When the mouse pointer is clicked on a part with a rotational constraint, the intersection point of the mouse pointer and the part, \( P_1 \), is calculated. A plane, \( L \), is defined having unit normal \( \mathbf{n} \) and passing through \( P_1 \). The center of rotation, \( P_C \), for this part is calculated by finding the intersection between \( L \) and the rotational axis. The axis of rotation is defined by \( P_A \) and \( \mathbf{n} \). \( P_C \) is calculated as follows:

Calculate the distance \( d \) between the plane \( L \) and a parallel plane passing through \( P_A \).

\[
d = (P_1 - P_A) \cdot \mathbf{n}
\]

(Eqn. 4.6)

Next, calculate \( P_C \) by moving the distance \( d \) along \( \mathbf{n} \) from \( P_A \).

\[
P_C = P_A + d \mathbf{n}
\]

(Eqn. 4.7)

Calculate the intersection of the mouse pointer and the plane \( L \). \( \mathbf{V} \) is the direction of the line from the mouse pointer into the workshop, and \( P_O \) is a point on that line. Of course, if \( \mathbf{V} \cdot \mathbf{n} = 0 \), it is impossible to control the rotation of the machine part because the rotational axis is parallel to the screen. When this occurs, the part is not moved.

Calculate the new intersection point \( P_2 \).

\[
P_2 = P_O + \left( \frac{(P_A - P_O) \cdot \mathbf{n}}{\mathbf{V} \cdot \mathbf{n}} \right) \mathbf{V}
\]

(Eqn. 4.8)

Calculate the angle of rotation between the vectors \((P_1 - P_C)\) and \((P_2 - P_C)\). Start by calculating the sine and cosine of the angle.

\[
\mathbf{V}_{c1} = P_1 - P_C \quad \text{and} \quad \mathbf{V}_{c2} = P_2 - P_C
\]

(Eqn. 4.9 and 4.10)

\[
\sin(\alpha) = \frac{\mathbf{n} \cdot (\mathbf{V}_{c1} \times \mathbf{V}_{c2})}{|\mathbf{V}_{c1}| |\mathbf{V}_{c2}|}
\]

(Eqn. 4.11)

\[
\cos(\alpha) = \frac{\mathbf{V}_{c1} \cdot \mathbf{V}_{c2}}{|\mathbf{V}_{c1}| |\mathbf{V}_{c2}|}
\]

(Eqn. 4.12)

\[
\alpha = \tan^{-1} \left( \frac{\sin(\alpha)}{\cos(\alpha)} \right) = \tan^{-1} \left( \frac{\mathbf{n} \cdot (\mathbf{V}_{c1} \times \mathbf{V}_{c2})}{\mathbf{V}_{c1} \cdot \mathbf{V}_{c2}} \right)
\]

(Eqn. 4.13)
The \texttt{atan()} \textsuperscript{2} function for calculating the arc tangent returns an angle in the range of \(-\pi/2\) to \(\pi/2\). This limits how quickly the machine parts can be rotated because \(P_2\) must stay within 90 degrees of \(P_1\) during rotations. A function called \texttt{atan2()} \textsuperscript{3} uses the signs of the sine and cosine to determine the quadrant in which the angle is located. This function can return an angle value in the range of \(-\pi\) to \(\pi\), allowing a greater angle between the two points during rotation. However, the mouse movement speed is usually small enough so that the \texttt{atan2()} function is not necessary.

After each incremental rotation, \(P_1\) is assigned the value of \(P_2\) before beginning the next rotation calculation.

This method of calculating rotation values is extremely effective and has been found to be very intuitive as a method of directly controlling rotationally constrained parts. Two other control methods were tried in earlier versions of the workshop. One method looked at the mouse as it rotated around a predetermined point on the rotational axis which was not necessarily on the plane \(L\). Rotating a part around a predetermined point was found to be extremely difficult in circumstances where the point was far from the part being rotated, and on large rotating parts, the center of rotation cannot be near every point on the part’s surface. With a fixed center, the motion of the rotating part was both unpredictable and difficult to control (see Figure 4.9).

---

\textsuperscript{2}Function prototype: double \texttt{atan(double tan)}. This function is available in the math libraries of C compilers complying with the ANSI C standard.

\textsuperscript{3}Function prototype: double \texttt{atan2(double sine, double cosine)}. This function is also available in the math libraries of C compilers complying with the ANSI C standard.
A second method involved dragging graphical elements called sliders to set the rotation angle. Sliders are controllable but non-intuitive because the slider is not part of the rotating machine part. Direct manipulation of the part to be moved is always preferable when possible.

**Planar Constraint**

A planar constraint allows motion in a plane (corresponding to two linear constraints). Consequently, the data structure matches that of two linear constraints. The primary difference between a planar and two linear constraints is that a planar constraint can be applied to a single part, whereas a single part can only hold one linear constraint.

The format of the constraint line is:

```
```
The components of the planar constraint are:

<table>
<thead>
<tr>
<th>Constraint Type: Planar</th>
<th>Type of constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>DirectionA: (x, y, z)</td>
<td>First direction vector</td>
</tr>
<tr>
<td>DirectionB: (x, y, z)</td>
<td>Second direction vector (perp to first)</td>
</tr>
<tr>
<td>IntervalA: [minA, maxA]</td>
<td>First motion interval</td>
</tr>
<tr>
<td>IntervalB: [minB, maxB]</td>
<td>Second motion interval</td>
</tr>
<tr>
<td>SnapToA: [val1, val2, ...]</td>
<td>Snap to points</td>
</tr>
<tr>
<td>SnapResA: resval</td>
<td>Snap when (abs(location - val1) &lt; resval)</td>
</tr>
<tr>
<td>SnapToB: [val1, val2, ...]</td>
<td>Snap to points</td>
</tr>
<tr>
<td>SnapResB: resval</td>
<td>Snap when (abs(location - val1) &lt; resval)</td>
</tr>
<tr>
<td>OffsetA: offsetval</td>
<td>Added to interval for part indicator</td>
</tr>
<tr>
<td>OffsetB: offsetval</td>
<td>Added to interval for part indicator</td>
</tr>
</tbody>
</table>

The above constraint would appear in the machine file in the following format (with example values):

```
Constraint: Planar : (1,0,0) : (0,1,0) : [-10,10] : [-15,0] :
[-10,-5,...] : 0.2 : [-15,-14,...] : 0.05 : 0 : 0
```

When a plane-constrained part is selected with the mouse, a plane, \( L \), is created using the intersection point and the two direction vectors. The new plane is used to calculate the new position of the part as the mouse is moved around on the screen.

![Figure 4.10 Motion of a part with a planar constraint.](image)

As long as the plane \( L \) is not perpendicular to the screen, any line through the workshop normal to the screen will intersect the plane. The intersection point might not be within the intervals specified by the constraint, but that is easily checked after calculating the intersection point of the mouse pointer and the plane.

Plane \( L \) is created using the intersection point on the part and the A and B direction vectors \((V_A \text{ and } V_B)\) from the planar constraint. \( L \) is defined by a unit normal vector, \( n \), and the point \( P_1 \). \( n \) is calculated from \( V_A \text{ and } V_B \).
\[ n = \frac{V_A \times V_B}{|V_A \times V_B|} \]  
(Eqn. 4.14)

The line from the mouse pointer going through the workshop is represented by a direction vector, \( V \), and a point on the line, \( P_O \). The intersection of this line with the plane is \( P_2 \).

\[ P_2 = P_O + \left( \frac{(P_1 - P_O) \cdot n}{V \cdot n} \right) V \]  
(Eqn. 4.15)

The change in location from \( P_1 \) to \( P_2 \) is calculated from \( P_1 \) and \( P_2 \).

\[ v_A = \frac{V_A}{|V_A|} \quad \text{and} \quad v_B = \frac{V_B}{|V_B|} \]  
(Eqn. 4.16 and 4.17)

\[ dV_A = (v_A \cdot (P_2 - P_1))v_A \]  
(Eqn. 4.18)

\[ dV_B = (v_B \cdot (P_2 - P_1))v_B \]  
(Eqn. 4.19)

\( dV_A \) and \( dV_B \) are the components of the translation in the \( V_A \) and \( V_B \) directions respectively and are used to reposition the machine part and update the locations in the constraint.

**Machine-Workpiece Interaction**

Machine parts have planes that constrain the motion of the workpiece in the machine. The planes are called *workplanes* and are the surfaces against which the workpiece rests. A table saw has one main workplane - the surface of the table. When a fence like the ripping fence is attached, the workpiece is additionally constrained to be held against the fence which contains a second workplane. Each machine may have up to three workplanes active at one time. A single workplane constrains the workpiece to move on a plane. Two workplanes constrain the workpiece motion to the straight line that corresponds to the intersection of the planes. Adding a third workplane fixes the workpiece in the machine. When three workplanes are active, the workpiece cannot be move independently. However, moving one of the parts containing a workplane will move the workpiece.

**Positioning**

The position of the workpiece depends on the number and location of the active workplanes in the machine.
When one workplane is active, the workpiece is free to move around on that plane. When the workpiece is selected and dragged with the mouse, the motion is calculated the same way motion for a part with a planar constraint is calculated. A plane is created that contains the pointer-workpiece intersection point and has the same normal direction as the workplane. The workpiece is moved on that plane in a way that keeps the intersection point on the workpiece directly under the mouse pointer until the mouse button is released.

When there are two active workplanes in the machine, the workpiece is constrained to move in a straight line. The motion of the workpiece occurs the same way as an unbounded linearly constrained part would move. For a description of linearly constrained motion, see "Linear Constraint" on page 79.

Finally, when three workplanes are active in the machine, the workpiece can only move when one or more of the parts containing the workplanes is moved.

The workpiece is always adjusted to maintain contact with the workplanes. If the table on a drill press is raised, the workplane that it contains is also lifted by the same amount. The illusion to the designer is that the workpiece is resting on the table.

**Cutting**

Cutting, or removing material from a workpiece, occurs every time there is an intersection between the swept blade and the workpiece. In other words, when a blade is swept along a path and that path intersects the workpiece, a swept blade is generated and subtracted from the workpiece. Conversely, when the workpiece is moved (as in a table saw) and its path intersects the blade, the swept blade is also generated and subtracted from the workpiece. In this case, the motion of the workpiece is inverted and applied to the blade to generate the swept geometry. A full description of how the swept geometry is created is contained in the section 4.6.

Checking for workpiece-blade intersections can be computationally expensive and is accelerated using bounding boxes. A bounding box is a box defined by two points that describe a volume completely enclosing a body. The description of the box only requires two points because it is always aligned with the current coordinate system. The first point, or minimum point, contains x, y, and z values that are less than or equal to any point contained in the volume of the body. The second (maximum) point contains values that are greater than any contained in the volume of the body. A sphere with a radius equal to 1 centered on the origin is completely contained in a bounding box [(−1, −1, −1), (1, 1, 1)].
When either the workpiece or the blade is moved, the bounding boxes are created for the workpiece and the blade both before and after the movement. The before and after bounding boxes are combined to create a single bounding box that encloses both the before and after positions of each object (see Figure 4.11).

![Figure 4.11 Intersection (a) and bounding boxes (b) of blade and workpiece.](image)

If those two bounding boxes intersect, the sweep is calculated and subtracted from the workpiece. Otherwise, there is no possibility of intersection between the two objects, and no intersection is calculated.

![Figure 4.12 Blade swept in arc intersects workpiece, but before and after bounding boxes do not always indicate intersection.](image)

Special care must be taken for the case where the blade is moved in a complex motion that sweeps through the workpiece, but leaves the blade in a position where the box containing the before and after bounding boxes does not intersect the workpiece bounding box (see Figure 4.12). For non-linear motions of the blade, the blade sweep is calculated and the
blade sweep bounding box is compared to that of the workpiece. The sweep is subtracted from the workpiece if there is an intersection.
4.4 Part Representation

Data Structure

The part description data required for each machine part is listed in Table 4.5.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name, pathname</td>
<td>Name and pathname of part.</td>
</tr>
<tr>
<td>Descendants list</td>
<td>Pointers to the child parts of this part.</td>
</tr>
<tr>
<td>Lock</td>
<td>Lock for this part.</td>
</tr>
<tr>
<td>Lock children</td>
<td>List of locks attached to this part</td>
</tr>
<tr>
<td>Visibility information</td>
<td>Visible and hide-able Boolean values.</td>
</tr>
<tr>
<td>Body</td>
<td>Geometric information (appearance and geometry of the body)</td>
</tr>
<tr>
<td>Pmesh</td>
<td>Display polygons, points, and normals that get drawn on the screen.</td>
</tr>
<tr>
<td>Driven part</td>
<td>Part driven by this part.</td>
</tr>
<tr>
<td>Constraint</td>
<td>Allowable motion information</td>
</tr>
<tr>
<td>Parent</td>
<td>Pointer to parent part (NULL if machine is parent).</td>
</tr>
<tr>
<td>Machine</td>
<td>Pointer to machine containing part.</td>
</tr>
<tr>
<td>Conflicts</td>
<td>List of conflicting parts.</td>
</tr>
<tr>
<td>Workplane</td>
<td>Plane for attaching to workpiece and information about parts whose workplanes conflict with this one.</td>
</tr>
<tr>
<td>Material</td>
<td>Information for displaying part.</td>
</tr>
<tr>
<td>Blade</td>
<td>Logical value indicating if blade is attached to this part.</td>
</tr>
<tr>
<td>Linked list information</td>
<td>Keeps track of all of the child parts for a machine or parent part.</td>
</tr>
</tbody>
</table>

Table 4.5 Information stored in the part.
Name, Pathname

The name and pathname of the part are used when displaying information about a part, and also for resolving references between parts.

When a part is created, it must be connected to several other parts, including its parent part, its descendant parts, any part that it drives, etc. Since the linkage is so complex, it is impossible to create the parts in a way which allows the workshop to resolve all of the complex links without using a naming scheme.

Each part’s name and pathname completely describe its location in a machine. Parts can have the same name, as long as they are not attached to the same parent part.

Part names and paths are similar to UNIX filenames. Each part above a part is in the pathname of that part. The part names in the path are separated by periods. The Rip Fence in the Table Saw is called “Base.Table.Rip Fence”. In other words, the rip fence is attached to the table which is connected to the base of the table saw.

As part information is read in from the machine file, the name of each part is added to the part along with its pathname. All of the parts are kept in a list of parts for that machine. As soon as all of the parts have been created, the links are resolved using the partnames and paths. For instance, if one part drives another part, the driven part’s name is stored in the part until the pointer to the driven part is found and copied into the driver part.

The pathname of the part is redundant once all of the links have been established in the machine. It can be retrieved by following the pointers to the other parts. However, the pathname of the part is retained for searches, rather than visiting each part in the path to reconstruct the pathname every time it is needed.

Descendants List

Each part has a list of descendants. This list contains the pointers to all of the parts attached to this part. The workshop can traverse the parts in a machine using this list of children and the pointers to the part’s parents.
Lock

A lock is used to fix and release the constraint attached to a part. Even if a part is allowed to move, the lock keeps it from moving when it is grabbed and dragged with the mouse pointer.

If this part has a fixed constraint, or cannot be locked, the pointer to the lock is empty (NULL).

Lock Children

A part can hold locks for any other part on the machine. For instance, the lock on a table saw rip fence is located on the fence and moves with the fence. The lock on the blade guard on a band saw is attached to the main part of the machine, and not to the guard itself.

When a part is moved, all of its children move with it. However, the lock is not moved automatically unless it is attached to that part.

Visibility Information

Each part of the machine contains information about whether it can be hidden and whether or not it is visible. These two variables are used for turning on and off fixtures and fences in the machine.

Parts of the machine that cannot be used at the same time as other parts must be hidden, or turned off, using the visible variable. Parts that must always be displayed, such as the table saw base, can't be made invisible and the hide-able variable is set to FALSE to indicate that fact.

Body

The body is an ACIS solid model which contains the geometry of the part. The solid model is stored in the PartGeometry directory and its name is based on the part name, its path, and the machine name. When the part is created, the body is read from disk and displayed when appropriate.

The body is used for intersection calculations and creates the visual image of the part on the machine.
Pmesh

The Pmesh contains a list of polygons, points, and normals for displaying a polyhedral approximation of the ACIS solid.

Driven Part

The driven part is a part that moves when this part moves. Driving and driven parts are paired, and motion of either part will move the other part. However, in the input file, only one part is specified as driving another part and the connection the other way is made automatically.

When two parts are connected as driving pairs, their constraints are matched during motion. If one part is moved from its minimum to its maximum position (rotational or linear) the driven part is also moved from its minimum to its maximum position. The only requirement is that the zero point in the interval be the same percentage along the interval in both parts.

\[
\left( \frac{-\text{min}_1}{\text{max}_1 - \text{min}_1} \right) = \left( \frac{-\text{min}_2}{\text{max}_2 - \text{min}_2} \right)
\]  \hspace{1cm} (Eqn. 4.20)

If the driver has a constraint interval of [-1, 10] then one valid constraint interval for the driven part could be [-2, 20]. The interval [-10, 1] is not valid. The intervals must match so that a part cannot be moved to an invalid position even when driven by another part.

Constraint

Each part contains a constraint. A detailed discussion of the constraints is contained in section 4.3, "Machine Representation and Control".

Parent

The parent of a part is the part that holds this part. Parts mounted directly on the machine have no parent and this pointer is empty (NULL).

Machine

This is a pointer to the machine containing the part.
Conflicts

Several parts in machines may conflict with other parts because of geometric interferences or for safety reasons. This list of parts tells the Virtual Workshop which parts cannot be visible at the same time as this part. If this part is made visible, all of the parts in this list and their children are hidden.

Workplane

Each part can contain a single workplane. The workplane constrains the workpieces motion.

Material

The material describes the visual appearance of the surface of the part.

Blade

The blade in a machine is attached to one part. If this part holds the blade, it is indicated in this variable in the part object.

Linked List Information

This information allows the workshop to keep track of all of the parts in the machine.
4.5 Lock Representation

<table>
<thead>
<tr>
<th>Data</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the lock.</td>
</tr>
<tr>
<td>Boolean value</td>
<td>Indicates current state of the lock (on or off)</td>
</tr>
<tr>
<td>On body</td>
<td>Body that represents the lock’s on position</td>
</tr>
<tr>
<td>Off body</td>
<td>Body that represents the lock’s off position</td>
</tr>
<tr>
<td>Containing part</td>
<td>Pointer to the part to which this lock is attached.</td>
</tr>
<tr>
<td>Controlled part</td>
<td>Pointer to part controlled by this lock.</td>
</tr>
<tr>
<td>Materials</td>
<td>Material information for displaying the lock.</td>
</tr>
<tr>
<td>Lock linked list</td>
<td>List of all the locks in the machine are maintained in this linked list.</td>
</tr>
<tr>
<td>Machine</td>
<td>Pointer to the parent machine holding this lock.</td>
</tr>
</tbody>
</table>

Table 4.6 Description of lock data.

Name

The name of a lock is used to provide access to the lock directly. When reading in a machine input file and creating the machine, the name is used to attach parts and locks together within the machine.

Boolean Value

This value indicates whether the lock is on or off. When the lock is on, the part controlled by that lock cannot be moved.

On and Off Bodies

Since the lock can only be in one of two states, it is not necessary to use constraints to control the positions of the locks. Each lock simply has two separate bodies. One body corresponds to the lock in the off position and the other corresponds to the lock in the on position. The body displayed reflects the state of the lock.
Containing Part

Each lock is attached to a part. When that part, called the containing part, moves, the lock is transformed with it. Both the on and off bodies are transformed with the part so that they will appear in the correct place on the part when they are displayed. One of the lock bodies is always displayed unless the part containing the lock is hidden.

Controlled Part

The controlled part is the part whose motion is restricted when the lock is turned on.

Materials

Two materials are specified in the machine input file for a lock. One material is for the on body and the other for the off body. If the person designing machines wants the lock to be red when it is activated and green when it is off, a red material can be specified for the on body and a green material for the off body and the bodies will be displayed correctly.

Lock Linked List

There is a linked list in the machine that keeps track of all of the locks in the machine. The list is used for searching for a specific lock from in a machine.

Machine

- This is a pointer to the machine containing the lock.
4.6 Blade Representation

Blade is a generic term that will be used to mean any bit, blade, tool, or cutter that is used as a cutting tool by a machine. They may be used in only one type of machine (such as the blade in a band saw) or may be interchangeable between machines (like saw blades that can be used in table saws or radial arm saws). The goal of creating a generic blade representation is to allow the blade to keep track of its own limits and uses and be transportable between machines.

The task of a blade is to generate a swept volume that would represent the volume swept out by motion of the blade with respect to the workpiece. The blade must also be able to determine whether or not it can be inserted in a machine and which types of motions are allowed for sweeping.

Blades can only be used in a machine that has the proper mount. A table saw holds its blades on an arbor with a certain diameter and maximum blade width. Any blade that will fit on that arbor within the width constraint can be mounted on the table saw.

The blade, bit, or cutter contains the following information:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mount information</td>
<td>The mount information includes the type of mount (arbor, chuck, collet, flywheel, or jaws), and all necessary geometric information.</td>
</tr>
<tr>
<td>Solid model (body)</td>
<td>A visual representation of the blade that is used for displaying the blade and is subtracted from the workpiece when a cut is made.</td>
</tr>
<tr>
<td>Profile array</td>
<td>Array of sweep profiles for generating swept volumes of blades.</td>
</tr>
</tbody>
</table>

Table 4.7 Information contained in a blade.

**Mount Information:**

The following line is the mount information contained in the Table Saw machine description file:

Blade Mount: Arbor, 0.625 in, 1 in, unkeyed
The information contained in the Table Saw description file indicates that blades that use unkeyed arbor with a diameter of 5/8 in can be used on the table saw. The 1 inch indicates a maximum blade width. Arbors can be unkeyed, keyed, or splined. No other information about the mount is stored or required for this implementation.

The mount information stored in the “Dado Blade” blade description file is:

Mount: Arbor, 0.625 in, .875 in, unkeyed

The Dado blade information file indicates that it is to be mounted on a 5/8 in arbor and it’s thickness is 7/8 of an inch. The blade is not keyed. Since the information in the dado blade description file matches or is within bounds of the table saw mount description, the blade is allowed to be mounted on the machine.

Mounting Blades on Machines

Mounting a blade on a machine is not as simple as just displaying the blade. The blade must be positioned in the machine correctly. In order to determine how to transform the blade into the right position to be accepted into the machine, the position and orientation information must be included in both the machine file and the blade file. The difficulty of positioning the blade is compounded by the fact that the part which holds the blade may be rotated or moved in the course of making adjustments to the machine. A table saw blade might be raised, rotated to an angle or both. The machine must keep track of how the blade mount has been transformed in order to mount new blades properly.

The machine contains a part on which the blade is mounted. In the case of the table saw, the blade is mounted on the arbor. The machine file contains a line like the following:

Blade Location:
Base.Blade Height Adjustment.Blade Angle Adjustment.Arbor : (0,0,-5) :
(0,-1,0) : (1,0,0)

The first part of the line specifies the path and name of the part that the blade is to be mounted on. The path and part name are essential because if any of the listed parts are moved (Base, Blade Height Adjustment, Blade Angle Adjustment, or Arbor) the blade and blade mount information will also be transformed appropriately. The second part of the ‘Blade Location’ line (between the second and third colons) indicate the origin of the blade when it is in the machine. The third and fourth sections contain the primary and secondary axes to specify what rotations must occur to transform the blade from its original position and orientation to the machine mount position and orientation.
The blade file contains two lines that correspond to the information in the machine file.

Position: \((0,0,-5)\)
Axis: \((0,-1,0); (1,0,0)\)

'Position' indicates the location of the origin of the blade. 'Axis' indicates the primary and secondary axis of the blade so that the proper rotations can be calculated for inserting the blade into the machine.

In the example above, the origin and principle axes of the blade correspond to those of the machine. However, if certain parts on the machine are moved, the origin and axes may no longer coincide. For instance, if the blade height adjuster on the table saw is raised 2 inches, the origin in the machine will now be \((0,0,-3)\) and a new blade read in by the system will have to be translated into the correct position.

When a blade is read into the system, the computer checks to see if the mount information matches. The type of mount must match exactly (arbor = arbor, flywheel = flywheel) and the diameters and maximums must correspond. If the information about the mount matches, then the blade is created. However, it must still be mounted on the machine.

In order to mount the blade, the program creates two transformation matrices, \(T_b\) and \(T_m\). \(T_b\) is calculated as the matrix that will move the origin \((0,0,0)\) and principal axes \((1,0,0)\) and \((0,1,0)\) to the position and axes specified in the blade input file. \(T_m\) is a corresponding matrix for the machine position and orientation information. In order to install the blade in the machine, the blade is transformed by the matrix \(T\):

\[
T = [T_b]^{-1}T_m
\]  
(Eqn. 4.21)

To transform a blade, the solid model (or body) is transformed (all points, edges, surfaces, etc.) to put the body in the correct orientation. Also, all other geometric elements of the blade are transformed, including the visual representation of the body, the array of profiles for sweep generation and the mount information stored in the blade (which, after transformation, matches the mount information in the machine).

Profiles

Profiles are the part of the blade that specify how the sweep is created and in which directions the blade is allowed to cut. A blade must have at least one profile, but can contain as many profiles as necessary to completely define its cutting capabilities. A drill bit can only cut in a linear direction along the axis of the bit and therefore only contains a
single circular profile. A flat end milling cutter can cut either straight along the axis of the cutter, or in any direction in the plane whose normal is along the axis of the cutter.

A profile consists of a planar outline, a normal to the outline plane, and a cut constraint. The cut constraint (linear, planar, curvilinear, or spherical) specifies the direction and maximum curvature along which the outline can be swept. The normal of the profile outline is simply there for convenience, since it can easily be calculated from the profile.

The cut constraint can be one of four types: linear, curvilinear, planar, or spherical. Linear cut constraints are used for things like drills, table saws, and other machines where cutting is only allowed in a straight line. Planar constraints are useful for machines like mills, routers, and other cutters where the edges of the cutter are also available for cutting operations. A blade with a planar constraint allows the cut to occur in any direction in a plane. A spherical constraint is available for cutters like ball end mills that can be used in 5 axis milling machines or numerically controlled 3 axis mills. The curvilinear constraint is a special version of the linear constraint reserved for machines like the band saw, where some curvature is allowed in the cut, but the blade will bind if the curve is too tight.

A milling machine cutter is a good example of a blade with two profiles, one planar and one linear. Keep in mind that the term blade will be used interchangeably with tool, bit, and cutter as a generic term referring to the cutting device that is placed in a machine. The milling cutters: for the milling machine in the Virtual Workshop can cut directly down into a part, just like a drill, or in a side to side fashion. A circular profile with a linear constraint takes care of the drilling motion, and a rectilinear profile with a planar constraint creates the side to side sweeps (see Figure 4.13).
When either the blade or the workpiece is moved, the system checks to see if an intersection occurs during the move or if the blade and workpiece pass near each other. If there is even a possibility of intersection, then the blade is swept and subtracted from the workpiece.

**Sweeping Blades**

In order to generate a swept solid for a cutting operation, both a profile to sweep and a wire defining the sweep path are required. The sweep path is generated by recording the motion of the workpiece with respect to the blade. The profile is selected from the profiles available in the blade. Which profile is selected from the blade depends on the sweep path and profile constraints.

If a workpiece is moved past a blade, a sweep path is generated using the motion of the workpiece as a guide. If the workpiece is moved in a straight line, the sweep path is also a straight line. If the motion is circular or follows a spline, the sweep path is an arc or spline. The endpoint of the sweep path is moved to the centroid of the blade and the appropriate profile is chosen from the blade.

One by one, the system compares the profiles described in the blade with the sweep path. Table 4.8 describes the rules by which profiles are selected from blades.
Cut Constraint

<table>
<thead>
<tr>
<th>Curve Type</th>
<th>Linear</th>
<th>Curvilinear</th>
<th>Planar</th>
<th>Spherical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Straight</td>
<td>( t_{SP} = t_{CC} )</td>
<td>( t_{SP} = t_{CC} )</td>
<td>(</td>
<td>t_{SP} \cdot n_{CP}</td>
</tr>
<tr>
<td>Planar Curve</td>
<td>Never valid</td>
<td>( t_{SP} = t_{CC} )</td>
<td>( n_{C} = n_{CP} )</td>
<td>Always valid</td>
</tr>
<tr>
<td>3 Dimensional Curve</td>
<td>Never valid</td>
<td>Never valid</td>
<td>Never valid</td>
<td>Always valid</td>
</tr>
</tbody>
</table>

Symbols:
- \( t_{SP} \): Unit vector tangent to sweep path at beginning of path
- \( t_{CC} \): Unit vector tangent to cut constraint direction
- \( n_{CP} \): Unit normal to plane of planar cut constraint
- \( n_{C} \): Unit normal to plane containing sweep curve
- \( \kappa_{MAX} \): Maximum curvature of sweep curve
- \( \kappa_{CC} \): Maximum allowable curvature of curvilinear cut constraint

Table 4.8 Rules for selecting profiles from blades.

The different types of cut constraints correspond to different levels of symmetry in a blade. Drill bits, band saw blades, and table saw blades are only designed to cut in one direction. However, a flat end mill cutter can cut along its axis and also from side to side. The cutter has a rotational symmetry about its axis and so a single profile can represent cutting motions in all directions normal to that axis. A planar constraint in the cutter indicates that rotating the profile before sweeping will not change the outline of the cut and is therefore permissible. The spherical constraint will only work for objects with spherical symmetry, of which there is only one: a sphere. A ball end mill can be made with a spherical constraint, as long as the designer does not try to cut away material using the shank of the cutter.

When a linearly constrained profile is selected with a straight sweep path, the profile is simply swept along the path to create the geometry of the swept blade. A rectangular profile will sweep to a block and a circular profile will create a cylindrical shape. More complex profiles generate correspondingly more complex solids.

Things get much more confusing when generating a sweep for a profile which is associated with a planar constraint. Planar constraints have a normal, \( n_{CP} \), which is the normal to the plane in which a cutting motion is allowed. The profile associated with the planar
constraint also has a normal, \( \mathbf{n}_p \). However, the profile's normal indicates the normal of the plane containing the profile outline. The profile outline plane is defined by any three points on the outline \((P_1, P_2, P_3)\) and can be calculated using the following equation:

\[
\mathbf{n}_p = (P_1 - P_2) \times (P_3 - P_2)
\]  
(Eqn. 4.22)

The profile must be symmetric about the blades rotational axis (which corresponds to \( \mathbf{n}_{CP} \)) or else incorrect results will be returned for the swept profile. Since the profile is symmetric, which three points on the outline are used to calculate the normal is unimportant as long as the points are not collinear.

When a profile with a planar constraint is chosen, it is possible that the tangent direction of the start of the sweep path, \( t_{SW} \), is not in the direction of \( \mathbf{n}_p \). It is then necessary to rotate the profile so that \( \mathbf{n}_p \) points in the same direction as \( t_{SW} \) before beginning the sweep. Of course, the profile can only be rotated around an axis aligned with \( \mathbf{n}_{CP} \) so that the \( \mathbf{n}_p \) is not transformed out of the plane of the constraint. After \( \mathbf{n}_p \) is rotated into position, the sweep is created using one of ACIS's sweep generation commands (\texttt{api_sw_face_wire()}). This command sweeps the profile along the sweep path. Once the sweep has been generated, it can be subtracted from the workpiece using the ACIS function \texttt{api_subtract()}.

Remember that the profile knows nothing about the shape of the blade in 3 dimensions. It simply holds profile and sweep information. After the swept profile has been subtracted from the workpiece, the body of the blade is subtracted from the workpiece, to account for the 3 dimensional geometry of the blade.

**Blade Text Representation**

Each blade is stored in two separate files. One file contains information about the mounts, profiles, and cut constraints. The other file contains the ACIS model which is used for visualization and cutting. These files could easily be combined using because they are both text based files, but they have been left separate for convenience.

The ".blade" file contains the following information:

<table>
<thead>
<tr>
<th>Data</th>
<th>Contents (example):</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>7-8 Dado Blade</td>
</tr>
<tr>
<td>Brand</td>
<td>KraftsMan</td>
</tr>
</tbody>
</table>

Section 4.6 Blade Representation 105
Style Dado Blade
Material Steel
Mount Arbor, 0.625 in, .875 in, unkeyed
Axis (0,-1,0); (1,0,0)
Position (0,0,-5)
Wire (0,-1,0); (0,0,0); 0; (0,0,-10); 0; (-.875,0,-10); 0; (-.875,0,0); 0; (0,0,0)
Constraint Linear; (0, -1, 0)

Table 4.9 Example data from blade file.

<table>
<thead>
<tr>
<th>Data</th>
<th>Contents (Purpose and Description)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name field is the same as the name of the file in which the blade is stored.</td>
</tr>
<tr>
<td>Brand, Style</td>
<td>Brand and style are primarily used for convenience and to provide information to the designer.</td>
</tr>
<tr>
<td>Material</td>
<td>Currently, this field is only used for display purposes. When the blade is drawn on the screen, the material specified here is mapped to a material defined in the lighting model and the blade is displayed using that material.</td>
</tr>
<tr>
<td>Mount</td>
<td>The mount specifies the type of mount and dimensions for determining whether or not a blade can be put in a machine.</td>
</tr>
<tr>
<td>Axis</td>
<td>The axes specified in this field are the principle axes of the solid model.</td>
</tr>
<tr>
<td>Position</td>
<td>This position is the origin of the solid model. This field and the axes specified in the Axis field allow an arbitrary orientation of the solid model. They are primarily for the convenience of the person designing the blades.</td>
</tr>
<tr>
<td>Wire</td>
<td>This field specifies the outline of the profile. See below for a more detailed description of the profile outlines and the format of the Wire field. Wires must come before Constraints in the blade input file and must always be followed immediately by a constraint description.</td>
</tr>
<tr>
<td>Constraint</td>
<td>The constraint field specifies which type of cut constraint should be associated with the profile described in the Wire field. See below for a more detailed description of the cut constraints. Constraints must always be preceded by a Wire description.</td>
</tr>
</tbody>
</table>

Table 4.10 Description of data found in blade input file.
Blade Names

The name field in the blade file is redundant because the name always matches the name of the blade input file (minus the "blade" extension). The name field is also the source of the name of the ACIS file containing the solid model of the blade. Spaces (' ') in the name are converted to underscores ('_') before it is used as a filename\(^4\).

Profile Representation

The wire field in the blade file contains a description of the blade outline or profile. The format of the profile description is:

\[ n_p; \ P_1; \ C_{V1}; \ P_2; \ C_{V2}; \ P_3; \ C_{V3}; \ P_4; \ldots \]

The first vector specified is the normal of the outline, \( n_p \). The normal is specified as three floating point numbers in parentheses separated by commas (\( n_{px}, n_{py}, n_{pz} \)). The points \( (P_X) \) are specified with the same format and the values \( (C_{VX}) \) are simply floating point numbers without parentheses. As mentioned earlier, the normal could be calculated from points, but it is required in the input file as a validity check on the profile.

The points \( (P_X) \) and values \( (C_{VX}) \) are the vertices of the outline interleaved with numbers that indicate what type of arc to insert between the vertices. Each pair of vertices may have a straight line or an arc between them. The values \( (C_{VX}) \) between the points indicates the rotation of the tangent vector in moving from one point to the next. A zero indicates that a straight line should be placed between the two points. 90 degrees (or \( \pi/2 \) radians) will create an arc that bends 90 degrees from the first point to the second. Naturally, the arc lies in the plane of the profile. The value specifying the arc can range from -180 to 180 degrees.

At least three points must be specified in the outline description and there is always one fewer value \( (C_{VX}) \) than number of points \( (P_X) \). The values between the points are similar to the values used in the kwire command of the ACIS test harness.

The following is an example of a circular profile description:

\[ \text{Wire: } (0, 0, 1); \ (1,0,0); \ 180 \text{ deg; } (-1, 0, 0); \ \text{pi rad; } (1,0,0) \]

\(^4\)Although files with spaces in their names can exist in the UNIX file system, they are inconvenient to deal with on the command line.
The circular profile lies on the XY plane, with a radius of one inch (inches are the default units). As in the example, the first and last points in a profile must always be coincident so that the wire is closed because open wires do not always create solids when they are swept. Also notice that the values \( C_{vX} \) can be specified in either degrees or radians.

The use of the above format limits the blades to profiles made up of circular arcs and lines. Although having only arcs and lines available for a profile is overly restrictive, it was adequate for this research system.

**Cut Constraints**

Cut constraints may be linear, curvilinear, planar, or spherical. Several examples of valid cut constraints follow:

- Constraint: Spherical
- Constraint: Planar; \((1, 0, 0)\)
- Constraint: Linear; \((0, 0, 1)\)
- Constraint: Curvilinear; \((0, -1, 0)\); 0.5

Spherical constraints require no extra values to be specified. A planar constraint requires a vector corresponding to the normal of the plane. Linear and curvilinear constraints require a vector which specifies the direction of the constraint. Curvilinear constraints also require a maximum curvature value.
4.7 Stock Representation

One of the fundamental differences in the way workshops and CAD systems operate is the way material is selected. In a CAD system, material specification can easily be forgotten or added as an afterthought. In a real workshop, without material or stock, there is nothing to do.

In a workshop metaphor CAD system, what you make depends on what materials are available. In most current CAD systems, no stock or material access is assumed. A designer always starts working from a blank screen by creating a line or a solid primitive. Some CAD systems, especially of 2D drawing systems, allow the designer to select designs from a library of static, predrawn parts.

Stock in the Virtual Workshop contains the following information:

<table>
<thead>
<tr>
<th>Data</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the workpiece. This is a descriptive name that identifies the stock.</td>
</tr>
<tr>
<td>Cost</td>
<td>Cost can be used directly to calculate the total cost of the stock for the entire project. The data used for the Virtual Workshop was taken from real lumber yards in the Cambridge, Massachusetts area.</td>
</tr>
<tr>
<td>Material properties</td>
<td>Material properties are kept with the stock for convenience. They were originally used for dynamic simulation calculations.</td>
</tr>
<tr>
<td>Geometry</td>
<td>The geometry is an ACIS solid model.</td>
</tr>
</tbody>
</table>

Table 4.11 Description of information specific to stock in Virtual Workshop.

Having direct access to stock available in stockrooms or from material vendors is very useful. Projects can be based on parts that are readily accessible. Knowing the cost of each piece can help to keep overall project cost down by increasing the designers sensitivity to price, or even just making the designer aware of price.

When stock is selected for addition to the project, a workpiece is created using the information contained in the stock. It is then added to the project.
4.8 Workpiece Representation

Workpieces are one of the primary focal points of the workshop environment. Workpieces in the Virtual Workshop can be cut and assembled. All of the information necessary for working with multiple workpieces within a project are contained in the workpiece data structure.

A workpiece contains the following information:

<table>
<thead>
<tr>
<th>Data</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>This is the name of the workpiece. Names can be any valid string of characters.</td>
</tr>
<tr>
<td>Body</td>
<td>This is an ACIS body. The body is the solid model and is used for cutting and display operations.</td>
</tr>
<tr>
<td>Pmesh</td>
<td>The pmesh is a list of polygons used for displaying the solid model.</td>
</tr>
<tr>
<td>Project</td>
<td>This is the project that contains the workpiece.</td>
</tr>
<tr>
<td>Type indicator</td>
<td>Each workpiece can be either regular, scrap, or trash.</td>
</tr>
<tr>
<td>Mass properties</td>
<td>Each workpiece may contain mass property information of its body.</td>
</tr>
<tr>
<td>Center of gravity</td>
<td>The center of gravity is used to indicate the location of the workpiece in three dimensional space.</td>
</tr>
<tr>
<td>Transformation</td>
<td>The transformation is the accumulation of all of the motions that the workpiece has gone through since its creation.</td>
</tr>
<tr>
<td>Material</td>
<td>This field contains information used when drawing the workpiece.</td>
</tr>
<tr>
<td>Workplanes</td>
<td>When workpieces are put in machines, the workplanes of the machine are stored here for convenience.</td>
</tr>
<tr>
<td>Linked list</td>
<td>Each workpiece is kept in a linked list in the parent project.</td>
</tr>
<tr>
<td>information</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.12 Information stored with each workpiece.

**Workpiece Name**

Currently, the name of the workpiece is created automatically by the Virtual Workshop. When a workpiece is cut into separate pieces, each newly created workpiece retains the
same name with a slight modification to distinguish it from the other workpieces in the project. Eventually, the designer will be able to name the pieces as an aid to keeping track of each element in the project.

Body

The most vital piece of information stored in the workpiece is the body. The body represents the geometry and appearance of the workpiece. Each workpiece only contains one body and each body only has one ACIS lump. When workpieces are cut into two or more pieces, the lumps from the body are separated into individual bodies and new workpieces are created to hold those bodies.

The body in the workpiece really holds two types of information. The solid part of the body holds the current geometry of the body. The wireframe part contains a list of wires that make up the marks on the body. Marks are discussed in more detail in section 4.11, "Mark Representation”.

Surfaces in an ACIS body can contain attributes indicating how that surface was made, including both process and machine setup information. Attributes may be used in a future version of the Virtual Workshop to track tolerances and machine use for each workpiece.

Pmesh

The Pmesh is the list of polygons, including points and normals, used for displaying the body. They are generated from the body using the faceting capabilities provided in the ACIS faceter interface routines. Each time a machine cuts the body, the pmesh is regenerated so that the display reflects the new geometry of the workpiece.

Project

Each workpiece belongs to a project. A project is simply a list of workpieces that are to be used for a common purpose. The project pointer stores the address of the workpiece’s project.

Type indicator

Workpieces can be one of three types: regular, scrap, or trash. Scrap workpieces are pieces that are not immediately useful, but may be useful in the future and should not be discarded. Trash workpieces can be thrown away because they are no longer needed.
These three classifications are a way of ranking workpieces by usefulness. When a designer is selecting workpieces in the Virtual Workshop, only workpieces that have some significance may be of interest. Scrap workpieces can be left out of the selection list unless specifically requested.

**Mass Properties**

Mass properties are useful in many circumstances. For instance, the volume and density of a workpiece can be used to calculate its weight.

**Center of Gravity**

The center of gravity is used as the reference location point of the workpiece. When the Workpiece Location Indicator is visible, the location of the center of gravity is displayed along with the distance between the current location and the previous location.

**Transformation**

A transformation is stored with the workpiece which is an accumulation of all of the workpiece’s transformations since it was created. This transformation can be inverted and used to reposition the workpiece in its original orientation and placement.

**Material**

This information in the workpiece indicates the substance of the workpiece and what the workpiece should look like.

**Workplanes**

When a workpiece is in a machine, it is snapped to the current workplanes in that machine. Workplanes may be attached to tables, fences, or other fixtures on the machine. Rather than searching the machine for the current workplanes every time the workshop is redrawn, pointers to the workplanes are stored in the workpiece. The number and location of the current workplanes constrain and guide the workpiece within the machine. When a machine part is moved that contains one of the current workplanes, the workpiece is moved with it.
Linked list information

This linked list is used to keep track of all of the workpieces. Any workpiece can be searched for by name by traversing this list.

Assembling Workpieces

Simple assembly operations have been built into the Virtual Workshop to allow designers to put workpieces together after machining. The designer can select a face of the current workpiece and attach another workpiece to that face. This attachment works like a planar constraint between a workpiece and the machine (when 1 workplane is active). When the attached workpiece is in the correct location with respect to the first workpiece, it can be fixed in place. Multiple attachment constraints can be used to line up two edges or to make two points coincident.
4.9 Project Representation

A project is simply a group of workpieces.

A project contains the following information:

<table>
<thead>
<tr>
<th>Data</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the project.</td>
</tr>
<tr>
<td>Linked list</td>
<td>List of all of the workpieces in this project.</td>
</tr>
</tbody>
</table>

Table 4.13 Information stored in a project.

The name of the project is currently just used for the convenience of the designer. Eventually, entire projects will be saved and stored at one time, instead of just individual workpieces. At that time, the name of the project will either be added to the beginning of the name of each workpiece, or it will be used as a name for a directory holding all of the workpieces from that project.

The workpieces are stored in a linked list in the project.
4.10 Journals and Rollback

One of the advantages of using ACIS API routines when manipulating ACIS solid models is that they keep track of the state of the model. If at any time an invalid model results from an operation on a solid, the model is rolled back to the state that it was in before the operation. This style of transaction processing is also used with databases and in other situations where it is vital that the database not be left in an invalid state.

ACIS also keeps a journal of all of the operations performed on a model. This journal can be replayed to show how the model was modified and in what order each feature was created.

The Virtual Workshop is capable of creating a journal. Each of the machines, parts, blades, and workpieces have a name. Every time anything is moved or adjusted, the name of the object along with the adjustment, can be stored in a file. This file could then be used to drive the workshop to replay operations that created a particular workpiece. The replay capability is not yet built into the Virtual Workshop.

The journaling capability could be used for communication in several ways. The workshop journal could be replayed to show someone how a part was created. It could be sent in over a network so that two or more people at separate locations could watch the same operation performed on a workpiece at the same time.
4.11 Mark Representation

Marks are stored in the workpiece body as a linked list of wires. As a mark is created, it is added to the wire list in the body. Although the marks in the current implementation are only straight lines and ellipses, the ACIS modeler allows more general curves to be stored as wires.

When a offset, circle, line, or polygon is created, it is stored temporarily in an object called a figure. The figure can be easily modified and is used mainly for temporarily displaying the object that is being created. The figure stores all of the geometric information necessary for drawing any of several types of lines and curves. The geometric information can be easily modified as the designer moves the mouse. As the geometry is modified, the figure constantly redraws itself on the screen. When the designer releases the mouse, the figure is used to create a wire which is then added to the body of the workpiece and the figure is destroyed.

When the designer chooses Line from the Marks menu, a figure is created but no information is placed into the figure and nothing is drawn. As soon as the designer clicks the mouse button on the workpiece, the intersection point on the workpiece is calculated and that point is stored in the figure as the beginning and end points of a line. The figure draws the line (just a point) on the screen. As the mouse pointer is moved around on the workpiece, the line’s end point is moved to the new intersection point. The line is continuously redrawn in the position that it would take if the mouse button were released at the new end point.

When the mouse button is released, the beginning and end points from the figure are used to create a straight wire on the surface of the workpiece. The new wire is added to the list of wires in the workpiece and the figure is destroyed.

In an ACIS wireframe, wires are stored as vertices connected by curves. Many curve types can be represented within the ACIS kernel, but only ellipses and straight lines are used for the marks in the Virtual Workshop. For more details about how wires are represented in an ACIS solid model, refer to the ACIS interface reference manual.
Figures

A figure contains all of the information required for drawing circles, lines, polygons, or splines. It consists of a type indicator, an array of points, a normal vector, and a color. The type can be Invalid, Line, Polygon, Circle.

If the figure is a Line, each pair of points in the point array represents an individual line segment. In other words, if there are $n$ points, $(n/2)$ line segments are drawn. A Polygon is different because $(n-1)$ lines are drawn for $n$ points. All of the line segments in the polygon are connected. Circles only have 2 points, the center and the radius, and no more than 2 points may be added to a Circle figure. Also, a Circle must have a normal to indicate the direction of rotation of the radius around the center point.

The figure object has routines for adding points, modifying the last point added and drawing itself. A figure can also be reset, the line color can be changed, and the type can be queried through the object interface.

The Invalid figure type is used to indicate that a figure has been reset and has no valid information to be displayed.
Before the completion of the Virtual Workshop, a brief design test was given to a group of individuals with different levels of experience in the design field. The results of that survey are discussed in this chapter.

When the Virtual Workshop was complete, several individuals and groups were given a demonstration. The groups included designers, model makers, experienced engineers and students. After the demonstration, they were allowed to experiment with the workshop.

They were each asked to respond to the workshop by filling out a questionnaire and commenting on their perceptions of the environment.
5.1 Design Problem Survey

One of the reasons for implementing the Virtual Workshop was to help designers consider process design during product design. A brief survey completed as part of this research had shown that experienced designers focus on process in the very early stages of design, while inexperienced designers consider only geometric representations or how the product “looks”. Perhaps lack of experience keeps design students from understanding the nature of the entire design process. Their limited experience may prevent them from drawing on practical solutions to design problems.

In order to study how level of experience affects design, several short design projects were created and designers of several experience levels were asked to complete the projects. The problem statement given to the participants is presented in Figure 5.1. The subjects ranged from design students with little or no experience, to accomplished designers with more than 5 years of commercial design experience. This test was not exhaustive or rigorous. It was simply a qualitative effort to understand some of the fundamental differences in how designers with and without experience approach design.
Design Problems

The purpose of this exercise is to generate designs and to communicate your design ideas to the person who will be making or building the designs that you create. Only one of each item will be made from your design (don't design something that requires development of a new technology or requires a hardened steel mold to be constructed.) Your idea should be developed well enough and in enough detail that someone can build what you design from your drawings, pictures, and notes.

1. Design a container to enclose a single deck of cards.

2. Design a vent cover. The purpose of the vent is to allow air to flow through a pipe, while keeping birds and small rodents out. The pipe is embedded in a wall and is 3 inches in diameter. The recess for the cover is 5 x 5 x 1/8 inches.

3. Two halves of a 2 inch diameter pipe (cut length-wise) must be clamped together. Design a clamp that will do the job. They must be positioned accurately (tying the pipe halves with rope won't work.)

Figure 5.1 Problem statement for design experiments.

The solutions produced by the participants were quite dissimilar. In answer to Design Problem 1, some of the inexperienced design students created designs like the one depicted in Figure 5.2. Details are not specified in the drawing. There is no indication of material or process and it is not clear if the part should be made from a single piece or several pieces assembled together. In this case, none of the criteria of the problem were met by the student.
Figure 5.2 Inexperienced designers solution to card holder (Problem 1).

Figure 5.3 shows an example of an answer to the same problem by an experienced designer. Although the experienced designer left out dimensions, many more details are present in his sketch. Joints were drawn indicating how the box would be assembled. The operation of the finished box is indicated. Material was specified also. With the addition of dimensions, the box could be constructed as specified.

Figure 5.3 Experienced designers sketch of card holder.
The other design projects showed similar differences between experienced designers and students of design. Table 5.1 enumerates some of those differences.

<table>
<thead>
<tr>
<th>Experienced Designer</th>
<th>Inexperienced Design Student</th>
</tr>
</thead>
<tbody>
<tr>
<td>Materials specified and integral to design.</td>
<td>Materials not mentioned.</td>
</tr>
<tr>
<td>Usually sketched in perspective.</td>
<td>2D sketches showing layouts. Occasional perspective sketches.</td>
</tr>
<tr>
<td>Joint and assembly details specified.</td>
<td>Joint and assembly details omitted.</td>
</tr>
<tr>
<td>Designs often defined in terms of process</td>
<td>No mention of process in design.</td>
</tr>
<tr>
<td>used to create them.</td>
<td></td>
</tr>
<tr>
<td>Close attention to details.</td>
<td>Little attention to detail.</td>
</tr>
</tbody>
</table>

Table 5.1 Differences in approach between designers with differing levels of experience.

Again, this survey was not comprehensive, but the results did provide some indication of the differences between designers at various experience levels.
5.2 Demonstrations

The objective of the demonstrations and survey described in this chapter was to understand how a system like the Virtual Workshop might be useful and in what areas it would provide advantages to a designer.

The demonstrations to the participants were purposely subjective and unstructured. Although specific features of the Virtual Workshop were demonstrated and discussed, the participants were free to ask questions and use the workshop at their discretion. General comments were requested, along with some comments about specific features of the workshop.

Demonstration Objective

There are many full-featured CAD systems available. A single individual working alone would have a very difficult time creating a CAD system that performs on the level of a commercial program. However, concepts can be demonstrated with the understanding that a full CAD system might be built around them. After the completion of the Virtual Workshop, the workshop was demonstrated and reactions of designers, engineers, and model makers were observed.

The purpose of the demonstration was to determine the value of the Virtual Workshop concept in a variety of settings. A demonstration to experienced engineers was especially important for understanding the value of the concept in an industrial setting.

Each demonstration lasted between 15 and 30 minutes. A short description of the Virtual Workshop environment was given at the beginning of the demonstration. When the program started, features of the workshop were shown to the participants. Each feature was explained and an example was shown. Participants could ask questions about the feature, ask for specific demonstrations, or try the feature themselves. Most participants did not use the Virtual Workshop, but were content to watch the demonstration. Several people used the machines and created parts in the workshop.

The participants responded to the workshop by filling out a brief survey and responding to open-ended questions about the workshop. Responses to the survey are summarized in the following section.
5.3 Response

The subjects were asked specifically to comment on several current and planned features of the workshop. Selected comments are in Table 5.2. Comments without quotation marks have been paraphrased.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Represents real machines in real workshops</td>
<td>“Realism not critical, but literal representation helps point out when jigging, etc. would be necessary.”</td>
</tr>
<tr>
<td></td>
<td>Direct manipulation a good idea. “They help new students understand how the machines work.”</td>
</tr>
<tr>
<td>Cuts generated automatically based on motion of workpiece or blade</td>
<td>“Tool cuts were accurately represented, especially drilling and milling processes.”</td>
</tr>
<tr>
<td>Design constraints match machining constraints</td>
<td>“To me this is the principal benefit of using the Virtual Workshop.”</td>
</tr>
<tr>
<td>Ability to ‘undo’ mistakes</td>
<td>“Essential.”</td>
</tr>
<tr>
<td></td>
<td>Would like to be able to edit/modify/generalize the final product plan.</td>
</tr>
<tr>
<td>Inexpensive to try out different processes</td>
<td>“And safer too.”</td>
</tr>
<tr>
<td></td>
<td>“May need a bit more feedback to do this, e.g. cutting time, shop floor layout.”</td>
</tr>
<tr>
<td>Replaying process plans</td>
<td>“This would be a great help with complex parts.”</td>
</tr>
<tr>
<td></td>
<td>“Probably best for educational purposes.”</td>
</tr>
<tr>
<td></td>
<td>“This is useful for complex cuts or very unusual processes.”</td>
</tr>
<tr>
<td>Creating process plans</td>
<td>“Another great application for this software.”</td>
</tr>
<tr>
<td></td>
<td>“Not too useful.”</td>
</tr>
<tr>
<td></td>
<td>I find it very useful for recording a process plan.</td>
</tr>
</tbody>
</table>

Table 5.2 Selected comments from survey concerning current and planned features of the Virtual Workshop.
<table>
<thead>
<tr>
<th>Feature</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard tools and standard stock available</td>
<td>“Needed as most places have standardized inventory.”</td>
</tr>
<tr>
<td></td>
<td>“Fine.”</td>
</tr>
<tr>
<td></td>
<td>Designers would use the same standards instead of odd-sized cutters.</td>
</tr>
<tr>
<td>Easy to design new machines</td>
<td>“Not critical once standard tools are defined. I think that the ability to quickly construct jigs will be needed.”</td>
</tr>
<tr>
<td></td>
<td>“I like the ‘easily update-able’ feature.”</td>
</tr>
<tr>
<td>Graphical environment</td>
<td>“Essential if designers are to use it.”</td>
</tr>
<tr>
<td></td>
<td>“Clear. Easy to understand the geometry of the machine.”</td>
</tr>
</tbody>
</table>

Table 5.2 (cont’d) Selected comments from survey concerning current and planned features of the Virtual Workshop.

Some general comments about the environment were:

“I would use it to verify the manufacturability of parts.”

“I can’t draw parts so I’d machine what I wanted then extract drawings for the shop.”

“Use this to introduce machining. It’s a lot safer too.”

“If you added safety features so that the user could see the results of improper use of machinery this could be a very useful teaching tool for how to use machine tools.”

“I would use it to quickly verify a design if some features were added to the present system.”

“As the system stands presently, I think it would be best used as a training aid.”

“With process planning added to it, it would be a great design verification tool.”

“I think it would need some refining to be truly useful – perhaps including the materials constraints in the process would make it a better teaching tool.”

“Great for co-op students training or for outside shop people such as sales people in this industry.”

“Makes it easy to show co-workers how you want part machined.”
Most of the designers, students, model makers and engineers who saw the Virtual Workshop agreed that the workshop was a good environment for training and communication. However, conceptual design was not perceived as the best use for the environment. Most of the participants felt that the workshop would be most useful for verification and process planning and as a final sanity check before the part was made.

The participants felt that the interface was intuitive and easy to learn. Direct manipulation of objects on the screen and similarity to a real workshop contributed to the intuitive feel of the interface.
6.1 Demonstration Results

The Virtual Workshop is not a complete design system. It is a research system designed to test a concept. For this reason, it was not compared to commercially available CAD systems which contain a full array of tools and capabilities.

The Virtual Workshop was demonstrated to about 20 people. Some were students at Massachusetts Institute of Technology and others were engineers and industrial designers with 12 or more years of industry experience. Some of the participants were model makers from Polaroid Corporation who spend their time building the designs created by the designers.

Overall, the reaction to the Virtual Workshop was positive, not as a design system, but for verification and training. The system seemed too constrained for conceptual design but was seen as a good tool for process planning in later stages of design. The subjects were most enthusiastic about using the Virtual Workshop as a training aid for shop apprentices, new engineers and others unfamiliar with workshops and manufacturing processes.

Disadvantages

The primary disadvantage of the Virtual Workshop as a design tool is the need to have many design details determined before using the system. In other words, conceptual design is difficult using the Virtual Workshop. A part cannot be made if the details are not resolved. Often, during design, a concept is developed a little bit at a time. The overall size of the piece, material, and exact location of features are not always known in advance. However, once the details have been solidified, it is possible to create the part in the workshop.

Another disadvantage is the lack of tools available in this version of the workshop. Real workshops are full of tools and equipment designed to help a model maker complete a task. Lack of access to essential tools is frustrating to a designer. Future versions of the workshop must have more tools available.

Other disadvantages mentioned in response to the demonstrations were in reference to minor interface issues that were not unique to the workshop metaphor and are not elaborated here.
Advantages

All of the people who watched or used the workshop thought that it was a comfortable, intuitive environment for process design or training. Interaction with the machines was easy to learn and grasp. One subject commented, "Puts fun into subject, easy to understand."

The participants felt that the graphical nature of the environment would make it an ideal platform for training and communication. People can learn about the tools and machines in a workshop without being in the shop. The machines act substantially the same way in the Virtual Workshop as in a real workshop.

The Virtual Workshop was a very complex environment drawing on research from a wide range of disciplines. However, this research showed that it was possible to design and create a system that simulated a workshop and was easy to understand and use.
6.2 Future Research

In its present state, the Virtual Workshop is a nice environment for testing interface approaches and machine representations. However, there are several areas of research that merit investigation.

Journals and Playback

If activity in the Virtual Workshop could be stored and played back, the workshop could be used for communicating process and setup plans. Each of the elements in the workshop have names and a journal file can use those names to record all of the interactions in the workshop. However, the journal file cannot be replayed. Adding a replay capability to the workshop would allow designers to communicate their designs and plans by showing what they have done, instead of just displaying the final geometry.

Workshop Shortcuts

Sometimes, designing using the machines in the workshop is laborious. Milling a pocket in a block can be time consuming. Perhaps, if a designer could sketch the shape of the pocket on the surface of the part and have the mill automatically cut the pocket using the current blade, the workshop metaphor would still be active, but the designer would not have to be concerned with every detail.

Sheet Metal Brakes and Wire Jigs

The only machines currently represented in the Virtual Workshop are cutting machines. Other types of processes available in a real workshop cause deformation of the stock. A sheet metal brake deforms sheet metal instead of removing material. Wire jigs are used to bend wire. Representing deformation processes in the workshop will require a fundamentally different approach than the Boolean operations used for cutting.

Tolerances

As mentioned in Chapter 2, tolerance representation is a very difficult task. As a part is created in the Virtual Workshop, cutting operations create new surfaces. Information could be stored with each surface about the process that created the surface. The information
could be used to calculate finish, accuracy, and tolerances. The information could then be used during assembly for determining fit and analyzing process decision consequences.

**Machine Configuration**

Computing interferences between machine parts overwhelms the workshop simulation and destroys the interactivity of the workshop. On the other hand, when interferences are not computed, the machine parts can intersect, reducing the integrity of the simulation. Obstacle avoidance research in robotic path planning could be applied to the workshop to guarantee that the machines never move to a configuration that causes self-intersection.

**Path Planning for Numerically Controlled Machines**

With the addition of appropriate tools to the workshop, a designer could program 3-axis and 5-axis numerically controlled (NC) machines. This type of NC programming would be an extension of Gossard's work in analogic part programming [Gossard 75].

**Sound**

Some users of the workshop have suggested that sound might be an interesting addition to the virtual workshop. Although the addition of sound might seem frivolous, many operations in the workshop would benefit from sound. When drilling a hole or milling a block, sound could indicate the initial intersection of the cutter with the stock. This frees up the designer's visual sense to concentrate on other parts of the operation, like watching the motion indicator to determine how far to move the blade.
Example Input Files

Appendix A

This appendix contains several example files that show the format of the Virtual Workshop text input.
A.1 Machine Input File

For more information about the contents and meaning of the machine files, refer to section 4.3, "Machine Representation and Control".

The following lines are from the Table Saw machine description file:

; Example Machine File - filename: Table_Saw.machine

; Lines beginning with semi-colons ';' are comments.
; Blank lines are also comments.

; The information before the first colon is the 'Field Name'
; and after the colon, before the first semi-colon is the 'Field Info'.

Brand Name: Table Saw brand name.
Style: Table Saw
Blade Mount: Arbor, 0.625 in, 1 in, unkeyed
Blade Location: Base.Blade Height Adjustment.Blade Angle
Adjustment.Arbor:(0, 0, -5):(0, -1, 0):(1, 0, 0)
Default Blade: Forrest Woodworker II Combination Blade

Name: Base
Visible: Yes
Hideable: No
Material: Steel

Name: Blade Height Adjuster
Visible: Yes
Hideable: No
Constraint: Rotational:(0, -14, -15):(0, -1, 0):[-2pi rad, 10pi rad]:[-2pi rad, -pi rad, ...]:5 deg:0
Drives: Base.Blade Height Adjustment
Material: Steel
Parent: Base

Lock: Height Adjuster Lock
Controls: Base.Blade Height Adjuster
On: Base.Blade Height Adjuster
Locked: No
Locked Material: Black Plastic
Unlocked Material: Black Plastic

Name: Blade Angle Adjuster
Visible: Yes
Hideable: No
Constraint: Rotational:(-20, 0, -15):(-1, 0, 0):[-2pi rad, 10pi rad]:[-2pi rad, -pi rad, ...]:5 deg:0

134 Appendix A Example Input File
Drives: Base.Blade Height Adjustment.Blade Angle Adjustment
Material: Steel
Parent: Base

Lock: Angle Adjuster Lock
Controls: Base.Blade Angle Adjuster
On: Base.Blade Angle Adjuster
Locked: No
Locked Material: Black Plastic
Unlocked Material: Black Plastic

Name: Blade Height Adjustment
Visible: Yes
Hideable: No
Constraint: Linear:(0, 0, 1):[0, 6]:0
Material: Steel
Parent: Base

Name: Blade Angle Adjustment
Visible: Yes
Hideable: No
Constraint: Rotational:(0,0,0):(0,1,0):[0 deg, 45 deg]:0
Material: Steel
Parent: Base.Blade Height Adjustment

Name: Arbor
Visible: Yes
Hideable: No
Material: Steel
Parent: Base.Blade Height Adjustment.Blade Angle Adjustment

Name: Table
Visible: Yes
Hideable: No
Material: Steel
WorkPlane: (0, 0, 1, 0)

Name: Blade Guard
Visible: Yes
Hideable: No
Material: Steel
Parent: Table
Constraint: Linear:(0,0,1):[0, 20]:0

Name: Right Crosscut Fence
Visible: No
Constraint: Linear:(0,1,0):[-20, 20]:0
Material: Steel
Parent: Table
Conflicts: Table.Left Rip Fence, Table.Left Crosscut Fence, Table.Saw Box, Table.Panel Cutter
WorkPlane: (0, 1, 0, 16.5)
Name: Angle
Hideable: No
Constraint: Rotational:(10, -20, 0):(0, 0, 1):[-45 deg, 45 deg]:[-45 deg, -40 deg, ...]:1 deg:0
Material: Steel
Parent: Table.Right Crosscut Fence
; WorkPlane: ( 0, 1, 0, -20)

Lock: RC Angle Lock
Controls: Table.Right Crosscut Fence.Angle
On: Table.Right Crosscut Fence.Angle
Locked: Yes
Locked Material: Black Plastic
Unlocked Material: Black Plastic

Name: Left Crosscut Fence
Visible: No
Constraint: Linear:(0,1,0):[-20,20]::0
Material: Steel
Parent: Table
Conflicts: Table.Right Rip Fence, Table.Saw Box, Table.Panel Cutter, Table.Right Crosscut Fence
WorkPlane: ( 0, 1, 0, 16.5)

Name: Angle
Hideable: No
Constraint: Rotational:(-10, -20, 0):(0, 0, 1):[-45 deg, 45 deg]:[-45 deg, -40 deg, ...]:1 deg:0
Material: Steel
Parent: Table.Left Crosscut Fence
; WorkPlane: ( 0, 1, 0, -20)

Lock: LC Angle Lock
Controls: Table.Left Crosscut Fence
On: Table.Left Crosscut Fence
Locked: Yes
Locked Material: Black Plastic
Unlocked Material: Black Plastic

Name: Panel Cutter
Constraint: Linear:(0,1,0):[-40,40]::0
Material: Walnut
Parent: Table
Conflicts: Table.Right Crosscut Fence, Table.Left Crosscut Fence, Table.Right Rip Fence, Table.Left Rip Fence, Table.Saw Box
WorkPlane: ( 0, 0, 1, 1)
Supercedes: Table

Name: Backstop
Visible: Yes
Hideable: No
Constraint: Fixed
Material: Walnut
Parent: Table.Panel Cutter
WorkPlane: (0, 1, 0, -40)

Name: Saw Box
Constraint: Linear:(0,1,0):[-40,40]:0
Material: Pine
Parent: Table
Conflicts: Table.Panel Cutter, Table.Right Crosscut Fence, Table.Left Crosscut Fence, Table.Right Rip Fence, Table.Left Rip Fence
WorkPlane: (0, 0, 1, 1)
Supercedes: Table

Name: Backstop
Visible: Yes
Hideable: No
Constraint: Fixed
Material: Pine
Parent: Table.Saw Box
WorkPlane: (0, 1, 0, -40)

Name: Right Rip Fence
Visible: Yes
Constraint: Linear:(1,0,0):[0.25]:[0,1,...]:0.150:0
Material: Steel
Parent: Table
Conflicts: Table.Left Rip Fence, Table.Left Crosscut Fence, Table.Saw Box, Table.Panel Cutter
WorkPlane: (-1, 0, 0, 0)

Name: Taper Jig
Constraint: Linear:(0,1,0):[-20,20]:0
Material: Aluminum
Parent: Table.Right Rip Fence

Name: Angle
Visible: Yes
Hideable: No
Constraint: Rotational:(-0.5,10,0):(0,0,-1):[0 deg,20 deg]:[0 deg, 5 deg]:1 deg:0
Material: Aluminum
Parent: Table.Right Rip Fence.Taper Jig
WorkPlane: (-1, 0, 0, 2)
Supercedes: Table.Right Rip Fence

Lock: RTJ Angle Lock
Controls: Table.Right Rip Fence.Taper Jig.Angle
On: Table.Right Rip Fence.Taper Jig
Locked: Yes
Locked Material: Black Plastic
Unlocked Material: Black Plastic
Name: Backstop
Visible: Yes
Hideable: Yes
Material: Aluminum
Parent: Table.Right Rip Fence.Taper Jig.Angle
WorkPlane: (0, 1, 0, -20)

Name: Left Rip Fence
Visible: No
Constraint: Linear:(-1, 0, 0):[0, 25]:[0, 1,...]:0.1:0
Material: Steel
Parent: Table
Conflicts: Table.Right Rip Fence, Table.Right Crosscut Fence, Table.Saw Box, Table.Panel Cutter
WorkPlane: (1, 0, 0, -2)

Lock: LR Fence Lock
Controls: Table.Left Rip Fence
On: Table.Left Rip Fence
Locked: No
Locked Material: Black Plastic
Unlocked Material: Black Plastic

Name: Taper Jig
Constraint: Linear:(0, 1, 0):[-20, 20]::0
Material: Aluminum
Parent: Table.Left Rip Fence

Name: Angle
Visible: Yes
Hideable: No
Constraint: Rotational:(0.5, 10, 0):(0, 0, 1):[0 deg, 20 deg]:[0 deg, 5 deg]:1 deg:0
Material: Aluminum
Parent: Table.Left Rip Fence.Taper Jig
WorkPlane: (1, 0, 0, 2)
Supercedes: Table.Left Rip Fence

Lock: LTJ Angle Lock
Controls: Table.Left Rip Fence.Taper Jig.Angle
On: Table.Left Rip Fence.Taper Jig
Locked: Yes
Locked Material: Black Plastic
Unlocked Material: Black Plastic

Name: Backstop
Visible: Yes
Hideable: Yes
Material: Aluminum
Parent: Table.Left Rip Fence.Taper Jig.Angle
WorkPlane: (0, 1, 0, -20)

Name: Push Stick

138 Appendix A Example Input File
Visible: No
Hideable: Yes
Constraint: Linear:(0,1,0):[-20,20]:0
Material: Pine
Conflicts: Table.Left Crosscut Fence
Parent: Table.Left Rip Fence
WorkPlane: (0, 1, 0, -10)

Name: Feather Stick
Visible: No
Hideable: Yes
Constraint: Linear:(1, 0, 0):[0,25]:0
Material: Pine
Parent: Table.Left Rip Fence
WorkPlane: (-1,0,0,0)

Name: Push Stick
Visible: No
Hideable: Yes
Constraint: Linear:(0,1,0):[-20,20]:0
Material: Pine
Conflicts: Table.Right Crosscut Fence
Parent: Table.Right Rip Fence
WorkPlane: (0, 1, 0, -10)

Name: Feather Stick
Visible: No
Hideable: Yes
Constraint: Linear:(-1, 0, 0):[0,25]:0
Material: Pine
Parent: Table.Right Rip Fence
WorkPlane: (1,0,0,0)

Name: Outfeed Roller (1)
Visible: No
Hideable: Yes
Constraint: Planar:(1,0,0):(0,1,0):[-60,60]:[40,80]:0:0
Material: Steel

Name: Outfeed Roller (2)
Visible: No
Hideable: Yes
Constraint: Planar:(1,0,0):(0,1,0):[-60,60]:[40,80]:0:0
Material: Steel
A.2 Blade Input Files

For more information about the contents and meaning of the blade files, refer to section 4.6, “Blade Representation”.

The following information is the file describing a 3/4 inch diameter flat end mill.

Name: 3-4 in flat end mill
Brand: Generic
Style: 3/4 in diameter, 3.5 in long.
Material: Brass
Mount: Chuck, 0.75
Axis: (0, 0, -1); (1, 0, 0)
Position: (0, 0, 0)
Wire: (0, 0, -1); (0.375, 0, -2.5); 180 deg; (-0.375, 0, -2.5); 180 deg;
(0.375, 0, -2.5)
Constraint: Linear; (0, 0, -1)
Wire: (1, 0, 0); (0, 0.375, 1); 0; (0, 0.375, -2.5); 0; (0, -0.375,
-2.5); 0; (0, -0.375, 1); 0; (0, 0.375, 1)
Constraint: Planar; (0, 0, 1)

The following information is the file describing a combination blade for the table saw.

Name: Forrest Woodworker II Combination Blade
Brand: Forrest
Style: Woodworker Combination Blade
Material: Silver
Mount: Arbor, 0.625 in, .125 in, unkeyed
Axis: (0, -1, 0); (1, 0, 0)
Position: (0, 0, -5)
Wire: (0, -1, 0); (.0625, 0, 0); 0; (.0625, 0, -10); 0; (-.0625, 0, -10); 0; (-
.0625, 0, 0); 0; (.0625, 0, 0)
Constraint: Linear; (0, -1, 0)
Hardware and Software Platform
Appendix B

Hardware Platform
Silicon Graphics Indigo2 Extreme
  32 Meg RAM
  R4000 processor
  120,000 shaded polygons per second

Software
  C++, version 2.1 based on CFront from AT&T
  ACIS solid modeler from Spatial Technologies
  OSF/Motif and X Windows
  Approximately 30,000 lines of C++ code


<table>
<thead>
<tr>
<th>Reference</th>
<th>Citation</th>
</tr>
</thead>
</table>


