

## VIRTUAL TRAINING ENVIRONMENT FOR A 3-AXIS CNC MILLING MACHINE

Tamer M. Wasfy

Advanced Science and Automation Corp., Indianapolis, IN

Ayman M. Wasfy

Hazim El-Mounayri

Mechanical Engineering Department  
Purdue School of Eng. & Tech., IUPUI, Indianapolis, IN

Daniel Aw

### ABSTRACT

A virtual training environment for a 3-axis CNC milling machine is presented. The key elements of the environment are: (a) textured 3D photo-realistic virtual models of the machines and lab; (b) machine simulator for the machines' controls and moving parts; (c) semi-empirical model of the machining operation; (d) hierarchical knowledge-base for process training; (e) unstructured knowledge-base for lecture delivery; (f) natural-language human-like intelligent virtual tutors. Applications of the AVML include: training students to operate manufacturing machines in a safe environment, allowing students and researchers to view and interact with highly accurate physical simulation of manufacturing machines, and optimization of the manufacturing process plan by testing various plans on the virtual machine before actual machining. The virtual training environment will significantly reduce the cost and increase the accessibility and safety of advanced manufacturing training.

### 1. INTRODUCTION

Virtual environments (VEs) provide safe and cost-effective environments for learning and "hands-on" training. Recent strides in computers and graphics cards speeds are making VEs increasingly more realistic (i.e. closer to physical environments) especially from the visual and auditory perspectives. Accordingly, VEs are becoming increasingly attractive in education and training applications. In this paper we will describe a software system called "AVML" (Advanced Virtual Manufacturing Lab). The AVML is a virtual training environment (VTE) which includes a 3-axis CNC milling machine. The AVML software will eventually include other types of manufacturing machines such as lathes and 5-axis milling machines. The AVML can be used to provide hands-on training on the operation of CNC machines. The students can also experiment with the virtual machines and test operating procedures and CNC programs. Hands-on training and experimentation enhance students learning and retention and

increase students creativity and problem-solving capabilities. In addition, the AVML provides an effective knowledge capture and reuse capability. The traditional knowledge capture method of relying on operating manuals fails to capture the dynamic and time-dependent aspects of operating CNC machines. The three main advantages of VTEs over traditional training on physical CNC machines are:

- *Cost*: The VTE runs on Windows and Linux based PCs. Thus, small colleges can provide CNC machining training without investing a lot of money to setup a CNC machining lab. Even for larger colleges, there could be budget limitations that prevent the initial investment (especially, for sophisticated machines) or the subsequent upgrading to keep up with the changing technology. In addition, the AVML saves money on consumables such as tools and workpieces.
- *Safety*: The students can experiment without risk of damaging expensive equipment and facilities or injury.
- *Access*: Unlike manufacturing labs with limited lab hours and required supervision, the AVML is easily accessible at any time without supervision and can be accessed via the Internet.

Motivated by the advantages of virtual-reality based training, many research groups and commercial companies developed computer-based CNC machine training. Suh *et al.* 2003 [1] developed a web-based virtual CNC machine that includes geometric and kinematic models of the machine and that uses the VRML COSMO player along with JAVA-applets. Lin *et al.* 2002 [2] developed a virtual-reality CNC training prototype system driven by the WorldToolKit from Sense8 and consisting of five modules: training task-planning, machine simulator, performance evaluation, instruction, and interface. Petri nets were used to specify training task plans. Ong and Mannan 2004 [3] and Ong *et al.* 2002 [4] developed a web-based interactive manufacturing teaching module that uses VRML and JAVA applets. A G-code interpreter is implemented using a JAVA-applet that controls VRML scene

to produce the tool motion corresponding to the G-code program. The VRML ElevationGrid object was used to model the workpiece. The heights of the ElevationGrid are changed based on the intersection of the surface with the tool. A simple average cutting force model based on the power required to remove a unit volume of material was used. Luo *et al.* 2002 [5] developed a web-based virtual milling machine that combines a geometric model for the machine with stitched images for the rest of the workshop and that uses the aforementioned ElevationGrid VRML technique. Kong *et al.* 2002 [6] also developed a web-based virtual milling machine that uses the aforementioned ElevationGrid VRML technique. Chang *et al.* 2001 [7] developed a five-axis virtual milling machine that includes haptic and aural rendering. Haptic rendering was used so that the user can feel the cutting forces and surface roughness. The analytic cutting force model developed by Abrari and Elbestawi 1997 [8] was used. Aural rendering was done using Microsoft DirectSound functions. The cutting force was used to determine the amplitude of the cutting sound and the spindle speed was used to determine the frequency. Ustarroz *et al.* 2004 [9] developed VIRTOOL – a virtual learning environment for milling machines. The system was still under development but was to eventually include intelligent agents to guide the student through the process of operating the CNC machine, however, the project seems to have ended in late 2003. Denford Ltd., United Kingdom [10], developed a virtual training environment for micro milling machines. The 3D real-time work-piece solid model is not accounted for. Delmia, France [11] offers CNC machining verification modules that include fast solid modeling for predicting the updated part geometry during cutting.

The AVML uses some of the techniques and concepts presented in the above literature. In addition, it surpasses the previously reported virtual CNC milling machines in the following five aspects: visual and audio fidelity of the virtual machine, naturalness of interactivity, accuracy of modeling the cutting process, use of virtual tutors, and web access. A visually realistic virtual machine should look and sound almost the same as the physical machine. All existing virtual CNC machines fall far short from that goal. In the AVML, the virtual CNC machine is near-photorealistic and produces near-real sounds including machining sounds. In addition, the user can make parts of the machine such as the sheet-metal enclosure semi-transparent or hide parts of the machine in order to more clearly see the motions of the other machine parts. Also, previous virtual CNC machines did not incorporate the machine sounds including the cutting sounds. Chang *et al.* 2001 [12] include a rough model of the cutting sounds. Most virtual CNC machines use 2D windows and widgets to model the machine controller. Some have a 3D immersed controller that is not photorealistic and does not have the same shape of knobs, buttons, and switches as the actual machine. In the AVML, the machine controller and machine software screens are almost identical to the actual CNC machine. Also, the control knobs, buttons, and switches are near-photorealistic and can be manipulated by the user in a variety of natural ways: by pointing, clicking, and dragging using the mouse, by natural-language voice commands, or by selecting the command from a hierarchical command-tree. The AVML provides better than “computer-game” quality navigation in the VE that uses a physics-based dynamic motion model along with a penalty-based frictional

contact model. Most previous CNC machines use COSMO player which provides VRML type navigation that is purely kinematic and that does not include effects such forces, acceleration, and friction.

Another aspect that distinguishes the AVML from previous virtual CNC machines is modeling of the cutting process. Previous virtual machines only include the geometric aspect of the cutting process. The accuracy of the geometric modeling varies from the very accurate system of Delmia Inc. to the less accurate VRML ElevationGrid-based geometric modeling technique. The AVML geometric modeling uses ACIS solid modeling engine [12] which is similar to the Delmia system in terms of accuracy and speed. Among the systems that include the physical aspects of the cutting process, Chang *et al.* 2001 [7] and Ong *et al.* 2002 [4] include a rough average cutting force model. Other physical aspects such as instantaneous cutting forces, tool deflection, surface errors, surface roughness, and chatter are generally not modeled. The AVML includes an accurate artificial neural network (ANN) instantaneous cutting force model. The force model is used to predict tool deflection, surface errors, and surface roughness.

The AVML includes photo-realistic animated virtual human instructors (Figure 1) that can show, guide, or supervise the student to perform the various operating procedures. The agents use very naturally sounding text-to-speech using Microsoft SAPI to deliver the instruction to the user as well as voice recognition along with a rule-based expert system to recognize natural-language voice commands. Snow and Williges 1998 [13] have shown that adding a virtual human avatar in the VE can increase the sense of presence. The virtual human helps the user to perceive the scale of the various objects. Also, demonstration of an operating step in front of the student by the virtual instructor increases the retention of the student. No other virtual machine includes virtual-human instructors. Also, no other virtual machine includes speech-driven instruction and training.



Figure 1 AVML web-based interface.

The AVML runs inside a web-browser and provides password-based user login and authentication. The AVML uses the IVRESS (Integrated Virtual Reality Environment for Synthesis and Simulation) player [14] and the LEA (Learning Environment Agent) engine [15], which are ActiveX controls

that can be embedded in a web-page. The interface consists of six windows (Figure 1): a VE window, a multimedia instruction window, a speech window, a hierarchical lecture outline window, a hierarchical list of voice commands window and an agent-options window. All the windows can be moved and resized to suite the user's preferences. About half of the reviewed virtual CNC machines run on the web. However, unlike the AVML, the ones that run on the web generally have lower quality in terms of visual realism, naturalness of interactivity, and accuracy of the geometric simulation. The AVML provides the highest available level of visual quality, interactivity, and both geometric and physical cutting process modeling accuracy while being accessible through a web-browser. In summary, the AVML offers the following functions:

- Access to a fully-functional virtual CNC milling machine. Students, researchers, and manufacturing engineers can practice and test CNC programs.
- Training on key operating procedures of the CNC machine.
- Multimedia lectures on the concepts of CNC milling and on the design and functions of the various components of CNC milling machines.

## 2. AVML FRAMEWORK

Figure 2 shows the AVML object-oriented framework. The framework integrates the visualization, process modeling, and control of the virtual CNC machines. It consists of three software modules, which communicate with each other using a TCP/IP network socket interface: (1) a CNC milling machine simulator, (2) a virtual-environment display engine, and (3) an intelligent-agent engine. The three modules run on a single computer in a web-browser (Figure 1).

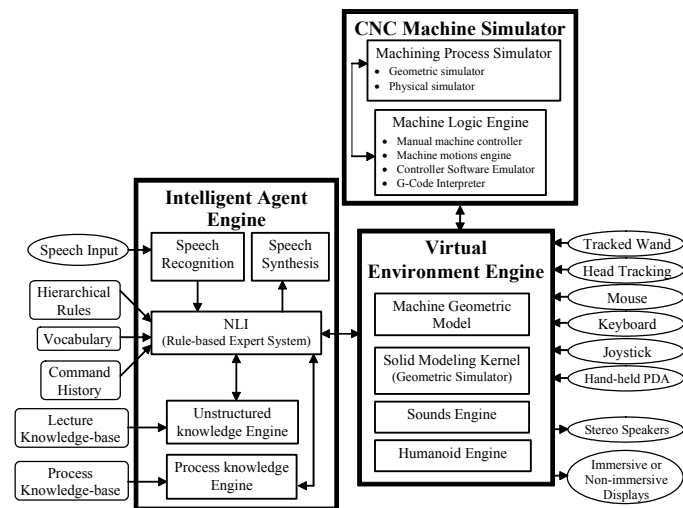


Figure 2 Architecture of the VTE for CNC milling.

The CNC milling machine simulator simulates the various physical and logical characteristics of the actual machine and the machining process. The simulator is written as a standalone module using Visual Basic 6.0 that interfaces with the virtual environment engine using a TCP/IP network socket interface. It includes two sub-components: 1) A machining process simulator and 2) a machine logic engine. The machining process simulator performs the following functions: discretize the tool-motion and predict the cutting forces and other

physical machining effects. The machine logic engine maintains the current state of the CNC machine and properly propagates state changes. It includes an emulator of the machine controller software and a NC-code interpreter.

The virtual-environment is displayed using the IVRESS object-oriented scene-graph virtual-reality engine which includes software objects for: (a) displaying a textured photo-realistic animated geometric model of the machine; (b) boolean-based solid-modeling for geometric simulation of the cutting process through ACIS™ solid modeling kernel; (c) a sounds engine; (d) a humanoid display engine (through the Haptek™ API [16]) for displaying photo-realistic animated humanoid models of virtual instructor(s).

The intelligent-agent engine (LEA) has facilities for unstructured knowledge storage, retrieval and search; a hierarchical process knowledge-base; and a hierarchical rule-based expert system for interpreting and executing the user's natural-language commands. It also has facilities for speech recognition and synthesis. Typical user's commands include: asking the agent to press a machine button, asking the agent to guide the user step-by-step through operating the CNC machine, or asking the agent to describe the function of a machine component. More details on each block in Figure 2 are presented subsequently.

Both the IVRESS and LEA engines are ActiveX controls that can be embedded in a web page and that can access data from the internet using HTTP URL or FTP protocols. Thus, the machine sub-components models can reside anywhere on the web. The AVML framework is modular and object-oriented in order to allow for component reuse and ease of integration of new capabilities and sub-components into the environment. The AVML framework allows various groups to almost independently develop the various components of the virtual machine.

## 3. CNC MACHINE SIMULATOR

The CNC machine simulator consists of two sub-components: 1) A machining process simulator and 2) a machine logic engine.

### 3.1 Machining Process Model

A cutting process model was implemented that allows simulation of the basic operation procedures and training of a 3-axis CNC milling machine. As mentioned above, the machining process simulator is part of the CNC machine simulator (Figure 2) which is written as a standalone module using Visual Basic 6.0 that interfaces with the VE engine using a TCP/IP network socket interface. The machining process simulator uses the properties and methods of the various objects in the VE to simulate the cutting process. The machining process simulator (Figure 2) consists of two sub-modules: the geometric simulator and the physical simulator.

#### 3.1.1 Geometric Simulator

The geometric simulator runs in real-time. First, the tool motion is discretized into small line-segments (whenever linear or circular interpolation G-code commands are issued). Then, an IVRESS "cut" method which wraps ACIS solid modeling sweep and subtraction routines is executed. In this method, the

driving edges of the cutting tool are swept through the line-segment. The resulting volume and the tool volume are subtracted from the workpiece in order to generate the new workpiece geometry and a solid model of the removed material. The solid model of the removed material is used to calculate the radial depth of cut using the following formula:

$$r = \dot{V} / (f a)$$

where  $\dot{V}$  is the rate of volume removed which can be calculated by dividing the volume of the removed material by the time required to move the small linear-segment,  $f$  is the feed-rate, and  $a$  is the axial depth of cut which can be calculated using the maximum length of the removed material along the direction parallel to the tool-axis. Figure 3 shows an example of workpiece updating using this algorithm.



Figure 3 Work-piece updating during the geometric simulation.

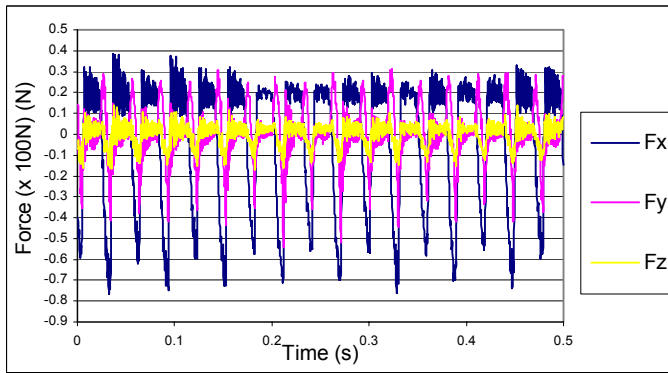


Figure 4 Cutting force measurement over 0.5 sec for a 1/4" diameter tool, 25% axial depth of cut, 25% radial depth of cut, 100 mm/min. feed rate, and 1000 RPM spindle speed.

### 3.1.2 Physical Simulator

The physical simulator simulates the various physical effects present due to the interaction of the cutting tool with the workpiece during cutting. The principal physical effect is the instantaneous cutting force between the tool and the workpiece. This force directly affects other physical process parameters such as tool deflection, dimensional errors, and surface finish. The basic physical effects modeled in the AVML are:

- *Instantaneous cutting forces.* The artificial neural network (ANN) technique developed by Tandon and El-Mounayri 2001 [17] and extended by Briceno *et al.* 2002 [18] was used. The input layer of the neural-network consists of the cutting conditions, including: in-cut geometric information (radial and axial depths of cut), feed-rate, spindle speed, and cutting tool diameter. The in-cut geometric information is calculated using the geometric process simulator. Milling experiments for a HSS flat end-mill which measure the cutting force as a function of the cutting conditions were performed. The cutting force was measured using a three-component force dynamometer. A total of 384 cutting force measurements were conducted

for all the combinations of: 4 spindle-speeds (600 RPM, 750 RPM, 900 RPM, and 1000 RPM), 3 feed-rates (100 mm/min., 120 mm/min, and 150 mm/min.), two cutter diameters (1/2" and 1/4"), 4 axial-depths of cut (25%, 50%, 75%, and 100% of the tool diameter), and 4 radial depths of cut (25%, 50%, 75%, and 100% of the tool diameter). The output of a typical measurement is shown in Figure 4 and it consists of the three force components over 0.5 sec sampled at 2,500 Hz. The main output of the neural network is the maximum instantaneous cutting force. An IVRESS neural network object was implemented and trained using the data obtained from the above experiments. The force model is accurate and suitable for basic training on a number of pre-selected cutting tools and workpiece materials.

- *Tool deflection.* The maximum instantaneous cutting force  $F$  obtained using the ANN technique is used to calculate the tool deflection. The tool deflection  $\delta$  at the tip of the tool is given by  $\delta = F/k$ , where  $k$  is the effective stiffness of the tool, tool-holder and spindle.  $k$  can either be determined experimentally or by using the Canteliver beam deflection formula for the tool, tool-holder and spindle.
- *Surface errors.* The tool deflection was used to calculate surface inaccuracies. The machining error is given by  $e = \delta \sin(\phi)$ , where  $\phi$  is the angle between the surface normal to machined surface and the axis of the tool [25]. A warning is issued if the machining error is larger than a certain tolerance.
- *Tool breakage.* The tool is a Cantilever beam. The maximum stress at the root of the tool is compared to the yield stress to predict the onset of tool breakage.
- *Surface roughness.* The surface roughness  $R$  is calculated using the following equation [26]:
$$R = 10a r^b f^c / w^d$$
 where  $a$ ,  $b$ ,  $c$ ,  $d$  are constants to be determined experimentally,  $r$  is the radial-depth-of-cut,  $f$  is the feed rate and  $w$  is the spindle speed. The equation's constants were determined using the particle-swarm technique (PSO) presented in El-Mounayri *et al.* 2002 [26]. If the surface roughness exceeds a certain threshold, then a warning message is displayed.
- *Cutting sound.* A nominal cutting sound was obtained by subtracting the spindle sound at 1000 RPM from the spindle plus the cutting sounds at the same RPM. The pitch of the sound is varied from 0 to 7.5 based on the spindle's RPM to simulate the cutting tool angular velocity from 0 to 7500 RPM. The amplitude of the sound is set proportional to the cutting force. The sound is then played continuously during cutting.
- *Chip separation and flying model.* The spindle-speed and cutting force are used to calculate the size, flying speed, and frequency of chips. A "ParticleAnimator" IVRESS object was designed to model the flying chips. The object allows emission of particles from a point source and animation of the emitted particles. The object has the following properties: particle source position; frequency of particle emission; speed range of the emitted particles;

scale range of the emitted particles; angular velocity range of the emitted particles; polar angles range of the emitted particles; life time range of the emitted particles; gravity direction; and gravity magnitude. The particle source position is set to the position of the tool's tip. The particle emission frequency  $F_p$  in Hz is given by:

$$F_p = T w / 2\pi$$

where  $T$  is the tool's number of cutting flutes and  $w$  is the spindle angular velocity in rad/sec. The particle speed  $v_p$  is given by:

$$v_p = w D \sin(\phi) / (2 \cos(\phi - \alpha))$$

where  $D$  is the tool diameter,  $\phi$  is shear angle and  $\alpha$  is the tool's rake angle. Cubical boxes are used to represent the chips. The chip's length ( $X$  size) is determined using:

$$X_p = f / F_p$$

where  $f$  is the feed rate. The chip thickness ( $Y$  size) varies between 0 and the radial depth-of-cut. Therefore, chip thickness will be assumed to be half the radial depth-of-cut. The axial depth of cut is used to determine the chip's height ( $Z$  size). The angular velocity of the particle is set to 10% of the spindle's angular velocity. The above properties are dynamically set during the virtual cutting operation in the linear and circular interpolation G-code script subroutines. The "ParticleAnimator" dynamically generates the animation of the flying chips. Figure 5 shows a snapshot taken during machining with the flying chips. This model works in real-time and produces somewhat realistic flying chips.

Table 1 List of typical machine state variables.

State variable	Description
machinePowerSwitch	Whether the machine power lever is set to <i>On</i> or <i>off</i> .
machinePowerState	Whether the machine is turned <i>On</i> or <i>off</i> .
spindleState	Whether the spindle is on or off.
spindleSpeedProgrammed	The programmed spindle speed in RPM
spindelSpeedActual	The actual spindle speed in RPM.
spindelSpeedMultiplier	A multiplier between 0 to 200% for the spindle speed.
xAxisPosition	Absolute current position of the x stage
yAxisPosition	Absolute current position of the y stage
zAxisPosition	Absolute current position of the z stage
aAxisPosition	Absolute current position of the a rotation stage
bAxisPosition	Absolute current position of the b rotation stage
cAxisPosition	Absolute current position of the c rotation stage
Coolant	Whether the coolant is turned on or off
coolantFlow	The coolant flow rate percentage between 0 to 100%
feedRateProgrammed	The programmed feed rate
feedRateActual	The actual feed rate.
feedRateMultiplier	A multiplier for the feed rate between 0 to 150%
toolNumber	Current tool number.
toolOffsetNumber	Tool offset number in the tool offset table.
toolOffset	Current tool offset
motionType	Current motion type: G0: linear rapid, G1: linear interpolation, G2: circular clockwise, G3: circular counter-clockwise.
Unit	G70: inches; G71: mm
coordinateSystem	G90: absolute; G91: incremental
machineDoor	Whether the machine main doors are closed or open.

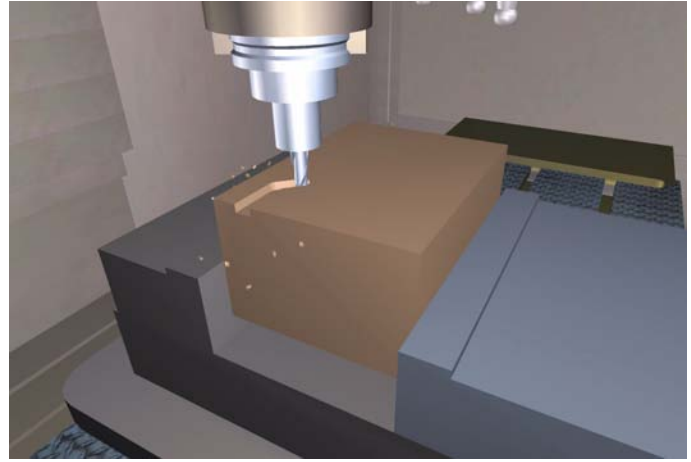


Figure 5 Snapshot taken during machining a work-piece showing the flying chips.

### 3.2 Machine Logic Engine

The machine logic engine simulates the CNC machine, including: 1) manual machine controller interface; 2) motions of the various parts of the machine; 3) machine controller software; and 4) G-code interpreter. The machine logic engine maintains the current state of the CNC machine. Typical state variables are listed in Table 1. The operation logic of the machine controller and software are simulated using event-handling subroutines in the machine logic engine and the VE.

#### 3.2.1 Manual Machine Controller Interface

Figures 6 and 7 show the machine controller and the machine power box, respectively, as displayed in the VE. These machine components form the manual machine interface. The manual machine controls are modeled in the VE using two types of objects: "ButtonWidget" and "DialWidget." The ButtonWidget is used to model the following machine controls:

- *Buttons*: When pressed, they trigger the execution of a corresponding "click" event handling routine. However, they do not retain a state of pressed or not pressed. Buttons include the keyboard buttons. Figure 8a shows the IVRESS definition of the controller green start button (see Figure 6). The "ButtonWidget" has properties specifying that the button is pressed by changing the "yTranslation" of the "vmc3016\_startButton" object from a Y value of 0 to -0.01. The property "animationTime" indicates that the button press animation lasts 0.15 sec.
- *Two state switches*: Same as Buttons, except that they retain a state of *on* or *off*. Switches include the video on/off and Light on/off switches of the machine controller in Figure 6. They also include the machine on/off power switch at the back of the machine (Figure 7).
- *Discrete state knobs*: Same as Buttons and switches, except that they have more than two set states. Figure 8b shows the IVRESS definition of the axis selector knob. The knob has seven discrete positions namely X, Y, Z, A, B, C or remote axis (Figure 6). This object controls the "rotationAngle" of the "Transform" object that includes the knob geometry. When the user clicks on the knob, the ButtonWidget changes (animates) the Transform rotationAngle to the next position in 0.1 sec.

The DialWidget is used to model the following machine controls:

- *Dials*: Used to indicate the value of a continuous variable along the variable range. The dial can be linear or rotary. For example, the “spindle-load %” is a linear dial (Figure 6).
- *Continuous knobs*: Similar to dials, except they also allow the user to manually change the value of the continuous variable. Continuous knobs include the spindle-speed multiplier knob and the feed rate multiplier knob (Figure 6). Figure 8c shows the IVRESS definition of the spindle-speed multiplier knob. The user can rotate the knob between angles 0.3 to 5.7 radians from a value of 0 to 200%. When the user clicks on the knob it rotates a value of “smallChange” or 2%. The user can also scroll the mouse wheel to turn up/down the knob.



Figure 6 Virtual machine controller.

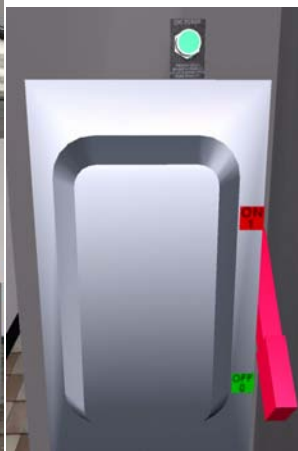


Figure 7 Virtual machine power box.

```

(a)
DEF Com_v3016_startButton ButtonWidget {
  buttonObject USE vmc3016_startButton
  buttonProperty yTranslation
  offPosition 0 #off
  onPosition -0.01 #press
  animationTime 0.15
  toggle 0
  arrowObject USE vmc3016_startButton_arrow
}

(b)
DEF Combo_v3016_axisKnob ButtonWidget {
  buttonObject USE vmc3016_cont_AxisKnob
  buttonProperty rotationAngle
  numPositions 7
  position0 2.5 #X
  position1 3.0 #Y
  position2 3.5 #Z
  position3 4.0 #A
  position4 4.5 #B
  position5 5.0 #C
  position6 5.5 #Remote
  animationTime 0.1
  toggle 1
  value 0
}

(c)
DEF Slider_v3016_spindleSpeedMultiplier DialWidget {
  dialObject USE vmc3016_cont_SpindleKnob
  dialProperty rotationAngle
  positions [0.3 5.7]
  values [0 200]
  value 0
  smallChange 2
  largeChange 10
  speed 60
}

```

Figure 8 IVRESS object definitions for (a) a button widget for modeling the controller green start key; (b) axis-control selector knob; and (c) the spindle speed multiplier knob.

The simulation of the machine logic is performed through event-handling subroutines, which propagate state changes to other machine objects. Figure 9 shows the event-handling routine of the machine green CNC power button (see Figure 6). When the user clicks this button, if the machine is already on, then nothing happens. If the machine is off and the red power switch is on, then the machine is powered on. This is done by setting the caption of the “Lab\_MachinePower” object to 1 (which in turn triggers the “change” event-handling routine

associated with the “Lab\_MachinePower”) and calling the video screen switch and light switch event handling subroutines in order to check whether the video screen and machine light should be turned on. Another example of an event-handling routine is the machine controller “Light switch” (Figure 6) ButtonWidget click event-handling subroutine. It contains an instruction to set the “on” property of the IVRESS “DirectionalLight” object which represents the lamp inside the CNC machine enclosure to the value of the ButtonWidget.

```

Sub Com_v3016_PowerButton_Click()
If Lab_MachinePower.caption = "1" Then Exit Sub

If Chk_v3016_PowerSwitch.value = 1 Then
  Lab_MachinePower.caption = "1"
  Call Chk_v3016_video_Click
  Call Chk_v3016_light_Click
  Scr_main.visible = True
  Lab_AVML.visible = True
  Lab_CurrentScreen.caption = "Scr_main"
End If
End Sub

```

Figure 9 IVRESS event-handling subroutine for the machine power button.

### 3.2.2 Machine Motions Engine

The machine motions engine handles the motion of: the stages (axes); tool changer; tool changer turret; and spindle rotation. Machine motions are handled using the state-variables and event-handling routines. For example, the “xAxisPosition”, “yAxisPosition”, and “zAxisPosition” state variables store the absolute position of the x, y, and z stages positions. Any action, which changes the stage position, such as jogging the axis, or executing a G0 (rapid travel) or G1 (linear-interpolation) commands, changes these state variables. When these state variables are changed, the axis change event is called which calls the associated event-handling routine. This routine updates the translation of the IVRESS object which represents the stage in the VE Engine. It also updates the values of software emulator text boxes, which show the position of the stage.

### 3.2.3 Software Emulator

The software emulator models the software screens, which are displayed on the screen of the machine controller (Figure 10). Each software screen is modeled using grouping frames, text labels, input text boxes, and buttons. The machine software is written in Visual Basic 6.0. IVRESS can directly read and display Visual Basic forms and controls such as labels, text boxes, buttons, and sliders. IVRESS also has a VB-script interpreter, which can execute the Visual Basic event-handling routines on the Visual Basic form.

One of the machine logic state variables is the “current software screen.” The event-handling routine for each controller keyboard key uses the “current software screen” state variable to model the software and machine behavior when the user clicks on a machine key. For example if the user is in one of the “quick keys” menu screens (Figure 10a) and clicks on “space bar” then, the “space bar” key click event-handling routine is triggered. This routine displays the next “Quick keys” menu screen based on the value of the “current software screen” variable.

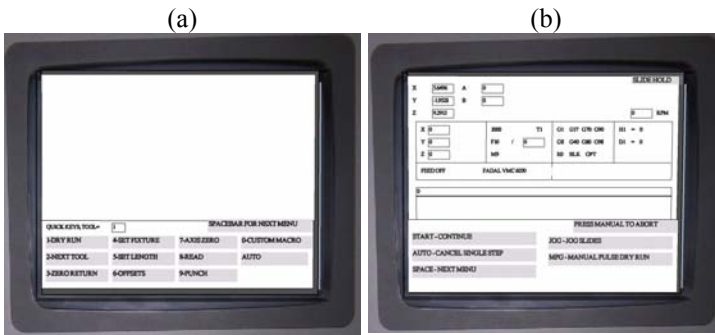


Figure 10 Typical machine software screens. (a) Quick keys menu number 1 and (b) program start screen.

### 3.2.4 NC-Code Interpreter

The NC-code interpreter loads the whole NC-Code program. The program is then broken down into blocks or lines separated by carriage return. The blocks are then broken down into individual NC-code words. Each word has an associated script subroutine which performs the task defined by the word. The NC-code subroutine changes the values of the machine state variables which in turn reflect the change in the VE and trigger associated event handling-routines. For example, the G1 linear interpolation subroutine sets the “xAxisPosition,” “yAxisPosition,” and “zAxisPosition” state variables, which in turn propagate the changes to the machine stages and the software text boxes. Typical NC-commands include: rapid travel, linear and circular interpolation, tool selection, tool change, spindle on/off, spindle rotation direction, setting the spindle speed, and setting the feed-rate.

## 4. VIRTUAL ENVIRONMENT ENGINE

The visual and aural model of the machine is driven by the virtual environment engine (IVRESS). IVRESS is an object-oriented scene-graph based virtual-reality display engine. IVRESS has four classes of objects that can be used to construct VEs:

- *Interface objects* include many types of user interface widgets (e.g. label, text box, button, check box, slider-bar/dial/knob, table, and graph) as well as container objects (e.g. Group, Transform, Billboard, etc).
- *Geometric entities* represent the geometry of the various physical components. Typical geometric entities include indexed face-set surface, box, cone and sphere.
- *Finite elements* represent solid and fluid computational domains.
- *Support objects* contain data that can be referenced by other objects. Typical support objects include material color, position coordinates, and interpolators.

All objects in IVRESS have the same basic structure. Each object has properties that determine its state and behavior, and methods, which are functions that it can perform. In addition, interface objects have events that are triggered when certain conditions, initiated by the user or the passage of time, are met. An event is triggered by calling a script-subroutine associated with that event. The subroutine name consists of the object name concatenated with an underscore and the event name (e.g., *object-name\_event-name*). IVRESS includes two scripting languages VB-script and ECMA-script. Scripting

allows setting the properties of the various objects, and writing custom event handling routines. IVRESS can interface with output devices, including immersive stereoscopic screen(s) and stereo speakers; and a variety of input devices, including body tracking (head and hands), haptic gloves, wand, joystick, mouse, microphone, and keyboard. IVRESS can read and write file formats for geometry data such as VRML 2.0 [27], pictures such as Bitmaps, PNG, JPEG, and GIF; and movies such as MPEG and AVI. IVRESS objects perform the following functions:

- Display a textured photo-realistic animated model of the machine and lab (Figure 11).
- Boolean-based solid-modeling for geometric simulation of the cutting process through ACIS™ solid modeling kernel.
- Generate the machine sounds.
- Display animated photorealistic human avatars of virtual instructor(s) through the Haptek™ API.

### 4.1 Machine/Lab Model

The machine model (Figure 11) is a hierarchical scene-graph that includes the machine geometry and the machine control widgets. The machine geometry consists of textured surfaces of the various machine parts. The machine parts were modeled in Pro-Engineer Wildfire® as solid models and then exported in VRML format as tessellated surfaces. The surfaces were textured using photographs of the actual machine to produce a near-photorealistic model of the machine. The machine control widgets include: “ButtonWidget” and “DialWidget,” which are used to model the machine buttons, switches, discrete knobs, continuous knobs, and dials (see Section 2.3.1). A “SpeedControl” object is used to set the spindle rotational angular velocity at the desired value. In addition to the machine, a workpiece preparation area (Figure 12) and the lab room (Figure 13) were modeled. The workpiece preparation area allows the user to select the cutting tools, place the parallels in the workpiece fixtures, use the mallet to secure the workpiece on the fixture and define the dimensions of the raw workpiece block. The lab room will eventually contain other CNC machines such as a CNC lathe.



Figure 11 CNC machine Virtual Model.

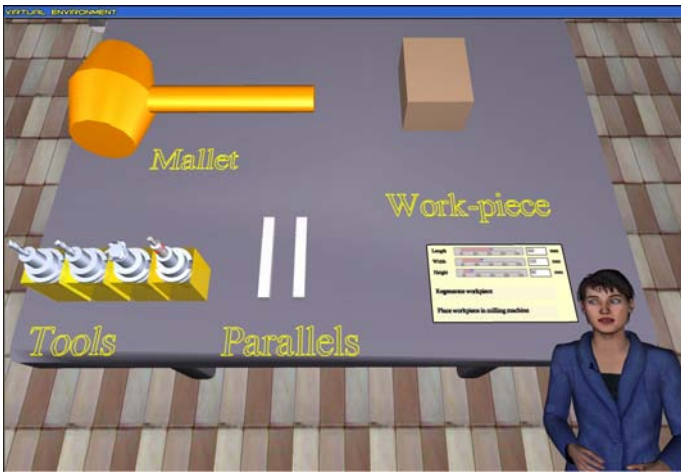


Figure 12 Work-piece preparation area.



Figure 13 Virtual manufacturing lab.

## 4.2 Solid Modeling Kernel

ACIS<sup>®</sup>, a commercial object-oriented geometric modeling engine from Spatial Technology Inc. [12], is used to perform the geometric solid modeling of the material removal process. ACIS<sup>®</sup> provides the following functions that are needed to implement the basic geometric procedures of the geometric simulation: Boolean intersection and subtraction operations, sweeping, extraction of polyhedral representation of the workpiece and the chip, and calculation of the volume of a solid. ACIS<sup>®</sup> is integrated with IVRESS through ACIS C++ API. Two wrapper objects, one for a solid and one a solid Boolean operation were created in IVRESS. Those objects along with the geometric simulator described in Section 3.1.1 are used to update the work-piece geometry during cutting in-real time (see Figure 5) and to generate the cut volume geometry, which is used to calculate the cutting forces.

## 4.3 Machine Sounds

IVRESS includes the VRML AudioClip and Sound node (object) types. The AudioClip object specifies audio data in the form of a WAV file that can be referenced by Sound objects. The Sound object specifies the spatial representation of a sound in the VE, including location, direction, intensity, and sound ellipsoid. IVRESS uses DirectSound3D to play the sound with the correct intensity and spatialization and to mix sounds.

The sounds of: switching on/off the machine, toggling the various controls (buttons, switches, and knobs), opening/closing the machine door, tool change, etc. were recorded into WAV files. The sounds are synchronized with the actions/motions of the various parts of the machine by executing a sound play method in the event-handling routines associated with the corresponding action.

The spindle rotation sound was simulated using the following technique. The spindle sound at 2000 RPM was recorded. Then the pitch AudioClip property is varied from 0 to 3.75 based on the spindle RPM to simulate the spindle sound from 0 to 7500 RPM. The sound is then played with the “loop” property of the AudioClip set to true. This technique was found to produce a close-enough sound to the actual spindle sound.

## 4.4 Humanoid Model Engine

A wrapper IVRESS object encapsulates the Haptik API [16]. This allows displaying full body textured highly detailed male and female characters in a VE. The character has a large set of pre-defined gestures. Typical gestures include: looking up, down, right and left; torso bend, twist, and bow; right/left hand; smile; blink; walk; etc. In addition, the gestures also include the visemes (e.g. aa, ih, g, s, eg, uh, etc.) which are lip and face positions for lip-synching. Also, the API allows setting the character’s joints rotations and positions to any desired value. In addition, the IVRESS wrapper object allows animation of the character hand motions by linear interpolation of the joint positions or angles. Two humanoid instructors are used in the AVML (see Figure 13):

- An “always on top of the screen” agent or screen agent is always visible and provides guidance and answers to the user’s questions.
- An assistant in the environment directly assists the user by showing the correct way to perform the process steps.

The two agents are controlled using the LEA engine and collaborate to assist the student. For example, when the student asks the screen agent a question, the screen agent will answer and display any visuals which support the answer (See Section 5.1). The screen agent will also command the assistant in the VE to do a demonstration if necessary. Visual cues such as 3D flashing arrows are used during the demonstration (Section 5).

## 5. INTELLIGENT AGENT ENGINE

LEA [20] is an intelligent-agent engine which includes facilities for speech recognition and synthesis, a rule-based expert system natural-language interface (NLI) for recognizing the user’s natural-language commands [19, 20], a hierarchical process knowledge base engine [20], and an unstructured knowledge base engine. LEA is the engine behind the AVML web-based framework. LEA is encapsulated in an ActiveX control which can run in a web-page and can display various user defined, sizable and movable mini-web browsers sub-windows (that can display any web content such as HTML, Flash, etc.)

### 5.1 Multimedia Lectures

One of the capabilities of LEA is the unstructured knowledge base engine. The unstructured knowledge consists of “knowledge items” that can be arranged in any order to form

a lecture. A knowledge item is a short HTML segment with embedded multimedia content and LEA-specific tags. A knowledge-base file is simply a collection of knowledge items separated by new-line characters. LEA allows multiple knowledge-base files to be loaded. A file is loaded using its URL. Thus the knowledge base can lie anywhere on the web. LEA includes a search capability in order to be able to answer the user's questions from the unstructured knowledge-base. The following lectures were developed using LEA:

- *Introductory lecture to CNC milling.* Before the hands-on training starts, the screen agent can give the trainee a brief introductory overview on CNC milling. The lecture outline is in the window labeled "Introduction" (see Figure 1). The outline allows the user to skip to specific section in the lecture or return to previous sections. Figure 14a shows an exploded view of the "Introduction to Machining Centers" lecture outline. Figure 15a shows a snapshot of the lecture in which the screen agent is describing the types of CNC milling machines.
- *Introductory lecture to the FADAL CNC machine.* The screen agent automatically proceeds to introduce the FADAL CNC machine. The basic components of the machine are introduced. It functions the same as the introductory lecture to CNC machining process. Figure 14b shows an exploded view of the outline. In Figure 15b, the screen agent is describing the base of the FADAL CNC machine. The other parts of the machine are made translucent so that the user can clearly see the base.

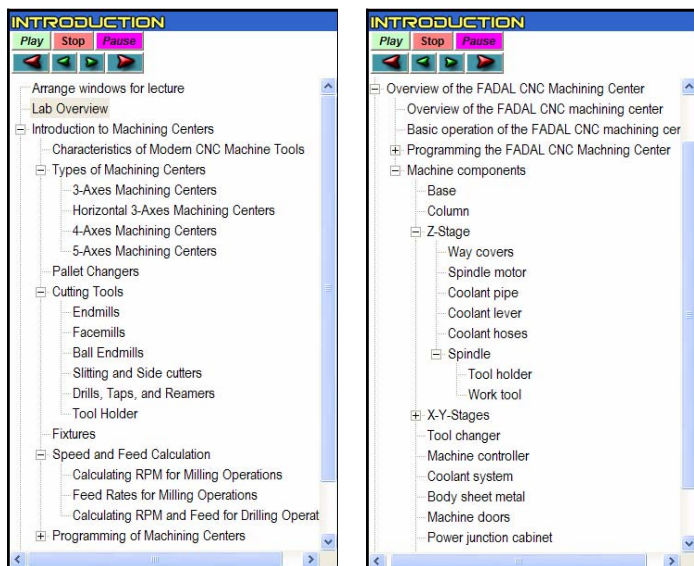


Figure 14 Partial exploded view of the introductory lectures outlines (a) CNC milling; (b) CNC machine components.

## 5.2 Step-by-Step Process Training

LEA also includes a structured hierarchical process knowledge capability that allows training users step-by-step through operating procedures [21]. Each process consists of a set of steps as well as other sub-processes. Each process and step can have pre- and post- constraints. Pre-constraints have to be satisfied before the step/process can be started. Post-constraints have to be satisfied before the step/process is completed. The agent can disseminate process knowledge using one of the following training modes. The desired mode is

triggered using a natural-language user's command interpreted using the LEA rule-based expert system. The training modes are:

- *Process Tutor.* The agent performs the process steps while the user is watching. The user can pause/resume, repeat (go back) a step, or skip a step.
- *Process Info.* The tutor mode will only recite the process steps to the user.
- *Process Guide.* The agent guides the user step by step through the process. The agent will not go to the next step until the user says a command such as "continue" or "proceed." The user has to perform each step. The agent checks the process constraints to determine if the user performed the step correctly. If a constraint is violated, then the agent instructs the user to repeat the step. If the user does not perform the step correctly three times in a row, then the agent performs the step.
- *Process Supervisor.* The agent instructs the user to perform the process. At the end of each sub-process the user lets the agent know that s/he is done. At that point the agent checks the sub-process constraints. If no mistakes were detected, then the agent instructs the user to perform the next sub-process. If mistakes are detected, then the agent lists the mistakes and instructs the user to repeat the sub-process.
- *Process Certification.* This mode is similar to the process supervisor mode except that the user performs all the process steps and the agent keeps track of any mistakes the user made. Finally, the user indicates that he completed the process and the agent informs the user of the mistakes made in the process. If no mistakes are detected then the agent certifies the user in this process.
- *Intelligent Virtual Assistant.* The user asks the agent to perform a process. The agent performs the process while the user can either watch the agent or do something else.

The following CNC milling machine training processes were developed: machine start-up; machine shut-down, load G-code from disk, and running an existing G-code. For example the machine-start up process consists of 6 steps:

1. Going to the milling machine power junction cabinet.
2. Switching on the red power lever (see Figure 16).
3. Pressing the green power button.
4. Going to the milling machine controller.
5. Pressing the green start button.
6. Pressing the keyboard manual button.

## 6. CONCLUDING REMARKS

A prototype virtual lab that includes a modern CNC milling machine was presented. A framework that includes the models and capabilities required for training on the operation of the CNC machine was developed. The following basic training functions were demonstrated using the prototype:

- Access to near-realistic virtual CNC milling machine, including the machine manual controller and geometric/physical simulation of the cutting process.
- Process training on key operating procedures of the CNC machine.

- Multimedia lectures on the concepts of CNC milling and on the functions of the various components of CNC milling machines.

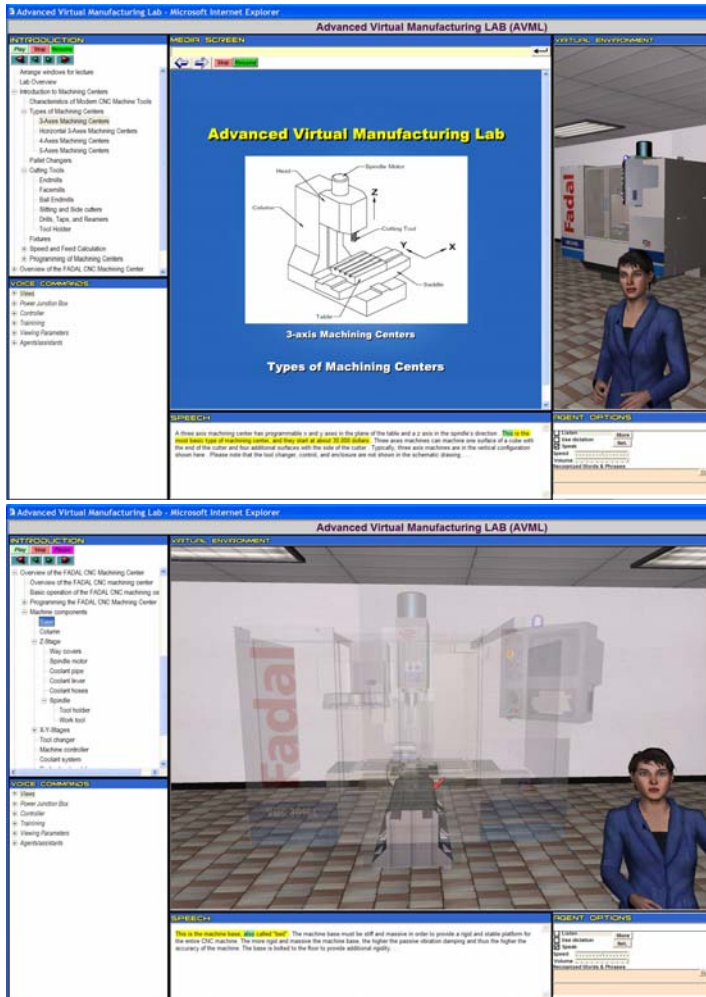


Figure 15 Snapshots of the introductory lectures on CNC milling (top) and the CNC machine components (bottom).

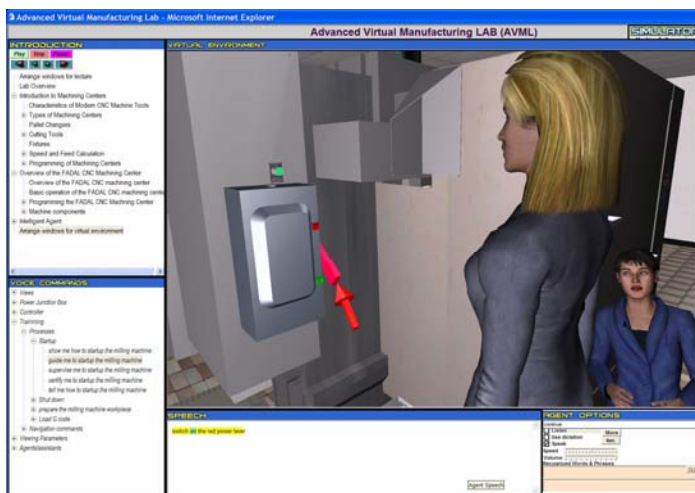


Figure 16 A snapshot taken during training step 2 of the machine start-up procedure.

## ACKNOWLEDGMENTS

Support for the AVML project was provided by NSF under STTR grant number DMI-0339024 and Indiana 21<sup>st</sup> Century Research and Technology Fund under STTR matching grant number 0043. The authors would like to thank Haptek Inc. for providing the human-like avatars display engine and Spatial Corp. for providing ACIS solid modeling kernel. IVRESS and LEA were provided by Advanced Science and Automation Corp.

## REFERENCES

1. Suh, S-H., Seo, Y., Lee, S-M., Choi, T-H., Jeong, G-S., Kim, D.-Y., "Modeling and implementation of internet-based virtual machine," *International Journal of Manufacturing Technology*, Vol. 21, pp. 516-522, 2003.
2. Lin, F., Ye, L., Duffy, V.G., Su, C-J., "Developing virtual environments for industrial training," *Information Sciences*, Vol. 140, pp. 153-170, 2002.
3. Ong, S.K. and Mannan, M.A., "Virtual reality simulations and animations in a web-based interactive manufacturing engineering module," *Computers and Education*, Vol. xx, 2004.
4. Ong, S.K., Jiang, L. and Nee, Y.C., "An internet-based virtual CNC milling system," *International Journal of Manufacturing Technology*, Vol. 20, pp. 20-30, 2002.
5. Luo, Y.B., Ong, S.K., Chen, D.F. and Nee, A.Y.C., "An internet-enabled image- and model-based virtual machining system," *International Journal of Production Research*, Vol. 40(10), pp. 2269-2288, 2002.
6. Kong, S-H., Park, J., Han, Y-G., Kim, G. and Lee, K-I., "The internet-based virtual machining system using CORBA," *Integrated Manufacturing Systems*, Vol. 13(5), pp. 340-344, 2002.
7. Chang, C-F., Varshney, A., and Ge, Q.J., "Haptic and aural rendering of a virtual milling process," *ASME 2001 Design Engineering Technical Conference and Computers and Information in Engineering Conference, Pittsburgh, PA*, pp. 105-113, 2001.
8. Abari, F. and Elbestawi, M.A., "Closed form formulation of cutting forces for ball and flat end mills," *International Journal of Machine Tools and Manufacture*, Vol. 37(1), 1997.
9. Ustaroz, A., Lozano, A., Matey, L., Siemon, J., Klockmann, D., and Berasategi, M.I., "VIRTOOL – virtual reality for machine-tool training," *Mecanique & Industries*, Vol. 5, pp. 207-212, 2004.
10. <http://www.denford.co.uk/vrmilling.htm>
11. [http://www.delmia.com/gallery/pdf/DELMIA\\_VirtualNC.pdf](http://www.delmia.com/gallery/pdf/DELMIA_VirtualNC.pdf)
12. <http://www.spatial.com>
13. Snow, M.P. and Williges, R.C., "Empirical models based on free-modulus magnitude estimation of perceived presence in virtual environments," *Human factors*, Vol. 40(3), pp. 386-402, 1998.
14. IVRESS (Integrated Virtual Reality Environment for Synthesis and Simulation), <http://www.ascience.com/Science/ScProducts.htm>, *Advanced Science and Automation Corp.*, 2004.
15. LEA (Learning Environments Agent), <http://www.ascience.com/Science/ScProducts.htm>, *Advanced Science and Automation Corp.*, 2004.
16. [www.haptek.com](http://www.haptek.com)
17. Tandon, V. and El-Mounayri, H. "A Novel Artificial Neural Networks Force Model for End Milling," *International Journal of Advanced Manufacturing Technology*, Vol. 18, pp. 693-700, 2001.
18. Briceno, J., El-Mounayri, H. and Mukhopadhyay, S. "Selecting an Artificial Neural Network for Efficient Modeling and Accurate Simulation of the Milling Process," *International Journal of Machine Tools and Manufacture*, Vol. 42, pp. 663-674, 2002.
19. Wasfy, T.M. and Noor, A.K., "Rule-based natural-language interface for virtual environments," *Advances in Engineering Software*, Vol. 33(3), pp. 155-168, 2002.
20. Wasfy, H.M, Wasfy, T.M. and Noor, A.K., "An interrogative visualization environment for large-scale engineering simulations," *Advances in Engineering Software*, Vol. 35(12), pp. 805-813, 2004.
21. Wasfy, A.M., Wasfy, T.M. and Noor, A.K., "Intelligent virtual environment for process training," *Advances in Engineering Software*, Vol. 35(6), pp. 337-355, 2004.