A WEB-BASED INTERACTIVE TEACHING PACKAGE IN MANUFACTURING ENGINEERING

SK Ong[1] and MA Mannan[2]

Abstract *— Increasing demand for a greater reach out from lecturers to students has resulted in the development of web-based multi-media packages that serve to aid the lecturers as teaching tools. This paper presents a web-based interactive teaching package that provides a comprehensive and conducive yet dynamic and interactive environment for a module on automated machine tools in the Manufacturing Division at the National University of Singapore. The use of Internet technologies in this teaching tool makes it able to conjure visualisations that cannot be achieved using traditional teaching materials such as transparencies. This is especially useful in the teaching automated machine tools, which deals primarily with the numerical control (NC) of the motions of automated machine tools. This teaching package provides web-based simulations and animations to enhance students' understanding. These simulations are suitably placed in the package to generate interactions. Customised question types were also designed and implemented with a tutorial monitoring application.*

Index Terms — Animation, Simulation, Interactivity, Java

INTRODUCTION

Increasing demand for a greater reach out from lecturers to students has resulted in the development of web-based multi-media packages. This coincides with the rapid acceptance of the Internet as the tool for disseminating information [1, 2]. The use of the Internet for the teaching of subjects is not new. Many instances of these teaching packages can be found on the WWW. The development of these web-based multi-media teaching packages strives to produce a comprehensive, conducive, dynamic and interactive environment for the purpose of imparting knowledge in the various disciplines in the curriculum [3, 4].

Previously, the mode of teaching is one lecturer to many students. Currently, the use of the Internet provides a new paradigm for imparting knowledge. While it does not have the flexibility in teaching by a lecturer, it is able to conjure visualisations that the lecturer cannot provide using tools such as transparencies, whiteboard, and papers. This is especially useful in the teaching of manual and automated programming of CNC control machine tools, which deals primarily with the NC control of the motion of cutting tools in manufacturing. It is not feasible economically to provide a workpiece to be machined every time a student learns a new NC code. Hence, it is timely to provide for an alternative, which is to use computer simulations and animations to enhance students' understanding of the topic.

Web-based teaching applications resolve a peculiar problem observed in Asian schools. Asian students are seemingly unwilling to participate in lectures and tutorials. The reasons are many and shyness is one of them. Web-based teaching packages allow shy students to participate in the course at their own time and in their own personal space.

On the point of dynamism and interactivity, the use of web-based teaching packages meshes the students and lecturers in a way that enhances time and spatial co-ordination. Busy lecturers will not have to see streams of students at different times, performing the monotonous repeated explanations of the same concepts. Similarly, time constraint students will not have to make appointments to seek subject understanding from the lecturers. Students can seek answers to problems that require immediate attention by sending them to a forum for discussion, whilst awaiting the response from the lecturers. Hence, both lecturers and students would be able to discuss questions and answers behind terminals, not having to book lecture halls or extra tutorial sessions as such sessions can be done at computer clusters or at home.

Despite all the advantages of web-based teaching packages, they should not be used to replace lecturers and lectures in its entirety. More useful as a supplement than a replacement, the implementation of these packages allows the users to better co-ordinate time, enhances learning and provides for a better working environment.

This paper presents a web-based interactive teaching package for a third year module, Computer Automated Manufacturing Systems (CAMS) in the Manufacturing Division at the National University of Singapore. HyperText Mark-up Language (HTML) is used for displaying of information, embedding of Java applets, and Virtual Reality Modelling Language (VRML) plug-ins. VRML is used to create the virtual machining environments. Java applets are used to provide simulations and animations. Two-dimensional (2D) simulations demonstrate the tool motions

[1] S K Ong, National University of Singapore, Mechanical Engineering Department, 10 Kent Ridge Crescent, Singapore 119260
[2] M A Mannan, National University of Singapore, Mechanical Engineering Department, 10 Kent Ridge Crescent, Singapore 119260

of the NC codes. There are some interactive applets which behaviour depends on the data input from users. Three-dimensional (3D) simulations are provided for using VRML. Many of these simulations will generate interactions between Java and VRML that require user inputs using the External Authoring Interface (EAI) technique. The display of the VRML worlds will require the use of Cosmoplayer plug-in. Tutorial questions and a tool for monitoring the students' progress were also implemented.

DEVELOPMENT TOOLS

HyperText Mark-up Language

HTML is the standard for writing webpages [5]. It is used to develop the CAMS web-based teaching package. Some JavaScripts in this teaching package were also embedded within the HTML format. For example, users on moving the mouse over a certain line in HTML will see effects such as text or document changing colour. Other display devices such as VRML plug-ins were also embedded within HTML. HTML thus provides the tool to format the display in an organised manner, without which the webpages would have been less colourful, interactive and interesting.

Java Applets

When the Internet gains popularity because of its neutral platform, it faces other problems such as the vast number of protocols produced when content developers integrate new types of media. This meant that webpages written using certain protocols would not be usable by all visitors. Since it was not feasible for manufacturers to build-in every new protocol that comes along, the industry has come to accept Java applets as the standard to address this protocol problem [6, 7]. Java can be used to integrate new protocols for use on the browsers. Java applets have been used extensively in this package. The EAI technique is employed to integrate Java with another display media.

Virtual Reality Modelling Language

VRML was developed to bring people closer to reality on the WWW by enlivening displays and simulations in the 3D format [8]. VRML allows a user to uniquely define a scene with its vast library of codes, which include many types of sensors and interpolators. There are many authoring tools currently available in the market for developing VRML worlds. A few examples are the Platinum's VR creator and the ParaGraph's Virtual Home Space Builder. For this project, the VRML worlds were modelled using UniGraphics and V-REALM builder. UniGraphics is user-friendly in the modelling of 3D objects, whilst V-REALM, while lacking in its modelling capability, is useful for animating objects and entities in the scenes with its ready array of sensors buttons and key frame animation.

External Authoring Interface

The capability to control and manage events is gained from using Java applets. The modelling and animation scenes are gained through using VRML. However, for external events to control the animated scenes, either the JavaScript Authoring Interface (JSAI) or the EAI is required. JSAI is easier to master and control, but not suitable for use on highly detailed scenes. EAI is more involved computationally as it uses Java applets to link with the VRML worlds [9]. This allows for more flexibility in events handling, but is harder to master. In this project, where tutorial questions are expected to solicit dedicated 3D motions of the cutting tools, the EAI technique was chosen over JSAI. To activate EAI, the *vrml.external* package has to be imported during compilation. This will allow Java applets to manipulate behaviours in the VRML worlds. There are four steps in using EAI to control entities in a scene, namely (1) getting a reference to the VRML browser, (2) getting a reference to a VRML node, (3) sending information from an applet to VRML as *EventIn*, and (4) sending information from VRML to entities/objects as *EventOut*.

Monitoring Tool

In this project, a monitoring tool, namely, the Flying Fish developed by researchers from the University of Western Australia, is incorporated to monitor the progress of students. It is a tutorial environment that is capable of monitoring the student progress, setting deadlines, as well as the awarding of marks. The Flying Fish environment allows administrators and lecturers to set tutorial questions that are supported by Java. Security is improvised as students need to login using issued passwords. Such a monitoring system is convenient and useful, as it is not possible to monitor each student's progress in the normal tutorial lessons.

DEVELOPMENT TECHNIQUES FOR 2D JAVA APPLETS

2D Applets - Viewing

Figure 1 demonstrates the effectiveness of using Java to illustrate the motion of a cutting tool for a certain NC program. The applet in the display is divided into four main panels, two of which display the top and side views of the motion. For most of the 2D applets developed, only the top and side views are illustrated as these two views are sufficient in displaying the entire motion in this dynamic interface, as compared to a static display on paper. The middle panel lists the NC codes that are currently being executed by the applet. The last panel is a *TextArea* that prints lines of explanation corresponding to the position of the cutting tool. This is a useful feature as it highlights important points relating to the tool motion.

Figure 1 shows an instant after the applet has been initiated. When a user clicks on the RUN button, a set of reactions from the Java class is executed. The RUN button made a call to the applet, initiating a thread in the *run()* method in the class. In the *run()* method, a complicated mix of *while*-, *for*- and *if*-loops are initiated. Three combinations of these loops are used to control the motion of the graphical interface, one for the movement of the tool as seen in the top and side views, another for controlling the display of the list of NC codes, and one for controlling the messages to appear in the designated *TextArea*.

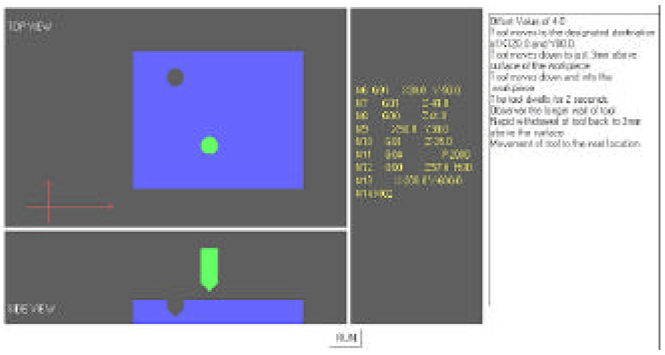
Next, the enabled thread starts to perform calculations in the *run()* method that define the pixel locations of the tool in the top and side views. The applet developed was not multi-threaded in controlling the motion of the top and side tools. The reason being that multi-threads are more difficult to control, utilise more resources and present a certain problem when prioritising the usage of the individual thread that demands resources from the computer. To achieve synchronised motions of the tool in the two views, the set of variables for the top view was selected to become the 'master' variables and that of the side view, the 'slave' variables. Whenever dual motion is required, the 'slave' variables would tag along on a ratio to the motion of the 'master' variables to ensure synchrony. Variables are placed in the *paint()* method of the Java class as the input values for the locations of the tools, where *paint()* is the method for displaying the graphical information in Java applets.

Markers are placed in the Java code for the removal of NC codes in the middle panel as in Figure 1, as well as for the appearance of messages in the *TextArea* panel. This can only be done, however, by 'framing' the entire process. In essence, the above motion falls under many different *for*-loops within a *while*-loop. Each different step of the framing process controls the appearance or disappearance of information in the middle and *TextArea* panels. This algorithm provides a clean and systematic way of development. After the cycles of frames have been completely displayed, it encounters the main *while*-loop, which enables it to run on indefinitely.

Interactive 2D Applets - 'insert the word'

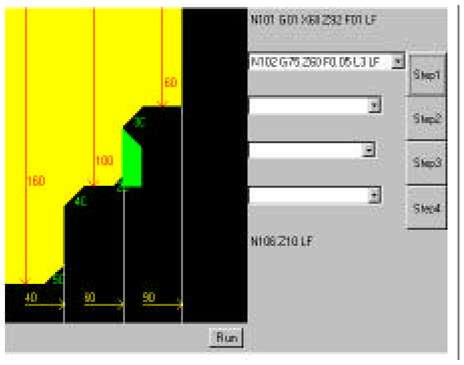
Like the applets in Figure 1, the applet in Figure 2 shows the tool motion for a certain set of NC codes. However, such applets are modified to provide interaction. Like the earlier type of applets, users will have to click the RUN button to set the tool moving to a predetermined point. Thereafter, users will have to correctly answer the 'insert-the-word' (*ITW*) questions provided in order to continue viewing the rest of the tool motion. A user, on answering the first question correctly and clicking on the 'Step1' button will be able to see the scene in Figure 2.

Providing the correct answer for the remaining questions will see the tool moving to the completion of a round. Clicking the RUN button will restart the cycle. The algorithms for such applets are similar to the *viewing* applets. Such applets come in handy to be placed at certain parts of the web-based package, as too much reading of information may reduce the viewer's attentiveness. The *ITW* question type is chosen because this allows all the selected codes to be displayed to the user in one glance. This enables easy spotting of erroneous answers as well as providing a clear understanding of the order of the codes to be placed.



Offset Value of 4.0
Tool moves to the designated destination at X120.0 and Y80.0
Tool moves down to just 3mm above surface of the workpiece
Tool moves down and into the workpiece
The tool dwells for 2 seconds
Observe the longer width of tool
Rapid withdrawal of tool back to 3mm above the surface
Movement of tool to the next location.

Figure 1: 2D Applets – Viewing.



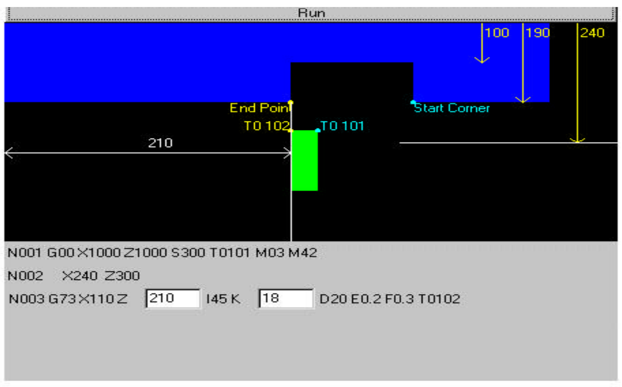
N101 G01 X68 Z92 F0.1 LF
N102 G75 Z60 F0.05 L3 LF
N106 Z10 LF

Figure 2: Interactive 2D Applet.

Interactive 2D Applets - 'fill in the blank'

Figure 3 shows another type of applets implemented. It consists of *Textfields* that require user input. In Figure 3, the inputs are parameters that will determine the final distance of the tool motion and the intervals at which the tool will cut. Figure 3 displays the effect of entering the default set of inputs from the course notes. It shows the retrieval of the tool after the final cut according to the input, where the cut-out portion of the workpiece is shown. The inputs are read into Java with the command *Integer.parseInt()* placed amongst the codes. Double *for*-loops are utilised in this example. One *for*-loop is used to control the repetitive cutting of the workpiece after every interval and another *for*-loop is for the motion in the negative Z axis, which is dependent on the input in the first *Textfield*. This kind of applets differs from the previous two types as it provides a greater degree of freedom and interactivity. The viewing type is generally passive, the *ITW* applets allow more engagement from the users but calculations were not needed.

Finally, the *fill-in-the-blank* applet allows for full engagement from the users.



```
N001 G00 X1000 Z1000 S300 T0101 M03 M42
N002     X240 Z300
N003 G73 X110 Z  210  I45 K  18  D20 E0.2 F0.3 T0102
```

Figure 3: “Fill in the blank” Applet.

DEVELOPMENT TECHNIQUES FOR 3D APPLETS

Virtual Reality Modelling Language

VRML is used for developing 3D objects. It is based on the ‘behaviour’ concept where each object in a scene possesses certain qualities. These qualities include data array sets such as *Position*, *Colour* and *Sensor.* Motion in a VRML world is achieved using interpolators. Interpolators are nodes that take a set of values, which define a range, and generate the intermediate values automatically. Input to the interpolators is known as *key* and *keyValue*. An array for a *key* consists of three values. These are the intersection values and can only range from 0 to 1. *keyValue* also consists of three sets of values, with three numbers per set separated by commas. Each set of these values represents the position data of an object. The applet in Figure 4 controls the position of the red tool. Labelling it as a *PositionInterpolator* would define these values to be used for storing the position data, preventing confusion with other interpolators that include the *OrientationInterpolators* and the *ScaleIntepolators*.

VRML also offers an entire range of sensors, which includes the *ProximitySensor, TouchSensor, VisibilitySensor* and *TimeSensor.* In this teaching package, *TimeSensors* are heavily used to control the initiation of a VRML scene. The *EventIn* fields of the *TimeSensor* include ‘enabled’, ‘cycleInterval’, ‘startTime’, ‘stopTime’ and ‘loop’.

For viewing 3D scenes, the Cosmoplayer plug-in was used. This user-friendly interface allows first-time users to manoeuvre the scene easily. By suitably placing viewpoints around the objects and naming them under ‘description’, users can view from different viewpoints of the tool and workpiece during the course of the tool motion.

Interactive 3D Applets - Viewing

The 3D applets in this package were done with the aid of the EAI method. The structure of the VRML language revolves around nodes, much as objects are to the Java language. Each node contains properties of a particular object. EAI allows Java applets to control and send messages to the VRML nodes, hence controlling its motion or appearance.

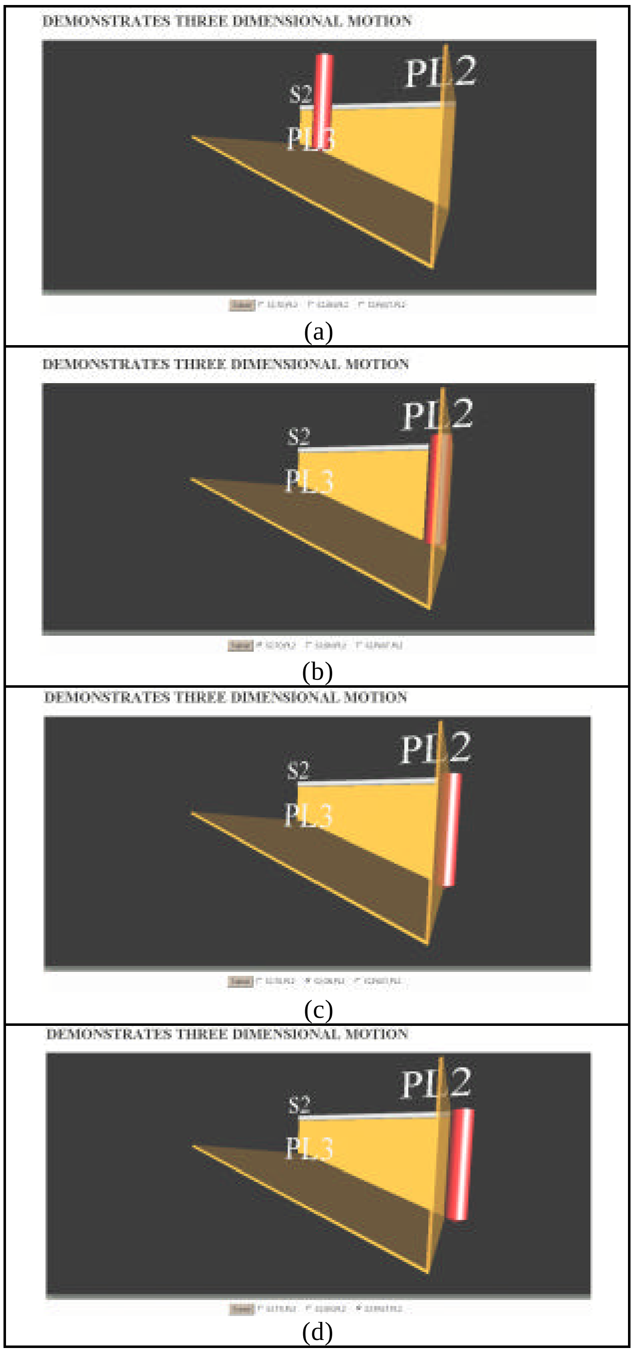

DEMONSTRATES THREE DIMENSIONAL MOTION
PL2
S2
PL3
(a)
DEMONSTRATES THREE DIMENSIONAL MOTION
PL2
S2
PL3
(b)
DEMONSTRATES THREE DIMENSIONAL MOTION
PL2
S2
PL3
(c)
DEMONSTRATES THREE DIMENSIONAL MOTION
PL2
S2
PL3
(d)

Figure 4: Interactive 3D Applet for Viewing.

Figure 4 shows an example of such an applet. In this example, the tool was to simulate the PAST, ON and TO positions of the tool relative to a plane defined as PL2. For such viewing examples, the *Checkbox* methods provided in the Abstract Windows Toolkit (AWT) of the Java package was utilised. This was preferred over other possibilities from the AWT such as the *Choice* methods or the *Textfields* methods. The focus of such applets was to demonstrate the motion of a tool in a 3D format. No high level interaction was necessary. The use of the *Checkbox* methods would provide an immediate assessment of the possibilities (PAST, TO and ON in this example) that the user may apply. Hence, *Checkbox* was chosen over other methods. Figure 4(b) shows the tool at the TO position. The tool travels down TO plane PL2 where the 3D profiling is defined by PL2 along S2. Using VRML, the block defined by PL2 was made to be semi-transparent. This allows users to clearly view the position of the red tool from an isometric viewpoint. Figures 4(c) and 4(d) show the ON and PAST positions respectively.

EAI is the interface between Java and VRML. To load a VRML browser from a Java applet, a reference must be made. The reference can be established by declaring a variable to hold an instance of the browser object in the applet using *Browser Instanceofbrowser*. The reference can then be obtained using the *getBrowser* method.

With the activation of the browser, communications between VRML and Java can be accomplished. In Figure 4, by clicking on the Submit button after selecting one of the *Checkbox* options, a message is sent to Java to return a true value to 'enabled', an *EventIn* field for a *TimeSensor*. The algorithm for this example is such that at any instant, all three positions of the tool are present. In short, three tools are constantly present and moving. However, selecting any of the *Checkbox* options would set off an *EventIn* process to VRML to make the tool of the two other unsolicited options to transparent. To the human eye, it thus presents itself as only one tool. Clicking on another option followed by the Submit button repeats the process albeit with the chosen tool as visible.

Interactive 3D Applets - Assessing

Besides viewing purposes, students can assess their own capability through the use of interactive 3D applets. Figure 5 shows such an applet with two main panels. The left panel is the 3D interface and contains the VRML plug-in, Cosmoplayer. The right panel shows a list of APT codes, and the pull-down tags are the interfaces developed using Java AWT. The heart of this applet is the EAI technique that is employed to connect Java and VRML.

When a user clicks on the Submit button, correct answers from the pull-down menus will allow the Java class to set EAI in motion, setting the value of the *EventIn* field 'enabled' in VRML to true. This sets a chain reaction in the VRML world, where the *KeyFrame Animators* is set to alert upon instructions from the *TimeSensor*. This animator will start the *PositionInterpolators* thereby changing the data sets of the objects in the VRML. Cosmoplayer will then capture the changing data arrays and convert it to visual.

The mechanisms of the assessment applets are similar to the viewing applets. The only difference is the step prior to activating EAI. Unlike the viewing applets, these applets have more conditions to satisfy before activating the EAI.

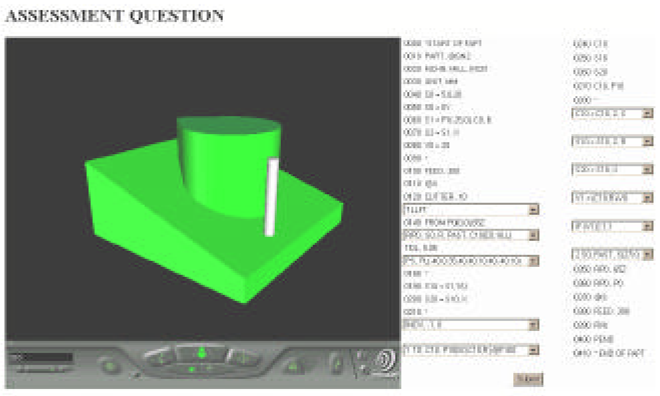


Figure 5: Interactive 3D Applets for Assessment.

TUTORIAL AND MONITORING ENVIRONMENT

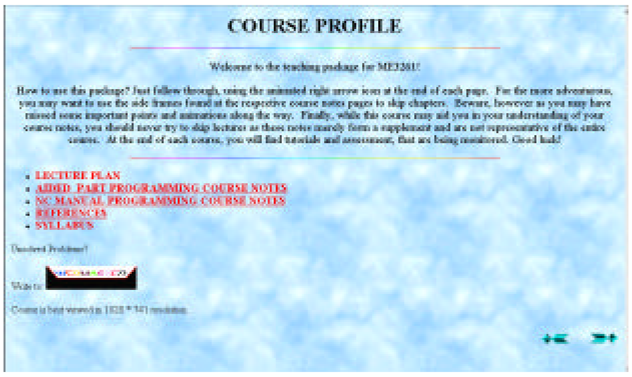
In this package, the Flying Fish tutorial and monitoring environment was selected and acquired from the University of Western Australia. The *ITW* type of questions was developed and linked to this tutorial environment. Flying Fish allows these tutorial questions to be set and corrected. It also allows the administrator or lecturer to monitor students' progress easily and conveniently. It also enables the administrator to observe the questions attempted wrongly, and the students who made no attempt to answer the questions. While students are logged on, this panel will appear on the administrator's screen showing the respective students' progress will produce a coloured square each time a student answers a particular question correctly.

WEB-BASED DEVELOPMENT

The layout of the teaching package was finalised and the developed applets, simulations and tutorials were mounted onto the WWW. Due to the number of webpages developed, only certain pages will be discussed and explained here. Figure 6 displays the main course webpage where the entire course notes are based and branched off from. Animated GIF files are used to captivate the attention of the users.

Figure 7 shows the frame-based webpage when the user clicks on the APT course notes in Figure 6. The left frame presents the entire course on APT, and each line on the left frame links to a new chapter. This allows students to easily manoeuvre around the course. The right frame contains the actual course notes to be read by the users. This right frame contains the embedded Java applets as well as all other peripherals used to enhance the course notes. Figure 7 demonstrates the usefulness of JavaScript. When the user places the mouse over the black underlined words in Figure 7, a pre-defined message will appear in the status window of the page. In Figure 7, the mouse was placed over the word PART, an explanation of PART is displayed in the status window at the bottom left corner. This helps to improve the readability and neatness of the webpage, instead of offering definitions and explanations on the same page.

Some course notes are presented in a question format as seen in Figure 8. Changing the format from a fact-giving one to a question-asking format increases the attractiveness of the page as it breaks the monotony. Instead, such questions may be used to highlight certain important details that students may tend to forget or miss out. Figure 8 shows the presence of a link to a page with embedded applet in the words "Minor words for Tool Position Specification". The reason for not embedding the applet on the same page was to shorten the loading time. Embedding all applets on the same page will cause a long downloading time due to the transmitting of information. This may irk the users who may feel that time was wasted waiting for the download of applets that were already viewed. Hence, all applets and a few loading intensive pictures are embedded on sub-pages linked to the main page.



COURSE PROFILE

Welcome to the teaching package for ME3281!

How to use this package? Just follow through, using the animated right arrow icon at the end of each page. For the more adventurous, you may want to use the side frames found at the respective course notes pages to skip chapters. Beware, however as you may have missed some important points and animations along the way. Finally, while this course may aid you in your understanding of your course notes, you should never try to skip lectures as these notes merely form a supplement and are not representative of the entire course. At the end of each course, you will find tutorials and assessment, that are being monitored. Good luck!

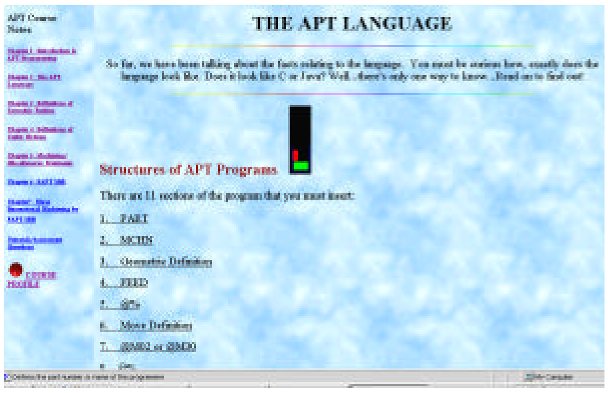
- LECTURE PLAN
- AIDED PART PROGRAMMING COURSE NOTES
- NC MANUAL PROGRAMMING COURSE NOTES
- REFERENCES
- SYLLABUS

Unsolved Problems?

Write to:

Course is best viewed in [illegible] resolution.

Figure 6: The Course Main Page.



APT Course Notes

THE APT LANGUAGE

So far, we have been talking about the facts relating to the language. You must be curious how, exactly does the language look like. Does it look like C or Java? Well...there's only one way to know...Read on to find out!

Structures of APT Programs

There are 11 sections of the program that you must insert:

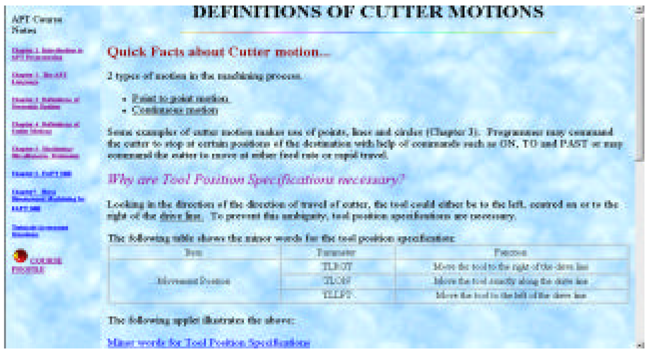
1. PART
2. MCHIN
3. Geometric Definition
4. FEED
5. [illegible]
6. Motion Definitions
7. [illegible]
8. [illegible]

COURSE PROFILE

Figure 7: Frame-based Layout.

CONCLUSIONS

In the new teaching package reported in this paper, Internet-based tools and techniques were used. The main form of presentation of information comes in HTML, as well as the use of JavaScript within HTML pages. These tools allow for a more captivating and neater method of display of information. The EAI technique was used to develop functional and interactive 2D and 3D applets for simulations and assessment questions. This teaching package includes materials on manual NC programming and APT programming. The successful implementation of this package saw the culmination of all the work involved to produce a package that is now comprehensive of the entire module, interactive in 2D and 3D, as well as incorporating a monitoring system. Techniques as well as tools developed for this package can serve to be templates for future reference and further development.



APT Course Notes

DEFINITIONS OF CUTTER MOTIONS

Quick Facts about Cutter motion...

2 types of motion in the machining process.

- Point to point motion
- Continuous motion

Some examples of cutter motion makes use of points, lines and circles (Chapter 3). Programmer may command the cutter to stop at certain positions of the destination with help of commands such as ON, TO and PAST or may command the cutter to move at either feed rate or rapid travel.

Why are Tool Position Specifications necessary?

Looking in the direction of the direction of travel of cutter, the tool could either be to the left, centred on or to the right of the drive line. To prevent this ambiguity, tool position specifications are necessary.

The following table shows the minor words for the tool position specifications:

Item	Parameter	Function
Movement Position	TLRGT	Move the tool to the right of the drive line
	TLON	Move the tool exactly along the drive line
	TLLFT	Move the tool to the left of the drive line

The following applet illustrates the above:

Minor words for Tool Position Specifications

COURSE PROFILE

Figure 8: Definitions of Cutter Motions.

REFERENCES

[1] Hites, J,M, Ewing, K, "Designing and Implementing Instruction on the World Wide Web: A Case Study", http:// lrs.stcloud.msus.edu/ispi/proceedings/html, 1996.

[2] Ward, R, "An HTML-based Case Scenario for Teaching Business Computing Skills", http://www.cti.ac.uk/publ /actlea/issue2/ward/index.html, 1996.

[3] Mandal, P, Wong, K,K, Love, P,E,D, "Internet-Supported Flexible Learning Environment for Teaching System Dynamics to Engineering Students", Computer Applications In Engineering Education, Vol. 8, No. 1, 2000, pp 1-10.

[4] Oreta, A,W,C, "Developing a Web-Based Learning Module in a Basic Civil Engineering Course", Computer Applications in Engineering Education, Vol. 7, No. 4, 1999, pp 235-243.

[5] Lemay, L, "Teach Yourself Web Publishing with HTML 4 in a Week", 4th Edition, Sams.net Publishing, 1997.

[6] Hughes, M, Hughes, C, Shoffer, M, Winslow, M, "Java Networking", Manning Prentice Hall, 1997.

[7] Weber, J, et al, "Special Edition Using Java", QUE Publishing, 1996.

[8] Carey, R, Bell, G, "The Annotated VRML 2.0 Reference Manual", Addison Wesley Developers Press, 1997.

[9] Anon., "External Authoring Interface", http://vrml.sgi.com/developer/eai/index.html.