Session 3263

Animations as support for the teaching of manufacturing

Marek Balazinski, Aleksander Przybylo

École Polytechnique de Montréal, Mechanical Engineering Department

Abstract
In this paper a variety of computer animations are presented. These animations are used during the Advanced Manufacturing course given in Mechanical Engineering Department at École Polytechnique de Montréal. The project has been realized using the Macromedias Flash 5 and Corel Draw 10 software. The teaching evaluation of the animations as a lecture tool proved that this new learning technology produces excellent results and enhances the teaching process.

Key words: teaching, manufacturing, simulation, animation.

1 Introduction

Teaching manufacturing processes requires students to acquire a good understanding of theories related to strength of materials, heat transfer, materials structure, etc. Manufacturing processes are often very complex and difficult to explain; therefore, the implementation of numerous laboratory sessions is required. Laboratory sessions are expensive, long to prepare and their efficiency is sometimes affected by parasitic phenomena that make the interpretation of laboratory results difficult. The use of films is also long and costly. In addition, films make it impossible to separate the different phenomena that come into play in a manufacturing process.

Computer animations and simulations more easily show the individual process of interest. Several studies [1, 2] find multimedia instruction both more effective and more efficient than conventional instruction. Recently developed software libraries and tools such as Macromedia Flash 5™ make the development of animations and simulations possible, even though they are not specially developed for this purpose.

During the Advanced Manufacturing course given in the Mechanical Engineering Department at École Polytechnique de Montréal, in order to explain some manufacturing problems, a variety of computer animations have been realized. The slide-shows used during this course (which were already containing short movies) have been enhanced with animations available through the course web site at the address: http://www.cours.polymtl.ca/mec4530/Anim/Menu.swf

2 Project context

The Center of Teaching and Learning at École Polytechnique de Montréal strongly supports the development of new multimedia teaching tools with the Information and Communication Technologies (ICT) program. Therefore, a mechanical engineering student developed the animations described in this paper during a four months internship.

Two different tools were used in the development of the animations. Corel Draw 10™ was used to create the figures, and Macromedia Flash 5™ with the integrated Action Script language was used to program the animations.

3 Animation description

A total of six animations presenting: chips compression ratio, types of chips, cutting forces, temperature distribution in the cutting zone, machine tool (lathe) rigidity, and electrical-discharge process were developed. The first five animations are interactive. Students may vary different parameters involved in each process, and observe the influence of the variation both on the process and on the results. The last animation shows the successive stages of the electrical-discharge machining.

3.1 Chip compression ratio

This animation presented on Figure 1 allows students to observe the influence of the rake angle (γ) variation on the chip compression ratio.

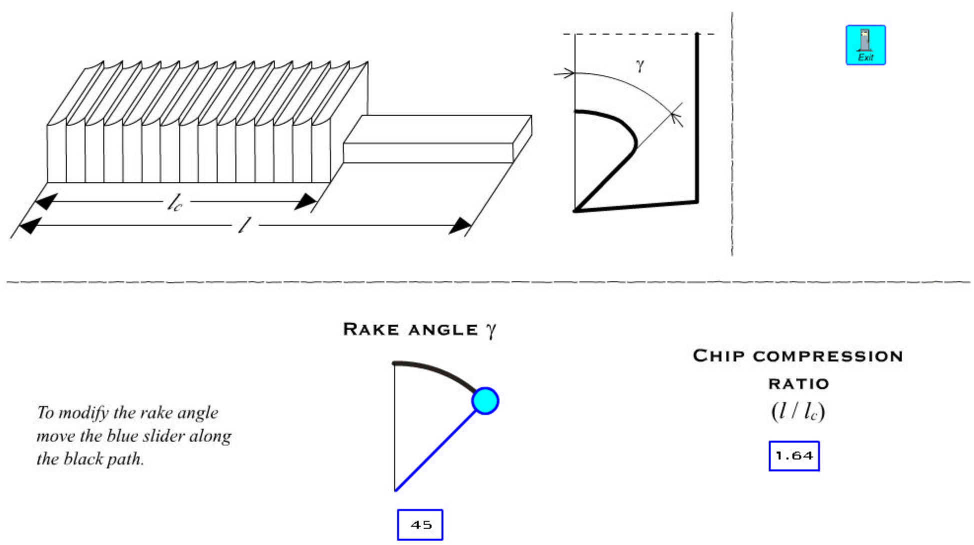

Figure 1 - Screen shot of chip compression ratio animation

3.2 Types of chips

This animation permits students to observe how different parameters (feed and depth of cut) influence the shape of the chip during the turning process. It permits also to distinguish between the appropriate and inappropriate chip shapes. The screen shot of this animation is presented on figure 2.

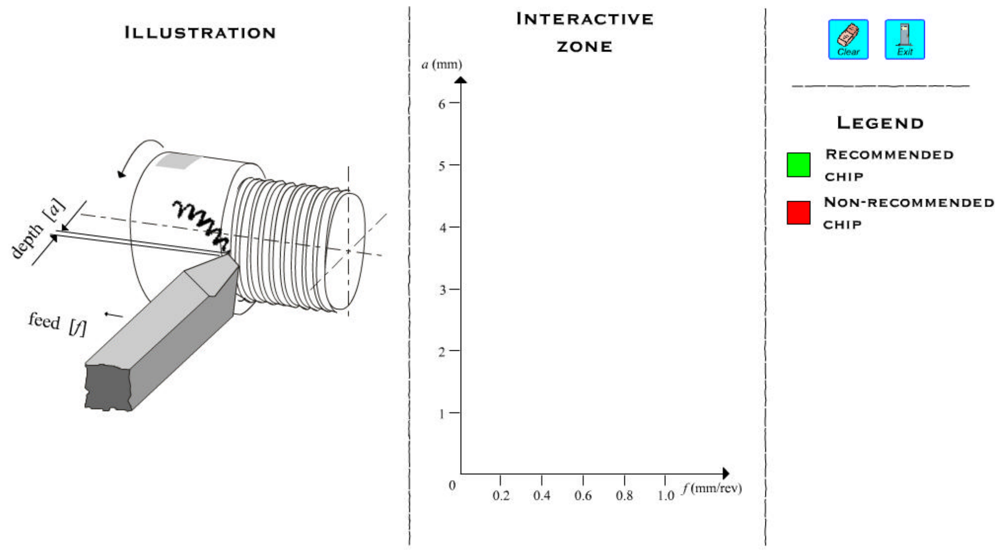

Figure 2 - Screen shot of type of chip formation animation

3.3 Cutting forces

This simulation permits students to observe the influence of different parameters on the cutting forces. These parameters include the cutting process parameters, the tool angles and the tool wear. An example of the animation in process is presented on figure 3.

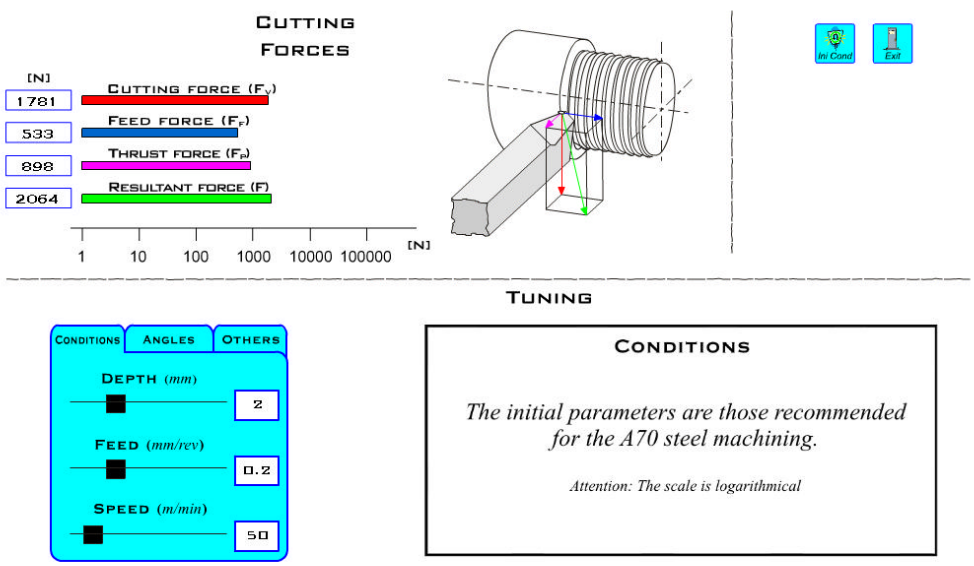

Figure 3 - Screen shot of the cutting forces simulation

3.4 Temperature distribution in the cutting zone

This simulation allows students to observe the influence of different machining process parameters on the temperature distribution in the tool-chip interface. The screen shot of this animation is presented on figure 4.

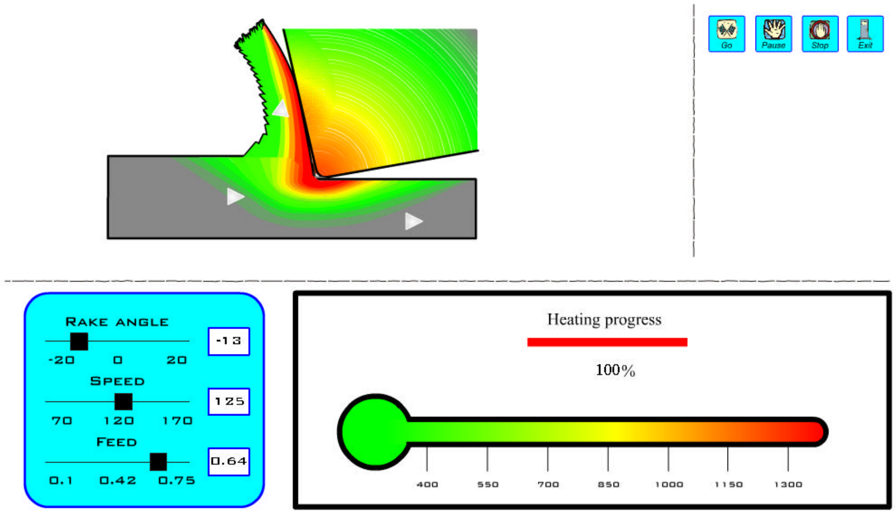

Figure 4 - Screen shot of the temperature distribution simulation

3.5 Machine-tool (lathe) rigidity

This animation permits students to experiment with various rigidity parameters configuration during the turning process. Students may set various rigidity of the lathes components (carriage, headstock, tailstock), workpiece rigidity, cutting force magnitude and observe the shape of the machined workpiece. It is possible to compare the results of different simulations. The animation may be paused at any time to show the current state of the system deformation. An example of the animation in process is presented on figure 5.

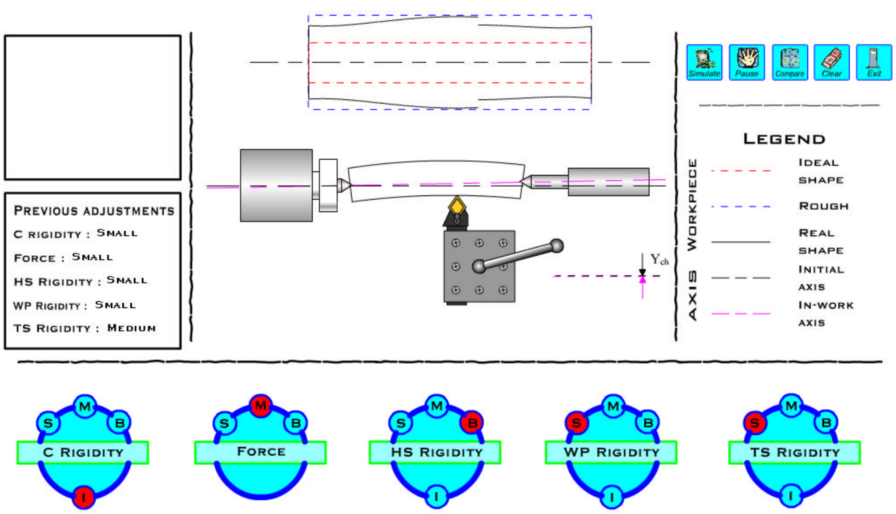

Figure 5 - Screen shot of the lathe rigidity simulation

3.6 Electrical-discharge process

This only non-interactive animation shows the different stages of the electrical-discharge machining process.

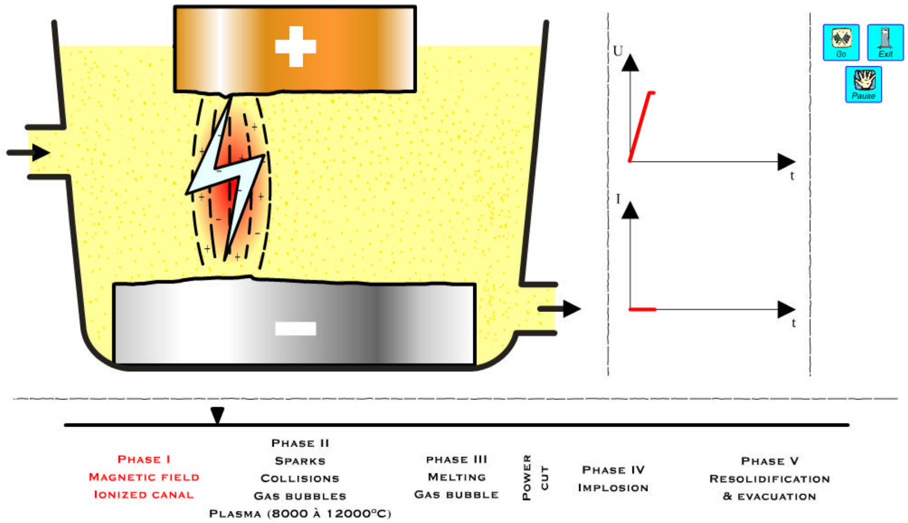

Figure 6 - Screen shot of the electrical-discharge process

4 Project challenges

Several challenges accompany the development of phenomenon animations and simulations:

1. Choosing the right level of detail in the presentation of each process. Finding the right balance between the complexity of the problem and the simplification of the presentation: choice of appropriate number of parameters, and others.
2. Finding appropriate illustrations for each phenomenon. Illustrations must faithfully depict a phenomenon while facilitating its understanding. Animations must convey the essential information with clarity. They must help develop a clear intuition of the process. This requires from the developer of the animation a very detailed understanding of each process, and an adequate understanding of human cognition.
3. Matching student level of knowledge. Animations and simulations must be targeted at a specific audience.
4. Lack of experience in the design and development of pedagogical animations.

A lack of tools specialized for technical animations is an additional difficulty for the development of animations. Some obvious phenomena are often time consuming and difficult to represent. Macromedia Flash facilitates the task but lacks certain key features such as the lack of a 3D modeling device or preset libraries for standard technological items. A minor change in the visual output of animation may require completely re-drawing the underlying figure. Fine tuning animations is thus time-consuming.

Proceedings of the 2002 American Society for Engineering Education Annual Conference &Exposition

Since the teaching evaluation of this lecture gave excellent results, this method should be applied for other technology courses.

5 Animation design

As it has already been said, two basic types of animations were prepared: interactive and illustrative. The interactive ones let the user set different parameters and the resulting animation is then calculated in real-time, whereas the illustrative ones are already preconceived, therefore every viewing is exactly the same.

In order to make animation interactive, it is necessary to program them in a language recognizable by the animation software. An obvious choice was then Action Script, an object-based language included with Macromedia Flash. Each animation contains its own code linking the parameters to the animation itself. The basic structure is almost the same for each animation.

5.1 Interface

The upper-left area is reserved for any custom information about the different parameters (see figure 5). The upper-right area contains the buttons used to perform such basic tasks as screen cleaning (reinitializing), starting the animation, exiting and others. In the lower area, the user can select the values for each parameter using buttons, sliders, check-boxes or input areas. Finally, the animation itself is contained in the upper-center area. To start the animation, the user must set all the parameters and push the start button.

5.2 Programming

In this section we will present a simple example of the animation programming. The animations are programmed with Action-Script, a language similar to C++. The main additions are the built-in methods and properties operating the movie clip's frames, such as *gotoAndStop()* or *_currentframe* attached to the *MovieClip* objects. In object-based language terminology, an object is a basic element of a program such as a rectangle or a movie clip; an object's property is an intrinsic attribute such as its width or height; an object's method is a function attached to the object such as double the area or move to the left. The programming consists in creating and manipulating different objects through their properties and methods. A simple example of a code written in the Action Script language is shown below (the reader should first read the description of the animation "Machine-tool rigidity" in order to understand its functionality).

```
// Memorizing the objects' initial
position
Fix1Init=_parent.Fix1._y;
Fix2Init=_parent.Fix2._y;
PieceInit=_parent.Piece._y;
OutilInit=_y;

// Function defining the resulting   //The function depends
part's width                         only on the axial
function calculerTaille(x) {         position x
                                     //Calculating the
                                     actual width
```

```
Taille_= 52-_root.force +
_root.rigid4 + 38 * ((_root.rigid2
* (x / 150) * (x / 150) +
_root.rigid1 * ((150 - x) / 150) *
((150 - x) / 150) + (((150 - x) *
(150 - x) * x * x) / (3 *
_root.rigid3))));                   //Command returning the
return Taille_;                     resulting value
}

// Functions defining the position
of the machine's supports
function calculerFix1(x) {
Fix1_loc = 10 * _root.rigid1 *
((150 - x) / 150) * ((150 - x) /
150);
return Fix1_loc;
}

function calculerFix2(x) {
Fix2_loc = 10 * _root.rigid2 * (x
/ 150) * (x / 150);
return Fix2_loc;
}
```

In this particular program the only variable is the axial position *x*. Hence depending on the three parameters (the machine's supports rigidities *_root.rigid1* and *_root.rigid2* and the part's rigidity *_root.rigid3*) which for each particular experience are constants, the part's resulting width and the machine's supports' positions will be calculated for each position *x* while the experience is animated.

5.3 Tweening

In Corel Corporation's words [3], "*tweening lets you animate static objects in a movie by changing their position and appearance at specific frames in their life spans.*" Taking as an example the tool from the Figure 8, we can set for this object a specific path along which it will be moved from one end to another. This is exactly what is done in this animation, only that at each frame, the tool's position *x* is checked and all the necessary attributes are calculated in real-time. Depending on the software used, the tweening function is more or less flexible and powerful. As it is said in Corel Corporation's definition, it would be also possible to modify the tool's shape during its movement, but in this case that wasn't necessary. The figure 8 illustrates the tool and its tweening path.

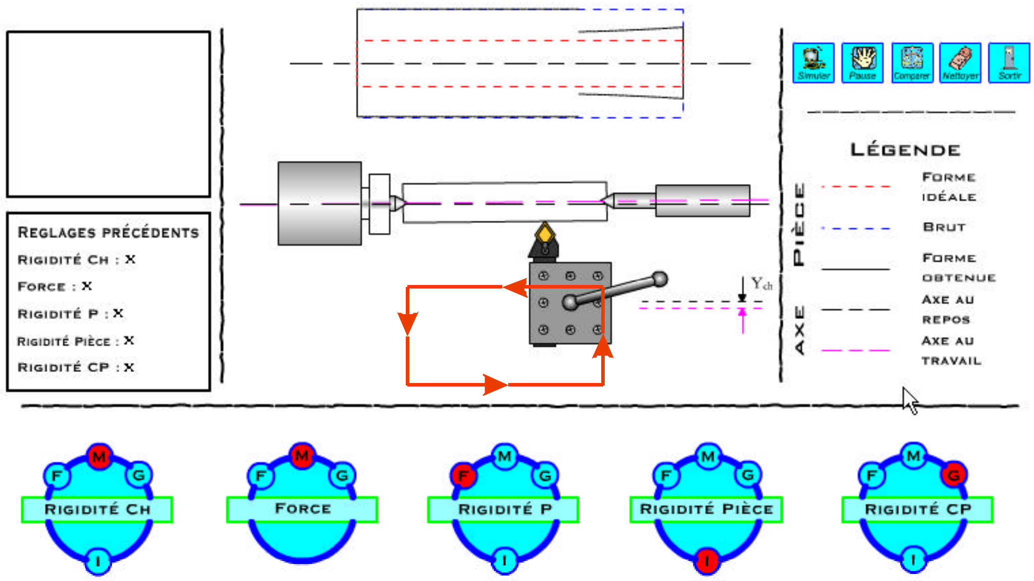

Figure 7 - Object tweening

5.4 Buttons

Three basic object (in Macromedia Flash called symbols) types are recognized by Macromedia Flash: the movie clips, buttons, and graphics. Graphics are very rigid and rarely used (actually they can be completely omitted without any harm). The buttons on the other hand are the key to interaction and most of the animation's commands are done via them. In the animation shown on figure 5, there are as much as twenty four buttons. Nineteen of them let the user set all of the parameters whereas the remaining six ensure the animation's global commands such as described before.

6 Conclusions

Six animations were successfully developed in a short time frame and with a limited budget.

The animations developed in this project were greatly appreciated by students who found them helpful in understanding the material, remembering, and so on. For some students, animations help convey the intuition behind a phenomena. They present the process without using mathematical equations. This characteristic might be particularly important for continuing education and industry training.

In order to assess the impacts of the animations on the understanding of the presented problems, a quick survey with the 27 students following the lecture has been made. The survey question was stated as follows: "In what degree from 1 (not helpful at all) to 5 (very helpful), the animations did help in the understanding of the theory". The average result obtained was 4.6, which indicates that the animations are very helpful in the understanding of the technological problems.

The main challenges lie in coming up with the appropriate representations for each phenomenon. With future improvement of software libraries and tools, animations and simulations should become easier to develop.

Acknowledgement

The work presented in this paper was conducted under the support of Center of Teaching and Learning of École Polytechnique de Montréal, which the authors greatly appreciate.

Bibliographic information

[1] Kadiyala, M., Crynes, B. L., "A Review of Literature on Effectiveness of Use of Information Technology in Education" Journal of Engineering Education, April 2000, pp. 177-189

[2] "Skill Dynamics," A Review of the Litterature Investigating the Learning Benefit of Education Presented in a Multimedia Individualized Format, University of Oklahoma, 1992.

[3] Corel R.A.V.E. 1.0 Help. Corel Corporation.

[4] Colin Moock, "ActionScript – The Definitive Guide." O'Reilly & Associates Inc.

Biographical Information

MAREK BALAZINSKI is a specialist in the area of manufacturing, fuzzy logic and genetic algorithms. He has obtained his bachelor, master and doctoral degrees from the Technical University of Cracow in Poland. Dr Balazinski is currently professor and head of the Manufacturing Section in the Mechanical Engineering Department at École Polytechnique de Montréal

ALEKSANDER PRZYBYLO is a student of Mechanical Engineering Department at École Polytechnique de Montréal. He's been working with Dr. Balazinski for two four months internship.