Journal for Geometry and Graphics
Volume 2 (1998), No. 1, 101–108

A Manual to Teach Computer Graphics by JAVA

Masahiro Takahashi,[1] Hisashi Sato,[2], Kunio Kondo,[1], Shizuo Shimada[3]

[1] *Dept. of Information and Computer Sciences, Saitama University*
255 Shimookubo, Urawa-shi, Saitama, Japan
email: masa@ke.ics.saitama-u.ac.jp, kondo@ke.ics.saitama-u.ac.jp

[2] *Course of Information and Computer Sciences, Kanagawa Institute of Technology*
1030 Shimo-Hagino, Atugi, Kanagawa, Japan
email: sato@ic.kanagawa-it.ac.jp

[3] *Polish-Japanese Institute of Computer Techniques*
ul. Koszykowa 86, 02-008 Warszawa, Poland
email: shimada@pjwstk.waw.pl

Abstract

Techniques of computer graphics are used in various fields; education, business, science and so on. It is difficult to edit a conventional manual about the computer graphics related with texts and pictures. We developed a system to make students understand the complicated concepts of computer graphics through a screen aided by the manual. The computer network is expected to communicate with many distributed people by texts, pictures, sounds, etc. through the World Wide Web (WWW) [2]. JAVA [5] is one of such utilities to help the virtual machine environment on WWW that can manage computer network accessibility. In order to teach computer graphics at the department of computer sciences, a local network is applied as similar as Internet.

Key Words: Computer Graphics, CAI, WWW, Remote Education

1 Introduction

Computer graphics is applied to various fields. Many people are willing to study and to become experts of computer graphics by identified examination. Education of computer graphics is classified into three categories [3, 4];

1. Programming
   A method of usual programming is applied to compose some graphical results. Any of computer languages, such as BASIC, PASCAL, C etc. are taught.

ISSN 1433-8157/$ 2.50

2. Ready programming
   A teacher prepares some applications for computer graphics. Original computer programming is hidden to students, and they have only to study how to prepare commands and data for figures.
3. Semi programming
   In the class of programming, a teacher supports some graphics libraries.

A manual to teach computer graphics is usually edited by poor texts and less pictures so that students feel difficult to understand. We were ever teaching computer graphics by theory and algorithms using written manuals. Students must study the theories and then try to practice by their programming, on which they have to pay a lot of time and efforts. We developed a computer assisted instruction system (CAI) on a network to aid each independent student to study computer graphics along with his personal understanding steps.

2 Purpose

Our CAI system aims to teach about 50 students in the class of computer graphics at every terminal connected to the local network. The concepts to construct the CAI system are as follows:

- Easy understandable manual.
  It supports more conventional media than usual reference books and pictures.
- Simultaneous and independent usage by a lot of people.
  The manual can be used as a common text in the class as well as a personal aid when a student wants to study alone.
- Hardware independency through the network.
  Various computer systems can be connected to the network. The teaching aid software must be therefore independent from any environment.
- Open publicity of pictures among students.
  Homeworks of students can be widely demonstrated to all other students.

3 Systems design of teaching computer graphics

3.1 Systems environment

We constructed our CAI system on the network shown in Fig. 1. Each student corresponds to clients in the figure. A teacher has a server with a database and the client-server system. Our system is able to serve not only students but also outside public clients through WWW (Fig. 2).

The database has HTML (Hyper Text Markup Language) data and JAVA applet, a small application software supplied by Sun Microsystems.

JAVA will be more available in the world rather than using our own programming tool. It was chosen among other utilities for the following reasons:

- Object oriented language
- Similar grammatical manner as C and C++
- Simple syntactic structure

In the Japanese manuals, *Kanji*, Chinese symbolic characters, play very important functions along with alphabetical letters. As *Kanji* is a sort of graphics symbols, the teaching

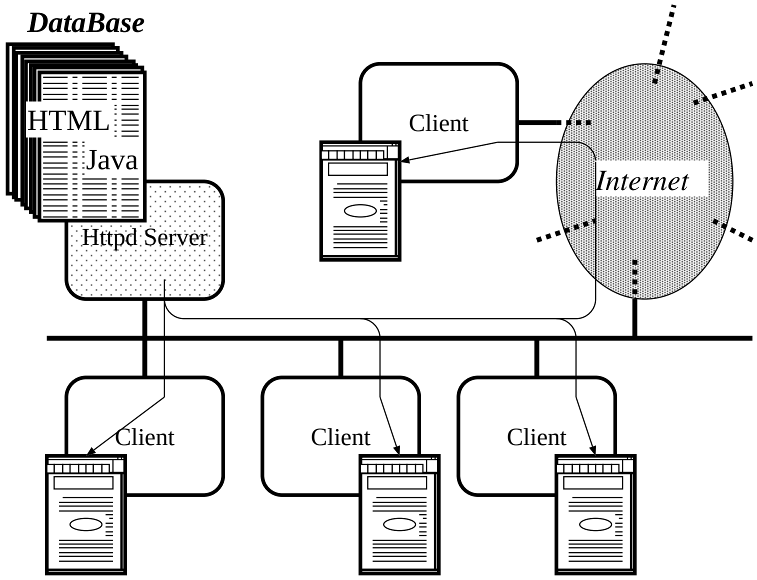

Figure 1: Network environments

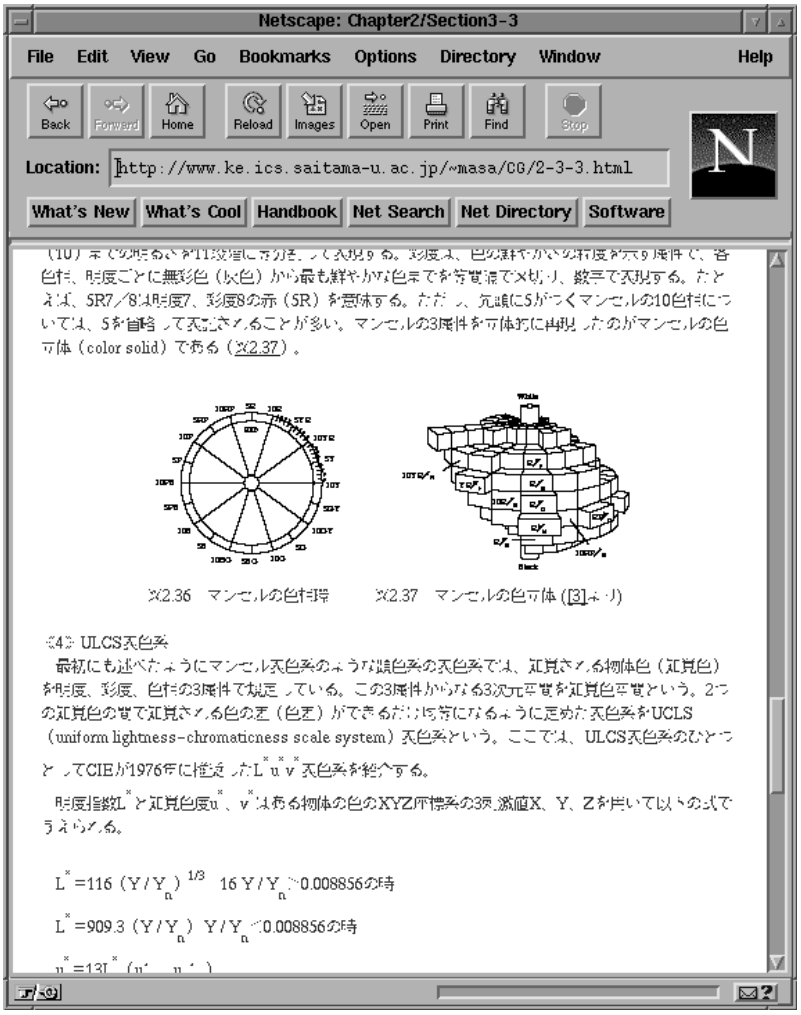

Figure 2: Computer graphics text on WWW system

manual should be edited graphically. This is the reason why JAVA environment is applied in our CAI system.

3.2 Examples of applet

Applet is the name of a unit of applications supported under the control of the whole JAVA system. Some examples of graphical topics are explained.

1. Bézier curve applet:
   In order to generate a smooth curve (Fig. 3), a Bézier curve of 3rd order is the basic tool. A user can study how the curve is generated by given 4 positions on the interactive screen.
   This program, the applet, consists of three operations:
   - init()
   - mouseDown()
   - paint()

   The program list is shown is Table 1, however, the programming is hidden to the user and he only clicks the four positions on the screen, and then the last click activates the drawing of a smooth curve.
2. Affine transformation applet:

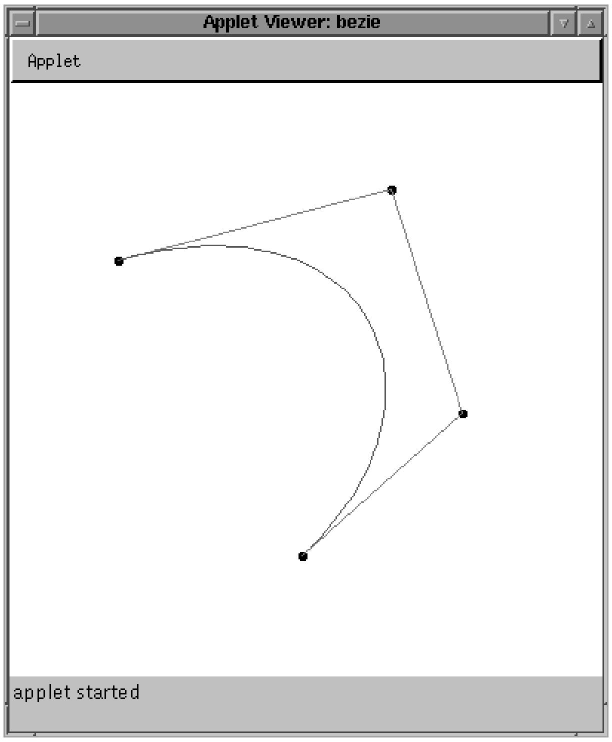

Figure 3: Bézier curve applet

Fig. 4 exemplifies the affine transformation of a plane figure.
Affine transformation is a matrix operation which causes a deformation and a movement of a figure. A student should study the theory of the transformation on the same screen, and he specifies the matrix parameters in the corresponding columns.

```
import java.applet.Applet;
import java.awt.*;
public class bezie extends Applet {
 Point[] p = new Point[4];
 int   n = 0;
 int ndiv = 50;
 public void paint(Graphics g) {
  int i;
  double u;
  double[] w = new double[4];
  Point old,current;
  g.setColor(Color.black);
  for(i=0;i<n;i++) g.fillOval(p[i].x-3 ,p[i].y-3 ,7,7);
  g.setColor(Color.green);
  for(i=1;i<n;i++) g.drawLine(p[i-1].x,p[i-1].y, p[i].x,p[i].y);
  if (n==4) {
   g.setColor(Color.red);
   old = new Point(p[0].x,p[0].y);
   for(i=1;i<ndiv;i++){
    u = (double)i /(double)ndiv;
    w[0] =     (1-u)* (1-u)* (1-u);
    w[1] = 3 *    u * (1-u)* (1-u);
    w[2] = 3 *    u *    u * (1-u);
    w[3] =        u *    u *    u ;
    current = new Point((int)(w[0]*(double)p[0].x+ w[1]*(double)p[1].x+
                              w[2]*(double)p[2].x+ w[3]*(double)p[3].x),
                        (int)(w[0]*(double)p[0].y+ w[1]*(double)p[1].y+
                              w[2]*(double)p[2].y+ w[3]*(double)p[3].y));
    g.drawLine(old.x,old.y, current.x,current.y);
    old = new Point(current.x, current.y);
   }
  }
 }
 public void init() {
  setBackground(Color.white);
  resize(400,400);
 }
 public boolean mouseDown(Event evt, int x, int y){
  if(n < 4) p[n++] = new Point(x,y);
  else      n=0;
  repaint();
  return true;
 }
}
```

Table 1: Program list of a Bézier curve (JDK1.0.x style)

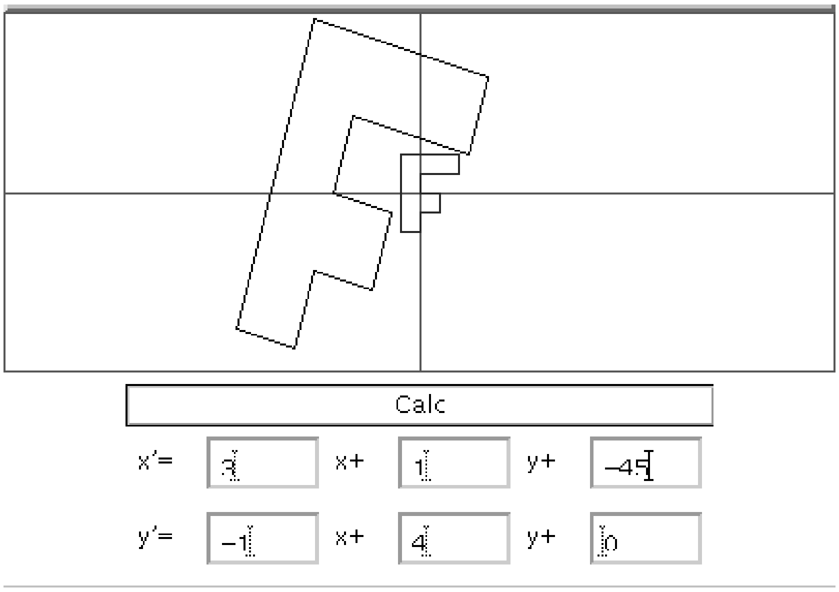

Figure 4: Affine transformation after execution

After completing the input data and clicking the "calc" button, the transformed figure is shown with the original image as shown in Fig. 4.

3. IFS applet:

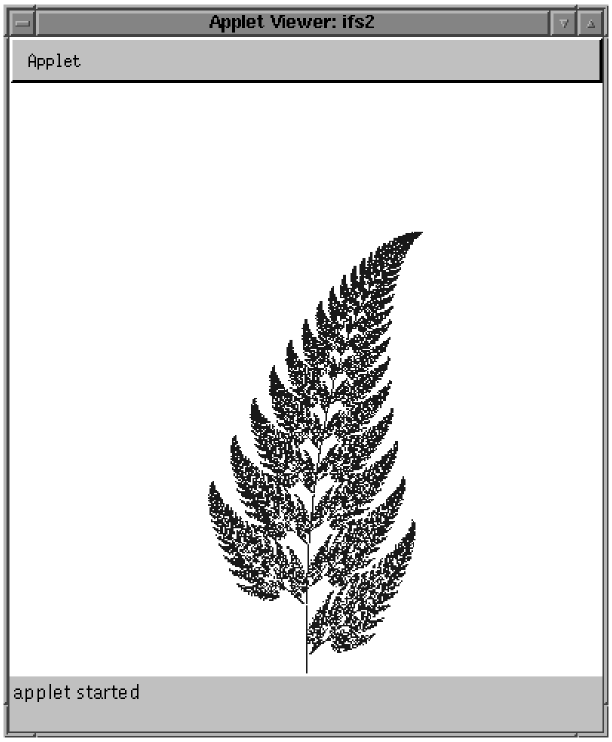

Figure 5: IFS applet

Fig. 5 shows an example of IFS (Iterated Function System) to generate a fractal figure which simulates a leaf of fern. Students will be interested to watch the graphical output step by step on the screen.

4. Three dimensional shapes projection applet:

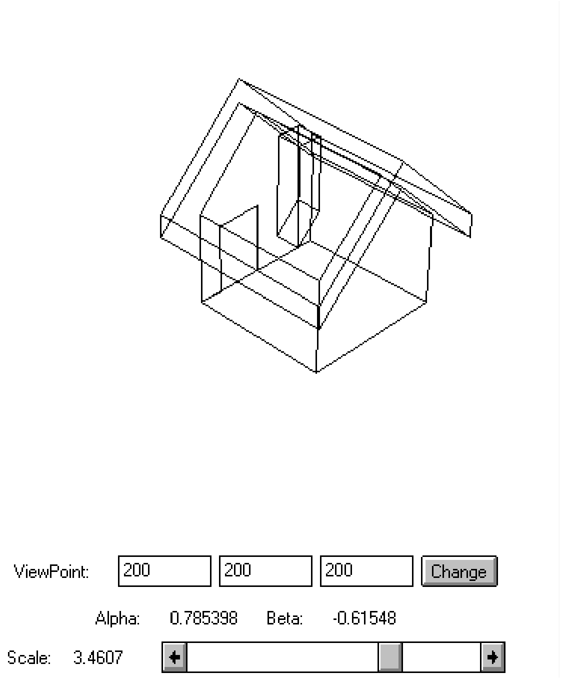

Figure 6: 3D shapes projection applet

Fig. 6 shows the projection of a simple 3D object. We can express only 2D information on WWW contents and textbook. Students can try to change the view point by themselves. It is a typical subject which is hard to explain in a usual text.

5. Primary animation applet:

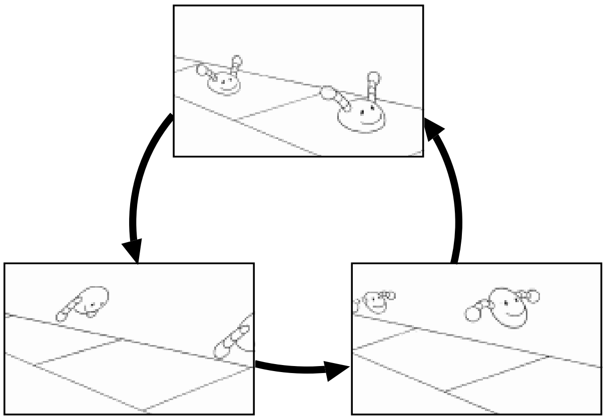

Figure 7: Primary animation

Fig. 7 shows the primary animation applet. Usual WWW systems have a weak point which is showing animations in a browser window. We made teaching materials on animation by the use of the JAVA animation applet. It is difficult to explain animations in usual textbooks, because a textbook is static.

3.3 Discussions about examples

Teaching computer graphics should be designed to work interactively and dynamically, because to watch the creation of an image step by step is more important than to look at a statically complete figure.

Questions and problems raised by students were as follows:

- Algorithms are difficult to understand from the tutorial explanation in the class.
- Three dimensional procedures are requested as an extension of two dimensional figures.
- Motion picture techniques are requested as the combination of image transformation.

4 Conclusion

A teaching manual was edited for computer graphics under the interactive environment connected to the local network which is also connected to WWW.

1. Accessibility of HTML (Hyper Text Markup Language) depends on the browsing on the machine dependent terminals and browsers.
2. As far as the present specifications of JAVA, processing of plane figures gives a reasonable result for teaching.
3. Our teaching manual is based on the text [1] written in Japanese language, for which the cooperative text and graphics are adapted for the use JAVA.

Acknowledgment

The Computer Graphics Arts Society (CG-ARTS) is deeply acknowledged.

References

[1] *Basic textbook for CG editing group,* Technical parts, CG-ARTS (1995).

[2] T. Berners-Lee, R. Cailliau, H.F. Nielsen, A. Secret: *The World-Wide Web*, ACM Comput. Surv. Aug. 1994, 76–82.

[3] K. Kondo: *The Integration of Computer Aided Visual Communication and Visual Thinking in Computer Science Education.* China-Japan Joint International Conference on Graphics Education (1995).

[4] K. Suzuki, K. Yoshida, K. Kajiyama: *Restructuring of Graphic Science Engineering Design and Drafting Courses at Japanese Universities – Based on Implementation of CG, CAD.* China-Japan Joint International Conference on Graphics Education (1993).

[5] Sun Microsystems Inc.: *The Java™ Language Specification*, March 1995.

Received November 26, 1996; final form June 13, 1998