Using *processing* in an introductory computer graphics course

Jordi Linares
Jordi Santonja
Pau Miro
David Cuesta
Background

- A one-semester computer graphics subject: 45 hours, 15 for laboratory work
- The only CG subject for the majority of our students
- Our previous approach for laboratory sessions: OpenGL + GLUT with C
- **Problem**: It’s difficult to provide a comprehensive CG experience in so little time
- **Challenge**: is it possible to work with a bit of everything (2D, 3D, animation, interaction...) in just 15 hours?
- **Processing** appeared as an attractive alternative for our laboratory sessions
What is *processing*?

- *Processing* is a Java dialect, a graphics library and a development environment.
- Specifically designed for generating visual art, animation and computer graphics applications.
- Developed as a bridge towards CG programming for artists and designers (procedural animation & modeling).
- *Processing* is free software and multiplatform (Mac OS, Linux, Windows etc.).
- Initially developed in 2001 at the MIT by Casey Reas and Ben Fry (Aesthetics and Computation Group).
- [http://processing.org](http://processing.org)
Why *processing*?

- Very, very easy to learn... first sketches in just a few minutes
- It has its own IDE (very easy)
- Library inspired in OpenGL and PostScript
- It allows video acquisition and generation, image processing, computer vision, AR, sound etc.
- Scalable. A processing program can be easily migrated into a pure Java application (and use Eclipse or Netbeans)
- Tool of choice for hundreds of artists, designers and visual arts courses
The **processing** language

- **processing code** (.pde)
- **Java code** (.java)
- **bytecode** (.class)
- **Preprocessor** (processing)
- **Compiler** (javac)
- **Execution**

**JVM** (Java Virtual Machine)

Using **processing** in an introductory computer graphics course
The *processing* language

- Processing programming language allows 3 different approaches:
  - **Basic mode**
    - Just some sentences without being part of any function
    - Very similar to an interpreter
  - **Procedural mode**
    - The C-like way of doing the things
  - **Object-Oriented mode**
    - Full Java programming
The *processing* language

- Basic mode

```java
// Example of just some sentences in processing
stroke(255,0,0);
for (int i=0; i<=10; i++)
    ellipse(i*10, i*10, 10, 10);
```
The *processing* language

- Basic mode

```java
// Let's draw an ellipse,
// in the middle of the window
// and rotated 45°
size(200, 200); // Window size

// Transformation matrix
translate(width/2, height/2);
rotate(PI/4.0);

// We draw the ellipse
// 0,0 centered
ellipse(0,0,100,200);
```
The **processing** language

- Procedural mode

```java
// If own functions or classes are defined, an
// implementation for setup() must be defined
void setup()
{
    size(100, 100); // Screen size
    noStroke();
}
void draw() // Called each frame
{
    drawRandomCircle();
}
void drawRandomCircle()
{
    int r = int(random(50));
    // Fillcolor
    fill(random(255), random(255), random(255));
    // Circles with random circles (position, radius, color, sizes)
    ellipse(random(100), random(100), r, r);
}
```
The *processing* language

- The basic mode is ideal to show the students the different possibilities of the library and some important CG concepts (i.e. composition of geometric transformations)
- The procedural mode is ideal for students without a previous Object-Oriented background (C-like)
- Classes and full Object-Oriented features can be used with students with the appropriate background
- Students can evolve their code from one mode to the next one, always reusing their software
- Finally, a pure Java application can be generated
The *processing* language

Using *processing* in an introductory computer graphics course

![Processing code example](image-url)
The **processing PDE**

- A Java application
- Linux, Mac OS X and Windows
- Exports directly to:
  - Linux, Mac, Windows ready-to-run apps.
  - Java Applet for any browser !!

Using **processing** in an introductory computer graphics course
The processing library

- Processing provides 4 different rendering modes: JAVA2D, P2D, P3D and OPENGL

- The 4 modes share almost the same set of primitives
The *processing* library

- Processing primitives allows strokes of different weights and colors, points, lines, ellipses, Catmull-rom splines, Béziers, rectangles, arcs, quadrilaterals and shapes.
- Shapes are similar to those available in OpenGL through the use of `glBegin()` / `glEnd()`
- A very comfortable set of image and image processing functions
- 2D and 3D transformation matrices with matrix stack (every drawing primitive is effected)
- 3D possibilities are similar to those offered in OpenGL
Animation and interaction

• The `draw()` function is launched in a separate thread and continuously executed at a specified framerate, with default double-buffering.

• Interaction is managed through system variables or event handlers.
Animation and interaction

```java
void setup()
{
    size(200, 200, P3D);
    fill(255);
}

void draw()
{
    // Let's rotate at a
    // constant framerate
    // (by default 60 hz.)
    translate(width/2, height/2);
    rotateX(frameCount*PI/60.0);
    rotateY(frameCount*PI/60.0);

    background(0);
    box(75, 75, 75);
}
```
Animation and interaction

Using processing in an introductory computer graphics course
More possibilities

- Video generation is straightforward
- PDF and SVG formats
- With additional libraries:
  - Video acquisition, computer vision, sound generation and synchronization, GUI, AR etc.
- Mobile applications (http://mobile.processing.org)
- Hardware projects (http://hardware.processing.org)

- All these possibilities are very attractive when teaching CG!
Our previous approach

- OpenGL + GLUT in C
- 15 laboratory hours in 3 parts/objectives:
  - An overview of 2D-oriented primitives
  - An introduction to OpenGL and GLUT. The main objective was to develop a 2D vectorial drawing tool
  - A general vision to 3D possibilities, where the students had to develop a 3D object browser and a little 3D modeler based on circular extrusion. No lighting or texturing were covered.
Our previous approach

• **Problem:**
  • We spent too much time introducing the programming framework, the GLUT library and the different initial templates

• **Consequence:**
  • There was no time to cover other aspects we consider essential:
    • Animation, image processing, more advanced interaction and additional 3D topics
Using *processing*

- We prepared a series of seven laboratory sessions:
  - An introduction to processing and its IDE
  - 2D primitives
  - Images and basic image processing techniques
  - 2D transformations
  - Animation (gravitational examples, 2D planetary system)
  - Interaction (‘lunar landing’ game)
  - 3D: basic concepts, modeling techniques, lighting and texturing

[http://processing.tk](http://processing.tk)
Using *processing* in an introductory computer graphics course
Using processing

• Advantages:

  • Students are able to type code and explore graphics primitives from the word go
  • More topics are covered (no initial templates, no time for complex frameworks)
  • All student’s work is reusable
  • Processing is inspired in OpenGL and PostScript (good bases to migrate towards other frameworks)
  • 2D and 3D covered within an homogeneous set of primitives
Using processing

• Advantages:
  • Interaction and animation are easy to learn and explore (developing a game, more motivation)
  • 3D topics like texturing and lighting can be introduced in an introductory subject
  • No object-oriented background required
  • Multiplatform and free software
  • Applications can also be exported as applets for any web browser
  • We can use it in mobile devices
  • More: video, computer vision, image processing...
Using *processing*

- More advantages !!
  - Perfect tool when introducing theoretical concepts, e.g. geometric transformations, using processing like almost an interpreter
  - Processing is a powerful prototyping tool
  - Researching in some areas can be facilitated thanks to its features
Conclusion

- More in less time (and it’s no just a teaching tool)
- Students consider processing more attractive => more motivated
- Perfect for introductory CG subjects
Using *processing* in an introductory computer graphics course

Jordi Linares
Jordi Santonja
Pau Miro
David Cuesta

Polytechnical University of Valencia
Campus of Alcoi
SPAIN