IP-LAB: A Tool to Teach Image Processing Programming in Java

- Challenge in Swiss Federal Institute of Technology Lausanne -

IP-LAB : Java による画像処理プログラミング教育ツール

− スイス連邦工科大学ローザンヌ校における実施例−

○Akira Hirabayashi※1　Daniel SAGE※2　Michael UNSER※2

○平林 晃※1　ダニエル サージュ※2　ミカエル ウンサー※2

Keyword: Image processing programming, Java, Educational tool

キーワード：画像処理プログラミング，Java，教育ツール

1. Introduction

Image processing is getting more and more important because of the widespread use of digital imaging devices, such as digital cameras or digital videos. While image processing is a very practical discipline, it is perceived as being rather theoretical. It is actually a subject that involves a rigorous, mathematical treatment. Hence, what is important in image processing education is that instructors make students not only understand such mathematical formulas but also become to be able to use them. To this end, computer laboratories accompanied by lectures are very effective for image processing education. In order to make the best use of them, computer systems which will be used by students play a very important role. IP-Lab is a system in Java that was developed by the second and the third authors of the present paper in the Swiss Federal Institute of Technology Lausanne (EPFL) to offer an environment where the students could implement the algorithms literally as they are seen in the course.

2. Conventional Tools

There are a number of tools for image processing education. It is desirable for tools to be ready to program algorithms literally in the textbook as mentioned above. Further, tools should be freeware. Low-level languages offer the advantage of computational speed, but tend not to be good at basic operations such as reading files, data types, accessing pixels, and displaying images. High-level languages offer a rich functionality for those operations, but tend to hide many important aspects of imaging routines. They are also commercialized and expensive. In order to compromise the advantages of them, IP-Lab was developed.

3. IP-LAB

IP-Lab [1] is an environment based on Java, ImageJ, and originally developed interface layer called ImageAccess.

Java

The programming language chosen in order to develop the system is Java. The main arguments of this choice are: 1) Java is free; 2) Java is platform neutral, hence well adapted to the diversity of the student community; 3) Java is robust with a good handling of errors and garbage collection; 4) Java is syntactically close to C and easy to learn if we provide examples and templates for the methods. 5) Java is reasonably fast.

In addition to them, students are attracted by Java, a modern and fashionable language that plays a major role on the Web.

ImageJ

ImageJ [2] is one of the most comprehensive image processing freeware available. Its graphical user interface provides convivial interaction with the full functionality of an image processing application. It can run on any platform with a Java Virtual Machine (Mac, Windows, and various flavors of Unix). Applications and sources are freely available.

ImageJ has an open architecture that allows extensibility by the addition of Java plug-ins. We take advantage of this functionality for adding our educational modules. Java also provides a mechanism for loading the plug-ins dynamically without having to restart the application after each modification of the code; this functionality offers a fast and comfortable way to edit-compile-execute a program.

ImageAccess: The Interface Layer

ImageAccess is a "student-friendly" software layer that has been developed by the second and the third authors; it simplifies and robustifies the access to pixel data without having to worry about technicalities and the interfacing with ImageJ.

ImageAccess was designed by applying two well-known principles of software development.

▲ Abstraction. For the user, an image is simply an instance of the ImageAccess class. The pixel data is always retrieved and stored in "double" format, independently of the underlying ImageJ image type. In this way, students do not have to worry about rounding, truncation, or conversion of pixel data. Moreover, pixel data can be accessed "anywhere" through the use of consistent mirror symmetric boundary conditions. The aim of applying abstraction is to let the source code express the original algorithm more clearly. The full documentation of the class is available at [3].

▲ Encapsulation. The fact of working with Image Access objects prevents the students from having to worry about implementation details. The typical way to program is to retrieve an image block by using a method that begins with get...(). The block is processed and the result is written in the image using a put...() method. The block can be a single pixel, a row, a column, a 3 × 3 or a 5 × 5 neighborhood window.

Listing 1. Example of an implementation

```
public ImageAccess
  filter2D_NonSeparable(ImageAccess input)
{
  int nx = input.getWidth();
  int ny = input.getHeight();
  ImageAccess output = new ImageAccess(nx, ny);
  double block[][] = new double[3][3];
  double value = 0.0;
  for (int x=0; x<nx; x++) {
    for (int y=0; y<ny; y++) {
      input.getNeighborhood(x, y, block);
      value = (block[2][0] - block[0][0]
             + block[2][1] - block[0][1]
             + block[2][2] - block[0][2]) / 6.0;
      output.putPixel(x, y, value);
    }
  }
  return output;
}
```

Sample Source Code

The students who participate in the laboratories do not necessarily know Java. Hence, an example of a Java method that does an operation similar to the assignment is always provided to the students.

Listing 1 is an example provided to the students. This is a digital filter that detects vertical edges. We can see that the code is relatively straightforward; it is essentially a literal translation of the textbook versions of the algorithm. This owes much on the interface layer.

4. Conclusion

IP-Lab has been used for a course in EPFL from 2000 (Fig.1). Even though students do not know Java, "learn by example" strategy has been providing good success. The first author is going to use IP-Lab in Yamaguchi University in the course for graduate students from 2005. Because of the ease and the robustness of the system, it will not be so difficult for the author to have good achievements.

(a-hira@yamaguchi-u.ac.jp, daniel.sage@epfl.ch)

Fig.1 Scene in the computer laboratory in EPFL

References

[1] D. Sage, M. Unser, "Teaching image-processing programming in Java," IEEE Signal Processing Magazine, vol. 20, no. 6, pp. 43-52, November 2003.

[2] W. Rasband, ImageJ, National Institutes of Health, Bethesda, MD., http://rsb.info.nih.gov/ij/.

[3] http://bigwww.epfl.ch/teaching/iplabsite/