

A Pedagogical Toolkit for Image Processing Visualization and Experimentation

Eugene AGEENKO and Gaetano LA RUSSA

Department of Computer Science, University of Joensuu,
Box 111, FIN-80101 Joensuu, Finland
{ageenko|larussa}@cs.joensuu.fi

ABSTRACT

The widespread use of imaging technologies and devices has imposed an increased need for training specialists who are expected to be proficient in these new technologies. Many educational institutions are organizing themselves to meet these demands by offering courses in image processing (IP). The IP courses have to deal with the concretization of various concepts and ideas that most of the times are of abstract scientific nature. Imagination and interpretative skill as well as image processing visualization skills and extended laboratory hours are a must for the full understanding of many concepts involved in IP.

In this paper the authors present a framework for visualization, demonstration and experimentation with the image processing operations and algorithms. The framework is developed on the fundamental concept of human cognition process and is capable of demonstrating, in an intuitive visual form, all main connections that exists between an algorithm and its visual data. The framework is oriented to aid both teachers and students alike. The framework software (i.e. the pedagogical toolkit), built on Java language and provided under GPL license, has proved to be valuable educational software capable of demonstrating basic as well as advanced concepts of IP.

Keywords: image processing, education, research, teaching, requirements, improviser.

INTRODUCTION

In our ever-growing Information Society, the relevancy and importance of visual tools as well of image related expertise is becoming undisputable. The fact that Image Processing courses take more space in Universities curricula [6] show that an advanced technological society can not cope with success with many aspects of everyday life without properly formed IP experts. Many big companies have endeavored themselves in the production of specialized and powerful tools for image handling and editing (such as MATLAB, Khoros, etc.) [3][2].

Most of the available commercial tools have very complicated (even though rich) interfaces, full of functions and features. But none of them allow students to reason and explore the motivations for image variations. After all, the scope of the commercial product interfaces is to teach students (or *learners* in general) to properly use the software and not to explain the concepts and ideas of IP. It was therefore clear that an intellectual gap had to be filled if wanting that students of IP would understand and easily become experts in the field.

In our case we did not want to provide a copy of something already existing in the market, especially because our interest was to interactively teach IP principles while having students

understanding the subjects according own paces [4]. We therefore focused on a visualization toolkit that allows teachers and students to visualize and represent the various complex concepts that regulate IP [10].

The authors intend to present in this paper an image processing toolkit capable of demonstrating, in an intuitive visualized form, all main correlations that exists between an algorithm and its visualized data. The toolkit, which is based on a Java platform, has proved to be an efficient pedagogical software capable of representing both basic and advanced concepts of IP [8]. The toolkit is meant for teachers and students of IP. The teachers can convey in a more intuitive way the various complex concepts that are related to the image processing. Students can quickly learn and define the various IP taught topics by associating the correct meaning of variability of parameters in the algorithms. This happens by correlating the variation of parameters to the visual variations of the images.

The impact that the visualization of algorithms has on the understanding of a complex processes (e.g. filtering algorithm) is indeed cognitively high and in general superior to the plain textual description of the concept itself, that often requires a lot of imagination and interpretative skills. Efficient teaching and learning of IP fundamentals is possible if much attention is paid to the visual representation of the algorithms and to the experimental works of the students. Therefore, better pedagogical results can be achieved if the proper visualization tools, for the demonstration of the basic aspects of image processing [11], are made available to teachers.

The advantage of the IP visualization tool on traditional teaching is that it can be used by the students also after the lecture hours. The experimental assignments (laboratories) can be organized in such a way that students verify the course presented knowledge, theory and technologies. During the laboratory hours the students can complement their basic lectures by processing and manipulating the images. Students can also create or define their own image processing tools. Quite important is the fact that all of this is possible within constricted time frames. This direct experimentation can significantly facilitate the learning of the mathematical concepts of image processing [9]. Students can conduct their own experiments with the toolkit and consequently learn-by-doing. In few experimentations they acquire enough knowledge to be able to create personal IP plug-ins.

IP EDUCATION – METTING THE DEMANDS

1. About the cognition process

According to Prof. Skvorcov, the cognition process is an infinite spire. On each temporal cut the cognition spire forms a turn leading from the cognizance (recognition) to knowledge. Even though the cognition is the process of intuition, the cognizance

has three distinctive steps:

- 1.a.i.analyzing the concept,
- 1.a.ii.testing and experimentation with the concept,
- 1.a.iii.concretize the formation of the conceptual representation of the concept and building a system upon it (at first) and the system (at last).

The proposed education framework supports knowledge creation on each phase of cognition via its functionality and features.

Kolb [7] describes the learning as a cycle based on experience and practice (see Figure 1). The learning stems from concrete experimentation that induce reflective observation. This allow abstract conceptualization that need to be concretized in active experimentation, which in turn will produce some sort of new concrete experimentation, hence restarting the learning-cycle.

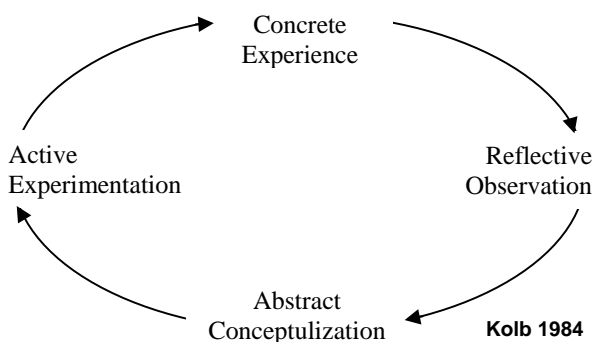


Fig 1. Kolb's learning cycle

2. The main scenarios for teaching IP

The IP tool is a mean for demonstration in class and for student use in various cases (experimentation, verification, etc.). The toolkit can be used by and according the following:

- 2.a.teachers of the IP classes for demonstrating various aspects of image processing and algorithms, and
- 2.b.individual students, for experimentation work including, but not limited to:
 - 2.b.i.self-study,
 - 2.b.ii.home work,
 - 2.b.iii.demonstration of the home work results in the class,
 - 2.b.iv.development of new algorithms for expanding the system.
- 2.c.In the future this feature will be extended to the simultaneous use by collaborative groups of students.

3. The main pedagogical goals for each process

The intention of the designers of the toolkit is to bridge the abstract concepts (which can be in terms of algorithm variables, complex plug-ins, etc.) with a real perception of what the variations of components produce in image editing or filtering. Therefore the two combine features of *teaching tool* (as a mean to describe visually the effects of algorithms on the images) and *learning tool* (as an experimentation tool to verify and concretize the effects of the algorithms on the images) have been integrated in the structure of the toolkit. Recurring similar algorithm applications on different images give the opportunity

to compare and deduct conclusions on the meaning of the applied algorithms. This applies also to the changing of the variables and application of plug-ins. For a sample scenario to use the toolkit, see Fig.2.

4. The pedagogical requirements and constraints for the toolkit operating processes

The toolkit had to be constructed as a flexible and adaptable tool, capable of meeting students' expectations to satisfy the curiosity to experiment. The purpose was to incentive their constructive behavior and stimulate their learning by doing. In addition to this, the tool had to be sophisticated enough to allow the teacher to represent the algorithmic application impact on the images and draw the attention of the students on the abstract concept. Both things have worked out perfectly. Now the toolkit even allows multiple windows for synchronous movement of the processed images in comparison to the originating image. The toolkit has also a zooming feature allowing the detail visualization for easy comparison. The system can also save all processed images and modified plug-ins. The history of their application to images is also recordable for reproduction of similar experiments.

The toolkit has an interface mainly constituted of graphical elements and uses intuitive elements to define its main functions. The design of the interface was conducted taking into account students' needs and generic symbol interpretation patterns.

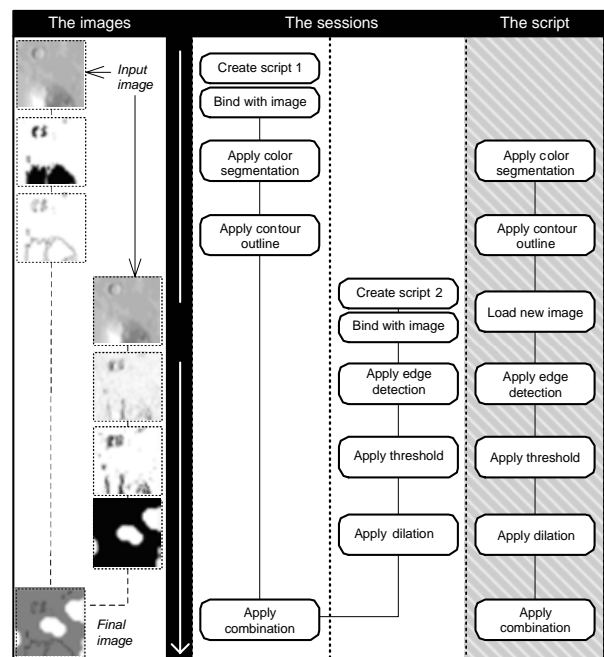


Fig 2. Sample scenario working with the system

5. The key requirements and strategies to facilitate learning

The toolkit, called also Improvizor (IMage PROcessing VIsualiZation and ExpeRimentation) is the result of the wanted combination of three main requirements:

- Visualization
- Demonstration
- Experimentation

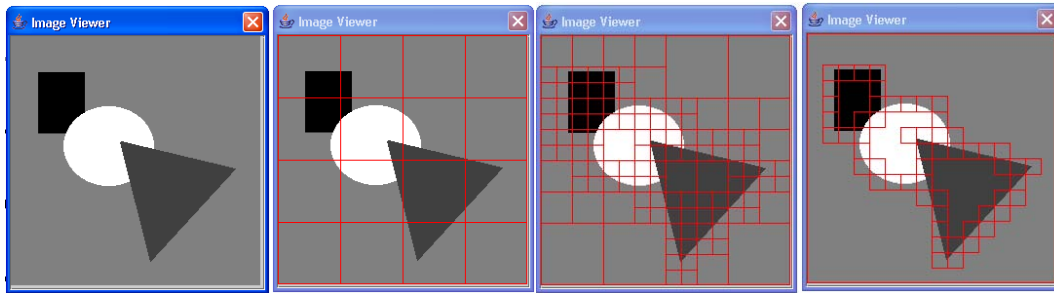


Fig 3. Example of Split-and-merge algorithm animation. Original image, after a first-step, after a split sequence, final result.

The key aspect of the system (its visualization feature) is achieved by providing the visual rendering of the algorithmically processed images to users for performing the visual assessments on the qualitative and quantitative impacts of the algorithms. According to this feature, the toolkit is capable of:

- 5.a.animating the processes, whenever the algorithm implementation supports it
- 5.b.creating a users' logical connection between the rendered images and the applied algorithms
- 5.c.the three fundamental image types (grayscale, binary and color) stored in popular image file formats (png, gif, jpg, bmp, and such) are supported by the system.
- 5.d.the experimentation is performed by studying the visual and numeric effects of the algorithms on the images by varying the algorithms, algorithm parameters, forming and varying the sequence of the applied algorithms and re-applying all of these on various images. The comparison of the results from various image processing leads to the understanding of the image processing (IP) correlations.

These three features are described more in detail in the following sub-chapter.

6. The Visualization feature of the toolkit

The effect of an algorithm is visualized using dual image view system, displaying side-by-side the processing and processed images with possibilities of synchronous panning and zooming. For better assessment and comparison, any images that ever were processed in the project, as well as the animation steps and other images generated by the algorithms may be also visualized (displayed) in own frames. Via toolbar it is possible to follow current situation of the animation of the applied plug-in (see Fig.4)

- 6.a.all the frames can freely move and resized on the desktop.
- 6.b.the frame content (of the images) can be:
 - 6.b.i.(synchronously) panned
 - 6.b.ii.(optionally) zoomed
 - 6.b.iii.(on request) the point of the image focus can be synchronized among the various open views to the same visual location relative to the frame.

Algorithm visualization is performed using animation (one example of split and merge algorithm animation is given in Fig.3):

- 6.c. By animation we understand the sequential display of the images generated by the algorithm during its execution.
- 6.d. These images can be either:
 - 6.d.i.the actual stages of the image processing or only
 - 6.d.ii.the visual representations of the algorithm logic. This behavior is further specified by the algorithm implementation.
- 6.e.the algorithms that support animation implement multi-step architecture so that they can perform one algorithmic step at a time (at single execution). The animation can be happen in the following modes:
 - 6.e.i.“step-by-step” – the manually controlled animation,
 - 6.e.ii.“run” – the algorithm executes in a continuous mode at maximal possible speed (this mode is not targeted for animation, rather for final result),
 - 6.e.iii.“visualize” – the algorithm executes continuously in a step-by-step fashion with specified time delay necessary to observe the intermediate results.
- 6.f.the framework has also capability to pause, resume, and stop multi-step algorithm execution.
- 6.g.in “Run” and “Visualize” modes the multi-step algorithms are capable to pause and restart their execution according to super-steps defined in the algorithm logic.

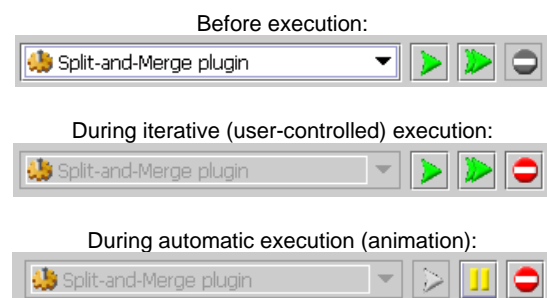


Fig 4. The view on the toolbar of the animated plug-in before, and during execution.

7. The Demonstration feature of the toolkit

Typical patterns for teachers using the system in the class:

- 7.a.demonstrating the aspects of some algorithm, modifying its parameters, studying the visual and

numerical effects of the algorithm on the images by comparison and detailed investigation of the images resulting from algorithm application,

- 7.b.demonstrating the algorithm progression by investigation the algorithm steps and their visual aspects,
- 7.c.demonstrating the sequence of the algorithm(s) application (using the same as in the case of iterative algorithms or various algorithms as in the case of morphological operations).

The typical patterns for the use for the system by the students:

- 7.d.self-studying the aspect of some algorithm,
 - 7.d.i.including algorithm description,
 - 7.d.ii.algorithm behavior
 - 7.d.iii.visual and numerical effect of the algorithm on the image.
- 7.e.performing the home work by:
 - 7.e.i.simulating the execution and comparing the results (possibly using the enlarged view mode),
 - 7.e.ii.searching for a appropriate parameter set or a correct sequence of algorithm that will lead to completion of the assignment.
- 7.f.demonstrating in the class the results of the home-work by
 - 7.f.i.visualizing the result images,
 - 7.f.ii.replaying the sessions.
- 7.g.developing new algorithms using open-source programming interfaces and Java 1.5 or later SDK, which in fact facilitates dual learning of IP and of Java language (which can be offered as a part of OOP class project work)[5].

The collaborative use [1]:

- 7.h.during the collaborative use, a (possibly guided) group of users (likely students) operates on the same project. The leader of the group, which is defined at the beginning of the session or tentatively during the session, is guiding the project development progressing from one step to another.
- 7.i.in between of the guided steps users are able to perform their own private experimentation, which can be exchanged with others if such a user will be elected as the group leader.
- 7.j.the collaborative use must enable the possibility for distant teaching and learning of the IP (in this regard all design has to take into account the multi-user working scenarios).

8. The Experimentation feature of the toolkit

The system allows for experimentation on various semantic levels, which is achieved by:

- 8.a.varying the images to which the selected algorithm will be applied to
- 8.b.varying the algorithms applied for the selected image
- 8.c.varying the algorithm parameters and their effect
- 8.d.varying the sequence and number of the algorithm applications
- 8.e.modifying and reconstructing the algorithm

To achieve the first requirement the system is equipped with special enlarged view operation mode, which operates with largely magnified view of the processing and processed image with editing capability. The magnified view is comprised of rectangular array of rectangles representing image pixels and their value. The dimension for images used in such mode can be limited to reasonable values.

THE VISUAL IP COMPUTER

The system architecture resembles the architecture of the visual computer and has the following components:

- The Visual Processor (a la system “ALU”) – performs image manipulation using algorithms.
- The Algorithm Manager (a la “BIOS”) – stores and manages the algorithms.
- The Image Manager (ala “Memory” and “Memory controller”) – stores and manages the images and their visual representations.
- The Session (ala “Cache memory”) – stores temporary images, that are the results of image processing operations recorded in the session, uses FIFO depletion queue data structure (the images are deleted when leaving the queue). The depletion rule is based on a temporal factor (each session has its own cache)
- The Script (ala “Software”) – scripts representing the sequence of applied algorithms.
- The Executor (ala “Controller”) – controls the sessions execution in the processor according to the script
- The Project Manager (a la “Working Space”) – aggregates images and the scripts.

In Fig.5 is a representation of the execution of the script.

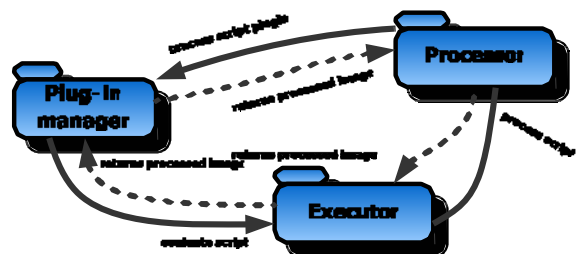


Fig 5. Execution of the script in the processor

The toolkit technical specs in brief are:

- The software has small size and its packaging is contained within few files (preferably into a single file);
- The system is portable among various computer platforms;
- Installation does not require administrative privileges;
- The software has simple user interface with minimum functional buttons and tabs required for the system operations;
- The system uses standardized symbols for visual representation of the system components to the user, and take into consideration non-experts (i.e. novices)

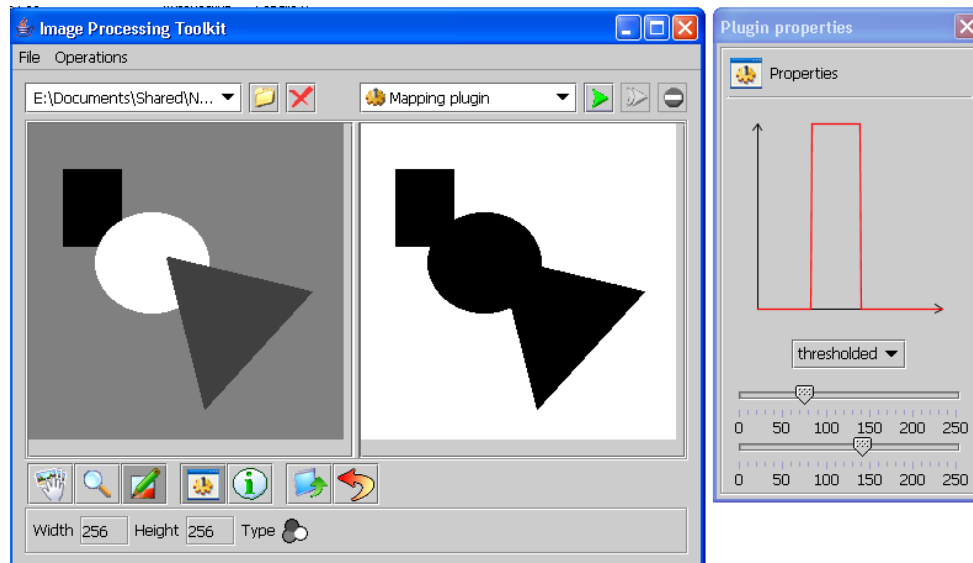


Fig 6. The GUI of the Visual Processor and IP algorithm property pane, illustration of the dual-view feature.

- intuition for the location of its functions and the determination of the correct function parameters;
- The system operates at speeds sufficient for the interactive processes and for visualizing animation for images up to 2 Mpixel (up to 10 sec delay), and provide nearly instant results for processing small images up to 0.25 Mpixel.

USABILITY ISSUES

The typical scenario to work with the system is in conditions where is needed self-studying and understanding the aspect of some IP algorithm, including algorithm description. The toolkit proves to be very good in analyzing the algorithm behavior and obtaining the visual and numerical effect of the algorithm on the images. It is possible to perform the home work by simulating the execution and comparing the results (it is possible to use the zoom view mode), searching for an appropriate parameter set or a correct sequence of algorithm that should lead to the completion of the assignment.

The toolkit can be used for demonstrating in the class the results of the home-work by visualizing the result images and replaying the process sessions. The toolkit require an external source (i.e. software) for the development of new algorithms (for example using open-source programming interfaces and Java 1.5 or later SDK - which in fact facilitates dual learning of IP and of Java language).

The Visualization Goal

The visualization is achieved in the system via the visualization of the algorithms and via the animation of the algorithm, in its steps and logics. The images can be assessed via multiple synchronous views with pan and zoom features. A dual-view visual processor with magnifier glass is integrated in the toolkit. See Fig.6 for an example of a dual view.

The Experimentation Goal

The experimentation occurs on all levels, and specifically by:

- Varying IMAGES
- Varying OPERATIONS
- Varying PARAMETERS

- COMBINATION of OPERATIONS
- CREATING NEW OPERATIONS using SCRIPT PROGRAMMING
- going LOWER LEVEL - MODIFYING ALGORITHMS using Java programming.

The Demonstration Goal

The demonstration features aim at showing and clarifying the various aspects of some algorithm. This happens by modifying the parameters of the algorithm and studying the visual and numerical effects of the algorithm on the images (also by the comparison and the detailed investigation of the images resulting from the algorithm application). In addition, the algorithm progression is demonstrated by investigation of the algorithm steps and its visual aspects. The sequence of the algorithm(s) application are demonstrated and in some cases by using the same algorithm, as in the case of iterative algorithms or as in the case of morphological operations.

CONCLUSIONS

The use of the Java platform makes the toolkit widely accessible (i.e. no restriction to specific main operating systems). The toolkit is also an extendable, open-interface, plug-in based framework with a simple but intuitive user interface. The toolkit works as a concept convener, easing students' understanding of complex ideas in Java programming and image processing fields. It allows students to verify the validity and functionality of the algorithms in the image processing.

The software has already exceeded our IP expectations granting students to quickly learn almost every basic aspects of IP. Students have been capable of becoming rapidly experts in IP via their experimentation and definition of new plug-ins. At the moment, tens of new plug-ins have been created since last year. The toolkit, in addition to its pedagogical aspects, has the advantage of flexible and easy to develop functionalities. The pug-ins can be easily and rapidly developed and various wanted effects of visualizing the image sequence, modifying the operation parameters and visualizing the algorithms are at hand for everyone.

REFERENCES

- [1] Barrows, H.S. et al. 1996. Computer-Supported Problem-Based Learning: A Principled Approach to the Use of Computers in Collaborative Learning, In T. Koschmann (Ed.) CSCL: Theory and Practice of an Emerging Paradigm, Lawrence Erlbaum Associates, Mahwah, New Jersey. pp.83-124.
- [2] G.W. Donohoe and P.F. Valdez, "Teaching digital image processing with Khoros," IEEE Trans. Educ., vol. 39, no. 2, pp. 137-142, 1996.
- [3] S.L. Eddins and M.T. Orchard, "Using MATLAB and C in an image processing lab course," in Proc. IEEE Int. Conf. Image Processing (ICIP'94), Austin, TX, 1994, vol. 1, pp. 515-519.
- [4] R.B. Fisher and K. Koryllos, "Interactive textbooks: Embedding image processing operator demonstrations in text," Int. J. Pattern Recognition and Artificial Intell., vol. 12, no. 8, pp. 1095-1123, 1998.
- [5] Flanagan, D., 1999. 3rd edition, November 1999. Java in a Nutshell: A Desktop Quick Reference (Java Series), 648 pages, O'Reilly & Associates
- [6] Gonzalez, R.C. et al, 2002. Digital image processing, 2nd edition, Addison Wesley.
- [7] Kolb, D.A. (1984) Experiential Learning: Experience as the Source of Learning and Development Prentice-Hall Inc., New Jersey.
- [8] D.A. Lyon, Image Processing in Java. Upper Saddle River, NJ: Prentice-Hall, 1999.
- [9] Naidu, S. et al, 2002. "The Experience of Practitioners with Technology-Enhanced Teaching and Learning", Journal of IEEE Education Technology & Society, Vol. 5 N.1, January 2002, 23-34
- [10] M.W. Powell and D. Goldgof, "Software toolkit for teaching image processing," Int. J. Pattern Recognition and Artificial Intell., vol. 15, no. 5, pp. 833-844, 2001.
- [11] D.S. Sohi and S.S. Devgan, "Application to enhance the teaching and understanding of basic image processing techniques," in Proc. IEEE Southeastcon 2000, Nashville, TN, 2000, pp. 413-416.