Re-inventing image processing course - an effective approach

Zbigniew Mrozek

J. Dietl Malopolska Higher Vocational School in Cracow (Małopolska Wyższa Szkoła Zawodowa w Krakowie), PL 31-532 Kraków, ul. Wincentego Pola 4, tel.: (48-12) 428-24-50

(21nd EAEEIE Annual Conference, Palanga, Lithuania, June 28-30, 2010)

,

Introduction

The way students are taught has been changed in recent years. Easy access of students and university staff to commercial or educational software and modern computer equipment makes teaching of computer graphics and image processing much easier than before [1,4]. But very often students do not understand what is inside of each black box they use. So it was decided, that students have to develop and implement basic image processing software themselves, to gain better understanding of the subject. MATLAB (without toolboxes) was found to be the most convenient computing environment for this task. Students are very impressed with fast and easy results obtained and show great interest in improving software they design and use.

Digital images

Digital images are saved as vectors or pixels. To use vector graphics one needs knowledge of essential mathematics (starting with points and vectors in 2D and 3D space) and algorithms for drawing and manipulation of points, lines, circles and other primitives on computer screen. An advantage of vector graphics is that scaling or rotating the vector image doesn't degrade quality of a picture.

Raster graphics is more suitable for photographs and photo-realistic images than vector graphics. It is also used in industrial image processing systems. Photographic images are saved as an array of pixels (picture elements). Typical resolution of digital camera picture is 10-15 Mega pixels (year 2010). Raster graphics is complementary to vector graphics.

Teaching fundamentals of raster graphics

This paper describes methodology for teaching fundamentals of raster graphics. It was decided that main reason of teaching students is not to support them in effective usage of popular graphic packages (e.g. paint shop or gimp) but rather to understand how it works. It was decided, that to gain better understanding of the subject, students will develop and implement basic image processing software themselves. MATLAB (without toolboxes) was found to be the most convenient computing environment for this task. It was used for computation as it is user friendly software package for high performance computation and visualization [2]. Reliability and powerful graphics makes MATLAB the premier software package for engineers and scientists. It was decided not to use functionality of MATLAB Image Processing Toolbox during this course. Using C language is not recommended as it needs more effort in programming, leaving less time for teaching graphics.

Choosing the test picture

To get repeatable results, all students should use the same picture. The well known lena.tif picture [7] was chosen for processing. Low resolution 256x256 or 512x512 greyscale picture may be imported on-line from internet [5,6] during each lecture or lab.

Picture data in computer memory

The goal of the first lecture is to show how to use simple mathematics (addition, subtraction and multiplication) in graphics processing.

The `[X,map]=imread(filename)` MATLAB command will move graphic data from graphic file to `X` variable. The variable `map` should be empty for b/w image. Student should see the numbers hidden in X variable (matrix 256x256 or bigger). There are two possibilities of variable type: unsigned integers of type `uint8`. Their values are in range `0..255`. Other possibility are float point numbers in `0..1 range`. Let's assume `uint8` numbers are used.

Fig 1. Original image `lena.tif` and its brighter copy

Using division, multiplication, subtraction, roots or powers, one can increase or decrease value of pixels, making picture lighter or darker. Often, result of computation has to be normalised to be within `uint8` limits (0..255). Some results are given on figure 1.

Fig.2 Binarisation (level=96/256) and negative of image

Other topics to be presented on lecture are binarisation, negative of image (fig.2), LUT (look up table) and enhancement of image quality based on values of histograms.

Laboratory work for students – reading and displaying digital images

Lecture gives general knowledge of presented topic but laboratory work is essential to get deep understanding of fundamentals of computer graphics. During lab student has to solve many unpredicted situations. As first lab may be more difficult, students are advised to work in pairs (two students and one computer) – unless some students prefer to work alone. Starting with second meeting – all students should work individually. If needed, they have possibility to discuss any lab problems with their neighbours or in small ad-hoc groups of few students. If problem is still unsolved, an experienced laboratory assistant will give advice, but he should not give the final solution. Some typical problems will be described later.

Reading colour graphic file (instead of b/w) into MATLAB variable may be confusing. If colormap variable `map` is not empty, it means that indexed colours are used. The `map` matrix (256x3 or 128x3) defines RGB colours used in image. More than 16 million colours is available with fullcolor graphics, where each pixel has 3x8 bits depth to describe its RGB colour. To get b/w picture, RGB colours should be recomputed into grey values, taking into account coefficients dependent on human eye sensitivity. Based on NTSC standard, following expression is used to get new matrix `Xbw` of picture data in grey levels:

```
Xbw=
.2989*X(:,:,1)+.587*X(:,:,2)+.114*X(:,:,3);
```

Another problem is to display b/w images in MATLAB. The default is usage of `jet` colormap. It starts with blue, passes through cyan, yellow, and orange to red. As result, b/w images are not displayed in levels of gray but in pseudo colours from `jet` colormap instead. This problem is easily solved with `colormap(gray)` command

Images are displayed in `figure` graphics window. Any new image replaces the old picture in actual figure window – if any exists. As a result, it is not possible to compare old and new image, as the old one is reused. This problem is easily solved with this series of commands:

```
figure, imagesc(X), colormap(gray)
```

Laboratory work – pixel processing

Students should try to run in computer laboratory the examples presented on lecture. As some of them may be seen too simple – other examples may be considered. Figure 3 presents photography of electronic circuit (part of Ethernet card). Text on big square chip is difficult to read and students may use known graphic processing algorithms to enhance its readability. Small part of text from centre of the big chip was extracted (bottom left picture) and binarisation on level 128/256 was done to enhance contrast of the text with an excellent result. This example shows importance of picture enhancement in industry (e.g. robot vision)

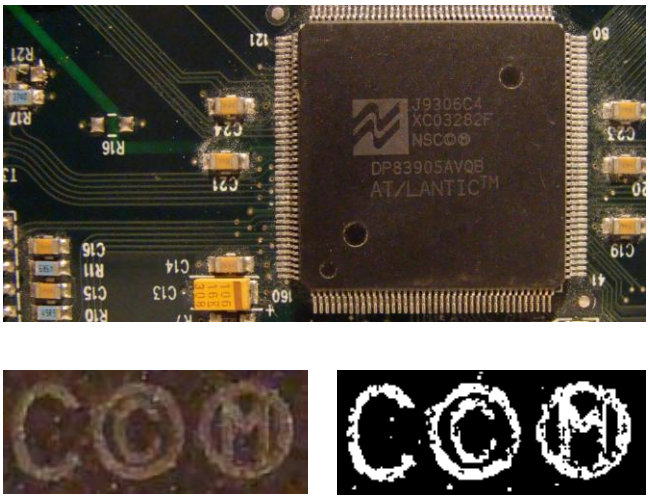

Fig.3 Original pictures (top and bottom left) and enhanced (bottom right) with binarisation (128/256)

Example of lecture: convolution filtering

Low-pass filter passes low-frequency signals and reduces the amplitude of signals with higher frequencies. In digital image processing it means that small change of light intensity along image will not be affected. But fast changes of contrast will be reduced. As result, high frequency noise (e.g. white and black pixels called salt and peeper) will be reduced. The drawback is some falling of sharpness of the image.

Low pass algorithm uses special small matrices (filters) of 3x3 dimensions or more. The filter size is uneven as it must have central point. The low pas filter examples are:

1/9 1/9 1/9	0.1 0.1 0.1	1/16 1/8 1/16
1/9 1/9 1/9	0.1 0.2 0.1	1/8 1/4 1/8
1/9 1/9 1/9	0.1 0.1 0.1	1/16 1/8 1/16

The convolution operation means substituting original value of each pixel with sum of product of value of this point with value of filter central point and products of its neighbours with corresponding values of filter. The new value of each pixel is weighted mean value of this pixel and its neighbours. Convolution algorithm is used in

MATLAB as `Xf=conv2(X,filter,shape)`, where `shape` parameter should be set to `'same'` dimension.

High-pass filter passes high-frequency signals and reduces the amplitude of signals with low frequencies. In digital image processing it means that fast change of light on image detail edge and sharpness of image will be strengthened. Two examples of filters used in high-pass filtering are:

```
-1  -1  -1          -1  -1  -1
-1   8  -1          -1   9  -1
-1  -1  -1          -1  -1  -1
```

High-pass filter makes edges more visible, but there are much more specialised edge detection filters as for example

```
 1   1   1          0   1   0
-1  -2   1         -1   0   0
-1  -1   1          0   0   0
```
NE-gradient (North/East), NE-Roberts,

```
 0   1   2          1   1   1
-1   0   1         -1  -2   1
-2  -1   0         -1  -1  -1
```
Sobel, Robinson,

```
-1  -1   2          3   3   3
-1   2  -1         -5   0   3
 2  -1  -1         -5  -5   3
```
Ehline-D, Kirsh

Many convolution filters with mathematical background (as gradient, Sobel, differ, Laplace (second order) and many other called „artistic“ filters may be found in professional graphic packages (e.g. paint shop) and even in menu of medium class digital cameras. Some professional filters are absolutely identical with examples given above and may be found in most of graphic packages as examples under menu of „user defined“ filters.

Examples from computer graphics laboratory on convolution filtering

Students do not have any serious problems with implementation of low-pass filtering of images.

They properly describe differences between original image and low-filtered image and they try to modify and improve filter matrix. If time permits, they even do three times filtering of RGB matrices of fullcolour images.

Fig. 4 Original image and low-pass filtered image

High-pas filtering and edge detection is more advanced challenge. Results are often completely unpredictable. Some of filtered images are completely black or have very low contrast (see figure.

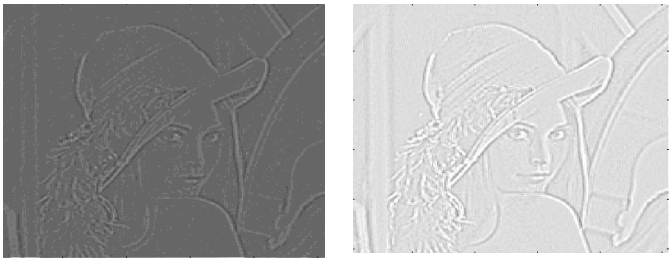

Fig.5 Unsuccessful high-pass filtering in students‘ lab

After easy success with low-pass filters, students are surprised and concerned. They are advised to work in groups and try to find solution in internet. But most of links go to complains of other students, looking for help with their projects or homework.

The next step is to point students attention to data types used in graphic matrices (`int8, uint8, short and double`), their range (-128..127, 0..255 for `int8` and `uint8` respectively) and functions used to process graphics data and functions used to display the results: `image(), imagesc(), colormap`.

The correct solution of high-pass filtration and edge detection needs to prevent overloading the range of uint8 graphic data during multiplication and to keep result within correct range of numbers. Practically students should convert graphic matrix X into `single` (recommended) or `double` precision, do normalisation to values within 0..1 range, do convolution with `conv2()`. When finished, result should be converted back to uint8. This will automatically limit all data within range 0..255. See fig. 6.

Fig. 6 Successful high-pass filtering (central value 8 or 9)

What is new in presented approach

1. Teaching is very flexible
2. Multimedia and internet are used during lecture
3. There is strong feedback with audience during lecture and lab
4. Individual work with computers in computer graphic laboratory and work in ad-hoc teams when needed
5. Deep understanding of data types and graphics display functions as basis of graphics processing

Ad.1 and 3 Slide-show sets the frames of the lecture.. Code examples are moved from slides to MATLAB environment on students eyes. Code is explained and then, depending on situation, code is fast run with demo data to show the results or is run line by line to explain details of its work. If needed, code is modified on-line to show extra properties or if students try to suggest improvements for the code. To attract students and keep their attention – lecturer may ask students to suggest the next step of computation or how to correct unpredicted result. As result, there is strong feedback with audience during lecture and lab and teaching is very flexible. If time do not permits for this flexibility – the discussion will continue on lab or during next lecture.

Ad 2 Following tools and equipment is used during lecture, lab and project work:
Slide show with high resolution digital projector (XGA), on-line animation of run of source code in MATLAB (debug mode may be used), on-line internet access during lecture and lab, Moodle e-learning environment is used as repository of teaching resources and educational database (student names and groups, their homework, projects and marks)

Ad 4 and 5 laboratory program is described in Moodle so student work individually on computer and do not disturb others. Reports of work done during lab should be uploaded to Moodle [3]. In case of problems student should discuss it with his neighbours (explaining the problem is higher level of learning and understanding) and if this fail –they should ask laboratory assistant for help. He may give them advice, or suggest them to work in ad-hoc teams of 2-4 persons.

Conclusions

This paper presents new approach to effective teaching fundamentals of computer graphics. The goal is to get deep understanding of fundamental concepts involved in the image processing and to show how to deal with unproductive results. Students have to develop and implement basic image processing software themselves, to gain better understanding of the subject. MATLAB (without toolboxes) was found to be the most convenient computing environment for this task. The code of examples may be found in [2]. As result, there is strong feedback with audience during lecture and laboratory.

References

1. Kurashima C.S. and Nascimento M.Z. Teaching Graphics and Image Processing in the Scope of Information Engineering http://www.matmidia.mat.puc-rio.br/sibgrapi2009/media/graphics_education/60051_2.pdf
2. Mrozek B, Mrozek Z. MATLAB i Simulink. Poradnik użytkownika. ed.3 (in Polish), chapter 7: Image Processing, Helion 2010 (in print)
3. Mrozek Z. Using Moodle, 19th EAEEIE Annual Conference, June 29 - July 2, 2008, Tallinn, Estonia
4. Rosenberger M. et.al, A novel approach for teaching digital image processing based on a new multi-scalable hardware platform XIX IMEKO World Congress Fundamental and Applied Metrology September 6−11, 2009, Lisbon, Portugal
5. http://www.idi.ntnu.no/~bilde/img/TIFF/lena.tiff
6. http://sipi.usc.edu/database/database.cgi?volume=misc The USC-SIPI Image Database
7. http://www.cs.cmu.edu/~chuck/lennapg/ The Lenna story, Imaging experts meet lenna in person

Date of paper submission

Zbigniew Mrozek

Re-inventing image processing course - an effective approach // Electronics and Electrical Engineering. – Kaunas: Technologija, 20XX. – No. X(XX). – P. XX–XX.

Abstract. Ill. X, bibl. X, tabl. X (in English; abstracts in English, Russian and Lithuanian).

The way students are taught has been changed in recent years. Easy access of to commercial or educational software and modern computer equipment makes teaching of image processing much easier than before. But very often students do not understand what is inside of each black box they use. So it was decided, that students have to develop and implement basic image processing software themselves, to gain better understanding of the subject. MATLAB (without toolboxes) is the most convenient computing environment for this task. Students are very impressed with fast and easy results obtained and show great interest in improving software they design and use.