A Raster Image Processing Application

PAVEL POKORNÝ, RASTISLAV PETRÁŠ
Department of Computer and Communication Systems
Tomas Bata University in Zlín, Faculty of Applied Informatics
nám. T.G. Masaryka 5555, 760 01 Zlín
CZECH REPUBLIC
pokorny@fai.utb.cz, http://www.utb.cz

Abstract: This paper describes an application for raster image processing which was developer last year at our faculty. This application is based on the .NET platform and was created for easy use by developers and users. Developers do not have to study the source codes in detail but they can use pre-created data classes without any other modifications. The data class has been programmed in such a way that a user has maximum control of the transformation algorithm. The programme enables us to perform many geometric and colour transformation algorithms which are described in this paper. Compared to commonly used programs, this enables one to change the parameters of calculations, which are not usually enabled in other applications. The next benefits are to create one’s own convolution mask for image filtration and to compare the time length of the same transformation in different ways of algorithm programming.

Key-Words: Image Editing, Image Processing, Image Filtration, Image Transformations, Colour Filters, Programming Application

1 Introduction
In the modern era, it is possible to see raster image processing very often, for example in newspapers, magazines, flyers, billboards or websites. Above all, these are where we can also find modified photos which do not capture the real situation. All these improvements are made with the help of a graphic program for editing raster images.

Simplicity and speed are a major aspect in the development of these programs, which mainly depend on the efficiency and optimization of the sub-algorithms. The modern hardware of electronic devices, like cameras or mobile phones, allows one to directly integrate graphic programs into them. These programs usually support simple modifications only and more complex image processing is applied later with a professional graphic program on a personal computer.

The most commonly used modifications among users are colour improvement and smoothing. But, raster image processing includes much more operations, like different geometric and colour transformations, working with filters, etc. The problem is that most of these transformations are difficult to process. This leads developers to optimize programs and hardware producers to develop new and faster electronic devices.

This paper describes an application for raster image processing. This application was developer last year at our faculty and it is based on the .NET platform. It allows us to make different geometric and colour transformations with images and user can also use some filtration algorithms.

2 Image Processing
An image, stored in the computer, is mainly represented by a matrix with the form: width x height. For the numbers of this matrix, the RGB colour model uses the lowest values (0) for black colour and 16,777,215 (2^24-1) for white colour [2]. And working with raster images is based on the manipulation with these points (pixels). According to the types of adjustments, image processing can be split into geometrical transformations, colour transformations and image filtering.

2.1 Geometrical Transformations
Geometrical transformations are the most commonly used operations in the field of computer graphics. The transformation can be split into two parts - linear and nonlinear. The Linear transformations include translation, rotation, scaling, mirroring, shearing and operations composed from these transformations. The characteristic for non-linear transformations is general deformation of images. An example of non-
linear transformation is warping, where an image is significantly distorted [1].

The application, which we have created, supports two linear transformations for raster images. The first of these is a scaling operation, which is based on linear interpolation. The second transformation is a rotation operation. In our application, there is the possibility of setting the rotation angle numerically or with the help of a slider. It is also possible to set the center of rotation.

2.2 Colour Transformations

Colour transformations are very important operations in image processing. This operation is often performed for the further image manipulation or printing. In order to keep as much image information as possible, it is necessary to maintain the hue of the original colours, and the transformation algorithm should change only the image intensity. The most commonly used colour operations are: transformation into grey-scale images, making a negative images, brightness and contrast adjustments, gamma corrections and working with a colour histogram [4]. All of these operations are supported by our application.

The simplest colour transformation is to make a grey-scale image. This operation processes each pixel of the raster image and is based on the base equation

\[ I = 0.266 \cdot R + 0.587 \cdot G + 0.114 \cdot B, \]  

(1)

where the I value is the output intensity (level of grey) and R, G and B are the basic components in the RGB colour model. Our application allows to change the numerical parameters in the equation (1) to get different output images.

The second simple image operation is its transformation into a negative image. In this case, the intensity of the pixel’s brightness is calculated in the following way:

\[ g(x, y) = L - 1 - f(x, y), \]  

(2)

where L is the degree of level brightness, \( f(x, y) \) and \( g(x, y) \) are input and output images. For example, L for an 8-bit colour image has the value of 256.

Brightness adjustment is the one of most commonly used operations in the course of image processing in order to improve an image’s quality. The change of brightness is based on the transformation of the brightness values, as follows:

\[ g(x, y) = T[f(y, y)], \]  

(3)

where the T symbol represents the transformation function. The application which we have created supports the percentage brightness changes in the default interval \([-10, 10]\), but it is also possible to change the minimum and maximum values of this interval.

It is necessary to improve the contrast of images that contain the important image information in a narrow range of brightness level. In this case, the image transformation is calculated in the following way:

\[ g(x, y) = \frac{f(x,y) - f_1}{f_2 - f_1}, \]  

(4)

where the \( f_1 \) and \( f_2 \) values represent the minimal and maximal brightness values inside an image. The application that we created allows us to change the image contrast with the help of the numerical coefficient in a defined range of positive and negative numbers.

Gamma correction is the most commonly used non-linear operation inside images. This operation is used to correct nonlinearities of imaging systems and exposures. The level of brightness in different parts of an image is balanced by the equation:

\[ i' = \left( \frac{i}{L} \right)^\gamma, \]  

(5)

Where \( \gamma \) is given by the type of display. This value is pre-set and some monitors can automatically calibrate it. The application which we have created allows us to set the gamma correction in two ways: either separately for each component of the RGB colour model or equally for all components. Furthermore, it is possible to set the minimum and maximum values for the changes of these components. All of these parameters are shown in Figure 1.

![Fig. 1 Setting parameters for gamma correction](image-url)

Histograms are one of the most effective tools for quality control and image editing. They
represent the vector of absolute frequency values which an image contains. Our application supports 4 diagrams of histograms. These histograms represent the R, G and B colour components in the image and in addition, frequency of brightness values.

2.3 Image Filtration

A process where some image information is highlighted or reduced is usually called an image filtration. The application which we have created supports some filters for smoothing, Gaussian smoothing, sharpening, medium sharpening, embossing and edge detection.

We use a convolution mask to realize graphics filters. The convolution mask is a matrix with specific dimensions. The application uses a 3 x 3 dimensions matrix. The principle of this matrix is based on the fact that there is a pixel in the middle which is surrounded by eight other pixels with a specific weight.

The total value of the matrix is divided by a factor and the result is added to a certain value. All these parameters can be set in the application - as is shown in Figure 2. These parameters can be also automatically set if we select the appropriate filter (Figure 2 – on the right).

A common image smoothing algorithm is suitable for use with images with very high details or with sharp gradients. If we select Smooth Choice in the application, the middle value of the convolution matrix is set to 8, and the boundary values are set to 1, as shown in Figure 2.

The Gaussian Image Smoothing algorithm works in a similar way. The difference between these cases is, the Gaussian smoothing algorithm gives us more natural and more vivid blur. The Gaussian smoothing algorithm automatically sets the lower values in the corners of the convolution matrix.

The Image Sharpening algorithm is the inverse function to the smoothing function. The principle of sharpening is based on the increase of the differences between the values of neighbouring pixels. In this case, the convolution mask only removes the values in the vertical and horizontal direction. To enlarge the degree of sharpness, we only need to change the middle value of the convolution mask.

Medium image sharpening (the Mean Filter) works on the same principle as the common sharpening described in the previous paragraph. The main difference is, the medium image sharpening function removes the values from all neighbouring points of the convolution mask.

The main task of the displacement pattern algorithm is to impose an impression of plasticity by edge highlighting. To this end, our application supports five different methods of displacement - Laplacian, Horizontal, Vertical, Loos and All Direction. Each method has its own setting of the convolution mask, so we can achieve different image outputs.

The edge detection algorithm is the last image filtration algorithm which our application supports. The algorithm is based on a very simple principle. The convolution mask usually has a constant value in the middle of the matrix - only the neighbouring pixels can change their value. The application that we have created supports five different edge detection algorithms - Edge Detect, Prewitt, Sobell, Robinson and Kirsha.

3 The Application’s Conception
The application which we created was developed in the C# programming language and it is based on the .NET Framework platform. The C# language is a very powerful and logically structured language that inherited the advantages of its predecessor languages - C++ and Visual Basic[3].

The .NET Framework core is based on the principles of object oriented programming principles. It also supports classes, methods, constructors, inheritance, polymorphism, threads, etc. Moreover, the .NET technology does not make any differences between the 32 and 64 bit systems – the program code is shared by all platforms[5].

The application architecture is based on the main class which serves the entire user interface of the program, takes care of communication and provides for all of the common commands and work with files.

The main class, that contains all of the implemented image processing algorithms, is called ImageAlgorithms. The actual transformation algorithms are located in private classes and all the necessary methods, by which the algorithms are called, are public. Each public operation runs a calculation method in the thread and displays a dialogue box during the operation.

Another class – ConvolutionMatrix - contains the convolution mask structure, which is used for filters. It contains all of the required methods and attributes to set or reset the matrix.

The programme also includes the AboutBox dialog box, which provides the user with information about the program. The second dialog box is the Progress box, which is running throughout the calculation time and informs the user that the program is still working.

The application’s user interface (Figure 4) is divided into three parts. The first part is the command menu on the top of the main window. In this menu there we can find all of the control program commands. These commands are grouped into menus File, Edit, Algorithm Settings and Info. The second part of our application is located on the left side of the main window. It contains all control elements and settings for implemented transformation algorithms. All transformation algorithms have set default values that can be simply change by entering new values. The third part of the user interface is located on the right of the main window and shows the loaded image with applied transformation algorithms.

All images can be simply loaded and saved. Our application supports most commonly used 2D image raster graphic formats - .bmp, .gif, .png, .tiff and .jpg.

4 Conclusion
This paper describes the application for 2D raster image processing which was developed at our faculty. This application is fully functional, but we are still working on its enlargement and completion. We are planning to implement some new additions to this application – more setting parameters for transformation algorithms, the preview function implementation before applied transformations, to add shortcut keys for commands, to add tooltips and to improve Undo command, to it works more effective.

References:

Fig. 4The user interface of our application