DEVELOPMENT OF A MATHEMATICAL MORPHOLOGY TOOL FOR EDUCATION PURPOSE

César C. NUÑEZ and Aura CONCI
Federal Fluminense University of Rio de Janeiro, Brazil

ABSTRACT: Morphology (the study of the form of objects) is very important for image analysis and processing. Mathematical Morphology is a branch of Digital Image Processing indicated to object and pattern identification. It is based on a new way of consider image representation and description. It simplifies an image by removing irrelevant details and noises with which the essential characteristics of the form of objects remain intact. Then it is a very important help on objects and pattern recognition. For morphological operations on bi level images (i.e. black and white) we use sets in Z^2 (Cartesian product of Integer number set Z). Conventionally, the two coordinates (x,y) of a image point, in a collection of objects or elements (set), are registered if it represent one pixel of the object depicted in the image. Pixels of the background are not registered as an ordered pair. Concepts of set theory as: universal set, intersection, union, complement, difference, subset, inclusion and reflection are used on the construction of the main operations of Mathematical Morphology (which are: dilation, erosion, opening and closing). In these operations, a small set named structuring element is used. The obtained results depend not only of the type of operation but also of this set. Another characteristic is the construction of new operations using others previously defined. For example, opening is defined as erosion followed by dilation and closing is defined as dilation followed by erosion. Mathematical Morphology has been implemented in this work using a Object Oriented JavaScript environment that can be used for learning proposes in several different ways. In the simplest way, on the internet, it allows experiences using the implemented operations (as expansion, contraction, dilation, erosion, opening, closing, intersection, union, subtraction, complement, reflection, etc). It permits direct experiences using binary images with any structuring element that can be (both) directly drawn on the screen using painting tools of various types. For students following a formal course of image analysis it is possible to try combining operations like top-hat, bottom-hat, hit-miss and any other which have been previously defined, in a form of script that is composed on the screen of the tool by the user. New morphological function can be included as a specific function in a new code version in a manner such as buttons or options from the menu. Combined operations like morphological watersheds can be added easily. Students of JavaScript language can learn the basic structure of the program and improve it including new functions since the code is open and well documented. Students interested on special topics on image analysis, for instance, after trying the functions in a combined way can use the code in their specific codes. For beginner students taking a course of mathematical set theory it can be used as a program of self study because its tutorial explains what is the propose of each stage. In addition, it works as laboratory experiments for use in classrooms.

Keywords: Mathematical Morphology, Image Analysis, Digital Image Processing.

1. INTRODUCTION

The Mathematical Morphology is related with the mathematical sets theory. This way, it offers a powerful standardized approach to various image processing. In Mathematical Morphology the sets represent the object forms on an image. For example, the set of all black points on a binary image - considering a universe of defined points - is a complete description of this image. On binary images, the sets in question are members of the bi-dimensional space of integer numbers (Z^2), in which, each element of the set is a bi-dimensional vector, are the ordered pair or the Cartesian coordinate (x,y) of the black points of the image. Digital images can be represented by sets whose components are in larger dimensional spaces, and may contain other attributes of the image, such as various levels of gray, color, or components that may vary with time. In this article, we consider exclusively binary images that are images represented only by their black point sets. We do not present the development of the mathematical morphology itself; we suppose that the user is familiarized with this matter. However, we present some examples that will make clearer some of the results obtained with each morphologic transformation, applied separately, or in group.

The following section 2 lists the tool and its functions. Then section 3 and 4 treat of the use of the system showing some examples of operations. Finally section 5 presets some conclusion.

2. THE TOOL

The tool introduced in this article is an open code tool and it has been created to be used on the main browsers/compatible in the market, as long as they are supported by CSS (cascade style sheets) and DOM (Document Object Model). There is no need for any additional plug-in or any other components. Its codification uses object orientation paradigms and it was developed basically in Javascript. It makes possible the construction of simple images in a matrix of 30x30 points, with color depth of 1 bit (black and white), and it applies up to 11 transformation operations combined in any quantity and order.

2.1 Tool's Objects

The tool has two main classes of objects: matrixDisplay and imageObject.

The former defines a type of object that is simulated on the monitor screen. It creates a square block matrix (each with 9x9 pixels), which works as the elements of an image in the user display. This matrix's dimension can be dynamically defined, on the creation of the object's instance. Each block can assume two states: black and white, it allows the representation of images of the bipmap type. The data about each block's conditions are kept in a bi-dimensional array and can be altered according to the image being viewed. This image is linked to an object propriety, and at any moment, the active image can be altered, as long as a new image is created (also stored as an array, as commented further) to this propriety. This class has only one method, clearDisplay, which is used to clean the display, changing all blocks into white.

On the demo applicative, two displays have been created: The main display, with 30x30 blocks, and the Structuring Element display, with 5x5 blocks.

Main display specifies the proprieties and methods that will serve to define an image. Each image has, basically, two proprieties: Its own identifier, and the bi-dimensional coordinate array, which represents the active image's points. Its five methods are: addPixel; delPixel; showArray; showImage and showImage.

The method addPixel adds a new pair of coordinate to the array, at an specific point in the display. Method delPixel removes a determinate pair of coordinates from the array. Method showArray sets up a string that serves to represent, textually, the coordinate's array. Method showImage shows the linked display's image. Method clearAll deletes or annuls the

array's coordinates (destroys the image).
In addition to these two classes, the tool has various functions which perform tasks related to interaction with the user and apply on the image on use the operations, transforming the original image.

2.2 User interface

The user can interact with the tool through four distinct areas: The brush area, the main design area, the structuring element, and operations definition area. Figure 1 shows the interface of the tool.

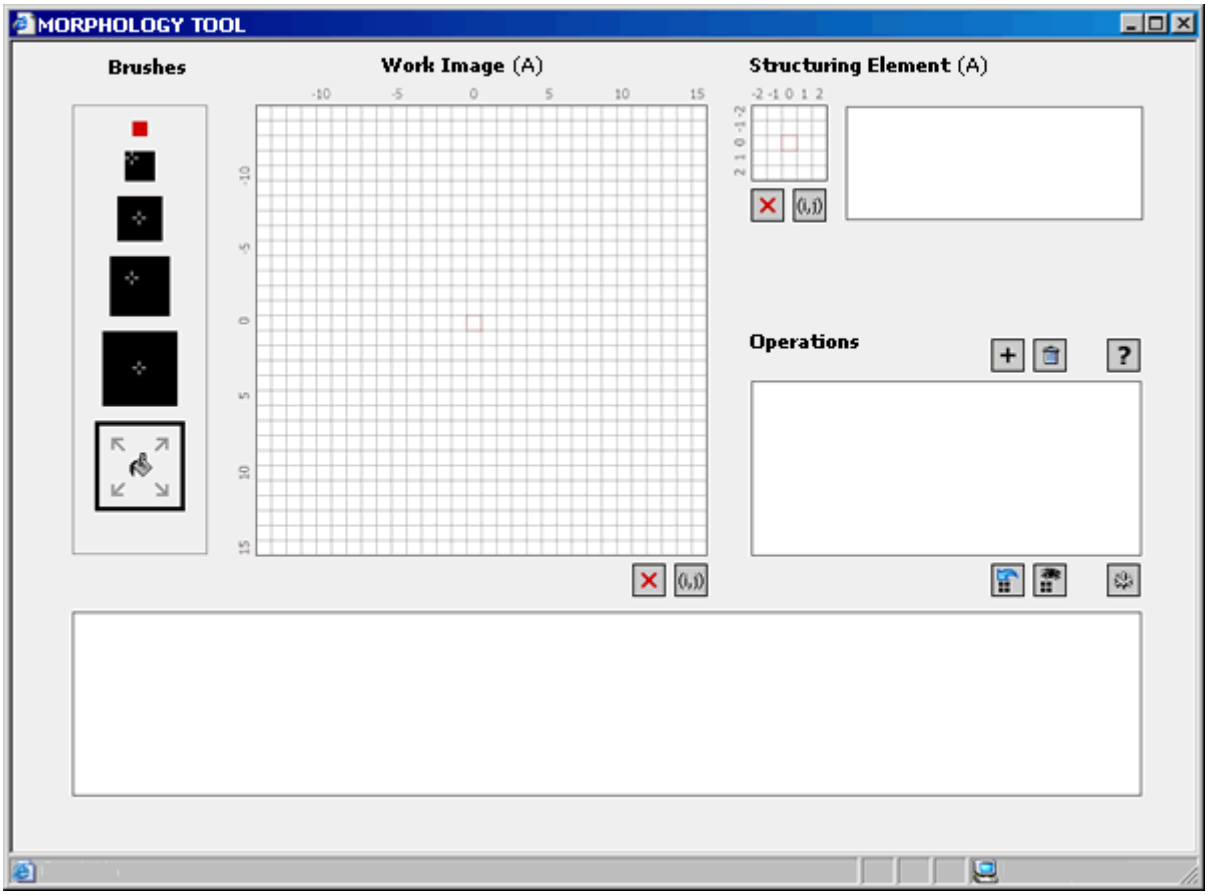

Figure 1: Tool Interface.

Let see the functions of each one. Brush areas, located on the top-left are used for definition of the brushes to be used for image construction. The possible brush type are arranged vertically on the side, with areas of 1, 2x2, 3x3, 4x4 and 5x5 points. There is also a bucket, that is a tool for filling quickly an entire area. When clicking with the brush on a white spot, the correspondent spots in this area will be painted black. If the clicked spot has already been painted, the brush will serve as an eraser, removing the black points from the area. The brushes only work on the design area.
Design Area is the area where the original image and its operations are shoed. It is presented as a matrix with 30x30 points, where the on work image is created. This image will suffer the selected transformations, and the resultant image will be viewed on the same position. The work image is linked to the letter "A" for identification on the transformation operations. The heading will indicate at the current time what image is being viewed. Under the matrix there is a box where image's

coordinate points are showed.
The Structuring Elements area, similar to the main design area, is a 5x5 point matrix for the construction of structuring elements. This element is necessary to compute many operations like dilation, erosion, opening and closing operations. Here the brush size is always 1 point, independently from whatever is selected.
On the right hand side there is a box showing the coordinate of the structuring elements points. Under the matrix there're also buttons for cleaning the design and for hiding/showing the coordinate box.
The operations definition area is the location where the used defied the transformations to be applied to the image. It permits to specify cascade of operational transformation which will be sequentially used. The controls in this box are concentrated in five operations: Add; delete; make; quick access; done and open help file. They promote the addition of a new operation to the stack; remove the non-selected operations from the stack; make the work image the active image in the design matrix; allow a quick access to the work image without making it the active image; perform the operations stored in the stack; and open the help file.

2.3 Shortcut keys

The interface has the following shortcut keys, correspondent to the interface visual controls:
TAB changes the brush size, sequentially;
1,2,3,4,5,6 turns possible to select directly the brush size; "1" being the smallest, and "6" the bucket;
DELETE cleans the main matrix active image. If the active image is the resultant image, the work image becomes the active image;
F2 key hides or exhibits the box with the active image coordinate on the main matrix;
DEL (in numeric keyboard part) clears the structuring element's image;
F4 key hides or exhibits the box with the structuring element's coordinate;
+ (plus key in the numeric keyboard) adds an operation to the operation stack;
- (minus key also in the numeric keyboard) removes the non-selected operations from the operation stack;
F1 key opens a pop-up window with the help file;
ESCAPE key makes the work image the active image of the main matrix;
SPACE BAR key exhibits the work image on the main matrix, without making it the active image. When releasing the space bar, the active image returns as the resultant image;
ENTER key performs the operations selected from the stack;

3. CONFIGURING A SEQUENCE OF TRANSFORMATIONS

This section will show how to make successive operations. To configure a sequence of transformations, the user must perform the following operations:
Click the add operation button.
The first box has letter "A", already, that represents the work image. The first selected operation must always have letter "A" in this box.
The user then selects the operation from the list-box. The dilation, erosion, opening, and closing operations will exhibit the structuring element (SE), which will appear in the next box.
In case oft the selected operation not be one of the above, the next field can be used with the application of the operation on another image that has already been calculated (each image may be represented by one letter only).
In the last field to the right of each operation one letter has to be chosen for the resultant image. This letter can be used in other operations, if one wants to apply transformations on cascade.
In case of many cascade operations, the resultant image will be the answer to the last operation, or their sequence, if they have been configured on cascade form. To cascade the operations, the letter of the last field has to be

the same as the letter of the first field of the next operation. Figure 2 illustrates an example of two cascade operations:

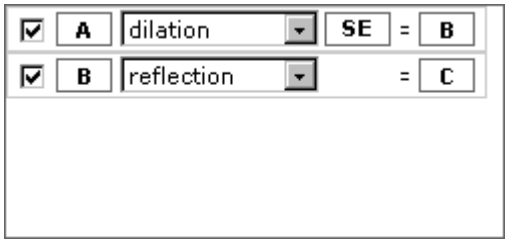

Figure 2: Cascade operations.

The first operation is dilation, applied to image "A" (work image) through the structuring element (SE), which resultant image is associated to letter "B".
The second operation is a reflection, applied to image "B" (calculated on the previous operation), which resultant image is associated to letter "C".
Since there is not any more selected operation, image "C" will be exhibited in the main matrix as the final image, and it is the result of the transformations of dilation and reflection, applied in this order to image "A".

4. EXAMPLES OF TRANSFORMATION

This section will show some figure examples of image transformation by applying some of the operations implemented in the demo applicative.

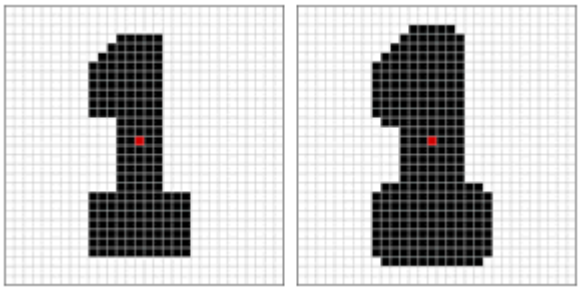

Figure 3: Work image and expansion applied with "four" neighborhood.

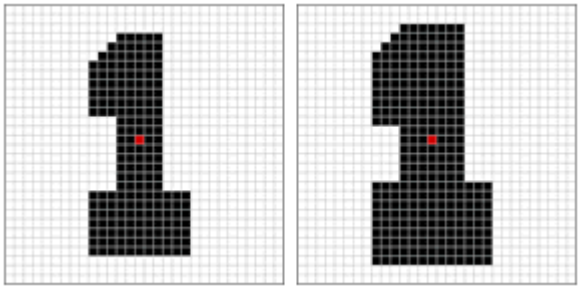

Figure 4: Work image and expansion applied with "eight" neighborhood.

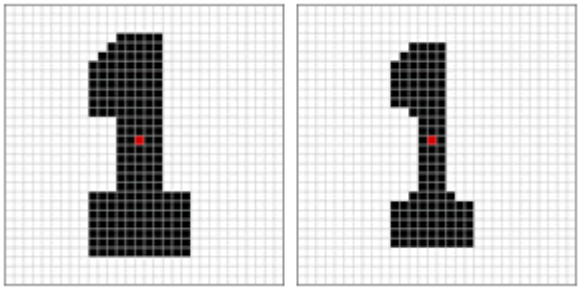

Figure 5: Work image and contraction applied with "four" neighborhood.

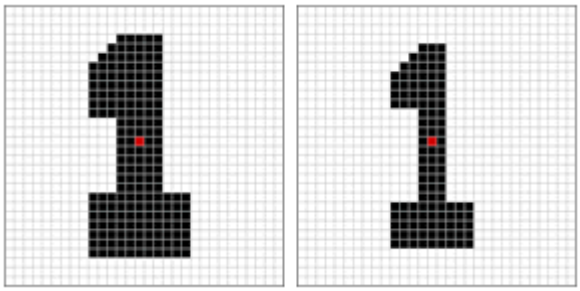

Figure 6: Work image and erosion applied with a 3 point-side square Structuring Element.

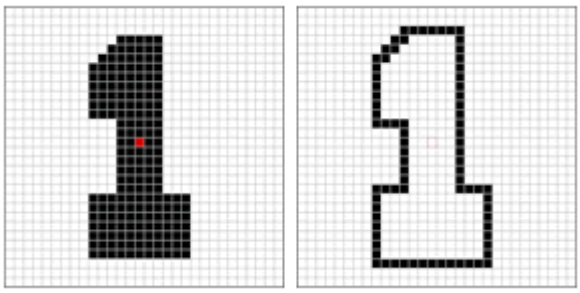

Figure 7: Work image and expansion with "eight" neighborhood followed by subtraction of the work image itself.

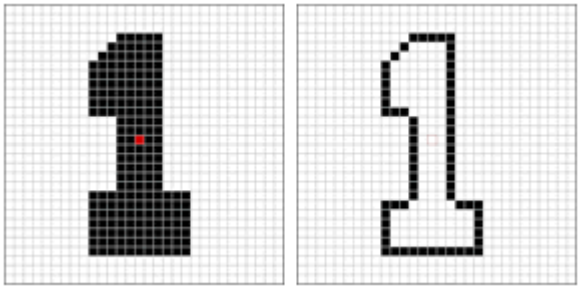

Figure 8: Work image subtracted by a contraction with “four” neighborhood.

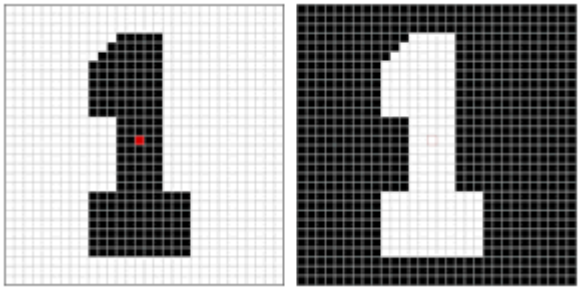

Figure 9: Work image and its complement.

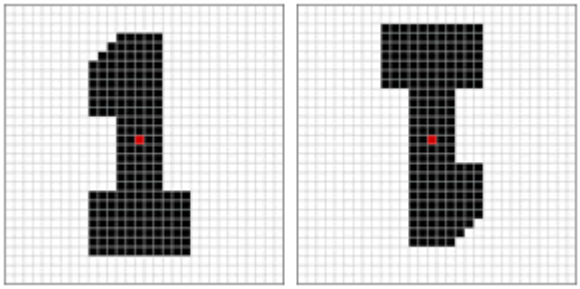

Figure 10: Work image and its reflection on the vertical and horizontal axes.

5. CONCLUSIONS

The tool presented in this article may be of great help to the study of Mathematical Morphology applied to image’s analysis. Its capacity to perform morphologic transformation operations in real time aggregates value to class explanations. It can be used as an experimental lab in the classroom or at the student’s home. Due to its simplicity and portability (can be stored on a floppy disk), it can be used on PCs, through private networks, or even through internet, since it only requires a browser to be executed. In addition, it has an open code, and it may receive new implementations which will add a greater number of morphologic operations to it, as to a gain in performance. On this way it can be used also in Javascript classes. This tool can be accessed or downloaded directly in http://www.ic.uff.br/~aconci/morphologytool.zip or by the site www.ic.uff.br/MM.html, where users can also make the download of the source-code.

REFERENCES

[1] Gonzales, R. C. and Woods, R. E. Processamento de Imagens Digitais. Editora Edgar Blücher (2003), 369-402.

[2] Facon, J. Morfologia Matemática: teoria e exemplos. Curitiba, 1996
Website: http://www.ppgia.pucpr.br/~facon/MorfologiaMatematica/ApostilaMorfoBinaria.zip

[3] Serra, J. Image Analysis and MAtematical Morfology, Academic Press, Londos, 1982
Website: http://cmm.ensmp.fr/^serra/cours/

ABOUT THE AUTHORS

1. César de C. Nuñez received his BSc in Industrial Design and Graphic Design from the University of Rio de Janeiro (Brazil) in 1980. At present, he is a MSc student at Computer Institute in the Federal Fluminense University, in the field of Visual Computation and Interface research, and he is also a teacher of the graduate course (lato sensu) on Internet, Interface and Multimedia at the same university. His areas of interests are: Interface studies, Graphic Computation, and Image Processing. He can be reached by e-mail: cnunez@ic.uff.br.

2. Aura Conci, Dr.Sc. (since 1988) is currently titular professor in the Department of Computer Science in Federal Fluminense University at Niteroi (Brazil). Her research interests include Biomechanics, Applications of Computer Vision and Image Processing. She can be reached by e-mail: aconci@ic.uff.br Website: http://www.ic.uff.br/~aconci.