Morphological Operations Applied to Crack Detection in Digital Art Restoration

M. Kirbie Dramdahl
Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA 56267
dramd002@morris.umn.edu

ABSTRACT

This paper provides an overview of the processes involved in detecting and removing cracks from digitized works of art. Specific attention is given to the crack detection phase as completed through the use of morphological operations. Mathematical morphology is an area of set theory applicable to image processing, and therefore lends itself effectively to the digital art restoration process.

Keywords

mathematical morphology, edge detection, crack detection, crack removal, inpainting, digital art restoration

1. INTRODUCTION

The restoration of paintings and other works of art is an important aspect in preserving objects of artistic, cultural, or historic significance. However, this is also an expensive, demanding, and time-consuming process reserved for the abilities of specialists [8]. For this reason, the process of digital art restoration has been developed, which provides both a comparatively inexpensive alternative and a nondestructive tool to be used in the planning of physical restoration. Additionally, digital art restoration allows art historians an approximation of the initial form of a work [10].

While there are many forms of damage a piece of art may accumulate over time, one of the most common deteriorations is cracking. These cracks, or *craquelure*, may form as a result of climate, drying, or mechanical factors [3, 4, 10]. This paper examines the detection and removal of cracks from pieces of digitized art. The main emphasis is on the detection phase, as carried out by mathematical morphology, which is a method of image processing based in set theory.

This paper breaks the digital art restoration process into three steps: edge detection, crack detection, and crack removal. Section 2 covers edge detection by examining an optimal edge detector known as the Canny edge detection algorithm. Section 3 provides an examination of the four basic morphological operators of erosion, dilation, opening, and closing. This information is necessary to understanding Section 4, which covers crack detection, and is the main focus of this paper. Section 5 wraps up the digital art restoration process in covering the crack removal process (known as *inpainting*). The remaining sections provide a summary. Section 6 reviews the results obtained by various methods of digital art restoration, again with a specific focus on the success of various morphological operators in the crack detection stage, and Section 7 provides a summary of the information provided in this paper, as well as discusses possible further areas of interest and study.

UMM CSci Senior Seminar Conference, April 2014 Morris, MN.

2. EDGE DETECTION

Edges within a digitalized work of art are boundaries between areas of varying intensity. In other words, an edge occurs where there is “a jump in intensity from one pixel to the next” [6]. Intensity, in art, is defined as brightness or dullness of a color; the brighter the color, the higher its intensity. For example, in greyscale images, white has the highest intensity and black has the lowest intensity.

Edge detection is actually not necessary to the crack detection and removal process, and does not rely on mathematical morphology. However, this paper provides a brief discussion of the process for two reasons. The first is that edge detection provides the advantage of reducing the amount of data to be processed while still preserving the necessary edge information. The second is that the choice of whether or not edge detection is used affects the morphological filters used later in the crack detection and removal process [6]. While there are several possible methods of edge detection, this paper will examine the Canny algorithm [2, 6].

The Canny algorithm is based upon three criteria, which are as follows [2, 6]:

1. Accuracy - The edge detection algorithm should have a low error rate. This implies that there should be neither false positives (edges detected where they do not exist) nor false negatives (edges not detected where they do exist).

2. Localization - The edge detection algorithm should minimize the distance between the edge detected and the actual edge.

3. Uniqueness - The edge detection algorithm should have only one response to a single edge.

In order to fulfill these criteria, the Canny algorithm first uses a Gaussian filter to smooth the image. The Gaussian filter functions in a manner somewhat similar to the structuring element used in mathematical morphology, discussed in Section 3. Essentially, the filter passes over each pixel in turn, blending each into the surrounding pixels. The complete result is a slightly blurred version of the original image with no isolated, "noisy" pixels (pixels that exhibit a significant jump in intensity from all surrounding pixels).

Having eliminated noise, the algorithm then takes the gradient of the image. An image gradient is a map of the directional changes in intensity within an image. The gradient is then used to locate regions where there are significant jumps in intensity. The regions identified here are then searched for local maximum. All non maximum pixels are set to zero. In the final step, the Canny algorithm further refines the detected edges by comparing the remaining pixels to two thresholds. If the intensity of a pixel falls below the low threshold, it is set to zero. If the intensity of a pixel falls above the high threshold, it is set to one (made an edge). If the intensity of a pixel falls between the two thresholds, it is set to one if at least one adjacent pixel has an intensity above the high threshold, and set to zero otherwise [6]. The result of the Canny algorithm is a binary image depicting the edges of the initial input image.

3. MORPHOLOGICAL OPERATIONS

Following optional edge detection is crack detection, the focus of this paper. However, in order to understand this stage, it is first necessary to describe the basic principles of mathematical morphology. Typically, morphology is used in processing binary (black and white) images, although there are also variations used for greyscale images. This paper will examine both variations. Morphological functions take two inputs. The first input is the image to be processed. In binary morphology, the input image is divided into foreground (typically white) and background (typically black) regions. In greyscale morphology, the image is treated as a three-dimensional surface, where higher areas are brighter (white) and lower areas are darker (black). The second input is a structuring element, a (typically small, in comparison to the image) set of coordinate points. An example of a 3x3 square structuring element with the origin located at the center in comparison to both a 30x20 binary image and an 11x32 greyscale image is presented in Figure 1. The structuring element is then used to modify the input image. The image that results from the morphological operation is determined by the shape, size, and point of origin of the structuring element [1, 5, 7, 9]. Subsections 3.1, 3.2, 3.3, and 3.4 explain the fundamental operations of mathematical morphology.

3.1 Erosion

Erosion of an image strips away a layer of pixels from the boundaries of foreground regions, and is denoted by the equation

$$g = f \ominus s$$

where g is the resulting image, f is the original image, and s is the structuring element. The symbol $\ominus$ indicates erosion [1, 5].

In binary erosion, erosion is accomplished by placing the origin of the structuring element over every pixel of the foreground regions in turn. If every point within the structuring element is in line with a foreground pixel, the foreground pixel lined up with the origin of the structuring element is left unchanged. If at least one point within the structuring element is in line with a background pixel, then the pixel lined up with the origin of the structuring element is converted to a background pixel [9]. An example of binary erosion on a 30x20 image using a 3x3 square structuring element with the origin located at the center is presented in Figure 2. The erosion of foreground regions is equivalent to the dilation (discussed in Subsection 3.2) of background regions [9].

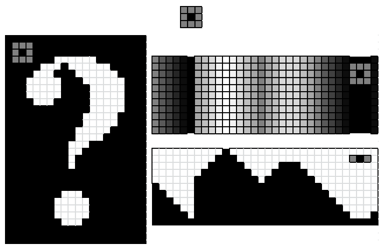
Figure 1: Structuring Elements: Top: Structuring Element. Bottom Left: Structuring Element in Comparison to Binary Image. Bottom Right: Structuring Element in Comparison to Greyscale Image (Above) and Cross-Section (Below).

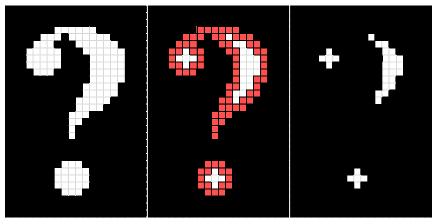
Figure 2: Binary Erosion: Left: Original Image. Center: Erosion Marked. Right: Results of Erosion.

Gray scale erosion follows the same basic pattern, but is somewhat more complex. Here, the origin of the structuring element is placed in line with a pixel on or below the surface. If every point within the structuring element is also either on or below the surface, the pixel lined up with the origin is left unchanged. If at least one point within the structuring element is above the surface, the pixel lined up with the origin of the structuring element is essentially removed (set to be above the surface). This process is repeated for every pixel on or below the surface [7, 9]. An example of greyscale erosion performed on a cross-section of an 11x32 image using a 3x3 square structuring element with the origin located at the center is presented in Figure 3.

3.2 Dilation

Dilation of an image adds a layer of pixels to the boundaries of foreground regions, and is denoted by the equation

$$g = f \oplus s$$

where the symbol $\oplus$ indicates dilation [1, 5].

In binary dilation, this is accomplished by placing the ori-

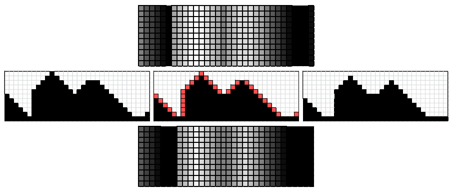

Figure 3: Greyscale Erosion: Top: Original Image. Center Left: Cross-Section of Image Surface. Center Center: Erosion Marked. Center Right: Erosion of Surface. Bottom: Results of Erosion.

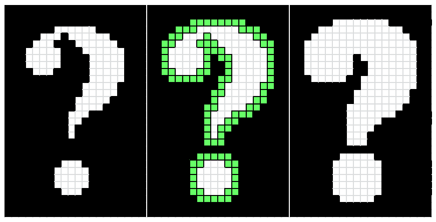

Figure 4: Binary Dilation: Left: Original Image. Center: Dilation Marked. Right: Results of Dilation.

gin of the structuring element over every pixel of the background regions in turn. If every point within the structuring element is in line with a background pixel, the background pixel lined up with the origin of the structuring element is left unchanged. If at least one point within the structuring element is in line with a foreground pixel, then the pixel lined up with the origin of the structuring element is converted to a foreground pixel [9]. An example of binary dilation on a 30x20 image using a 3x3 square structuring element with the origin located at the center is presented in Figure 4. The dilation of foreground regions is equivalent to the erosion (discussed in Subsection 3.1) of background regions [9].

Like greyscale erosion, greyscale dilation is a more complex variation its binary counterpart. Here, the origin of the structuring element is placed in line with a pixel above the surface. If every point within the structuring element is also above the surface, the pixel lined up with the origin is left unchanged. If at least one point within the structuring element is on or below the surface, the pixel lined up with the origin of the structuring element is set to be on or below the surface. This process is repeated for every pixel above the surface [7, 9]. An example of greyscale dilation performed on a cross-section of an 11x32 image using a 3x3 square structuring element with the origin located at the center is presented in Figure 5.

3.3 Opening

The opening of an image is an erosion followed by a dilation, and is denoted by the equation

$$g = f \circ s = (f \ominus s) \oplus s$$

where the symbol $\circ$ indicates opening [1, 5]. Similar to erosion, opening strips away foreground pixels at the boundaries of foreground regions, but is less destructive of the initial foreground regions than erosion. Opening is therefore typically used to preserve foreground regions with a similar size and shape to the structuring element, while removing or reducing other foreground regions [9]. An example of binary opening on a 30x20 image using a 3x3 square structuring element with the origin located at the center is presented in Figure 6. The opening of foreground regions is equivalent to the closing of background regions [9].

The greyscale opening of an image also consists of an erosion followed by a dilation. In this situation, however, both the erosion and dilation are greyscale rather than binary [7].

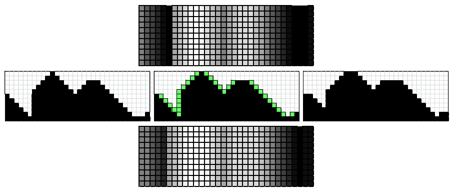

Figure 5: Greyscale Dilation: Top: Original Image. Center Left: Cross-Section of Image Surface. Center Center: Dilation Marked. Center Right: Dilation of Surface. Bottom: Results of Dilation.

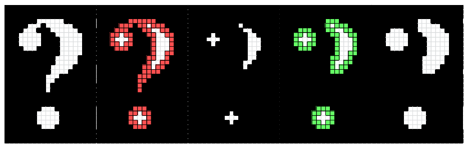

Figure 6: Opening: Left: Original Image. Second from Left: Erosion Marked. Center: Results of Erosion. Second from Right: Dilation Marked. Right: Results of Dilation (Opening Complete).

3.4 Closing

Closing of an image is a dilation followed by an erosion, and is denoted by the equation

$$g = f \bullet s = (f \oplus s) \ominus s$$

where the symbol $\bullet$ indicates closing [1, 5]. Similar to dilation, closing adds foreground pixels at the boundaries of foreground regions, but is less destructive of the initial background regions than dilation. Closing is therefore typically used to preserve background regions with a similar size and shape to the structuring element, while removing or reducing other background regions [9]. An example of binary closing on a 30x20 image using a 3x3 square structuring element with the origin located at the center is presented in Figure 7. The closing of foreground regions is equivalent to the opening of background regions [9].

The greyscale closing of an image also consists of a dilation followed by an erosion. In this situation, however, both the dilation and erosion are greyscale rather than binary [7].

4. METHODS OF CRACK DETECTION

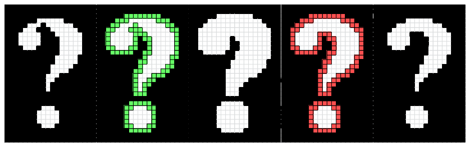

Figure 7: Closing: Left: Original Image. Second from Left: Dilation Marked. Center: Results of Dilation. Second from Right: Erosion Marked. Right: Results of Erosion (Closing Complete).

The next step in the crack detection and removal process (following optional edge detection), is crack detection. This section will discuss the application of various morphological filters to this problem.

4.1 Top-Hat Transform

The morphological filter most frequently used for detecting cracks within digitalized works of art is the top-hat transform. There are two variations on this filter, the black top-hat transform and white top-hat transform, which will be discussed in Subsubsections 4.1.1 and 4.1.2 respectively.

4.1.1 Black Top-Hat

The black top-hat transform (also referred to as the closing top-hat transform) is used for detecting darker details on a lighter background within greyscale images [3, 10]. This transform is defined as the difference between the closing of the original image by a specified structuring element and the original image itself, and is denoted by the equation

$$BTH = (f \bullet s) - f$$

where f is the original image and s is the structuring element. This transformation produces a greyscale image with the desired details enhanced.

4.1.2 White Top-Hat

The white top-hat transform (also referred to as the opening top-hat transform or, sometimes, the bottom-hat transform), is used for detecting lighter details on a darker background within greyscale images [3, 10]. This transform is defined as the difference between the original image itself and the opening of the original image by a specified structuring element, and is denoted by the equation

$$WTH = f - (f \circ s)$$

where f is the original image and s is the structuring element. Like its counterpart, this transformation produces a greyscale image with the desired details enhanced.

4.1.3 Multiscale Top-Hat

Subsubsections 4.1.1 and 4.1.2 explain classical top-hat transforms. While these are popular and frequently used, they are not perfect. Paintings often contain details (such as fine brush strokes) which are misinterpreted as cracks by these filters. The multiscale morphological top-hat transform may be used to counteract this, as well as to detect cracks of varying sizes [3]. This transformation is executed by performing a series of classical top-hat transforms with structuring elements of various shapes and sizes (Cornelis et al use square structuring elements "ranging from 3x3 to nxn pixels, where n depends on the width of the crack to be detected" [3]). The resulting images, or crack maps, are then further processed to remove isolated groups of pixels and bridge pixel gaps.

At this point, the crack maps are combined in order to build one final, comprehensive crack map. Cornelis et al accomplish this by first combining the crack maps derived from the three smallest structuring elements (the finer, thinner cracks) in what is referred to as the base map [3]. From here, the crack maps derived from the larger structuring elements (the coarser, heavier cracks) are layered onto the base map. This is done by only adding those coarser cracks which are connected to the finer cracks of the base map. The resulting final crack map both contains cracks of a variety of scales and reduces the occurrence of false positives [3].

4.2 Alternative Methods

While the top-hat method is the most common morphological filter applied to the problem of crack detection in digital art restoration, there are other methods of finding a solution. Karianakis and Maragos present one such alternative in their digital restoration of prehistoric Theran wall paintings [8].

In the method implemented by Karianakis and Maragos, the process begins by setting a threshold. If the variation in the intensity of one pixel from its neighbors exceeds the threshold, it is determined to be part of a crack. All other pixels are considered genuine. At this point, morphological closing is applied to the binary image obtained from the previous step. This groups isolated crack pixels, and removes some of the discoloration that may be found around the edges of cracks (this may be a result of light reflecting off paint ridges, or accidental removal of paint during any previous cleaning process [3]). The result is the binary mask of cracks [8].

Karianakis and Maragos then return again to the original input image. From this image, the Canny edge detection algorithm (described in Section 2) is used to obtain a binary mask of all edge pixels (both cracks and actual edges within the original painting). Morphological dilation is then applied to the crack mask in order to group any isolated pixels [8].

At this point, Karianakis and Maragos now have two binary masks: the crack mask obtained by thresholding, and the edge map obtained by the Canny edge detection algorithm. These two masks are then joined, and the resulting binary mask is then iteratively eroded until immediately before a certain percentage of edge information is lost. This process results in a smoother crack mask and, depending on how much edge information is lost, reduces the likelihood of false positives [8].

5. INPAINTING

The final stage in the digital art restoration process is crack removal, frequently referred to as inpainting. As with crack detection, there are many potential solutions to the crack removal process. This section will examine the method described by Spagnolo and Somma in their study [10].

In this method, the original input image is broken down into regions, which are further divided into neighborhoods (Spagnolo and Somma define 5x5 pixel neighborhoods and 100x100 pixel regions [10]). From the previous crack detection step, the defective pixels are already known. For this information, it is now possible to proceed with the inpainting process [10].

The first step is to select a defective pixel i, and to determine its context. The context of a defective pixel i consists of all the non-defective pixels in the neighborhood of i. From here, all other neighborhoods in the region containing i (neighborhoods containing defective pixels are excluded) are examined in order to determine which neighborhood in the region is most similar to the context of the defective pixel i. This is done by computing the sum of squared differences between the context of i and the corresponding pixels in the neighborhoods within the region of i. In this situation, the sum of squared differences is defined as the average of the change in intensity between corresponding pixels squared. The neighborhood with the lowest sum of squared differences is determined to be most similar to the context of i [10].

If the sum of squared differences for this similar neighborhood is below a set threshold (that is, if the neighborhood it determined not only to be the most similar to the context of i, but also similar enough to be an acceptable match), the defective pixels within the neighborhood of i (including i itself) are replaced by the corresponding pixels from the "matched" neighborhood. If the sum of squared differences is above the threshold, i is replaced with the median value of the non-defective pixels within its neighborhood. This process is repeated for every defective pixel [10].

6. RESULTS

The final result of the crack detection and removal process depends significantly on the effectiveness of the crack detection method implemented. The results of crack detection may be broken down into four categories, which are as follows [4]:

- True Positives - Cracks detected where they do exist.
- False Positives - Cracks detected where they do not exist.
- True Negatives - Cracks not detected where they do not exist.
- False Negatives - Cracks not detected where they do exist.

One way of evaluating the effectiveness of a particular crack detection algorithm is to determine the occurrences of these various results. Effective algorithms will produce both high rates of true positives and true negatives and low rates of false positives and false negatives, while less effective algorithms will produce low rates of true positives and true negatives, high rates of false positives and false negatives, or a combination of both. There are two methods of simplifying this process. The first is to use these categories to determine false positive rate and true positive rate, respectively denoted by the following equations [4]:

$$FP = fp/(fp + tn)$$

$$TP = tp/(tp + fn)$$

where tp is the number of true positives, fp the number of false positives, tn the number of true negatives, and fn the number of false negatives. The second option is to determine precision (fraction of retrieved instances that are relevant) and recall (fraction of relevant instances that are retrieved), respectively denoted by the following equations [8]:

$$P = tp/(tp + fp)$$

$$R = tp/(tp + fn)$$

Note that recall is identical to the true positive rate. Subsections 6.1 and 6.2 will examine the effectiveness of the various methods of crack detection discussed in Section 4.

6.1 Top-Hat Transform Results

The black top-hat transform and white-transform, discussed in Sections 4.1.1 and 4.1.2 respectively, were implemented in the methodology of Desai et al [4]. Prior to crack detection, Desai et al first classified images into three categories: crack thickness, number of cracks, and crack connectivity. In their study, they report both a maximum true positive rate of 0.98 and minimum false positive rate of 0.02 for those images classified by number of cracks. The maximum true positive occurred for images with a moderate number of cracks, while the minimum false positive was obtained from images determined to contain a low number of cracks. On the opposite end of the spectrum, crack connectivity produced both a minimum true positive rate of 0.86 and a maximum false positive rate of 0.124. Both occurred for images with less crack connectivity [4].

Cornelis et al also made use of the black and white top-hat transforms in their methodology [3]. However, while Desai et al used the classical implementation of these algorithms, Cornelis et al instead used the more involved multiscale variation described in Subsubsection 4.1.3 [3, 4]. While Cornelis et al do not provide any statistical data on the success of their algorithm, they do claim that in their restoration of the digitized *Ghent Altarpiece*, the multiscale top-hat transform successfully omitted painting details (such as letters in a book) from the final crack map. These details would otherwise have been determined to be cracks by the classical implementation of the black and white top-hat transforms [3].

6.2 Alternative Method Results

In comparison, the alternative methodology implemented by Karianakis and Maragos produced significantly different results [8]. As previously discussed in Subsection 4.2, the combined edge and crack masks were eroded until a certain amount of edge information was lost. The efficiency of the method relied heavily on the amount of information loss that was permitted. In the case that only 1% of edge information was discarded, the algorithm resulted in a recall of about 0.932, but a precision of only 0.497. As more edge information is lost, the recall decreases while the precision increases. For example, when 70% of the edge information is lost, the algorithm reported a recall of only 0.53, but an improved precision of 0.704 [8].

The statistics reported are summarized in Table 1. While the true positive rate, or recall, recorded by Karianakis and Maragos is proportional to that of the top-hat transform when the percentage of edge information lost is low, the alternative very quickly losses any comparability as the percentage of edge information lost increases. And while the precision does increase as edge information is lost, even the highest precision demonstrated by the alternative does not

Method	**Classification**	***tp***	***fn***	***tn***	***fp***	***TP* (or *R*)**	***FP***	***P***
Top-Hat Transform	Crack Thickness - Thin	220	30	230	20	0.880	0.080	0.917
	Crack Thickness - Medium	232	18	231	19	0.928	0.076	0.924
	Crack Thickness - Thick	235	15	238	12	0.940	0.048	0.951
	Number of Cracks - Few	242	8	245	5	0.968	**0.020**	0.980
	Number of Cracks - Medium	245	5	241	9	**0.980**	0.036	0.965
	Number of Cracks - Many	243	7	243	7	0.972	0.028	0.972
	Crack Connectivity - Low	215	35	219	31	**0.860**	**0.124**	0.874
	Crack Connectivity - High	218	32	221	29	0.872	0.116	0.883
Alternative Method	Edge Information Lost - 1%	-	-	-	-	**0.932**	-	**0.497**
	Edge Information Lost - 30%	-	-	-	-	0.857	-	0.594
	Edge Information Lost - 70%	-	-	-	-	**0.530**	-	**0.704**

Table 1: Crack Detection Results by Methodology: Results Specified in Text Emphasized, Modified from [4, 8].

compare with the precision of the top-hat transform. It is therefore reasonable to conclude that the top-hat transform, in both its classical and multiscale forms, significantly outperforms the alternative implemented by Karianakis and Maragos.

7. CONCLUSIONS

This paper has examined digital art restoration by breaking this process down into three primary components: edge detection, crack detection, and crack removal (inpainting), with a specific focus on the crack detection phase as completed using morphological operations.

While it has been demonstrated that there are many and varied effective methods of crack detection and removal, cracks are only one form of defect that may be encountered in the digital art restoration process. Areas for further study include the detection and removal of dust, scratches, fading, and missing sections from digitalized works of art [10].

8. ACKNOWLEDGMENTS

Thanks to Peter Dolan for his input, help, and advice.

9. REFERENCES

[1] Morphological image processing. Available at *https://www.cs.auckland.ac.nz/courses/compsci773s1c /lectures/ImageProcessing-html/topic4.htm.*

[2] J. Canny. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-8(6):679–698, Nov 1986.

[3] B. Cornelis, T. Ružić, E. Gezels, A. Dooms, A. Pižurica, L. Platiša, J. Cornelis, M. Martens, M. D. Mey, and I. Daubechies. Crack detection and inpainting for virtual restoration of paintings: The case of the ghent altarpiece. *Signal Processing*, 93(3):605 – 619, 2013. Image Processing for Digital Art Work.

[4] S. Desai, K. Horadi, P. Navaneet, B. Niriksha, and V. Siddeshvar. Detection and removal of cracks from digitized paintings and images by user intervention. In *Advanced Computing, Networking and Security (ADCONS), 2013 2nd International Conference on*, pages 51–55, Dec 2013.

[5] N. Efford. *Digital image processing: a practical introduction using Java.* Addison-Wesley, 2000.

[6] B. Green. Canny edge detection tutorial. Available at *http://dasl.mem.drexel.edu/alumni/bGreen /www.pages.drexel.edu/_weg22/can_tut.html.*

[7] R. Haralick, S. R. Sternberg, and X. Zhuang. Image analysis using mathematical morphology. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-9(4):532–550, July 1987.

[8] N. Karianakis and P. Maragos. An integrated system for digital restoration of prehistoric theran wall paintings. In *Digital Signal Processing (DSP), 2013 18th International Conference on*, pages 1–6, July 2013.

[9] A. W. R. Fisher, S. Perkins and E. Wolfart. Morphology. Available at *http://homepages.inf.ed.ac.uk/rbf/HIPR2/morops.htm.*

[10] G. S. Spagnolo and F. Somma. Virtual restoration of cracks in digitized image of paintings. *Journal of Physics: Conference Series*, 249(1):012059, 2010.