2. Literature Study

2.1. Introduction

This chapter contains the results of the literature study conducted during this study. The image processing concepts involved in the binarizing of the cosmic ray recordings are discussed. The literature study starts with a discussion of the tasks involved during image processing as well as the basic methods applied to images. This is followed by a study of adaptive document binarization and the most popular adaptive document binarization methods used in the past four decades. The chapter concludes with a summary of relevant research concerning the binarization and data extraction from legacy ionization chamber cosmic ray recordings.

Document image processing is used to convert information contained in printed documents to a digital format by processing and interpreting the content computationally (Jain *et al.*, 1995). According to Ejiri *et al.* (1984), the primary goal of document image processing is automation.

The field of document image processing consists of practices such as (Doermann *et al.*, 2003):

- Pattern recognition: Pattern recognition aims to classify data (patterns) based on either a priori knowledge or on statistical information extracted from the patterns;
- Graphic analysis: The study of interdependent phenomena by analyzing graphical representations;
- Digital image forensics: Consists of source device identification and semantic modification detection;

When a document processing method is designed, it is aimed at a specific set of documents or a class of documents (Aiello, 2002). Document knowledge is applied to ensure that the image is processed correctly. Cesarini *et al*. (1999) define two classes of document knowledge: generic and specific knowledge. However, during this study the Nagy *et al*. (1988) definition of three levels of knowledge will be used. These levels are generic knowledge, class-specific knowledge and publication specific knowledge.

Generic document knowledge is common to a broad class of documents. Examples of document-generic knowledge include:

- The first piece of text in the largest font in a document is probably the title;
- The text following this first text is in a slightly smaller font, but still larger than the average font size and is probably the subtitle;
- The text directly below a graphic image that is in a different font from the majority of the text, is probably a caption;

Class-specific document knowledge applies to certain types of documents such as newspapers or certain standardized forms. Examples of class-specific knowledge for newspaper type documents include:

- The largest piece of text at the top of the first page is the name of the newspaper;
- The pieces of text directly below the newspaper name contains the date and possibly the volume number, price and/or website address;
- Articles are written in column format and text in a larger font indicates the end of one article and the beginning of another;
- The small text written below the article title is probably the name of the author;

Publication-specific relates to a document that is always printed to a specific standard. This knowledge resembles class-specific knowledge, but is much more accurate, for instance:

- The largest piece of text at the top of the first page written in the Times New Roman font, size 80, is the name of the newspaper;

- The pieces of text directly below the newspaper name contains the date and price of the newspaper, written in the font Times New Roman of size 16;
- All article text is written using the Arial font with the text size set at 12;

Using this knowledge, a document processing method can be designed to search for and process specific pieces of a document and each part of the method can be designed to accurately process only the specific document object at which it is targeted.

Document processing consists of three phases, namely:

- Image capture and conversion to digital format;
- Image enhancement;
- Image interpretation;

2.2. Image capture

Documents printed on paper need to be digitized before they can be processed. This is usually achieved by scanning the original document. There are cases in which a digital camera is used instead of a scanner. The key difference between these two methods is that a scanner converts a document page to a digital format by scanning it one pixel-high line at a time. A digital camera captures an entire scene at once. The preferred method of image capture depends on the document being digitized as well as the intended use of the resulting document image.

From here on the document image to be used will be referred to simply as the image.

The data used during this study was obtained by scanning the original cosmic ray photographic recording strips using a Epson Perfection 4490 scanner.

2.2.1. Preparation of the document

The Records Management Services located in the state of Michigan, USA has established several standards which proper document capturing practices should adhere to (Michigan, 2009).

These practices include:

- Removal of any staples, clips or any other bindings from the document;
- Removal of any folds or creases in the document to reduce faulty scanning;
- Identification of important or vital information in the document before scanning;
- Listing of any missing documents;
- Arrange the documents in such a way that relevant groups of documents are scanned sequentially and thus keeping relevant data sequential;

The images used in this study comply with these standards. Each photographic strip has the same basic layout and the entire image may contain vital data, thus identification of specific areas of importance in different documents is not necessary in this case. Effective organization of these stacks of photographic paper from the 1940s greatly eases its use today.

2.2.2. Balance between quality and resolution

As the resolution of the scanned image increases, the storage space that is required increases and the detail and quality of the image improves. The processing time of image processing methods applied to the image also increases along with the resolution.

The scanning resolution chosen must be large enough to render small important details in the scanned image clearly visible, while still being small enough to keep processing time and required storage space to a minimum. Compression may be used to reduce required storage space, as long as the compression is reversible without loss of data.

The images used in this study were scanned at 600 dpi (dots per inch) and saved as an 8-bit image (each pixel in the image has a value between 0 and 255).

2.3. Image improvement

Once the document has been digitized, image processing methods are applied to improve the visual quality of the image. This phase also removes imperfections that may have been caused by the scanning process or have been present in the image from the start. Such imperfections may be especially abundant in historical and/or heavily degraded documents.

These imperfections include:

- Noise: Random variations of the intensity level or color of certain pixels in the image. Noise can make an image appear grainy and can reduce the accuracy of certain image processing methods;
- Low resolution: Caused by lossy compression or the image capture method;
- Non-uniform lighting: Appears when the document does not lie flat on the scanning surface. This has a pronounced effect on the background intensity of the image, which can make differentiation between the foreground and background difficult;
- Perspective disturbance: Also appears when the document does not lie flat on the scanning surface;
- Incorrect color and intensity quantification: Some image capturing devices do not recognize the actual color/intensity value of certain pixels in a document, assigning these pixels an incorrect value in the digitized image;

These imperfections are removed by applying a set of image processing methods to the image. The methods used depend on the type of image, the imperfections present and what the improved image will be used for. The optimal set of methods and the relevant parameters are usually derived by experimentation.

The following two subsections will describe different image processing methods that may be applied to an image in either the spatial or frequency domain. Specific attention has been paid to methods that may be useful when applied to images of cosmic ray recordings.

During the remainder of this section when the application of a method is discussed, it will be applied to either a grayscale image with pixel intensity values ranging from 0 (Black) to 255 (White) or a binary image in which pixels have a value of either 0 (White) or 1 (Black).

2.3.1. Spatial domain

The spatial domain refers to an image that is displayed and manipulated as a two-dimensional array, where the value at each position in the array is the intensity value of the pixel corresponding to that position. In 8-bit images, this value ranges between 0 and 255.

In a color image encoded in RGB or CMYK, the spatial representation of the image is a three-dimensional array. Each level of the array represents a group of colors within the image. In an RGB image, for example, one of the three two-dimensional arrays within the three-dimensional array will represent only R, meaning that specific two-dimensional array contains the intensity values of every pixel in the image that uses a red component to make up its final color value in the complete image (Gonzalez & Woods, 2008, 105).

Several mathematical functions exist that can be used to manipulate the values of these array cells to improve or manipulate an image. A function is applied to each pixel in the array.

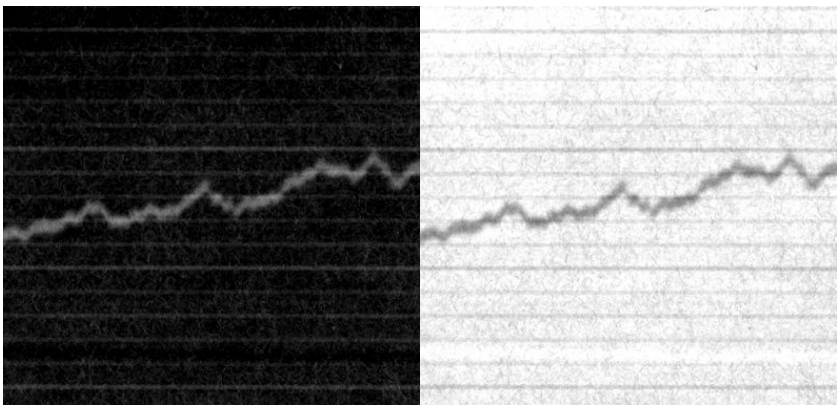


Figure 2.1.: An image (left) and its negative version (right).

2.3.1.1. Intensity transformation functions

An example of an intensity transformation function is used to calculate the negative of an image:

$$s = L - 1 - r, \tag{2.1}$$

where L represents the number of possible intensity values in the image (256 in an 8-bit image, ranging from 0 to 255), r is the intensity value of the current pixel and s is the new negative value of the current pixel.

The log transformation can be applied to an image to map a narrow range of low intensity input values into a wider range of output values, thus increasing the contrast of the image. The inverse of the log transformation will reduce contrast in an image. The log transformation is written as:

$$s = c\log(1 + r), \tag{2.2}$$

where c is a constant usually set to 1.

There are also Power-Law or Gamma transformations that can be applied to an image, such as

$$s = cr^{\gamma}, \tag{2.3}$$

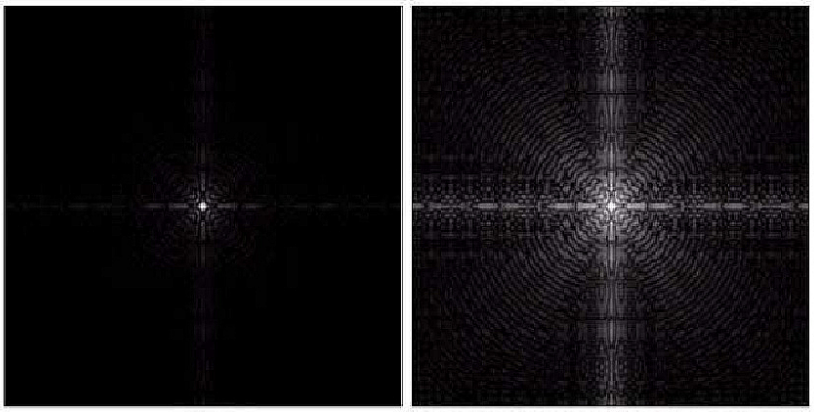


Figure 2.2.: Application of the log transform to an image (Gonzalez & Woods, 2008).

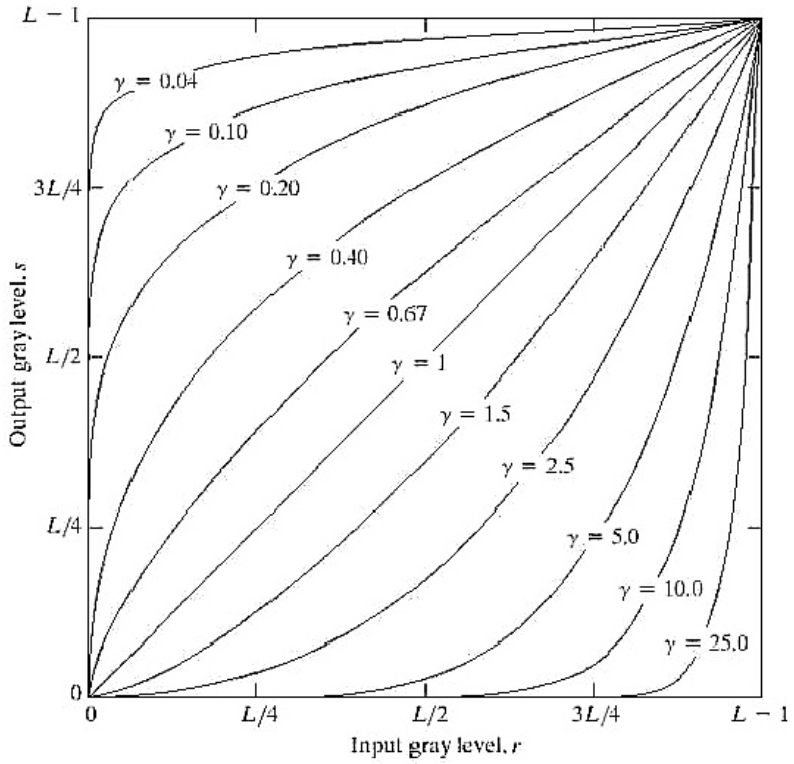


Figure 2.3.: Possible transformation curves of the Gamma transformation (Gonzalez & Woods, 2008).

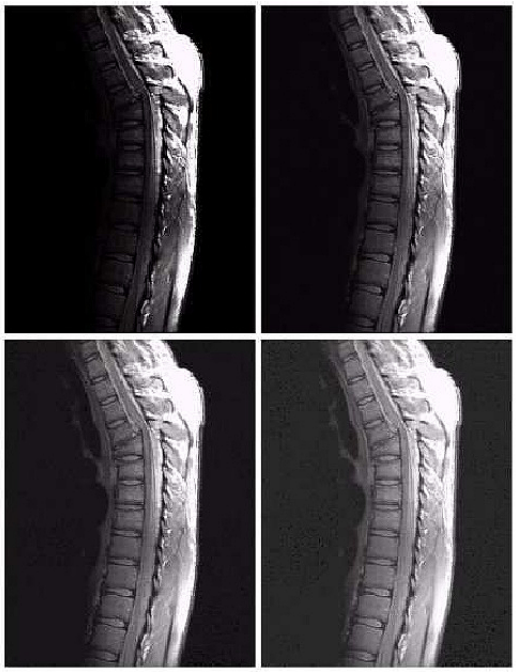


Figure 2.4.: Application of Gamma transformations (Gonzalez & Woods, 2008).

where c and γ are both positive constants. By varying γ, a range of transformation curves are obtained, as can be seen in Figure 2.3. Figure 2.3 shows clearly that curves generated with $\gamma < 1$ have the opposite effect on the image as those generated by $\gamma > 1$. It is also clear that Eq.2.3 becomes the identity transformation when $\gamma = c = 1$, which then only returns the original unchanged image as output (Gonzalez & Woods, 2008, 108 - 113). Examples of the application of Gamma transformations are shown in Figure 2.4.

These intensity level transformations can change the visual quality of an image by emphasizing different aspects of it or making important aspects within an image more visible. Thus it is important to experiment with these filters to identify which transformation with which variables will have the best effect on the image in question. The range of these variables makes these transforms very customizable and several transforms may also be combined to achieve the desired results.

2.3.1.2. Spatial Filtering

When using the above mentioned transformation functions, every pixel is manipulated separately. When spatial filtering is applied, resulting pixel values are influenced by the values of their neighboring pixels.

These filters, also known as masks, consist of a matrix of coefficients that is moved across the entire image so each pixel $f(x, y)$ is represented by the center of the matrix at one point. This pixel is then assigned a new value according to the function applied to its neighboring pixels. Linear spatial filtering applies the function coefficients of each cell of the mask to the values of each pixel falling within the filter and then summing the results to calculate the result and new value of the center pixel (Gonzalez & Woods, 2008, 144 - 166) .

The linear spatial filtering of an $M \times N$ sized image f by applying a $m \times n$ sized filter w is written as:

$$g(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t) f(x + s, y + t), \tag{2.4}$$

where $a = (m - 1)/2$ and $b = (n - 1)/2$ to enable filters with uneven sizes.

The filter is applied to every pixel in the image by having $x = 0, 1, 2, ..., M - 1$ and $y = 0, 1, 2, ..., N - 1$. Figure 2.5 shows an example of such a filter and how it is applied to an image.

When such a filter is applied to the outer rim of an image, some of the mask cells will fall outside of the image and return null values. This is avoided by zero padding the original image and applying the filter only to pixels $x = 1, 2, 3, ..., M - 2$ and $y = 1, 2, 3, ..., N - 2$, so the filter falls entirely within the image at all times.

Usually the resulting image values are read into a new blank image of the same size as the original. This is done to ensure that all input values to the filter of every pixel are the original values and not values that have just been altered by the filter.

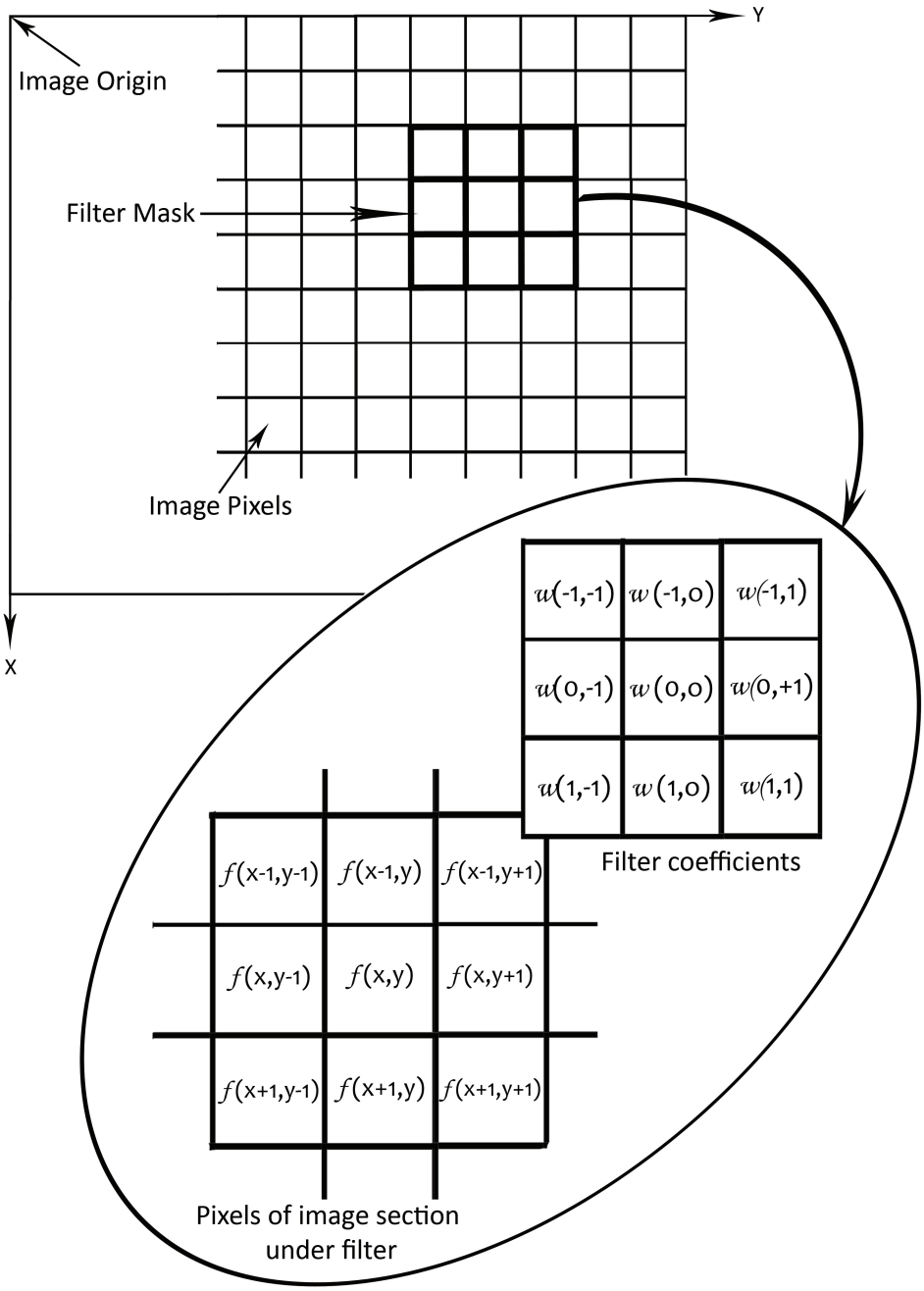


Figure 2.5.: Application of a filter mask to an image (Gonzalez & Woods, 2008).

$$\frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Figure 2.6.: An example of a 3 x 3 smoothing filter.

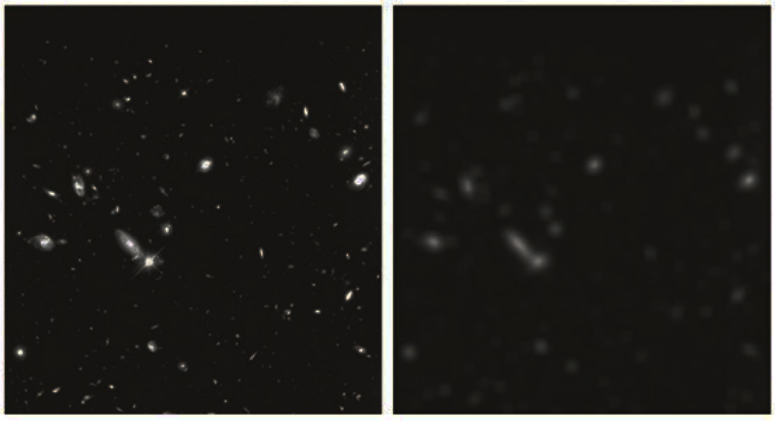


Figure 2.7.: Application of a 15 x 15 smoothing filter.

2.3.1.2.1. Smoothing spatial filters

Smoothing filters calculate new values for each pixel by calculating the average value of the pixels within the filter. Application of this filter removes large differences in intensity between adjacent pixels. This effect becomes more pronounced as the size of the filter increases.

This filter is usually applied to remove noise, but may also be used to smooth the image to either remove certain fine details or enlarge other details by spreading out the edges and making those details more noticeable/detectable.

A 3 by 3 pixel smoothing filter is shown in Figure 2.6 along with an example of its application in Figure 2.7.

These filters can also be weighted such as the filter shown in Figure 2.8. While the value of a non-weighted filter is simply calculated by dividing the sum of the values within the filter by the amount of values, the value of a weighted filter

$$\frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Figure 2.8.: Example of a weighted 3 x 3 smoothing filter.

can be manipulated by adding coefficients to certain original values and then dividing by the sum of the coefficients.

Figure 2.8 is an example of a weighted smoothing filter that causes less blurring than a non-weighted smoothing filter.

2.3.1.2.2. Order-statistic filters

Order-statistic filters are non-linear filters which calculate the resulting pixel value according to specific aspects of the image region contained within the mask. Some examples of order-statistic filters include:

- Median filter: The resulting pixel value is the median value of the region. The median filter is the most commonly used order-statistic filter. It removes salt-and-pepper noise much more effectively than an average filter by removing the noise without blurring the image, as can be seen in Figure 2.9;
- Maximum filter: The resulting pixel value is the highest intensity value of the region;
- Minimum filter: The resulting pixel value is the lowest intensity value of the region;

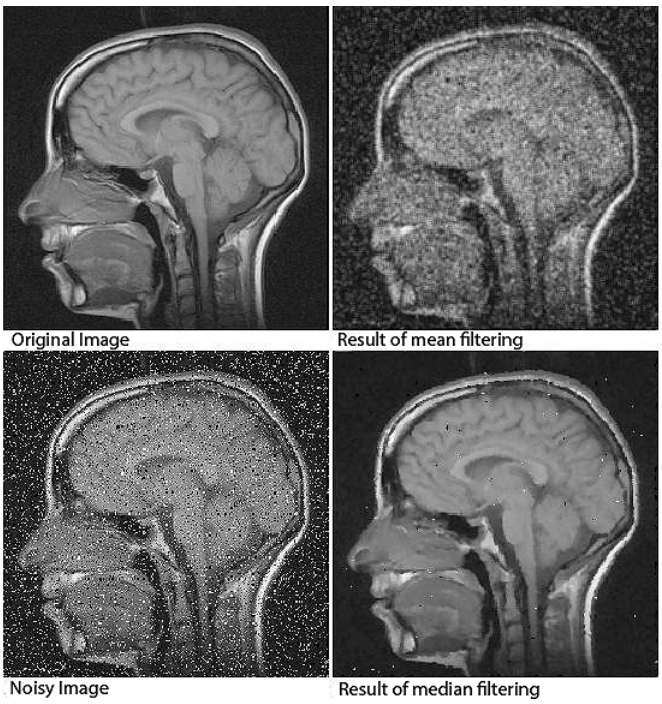


Figure 2.9.: Difference between median filtering and mean filtering (Gonzalez & Woods, 2008).

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

Figure 2.10.: Sobel filters.

2.3.1.2.3. Sharpening filters

Sharpening filters highlight transitions in intensity between pixels, more clearly defining fine details in an image. These filters make use of the difference between pixels in an area or the derivative of those pixels.

The first derivative is used in filters that sharpen the image in a specific direction. These filters include Sobel operators (Figure 2.10, (Sobel, 1970)), Prewitt operators (Figure 2.11, (Prewitt, 1970)) and Roberts Cross gradient operators (Figure 2.12, (Roberts, 1963)).

-1	-1	-1
0	0	0
1	1	1

-1	0	1
-1	0	1
-1	0	1

Figure 2.11.: Prewitt filters.

1	0
0	-1

0	1
-1	0

Figure 2.12.: Roberts filters.

The second derivative is used in Laplacian filters (Figure 2.13) to sharpen an image in all directions.

When a Laplacian filtered image is added to the original image, the transitions between pixels are defined more clearly and the image appears sharpened. The Laplacian image may be weighted to vary its effect, as can be seen in:

$$g(x, y) = f(x, y) + k(\nabla^2 f(x, y)), \tag{2.5}$$

where g represents the sharpened image, f the original image and k the weight that is applied to the Laplacian image.

-1	-1	-1
-1	9	-1
-1	-1	-1

Figure 2.13.: Laplacian filter.

Figure 2.14.: Application of a sharpening mask using Sobel operators (Gonzalez & Woods, 2008).

Figure 2.15.: Application of a sharpening mask using Laplacian operators (Gonzalez & Woods, 2008).

The difference between creating a sharpening mask using limited directional sharpening and creating a sharpening mask using full directional sharpening is shown in Figure 2.14 and Figure 2.15.

2.3.1.2.4. Unsharp masking and highboost filtering

Unsharp masking sharpens an image by creating a mask the size of the original image by subtracting a blurred version of the original from the original and then subtracting that mask from the original. The process is shown by:

$$f_s(x,y) = f(x,y) - \bar{f}(x,y), \tag{2.6}$$

where $f(x,y)$ **represents the original image,** $\bar{f}(x,y)$ **the blurred version of the original image and** $f_s(x,y)$ **the resulting sharpened image which can be seen in Figure 2.16.**

Figure 2.16.: Results of unsharp masking (Gonzalez & Woods, 2008).

0	-1	0
-1	A+4	-1
0	-1	0

Figure 2.17.: Highboost Filter.

Highboost filtering is an adaptation of unsharp masking as seen in:

$$f_{hb}(x,y) = Af(x,y) - \bar{f}(x,y), \tag{2.7}$$

where $f_{hb}(x,y)$ is the sharpened resulting image and $A \geq 1$. If $A = 1$ then the result is a Laplacian filtered image. The sharpening effect decreases as A increases. A side effect of highboost filtering is an increase in intensity values throughout the image, making highboost filtering ideal for sharpening darker images.

An example of a highboost filter is shown in Figure 2.17.

2.3.2. Morphological image processing

Morphology makes use of form and structure, thus morphological image processing is used to extract image components which possess a certain form and structure (Mak *et al.*, 2009). Morphological image processing makes use of set theory, in other words, it extracts certain sets of pixels that represent different objects within the image (Gonzalez & Woods, 2008, 628).

As in many spatial filters, the results of morphological image processing are written to new blank images as the original image is being processed. This is to avoid the results of already processed pixels influencing the results of pixels yet to be processed.

2.3.2.1. Set Theory

In order to explain morphological image processing methods, a brief explanation of basic set theory is given which Figure 2.18 summarizes (Serra, 1988).

Let A be a set composed of ordered pairs of real numbers. If a is an element within the set of A then it is written as:

$$a \in A. \tag{2.8}$$

If a is not a part of the set of A then it is written as:

$$a \notin A. \tag{2.9}$$

An empty set is represented by the symbol $\oslash$. A set is represented as the contents between two braces $\{\cdot\}$. These contents are individually referred to as elements.

If we have set A and set B and every element of set A can be found within set B, then A is a subset of B, denoted as:

$$A \subseteq B. \tag{2.10}$$

The collection of elements found within either set A or set B or both, is referred to as the union C of A and B and is written as:

$$C = A \cup B \tag{2.11}$$

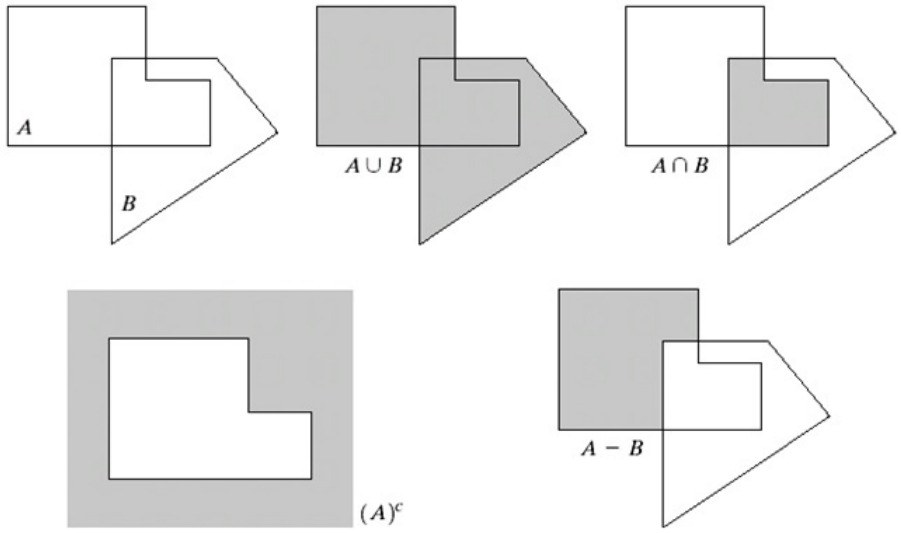


Figure 2.18.: Set theory operators (Gonzalez & Woods, 2008).

The collection of elements found within both set A and set B referred to as the intersection C of A and B and is written as:

$$C = A \cap B \tag{2.12}$$

If there are no elements which can be found in both sets A and B, then sets A and B are said to be disjoint or mutually exclusive, denoted by:

$$A \cap B = \oslash \tag{2.13}$$

The complement of set A contains all elements which are not found within A and is written as A^c.

The difference between two sets A and B, denoted $A - B$, is written as:

$$A - B = A \cap B^c \tag{2.14}$$

If set B represents a set of two-dimensional pixels, then the reflection of B is represented by $\hat{B}$, where the coordinates of all pixels in B, (x, y), have been

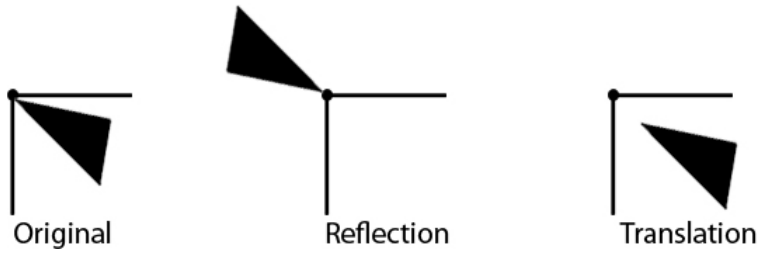


Figure 2.19.: Translation and reflection (Gonzalez & Woods, 2008).

replaced by the coordinates $(-x, -y)$. Reflection of B is written as:

$$\hat{B} = \{w|w = -b, \text{ for } b \in B\} \tag{2.15}$$

If set B represents a set of two-dimensional pixels, then the translation of B is represented by $(B)_z$, where the coordinates of all pixels in B, (x, y), have been shifted by a fixed amount in a specific direction as represented by $(x + z_1, y + z_2)$. Translation of B is written as:

$$(B)_z = \{c|c = b + z, \text{ for } b \in B\} \tag{2.16}$$

2.3.2.2. Erosion

The concepts mentioned above are applied in a range of morphological image processing methods (Serra, 1988).

Erosion is used to shrink image objects or make them thinner. Erosion is specifically applied to separate connected components that are wrongly barely connected. Erosion is also effective at removing noise from a binary image.

With A and B as sets in Z^2, the erosion of A by B is defined as:

$$A \ominus B = \{z|(B)_z \subseteq A\}, \tag{2.17}$$

where A represents a set of pixels that is eroded by the structuring element B. The erosion of A by B is the set of all points z such that B, translated by z, is contained in A.

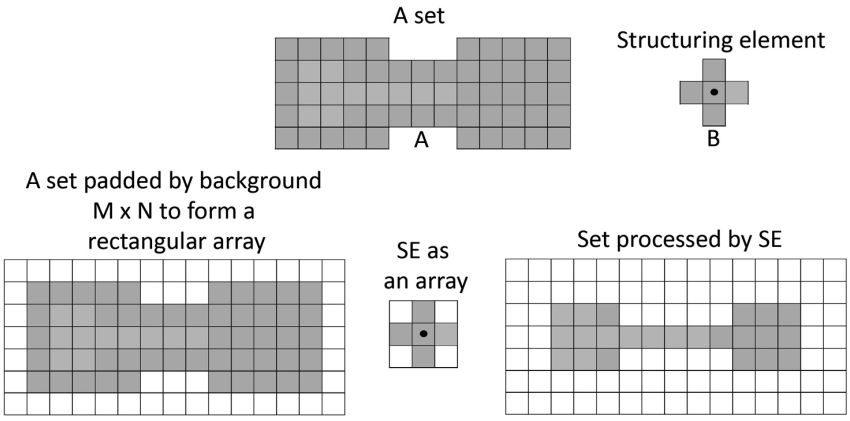


Figure 2.20.: The process of morphological erosion by a 3x3 structuring element (Gonzalez & Woods, 2008).

The center pixel of B is moved across every pixel of the image. In a binary image, when the center pixel of the structuring element B is occupied by a black (1) pixel which forms part of A, the values of all other pixels of the image that are occupied by non-center pixels of the structuring mask are checked to determine if these are white (0) or black (1). If any of these pixels are white (0), then the value of the pixel within A, which is currently occupied by the center pixel of B, is assigned the value of 0 (white).

In other words, if the structuring element B is not completely contained within A, the pixel within A on which B is centered is eroded, as can be seen in Figure 2.20.

2.3.2.3. Dilation

Dilation is the opposite of erosion and is used to fill gaps in image objects by expanding or thickening certain elements of that object (Heidenreich *et al.*, 2009). Broken or distorted characters on a digital document are fixed by dilation as shown in Figure 2.21. Dilation can also be used to make certain fine details of an image more visible, without necessarily losing the detail.

With A and B as sets in Z^2, the dilation of A by B is defined as:

$$A \oplus B = \{z|(\hat{B})_z \cap A \neq \oslash\}, \tag{2.18}$$

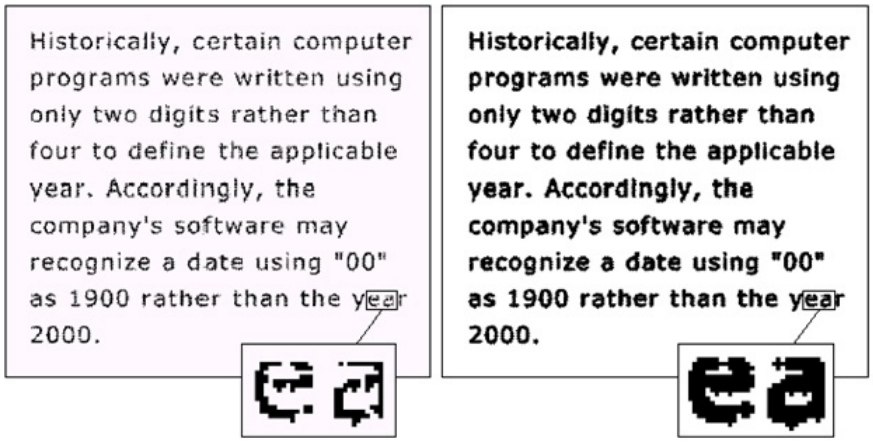


Figure 2.21.: Dilation of text to improve visibility (Gonzalez & Woods, 2008).

where A represents a set of pixels that is dilated by the structuring element B. The dilation of A by B is the set of all displacements z such that $\hat{B}$ and A overlap by at least one element.

The center pixel of B is moved across every pixel of the image. In a binary image, when the center pixel of the structuring element B is occupied by a black (1) pixel which forms part of set A, the values of all other pixels of the image that are occupied by non-center pixels of the structuring mask are changed to black (1). If some of these pixels are already black, then their values remain unchanged.

2.3.2.4. Boundary Extraction

Boundary extraction is used to identify and extract the outline of objects in a binary image. Using set theory, the method is written as:

$$\beta(A) = A - (A \ominus B), \tag{2.19}$$

where the boundary of A is extracted by identifying the difference between A and an eroded version of A, as shown in Figure 2.22.

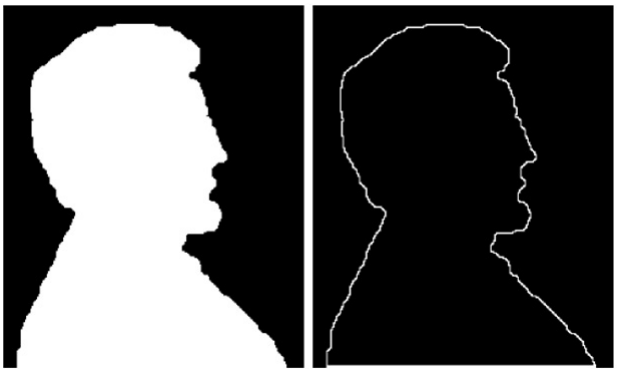


Figure 2.22.: Boundary extraction (Gonzalez & Woods, 2008).

2.3.2.5. Extraction of connected components

Extraction of connected components is used to identify and extract objects in an image that share 4-connected or 8-connected neighbors.

Let Y be a connected component within set A and assume that a pixel p is known to be a black pixel that is part of Y. All the pixels/elements of Y can be identified by:

$$X_k = (X_{k-1} \oplus B) \cap A \quad k = 1, 2, 3... \tag{2.20}$$

where $X_0 = p$ and B is a suitable structuring element. The algorithm terminates when $X_k = X_{k-1}$ and then $Y = X_k$.

With each iteration of the algorithm the structuring element checks the neighborhood around each pixel already part of Y. If any black (1) pixels fall within the structuring element then these are added to the connected component set Y.

The connected components are usually labeled so they may be individually manipulated at a later stage. An example of connected component extraction and labeling is shown in Figures 2.23 and 2.24. In this case every pixel in each connected component is assigned the value of the label of that component, where the labels range from 1 to the number of components identified.

Figure 2.23.: Extraction and labeling of connected components (Gonzalez & Woods, 2008).

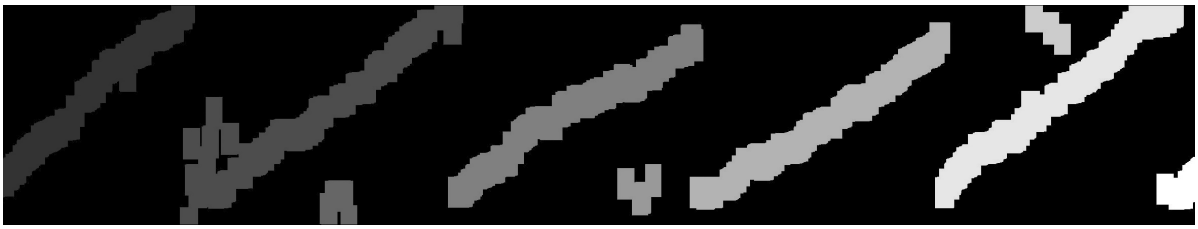


Figure 2.24.: A labeled set of connected components representing cosmic ray data.

2.3.2.6. Segmentation using morphological watersheds

Watershed transformation is a powerful tool for image segmentation (Beucher & Meyer, 1993). The watershed model works by treating the image like a virtual three-dimensional landscape, two spatial coordinates versus intensity (height). The lower intensity values represent valleys or holes, while the higher intensity values represent peaks in the three-dimensional landscape, as can be seen in Figure 2.25, which shows a segment of a cosmic ray data line as a three-dimensional landscape. The original image is inverted so the areas of interest (data lines) will be represented by the valleys in the topographical image, as they would be when applying this process to a normal text image.

This topographic representation of an image contains three types of points:

1. Points belonging to a regional minimum;
2. Points which would direct a drop of water, if placed at that point, to a single minimum;
3. Points which would direct a drop of water, if placed at that point, to possibly more

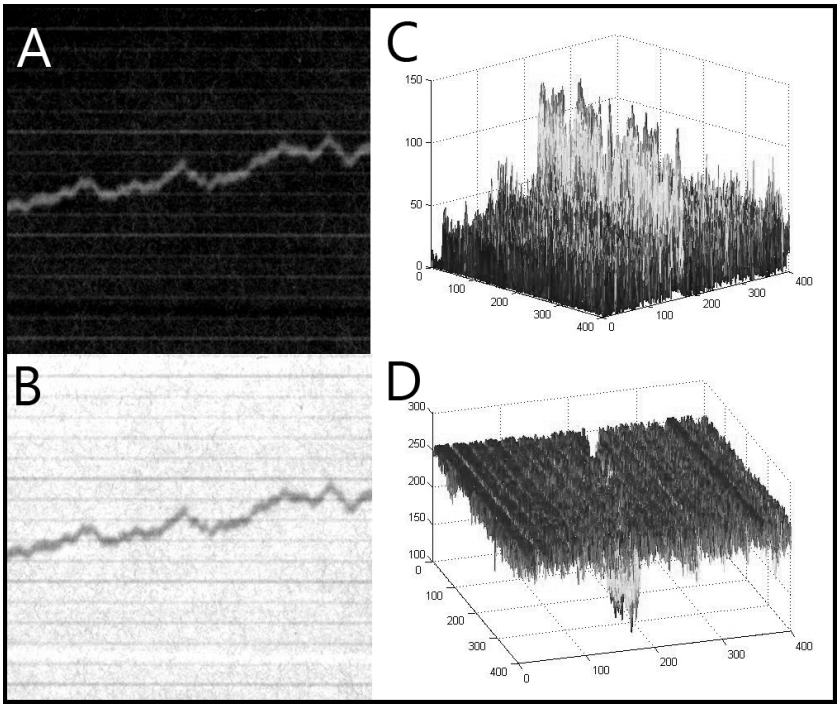


Figure 2.25.: Three-dimensional representation of a cosmic ray data line segment. (A) Original segment. (B) Topographical representation of A. (C) Inverted version of A. (D) Topographical representation of C.

than one minimum. Each minimum having an equal chance to receives the drop;

The set of points around a specific minimum, which satisfies condition 2, is called the watershed of that minimum. The set of points which satisfies condition 3, forms crest lines on the topographic surface and are called watershed lines. The main objective of this segmentation method is to find these watershed lines, as they define the boundaries between objects.

The basic idea behind this process is that the topographic landscape is flooded from within each minimum. Once a watershed/valley becomes fully flooded and its contents start to spill over to an adjacent watershed, a dam is constructed to keep the contents of the two watersheds separate. When the entire image is fully flooded, only the tops of the constructed dams will be visible. The lines formed by the tops of the dams are the watershed lines of the image.

The technical explanation behind this method is that each minimal component is dilated continuously while the image is flooded. If two components grow to connect to each other during a dilation, then all the intersecting pixels of that dilation are labeled as dam/watershed pixels. This process continues until the set of watershed lines of the image is complete.

The watershed segmentation process is summarized in Figure 2.26.

2.3.3. Frequency Domain

Complicated periodic functions can be written as the sum of simple waves mathematically represented by sines and cosines (Taneja, 2008). In other words, a function $f(t)$ of a continuous variable t that is periodic with period T, can be expressed as the sum of sines and cosines multiplied by appropriate coefficients. This sum is known as the Fourier series (Gonzalez & Woods, 2008, 202 - 210). Through convolution theory a relationship between the spatial domain and the frequency domain can be defined, where convolution in the frequency domain is analogous to multiplication in the spatial domain, the two domains being related by the forward and inverse Fourier transforms, respectively (Gonzalez &

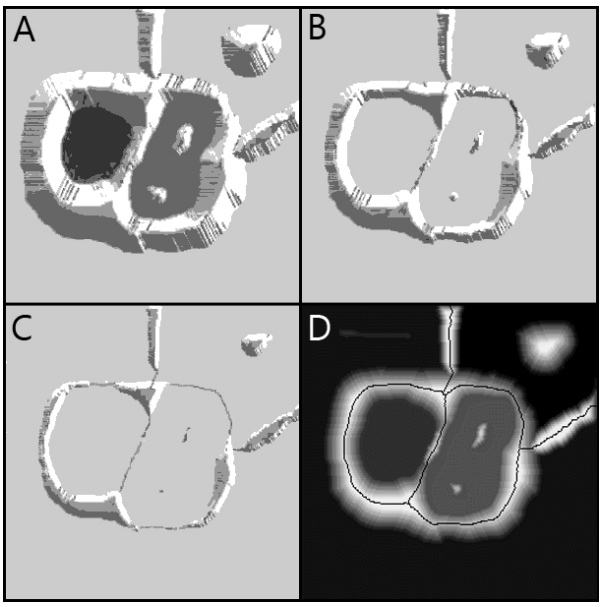


Figure 2.26.: Watershed segmentation process. (A) Topographic representation. (B) Initial stages of flooding. (C) Watershed lines begin to form. (D) Original image shown with a complete set of watershed lines (Gonzalez & Woods, 2008).

Woods, 2008, 235 - 236). Image improvement in the frequency domain is possible because of this relationship.

In order to filter an image in the frequency domain, it must first be transformed from a spatial domain image $f(x, y)$ to a frequency domain image $F(u, v)$, namely a Fourier spectrum. This is achieved by using the two dimensional Fourier transform 2.21. The image is then filtered by multiplying the Fourier spectrum $F(u, v)$ with a filter function $H(u, v)$. The results are then transformed back to the spatial domain using the inverse Fourier transform Eq. 2.22

$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi(\frac{ux}{M}+\frac{vy}{N})}. \tag{2.21}$$

$$f(x,y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) e^{j2\pi(\frac{ux}{M}+\frac{vy}{N})} \tag{2.22}$$

The process of filtering an image $f(x, y)$ of size $M \times N$ is explained in the following steps (Gonzalez & Woods, 2008, 263):

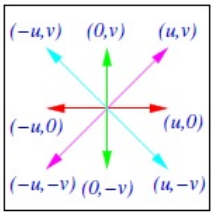


Figure 2.27.: Symmetry of a Fourier spectrum.

1. Create a padded image $f_p(x, y)$ of the input image $f(x, y)$ of size $P \times Q$ by appending the necessary number of zeros to $f(x, y)$. $P \geqq 2M - 1$ and $Q \geqq 2N - 1$. Typically, $P = 2M$ and $Q = 2N$;
2. Multiply $f_p(x, y)$ by $-1^{(x+y)}$ to center its transform;
3. Calculate the two-dimensional discrete Fourier transform using Eq. 2.21;
4. Calculate $G(x, y)$ by multiplying the Fourier spectrum $F(u, v)$ by the filter function $H(u, v)$;
5. After the filter has been applied, apply Eq. 2.23 to obtain the processed image $g_p(x, y)$;
$$g_p(x, y) = \mathscr{F}^{-1}[H(u, v)F(u, v)](-1^{(x+y)}) \tag{2.23}$$
6. Obtain the final filtered image $g(x, y)$ by extracting the $M \times N$ sized region from the upper left quadrant of $g_p(x, y)$;

The Fourier spectrum is symmetrical across four axes, when the input image is real-valued, as opposed to complex-valued. This can be seen in Figure 2.27 (Hossack, 2011). The spectrum has certain characteristics which may be removed or manipulated to improve the image in the spatial domain.

Frequencies indicate tempo of variation, which allows variations in intensity levels in the spatial domain to be represented by the frequencies of a Fourier spectrum. Frequencies that are positioned near the origin/center of the spectrum represent components in the original image with a slow tempo of variation, while frequencies that are positioned at a greater distance from the center/origin

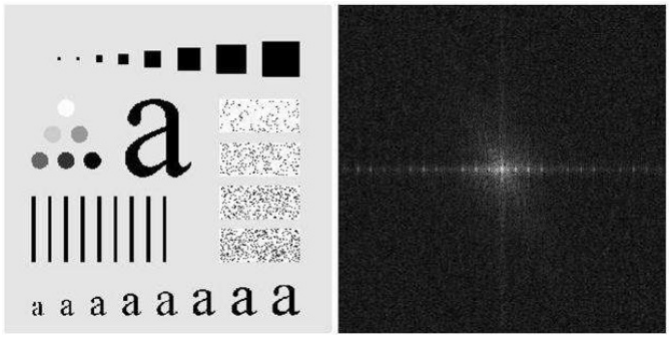


Figure 2.28.: Conversion of a spatial image to a Fourier spectrum (Gonzalez & Woods, 2008).

of the spectrum represent components with a faster tempo of variation (Gonzalez & Woods, 2008, 255 - 269). Lowpass and highpass filters make use of these characteristics to sharpen or smooth an image.

2.3.3.1. Lowpass Filters

The difference between a sharp image and a smooth image lies in the difference between intensity values of adjacent pixels. If this difference is substantial, then a noticeable edge occurs which clearly defines the objects which it separates, making those objects appear sharper. When a miniscule difference between adjacent pixels occurs, this defining edge is barely noticeable, causing the image to appear smooth or blurred. In other words, a high tempo of change in intensity causes sharpness and a low tempo of change in intensity causes smoothness. These tempos of change are targeted by lowpass and highpass filters.

An image is smoothed in the frequency domain by applying a lowpass filter that suppresses high frequencies that cause definition in the image while allowing low frequencies. Frequencies that fall outside a certain radius D_0 from the center of the spectrum are removed. D_0 is referred to as the cutoff frequency. The potency of the filter is regulated by varying D_0.

Some popular examples of lowpass filters are (Gonzalez & Woods, 2008, 269 - 276):

- Ideal lowpass filters: Passes, without attenuation, all frequencies within a circle of radius D_0 and cuts off all other frequencies as shown in Figure 2.29. The

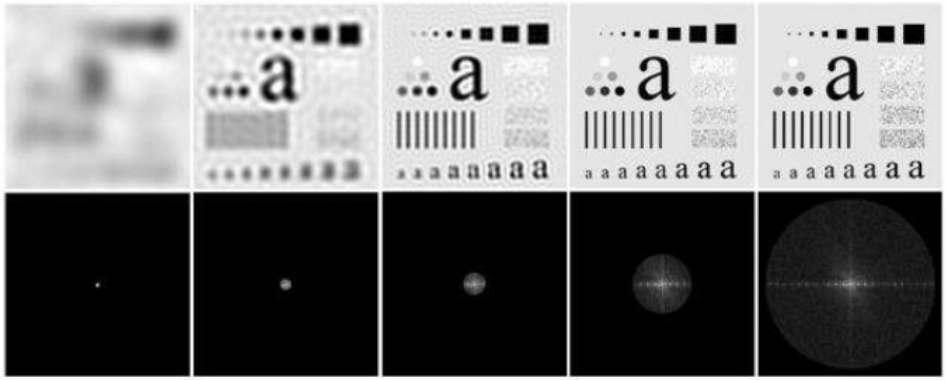


Figure 2.29.: Ideal lowpass filter with decreasing values of D_0 (Gonzalez & Woods, 2008).

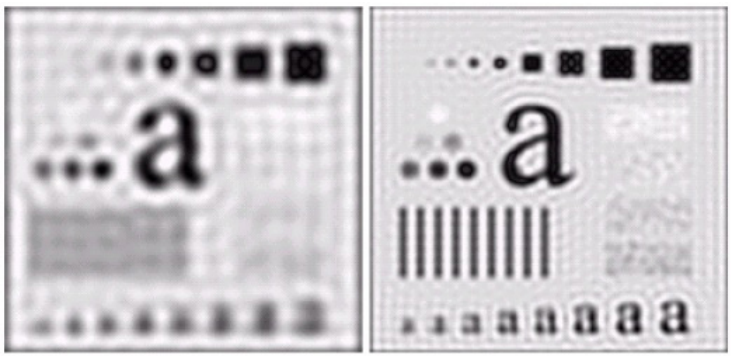


Figure 2.30.: Ringing in a filtered image (Gonzalez & Woods, 2008).

ringing effect makes this a less effective method for sharpening, as can be seen in Figure 2.30;

$$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \leq D_0 \\ 0 & \text{if } D(u,v) > D_0 \end{cases} \tag{2.24}$$

- Butterworth lowpass filters: Butterworth filters do not have sharp discontinuities which give clear cutoffs between passed and filtered frequencies. A Butterworth filter of order 1 ($n = 1$) causes no ringing in the spatial domain and almost no ringing with an order of 2. Significant ringing can occur at higher orders;

$$H(u,v) = \frac{1}{1 + [D(u,v)/D_0]^{2n}} \tag{2.25}$$

- Gaussian lowpass filters: Gaussian filters cause no ringing in the spatial domain

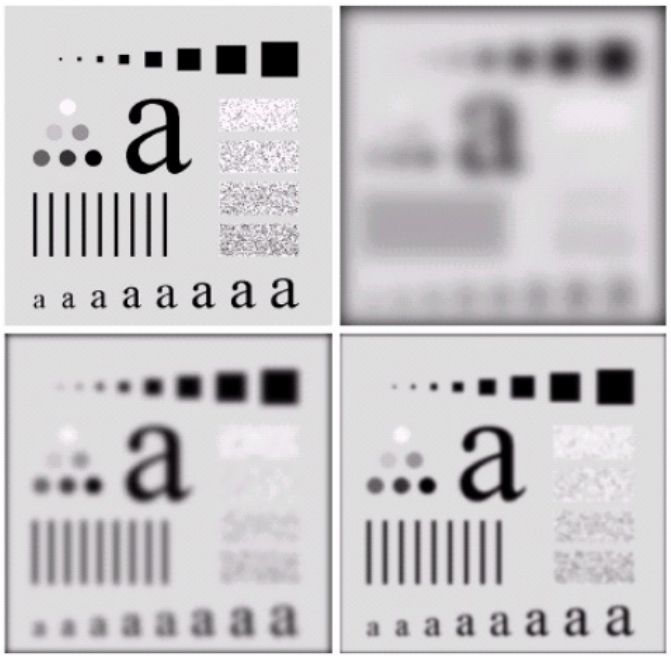


Figure 2.31.: Gaussian filter result with zero ringing (Gonzalez & Woods, 2008).

as can be seen in Figure 2.31. σ is a measure of spread about the center;

$$H(u,v) = e^{-D^2(u,v)/2\sigma^2} \tag{2.26}$$

2.3.3.2. Highpass filters

The filter mentioned above can be adapted to have the opposite effect of suppressing frequencies inside the radius D_0 and passing frequencies outside D_0, as shown in Figure 2.32.

A highpass filter is obtained from a lowpass filter using the following equation:

$$H_{hp}(u,v) = 1 - H_{lp}(u,v), \tag{2.27}$$

which converts the above mentioned equations to:

$$H(u,v) = \begin{cases} 0 & \text{if } D(u,v) \leq D_0 \\ 1 & \text{if } D(u,v) > D_0 \end{cases} \tag{2.28}$$

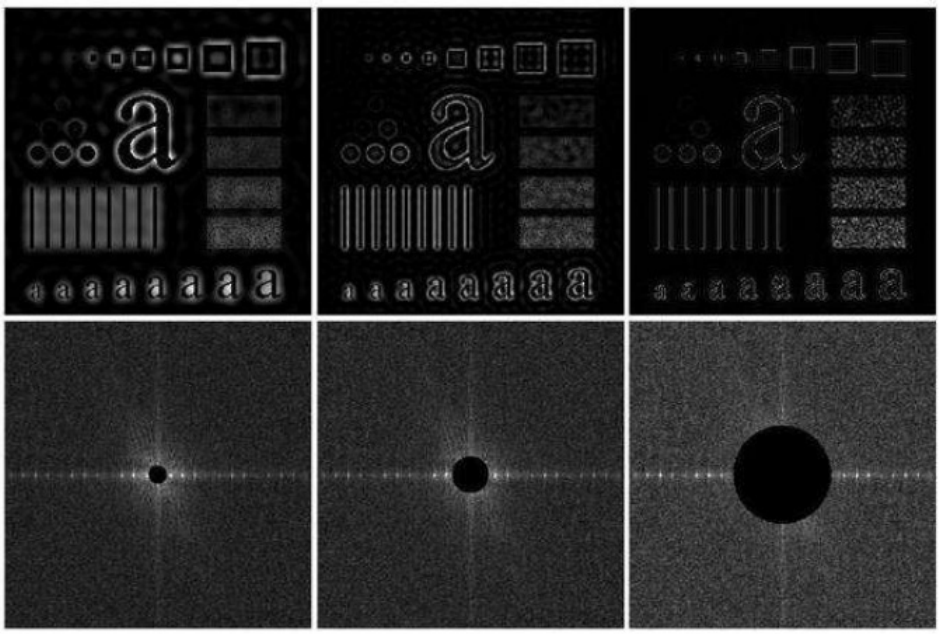


Figure 2.32.: Ideal highpass filter results (Gonzalez & Woods, 2008).

$$H(u,v) = \frac{1}{1 + [D_0/D(u,v)]^{2n}} \tag{2.29}$$

$$H(u,v) = 1 - e^{-D^2(u,v)/2\sigma^2} \tag{2.30}$$

2.3.3.3. Laplacian Filter

Image manipulation in the frequency domain is not limited to just smoothing and sharpening filters. A Laplacian filter may be modified to be applied in the frequency domain to achieve the same edge sharpening effect as it would in the spatial domain. As mentioned earlier, the choice of domain in which to filter the image depends on the image itself, as some images are filtered more effectively in the spatial domain, while others provide better results when filtered in the frequency domain.

The Laplacian filter function to be applied in the frequency domain may be written as (Paine & Lodwick, 1989) :

$$H(u,v) = -[(u - \frac{M}{2})^2 + (v - \frac{N}{2})^2]. \tag{2.31}$$

This filter function is used to calculate the Laplacian of the original image, which is then subtracted from the original image in order to create the sharpened image.

2.3.3.4. Homomorphic filtering

Homomorphic filtering differentiates between the illumination and reflective aspects of an image. High frequencies in the Fourier spectrum are associated with reflective aspects of the image and low frequencies are associated with the illumination aspects (Adelmann, 1998).

Homomorphic filtering first calculates the natural logarithm of the image, then applies the frequency domain filtering steps and finally calculates the exponent of the image, $e^{g(x,y)}$. One of the uses of homomorphic filtering is to enhance images with very low contrast between intensity values. The filter functions by suppressing the effect of low frequencies/illumination aspects $(Y_l < 1)$ and enhancing the effect of high frequencies/reflective aspects $(Y_h > 1)$. These concepts are usually applied through a highpass filter such as a Gaussian highpass filter:

$$H(u,v) = (Y_h - Y_l)[1 - e^{-c(D^2(u,v)/D_0^2)}] + Y_l, \tag{2.32}$$

where c is a constant that is varied to control the sharpness of the image.

2.4. Interpretation

Once the image is improved and its imperfections are removed, it can be interpreted. Interpretation of images involves the identification and/or measurement of various targets in an image in order to extract useful information from them (Canada, 2008).

Image interpretation can be split into two categories:

- Pattern recognition;
- Digitization;

2.4.1. Pattern recognition

Pattern recognition is the process where the image is analyzed to determine if any objects within it resemble (within an acceptable margin of error) any objects within a previously established database. If such a match is made then the object in the image can be identified as the same object which it has matched to in the database.

The most popular application of pattern recognition is optical character recognition (OCR). The image database is an alphabet of all characters in a specific font as well as numerical characters and written symbols (AIM, 2000). Each connected group of pixels in an image, which may represent a number, letter or symbol, is compared to all sub images in the database to see if a match occurs. If a set of pixel resembles the shape of a 'T', then that set is probably a 'T'.

2.4.2. Digitization

While pattern recognition seeks to identify and label image objects by matching them to objects of the same type, image digitization seeks to identify specific image objects and derive their numerical values according to their positions in the image or other visual characteristics. The primary goal of image digitization is the automation of visual data extraction (Niczyporuk & Miller, 1991).

2.5. Adaptive image binarization

The application of a collection of the techniques mentioned in this chapter thus far would enable one to extract data from cosmic ray graph images. However, this method would probably be configured to read a clear white line from a clear black background. The intensity levels of graph lines and the background of cosmic ray graph images vary greatly between images and different sections of the same image. Any data extraction method applied to these unpredictable images must be able to adapt to the changing intensity levels of the image.

Also, the actual intensity values of the background and graph lines in a cosmic ray graph image are irrelevant. The sole objective when extracting data

from such an image is the differentiation between graph lines and background. Thus, the image needs to be accurately binarized, within the set of true pixels (1) representing any data in the image and the set of false pixels (0) representing empty background space.

These requirements can be fulfilled by an adaptive binarization method. A study was conducted to identify the method or a set of techniques from different methods that would deliver the best results when applied to these greatly varying cosmic ray data graphs.

2.5.1. Otsu's method

One of the first adaptive binarization methods was Otsu's method. The algorithm assumes that the grayscale contains two classes of pixels (foreground and background) that are represented in a histogram by two peaks separated by a narrow sharp valley. It calculates the optimum threshold by separating those two classes so that their combined spread on a histogram is minimal, thus minimizing intra-class variance or maximizing inter-class variance (Otsu, 1979).

2.5.2. Niblack's method

Initial binarization algorithms were applied globally to a document image. Some statistic of the image such as the average or median grayscale value was manipulated by an algorithm or simply increased/decreased manually and this new value would be the threshold of the binarized image. This approach worked well for perfect grayscale scans of documents, but not for scans of degraded documents which contained stains, smudges and bleedthrough text. Some of these scans also had non-uniform lightning. For example, the top half of a page would be much lighter than the bottom half. If a threshold value is derived from such an image and applied globally to that image, the result would be a set of unrecognizable black and white objects.

Niblack (1986) devised a local binarization method that moves a rectangular window across the image and calculates the value of the pixel at the center of

that window (0 or 1) according to the threshold value derived from only the pixels within the window. This solved the problem mentioned above. The threshold value is calculated by:

$$T = m(x, y) + ks(x, y), \tag{2.33}$$

where m is the mean value of the pixels within the window, s the variance of those pixels and k is a constant which is usually set to -0.2.

The size of the window must be small enough to preserve local details and yet large enough to eliminate noise.

2.5.3. Sauvola and Pietkäinen's method

Sauvola and Pietkäinen (2000) devised an adaptive binarization method which uses local and global statistics to determine the foreground (text and graphic objects) and background (white or near-white space between text and graphic objects) of a document image and then applies a different binarization method to each.

First an algorithm is applied to the image to decide which binarization method should be applied to each part. The image is divided into tiles 10 to 20 pixels wide. The average grey value and transient difference (difference in local contrast) of each tile is then calculated. Scaling these variables to either 1 or 0 then defines whether each tile should be treated as mostly foreground or background.

2.5.3.1. Background binarization

To binarize background tiles, the weighted bound and threshold difference values for that window is calculated. The weighted bound ω_b is used for characterization of local pixel value profiles by tracking low, medium and high pixels in a small area (Sauvola & Pietkainen, 2000). The result of this step is a set of 3 ω_b curves calculated from the sets of low, medium and high pixels in that window.

The transient difference of the window is used again along with the ω_b curves to define a threshold value for the current tile.

2.5.3.2. Foreground Binarization

To binarize foreground tiles, a modified version of Niblack's method is used. The adapted formula used to binarize the foreground is:

$$T(x,y) = m(x,y) \cdot [1 + k \cdot (\frac{s(x,y)}{R} - 1)] \tag{2.34}$$

Where $m(x,y)$ is the local mean and $s(x,y)$ is the local standard deviation as in Niblack's method. R is the dynamic range of standard deviation fixed to 128 and k now takes on positive values, usually 0.05 (Sauvola & Pietkainen, 2000).

The speed of this method can be improved by only calculating the threshold for every nth pixel and then interpolating the rest.

2.5.4. Gatos and colleagues' method

Gatos *et al*. (2006) implemented a revised version of Sauvola and Pietkäinen's foreground binarization method along with pre-processing and post processing steps. The process of the method is as follows:

2.5.4.1. Preprocessing

During this phase a low-pass Wiener filter is applied to the image to remove noise, smooth background textures and enhance contrast between the background and text areas.

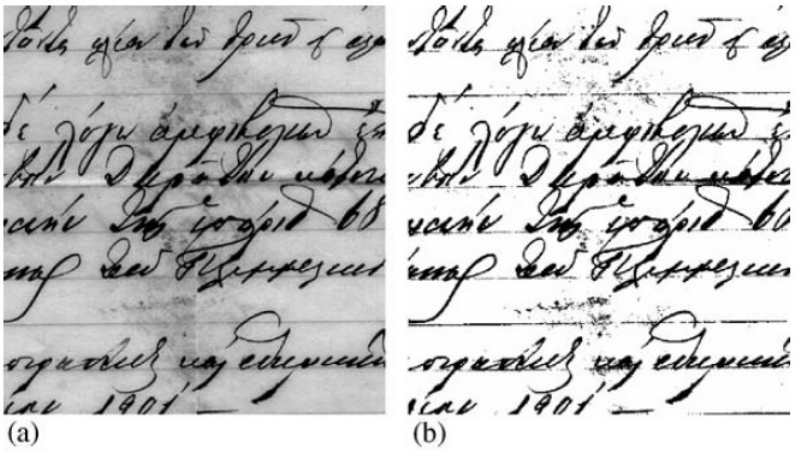


Figure 2.33.: Foreground estimation of Gatos *et al*.. Original image (a) and its roughly extracted foreground S (b) (Gatos *et al*., 2006).

The formula applied to achieve this is:

$$I(x,y) = \mu(\sigma^2 - \nu^2)(I_{s(x,y)} - \mu)/\sigma^2 \tag{2.35}$$

Where μ is the local mean, σ^2 the variance in a 3 x 3 neighborhood around each pixel and ν^2 is the average of all estimated variances for each pixel in the neighborhood. Let this image be I.

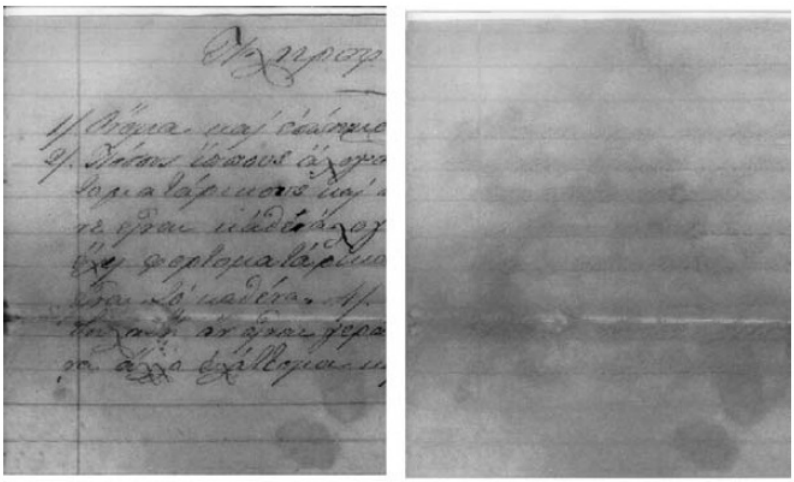
2.5.4.2. Rough foreground estimation

In this phase, Sauvola and Pietkäinen's foreground binarization method (Eq. 2.34) is applied to extract a rough representation of text areas in the original image. By applying the equation with $k = 0.2$, a rough binarized version of the original image is extracted. Let this image be S, as shown in Figure 2.33.

2.5.4.3. Background estimation

Using the binary image created in the previous phase, the background of the original image is extracted. Let this image be B, as can be seen in Figure 2.34.

If $S(x,y)$ has a value of 0 (white/background pixel), $B(x,y)$ is given the same value as $S(x,y)$.



(a) (b)

Figure 2.34.: Background estimation of Gatos *et al*.. Original image (a) and its extracted background B (b) (Gatos *et al*., 2006).

If $S(x, y)$ has a value of 1 (black/text pixel), the value of $B(x, y)$ is interpolated from a neighborhood around $S(x, y)$. This neighborhood should be large enough to cover at least two image characters.

2.5.4.4. Final thresholding

In this phase final thresholding is done by combining the estimated background with the preprocessed image. Test areas are then identified by areas where the combined height of B and I are above a certain threshold, which identifies a point in a histogram where there is a noticeable distance in the height between the foreground and background.

The average distance between the foreground and background is calculated by:

$$\delta = \frac{\sum_x \sum_y (B(x, y) - I(x, y))}{\sum_x \sum_y S(x, y)} \tag{2.36}$$

The minimum threshold d between text pixels and background pixels can be defined as $q \cdot d$. Where q is a weighting parameter which leads to successful OCR results if set at 0.8 and d is calculated by first calculating the average background values b of the background surface B that correspond to the text areas

of image S by:

$$B = \frac{\sum_x \sum_y (B(x,y)(1 - S(x,y))}{\sum_x \sum_y (1 - S(x,y))} \tag{2.37}$$

Then d is calculated by:

$$d(B(x,y)) = q\delta \left(\frac{(1 - p_2)}{1 + \exp(\frac{-4B(x,y)}{b(1-p1)} + \frac{2(1+p1)}{1-p1})} + p_2 \right) \tag{2.38}$$

The final binary image $T(x,y)$ is calculated by:

$$T(x,y) = \begin{cases} 1 & \text{if } B(x,y) - I(x,y) > d(B(x,y)) \\ 0 & \text{otherwise} \end{cases} \tag{2.39}$$

2.5.4.5. Post-processing

An erosion filter is applied to remove noise from the background. Any breaks or gaps caused by the erosion filter is removed with a dilation filter which is modified to prevent an increase of character stroke thickness. A final dilation filter is then applied to improve the quality of character strokes.

A comparison of the methods discussed thus far is shown in Figure 2.35.

2.5.5. An adaptive water flow model for binarization of degraded document images

All of the above mentioned methods have some difficulty binarizing document images with poor and variable contrast, shadow, smudge, and variable foreground/background intensities. Valizadeh and Kabir (2012) have devised a binarization method, based on morphological watersheds, that overcomes some of these challenges.

The key aspect that differentiates this method from the others is the extraction of blobs (connected groups) of pixels instead of single pixels and the identification of the foreground by using a watershed model. By extracting blobs instead of pixels, stroke connectivity is greatly improved.

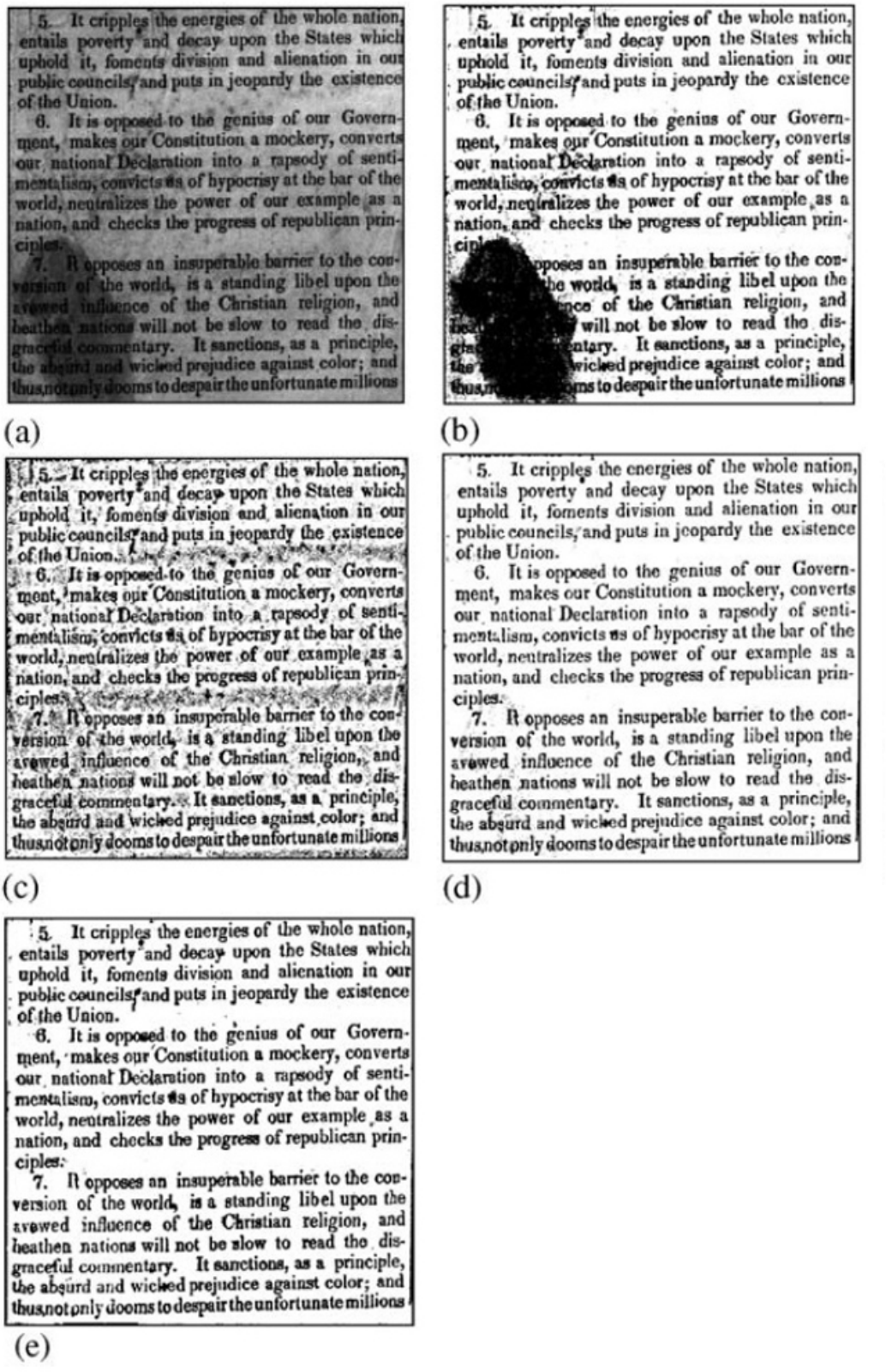


Figure 2.35.: Binarization of a typed document image (Gatos *et al*., 2006): (a) original image; (b) Otsu's method, (c) Niblack's method, (d) Sauvola and Pietkäinen's method, (e) Method of Gatos *et al*..

This method uses a rainfall model to add water to the landscape. As water is poured on the terrain, it gathers at local minima (valleys and holes). Once global stopping criteria are satisfied, the rainfall stops and the areas filled with water are binarized to extract the foreground.

The method is explained in detail in the following steps:

2.5.5.1. Region of interest extraction

Water is only poured onto edge pixels which are designated as regions of interest. To ensure that no regions of interest are missed, a Canny edge detector is applied to effectively identify all edges, including weak edges.

2.5.5.2. Stroke width measurement

The stroke width is used to set the rate of rainfall automatically and accurate measurement of this important text attribute improves binarization quality.

The stroke width is measured locally using Niblack's algorithm to binarize the image and then scan the result from four directions to obtain four Black Run Images or BRIs.

By scanning these BRIs, the straight lengths of connected black pixels are taken as possible stroke lengths. The local averages of stroke widths are identified using a 30 by 30 pixel window. The minimum value from the set of 4 BRI average stroke widths is taken as the stroke width of the image.

2.5.5.3. Stopping threshold for rainfall process

The local contrast of edge pixels is used as a stoppping threshold for the rainfall process. The contrast is measured by using the intensity of neighboring pixels. If (x, y) is an edge pixel, then the contrast is defined by:

$$C(x,y) = \max_{i=0,\ldots,3} \mid I(p_i) - I(p_i') \mid \tag{2.40}$$

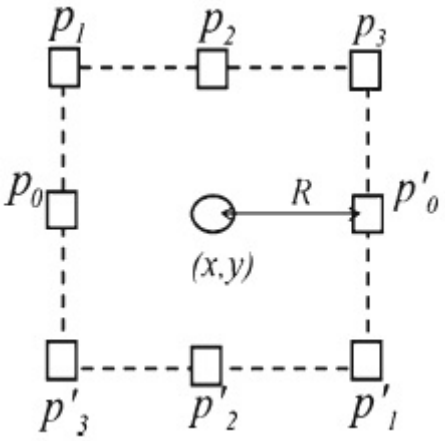


Figure 2.36.: Neighborhood for contrast measurement (Valizadeh & Kabir, 2012).

Where C is the contrast and $I(p_i)$ represents the intensity level of the original image at location p_i, as shown in Figure 2.36. $I(p'_i)$ and $I(p_i)$ are neighbors of (x, y) on opposite sides of (x, y).

An auxiliary image S is defined to control the rainfall process. Before any rainfall, all pixels of S are set to zero. During rainfall, $S(x, y)$ is examined to check if $S(x, y) < C(x, y)$. If this is true then water falls onto (x, y) and the value of $S(x, y)$ increases by one, otherwise the rainfall is stopped at (x, y).

2.5.5.4. Setting the rate of rainfall

'The rate of rainfall determines how much the height of a local minimum is raised when a drop of water fills it' (Valizadeh & Kabir, 2012) . This variable may be set manually, but to adapt it automatically prevents incorrectly connected and broken characters.

If W_1 is the amount of water that overflows the pond related to a character and W_2 is the amount of water that has filled the related pond when the rainfall stops, it has been observed through experimentation that if $W_2 = 0.5W_1$, the number of broken and touching characters is minimized. Therefore the stopping criteria for the rainfall is set to achieve this relation.

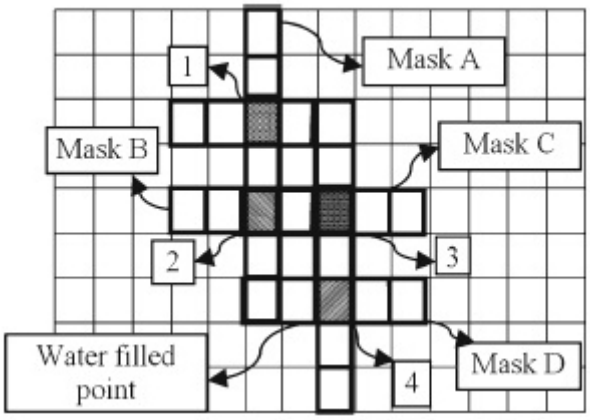


Figure 2.37.: The process of finding local minima (Valizadeh & Kabir, 2012).

If 0 and 255 are the intensity values of the foreground and background of the image respectively, then the water amount that overflows the pond related to a character and the water amount that flows from edge pixels to the corresponding pond are:

$$W_1 = 255 \cdot (\text{average character stroke width}) \tag{2.41}$$

$$W_2 = \text{rate of rainfall} \cdot \sum_{(x,y)\epsilon E} C(x,y) \tag{2.42}$$

Where $E = (x, y)$ and (x, y) is an edge pixel of the character. The rate of rainfall is calculated as follows:

$$RoR = \lambda \cdot 0.5 \tag{2.43}$$

$$\text{where } \lambda = \frac{\text{character stroke width}}{\text{number of corresponding pixels}} \tag{2.44}$$

The rate of rainfall is set proportional to the stroke width, therefore it may differ in different regions of the image if the size of the characters differ.

2.5.5.5. Fast algorithm for finding local minima

The water flow model guides droplets to local minima. These minima are found by iteratively finding the minimum in a small area, making that minimum the new center of the search mask and finding a new minimum in the new area until

the current minimum is the deepest point in the valley. This process is shown in Figure 2.37.

This can be done using a n by n pixel mask, but because this is the most time consuming phase of the method, a faster solution had to be devised. By forming a mask which consists of a center pixel and extending that center pixel only vertically and horizontally until the mask's total horizontal length and vertical length are both n, thus forming a cross or plus sign, the processing time during this step is reduced by almost a third.

This is possible because of the correlation between neighboring pixels, making them similar enough to be able to track the gradient of the grayscale 'landscape' using this method. The noticeable speed increase is caused by only having to search through $4n + 1$ pixels per mask instead of $(2n + 1)^2$ pixels.

The minima found by this search are then used as targets for rainfall.

2.5.5.6. Blob extraction

Once the rainfall stopping criteria have been satisfied, the dry and water-filled regions are separated. If $I(x, y)$ and $G(x, y)$ represent the height of terrain before and after rainfall respectively, then the binary image B, where 0 represents wet regions and 1 represents dry regions, is derived as follows:

$$B(x,y) \begin{cases} 0 & \text{if } G(x,y) - I(x,y) > 0 \\ 1 & \text{otherwise} \end{cases} \tag{2.45}$$

where each black connected component in B represents a pond. In addition to character representing blobs, some noise representing blobs are also extracted. Thus blobs are labeled to be classified later. The competitive characteristic of this method causes noisy blobs between characters to be ignored, preventing connected characters by an incorrectly extracted blob.

2.5.5.7. Blob classification and final binarization

Two features are extracted from each blob in order to classify them as text or non-text. These features are average water amount AWA, that represent the local contrast of text and average grey level AGL. These features are defined for blob φ as follows:

$$AWA(\varphi) = \frac{\sum_{(x,y)\epsilon S_1(\varphi)}(G(x,y) - I(x,y))}{\text{cardinality of } S_1(\varphi)} \times \frac{1}{M_1}, \tag{2.46}$$

$$AWL(\varphi) = \frac{\sum_{(x,y)\epsilon S_1(\varphi)}(I(x,y) - M_2)}{\text{cardinality of } S_1(\varphi)} \times \frac{1}{M_2 - M_3}, \tag{2.47}$$

where $S_1(\varphi) = \{(x,y)|(x,y)\,\epsilon$ blob $\varphi\}$ and M_1, M_2 and M_3 are normalization parameters calculated from the global attributes of the image.

These parameters are calculated by applying Otsu's method to the water filled image, $G(x,y) - I(x,y)$, to estimate the text and background pixels and then define these parameters as follows:

- M_1 represents the average of filled water for the text pixels;
- M_2 represents the average values of gray levels for the text pixels;
- M_3 represents the average values of gray levels for the background pixels;

This normalization makes text blobs from different images used as training data seem more similar and has the same effect on background blobs. This helps maintain a reliable classifier.

The AWA and AWL values for each blob is sent through a three layer Preceptron for classification. If the output for a blob is smaller than 0.5, then it is classified as text, otherwise it is classified as non-text. The proposed classifier removes almost all blobs related to the non-text and yields a clean binary image.

A summary of the method is shown in Figure 2.38.

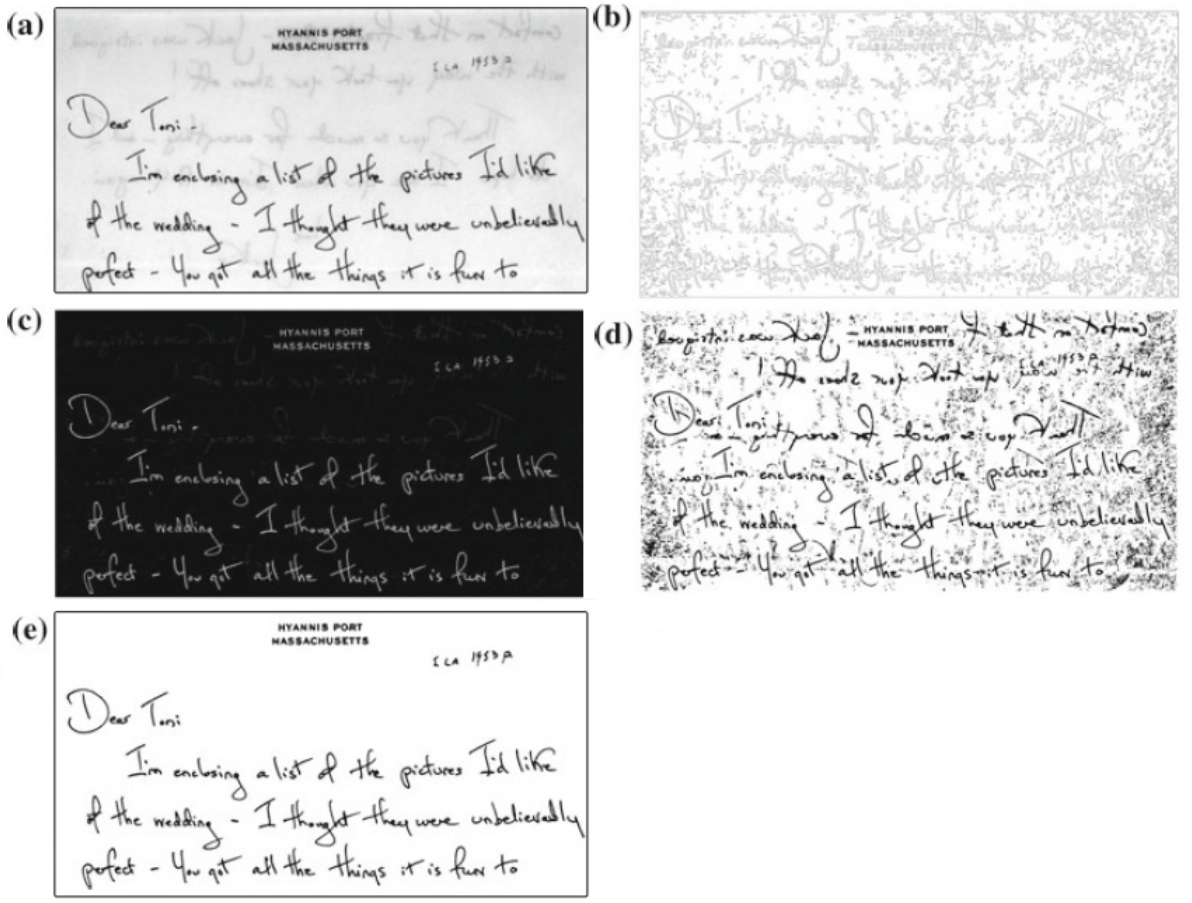


Figure 2.38.: Phases of adaptive water flow binarization (Valizadeh & Kabir, 2012). (a) A degraded document image. (b) The edges extracted by Canny detector. (c) Water amount after rainfall stopping. (d) Results of blob extraction algorithm. (e) Binary images after classification of blobs.

2.6. Digitization of cosmic ray recordings in the frequency domain

2.6.1. Drevin's method

The adaptive binarization methods discussed thus far all focus on image processing within the spatial domain and primarily target text.

Drevin (2008) designed a method to filter cosmic ray recordings in the frequency domain by applying a low-pass filter with a cut-off frequency determined using the Lorentz curve (Lorentz, 1905). The method filters the rows and columns of the image separately and so smoothes the scale lines and hour markers of the image respectively. These filtered images are used to extract the scale lines and hour markers in the process of binarizing the image.

The Lorentz curve, which is applied to filter the image, was originally used to measure the inequality of income. Goel and Vidakovic (1995) applied the Lorentz curve in a different field by replacing the wealth of a population of individuals with the energy of wavelet transform coefficients:

$$\begin{aligned} S(0) &= 0 \\ S(k) &= \textstyle\sum_{u=1}^{k} P(u),\ k = 1, 2, 3, ..., n, \end{aligned} \tag{2.48}$$

where $S(k)$ is the total power of the k coefficients with the lowest power, $P(u)$ represents the energy of coefficient u and n is the number of coefficients which are sorted by order of increasing energy. Drevin's method calculates $S(k)$ by sorting the coefficients in decreasing frequency order, making $S(k)$ the total power of the k highest frequencies.

The cut-off frequency is calculated by plotting the points:

$$(k/n, S_k/S_{n)}),\ k = 0, 1, 2, ..., n \tag{2.49}$$

which results in the Lorentz curve of the energy distribution of the transform. The point where the maximum distance between the Lorentz curve and the diagonal appears, is used as the cut-off frequency.

The one-dimensional Fourier transform representing the rows of the image and the one-dimensional Fourier transform representing the columns of the image are calculated and filtered individually using the following process:

In the case of the row Fourier transform: The cut-off frequency D_r is obtained from the power spectrum of the row median's Fourier transform. The Fourier transform of all the rows of the image is filtered using a Gaussian low-pass filter function as follows:

$$G_r(u) = F_r(u)H_r(u), \tag{2.50}$$

where $G_r(u)$ represents the filtered Fourier transform of all the rows in the image, $F_r(u)$ the Fourier transform of all the rows in the image and $H_r(u)$ is the filter function applied to the Fourier transform.

The use of the inverse Fourier transform, for all the rows, results in an image where all vertical lines have been smoothed. This image is subtracted from the original and the difference between them is used to calculate a threshold.

The distribution of differences is used, where the threshold is the first gray level i where the slope $s(i) = -1.$ This slope is calculated as follows:

$$s(i) = N(i+1) - N(i), \tag{2.51}$$

where $s(i)$ represents the slope, N the number of pixels normalized to the interval $[0, P]$ and P is the maximum positive gray level in the image.

This procedure is applied to the columns of the image and the result is combined with the thresholded row image using a logical OR function. The result is the final binarization of the image.

2.7. Concluding remarks

Drevin and Du Plessis (2009) evaluated popular binarization methods and their possible application to binarize and digitize cosmic ray recordings. They concluded that the method of Gatos *et al*. provided the best binarization results, although the method provides roughly the same results as Sauvola and Pietkäinen's method when binarizing images with a low gray-level average. Drevin's method proved to be the most appropriate method to handle cosmic ray recording images with high contrast.

The conclusion of this evaluation was that no single method would be able to binarize all of the cosmic ray recordings. A method could analyze the image and apply a certain secondary method, from a predetermined collection, that would probably yield the most accurate results. The alternative approach is to apply image enhancement techniques to the cosmic ray recording images in such a way as to standardize the data set to the point where a single binarization method could be applied to all of the images. The success of both these possibilities rests upon the creation of a method to analyze each recording image to determine its characteristics.

At the end of his dissertation, Du Plessis (2010) concluded that image processing techniques could certainly be implemented to digitize cosmic ray recordings. Thus, during the experimental phase of this study, the methods and techniques mentioned in the literature will be applied to cosmic ray recording images to determine if those methods or techniques could be utilized in the overall process of adaptively binarizing these images.