

IMAGE PROCESSING TECHNIQUES FOR MACHINE VISION

Alberto Martin and Sabri Tosunoglu

Florida International University
Department of Mechanical Engineering
10555 West Flagler Street
Miami, Florida 33174

E-mail: tosun@eng.fiu.edu

Abstract – *Image Processing Algorithms are the basis for Image Computer Analysis and Machine Vision. Employing a theoretical foundation – Image Algebra – and powerful development tools – Visual C++, Visual Fortran, Visual Basic, and Visual Java – high-level and efficient Computer Vision Techniques have been developed. This paper analyzes different Image Processing Algorithms by classifying them in logical groups. In addition, specific methods are presented illustrating the application of such techniques to the real-world images. In most cases more than one method is used. This allows a basis for comparison of different methods as advantageous features as well as negative characteristics of each technique is delineated.*

INTRODUCTION

The Image Algebra [10] forms a solid theoretical foundation to implement computer vision and image processing algorithms. With the use of very efficient and reliable high-level computer languages such as C/C++, Fortran 90, and Java, innumerable image processing and machine vision algorithms have been written and optimized. All this code written and compiled has become a powerful tool available for researchers, scientists and engineers, which further accelerated the investigation process and incremented the accuracy of the final results.

The discussion of the Basic Machine Vision and Image Processing Algorithms should be divided in five major groups [11]:

- Grey-Level Segmentation or Thresholding Methods
- Edge-Detection Techniques
- Digital Morphology
- Texture
- Thinning and Skeletonization Algorithms

GREY-LEVEL SEGMENTATION TECHNIQUES

Thresholding or grey-level segmentation is an essential concept related with image processing and machine vision. Thresholding is a conversion between a grey-level image and a bilevel image. Bilevel image is a monochrome image

only composed by black and white pixels. It should contain the most essential information of the image (i.e., number, position and shape of objects), but is not comparable with the information offered by the grey-level image. Most of the time pixels with similar grey levels belong to the same object. Therefore, classifying the image by grey-level pixels may reduce and simplify some image processing operations such as pattern recognition, and classification.

The most essential thresholding operation will be the selection of a *single threshold value*. All the grey levels below this value are classified as black (0), and those above white (1). Most of time it is impossible to segment an image into objects and background with a single threshold value because of noise and illumination effects. A tentative simple approach could be the use of the *mean grey level* in the image as a threshold. This would cause about half of the pixels to become white and the other half black.

Another easy method to find the threshold is the *P-tile method* [8]. This method uses the histogram of grey levels in the image. With the histogram and the percentage of black pixels desired, one determines the number of black pixels by multiplying the percentage by the total number of pixels. Then one simply counts the pixels in histogram bins, starting at bin 0, until the count is greater than or equal to the desired number of black pixels. The threshold is the grey level associated with last bin counted. This method has the advantage that it is possible to change the percentage of the black pixels desired.

The *Edge Pixel method* [8] produces a threshold value based on the digital Laplacian, which is a non-directional edge-detection operator. The Histogram of the original image is found considering only those pixels having large Laplacians. Then the threshold is computed using this new histogram.

The *Iterative method* [9] is a process in which an initial guess at a threshold is redefined by consecutive passes through the image. It thresholds the image into object and background repeatedly, using the levels in each of them to improve the value of T [9]. The initial guess at the threshold

is the mean grey level. Then, the mean grey level for all the pixels below the threshold is found (T_b), and the same process for the pixels above the initial threshold is also found (T_0). Now a new estimation is made as $(T_b + T_0) / 2$. Then the process is repeated using this threshold. The process stops when in two consecutive passes through the image no change in the threshold is observed.

In a fuzzy set, an element x belongs to a set S with a particular probability u_x . Thresholding an image means to classify pixels as belonging to either the set of background pixels or the set of object pixels. It is obvious that the application of fuzzy sets to image segmentation may result in a good method. A good attempt is made by using a measure of fuzziness (Huang 1995), which is a distance between the original grey level image and the thresholded image [8]. By minimizing the fuzziness, the most accurate thresholded version of the image should be produced. This method is called *Fuzzy Sets method*.

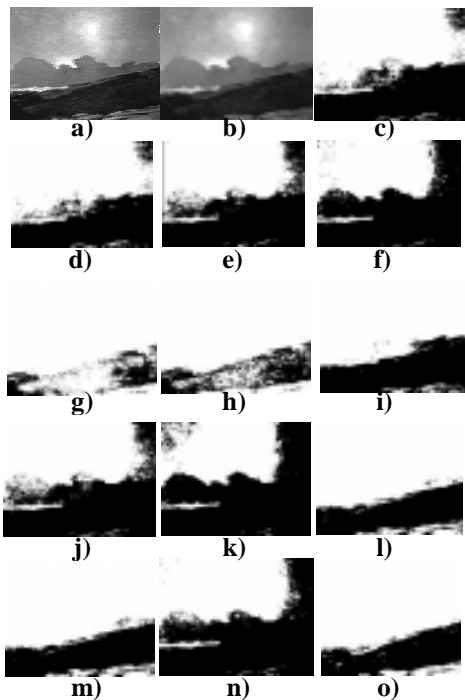


Figure 1. Comparison between thresholding methods. a) Original Image. b) Grey-level image c) Mean value ($T = 115$) d) P-Tile 40% black ($T = 110$) e) P-Tile 50% black ($T = 125$) f) P-Tile 60% black ($T = 136$) g) Two-Peaks method ($T=30$) h) Edge Pixels 15% ($T = 30$) i) Iterative selection ($T=97$) j) Entropy -Pun ($T = 125$) k) Entropy-Kapur ($T = 143$) l) Entropy-Johannsen ($T = 71$) m) Fuzzy entropy ($T =76$) n) Fuzzy Yager ($T = 128$) o) Min-Error ($T = 53$).

All the methods presented so far assumed that the background and object pixels have non-overlapping grey levels. Most of the time in the real life that ideal assumption

is not true. Therefore, the selection of a single threshold for an image is not possible in most of the case. However, all that is needed is for the two classes of pixels not to overlap over each of a set of regions that collectively form the image. The first step is to determine how many independent regions form the image and their sizes [9]. For each of them, it is possible to apply one of the previous thresholding methods. Therefore, there is not a single threshold value, but one per region. Only one thing to take care of when using regional thresholding method, it is necessary to make sure that either each region contains a sample of both object and background pixels, or that no thresholding is attempted when only one pixel class exists.

A good example of a regional thresholding algorithm is the one proposed by Chow and Kaneko in 1972. This thresholding method finds a bimodal histogram for each region. This histogram is intended to have two classes of pixels (object and background) [6].

Figure 1 illustrates various thresholding methods applied to a grey-level image (Figure 1 b) obtained from a color image (Figure 1 a).

EDGE DETECTION TECHNIQUES

In Image Processing, an edge is the boundary between an object and its background. They represent the frontier for single objects. Therefore, if the edges of image's objects can be identified with precision, all the objects can be located and their properties such as area, perimeter, shape, etc, can be calculated. Edge detection is an essential tool for machine vision and image processing [1].

Figure 2 illustrates an edge detection process. There are three overlapping objects in the original (Figure 2 a), with a uniform grey background. After the application of a edge detector technique, the objects have been isolated, and only the boundaries between the regions are identified (Figure 2 b).

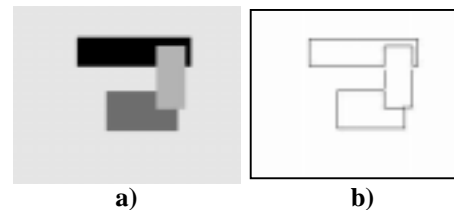


Figure 2. Example of Edge Detection. a) Original overlapping rectangles image. b) Edge-Enhanced image showing only the boundaries.

Edge detection is the process of locating the edge pixels. Then an *edge enhancement* will increase the contrast between the edges and the background in such a way that

edges become more visible. In addition, *edge tracing* is the process of following the edges, usually collecting the edge pixels into a list [8].

A model of an edge can be ideally represented by the *Step Edge*, which is simply a change in grey level occurring at one location. The step edge is an ideal model because in a real image never a change in a grey level occurs in the extreme left side of a pixel due to noise and illumination disturbances. Due to digitization, it is unlike that the image will be sampling in such a way that all of the edges happen to correspond exactly with a pixel boundary. The change in grey level may extend across various pixels. The actual position of the edge is considered to be in the center of the ramp connecting the low grey level to the high grey level. This is called *Ramp Edge*. Figure 3 shows these two cases [8].

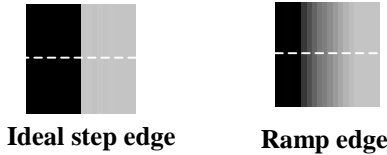


Figure 3. Edge Models

The *Sobel edge detector* is a Template-Based Edge Detector that uses templates in the form of convolution masks such as [9]:

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad S_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

These templates are an approximation to the gradient at the pixel in the center of the template.

Another edge detector technique is Kirsch Edge Detector [8]. The masks given by these templates try to model the kind of grey level change seen near an edge having various orientations. There is a mask for each of eight compass directions. For instance K0 implies a vertical edge (horizontal gradient) at the pixel corresponding at the center of the mask. To find the edge, I is convolved with the eight masks at each pixel position. The response is the maximum of the responses of any of the eight masks and the directions quantified into eight possibilities ($\pi/4 * i$).

Two advanced and optimized edge detectors are Canny Edge Detectors and Infinite Symmetric Exponential Filter (ISEF). Both are classified as Mathematical Edge Detectors.

$$K0 = \begin{bmatrix} -3 & -3 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & 5 \end{bmatrix} \quad K1 = \begin{bmatrix} -3 & 0 & 5 \\ -3 & -3 & 3 \\ -3 & -3 & 3 \end{bmatrix} \quad K2 = \begin{bmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} \quad K3 = \begin{bmatrix} 5 & 5 & -3 \\ 5 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix}$$

$$K4 = \begin{bmatrix} 5 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & -3 & -3 \end{bmatrix} \quad K5 = \begin{bmatrix} -3 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & 5 & -3 \end{bmatrix} \quad K6 = \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & -3 \\ 5 & 5 & 5 \end{bmatrix} \quad K7 = \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & 5 \\ -3 & 5 & 5 \end{bmatrix}$$

For John Canny an optimal edge detector need to satisfy these three conditions:

- The edge detector should respond only to edges, and should find all of them; no edges should be missed.
- The distance between the edge pixels as found by the edge detector and the actual edge should be as small as possible.
- The edge detector should not identify multiple edge pixels where only a single edge exits.

Canny edge detection algorithm

1. Read the image I .
2. Convolve a 1D Gaussian mask with I .
3. Create a 1D mask for the first derivative of the Gaussian in the x and y directions.
4. Convolve I with G along the rows to obtain I_x , and down the columns to obtain I_y .
5. Convolve I_x with G_x to have I_x' , and I_y with G_y to have I_y' .
6. Find the magnitude of the result at each pixel (x, y)

$$M(x, y) = \sqrt{I_x'(x, y)^2 + I_y'(x, y)^2}$$

The Infinite Symmetric Exponential Filter uses another optimization function to find the edge in an image this function can be written as:

$$C_N^2 = \frac{4 \int_0^\infty f^2(x) dx \cdot \int_0^\infty f'^2(x) dx}{f^4(0)}$$

The function C_N will be minimized with an optimal smoothing filter for an edge detector. This is called infinite symmetric exponential filter (ISEF):

$$f(x) = \frac{p}{2} e^{-p|x|} \quad f(x, y) = a \cdot e^{-p(|x|+|y|)}$$

1D 2D

The filter in the ISEF Edge Detector is presented as one-dimensional recursive filter. Assuming the 2D-filter function real and continuous, it can be rewritten as:

$$f[i, j] = \frac{(1-b)b^{|x|+|y|}}{1+b}$$

The use of recursive filtering speeds up the convolution. The value b can be entered by the user. Figure 4 illustrates different edge detectors techniques applied to a test image (Figure 4 a).

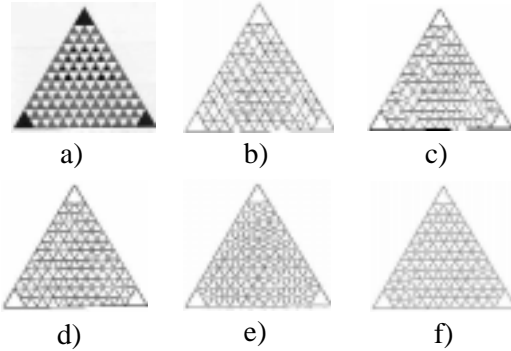


Figure 4. Output image using different edge detector techniques. a) Original Image. b) Gradient Edge Detector. c) Kirsch Edge Detector. d) Sobel Edge Detector. e) Canny Edge Detector. f) ISEF Edge Detector.

DIGITAL MORPHOLOGY

The concept of digital morphology is based on the fact that images consist of set of pictures elements called pixels that collect into groups having a two-dimensional structure called shape. A group of mathematical operations can be applied to the set of pixels to enhance or highlight specific aspects of the shape so that they can be counted or recognized [8], [9], [10].

This part of the image processing analysis deals with image filtering and geometric analysis of the structuring elements. *Erosion* or elimination of set of pixels having a given pattern (structuring element) and *dilation* or addition of a given pattern to a small area, are basic morphology operations. Binary morphological operations are defined on bilevel images.

In general, an operator is defined as a set of black pixels with a specific location for each of its pixels given by the pixel row and column indices. Mathematically, a pixel is thought as a point in two-dimensional space.

The **binary dilation** of the S by a set S_1 is:

$$S \oplus S_1 = \{c | c = s + s_1, \quad s \in S, \quad s_1 \in S_1\}$$

S represents the image being transformed, and S_1 is a second set of pixels, with a peculiar shape that acts on the pixels of S producing an expected result. This set S_1 is called *Structuring Element*, and its shape defines the nature of the dilation. Figure 5 shows a dilation morphology operation of an image S by using a structuring element S_1 [9]

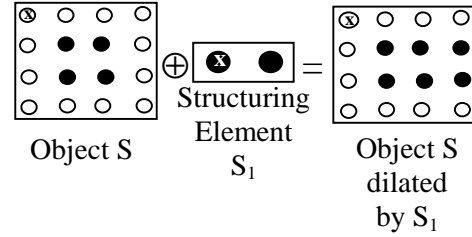


Figure 5. Dilation of set S using a structuring element S_1 .

It is important to notice that the pixel marked with an x is considered the origin of the image. The origin could be either a white or black pixel. The structuring element is not more than a template moving over the image. The dilation can be considered to be the union of all translations specified by the structured element, that is:

$$S \oplus S_1 = \bigcup_{s_1 \in S_1} (S)_{s_1}$$

And because it is a commutative operation, dilation can also be considered the union of all translations of the structuring element by all pixels in the image:

$$S \oplus S_1 = \bigcup_{s \in S} (S_1)_s$$

Figure 6 illustrates the dilation process using a 6x6 image and a 3x3-structuring element. The ASCII files are also presented to support the process.

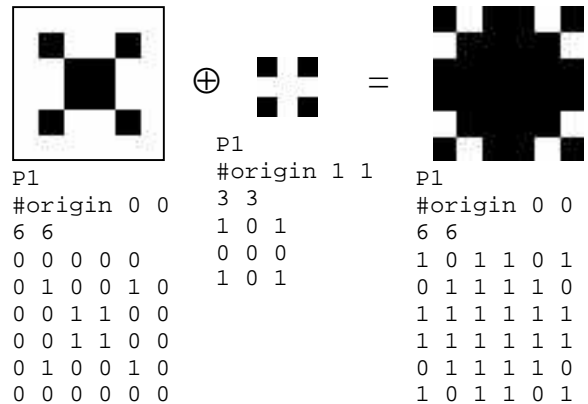


Figure 6. Dilation process in a test image.

The **binary erosion** of the S by a set S_1 is:

$$S \ominus S_1 = \{c | (S_1)_c \subseteq S\}$$

The Binary erosion [9] will be the set of pixels c such that the structuring element S_1 translated by c corresponds to a set of black pixels in S. As the result, any pixels that do not

match the pattern defined by the black pixels in the structuring element will not belong to the result.

Erosion and **Dilation** can be associated in the following manner:

$$(S \ominus S_1)^c = S^c \oplus \hat{S}_1$$

This means that the complement of erosion is the same as a dilation of the complement image by the reflected structuring element. Figure 7 illustrates the erosion process using a 10x10 image and a 3x3-structuring element. The ASCII files are also presented to support the process.

A combination of the simplest morphology operation: dilation and erosion will result in two very helpful image processing operation. They are called opening and closing [8].

Opening: Application of erosion immediately followed by a dilation using the same structuring element. This binary operation tends to open small gaps between touching objects in an image. After an opening objects are better isolated, and might be counted or classified. A practical application of an opening is removing noise. For instance after thresholding an image.

Closing: Application of a dilation immediately followed by erosion using the same structuring element. The closing operation closes or fills the gaps between objects. A closing can be used for smoothing the outline of objects after a digitization followed by

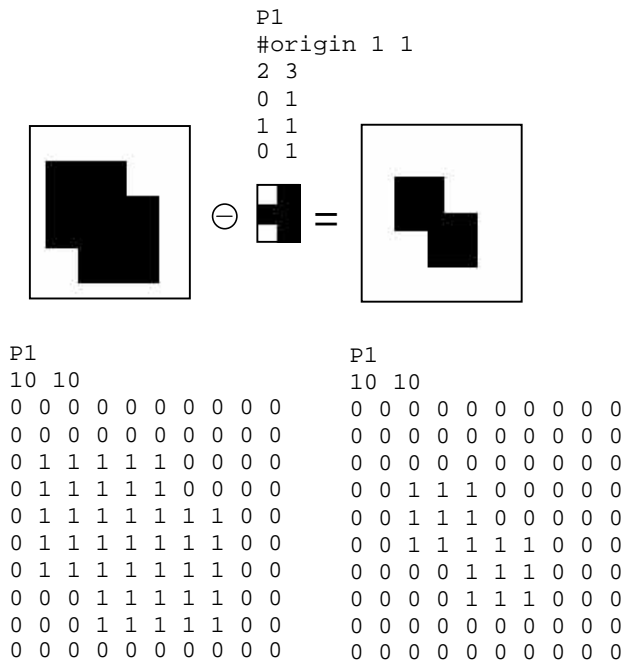


Figure 7. Erosion process in a test image.

Figure 8 shows the results of applying opening from depth 1 (1 dilations followed by one erosion) to depth 5. It can be observed that the result of multiple openings is a negative image with smoothing edges.



Figure 8. The result of applying multiple opening to an image.

TEXTURE

The repetition of a pattern or patterns over a region is called texture. This pattern may be repeated exactly, or as set or small variations. Texture has a conflictive random aspect: the size, shape, color, and orientation of the elements of the pattern (textons) [9].

The main goal identifying different textures in machine vision is to replace them by a unique grey level or color. In addition there is another problem associated with texture: scaling. Equal textures at different scales may look different for an image-processing algorithm. For that reason is unlikely that a simple operation will allow the segmentation of textured regions. But some combination of binary operations may result in an acceptable output for a wide range of textures.

The simplest way to perform the texture segmentation in grey-level images is that the grey level associated with each pixel in a textured region could be the average (mean) level over some relatively small area. This area is called *window* and can vary in size to capture different scales. The use of *windows* is very convenient in this case, since texture deals with region instead of individual pixels [9].

The method can be stated as following:

1. For each pixel in the image, replace it by the average of the levels seen in a region $W \times W$ pixels in size centered at that pixel.
2. Threshold the image into two regions using the new average levels. The exact location of the boundary between regions depends on the threshold method that is applied.

An improved method uses the standard deviation of the grey level in a small region instead of the mean. The standard deviation brings information about how many pixels in that regions belong to textons and how many belong to the background. The precision with which the boundary between regions is known is a function of the window's size.

A texture is a combination of a large number of textons. Isolating textons and treat them as individual objects, it is something doable. Once this is done, it should be possible to locate edges that result from the grey-level transition along boundary of a texton.

Analyzing some edge properties such as common directions, distances over which the edge pixel repeat, or a measure of the local density, it is possible to characterize the texture.

The number of edge pixels in a window can be found after applying an edge detector to that window. Then, the density is calculated dividing the number of edge pixels found by the area of the window. From here, useful information can be extracted. For instance edge direction, the mean x and y component of the gradient at the edge, and the relative number of pixels whose principal direction is x and y [7].

The combination of edge enhancement and co-concurrence is a smart solution (Dyer 1980 – Davis 1981) that can be used in grey-level texture segmentation. The computation of the co-concurrence matrix of an edge-enhanced image gives better results than the traditional method.

Figure 9 illustrates this method applied to Figure 1 b). Notice the difference between the contrast of the co-concurrence matrix, Figure 9 b), and the contrast of the co-concurrent matrix of the edge image, Figure 9 d). In this case Sobel edge detector was used. Figure 9 e) is the thresholded image. The entire upper region containing the sky is marked as black, except for the region with the sun. The edge enhancement method works quite well in a real image like this one.

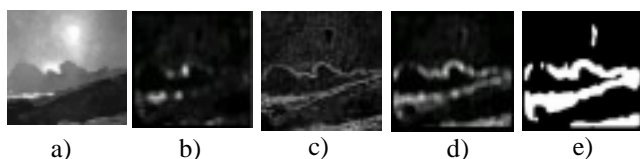


Figure 9. Combination of edge-enhancement and co-concurrence. a) Original image. b) Contrast of the co-concurrence matrix, distance 1 horizontal. c) Edge-enhancement image (Sobel). d) Contrast of the co-concurrent matrix of edge image. e) Thresholded image.

THINNING AND SKELETONIZATION ALGORITHMS

Skeletonization was introduced to describe the global properties of objects and to reduce the original image into a more compact representation. The skeleton expresses the structural connectivities of the main components of an

object and it has the width of one pixel in the discrete case. These kinds of techniques have a wide range of applications, for example skeletonization has been applied successfully in solving character recognition problems.

A basic method for skeletonization is thinning. It is an iterative technique, which extracts the skeleton of an object as a result. In every iteration, the edge pixels having at least one adjacent background point are deleted. All those pixels can be eroded, only if its removal doesn't affect the topology of the object. The skeleton represents the shape of the object in a relatively small number of pixels [9].

Thinning works for objects consisting of lines (straight or curved). This method does not work for object having shapes that encloses a large area. Thinning is most of the time an intermediate process, to prepare the object for further analysis. The subsequent processes determine the properties of the skeleton. For the same object, a skeleton may work find in one situation, but may not work in all situations.

The first definition of skeleton was made by Blum in 1967. He defined the *medial axis function* (MAF). Medial axis is basically defined as the set of points, which are the center points of largest circles that can be contained inside the shape or the object. To represent a shape as a skeleton, and still have the capability to reconstruct the original shape, there can be a radius function that is associated with the skeleton points. The MAF of a shape is the locus of the centers of all maximal discs contained in the shape. A maximal disc contained in the shape is any circle with its interior that is contained in the shape. A maximal disc contained in the shape is any circle with its interior that is contained in the shape. MAF is a reversible transform, which means it can be inverted to give back the original image [2], [8].

The MAF in its original implementation needs time and space and it is very difficult to implement directly. For that reason the continuous transform its converted to a discrete one.

A good approximation of MAF on a sampled grid is easily obtained computing first the distance from each object pixel to the nearest boundary pixel, and then calculates the Laplacian of the distance image. Pixels having large values belong to the medial axis. The way that distance between the object pixels and the boundary is measured has an influence on the final result (skeleton). Figure 10 illustrates the thinning process using the Medial Axis Transform.

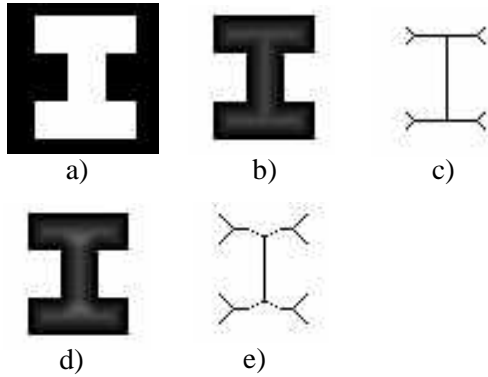


Figure 10. Skeletonization process using the medial axis transform. a) Sampled Image. b) Distance Map for 8-distance. c) Skeleton for 8-distance. d) Distance Map for 4-distance. e) Skeleton for 4-distance.

The Template-Based Mark-and-Delete Thinning Algorithms are very popular because of their reliability and effectiveness. This type of thinning processes uses templates, where a match of the template in the image, deletes the center pixel. They are iterative algorithms, which erodes the outer layers of pixel until no more layers can be removed. The Stentiford Thinning Method is an example of these kinds of algorithms [9], [14].

It uses a set of four 3×3 templates to scan the image. Figure 11 shows these four templates.

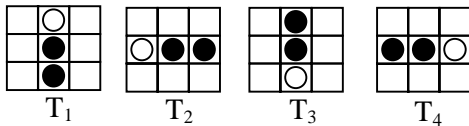


Figure 11. Templates to identify pixels to be eroded in the Stentiford Method. The empty white boxes belong to places where the color of the pixel does not need to be checked.

The Stentiford Algorithm can be stated as following [9]:

1. Find a pixel location (i,j) where the pixels in the image match those in template T_1 . With this template all pixels along the top of the image are removed moving from left to right and from top to bottom.
2. If the central pixel is *not an endpoint*, and has *connectivity number = 1*, then mark this pixel for deletion.

Endpoint pixel: A pixel is considered an endpoint if it is connected to just one other pixel. That is, if a black pixel has only one black neighbor out of the eight possible neighbors.

Connectivity number: It is a measure of how many objects are connected with a particular pixel.

$$C_n = \sum_{k \in S} N_k - (N_k \cdot N_{k+1} \cdot N_{k+2})$$

where: N_k is the color of the eight neighbors of the pixel analyzed. N_0 is the center pixel. N_1 is the color value of the pixel to the right of the central pixel and the rest are numbered in counterclockwise order around the center.

$$S = \{1,3,5,7\}$$

Figure 12 illustrates the connectivity number. Figure 12 a) represents connectivity number = 0. b) represents connectivity number = 1, the central pixel might be deleted without affecting the connectivity between left and right. c) represents connectivity number = 2, the deletion of the central pixel might disconnect both sides. d) represents connectivity number = 3, and e) represents connectivity number = 4.

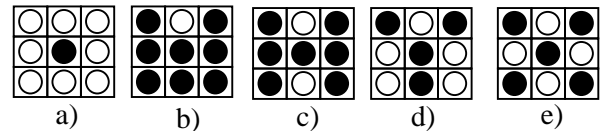


Figure 12. Connectivity numbers.

3. Repeat steps 1 and 2 for all pixel locations matching T_1 .
4. Repeat steps 1-3 for the rest of the templates: T_2 , T_3 , and T_4 .
 T_2 will match pixels on the left side of the object, moving from bottom to top and from left to right. T_3 will select pixels along the bottom of the image and move from right to left and from bottom to top. T_4 locates pixels on the right side of the object, moving from top to bottom and right to left.
5. Set to white the pixels marked for deletion.

Figure 13 shows some examples of the thinning process using the Stentiford Algorithm.

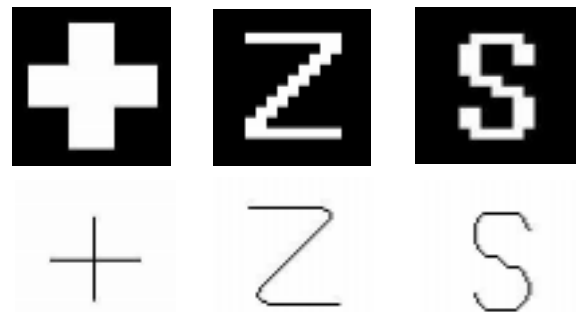


Figure 13. Skeletonization Process using Stentiford Algorithm.

Another type of skeletonization algorithm is The Zhang-Suen Thinning Algorithm [9], [14]. This skeletonization algorithm is a parallel method that means the new value obtained only depend on the previous iteration value. It is fast and simple to be implemented. This algorithm is made by two subiterations. In the fist one, a pixel $I(i,j)$ is deleted if the following condition are satisfied:

1. Its connectivity number is one.
2. It has at least two black neighbors and not more than six.
3. At least one of $I(i,j+1)$, $I(i-1,j)$, and $I(i,j-1)$ are white.
4. At least one of $I(i-1,j)$, $I(i+1,j)$, and $I(i,j-1)$ are white.

In the second subiteration the conditions in steps 3 and 4 change.

1. Its connectivity number is one.
2. It has at least two black neighbors and not more than six.
3. At least one of $I(i-1,j)$, $I(i,j+1)$, and $I(i+1,j)$ are white.
4. At least one of $I(i,j+1)$, $I(i+1,j)$, and $I(i,j-1)$ are white.

At the end, pixels satisfying these conditions will be deleted. If at the end of either subiteration there are no pixels to be deleted, then the algorithm stops. Figure 14 shows some examples of the thinning process using the Zhang-Suen Algorithm



Figure 14. Skeletonization Process using Zhang-Suen Algorithm.

CONCLUSION

Image processing is the study of representation and manipulation of pictorial information. Digital image processing is performed on digital computers that manipulate images as arrays or matrices of numbers.

The latest advancements in computer technology have opened the use of image processing analysis to fields that for their complexity would be impossible to be included in the past. High computational speed, high video resolution, more efficient computer language to process the data, and more efficient and reliable computer vision algorithms are some of the factors that let fields such as medical diagnosis, industrial quality control, robotic vision, astronomy, and intelligent vehicle / highway system to be included as a part of the large list of applications that use computer vision analysis to achieve their goals.

More and more complex techniques have been developed to achieve new goals unthinkable in the pass. Machine Vision researchers started using more efficient and faster mathematical approaches to solve more complicated problems. Convolution methods widely used in computer vision can be speed up by using Fourier Transforms and Fast Fourier Transforms.

The idea that an image can be decomposed into a sum of weighted sine and cosine components it is very attractive. The functions that produce such decompositions are called *basis* functions. Examples of basis functions are Fourier and Wavelet Transforms. A wavelet, like Fourier Transform has a frequency associated with it but in addition a scale factor has to be considered. Only some basis functions produce a decomposition that has a real importance in image processing.

To emulate human's ability to learn and the human brain's capacity to analyze, reason, and discern represent difficult tasks for computers. The brain is actually a highly complex, nonlinear and parallel computer. Its capacity to perform certain tasks such as image processing, pattern recognition, motor control, and perception is several times faster than the fastest available computer.

Studies to develop artificial tools, based on mathematical model that simulated the brain performance started 50 years ago. The Artificial Neural Network (ANN) is an example of this approach. The use of Neural Nets for symbol and pattern recognition is a good example of the application of this technique to Machine Vision.

Another advanced computer algorithm applied to Computer Vision analysis is the Genetic Algorithms. The idea of Genetic Algorithms is to simulate the way nature uses evolution. It uses Survival of the Fittest with the different solutions in the population. The good solutions reproduce to form new and hopefully better solutions in the population, while the bad solutions are removed.

The Genetic Algorithm is an optimization technique. It is very useful finding the maximum and minimum of a given function or event. In Machine Vision, since images have

two spatial dimensions, it is to be expected that an optimization technique involving images would take much longer than a simple 1D problem. In this case, techniques such as Genetic Algorithms become useful in speeding up the computational process.

REFERENCES

- [1] Bolhouse, V., "Fundamentals of Machine Vision", Robotic Industries Assn, 1997.
- [2] Davies, E., R., "Machine Vision: Theory, Algorithms, Practicalities" (Signal Processing and Its Applications Series), Academic Press, 1996.
- [3] Freeman, H. "Machine Vision". Algorithms, Architectures, and Systems. Academic Press, Inc., 1988.
- [4] Freeman, H. "Machine Vision for Three Dimensional Scenes", Academic Press, Inc., 1990.
- [5] Hussain, Z. "Digital Image Processing". Practical Applications of Parallel Processing Techniques. Published by: Ellis Horwood Limited, 1991.
- [6] Jain, R., Kasturi, R., Shunck, B., G., "Machine Vision (Mc Graw-Hill Series in Computer Science)", McGraw Hill College Div., 1995.
- [7] Myler, H., R., "Fundamental of Machine Vision", SPIE Press, 1999.
- [8] Parker, J., R., "Algorithms for Image Processing and Computer Vision", Wiley Computer Publishing, 1997.
- [9] Parker, J., R., "Practical Computer Vision using C", Wiley Computer Publishing, 1994.
- [10] Ritter, G., X., Wilson, J., N., "Handbook of Computer Vision Algorithms in Image Algebra", CRC Press, 1996.
- [11] Russ, J. C., "The Image Processing Handbook". CRC Press, 1992.
- [12] Sanz, J. L. "Advances in Machine Vision", Springer-Verlag, 1989.
- [13] Shirai, Y. "Three-Dimensional Computer Vision." Symbolic Computation. Springer-Verlag, 1987.
- [14] Sonka, M., Hlavac, V., Boyle, R., "Image Processing, Analysis, and Machine Vision", 2nd Edition, Pws. Pub. Co., 1998.
- [15] Zuech, N. "Applying Machine Vision". John Wiley & Sons, Inc. , 1988.