**Image Processing with MATLAB Part II – Assignment 1**

**Problem 1 – Implement the Skeletonising Procedure**

The main intention is to implement the skeletonising procedure to obtain the skeleton outline within the images. Topological Skeletonisation is a digital image processing technique that removes the foreground region from a binary image through iterations, this results in the skeletal structure for the original image. This method comprises of successive passes of two basic steps applied to the border points of the given region as the border point is any pixel with value 1 and has at one neighbour value.

The algorithm requires the development to test the thinning binary regions through the given images. It assumes that region points have assumed to have value 1 and background points as value 0. A border point refers to any pixel with a value of 1 and at least one neighbour with a value of 0, using the 8-neighbourhood notation as described in the lecture notes. Step 1 identifies contour points for deletion if they meet the following conditions from the question

Preparation

To prepare the implementation for topological thinning, it was beneficial to comprehend the concepts within the process. This dictates whether the current pixels can be removed and are only applicable if all conditions are satisfied.

The specification for each condition is…

1. The sum of non-zero neighbours lie between 2 and 6 then true else false.
2. The number of 0 to 1 transition in ordered sequence from p2 – p8 is true if equal to 1, else false.
3. The product of p2, p4, p6 is true if equal to 0, else false.
4. The product of p4, p6, p8 is true if equal to 0, else false.

When all conditions are met and only once every pixel in the image has checked, then all the flagged points are deleted (changed from 1 to 0).

For step 2, the initial condition A and B are unchanged, but conditions C and D are slightly changed.

The specification for each condition is…

1. The product of p2, p4, p6 is true if equal to 0, else false.
2. The product of p4, p6, p8 is true if equal to 0, else false.

When all conditions are met and only once every pixel in the image has been checked, then all flagged points are deleted (changed from 1 to 0).

MATLAB Implementation

First the variables are initialised within the algorithm that loads the images converted as a double. The Gaussian filter can be used to smooth the image if need and will store as a skeleton to save for future reference. A binarize filter is included for non-binarized images. In addition, the deletion kernel is created also with the same dimensions as the original image. Therefore, the implementation incorporates a “state” as the Boolean operator to control the entry and exit the iterative process that is initially set to true.

Text

Description automatically generated

Figure 1: Image initialised and prepared for skeletonisation.

Top of Form

Initially, the topological process is demonstrated in Figure 2 where the conditions are depicted in the next Figure 2. The algorithm commences with the fixed loop that terminates as the image has been skeletonised. During each iteration, the state begins as false and changes back as the thinning conditions are met. Subsequently, the iterative procedure accesses against the thinning conditions through each pixel within the entire image. Before the evaluation, the neighbouring pixels are stored as an array in the correct sequence to simplify the procedure. Through nested conditional statements, the conditions are checked from A to D where the middle pixel is added to the deletion list when all the conditions are satisfied. As the pixels are evaluated, the pixels in the deletion list is removed. Whereas the procedure terminates if either one of the conditions are not met.

Text

Description automatically generated



Figure 2: Step 1 of the skeletonisation process.

For Step 2, the procedure is recycled from the previous step, however the conditions from C and D are changed to match the topological thinning rules for step 2 from Figure 3. Moreover, the position where the pixel is flagged for deletion as the state changes back to true. Since the thinning has not been complete, this follows the same iteration process to start again.

Text, timeline

Description automatically generated

Figure 3: Step 2 of the skeletonisation process.

The procedure from Figure 4 skeletonised image is output with the original image for a clear contrast.

Text

Description automatically generated with medium confidence

Figure 4: Images shown in a figure side by side

Results

Through the skeletonising procedure, here are the results for the series of test images:

A picture containing logo

Description automatically generated

Figure 5: Skeletonized leg bone image

A picture containing diagram

Description automatically generated

Figure 6: Skeletonized Lincoln image

Diagram

Description automatically generated with medium confidence

Figure 7: Skeletonized Maple leaf image

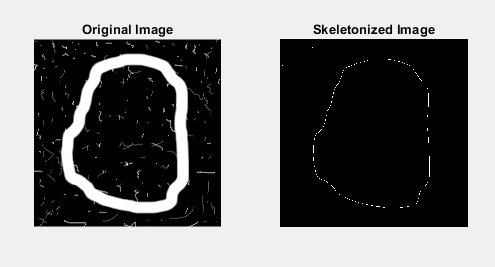


Figure 8: Skeletonized noisy stroke image.

Text

Description automatically generated

Figure 9: Using the Gaussian filtering to take out the noise.

To successfully skeletonise an image with noise, first the noise must be removed through a gaussian filter and binarised to enable the image to be ready for skeletonisation.

A picture containing text, tree, plant, conifer

Description automatically generated

Figure 10: Filtered Chromosome image

A black and white photo of trees

Description automatically generated with low confidence

Figure 11: Skeletonised Chromosome image

A picture containing text, plant

Description automatically generated

Figure 12: Binarised Chromosome image

Figures 10 and 11 demonstrate the process the chromosome image must go through before skeletonisation can occur.

Text

Description automatically generated

Figure 13: Code showing how the image goes through filtering and binarization before skeletisation.

Conclusion

In conclusion, the Skeletonisation process was successful because all the images produced the intended result. Although, the implementation encountered some key issues such as the skeletonised noisy chromosomes produced a skeletonised image compared to the original image. This would have been difficult to recognise the differences without the declared references. In addition, knowledge of the pixel thickness of the image within the procedure was required to determine the appropriate number of iterations for each image. For examples, a high number of iterations made a harder challenge to discover the precise value. Lastly, the images had a slow process in real-time, based on the four nested loops needed for each step that lead a time complexity becoming exponentially high as O(n)4.

**Problem 2 – Implement the Fourier Descriptor Scheme**

The task entails the implementation of the Fourier Descriptor scheme. This is a valuable application for capturing the boundary of an image carrying the information that enables the differentiation between boundary shapes. The number of descriptors should be reduced to a minimum through frequency coefficients to have the shape remain as recognisable. Furthermore, it is important to remember that the descriptors remain unresponsive to change in translation, rotation and scale to maintain accuracy. Since the descriptors are sensitive to these changes, specific parameters can be incorporated to make the descriptor invariant.

Through the Fourier Descriptor table, this revealed the impact for various transformations on the boundary and Fourier descriptors regions. Therefore, modifications were implemented to enable image rotation by parameter input in degrees that converts into radians to rotate the image. The scaling function was introduced to adjust the image size and the translation feature is added to move the modified image in any direction. These adjustments are made to the descriptor that accounts for the transformations in Figure 14.

Text

Description automatically generated

Figure 14: Image loaded in and Fourier descriptor values

The procedure begins with input constants that controls these transformations in Figure 14 and 15.

Graphical user interface, text, application

Description automatically generated

Figure 15: Mod\_ing function 1

The "mod\_img" parameters act as the Nodiscard that determines the number of coefficients that are removed from the original image. Specifically, the coefficients between 12 and -12 are discarded. Therefore, the 2868 boundaries in the original chromosome image are reduced to nearly 0.42% of their original number. As the number of descriptors decrease, the image becomes further removed from the original. The primary objective of the task was to minimise this number as much as possible while able to identify the original image.

Graphical user interface, text, application

Description automatically generated

Figure 16: Mod\_ing function 2

Results

This section demonstrates the impact for decreasing the number of descriptors on the original image with parameter code changes presented at each stage.

Step 1: At 100 descriptors

A picture containing text

Description automatically generated

Figure 17: Fourier descriptor chromosome at 100 descriptors.

Figure 17 resembles the original image that contains the primary distortions and maintains the overall shape.

Step 2: At 15 descriptors

A picture containing shape

Description automatically generated

Figure 18: Fourier descriptor chromosome at 15 descriptors.

The shape appears smoother than before, and the image still presents a human chromosome, evident in Figure 18, with clarity.

Step 3: At 5 descriptors

A picture containing shape

Description automatically generated

Figure 19: Fourier descriptor chromosome at 5 descriptors

There are noticeable similarities with the original image, Figure 19 has an altered shape immensely from the human chromosome.

Final Step: At 12 descriptors

A picture containing shape

Description automatically generated

Figure 20: Fourier descriptor chromosome at 12 descriptors

The research indicates the 12 descriptors as the frequency coefficients are the optimal amount because this preserves the original shape of the image while the sharper details are retained.

Invariance

In image processing, the invariance refers to the property of an image remaining unchanged after undergoing specific transformations such as rotation, scaling and translation. Through experimenting on the invariance for the Fourier Descriptors, this results in the shape outline remaining the same while being affected by any of the transformation.

Rotation and Scaling

As these are simultaneously applied to the image, the modified image rotated by 90 degrees and scaled to 0.5 of its original size. This procedure is shown in Figure 21 where the number of descriptors stays at 12 for the analysis.

A picture containing graphical user interface

Description automatically generated

Figure 21: FD chromosome at 12 descriptors, rotated to 90 degrees and scaled to half its size.

This exhibits the invariance along with the multiple transformations that are applied simultaneously. This creates a problem when rotating and scaling the image to ensure that the modified image was properly centred.

A picture containing text

Description automatically generated

Figure 22: FD chromosome at 12 descriptors, rotated to 90 degrees and scaled to half its size without centring.

The centring problem in Figure 22 can arise during the image transformation such as scaling and rotations. This issue is addressed through Figure 23 that was developed to ensure the modified image remained centred throughout the transformations.

Text

Description automatically generated

Figure 23: centring the FD image code.

As Figure 24 was implemented, it was evident that the image positioning must reset towards the end of the function to ensure it corresponds to the original positioning. This results in a centre modified image.

Text, letter

Description automatically generated

Figure 24: re-centring the FD image code

Translation

This controls the coordinates for the final location of the image can be managed through the Fourier function. Therefore, the function in Figure 25 includes square brackets with the translation parameter that specifies the modified image that should be position 100 pixels from the left and 300 pixels from the top of screen. This results in the image displayed at the bottom right corner of the screen.

Text, letter

Description automatically generated

Figure 25: modified translation code.

The expected outcome where the modified image outline should remain unchanged, and the image should be positioned at the specified location without any alterations for their form or shape.

A picture containing text

Description automatically generated

Figure 26: FD image moved in the translation to the bottom right.