Detection and Characterization of Stone Circles in Permafrost Zones in Antarctica

Francisco Tomás Teotónio Ferreira Soares Pereira

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisor(s): Prof. Jorge dos Santos Salvador Marques
Prof. Pedro Miguel Berardo Duarte Pina

Examination Committee

Chairperson: Prof. João Fernando Cardoso Silva Sequeira
Supervisor: Prof. Jorge dos Santos Salvador Marques
Member of the Committee: Prof. Pedro Manuel Quintas Aguiar

December 2019

Declaration

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Dedicated to my parents

Acknowledgments

I would like to thank my Supervisors Prof. Jorge Marques and Prof. Pedro Pina for the support and knowledge.

To my family who granted me the ideal conditions to complete this path that I will forever treasure.

To the ones I can rely on, in particular, the friends I met at Técnico for the constant support and the moments shared.

Resumo

Nas regiões polares, o congelamento e degelo sazonais do solo levam à criação de círculos de pedra, padrões de terreno naturais cujo estudo permite recolher informação acerca das condições climáticas que no passado motivaram a sua génese.

Estes padrões que ocorrem em conjuntos de milhares de elementos circulares podem ser favoravelmente caracterizados pela utilização de métodos automáticos. O desenvolvimento de métodos automáticos para a detecção e delineação dos círculos de pedra é pela primeira vez estudado nesta dissertação e estes baseiam-se em técnicas de Template Matching, no Sliding Band Filter e em Programação Dinâmica. Os métodos de detecção são testados num conjunto de 20 modelos digitais de elevação de resolução centimétrica obtidos por um veículo aéreo não tripulado numa campanha de campo na Península de Barton, King Georgle Island (Antártida), enquanto os métodos de delineação são avaliados a partir de 24 círculos selecionados de vários locais.

Para o problema de detecção, o melhor compromisso entre desempenho e complexidade foi atingido pela técnica de Programação Dinâmica com um F-score médio de 84.3%.

O método de Programação Dinâmica obteve a menor percentagem média de erros grosseiros na delineação do contorno (7.1%), uma redução de aproximadamente 10% face ao desempenho do método Sliding Band Filter, já que considera restrições na forma dos círculos que melhoram a sua robustez ao ruído do gradiente.

Apesar da variabilidade na forma e tamanho das estruturas únicas analisadas e dado o elevado desempenho obtido, pode considerar-se que o problema apresentado foi abordado com sucesso.

Palavras-chave: Antártida, Círculos de pedra, Elevação, Detecção de círculos, Delineação de círculos.

Abstract

In polar regions, seasonal freezing and thawing of the soils lead to the creation of stone circles, a natural type of patterned ground whose study provides information about the past climatic conditions that motivated its genesis.

These patterns, that occur in clusters of thousands of circular elements, can be advantageously characterized if automated methods are used. The automated methods for the detection and delineation of stone circles are addressed for the first time in this thesis and are based on Template Matching, the Sliding Band Filter and Dynamic Programming. The detection methods are tested in a dataset of 20 digital elevation models of centimetric resolution obtained after surveys with Unmanned Aerial Vehicles in Barton Peninsula in King George Island (Antarctica), while a set of 24 individual stone circles from different sites were selected to test the delineation methods.

For the detection problem, the best compromise between performance and complexity was achieved by the Dynamic Programming technique with an average F-score of 84.3%.

The Dynamic Programming method reached the smallest average percentage of gross errors in contour delineation (7.1%), a reduction of approximately 10% compared to the Sliding Band Filter's performance, as the additional shape constraints improve the gradient noise robustness.

Despite the variability in the shape and size of the unique structures analyzed and given the high-performance achieved, one can consider the proposed problem was appropriately addressed.

Keywords: Antarctica, Stone circles, Elevation, Circle detection, Circle delineation.

Contents

List of Tables

List of Figures

Acronyms

CNN Convolutional Neural Network.

CP Correct Point.

DEM Digital Elevation Model.

DP Dynamic Programming.

FN False Negative.

FP False Positive.

GE Gross Error.

GT Ground-Truth.

HT Hough Transform.

RANSAC Random Sample Consensus.

SBF Sliding Band Filter.

SE Small Error.

TM Template Matching.

TP True Positive.

UAV Unmanned Aerial Vehicles.

Chapter 1

Introduction

1.1 Motivation

In most cold regions, distinct surface patterns can be found in ice-free areas due to the seasonal freezing and thawing of the soils in the active layer of permafrost. These natural patterns appear in different configurations depending on the types and characteristics of the soils, as well as of the environmental variables. Some examples include polygonal networks, stone circles and stripes aligned downslope. Among these, the stone circles are the patterns that draw more attention due to their circular geometry and whose rock elements are separated according to their size, in an unique sorting phenomenon (see Figure 1.1).

The exceptional regular geometry found in these circular structures and the fact that they are currently forming and evolving attracts researchers in studying their genetic process as they can be very helpful to study past climates in the last ten thousand years. Understanding this genetic process has been limited due to the lack of systematic quantitative measurements of physical properties and processes [1], though some long studies and field campaigns have already contributed to understand the periglacial formation processes [2]. Some of the most distinctive sorted patterns expand in equidimensional areas without vegetation and consist on fine-grained soil (from few millimetres to few centimetres) surrounded by curved ridges of gravel. In these regions there can also be observed arrays of merging sorted circles that share common borders.

Figure 1.1: Example of an isolated Stone circle: gently convex upward darker fine-grained soil 1-2 m in diameter ringed by gravel approximately 0.2 m high [2].

As mentioned, the circles tend to be rather uniform in size, with outside diameters ranging from 3 to 4 meters while the outside gravel border is typically 0.5-1 m wide. The internal fine region of the circles is smooth and has a convex upward shape with the highest point in its center. Moreover, the transition between the fines domain and gravel is sharp, both in texture and topography since the gravel ring has a typical elevation of about 0.2 m [2].

The development of microprocessor-based data acquisition systems eased the monitoring of the soil temperature and motion leading to improved information about surface displacement patterns, particular physical states and processes and their evolution throughout the years [2]. Moreover, permafrost research is contributing to evaluate the sensitivity of the polar regions to the ongoing climate changes as well as how do these phenomena affect not only the climate, vegetation and permafrost landscape (ground thermal state [1]) in the Arctic and in the Antarctic, but also the global terrestrial climate in the future [2]. Some authors say that History repeats itself, so future climate changes may result from events that have already occurred in the past, which can be explained by stone circles as they are considered potential paleoclimatic indicators [3].

Monitoring these structure's evolution using data acquisition systems is one of many tools that can be used to improve the understanding on this field together with automated detection and delineation methods that can identify the morphometric properties of stone circles and advantageously replace the current manual measurements. These manual measurements correspond to the majority of the available characterizations of the stone circles yet, they include a limited number of structures [1, 4]. The reduced volume of information provided by the manual annotations determines the need to develop automated solutions that can identify and characterize larger areas containing these unique patterns, from image data acquired by Unmanned Aerial Vehicles (UAV).

1.2 Ultra-high resolution remote sensing

An efficient way to obtain data from relatively large areas with the resolution required to describe the stone circles and their rock elements is by using images acquired by UAVs.

The images containing the stone circles to analyze were obtained by a Portuguese team after a field campaign developed in the ice-free areas of Barton Peninsula, King George Island, in the Antarctic Peninsula (see Figure 1.2), in the frame of a project named CIRCLAR2 [1], with the support of PROPOLAR [2] and KOPRI [3].

The surveys with UAV allowed obtaining image mosaics and Digital Elevation Model (DEM) of milimetric and centimetric spatial resolutions, respectively. For application purposes, working with depth information is more suitable than analyzing raw RGB data as now we are interested in the elevation distribution of the stone circles rather in the detail provided by RGB images. The software used to create the orthorrectified mosaics and the DEMs from the RGB images was *Agisoft Photoscan* [3], based on

[1] Cartografia e monitorização de círculos de pedras ordenados com imagens de ultra elevada resolução na Antártida Marítima, Parte 2

[2] Programa Polar Português

[3] Korean Polar Research Institute

Figure 1.2: Barton Peninsula in King George Island: the location where the images were obtained [3].

SIFT (Scale Invariant Features Transform) and SfM (Structure from Motion) techniques. Examples of a mosaic image and the respective DEM built after the data processing tasks are illustrated in Figure 1.3.

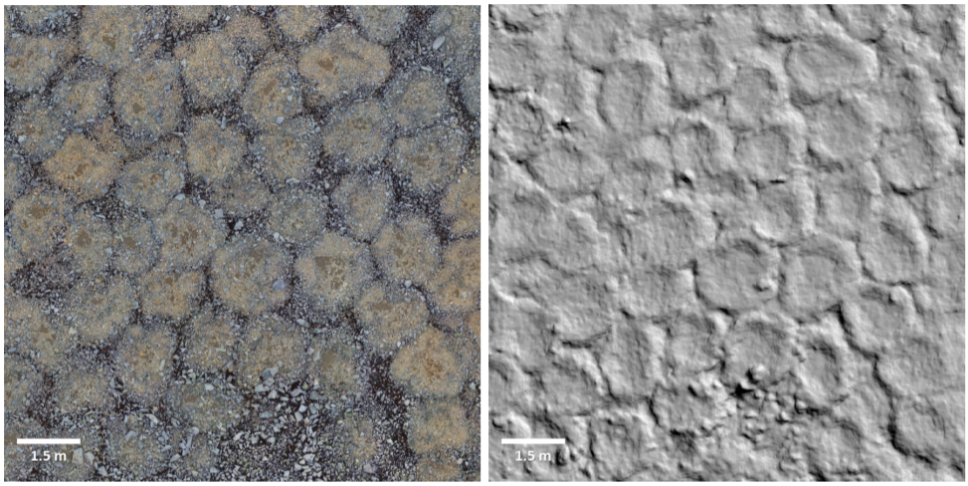


Figure 1.3: Detail of an orthorrectified mosaic (left) and respective DEM (right) [5].

1.3 Objectives and Thesis Outline

This thesis addresses the problem of automated detection of stone circles in DEM and consists in the implementation of the methods based on Template Matching, Sliding Band Filter (SBF) [6] and Dynamic Programming (DP) [7]. Using information from the center's location of each circle, the last two methods are also used to produce an estimate for the stone circles' contour.

This thesis is organized as follows: Chapter 1 presents the relevancy of automated methods for the stone circles' detection and the objectives to achieve with this work. Chapter 2 describes existing techniques for circular shape detection and some examples of their fields of application, including a brief discussion on their implementation conditions. Chapter 3 presents a description of the proposed methods for detecting and delineating the stone circles and in Chapter 4 are presented the experimental results, with a quantitative analysis of the methods' performance, given the model evaluation metrics and the ground-truth information of the structure's location and boundaries. It also includes a discussion on the advantages and drawbacks of each approach. Finally, Chapter 5 shows the conclusions and

possible improvements for future work.

Chapter 2

Background

The problem of object detection constitutes a relevant subject in computer vision, with a permanent search for accurate and efficient algorithms that can infer that kind of information from an image. In spite of the recent growing interest in stone circles' characterization, the problem of their automatic detection has never been addressed before. However, the detection of circular shapes has been widely explored, in a variety of applications that range from microscopy to planetary remote sensing scales.

Depending on the characteristics of the input image and the objects to detect, different techniques have been developed to address the identification task. On the other hand, several applications share main problems affecting object detection, for instance the issues of occlusion, object multiplicity and consequent overlapping, common, for instance, in the segmentation of cells [8] or in the iris detection when its region is hidden by the eyelids [9]. The shape variability is also a major challenge as changes in scale and orientation of the objects within the same image may damage the algorithm's robustness. This issue is present when detecting planetary impact craters whose diameter ranges from few metres to hundreds of kilometres [10] or in the detection of eyes in the presence of head movement and even when the eye is subjected to varying illumination conditions since the pupil size is sensitive to such variations [9] [11]. This chapter describes briefly some techniques developed for circle detection, applied to different input image types and with different complexity degrees.

2.1 Algorithm Overview

2.1.1 Hough Transform-based methods

A well known method for detecting arbitrary object shapes in images is the Hough Transform (HT) [12]. This technique is able to identify not only curves defined by analytic expressions but also arbitrary structures that are not generically parameterized. When dealing with the latter, the method requires an object description in the form of a table that stores its contour information. For both cases of parameterized curves or not, the HT maps the output of an edge detector, also named evidences, to curves in a parameter space. In the case of the detection of circles with unknown radius, each relevant evidence is transformed into a cone whose base verifies the circle equation centered at the edge point and

with a height given by the possible radius values (see Figure 2.1). The model parameters are obtained by a voting scheme since the edge points belonging to the objects boundary would correspond to the intersection of multiple cones or equivalently, to a peak in the vote accumulator.

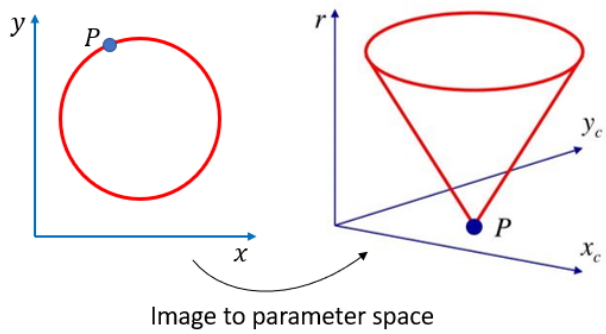


Figure 2.1: The edge point P is mapped into the cone surface in the parameter space, whose circular base is centered at P.

Improved versions of this method are scale and orientation invariant and they are fairly robust to occlusions as well, since the parameters are selected through the maximization of the accumulator and it is not required for all edge points to be detected.

Rad et al. (2003) proposed a similar approach to the HT that is based on the symmetry of gradient vector pairs on a circle's boundary [13]. In this fast circle detector, a pair of gradient vectors votes to a parameter triplet (center coordinates and radius) but only the vectors that verify geometrical conditions concerning their symmetry contribute to the accumulator. The biggest inconvenient of such solution is the requirement of a large intensity difference between the circle's region and the background. Silveira (2005) also applied the HT for the detection of concentric circles [14] yet, instead of searching in a 3D parameter space, she separates the problem in first detecting the center of the concentric circles and then estimating the radii. By selecting three edge points belonging to the same connected component, she computes the center of the circle that passes through those points. If the geometrical conditions that evaluate the concentric property are met, its coordinates get a vote in the accumulator. For the radii estimation, a voting scheme is also performed: the edge points and the center coordinates (a, b) that previously maximized the accumulator contribute with a vote for a radius value, r, using the equation

$$(x - a)^2 + (y - b)^2 = r^2. \tag{2.1}$$

The two peaks detected in this 1D accumulator space correspond to the inner and outer radius. This approach is robust to noise and occlusion.

Another example of a method based on the HT is the one proposed by Garrido and De La Blanca 2000 [8]. They developed a reformulated HT whose input are post-processed edge points organized into straight line segments. The method approximates each elliptical cell by a circle formed by 8 line segments and the accumulator measures the ratio of the contour detected by analyzing the length of each segment. They also assume a fixed radius for these "circle" and length of each line of the model. The selection of the accumulator peaks produces the cells' center coordinates. The boundary of the

cells are refined through an ellipse approximation method, by fitting local deformations to their coarse estimation.

Since the HT relies on edge information, its application implies specific fine-tuning pre-processing tasks depending on the image conditions and the complexity of the background. Moreover, in case of significant image noise, the peak detection of the accumulator may not be straightforward and its processing time and memory requirements increase with the number of parameters to compute [8].

2.1.2 Template Matching methods

Template Matching [15] is also a widely and commonly explored technique in pattern recognition. It processes the input image in the search for regions most similar to the template of interest. Bandeira et al. (2007) propose a Template Matching method for detection of different size impact craters by using a similarity measure that explores the frequency components of the template and the sensed image [16]. The method achieved favorable results when compared to other solutions yet, the number of intermediate steps with fine-tuning of parameters prevents its application to a broader set of planetary images with different acquisition conditions and spatial resolutions.

In addition to the Fourier methods, the Template Matching technique includes the correlation-like methods. Kim et al. (2005) used the correlation operator to verify the similarity between the detected crater candidates and predefined templates with several different radius [17]. However, selecting the craters whose correlation value was sufficiently high resulted in the incorrect elimination of those with irregular features such as eroded rims. This confirms the drawback of the Template Matching technique related to the lack of robustness when the input image presents deformed and particularly scaled versions of the template. Although not so severe when detecting circles, this method is also prone to detection failures in the presence of rotated instances of the template.

2.1.3 Random Sample Consensus

In an attempt to tackle the HT's demanding complexity and storage requirements, Lamiroy et al. (2007) showed that Random Sample Consensus (RANSAC) is a robust alternative technique for fitting a circle to contaminated data in line drawing images [18]. The method estimates the optimal circle parameters by randomly selecting 3 samples and chooses the model that minimizes the median of the residual error of the remaining data points to this model. The samples used by RANSAC are the points belonging to circular arcs that result from a segmentation method. In order to eliminate not fully detected circles, they project the obtained circle in the original image and it is validated if the ratio of contour points belonging to the circle is sufficiently large. Another filtering step consists in discarding multiple instances of the same circle by verifying the distance separating both circle center points and radii.

Stache and Zimmer (2007) applied RANSAC to determine the radius and position of the melt pool in a real-time monitoring system of a laser welding process [19]. They first detected the melt pool's contour points by analyzing the radial profiles in each frame, whose output was applied to the model fitting algorithm. This method achieved higher rates of throughput (above 200 frames per second) and

good estimates as the images only included an individual model to estimate.

Although RANSAC is generally robust for fitting models to data containing an outlier percentage up to 50%, its main disadvantages are related to the number of hyperparameters to tune in the estimation process and the excessive number of iterations or model hypothesis to test, particularly when the samples' inlier probability decreases. Moreover, to compensate for the lack of precision, the model parameters obtained are often recomputed by applying, for instance, the least squares algorithm to the largest inlier set resulting from RANSAC [20].

2.1.4 Deformable templates

Besides the HT, other techniques assume a parameterized model that defines the object to identify while the image information guides its location and shape computation. Despite considering a prior expected structure, the deformable template techniques accommodate local variations [11] [21]. For instance, Yuille et al. (1992) proposed an eye model described by 11 parameters and an iterative optimization procedure that adapts them to the image, specifically to relevant regions such as peaks, valleys and edges [21]. After the parameters are initialized, by choosing the coefficients that weight each function to minimize, it is possible to control which parameter is fine-tuned in each iteration. However, one drawback of this approach is the dependency on the parameter initialization since the optimization can lead to local minima.

Daugman (2003) developed an iris biometric system where the iris parameters (center and radius) were obtained by the maximization of the curve integral of the smoothed gradient magnitude along a circular arc [22]. This method may fail to locate the pupil given the corneal reflections that produce a perfect circle with a very high intensity level inside the dark pupil region. Since the method looks for a contour that passes through pixels with high gradient magnitude, a minimum radius must be set for the pupil detection or else this bright small circle would correspond to the curve that maximizes the optimization function.

2.1.5 Feature-based methods

Instead of trying to fit a predetermined model to the image like the aforementioned methods, the feature-based methods rely on the detection and location of image features such as image or gradient intensity that express the presence of the object.

The symmetry of the image gradient vectors is explored in the SBF method [6], for the segmentation of the nuclei and cells' cytoplasm. This technique evaluates the gradient's orientation in a region centered in the cells by comparing it with the ideal radial structure and searches for the pixels that maximize the similarity between those, expressed in the filter's response. This criteria also allows the computation of the boundary of the cells and performs well on low contrast images since it relies on the gradient convergence and not magnitude. Moreover, it is robust to scaling and orientation changes and depends on intuitive parameters related to the expected object size.

Marques and Pina (2015) proposed a delineation method for craters based on Dynamic Programming

[7]. This approach processes the image in polar coordinates, assuming the crater's center is known, to simplify the contour search as in this representation it can be approximated by a line. Just like the method in [6], the intensity transitions along radial directions diverging from the crater's center are explored. The crater's boundary is obtained from the minimization of an energy functional to accommodate small deformations yet to limit the overall shape to be circular. Merging the *a priori* knowledge about the crater geometry with the enhanced intensity variation information obtained by the edge map resulted in the high performance of the method. Additionally, the reduced number of the hyperparameters in the optimization problem and their intuitive choice allows the adaptation of this delineation method to other remote sensing applications, even in the absence of reliable assumptions on the edge search area given by an estimate of the circle's radius.

Other methods rely simply on the pixel's gray-level values. Thresholding [8] is a basic approach that separates the object's region from the background through a comparison with a given level yet, it is rarely applied on its own because the object and background may share common intensity values. A more robust technique that also explores the image intensity distribution is proposed in [10] for the delineation of craters. Marques and Pina (2013) analyzed the regular patterns that appear in these gray-level images characterized by larger brightness differences between the left and right parts of the craters. The original image is converted to polar coordinates and they compute the difference in the intensity levels along the same direction, for pixels with the same distance from the crater's center, obtaining a coarse estimation of the radius of the crater. The boundary is then refined by inspecting the gradient magnitude for each direction and by linear interpolation since parts of the rim may be eroded or the edges may not be detected.

In order to address the variability in the planetary images, Martins et al. (2008) developed a method for the detection of different size craters [23] based on the Adaboost algorithm (Viola and Jones), built for facial recognition. Being a feature-based detection method, this solution determines whether the image block is a crater or not depending on the output of a bank of weak classifiers, each one verifying if the extracted feature is relevant enough. The features are the rectangular Haar-like features that explore intensity variation patterns. This method's efficiency is due to the use of the integral image that allows the feature computation using only 4 to 8 addition operations. It is also scale invariant since the features are normalized to the block size and blocks of different sizes are tested. Computing the model (threshold) that minimizes the classification error for a given classifier implies minimizing a weighted sum (for all training samples) of the absolute differences between predicted and actual labels, where each sample weight depends on the previous classifier output. When updating these weights, the misclassified training blocks are given a greater weight, so the next classifier's prediction reduces their classification error. This process may lead to overfitting in the presence of noise [24].

2.1.6 Deep Learning

More recently, Deep Learning algorithms have become extremely important in computer vision tasks, mostly due to the high performances achieved and the availability of increased computational power,

memory storage capacity and annotated data. Falk et al. (2019) proposed a Convolutional Neural Network (CNN) architecture for semantic segmentation and detection for both 2D and 3D multi-channel biomedical images [25]. From an input image block, the network extracts hierarchical features by determining the convolutional layer's filter weights. The encoder creates feature maps with increasing depth as the number of filters (feature channels) doubles: the early layers extract low-level information such as edge orientation while the deeper layers can determine more abstract patterns. This down-sampling process decreases the spatial resolution of the image in exchange for learning more about it. In order to recover the spatial resolution, the layers in the decoder perform an up-sampling operation by using the low-level feature maps where the spatial information is present to generate successively larger blocks until the segmented image with 2 channels is created. The network outputs an image where each pixel is classified into "background" or "foreground" and is able to detect touching objects of the same class. It is invariant to the size of the input image and, in order to teach shape variations to the neural network from few annotated images, a data augmentation method based on deformation vector fields was implemented.

Maggiori et al. (2016) propose a framework with CNNs to perform a per-pixel classification for satellite imagery for precise agriculture and urban planning [26]. They adapted an existing architecture in order to overcome the issue of fuzzy object borders in the classified image and also improve the execution time for classification.

Despite the existence of pre-trained networks that allow the adaptation to new applications, the large amount of image data and corresponding annotation (for instance, counting of thousands of cells or drawing their outlines) required for their training remains a drawback in the application of such methods for the stone circle characterization.

2.2 Final remarks

The first solution proposed in this work for the stone circles' detection is based on the Template Matching technique, given its simplicity and easy implementation. An alternative method based on the Sliding Band Filter (SBF) [6] is implemented for both the detection and delineation problems, as it accommodates scale changes and shape deformations. Finally, a Dynamic Programming based technique [7] is developed, which uses a similar criteria for the stone's segmentation as the SBF, yet it considers additional constraints to improve the latter's performance.

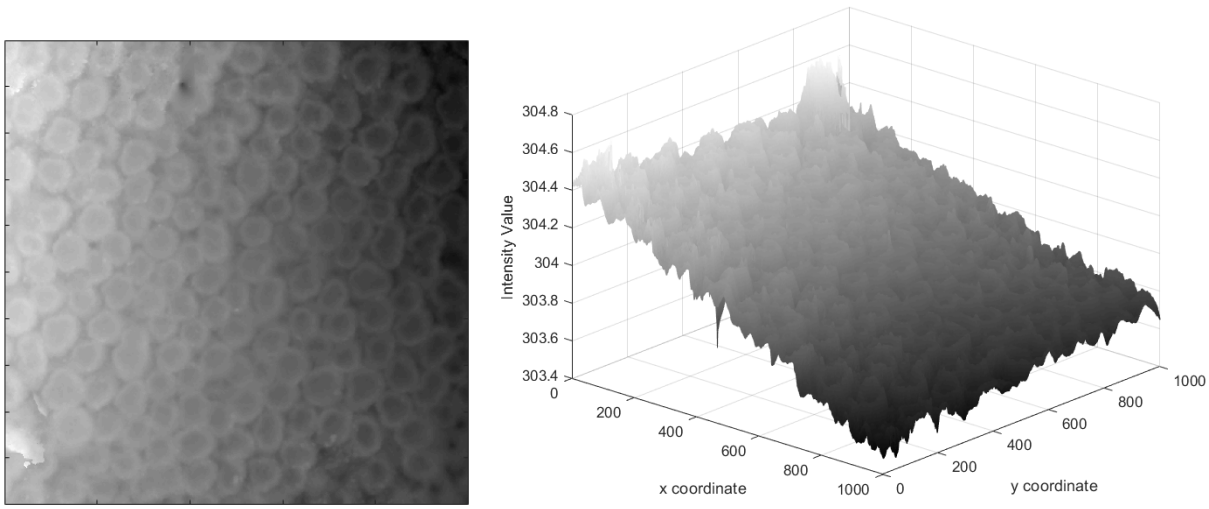
Chapter 3

Stone Circle Detection and Delineation

This chapter describes methods to detect and delineate stone circles in digital elevation maps.

3.1 Pre-Processing

The input image, I, is assumed to be a Digital Elevation Model where brighter pixels, with higher intensity values, correspond to regions with a higher elevation while darker, low-valued pixels, correspond to low elevation regions. Figure 3.1(a) represents the input image, I, and Figure 3.1(b) the corresponding 3D model, where the vertical axis expresses the elevation measured from the stones' surface. These stone circles often appear scattered across inclined terrains, hence the natural slope visible in Figure 3.1(b): the left side of the image is bright (high elevation) and it gets darker to the right, as the elevation decreases.



(a) Input Image, I.

(b) 3D model of I.

Figure 3.1: Digital elevation image, I, and the corresponding 3D surface.

In order to simplify the detection of the stone circles, the input image is processed to remove the slope of the terrain. To achieve this result, a high-pass filter is applied to the input image, whose imple-

mentation is described by the block diagram in Figure 3.2.

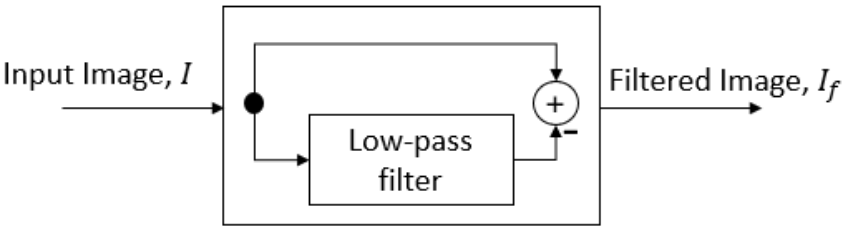


Figure 3.2: Pre-processing's block diagram: high-pass filter.

This high-pass filter is implemented using a low-pass one, which captures the undesired natural slope, which is then subtracted from the input image.

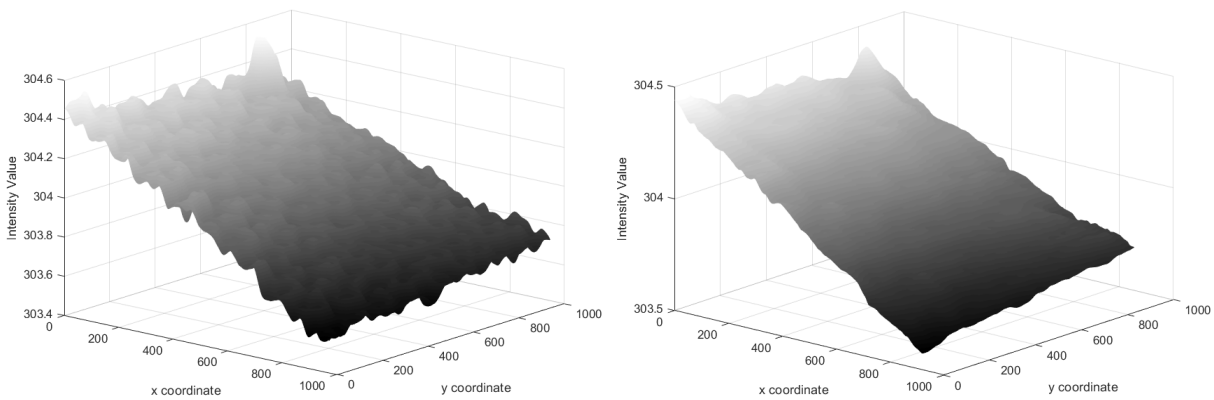
To obtain the slope, the input image is convolved with a filter mask H with dimensions $(2N+1) \times (2N+1)$, according to

$$Y(x,y) = I(x,y) * H(x,y) = \sum_{i=-N}^{N} \sum_{j=-N}^{N} H(i,j) I(x-i, y-j). \tag{3.1}$$

The filter mask used is the averaging filter

$$H = \frac{1}{n^2} \begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & 1 & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \dots & 1 \end{bmatrix}, \tag{3.2}$$

where $n = 2N+1$ controls its degree of smoothness (cut-off frequency): a smaller N may still preserve some sharp variations of the sorted circles, failing to estimate the slope; on the other hand, increasing N eliminates those high-frequency variations yet, creates parallel horizontal stripes along the direction of the slope. These two cases are shown in Figure 3.3.



(a) $N = 15$. (b) $N = 40$.

Figure 3.3: Output of low-pass filter for two values of N.

Given this trade-off, the image obtained after the convolution with $N = 25$ was chosen. After obtaining the slope shown in Figure 3.4, the filtered image, I_f, is given by computing the difference between I and

the low-pass filter output.

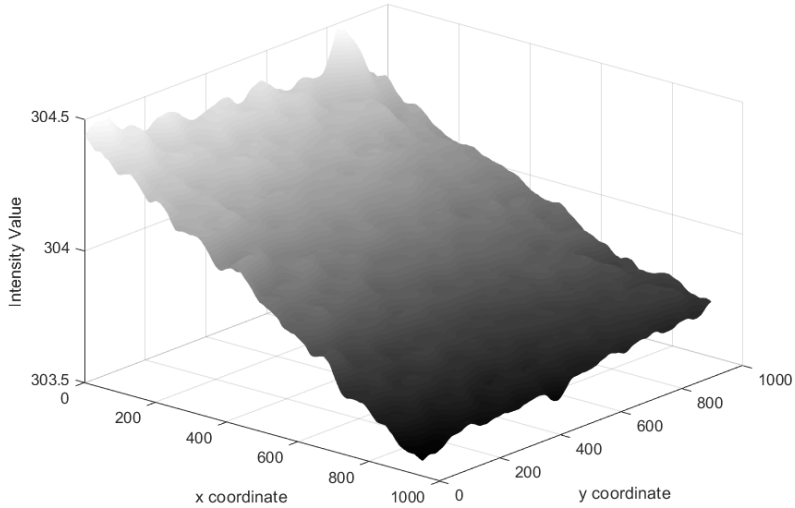
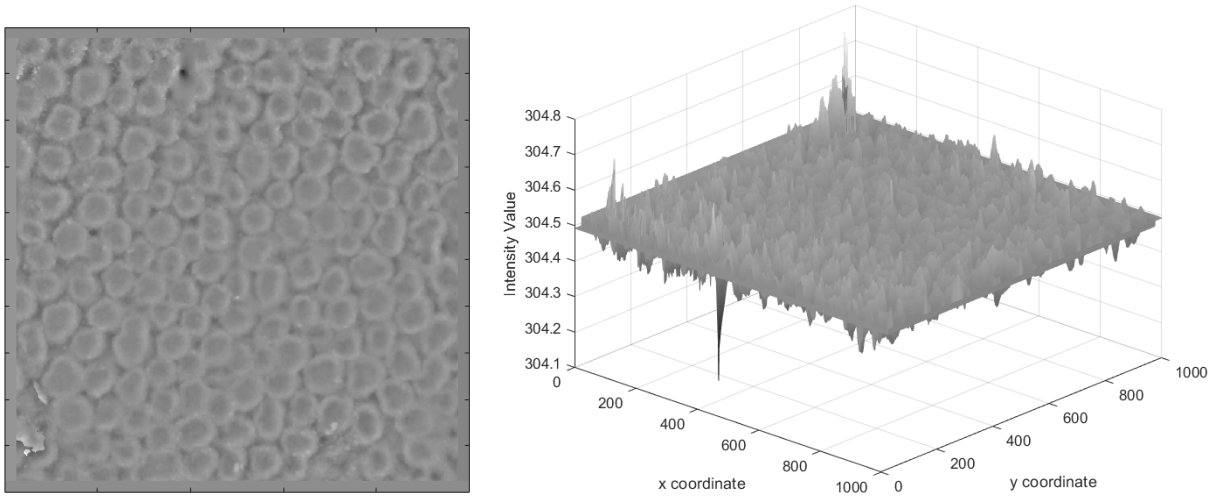


Figure 3.4: Output of low-pass filter for $N = 25$: slope is captured but there are still some acceptable sharp shapes kept ”unsmoothed”.

The filtered image shown in Figure 3.5(a) is the high-pass filter output. Figure 3.5(b) shows the corresponding surface image, where the stone circles to detect are displayed across an approximate horizontal plane.



(a) Filtered Image, I_f.

(b) 3D model of I_f.

Figure 3.5: High-pass filter output: slope removal.

The pre-processing operation preserves the shape of the stone circles, as their central region remains lower than the surrounding ring, in accordance with the image's intensity distribution. As expected, the brightness difference across the image is also eliminated. From now on, the filtered elevation map will be used and denoted by I, for the sake of simplicity.

3.2 Template Matching

In order to identify the stone circles' center location in the input image, a Template Matching algorithm is proposed. As the name suggests, the algorithm consists of searching across the input image for regions that most resemble the template T. Figure 3.6 shows the block diagram of Template Matching algorithm.

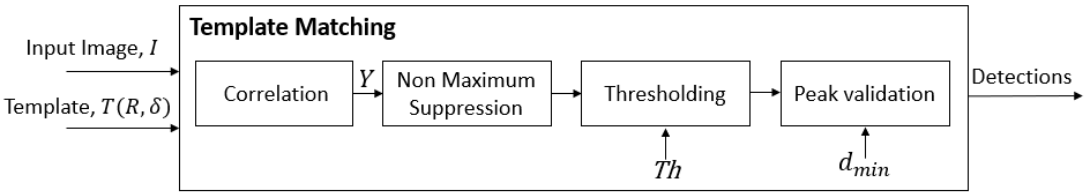


Figure 3.6: Block diagram of Template Matching algorithm. R, δ, Th and d_{min} are parameters to be specified.

In order to evaluate the similarity between the image and the template, there are several comparison metrics that quantify the degree to which the pixel values of I and T coincide, such as the Euclidean distance or Cross-correlation or even more complex ones like the Cosine coefficient [27]. Regardless of the comparison's operation chosen, an output image, Y, is generated, indicating how well the template matches the image in the vicinity of each pixel. In this thesis, the cross-correlation was selected and the pixels of Y corresponding to the most likely locations of the stone circles are assigned to higher values. For this reason, the output image is also referred to as score image. After obtaining the score image, a post-processing stage is implemented in order to extract the peaks associated to candidate stone circle centers. The image is filtered by the non-maximum suppression algorithm to extract the image's local maxima, followed by a selection stage where these local maxima are compared with a threshold. At this point the candidate peaks are computed yet, these are filtered by a peak validation method (spatial filter), in order to avoid multiple detections in a small neighborhood region.

This chapter is organized as follows: first, it describes the creation of the template image and the computation of the score image. Then the post-processing stages (non maximum suppression, thresholding and peak validation) are discussed.

3.2.1 Template creation

The template image is obtained considering the shape and dimensions of the stone circles: most of these have a regular circular shape, with an approximately constant radius. The structures to identify have a half torus shape, characterized by two parameters: the radius R i.e. an integer representing the distance from the template's center to the highest point in the ring, and the width, δR, which defines the concave shape of the template $(\delta \in [0, 1])$.

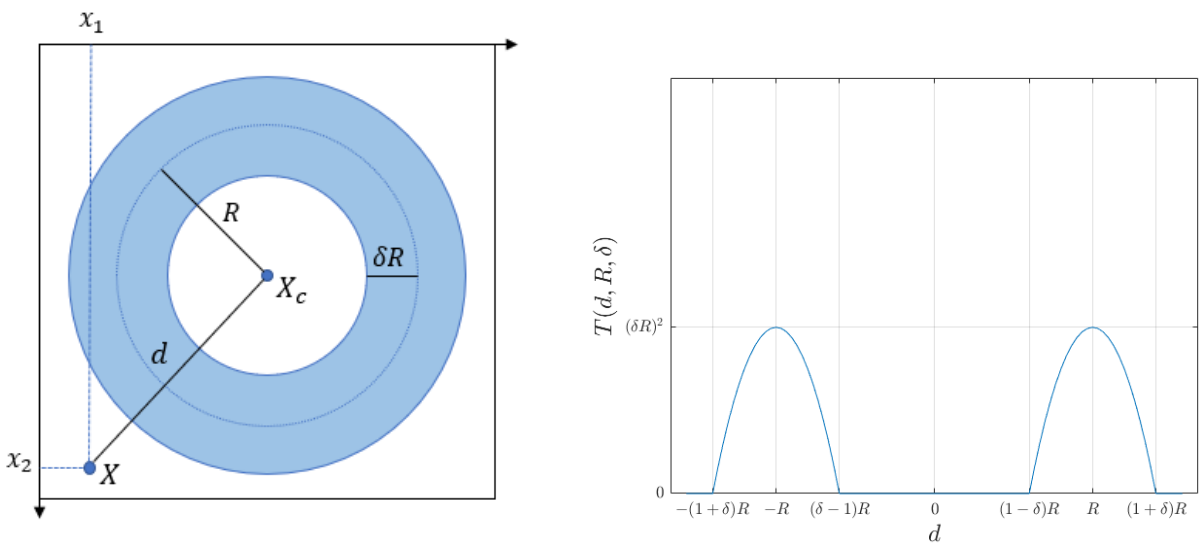
The template is a symmetric matrix with a number of rows given by $(2R(1 + \delta) + 1)$. The intensity value $T(x_1, x_2)$ depends on the Euclidean distance $d = \sqrt{(x_{c_1} - x_1)^2 + (x_{c_2} - x_2)^2}$ between the pixel

$X = [x_1, x_2]^T$ and the torus' geometrical center, $X_c = [x_{c_1}, x_{c_2}]^T$, according to

$$T(R, \delta) = \begin{cases} (\delta R)^2 - (d - R)^2, & d > \mid (1-\delta)R \mid \vee d < \mid (1+\delta)R \mid \\ 0, & \text{otherwise} \end{cases} \tag{3.3}$$

If d is exactly R, that pixel corresponds to the highest point of the template, with its intensity set to $(\delta R)^2$. For the pixels whose distance to X_c belongs to the interval [$\mid (1-\delta)R \mid, \mid (1+\delta)R \mid$], the intensity value decreases with the square of the distance from the pixel's location and R. Finally, the value zero is assigned to the remaining pixels.

Figure 3.7 shows a top-view and a cross-section detail of the template's image defined by equation (3.3).

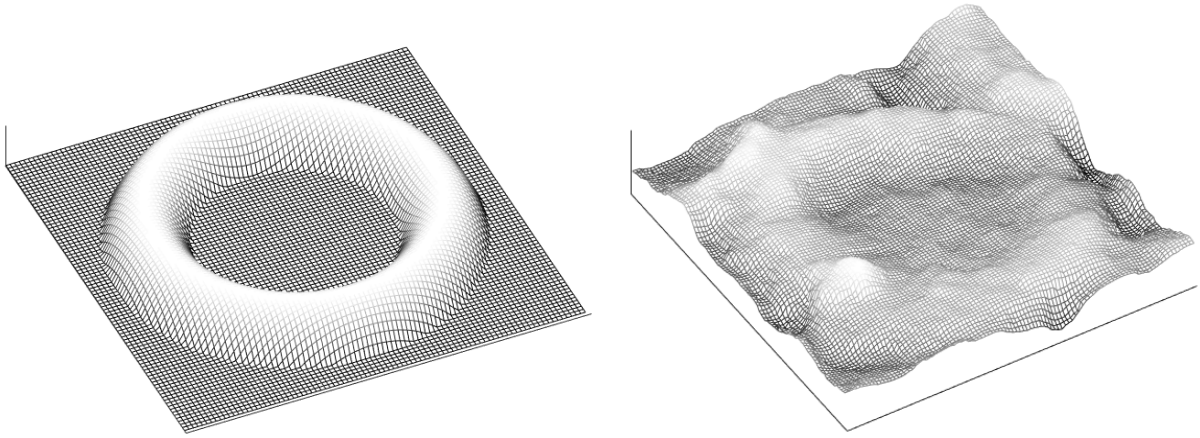


(a) Template's top view.

(b) Template's cross section.

Figure 3.7: Template's 2D representation.

In order to mitigate the influence of brightness variations across the image, possibly caused by different illumination conditions, the mean value of the template was subtracted from each pixel to get a zero-mean image. By doing so, the convolution removes the DC level of the input image. A 3D representation of the normalized template is shown in Figure 3.8(a), as well as the elevation map of a real stone circle in Figure 3.8(b).



(a) Template's 3D representation.

(b) 3D detail of a real stone circle.

Figure 3.8: Template and an example of a stone circle: in spite of its extremely regular shape, the template describes the overall appearance of the structures to identify.

As shown in Figure 3.8, the real stone circle corresponds to a deformed version of the template.

3.2.2 Score image computation

After defining the template, the score image is obtained by computing the correlation of the input image I with the template T,

$$Y(x,y) = \sum_{i=-N}^{N} \sum_{j=-N}^{N} T(i,j) I(x+i, y+j). \tag{3.4}$$

Figure 3.9 shows the score image Y obtained for a template with parameters $R = 30$ and $\delta = 0.4$.

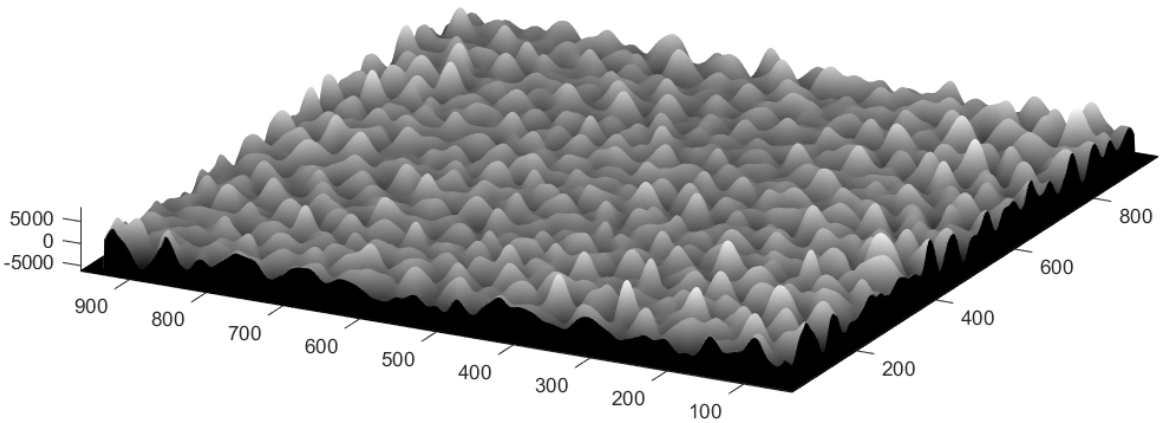


Figure 3.9: Score image obtained by correlation of I with the template T.

The score image contains many bell-shaped peaks (see Figure 3.9) that correspond to the center of stone circles. Peaks with high amplitude have a high degree of confidence while peaks with low amplitude are associated to a low degree of confidence.

In order to determine the locations of the stone circles from the peaks of the Score image, the Non-Maximum Suppression algorithm is used. This method inspects a $n \times n$ patch of the image (n odd),

centered at every image pixel. If the current pixel has the maximum value among the pixels of the patch, it is preserved; if this condition is not verified, the zero value is assigned to the current pixel. This method is repeated for each pixel of the input image and allows the identification of the pixels with highest values in their vicinity. Figure 3.10 shows the output of the Non-Maximum Suppression algorithm. Peaks that survived are shown as green dots superimposed on the score image.

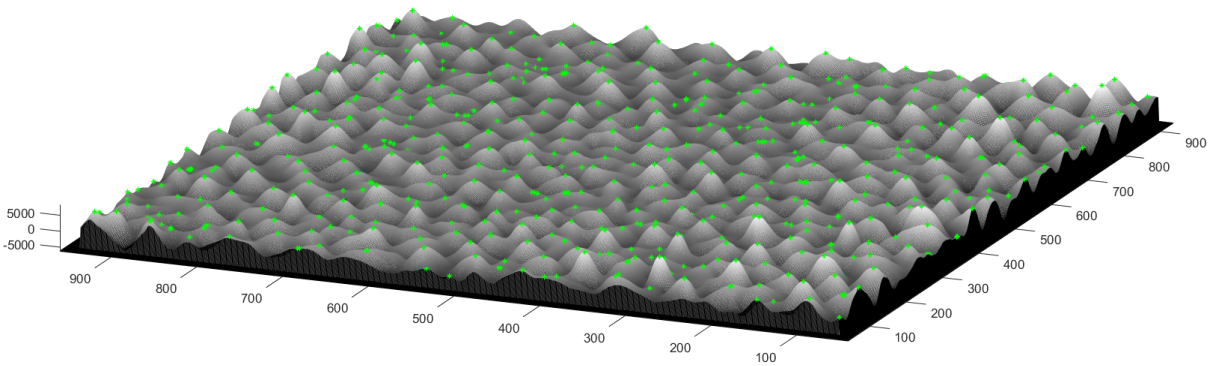


Figure 3.10: Non Maximum Suppression output: green markers represent the local maxima.

The local maxima of the Score image were obtained after the Non-maximum Suppression method with $n = 3$ and correspond to the peaks of the bell-shaped surfaces, as shown in Figure 3.10.

Having the location of the pixels that correspond to a higher probability of existence of a sorted circle in the image, it is necessary to filter these detections since not all of them correspond to a real existing circle. One way to discard invalid detections is by evaluating their score value: the detections with a sufficiently high correlation value are more likely to being a stone circle than those with a lower score. This approach suggests a comparison between the local maxima and a threshold Th, chosen as a percentage of the maximum value of the score image. It is expected that for higher thresholds, the number of detections gets reduced since only the ones with the highest score are selected. For lower thresholds, the number of detections is higher since there is not a significant penalization of the intensity value. Ultimately, this threshold determines the tolerance of the local maxima selection and is an important parameter affecting the performance of the detection method. Figure 3.11 exhibits several green pixels whose score value is greater than the threshold, corresponding to the height of the red horizontal plane.

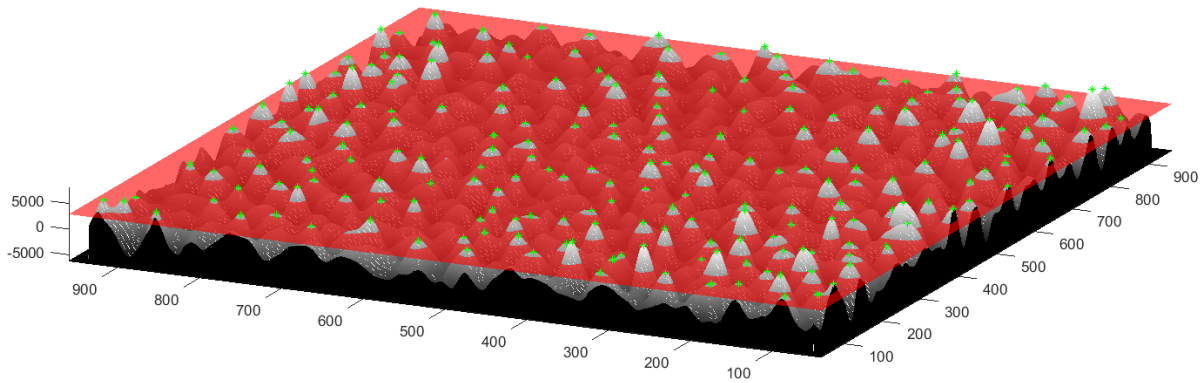


Figure 3.11: Thresholding output: the green marked pixels have a score value above the minimum required, represented by the red horizontal plane $z = Th \times \max\{Y\}$.

When comparing Figures 3.10 and 3.11, it is visible that the thresholding stage eliminated several false candidates. However, there are candidates that are still too close to each other to be accepted as the center of stone circles. To correct this issue, a validation step is performed, by filtering the candidates taking into account not only their score image value, but also the distance between them.

3.2.3 Peak Validation

The validation method described in this section relies on the score image, Y, and the accepted detections' location, after the thresholding process. The following procedure was adopted: first, the candidate points are ranked according to their score value and the candidate with the maximum score is included in the final set of detections, D. Then, the candidate with the second highest score is evaluated: if the distance from this candidate to the already validated detection is greater than a minimum distance, d_{min}, the current candidate is also inserted in D; otherwise, it is not accepted as a valid detection. This process is repeated for the remaining candidates. In each iteration a distance matrix is created, whose elements are the Euclidean distance from the current evaluated candidate to all accepted candidates in D. The closest detection to the current candidate is identified, corresponding to the minimum entry of this distance matrix, and the comparison with d_{min} is tested.

The value of the minimum distance parameter, d_{min}, should be greater than the average stone circle's radius, otherwise there could be accepted detections located near the stone's boundary. However, d_{min} must be smaller than twice the stone's radius, or adjacent detections would be discarded. Figure 3.12 presents the accepted candidates as green, while the red ones represent the location of the removed candidates. This task was performed using $d_{min} = 50$.

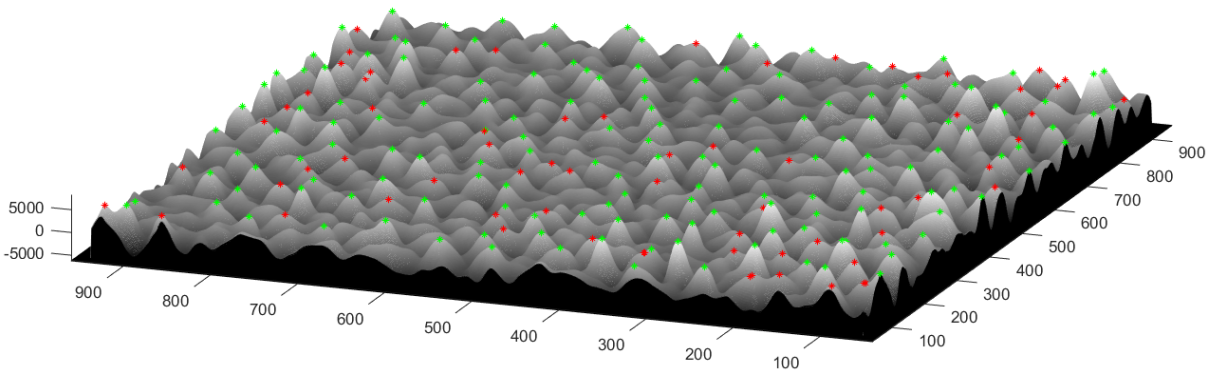


Figure 3.12: Peak validation: accepted peaks are represented by green markers and discarded peaks are represented by red markers.

Figure 3.13 depicts the validated and eliminated detections shown in Figure 3.12 superimposed with the input image. It reveals that most accepted candidates (green) are located approximately at the stone circles' center and many discarded detections are located in the intersection of stone circles.

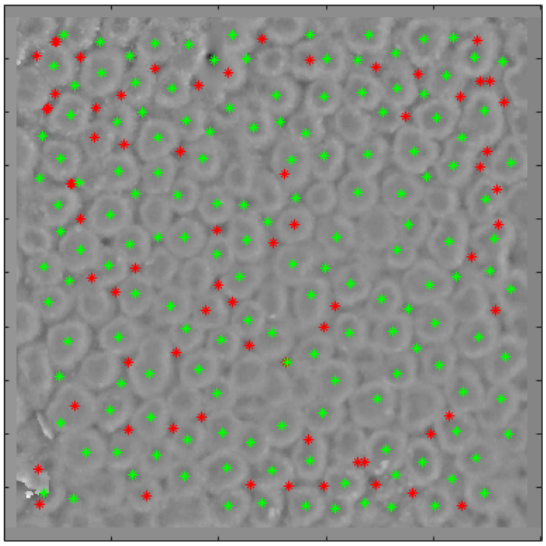


Figure 3.13: Input image superimposed with the final candidate centers (green). Most red detections were correctly eliminated, as they correspond to the intersection of stone circles.

The evaluation of the Template matching method is described in Chapter 4. It will be concluded that this method has a number of drawbacks. Therefore, an alternative algorithm was considered, based on the gradient of the image in order to improve robustness.

3.3 Sliding Band Filter

The Sliding Band Filter (SBF) is a method for the detection of circular shapes based on the gradient of the image [6]. This method assumes the gradient has a radial structure in the vicinity of a circle. By evaluating the similarity between the gradient direction and the direction of auxiliary radial lines (see Figure 3.14), this method assigns a score to each image pixel. The circle's center corresponds to the point with the highest score, as the alignment of the radial lines and gradient vector's directions is maximum.

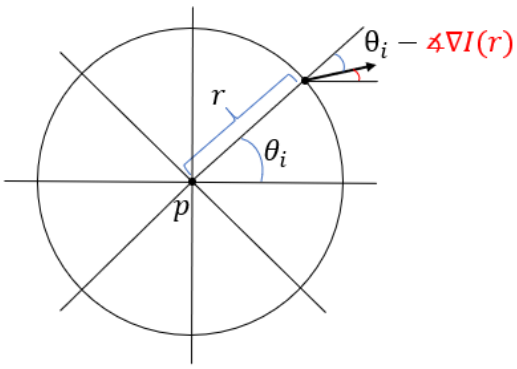


Figure 3.14: Alignment between the gradient direction, $\measuredangle\nabla I(r)$, and the direction of auxiliary radial lines, θ_i, in the vicinity of the circle's center.

Unlike the Template Matching algorithm, the SBF method allows the detection of different-sized circles as it is robust to scaling. The SBF method also performs well on low contrast images since it relies on the gradient's direction and not its magnitude.

In order to obtain the stone circles' center location, the detection method shown in Figure 3.15 is implemented.

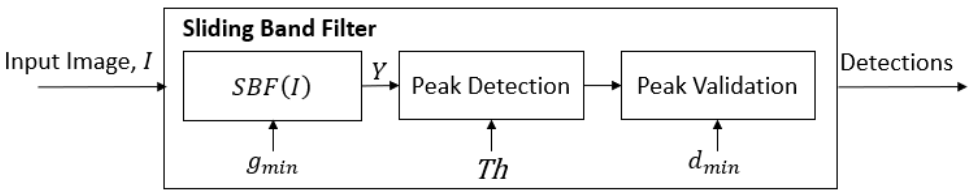
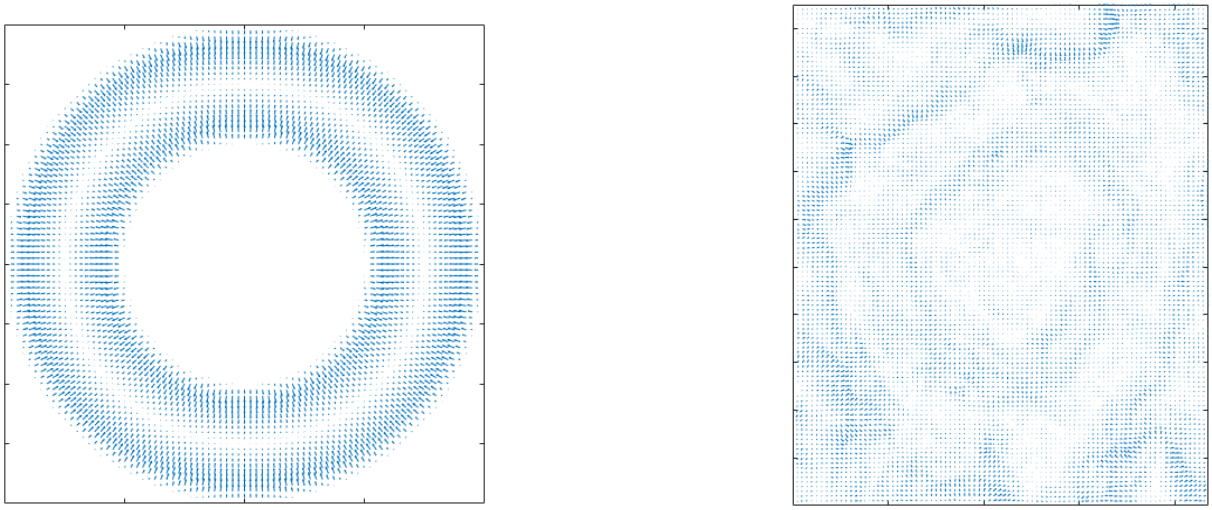


Figure 3.15: Block diagram of Sliding Band Filter algorithm.

The blocks presented in Figure 3.15 are identical to the ones in the Template matching's method, with exception of the computation of the score image, Y. For the SBF method, the score image is the result of evaluating the alignment between the image gradient and a radial structure in the support region, the annulus centered on each pixel of the input image. The local maxima of the score image correspond to the regions where this gradient's direction distribution is most identical to the desired radial structure obtained for a circle's center. After the score image is obtained, the procedure to locate stone circle centers is composed of a peak detection stage, which includes a non maximum suppression operation, followed by thresholding and peak validation stages, as before.

3.3.1 Alignment score definition

The Sliding Band Filter has been proposed in [6] for cell detection in microscopy imaging. These cells present a gradient vector with a radial structure pointing in an inward direction. However, when addressing the problem of stone circle detection, the structures to identify are characterized by a gradient field that points outwards and inwards, as the distance to the center increases (see Figure 3.16). This fact implies the need of adapting the method presented in [6] to the stone circle identification, preserving the main idea of the method.



(a) Synthetic circle's gradient field.

(b) Stone circle's gradient field.

Figure 3.16: Gradient field distribution: inwards and outwards radial field.

By comparing the gradient vector distributions of the synthetic stone circle in Figure 3.16(a) with the real one's in Figure 3.16(b), the smoothness contrast is visible, both in the vector's magnitude and orientation. Although the stone image presents a distorted radial structure, it is still noticeable the highest intensity variations corresponding to the interior and exterior borders of the circle, clearly not as evident as the ones of the artificial circle. In Figure 3.16(b) it is also worth noting the existence of non-null gradient vectors in the interior region of the stone circle and across the highest region of the torus, suggesting the irregularity and deformation of the structure.

In order to detect a circle's center, all image pixels $p = [x, y]^T$ are considered. Let us define N radial lines emerging from a given center point p, and let $\theta_i = 2\pi i/N$ be the direction of the i-th line, $i = 0, ..., N-1$. The alignment between the i-th line's direction and the image gradient direction computed at point

$$q = p + r[\cos(\theta_i), \sin(\theta_i)]^T \tag{3.5}$$

is given by

$$A_i(r) = \cos(\theta_i - \phi(r)), \tag{3.6}$$

where $\phi(r)$ denotes the direction of the gradient vector at point q , with $R_{min} < r \leq R_{max}$. Figure 3.17 displays a schematic representation of the alignment computation.

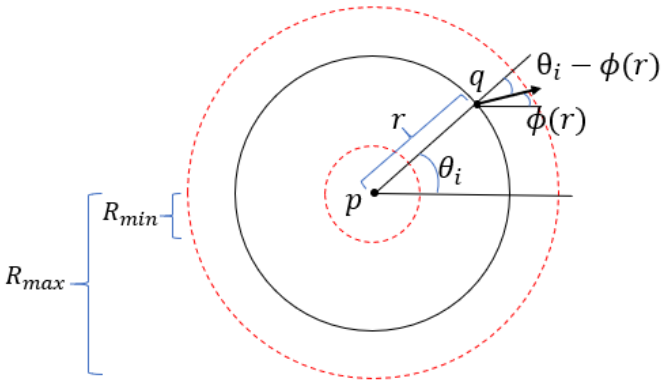


Figure 3.17: Alignment computation at point q, given the i-th line's orientation, θ_i, and the gradient direction, $\phi(r)$.

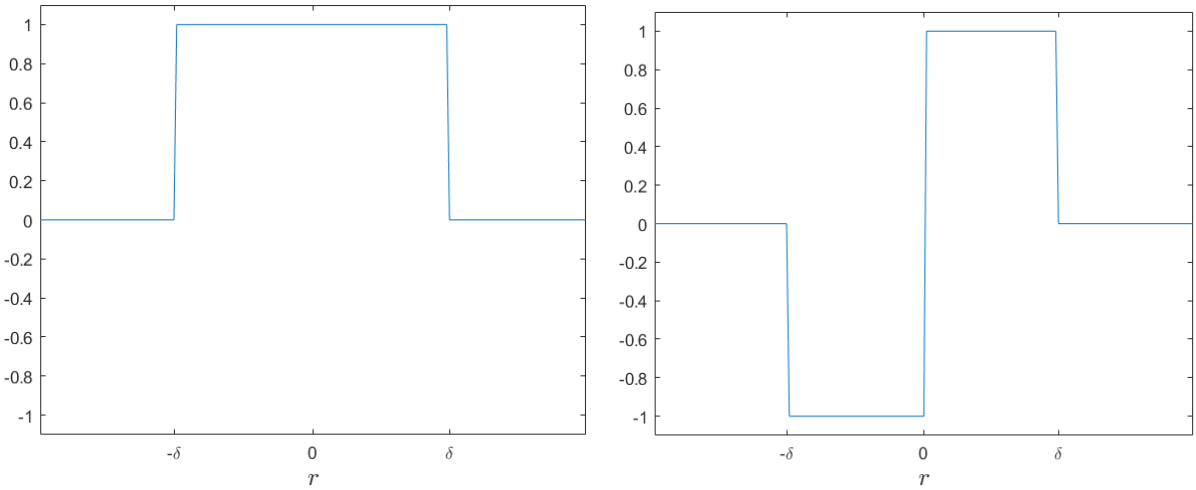
The distance, r, from the current evaluated point p is computed as $r = R_{min} + j\Delta r$, $j = 1, ..., M$, with the interval $\Delta r = (R_{max} - R_{min})/M$, where M corresponds to the number of points analyzed per direction.

Note that the alignment measure $A_i(r)$ is only computed when the image gradient's magnitude at point q is greater than a threshold, g_{min}. This constraint distinguishes the cases where the gradient's magnitude is approximately null - constant image region - and when there is a relevant intensity variation in a direction parallel to the positive horizontal axis. In both cases the gradient's direction, ϕ, is zero but considering the gradient's magnitude, only a sufficiently high gradient magnitude is regarded for the alignment's computation.

3.3.2 Score image computation

By detecting the maximum of $A_i(r)$ for each radial line i, it is possible to assign a score value to the point p, the candidate to the circle's center. For each i-th line, the boundary is referred to as the highest point of the stone circle and can be located by a Matched Filtering operation, given the alignment distribution along the radial line. This operation yields the title "sliding band" in the SBF's name and allows the detection of a signal when corrupted with white noise, given that the signal to detect, a template, is known.

The template used to determine the cell's boundary in [6] is a square pulse $T(r) = u(r + \delta) - u(r - \delta)$, where $u(r)$ denotes the unit step signal and 2δ the template's number of samples. The positive period of the template corresponds to the points q where the gradient direction coincides with the line's orientation, precisely where the cell's boundary is located. For the stone circle's problem, since there are two regions of intense image variation (first the gradient points outwards and then in an inwards direction, as r increases), the template to detect an ideal regular circle's boundary (Figure 3.16(a)) must be different from the one described in the cell detection problem in [6]. Therefore, the adapted template to be identified is $T(r) = -u(r+\delta)+2u(r)-u(r-\delta)$. In Figure 3.18 both template signals are represented, accordingly to the expected alignment signal to be detected.



(a) Cell detection [6].

(b) Stone circle's detection.

Figure 3.18: Comparison between the templates $T(r)$, used in cell and stone detection problems.

The template signal shown in Figure 3.18(b) enables the detection of a stone circle's boundary since, as r increases approaching the inner border, the gradient vector is aligned with the radial line and points outwards ($\theta_i - \phi(r) = 180°$), hence the negative pulse. Closer to the outer border, the gradient vector points inwards ($\theta_i - \phi(r) = 0°$) resulting in a positive alignment associated with the positive pulse of the template.

The filtering operation consists in the convolution of the signal $A_i(r)$ with an impulse response that is a reversed and time delayed version of the template $T(r)$. The best alignment for direction θ_i is obtained by maximizing the filter's output,

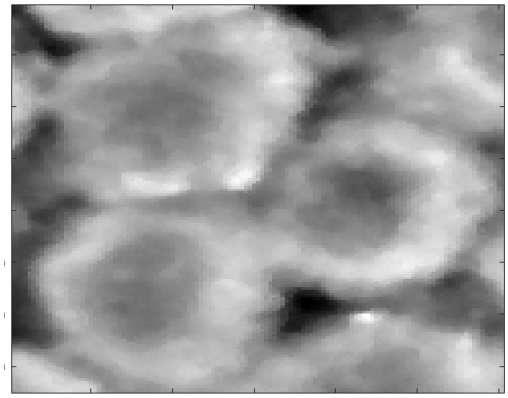
$$A_i = \max_{r_0}\Big\{\sum_r T(r_0 - r)A_i(r)\Big\}, \tag{3.7}$$

where r_0 is the sample corresponding to the output's highest peak.

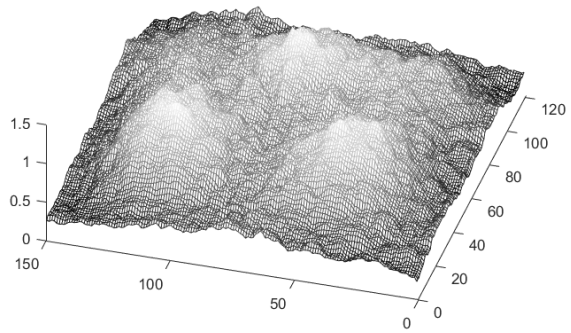
Since A_i is a measure of the similarity between $A_i(r)$ and the ideal alignment $T(r)$, by computing an average of all alignments A_i, for all directions $i = 1, ..., N$, it is obtained a score used to rank the center point p. The SBF's score image for the input pixel $p = [x, y]^T$ is then defined as

$$Y(x, y) = \frac{1}{N}\sum_{i=1}^{N} A_i. \tag{3.8}$$

Figure 3.19(b) shows the score image obtained after applying the SBF filter to the input image displayed in Figure 3.19(a).



(a) Input image.



(b) SBF's score image.

Figure 3.19: Input image with three stone circles and score image, obtained by the SBF filter. The latter exhibits three peaks, corresponding to the candidate center points.

Considering the noise in the score image from Figure 3.19(b), a smoothing operation is performed. This filtering operation eliminates the sharp variations locally while preserving the edge detail and the information about the alignment score. To achieve this goal, the SBF's output image is convolved with a Gaussian kernel, $H(x, y) = \frac{1}{2\pi\sigma^2} \mathrm{e}^{-\frac{x^2+y^2}{2\sigma^2}}$, and the filter's output image is a weighted average of a patch. The central pixel of the patch has the highest weight, and as the distance to it increases, the kernel values are reduced, according to a normal distribution. Figure 3.20 reveals the smoothing filter's output image.

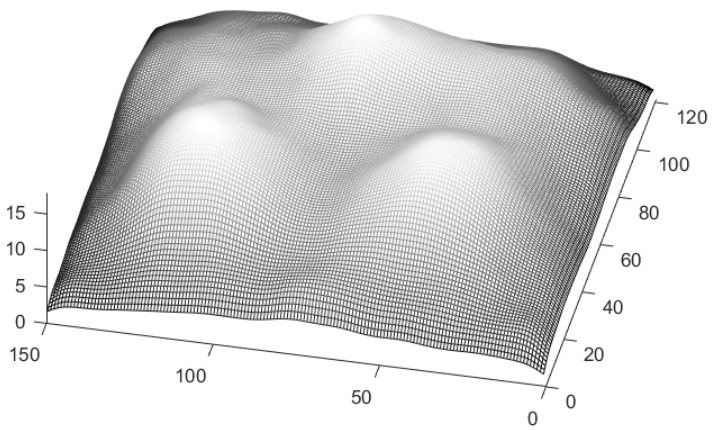


Figure 3.20: Filtered SBF's score image, $Y * H$.

After the filtering process, the score image in Figure 3.20 presents smooth bell-shaped curves with brighter rounded peaks, corresponding to local maxima of the original score image Y, and consequently, to possible center candidates for the stone circles shown in Figure 3.19(a). On the other hand, the darker pixels suggest an inconsistent alignment between the radial lines diverging from them and the gradient orientation.

In order to obtain the peak locations of the score image, the same method used for the Template Matching's score image is applied: the non maximum suppression algorithm detects local maxima, removing the neighbouring points whose score is inferior to the maximum. This strategy is followed by

a comparison with a threshold, Th, to eliminate the low-amplitude peaks. As mentioned, this intensity bound adjusts how rigorous or permissive the peak selection method is. Figure 3.21 shows the detections preserved after the thresholding (green stars), superimposed with the SBF's score image obtained for the input image in Figure 3.5(a).

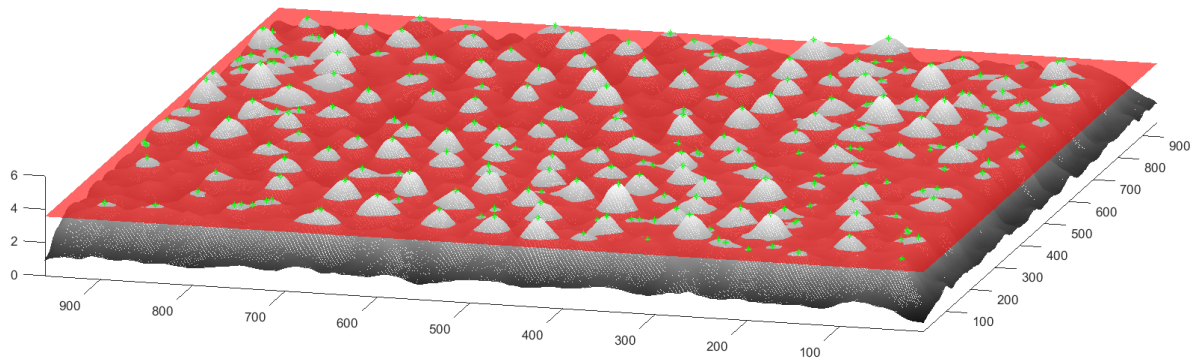


Figure 3.21: Thresholding output: the green marked pixels have a score value above the minimum required, represented by the red horizontal plane $z = Th \times \max\{Y\}$.

Some detected peaks in Figure 3.21 are too close together to be considered stone circles. Therefore, the spatial filter described in Section 3.2.3 was implemented, resulting in the elimination of many false detections, as Figure 3.22 illustrates.

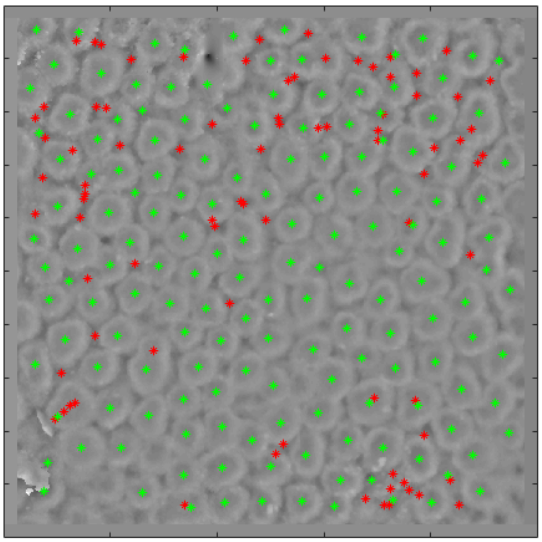
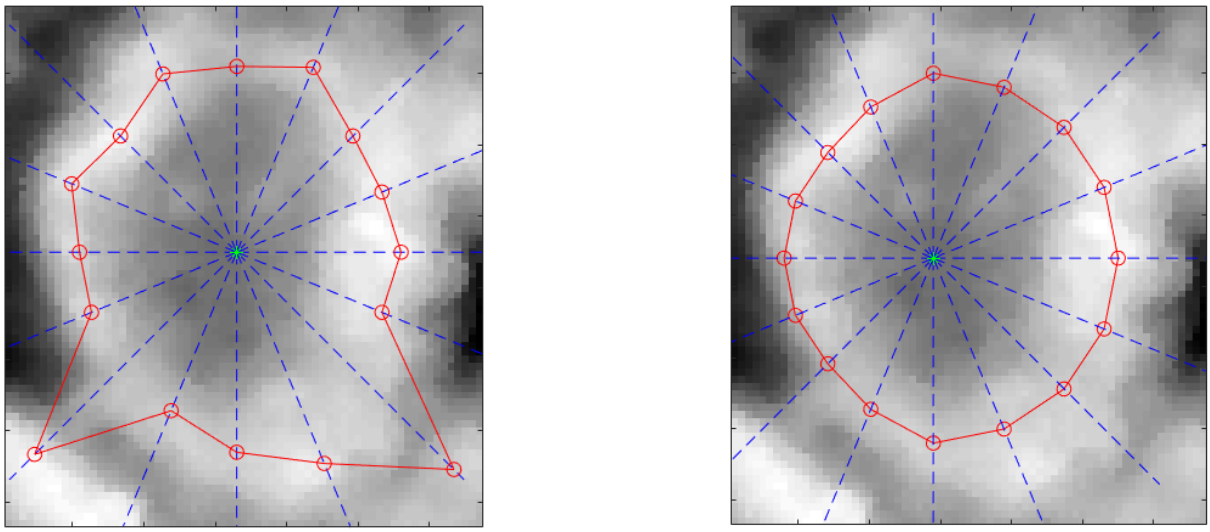


Figure 3.22: Detected peaks after spatial filtering (green marks). The red marks correspond to eliminated peaks.

Since the SBF method explores the gradient information of the image, it may be prone to outliers. Section 3.4 presents a similar method that also relies on the alignment score in the vicinity of a center candidate yet, it considers an additional smoothing constraint which improves the robustness of the stone's detection method. In fact, the SBF method is a particular case of the proposed method described in the next section.

3.4 Dynamic Programming

The method proposed in this section is based on a Dynamic Programming technique [7]. The criteria used for detecting and delineating the stone circles is similar to the one used in SBF: both methods use the contour information to evaluate if a given pixel is a valid stone circle center but the main difference resides in the fact that Dynamic Programming estimates the shape taking a smoothness criterion into account. Figure 3.23 illustrates this difference, affecting both the stone's center localization and contour extraction.



(a) Detection and contour extraction with SBF.

(b) Detection and contour extraction with DP.

Figure 3.23: Stone detection and contour extraction using SBF (left) and the Dynamic Programming (right) method, with $N = 16$ directions.

The score image obtained by the Dynamic Programming method includes not only a term related to the alignment between the gradient direction and a radial structure, but also a transition cost, enforcing the smoothness of the shape. As Figure 3.23(b) shows, this approach avoids sharp transitions of the radius composing the stone's contour. On the other hand, the SBF method estimates the contour independently for each direction which may lead to outliers, visible in Figure 3.23(a).

Figure 3.24 displays the block diagram of the proposed solution, whose structure is similar to the one described for the other methods. The first block computes a score image Y, whose pixel values reflect the likelihood of existing a valid stone circle contour centered in the corresponding input image pixel. The score image is post processed in order to apply the non maximum suppression and thresholding methods to produce the stone circle candidates, which are filtered in the peak validation stage.

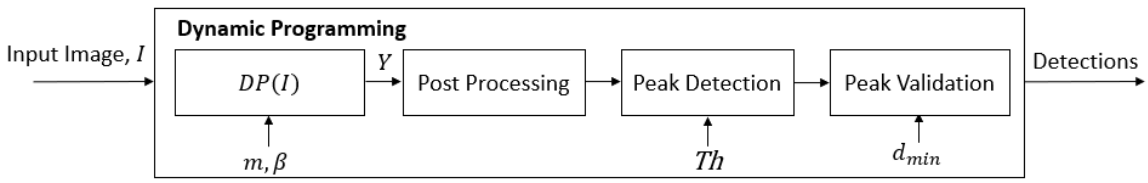


Figure 3.24: Block diagram of the Dynamic Programming method for the stone's detection.

This section is organized as follows: first there is a brief historic introduction about the dynamic programming technique, its foundations and fields of application; then there is a description of the implementation of these technique applied to the detection and delineation of the stone circles, where the minimization of an energy functional is analyzed. This includes the formulation of a multi-stage decision problem in order to obtain an optimal path, corresponding to the contour delineation output. The energy functional is used to compute a score image, from which the center candidates are created.

3.4.1 Dynamic Programming and Bellman's optimality principle

Dynamic Programming aims to solve optimization problems involving multi-stage decision processes. It was proposed in the early 1950s by Richard E. Bellman, an American applied mathematician, at the same time he investigated control theory and time-lag processes, becoming a major figure in modern optimization [28] [29].

These multi-stage type problems share a general definition. There is a physical system whose state is described at each time instant by the state variables. At certain times, decisions affecting the current system's state can be taken, resulting in a transformation of these state variables. Since each decision influences future ones, the challenge resides in selecting each decision in order to maximize some function of the variables of the final state [30].

The decision problems first studied ranged from a variety of fields, from the Industry to Economics. Some examples include the scheduling of the resources to be transformed by different machines or the planning of the merchandise purchases to fulfill the future demand. The techniques developed to solve these complex issues are still important topics currently, in Finance and Manufacturing, but also in Robotics, where Reinforcement learning plays a major role. In all these situations, the purpose consists in achieving the best outcome possible according to some criterion and the selection of the most advantageous sequence of decisions - the optimal policy - is undertaken.

The classical approach for determining the optimal policy consisted in the brute force examination of all policies, an inefficient method that becomes unfeasible when the number of stages and available decisions per stage increase. The search process using dynamic programming reduces the problem of excess dimensionality since it is based on an important property of multi-stage decision processes: Bellman's principle of optimality [31]. This principle states the following: *"An optimal policy has the property that whatever the initial state and initial decisions are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision."*

A dynamic programming problem can be defined through a functional equation, whose simplest case is referred to discrete deterministic process [32]. A system is determined at any time instant by an M-dimensional state vector $p = (p_1, p_2, ..., p_M)$ subjected to lie in region D. Considering an N-stage process, the purpose is to select the N decisions or transformations T_k in order to maximize a scalar function $R(p)$ of the final state, also know as the N-stage return function,

$$f_N(p) = \max_{(T_1,...,T_N)} R(p_N). \tag{3.9}$$

The set of N transformations $(T_1, ..., T_N)$ defines a policy, such that the state p_{j+1} is reached and uniquely determined as a result of $T_{j+1}(p_j)$, for $j = 1, ..., N-1$.

The principle of optimality is used to rewrite (3.9), enabling the separation of the global problem into equivalent sub-problems. Assuming the transformation T_k is selected in the first decision, the system's new state is $T_k(p)$, and the maximum return achievable for the remaining $N-1$ stages is, by definition, $f_{N-1}(T_k(p))$. The transition k must then maximize this $(N-1)$-stage return function. This process is performed until all N decisions are computed, according to the recursive equation

$$f_n(p) = \max_k f_{n-1}\Big(T_k(p)\Big), \tag{3.10}$$

for $n = 2, ..., N$.

This result will be applied in the problem of the detection and contour delineation of the stone circles, where an optimal path equivalent to the optimal policy is determined.

3.4.2 Score image computation

The transformation of each input image pixel to the score image, which states the likelihood of that pixel being a stone's center, is similar to the method used in the SBF algorithm, as mentioned. It is based on the analysis of the alignment between the image gradient direction and the orientation of auxiliary radial lines intersecting the current pixel evaluated. Assuming the average alignment score obtained in (3.8) is a measure of the contour's energy, in the SBF method, the higher this energy, the more acceptable is the corresponding pixel as a stone's center. In the case of the Dynamic Programming method, this energy has a distinct interpretation and contemplates not only the average alignment but also the overall circular geometry of the contour to be obtained.

Given the dependency between the contour points, the contour's energy computation can be seen as a shortest path problem, usually studied in graph theory. The graph is composed of $M \times N$ vertices, M being the number of admissible radius values representing the distance from the stone's center to the contour point along the i-th direction, with i ranging from the first to the N-th direction. The vertices corresponding to the i-th direction are all connected to the vertices of the $(i+1)$-th stage by weighted edges, reflecting the deviation between the radius lengths, as show in Figure 3.25.

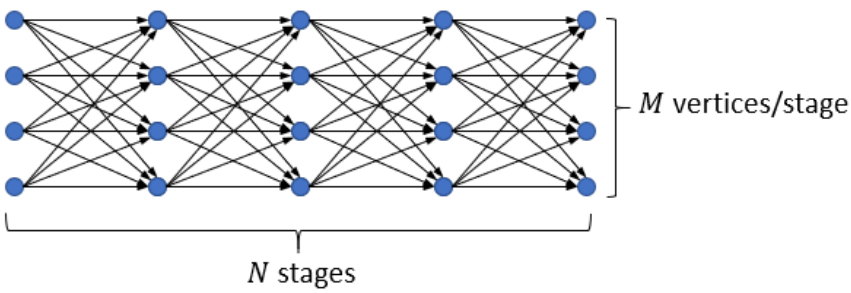


Figure 3.25: The delineation problem consists in the selection, for each stage $i = 1, ..., N$ (direction θ_i), of a vertex $j \in \{1, ..., M\}$ corresponding to the distance between the stone's center and the i-th contour point.

The purpose of this shortest path problem is to determine the contour, a sequence of radius $r = [r_1, ..., r_N], r_t \in \{1, ..., M\}$, that minimize a cost function with two terms: an alignment term (see (3.7)), and a transition cost term associated to the graph's edges. In order to solve this search problem, such graph can be represented by a matrix $V \in \mathbb{R}^{M \times N}$, whose rows and columns designate the coordinates of points in polar form. The matrix rows represent the distance from the contour point to a pixel p, a center's candidate, and its columns refer to the direction of a radial line diverging from p. For instance, the entry $V(j, i)$ refers to the point

$$q = p + r_j[\cos\theta_i, \sin\theta_i]^T, \tag{3.11}$$

and stores the symmetrical of the filtered alignment value computed at point q,

$$s_i = -\sum_r T(r_0 - r)A_i(r), \quad V(j, i) = s_i(j). \tag{3.12}$$

As defined for the SBF method, the index i refers to the angle $\theta_i = \Delta\theta(i-1)$, with $i = 1, ..., N$ and the index j determines the radius $r_j = R_{min} + j\Delta r$, with $j = 1, ..., M$, where the angle and radius intervals are defined, respectively, by $\Delta\theta = 2\pi/N$ and $\Delta r = (R_{max} - R_{min})/M$.

The greater the alignment computed at point q, the smaller the amount $s_i(j)$, which increases the likelihood of candidate p being a stone circle's center. At this point, the contour obtained by selecting, for each direction i, the radius that minimizes the quantity defined in (3.12) would coincide with the one found with the SBF method. By introducing the transition cost $c(r_i, r_{i+1})$ that penalizes deviations among radius values for consecutive directions i and $i+1$, the contour point computed for a given direction becomes dependent on all contour points obtained for previous directions. In this conditions, the optimal set of radius that form r minimize the energy functional [7]

$$E(r) = V(r_1 = k, 1) + \sum_{t=2}^{N} V(r_t, t) + c(r_{t-1}, r_t), \tag{3.13}$$

with the cost of the transition from r_{t-1} to r_t defined as

$$c(r_{t-1}, r_t) = \begin{cases} m \mid r_t - r_{t-1} \mid, & \text{if} \mid r_t - r_{t-1} \mid \leq \beta \\ \infty, & \text{otherwise} \end{cases}, \tag{3.14}$$

where m and β are parameters that define the permissiveness and smoothness of the variation between consecutive radius. The greater the slope m, the more restrictive are the allowed jumps since the cost increases faster and by choosing a smaller β, the algorithm also ensures smoother transitions as bigger deviations are avoided.

Although r is a closed contour, where the first and last contour points are the same, given the definition of θ_i the first radial line direction does not match the N-th one. To fix this issue, an extra column is added to the V matrix, whose filtered alignment values corresponds to those computed for the first direction, since $\theta_1 = \theta_{N+1} = 2\pi$ rad. In this way, the closed contour constraint is defined as $r_1 = r_{N+1}$. Figure 3.26 shows a representation of the complete V matrix creation. The minimization of the contour's

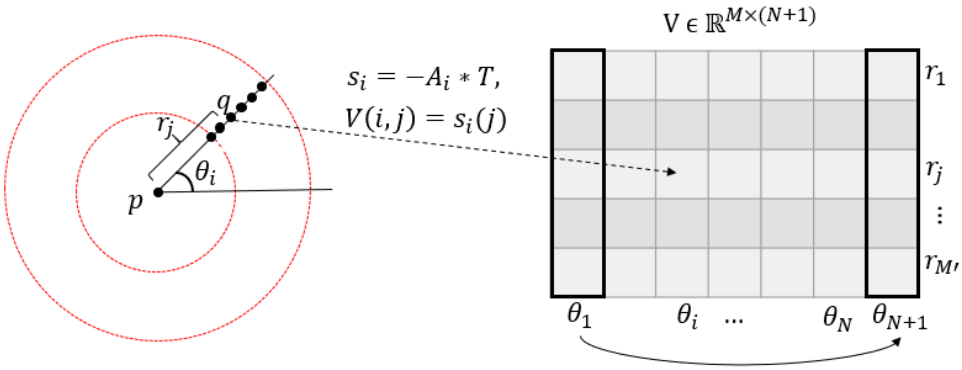


Figure 3.26: V matrix creation for pixel p, from the alignment information.

energy $E(r)$ implies the alignment for each r_t should be high - minimum $V(r_t, t)$ - and the deviations between adjacent radius should be small. A bottom-up approach of the Dynamic Programming method determines the optimal path leading to the optimal energy value. This method comprises two stages: the forward recursion and the backward recovery step.

The forward recursion procedure computes the optimal energy to reach the element in row j and column t, from the first path element located in row k and column 1, according to

$$E_t(j) = \min_{r_2,\ldots,r_t;r_t=j} \Big[V(r_1 = k, 1) + \sum_{p=2}^{t} V(r_p, p) + c(r_{p-1}, r_p)\Big], \tag{3.15}$$

which can be written recursively as

$$E_t(j) = V(j,t) + \min_i \Big[E_{t-1}(i) + c(i,j)\Big]. \tag{3.16}$$

This operation enables the computation of the optimal energy to reach a state j at any stage t and is the only feasible approach when the final state is unknown. After this step, in addition to all energy values $E_t(j)$, the sequence of indexes i representing the previous state that minimized $[E_{t-1}(i) + c(i,j)]$ is also available. This set of pointers is obtained from

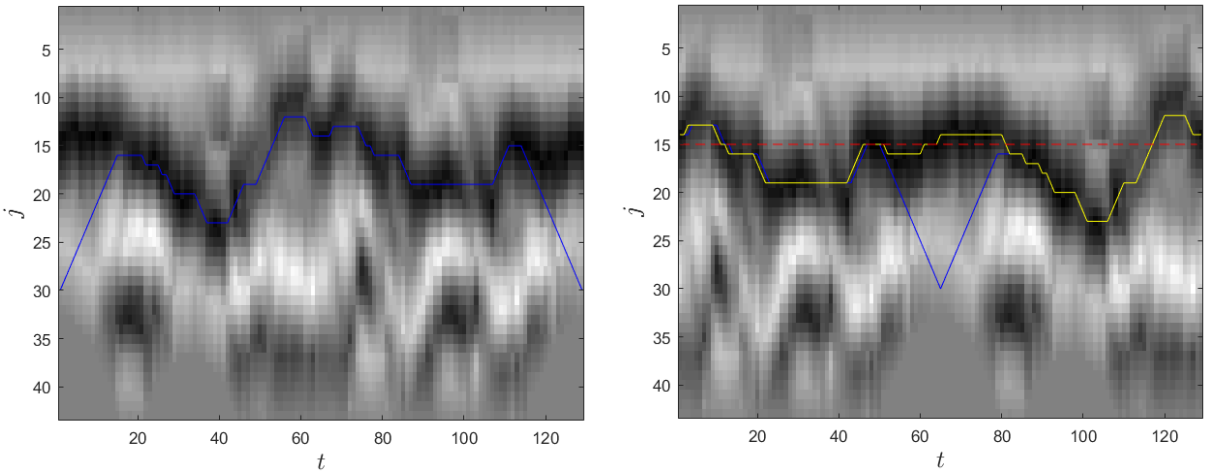
$$\phi_t(j) = \arg\min[E_{t-1}(i) + c(i,j)]. \tag{3.17}$$

The backward recovery step uses this information to identify the optimal path $r^* = [r_1^*, r_2^*, ..., r_N^*, r_{N+1}^*]$. The last contour point is given by the constraint $r_{N+1}^* = k$ and the remaining ones are computed according to

$$r_{t-1}^* = \phi_t(r_t^*), \quad t = N+1, ..., 2. \tag{3.18}$$

Since the optimal initial condition $r_1 = r_{N+1} = k$ is unknown beforehand, an optimal contour is computed for each possible starting radius value $k \in (r_1, ..., r_M)$ and the one whose contour has the smallest energy is set as the constraint. It is possible to improve this approach by only computing the forward and backward routines twice, instead of M times [33]. This efficient approach is defined under the assumption that, regardless of the initial radius k, halfway of the optimal path the radius value is

always the same. Since the only condition is for the contour to start and end at the same point, this central radius value is correctly accepted as the optimal initial condition. Its computation is performed as follows: first, the columns of the V matrix are rearranged, in such a way the columns $N/2+1$ up to $N+1$ are shifted to the left, followed by the first and up to the $N/2$-th column. The optimal energies are computed using the forward recursion with the constraint $r'_1 = r'_{N+1} = M$, set as a rule of thumb, and from the backward recovery, an auxiliary optimal contour r' is obtained. Note the first and last elements of r' correspond, respectively, to the $N/2+1$ and $N/2$ columns of the original V matrix, given the shifting operation involved. The optimal initial condition corresponds to the central element of the path, the radius value $r'(N/2+1)$. Once k is known, the V matrix is again rearranged to its original form and a second computation of the contour is performed, where the search begins at direction 1 up to the $N+1$-th one, subjected to $r_1 = r_{N+1} = r'(N/2+1)$. Figure 3.27(a) shows an example of the auxiliary contour r', superimposed with the rearranged V matrix, in order to determine k. The optimal contour r, with $r_1 = k$, is plotted over the original V matrix in Figure 3.27(b).



(a) Auxiliary contour r'.

(b) Optimal contour r.

Figure 3.27: Contour delineation output: the blue contour represents the auxiliary path for the computation of the boundary conditions $r_1 = r_{N+1} = k$. The image on the right displays the reordered blue contour superimposed with the final contour in yellow. The horizontal curve in red corresponds to a circle with radius $R = 30$.

The contour illustrated in Figure 3.27 is composed of $N = 128$ contour points where the possible radius values range from $R_{min} = 15$ to $R_{min} + M\Delta r$, with $M = 30$ and $\Delta r = 1$. The cost parameters used were $m = 0.3$ and $\beta = 1$, restricting the variations between consecutive radius, as visible through the constant segments of r and the absence of sharp transitions.

In order to obtain the contour points r from polar to Cartesian coordinates, the following conversion is performed: the element $r_t \in \{1, ..., M\}$ represents the point $\rho e^{j\theta}$, where

$$\begin{cases} \rho = R_{min} + r_t \Delta r \\ \theta = \Delta\theta(t-1) \end{cases} . \tag{3.19}$$

From the parameters ρ and θ, the coordinates of the contour point X_c in the Cartesian plane are deter-

mined by

$$X_c = p + \rho[\cos\theta, \sin\theta]^T, \tag{3.20}$$

where p is the stone circle's center. Similar to the process explained in the contour extraction using the SBF method, the radius ρ must be shifted by d samples so the boundary location is computed. Figure 3.28 exhibits the contours obtained for an input image with three stone circles. The contour of

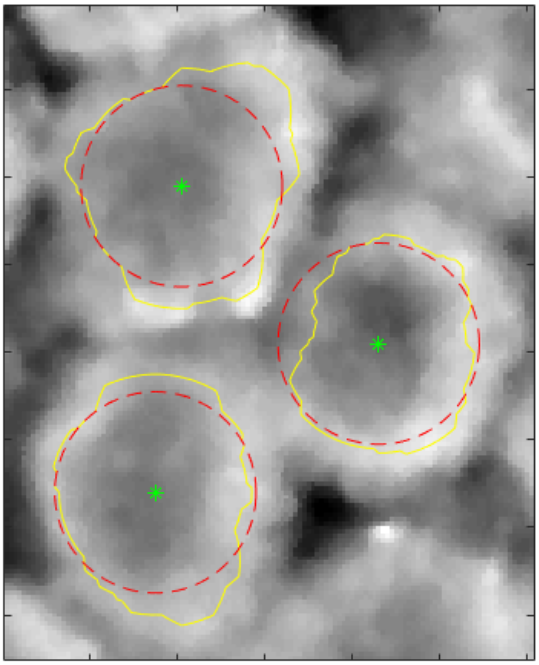
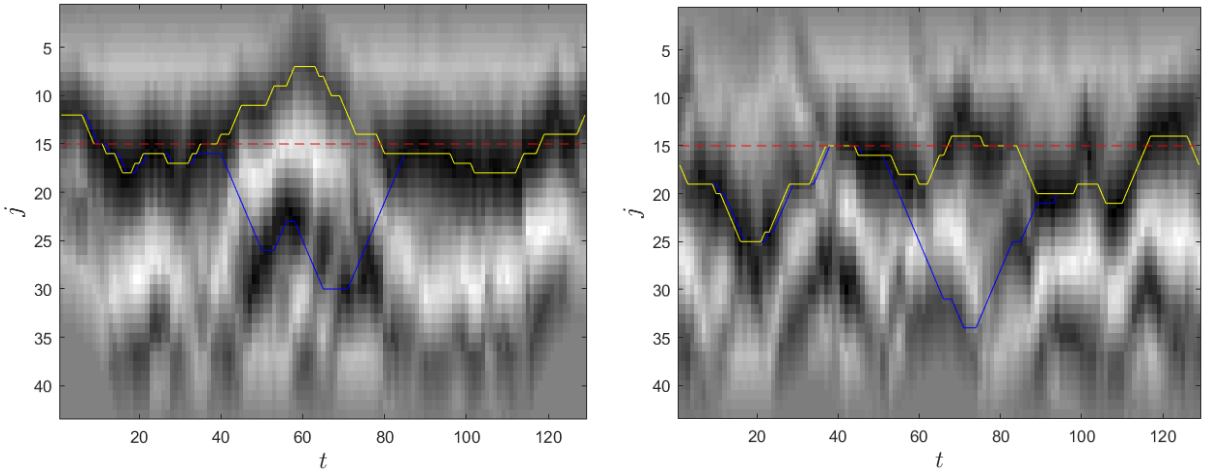


Figure 3.28: Optimal contour r in Cartesian coordinates, for three stone circles.

Figure 3.27(b) refers to the bottom-left detection of Figure 3.28. The red contour is displayed to confirm the same information is conveyed in both images in Cartesian or Polar form: for $t = 1, \theta = 0$ rad to approximately $t = 15, \theta = \pi/4$ rad, the radius values of r are smaller than $R = 30$. After this, the red and yellow curves intersect and the contour is greater than R up to the direction $t \approx 43, \theta \approx 2\pi/3$ rad. Moving through the columns of V, it is visible the red and yellow contours are similar until the distance from r to the center starts to increase, around $t = 80, \theta = 5\pi/4$ rad. Finally, roughly about $t = 105, \theta = 26\pi/16$ rad, the radius values decrease and r approaches the red curve again, reaching the boundary condition $r_{N+1} = 14$. The same analysis can be made for the middle and the upper stone circles of the input image, whose contours are displayed in Figure 3.29, superimposed with the respective alignment matrices V.

This solution for the contour delineation problem provides also a method for the stone's detection. As stated in Section 3.4.2, the alignment information, that determines the score image value for a pixel p in the SBF method, can be seen as the energy of the contour obtained for pixel p. In the Dynamic Programming's method, the same assumption is made, although with a different meaning: the pixel whose contour energy is the smallest is more likely to be a correct stone circle's center, since the alignment is higher for minimum energy contours according to (3.13). Therefore, given the optimal contour $r^* = [r_1^*, ..., r_N^*, r_{N+1}^*]$ for the center pixel p, the DP's score image Y is proportional to the



(a) Contour obtained for the middle detection.

(b) Contour obtained for the upper detection.

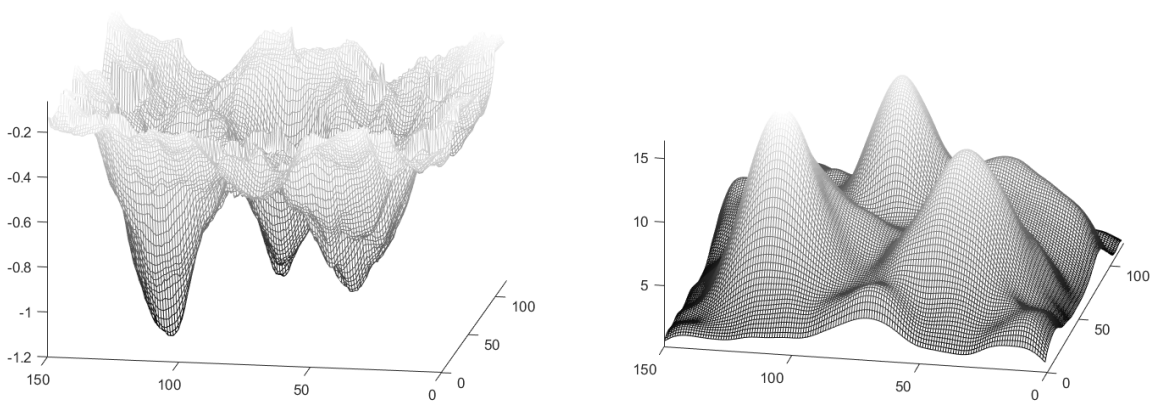
Figure 3.29: Contour delineation output for the middle and upper detections of the input image.

contour's energy,

$$Y(x,y) = \frac{E(r^*)}{N+1} = \frac{1}{N+1}\Big[V(r_1^*,1) + \sum_{p=2}^{N+1} V(r_p^*,p) + c(r_{p-1}^*, r_p^*)\Big]. \tag{3.21}$$

In order to determine the center candidates from the score image Y, it is implemented a post processing operation, after which the peak detection and validation methods described in the SBF section are applied. These methods' input is a score image whose pixels with higher values are more likely to be chosen as stone circles candidates yet, in the score image produced by the Dynamic programming method, the smaller image values correspond to stone detections. The post processing stage produces a reversed version of the score image, Y_o, whose smaller values are transformed into higher ones and higher input values are decreased, through the expression $Y_o = \max\{Y\} - Y$.

Figure 3.30(a) shows the original score image of the DP method obtained for the input image in Figure 3.19(a) and the output of the post processing stage is displayed in Figure 3.30(b).



(a) Dynamic programming's score image Y.

(b) Score image after post-processing , Y_o.

Figure 3.30: Original (left) and post-processed (right) score image: the negative original peaks correspond to the valid center candidates, mapped to the maxima of Y_o.

As Figure 3.30 exhibits, the pixels in the original score image whose contour energy is the smallest have an increased probability of corresponding to valid detections. These detections match the maxima of the post-processed score image, Y_o, corresponding to green pixels highlighted in Figure 3.28.

The post-processing stage also includes the smoothing operation $Y * H$, so the bell-shaped curves are preserved and its peaks emphasized. The filter kernel H corresponds to the same Gaussian blur used to produce the SBF's score image in Figure 3.20.

Given the score image, the final stage of the detection algorithm consists in selecting its maxima. The non-maximum suppression algorithm outputs the local maxima location and the comparison of their score value with a threshold allows the computation of the stone center's candidates. Figure 3.31 shows the score image's local maxima obtained after this peak detection procedure, superimposed with the minimum required score, represented as the red horizontal plane.

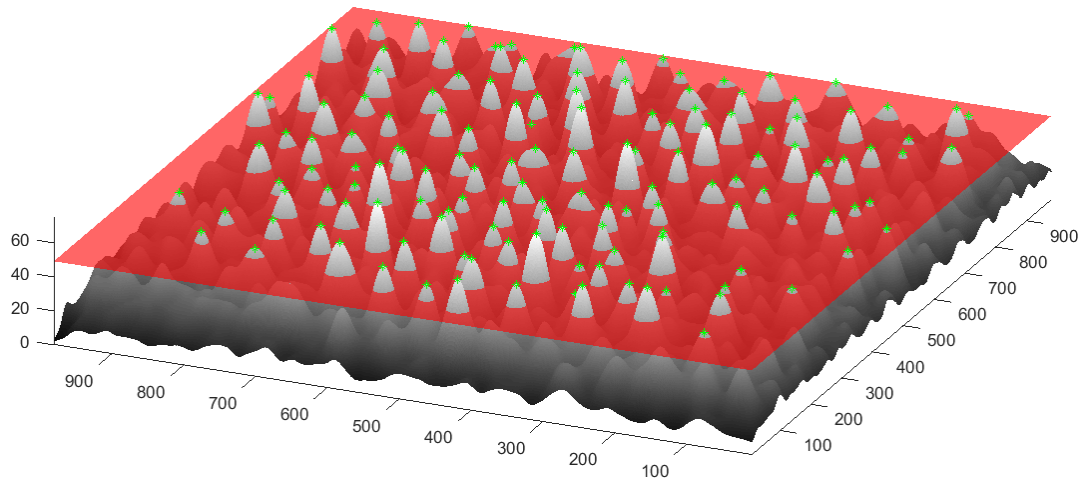


Figure 3.31: Thresholding output: the green marked pixels have a score value above the minimum required, represented by the red horizontal plane $z = Th \times \max\{Y\}$.

The score image obtained using the DP method (see Figure 3.31) presents pointy bell-shaped curves, with further separated peaks. However, once there are detections nearby a higher local maxima, those should be discarded. The peak validation method described in Section 3.2.3 filters these candidates. Figure 3.32 reveals the final detections after this validation process.

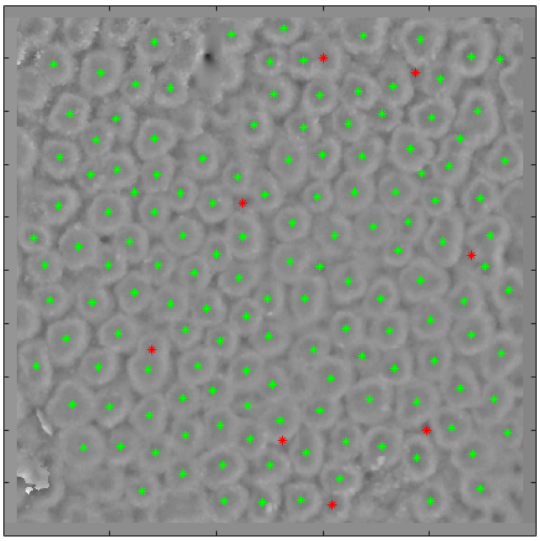


Figure 3.32: Peak validation output: Most candidates (green) correspond to an accurate placement of the stone circles. Note the reduced number of detections in the intersection of the structures.

Geologists wish to know not only the stone circles' centers but their boundary as well, in order to study the influence of the orography and climate in the circle's dimensions and shape. Section 3.5 presents two solutions for the contour extraction problem.

3.5 Contour Delineation

This section describes how the SBF and DP methods can be applied in the stones' boundary extraction.

3.5.1 Sliding Band Filter

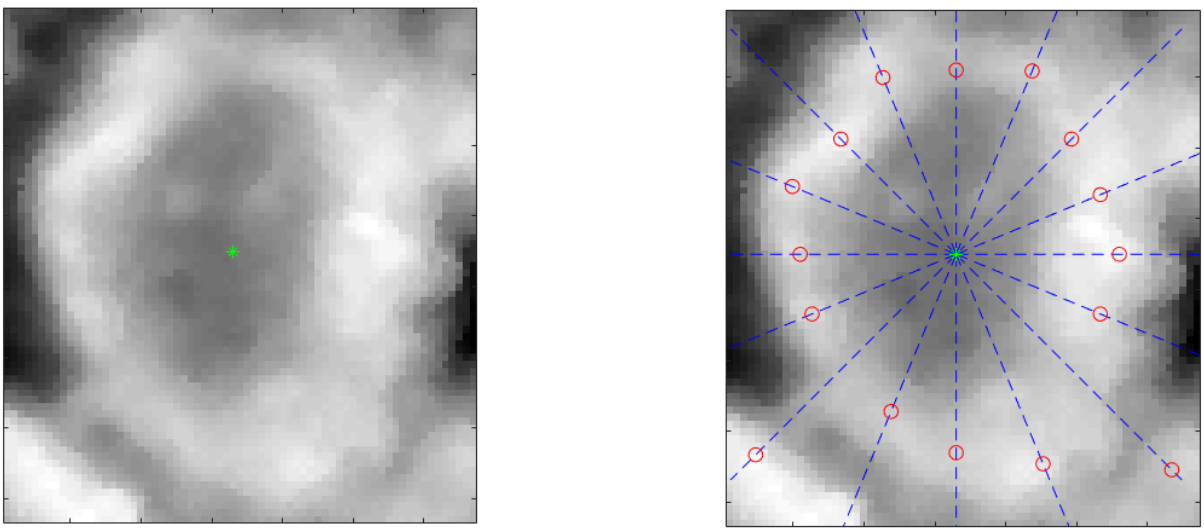
The SBF method presents a solution for the contour extraction problem based on the approach used in the detection of the stone circles. In this work, a contour point is defined as the highest point of the stone circle, along a given direction θ_i, for $i = 1, ..., N$. In the cross-section representation of the synthetic circle from Figure 3.7(b), the two contour points correspond to those where $d = \pm R$, for opposite directions.

In order to obtain these, for each center candidate location, it is used the information provided by the alignment score in (3.7). By computing the sample that maximizes the filtered alignment,

$$r_0^{'}(i) = \underset{r_0}{\arg\max} \Big\{ \sum_r T(r_0 - r) A_i(r) \Big\}, \tag{3.22}$$

for each direction $i = 1, ..., N$, the circle's boundary is identified. However, since the contour point corresponds to the transition from the negative to the positive pulse of the template (see Figure 3.18(b)), and the matched filter output is maximum for the last sample of the template, a shifting operation on $r_0^*(i)$ is required. Therefore, the contour point along the i-th direction is computed as $r_0^*(i) = r_0^{'}(i) - d$, where d is half the number of samples of the template.

Figure 3.33 illustrates the contour points (red markers) obtained from expression (3.22), corresponding to the detected stone's center marked as green. The $N = 16$ auxiliary radial lines are also superimposed in Figure 3.33(b).

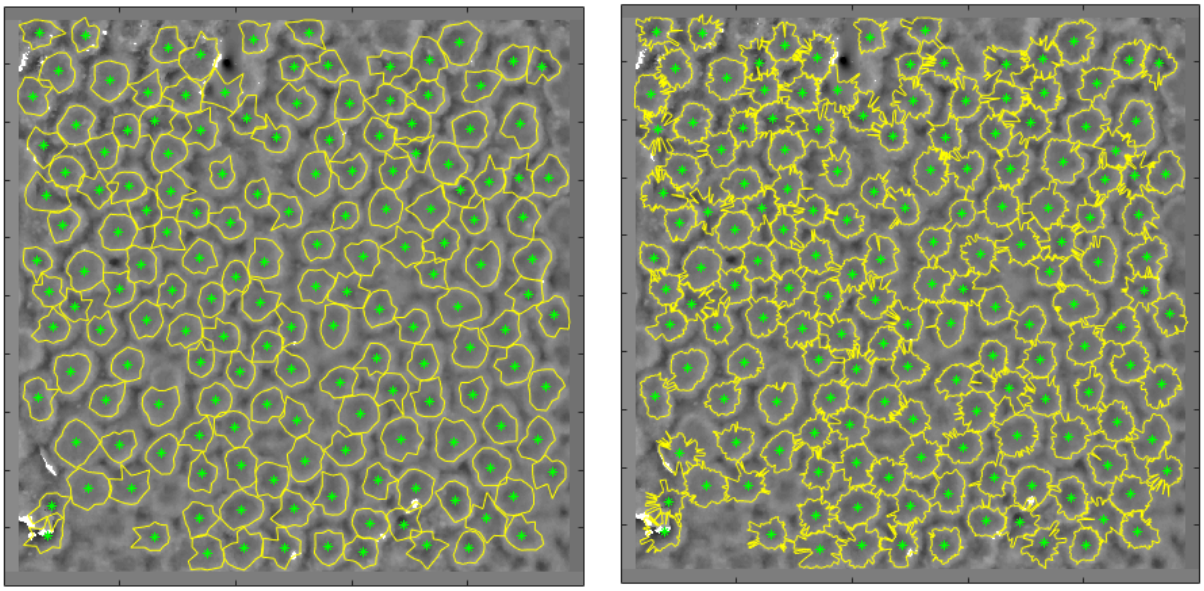


(a) Detected stone circle center.

(b) Contour points obtained per direction.

Figure 3.33: Detection and respective contour points obtained using SBF method with $N = 16$ directions.

Figure 3.34 shows the contour estimation output for the SBF method for $N = 16$ and $N = 128$ directions, for an input image with multiple stone circles. All contour points are connected, creating a closed curve for each stone circle detected. It is visible that the delineation with a smaller N results in a coarse approximation of the overall stone shape, and as the number of points in the contour, N, increases, the more detailed and complex the contour is. For both values of N, the contour delineation method exhibits some noise, producing contour points that lie farther from the stone circle, in the proximity of its neighbor. This may be avoided by reducing the outer limit of the radius' search region, R_{max}, but since the circles' size may differ, this region should be kept as general as possible.



(a) Contour for $N = 16$.

(b) Contour for $N = 128$.

Figure 3.34: Contour delineation obtained with $N = 16$ and $N = 128$ directions, for SBF method. For most stone circles, a smaller N results in a reasonable contour estimation while a greater N presents more noise and sharper transitions as the angular step size is reduced.

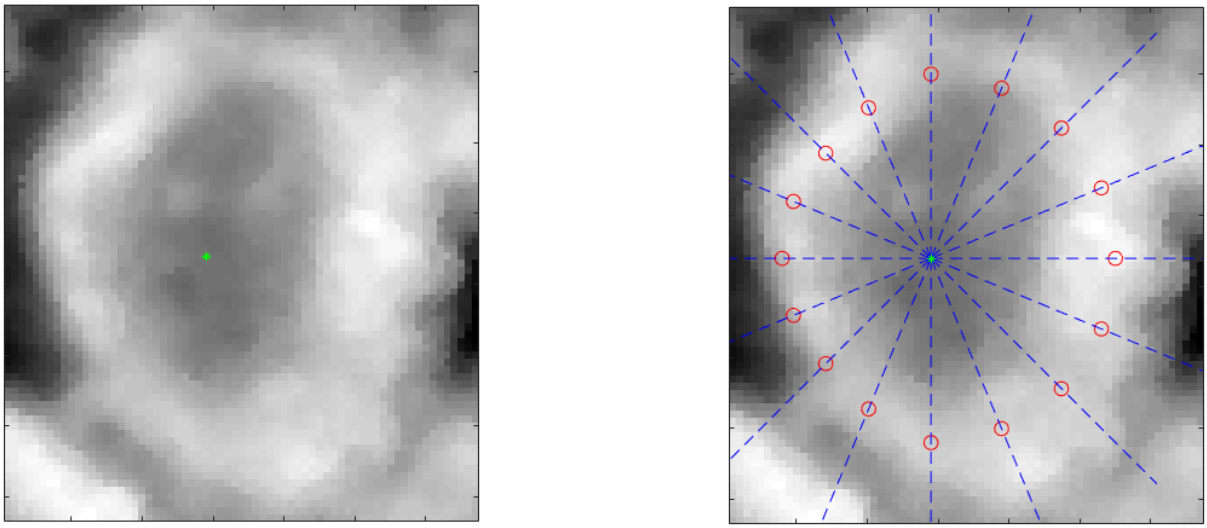
Another alternative to reduce the large variation between consecutive radius r_0^* is to introduce some dependency between contour points instead of accepting the radius that maximizes the alignment for each individual direction. Given the circular shape of the structures to outline, it is possible to include a restriction imposing the distance from the contour points to the circle's center to be constant. By associating a cost to the deviation between this distance for consecutive directions, the choice of a contour point must not only maximize the alignment, but also minimize this transition cost between the radius for direction θ_{i-1} and θ_i. This trade-off would smooth the oscillations, more frequent for a larger N, since a higher transition cost would decrease the probability of distant points being chosen, regardless of their alignment score. Dynamic programming allows this adaptation for choosing the optimal contour.

3.5.2 Dynamic Programming

A Dynamic Programming approach is used to compute the optimal contour according to an optimization criteria of the contour's energy. This energy measures both the alignment score introduced in

(3.6) for each candidate contour point and the deviation between consecutive radius values. The latter introduces a dependency between the contour points, a concept not considered in the SBF method. It is possible to control the penalization due to the radius deviation by selecting the cost parameters: a lower transition cost results in contour points mostly determined by the alignment score while a higher penalization may prioritize a smaller radius variance over the quality of the alignment. The method computes the optimal contour by solving a graph search problem defined by the forward recursion (3.16) and backward recovery (3.18) equations.

Figure 3.35 illustrates the contour points (red) obtained with the DP method, including the stone's center location (green) and the $N = 16$ auxiliary radial lines.

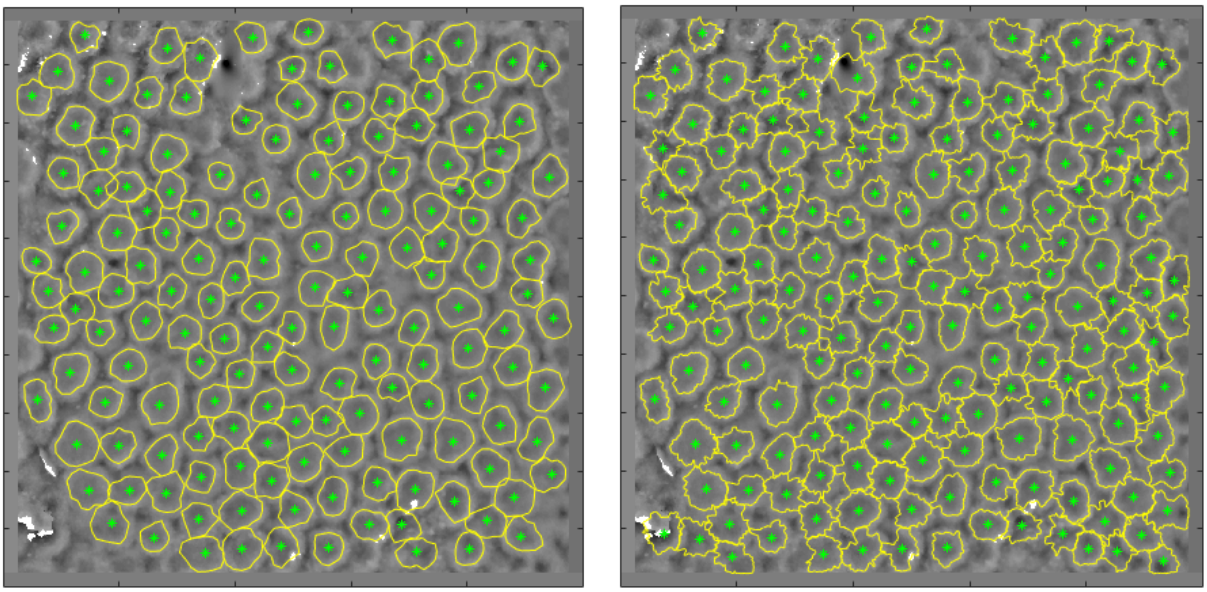


(a) Detected stone circle center.

(b) Contour points obtained per direction.

Figure 3.35: Detection and respective contour points obtained using DP method with $N = 16$ directions.

Figure 3.36 shows the output of the DP contour extraction method for $N = 16$ and $N = 128$ directions. The N contour points are connected, hence the continuous curves observed. The majority of the contours obtained with the smaller number of directions are accurate estimations of the stones' shapes and the higher angle interval between contour points results in a smooth outline. On the other hand, for $N = 128$, the contours are more complex and the small penalization given by the cost parameters $m = 0.03$ and $\beta = 10$ contributes to the oscillations observed.



(a) Contour for $N = 16$.

(b) Contour for $N = 128$.

Figure 3.36: Contour delineation obtained with $N = 16$ and $N = 128$ directions, for DP method. For $N = 16$ the contours are accurate estimates of the stone circles while a higher N produces a more detailed but still noisy outline given the small penalization used.

Chapter 4

Results

This chapter presents the experimental evaluation of the proposed methods of stone circles' detection and contour extraction. It is organized as follows: first the dataset used in the experiments is presented, as well as the Ground-Truth (GT) computation. The evaluation criteria and metrics are then described and additionally, a comparison analysis of the proposed methods is performed, where several parameter combinations are tested.

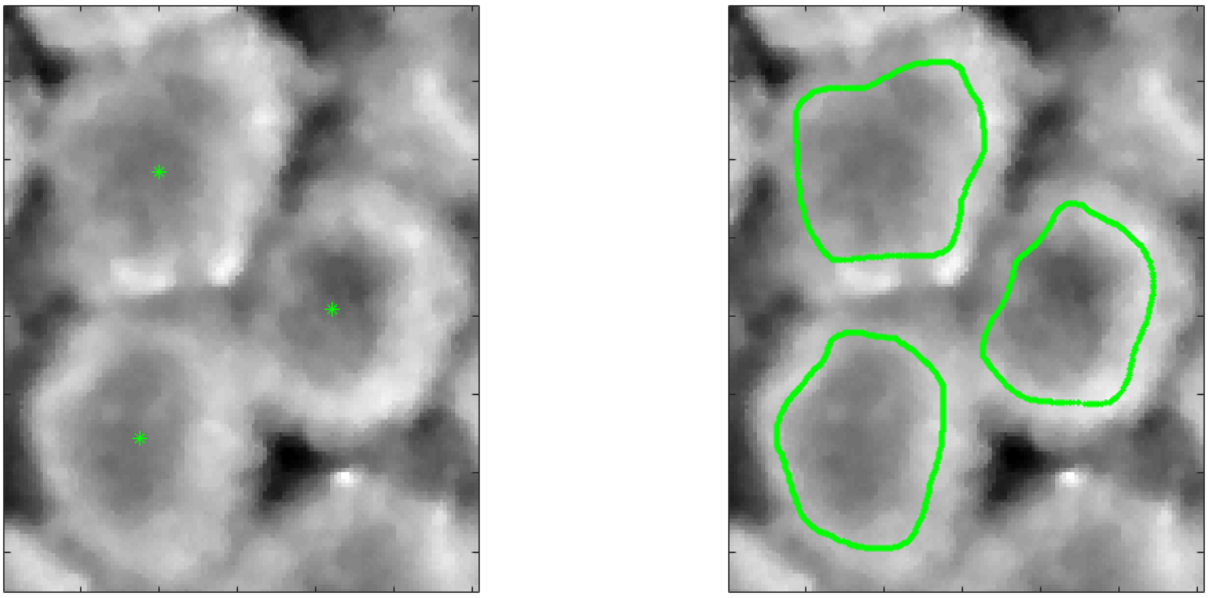
4.1 Dataset

The dataset used in this thesis is constituted by elevation maps built after the processing of thousands of RGB images, captured by an UAV during a field campaign conducted by Prof. Pedro Pina and Sandra Heleno in Antarctica in 2018. Several flight surveys were conducted along the ice-free areas of Barton Peninsula in King George Island (62º14'S, 58º46'W), at very low heights (10 meters above ground) to obtain very detailed imagery. Each flight surveyed an area of about 80 m $\times$ 80 m with a forward and side overlap of 80%, so the same region would be covered from multiple views. The dataset includes both regions of stone circles with a stationary quasi-periodic structure and regions with irregular ones [3] [5].

The dataset used to assess the stone circles' identification is composed of 20 elevation images with size 1000×1000 pixels and spatial resolution between 2.2 and 2.4 cm/pixel. Given an input image, the ground-truth for the stone detection problem corresponds to the center coordinates of the stone circles completely contained in the image.

As for the contour extraction problem, the ground-truth consists of the coordinates of the stones' boundary yet, these were only obtained for 24 stone circles (2 circles per site), given the complexity and uncertainty of the manual marking task.

Both center and boundary coordinates were manually created with the help of Prof. Pedro Pina. Figure 4.1 shows examples of the ground-truth points obtained for both problems.



(a) Ground-truth points of the stones' centers.

(b) Ground-truth points of the stones' boundary.

Figure 4.1: Examples of the ground-truth for stone detection and boundary extraction (detail).

4.2 Evaluation metrics

The method's evaluation aims at measuring the quality of the location of the stone circle's center and boundary, in comparison with the "ideal" placement given by the ground-truth information. Each of these problems will be presented and discussed separately.

4.2.1 Stone circle detection

The performance of the proposed methods for the stone detection is assessed by establishing a one-to-one correspondence between the ground-truth circles and the algorithm's detections. This association is achieved according to a matching criteria which takes into account the closeness between the ground-truth coordinates and the algorithm's output. Details about the matching method are presented in Appendix A.

The outcome of this process results in the classification of the ground-truth points in the categories "detected" or "not detected" and the candidates to the stones' center into "correct" or "false" detections [34]. This information can be summarized by the following quantities:

- A **True Positive** (TP) occurs when a detection is paired to an existing ground-truth point.
- A **False Positive** (FP) occurs when a detection is not paired with any existing ground-truth point.
- A **False Negative** (FN) occurs when a ground-truth point has no corresponding match detection.

Figure 4.2 illustrates an example of these three types of events that may occur at the output of the matching algorithm.

In order to detect a valid pair, the matching criteria states that the localization error between a GT and a detection point must be sufficiently small and inferior to a given distance λ (see Appendix A). The value

chosen for λ is the average stone's radius $R = 30$. Selecting a larger maximum distance would produce unreliable results as the classifier may be associating points that belong to different stone circles. The green points marked in Figure 4.2 verify this condition and represent a matched GT and detection point (true positive). By the same token, a detection whose Euclidean distance from a ground-truth point is higher than the maximum distance allowed, should not be paired with it. An example of two unmatched GT points (false negatives) is represented as yellow markers in Figure 4.2, as well as an unmatched red detection (false positive).

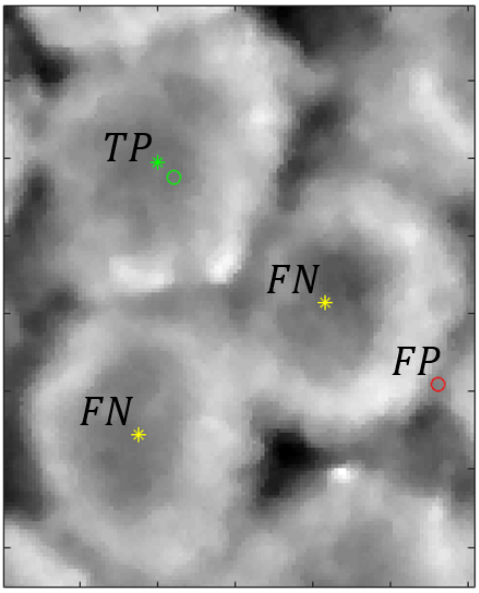


Figure 4.2: Matching output: correspondence between the ground-truth points (stars) and the output of the detection algorithm (circles).

Figure 4.3 shows an example of the classifier's output for the Template Matching algorithm obtained with a threshold $Th = 25\%$: the detections classified as TP are marked as green circles while the corresponding ground-truth points are green stars; the detections classified as FP are marked as red circles and the missing detections, FN, are represented as yellow stars.

The number of TP, FP and FN allows the computation of Precision and Recall, the quantities used to assess the binary classifier's performance. The first value measures the percentage of the classifier's positive examples correctly assigned while the latter indicates the percentage of real positive examples correctly identified. Therefore, Precision and Recall are computed according to

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}. \tag{4.1}$$

It is useful to combine the two statistics into a single performance index. The F-score quantifies the balance between the aforementioned metrics, corresponding to the harmonic mean

$$\text{F-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{4.2}$$

The detector's performance depends on one or more parameters being interesting to represent the

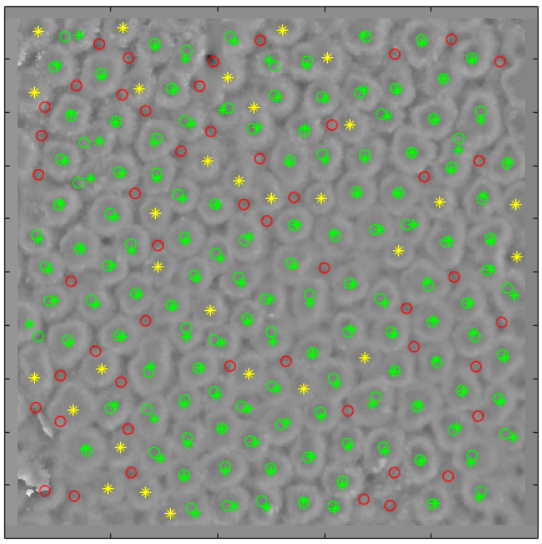


Figure 4.3: Classifier's output for the Template Matching algorithm: true positives (green), false positives (red) and false negatives (yellow).

measure Precision as a function of Recall through a receiver operation characteristic (ROC) curve, where the effect of a single parameter of interest is studied. For example, Figure 4.4 shows the evolution of the number of False Positives and False Negatives for the Template matching method, when the threshold parameter Th varies and the radius R and width δ are fixed.

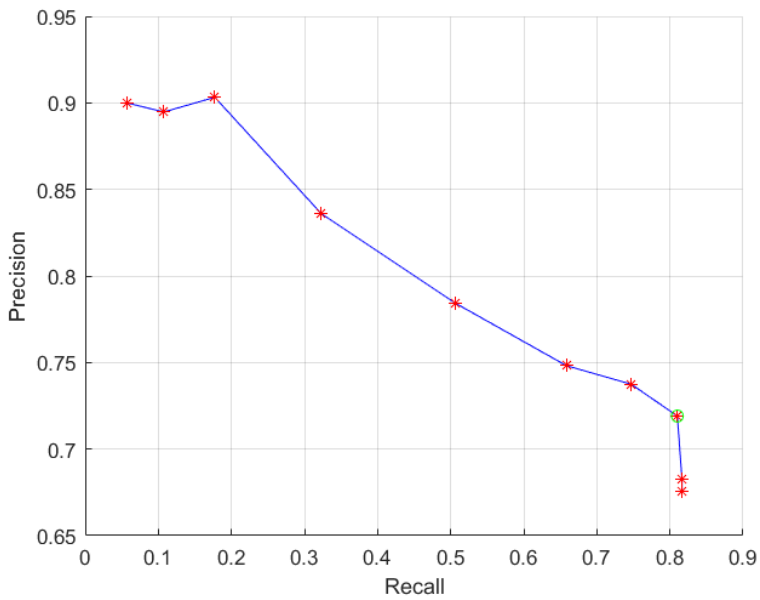


Figure 4.4: Precision vs Recall curve. The green mark corresponds to the value Th that maximizes the F-score.

Note that there is a trade-off between Precision and Recall: a highly permissive classifier presents a high Recall and reduced Precision (bottom-right graph points), while a demanding detector presents a high Precision, compromising the Recall (top-left graph points).

4.2.2 Contour Delineation

In order to assess the performance of the stone circle's delineation methods, the localization error between the ground-truth contour (B) and the detected contour (A) is analyzed. For each contour point detected by the algorithm, p, the displacement from the closest ground-truth point, given by

$$d(p, B) = \min_{q \in B} \|p - q\|, \tag{4.3}$$

allows its classification in one of the following categories:

- A **Correct point** (CP) occurs when the displacement is null, $d(p, B) = 0$, and there is no error.
- A **Small error** (SE) occurs when the displacement is within the interval $]0, \Delta[$, where Δ is the maximum deviation allowed.
- A **Gross error** (GE) occurs when the displacement exceeds the maximum deviation allowed, $d(p, B) \geq \Delta$.

The percentage of the errors is used as a distortion measure between both contours [10] yet, since not all errors have the same importance, the CP and SE are irrelevant when compared to the GE, as it is assumed the detected points are very close to the reference [35]. Therefore, the GE percentage determines the method's performance and includes the detected points whose deviation from B is above 15% of the average stone's radius, $R = 30$.

Figure 4.5 shows an example of these three types of events that may occur when evaluating the stone's contour extraction.

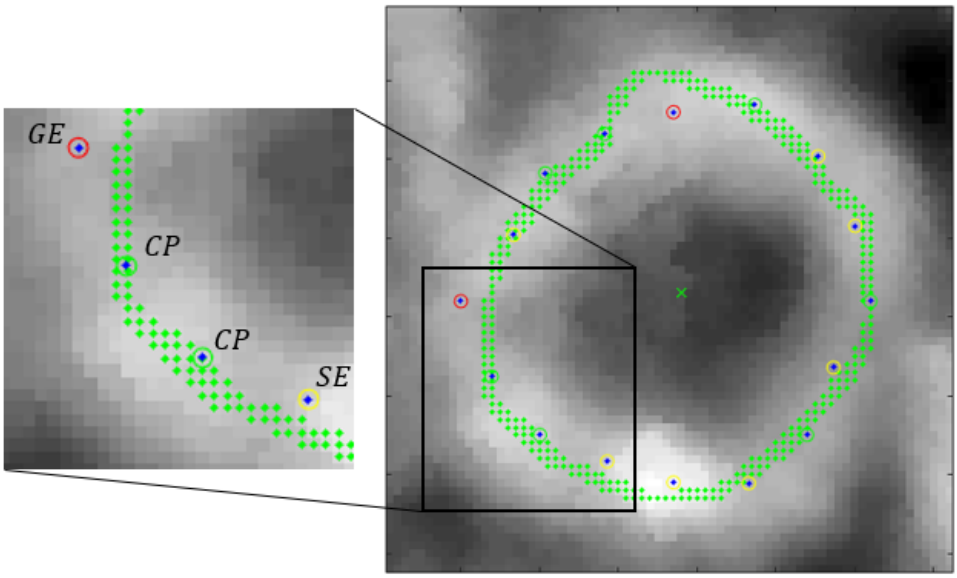


Figure 4.5: Contour evaluation output: a detected contour point (blue) is classified as a CP(green circle), SE(yellow circle) or GE(red circle), according to its distance to the closest ground-truth point (green).

The width of the ground-truth contour exceeds one pixel to include the uncertainty associated with its manual delineation performed by an expert, therefore the multiple green points shown in Figure 4.5.

As mentioned, the performance of the proposed methods depends on a set of parameters. By defining several discrete values for the parameters and computing the evaluation metrics described for each combination, one can determine the combination that leads to the best algorithm's output. Sections 4.3 and 4.4 describe this procedure, for the problems of stone detection and contour extraction, respectively.

4.3 Experiments - Stone Circle detection

For the stone detection problem, the optimal combination of parameters corresponds to the one that maximizes the average F-score for all images on the dataset. This section presents the study of the parameter's influence on the performance of each detection method.

4.3.1 Template Matching

In the Template Matching (TM) algorithm there are three hyperparameters. The algorithm's performance depends on the Template's radius, R, its width, δR, and on the threshold Th that selects the candidates based on their score image value, in the peak detection stage. The following values were considered for each parameter: $R \in \{25, 30, 35, 40\}$, $\delta \in \{0.1, 0.2, 0.3, 0.4\}$, $Th \in \{25, 30, 35, 40, 45, 50, 55\}$. The algorithm was executed for all $(\#R \times \#\delta \times \#Th)$ parameter combinations, for each image of the dataset and average values of the evaluation metrics were obtained. Figure 4.6 shows the average F-score computed per parameter combination and Figure 4.7 the average Precision and Recall metrics.

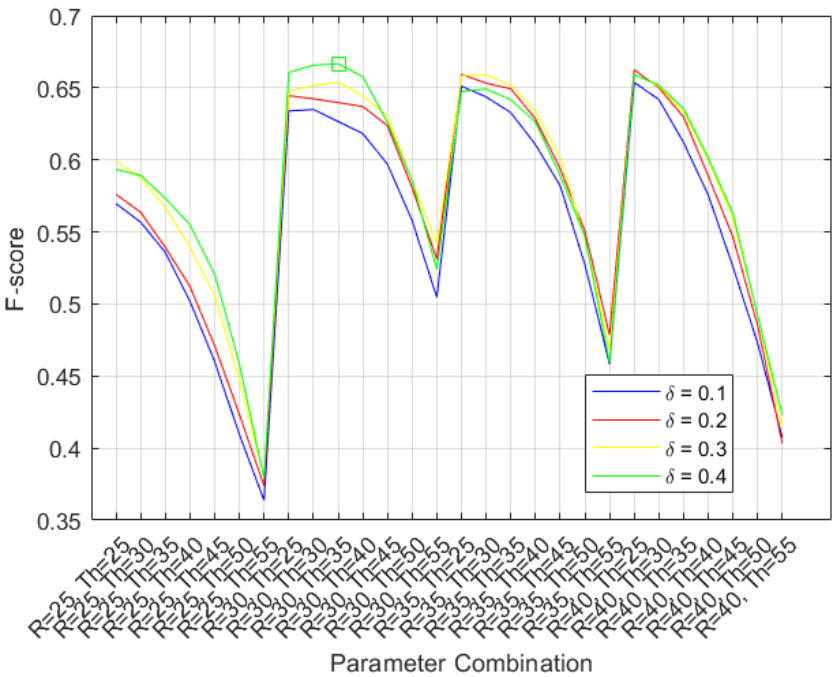
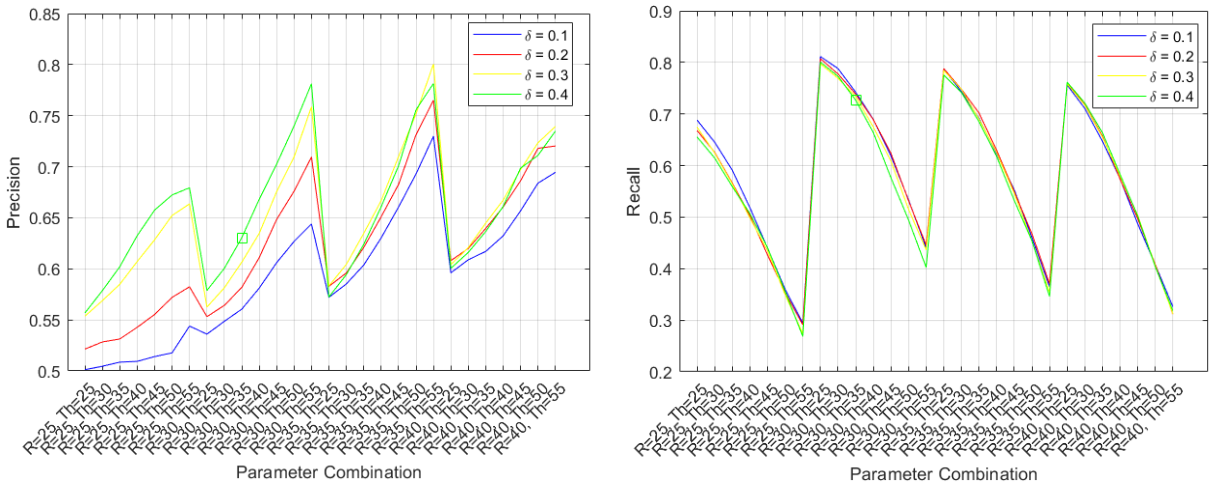


Figure 4.6: Template Matching evaluation: Average F-score per parameter combination.



(a) Average Precision per parameter combination.

(b) Average Recall per parameter combination.

Figure 4.7: Template Matching evaluation: Average Precision and Recall per parameter combination.

The horizontal axis of the graphs represents the radius R and threshold Th values while different widths δR are described by different curves.

Figure 4.6 shows that an increasing threshold value Th leads to a reduction of the F-score value, regardless of the radius R and width δ chosen. The reduced Recall obtained for combinations with higher thresholds causes the F-score to decrease, as it is observed in Figure 4.7(b). It is also noticeable that different width values do not affect the Recall, as the curves are rather similar, as opposed to the ones of Precision. Figure 4.7(a) shows that lower δ values lead to a larger number of false positives that reduces as R increases. Moreover, the Precision increases for larger width values, suggesting a higher and wider template leads to a better stone circle identification, at least for a radius inferior to $R = 40$.

The maximum average F-score for the Template Matching algorithm is 66.67% which is obtained with the parameters $R = 30$, $\delta = 0.4$ and $Th = 35\%$, corresponding to the green square mark superimposed in the graphs aforementioned. Although this is the optimal parameter configuration, Figure 4.6 also shows there are other 10 parameter combinations that achieved average F-score values with less than 1.35% of the optimal value.

In addition to the evaluation metrics discussed, it is also important to compute the computational time of the detection method as a function of the parameter combinations. Figure 4.8 shows the average computational time taken by the Template matching method, for all images of the dataset, per parameter configuration. For the majority of the parameter combinations, the method takes no more than a few seconds to produce the detections.

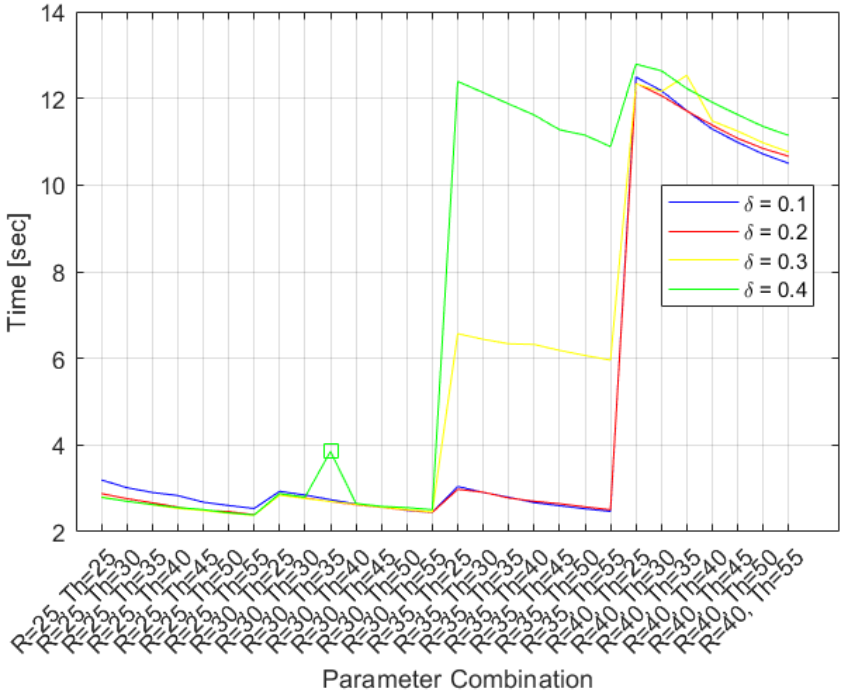


Figure 4.8: Template Matching: Average Time per parameter combination.

The average computational time of the Template Matching method (see Figure 4.8) increases as R increases, as a consequence of the larger number of pixels in the template and the larger number of computations performed in the convolution operation. This result is mostly visible for $R = 35$ and $R = 40$. Note that for the same R, the time gets reduced as Th increases, suggesting the less detections obtained (high Precision), the less the spatial filter contributes to the computational time.

4.3.2 Sliding Band Filter

The SBF method depends on three hyperparameters: the number of auxiliary radial lines or directions considered, N, the minimum gradient magnitude required for the computation of the alignment score, g_{min}, and the threshold Th used in the peak detection stage. We assumed the following ranges for the hyperparameters: $N \in \{16, 32, 64, 128\}$, $g_{min} \in \{0.0, 0.005, 0.0125, 0.02, 0.04\}$ and $Th \in \{60, 65, 70, 75, 80\}$. The evaluation metrics were computed for each configuration, as in the case of the Template Matching method. Figure 4.9 illustrates the average F-score as a function of the parameters, where the horizontal axis represents the configurations of g_{min} and Th. The average Precision and Recall measures per parameter combination are shown in Figure 4.10. The graphs are composed of 4 curves, each associated with the number of directions, N.

Figure 4.9 shows that the Sliding Band Filter method achieves higher performance values compared to the Template Matching, with some parameter configurations reaching F-scores above 80%. The same graph reveals the SBF is not sensitive to the number of directions, as the 4 curves are similar, regardless of the threshold and g_{min} value. One can conclude that higher values of g_{min} result in a reduction of the F-score, mostly caused by the increase of the number of misdetections (see Figure 4.10(b)). For $g_{min} = 0.04$, the SBF method's performance decreases significantly, suggesting this minimum gradient

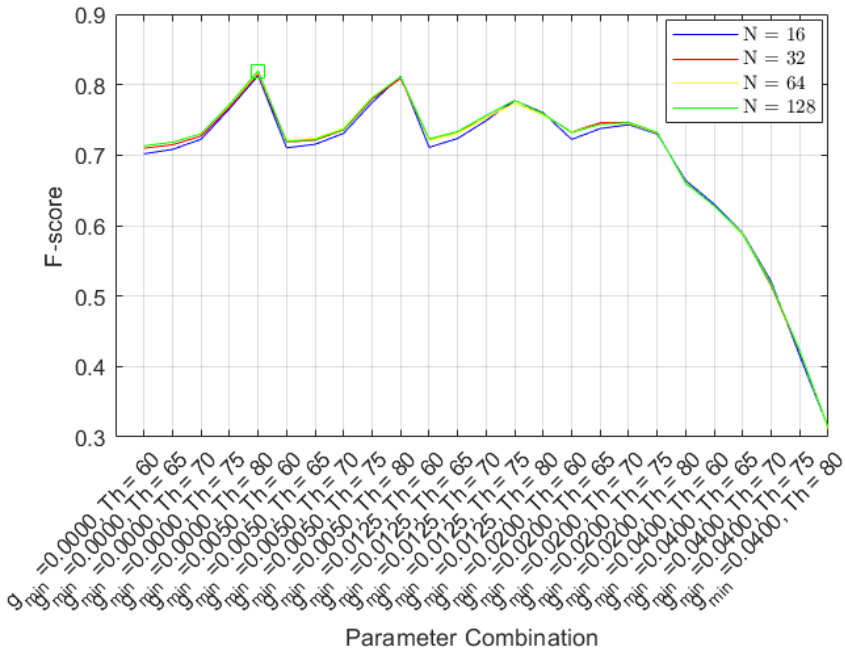
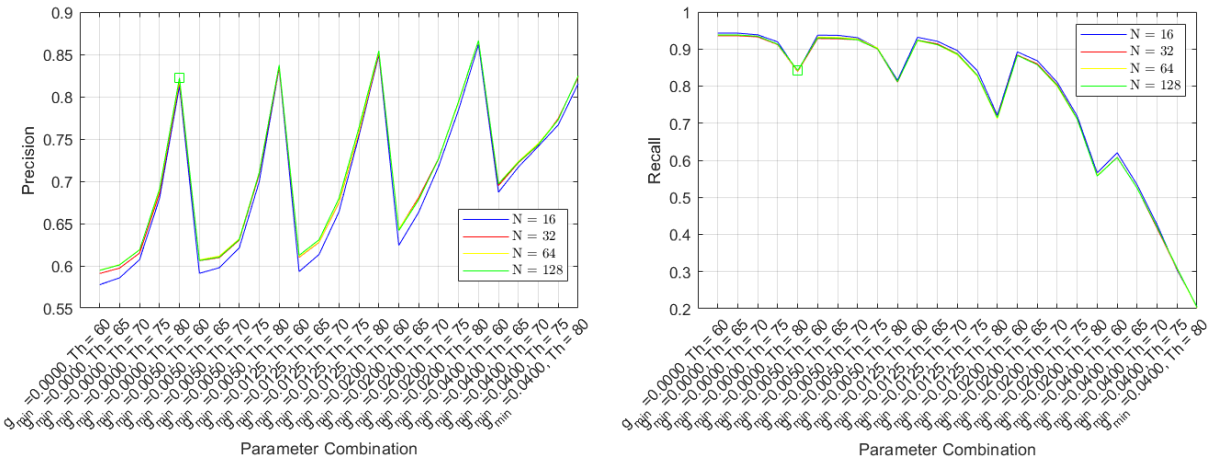


Figure 4.9: Sliding Band Filter evaluation: Average F-score per parameter combination.

value is so excessively high that the gradient distribution is discarded for the majority of the image pixels. On the other hand, the Precision measure (see Figure 4.10(a)) achieves peaks on the parameter combinations with the highest Th values, and the value of these peaks increase as the minimum gradient magnitude gets larger.



(a) Average Precision per parameter combination.

(b) Average Recall per parameter combination.

Figure 4.10: Sliding Band Filter: Average Precision and Recall per parameter combination.

The minimum number of false alarms, on average, is achieved with $g_{min} = 0.02$ yet, the large number of False Negatives for the corresponding parameter configuration limits the F-score, as shown in Figures 4.10(a) and 4.10(b). This trade-off is less affected for a more permissive method, with the optimal minimum gradient magnitude being $g_{min} = 0.0$, implying the alignment score should be calculated regardless of the intensity of the gradient vector at a given point. The discrete values chosen for g_{min} were obtained specifically for the dataset, after an analysis of the gradient intensity both in regions on

the stone circles' border and in regions without stones. Since it is possible to remove this variable from the parameter set (the gradient's magnitude is positive), the algorithm does not depend on the absolute value of the gradient's magnitude, allowing its application to any digital elevation map image, other than the ones examined in this work.

Given the similarities between the performance of the SBF method for different N values, an analysis of the computational time can provide useful clues about the best parameters to choose. Figure 4.11 shows the average computational time obtained for each g_{min} and Th combination and for the 4 number of directions tested. As expected, the increased complexity of the SBF causes the computational time to increase: the method analyzes each image in the order of minutes, a significant difference considering the few seconds taken by Template Matching.

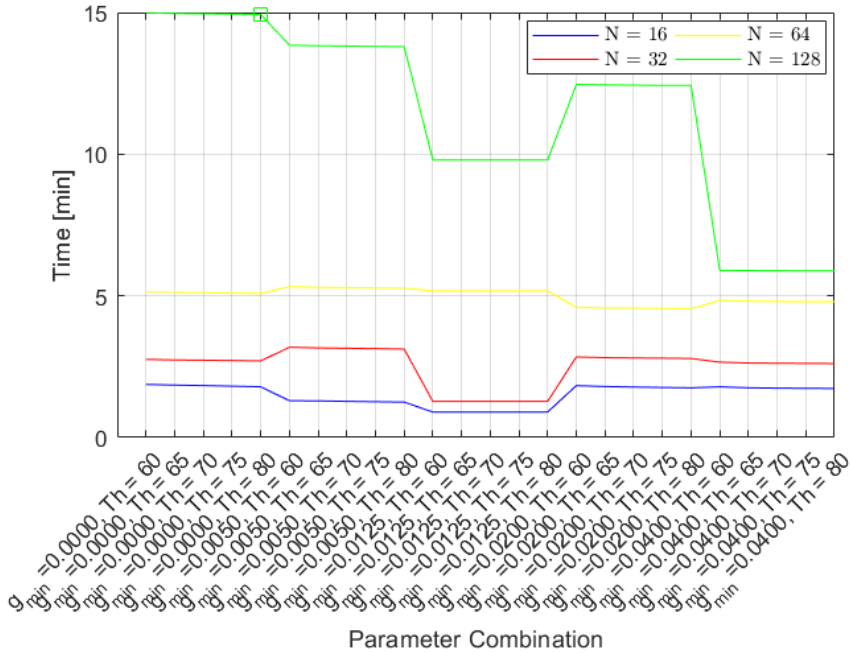


Figure 4.11: Sliding Band Filter: Average Time per parameter combination.

The maximum average F-score = 81.95% was achieved with the parameters $N = 128$, $g_{min} = 0$ and $Th = 80\%$ (see Figure 4.9). From analyzing Figure 4.11, it is observed that the computational time increases with the number of directions considered. Moreover, its value is approximately constant for the majority of the parameter configurations of g_{min}, Th and the three smaller N values. The computational time for $N = 128$ decreases as g_{min} gets larger and is consistently greater than the execution time for SBF considering smaller number of directions. This can be explained by the fact that the alignment score is computed for all image pixels when $g_{min} = 0$ yet, as this threshold increases, the less calculations are executed. This difference is not as relevant for the remaining N values.

Given the fact that the average time gets reduced about 8 times when the number of directions decreases in the same proportion (from $N = 128$ to $N = 16$) and the average F-score value is not compromised, the $N = 128$ value in the optimal parameter combination may be replaced by $N = 16$.

4.3.3 Dynamic Programming

The Dynamic Programming method depends on the number of directions, N, and the threshold, Th, similarly as the SBF method, and has two additional hyperparameters related to the transition cost in the contour computation: m indicates the slope of the cost function and β determines the biggest radius gap allowed. The values assumed for these parameters are the following: $N \in \{16, 32, 64, 128\}$, $Th \in \{60, 65, 70, 75, 80\}$, $m \in \{0.03, 0.3, 3\}$, $\beta \in \{1, 3, 5, 10\}$. As described for the other proposed methods, the DP algorithm was executed and evaluated for all parameter configurations but for a smaller subset of the initial dataset (4 randomly chosen images were obtained). The DP method was executed for fewer images than the TM and SBF given the unfeasible computational time taken to test each parameter combination per image (up to tens of minutes per image). Each data point plotted in the graphs of Figures 4.12 and 4.13 corresponds to the average evaluation measure obtained using the parameters in the horizontal axis (m, β, Th) and the N parameter, associated with each curve. Only few configurations are labelled in the horizontal axis due to visibility issues.

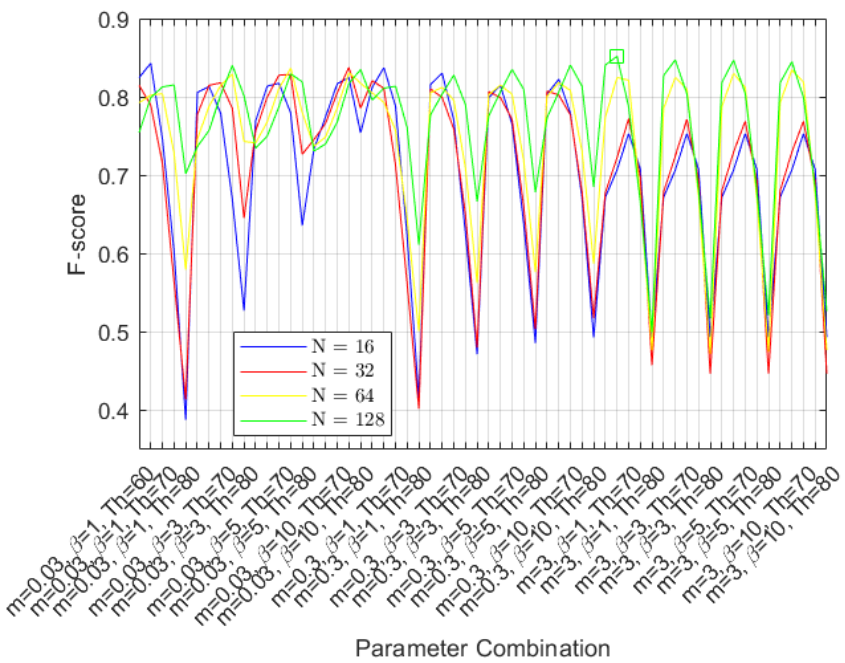
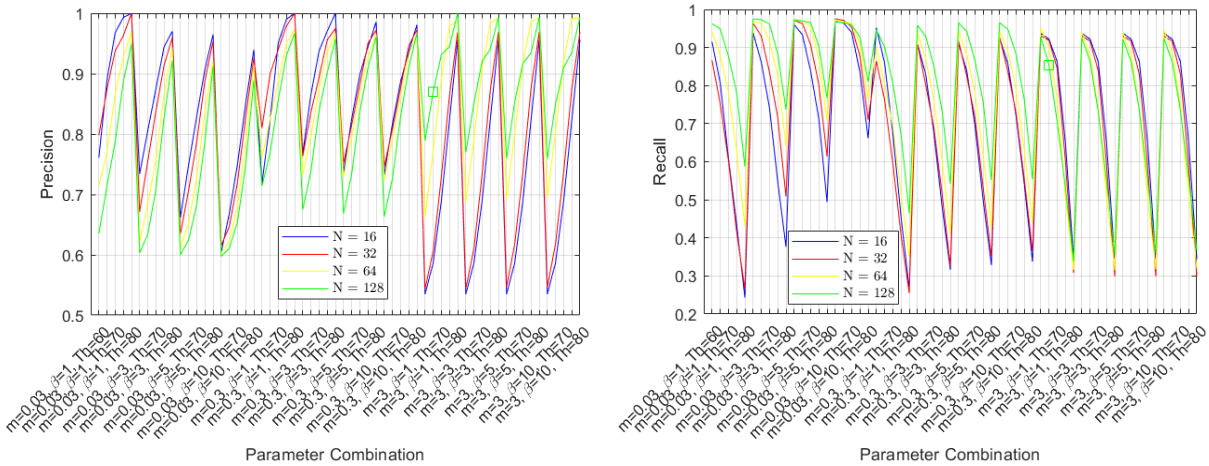


Figure 4.12: Dynamic Programming: Average F-score per parameter combination.

The oscillations observed in the average F-score measure (see Figure 4.12) illustrate the aforementioned trade-off between Precision and Recall: an increase in Th may lead to a higher Precision and reduced Recall and a smaller Th increases the number of False Positives and reduces the False Negatives, regardless of the m and β configuration. Figure 4.13(a) shows the Precision decreases for a larger N, for all configurations up to $m = 3$. For a larger slope m, the maximum Precision is achieved as N increases. On the other hand, the Recall increases with N for $m = 0.03$ and $m = 0.3$ (see Figure 4.13(b)) and decreases as N is higher for $m = 3$ yet, the difference is not as relevant as the one observed for Precision, as the number of directions change. When $m = 3$, it is noticeable that the DP's performance is not affected by the increase in β. For the remaining m values, as β increases, the Recall is higher,

particularly for larger Th values, while Precision is smaller for a smaller Th. Overall it is observed that the F-score presents peaks for smaller Th values, whichever m and β configuration are chosen. The maximum average F-score is 85.18% and is obtained with the parameters $N = 128$, $m = 3$, $\beta = 1$ and $Th = 65\%$ yet, since the performance does not depend on β for a larger cost slope, any configurations with $N = 128$, $m = 3$ and $Th = 65\%$ would lead to an average F-score above 84%.

It is also worth noting there are configurations for $N = 16$ and $N = 32$ resulting in an extremely reduced number of False Positives, up to the point where all the detections obtained, on average, form a match with a ground-truth circle.



(a) Average Precision per parameter combination.

(b) Average Recall per parameter combination.

Figure 4.13: Dynamic Programming: Average Precision and Recall per parameter combination.

Despite the improved performance achieved by the DP method in comparison with the other proposed detectors, an analysis of its complexity is relevant for choosing the best solution among the three presented. Figure 4.14 shows the computational time for the DP execution is approximately constant for all configurations tested with $N = 16$ and $N = 32$, with values of around 4 min and 9 min per image, respectively. On the other hand, the average time per image presents some oscillations for larger N values and, as expected, the complexity increases as more directions are considered: the method takes above 10 min for all parameter combinations and up to more than 70 min. The ratio between the time taken for different N values does not follow a consistent rule.

4.3.4 Methods comparison

The mean value and standard deviation of the evaluation metrics obtained for the optimal parameters per method are shown in Table 4.1.

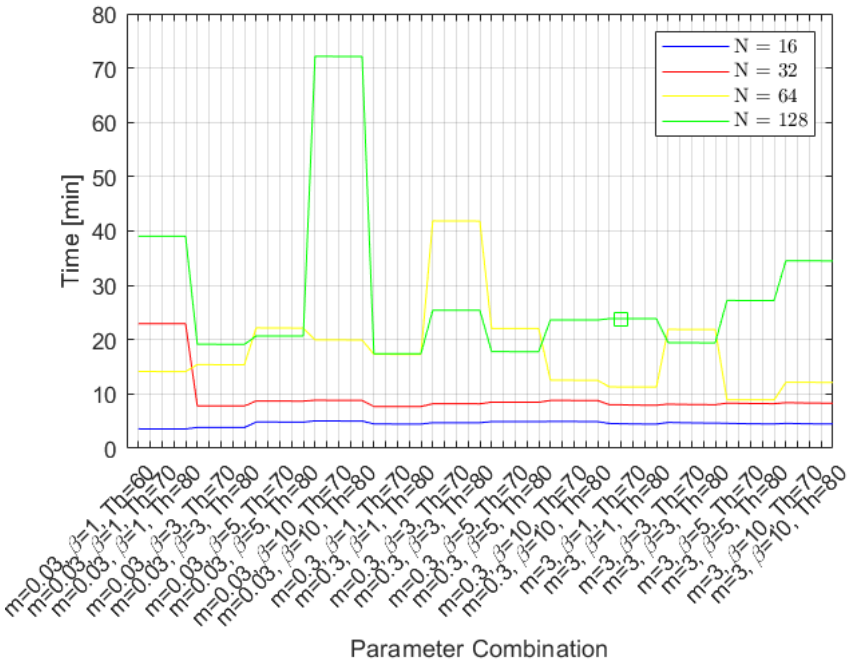


Figure 4.14: Dynamic Programming: Average Time per parameter combination.

Table 4.1: Optimal performance measures of TM, SBF and DP methods: mean and standard deviation computed for 20 images (except for DP, where 4 images were tested).

Method	F1[%]	Precision [%]	Recall [%]	Time [min]	Parameters
TM	66.67 (8.90)	63.05 (11.73)	72.67 (10.07)	**0.06 (0.08)**	$R = 30, \delta = 0.4, T = 35\%$
SBF	81.95 (6.56)	82.22 (14.46)	84.23 (8.76)	14.93 (1.01)	$N = 128, g_{min} = 0, T = 80\%$
DP	**85.18 (6.64)**	**87.01 (8.07)**	**85.47 (14.76)**	23.87 (12.85)	$N = 128, m = 3, \beta = 1, T = 65\%$

The best performance was obtained by the DP method which achieves an average F-score of 85.18%. This method and SBF (F-score$= 81.95\%$) clearly outperformed the Template Matching algorithm whose average F-score is 66.67%. The inferior performance of the Template Matching can be related to its rigid assumptions on the stone's shape and consequent inability to detect different sized and deformed objects, frequent across the same site. The SBF method allows more flexibility in the circles' detection as the relevant feature in the shape analyzed is its circular geometry and not its height. Finally, the DP method introduces an additional geometry constraint on the SBF's criteria that consists in an improved noise robust solution.

Table 4.1 also confirms the complexity difference between TM and the best methods. Its average computational time corresponds to a reduction of more than 150 times the time taken by SBF and DP. Given the excessive computational time obtained for the best methods with the optimal parameters, it is reasonable to discuss if a reduction in the number of directions would significantly compromise the F-score or it would be possible to slightly abdicate the performance by choosing a smaller N in exchange for a reduction in complexity. Table 4.2 shows the results corresponding to sub-optimal parameter configurations for methods SBF and DP.

Table 4.2: Performance measures of SBF and DP methods for sub-optimal parameters: mean and standard deviation computed for 20 images (except for DP, where 4 images were tested).

Method	F1[%]	Precision [%]	Recall [%]	Time [min]	Parameters
SBF	81.29 (6.93)	81.25 (14.43)	**83.91 (9.07)**	**1.79 (0.16)**	$N = 16, g_{min} = 0, T = 80\%$
DP	**84.31 (6.07)**	**88.99 (8.13)**	80.71 (8.39)	3.58 (1.61)	$N = 16, m = 0.03, \beta = 1, T = 65\%$

It is interesting to note from Tables 4.1 and 4.2 that maximizing the F-score for the DP method with $N = 128$ implies choosing restrict cost parameters $(m = 3)$ while, for a smaller number of directions, $N = 16$, it is preferable to reduce the transition penalizations by selecting $m = 0.03$. This may come as a consequence of the following: when choosing a larger angle interval $(N = 16)$ it is not adequate to assume the radius for the next direction is close to the one for the previous direction as there is more uncertainty given the great angular displacement between those directions. On the other hand, when the angle interval is smaller $(N = 128)$, it is reasonable to accept that the distance from the stone's boundary to the center does not vary significantly hence the increased weight given to the circular constraint expressed by the higher transition cost.

As can be concluded from Figure 4.9 and Tables 4.1 and 4.2, a reduction from $N = 128$ to $N = 16$ does not affect significantly the SBF's F-score yet, the computational time is reduced about 8 times (from 14.93 to 1.79 min). The same reduction on N in the DP not only results in a higher F-score (84.31%) when compared to the one obtained for the optimal parameters of the SBF (81.95%), but it also reduces its average computational time in about 7 times (from 23.87 to 3.58 min). For this reason the DP method with $N = 16$ and the other corresponding optimal parameters leads to the best compromise between performance and time complexity.

Figure 4.15 shows the performance of the proposed methods in two images extracted from different sites, chosen to demonstrate the variability between the input images. The example on the left shows well defined stone circles with clear and distinct borders, while the example on the right presents regions with stones that are still developing and with ambiguous borders, thus increasing the detection's difficulty. For each image it is shown the ground truth information (stars), the detected circles (circles) and the TP (green), FP (red) and FN (yellow).

The methods performance for the left image is similar, although the TM presents additional false positives in the intersection of existing stone circles. This is due to the similarity of the template with the half-torus shape defined by three adjacent stone circles. Nevertheless, most circles are detected by the three methods yet, the SBF method outputs a smaller number of false negatives when compared with the DP. When assessing the methods performance for the more difficult image, one can observe the SBF method achieved the best result, with the minimum number of false detections (FP) of the three methods.

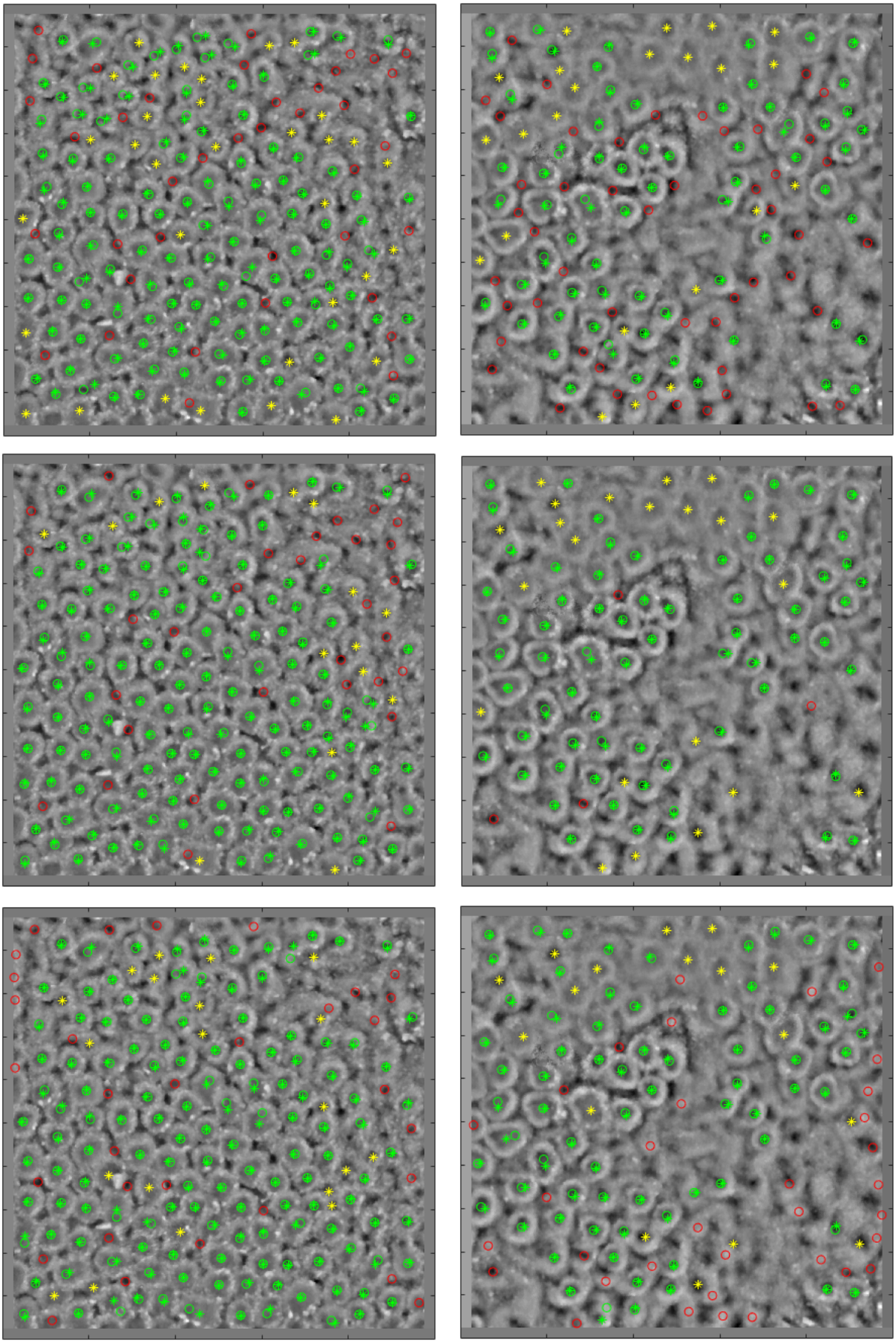


Figure 4.15: Examples from 2 field sites: outputs of Template Matching (top), SBF (middle) and DP (bottom). Colour code: true positives (green), false positives (red) and false negatives (yellow).

4.4 Experiments - Contour Delineation

The optimal parameter configuration of the delineation methods must minimize the average GE percentage measured for the 24 circles. This section presents the delineation results and a discussion on the effect of the parameters in the performance for the methods SBF and DP.

4.4.1 Sliding Band Filter

Since the SBF method measures the alignment score to obtain the stone's contour, it depends on the same parameters defined for the stone's detection problem. However, in the contour extraction task, the number of directions N is set to a high value $(N = 360)$ so the contour is well defined. The threshold parameter Th does not affect the contour computation, which implies g_{min} is the only hyperparameter to vary and the possible values defined for it are $g_{min} \in \{0.0, 0.005, 0.0125, 0.02, 0.04\}$. The method was applied to 24 individual stone circle images, being the average evaluation values computed for each parameter configuration. Figure 4.16 shows the average CP, SE and GE percentages as a function of g_{min}.

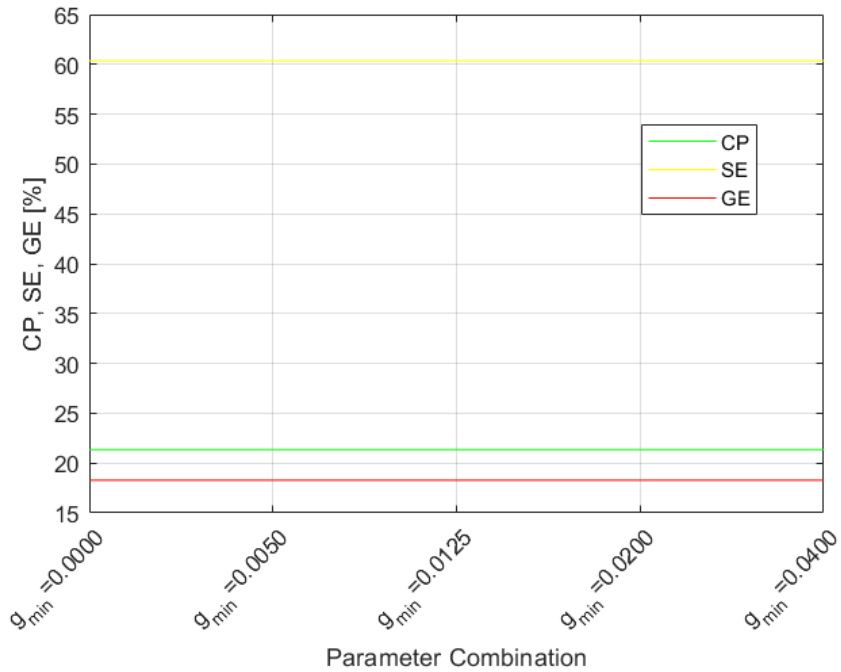


Figure 4.16: Average CP, SE and GE percentages per parameter combination for the SBF method.

From Figure 4.16 it is observed the proposed method is not sensitive to different minimum gradient magnitude thresholds, as the curves relative to the CP, SE and GE are approximately constant. The minimum average GE percentage obtained with SBF is 18.3%.

4.4.2 Dynamic Programming

The parameters that influence the DP method in the contour extraction task are the same ones described in the stone's detection problem, similarly to the SBF's case. The threshold Th is discarded

from the parameter configurations, just as the number of directions that is set to $N = 360$. Therefore, the cost parameters m and β are the only hyperparameters that affect the contour computation and assume the following values: $m \in \{0.03, 0.3, 3\}$, $\beta \in \{1, 3, 5, 10\}$. The DP method was executed for each parameter configuration and the metrics CP, SE and GE were computed for the 24 test circles. Figure 4.17 illustrates the metrics' average values as a function of the parameter combinations.

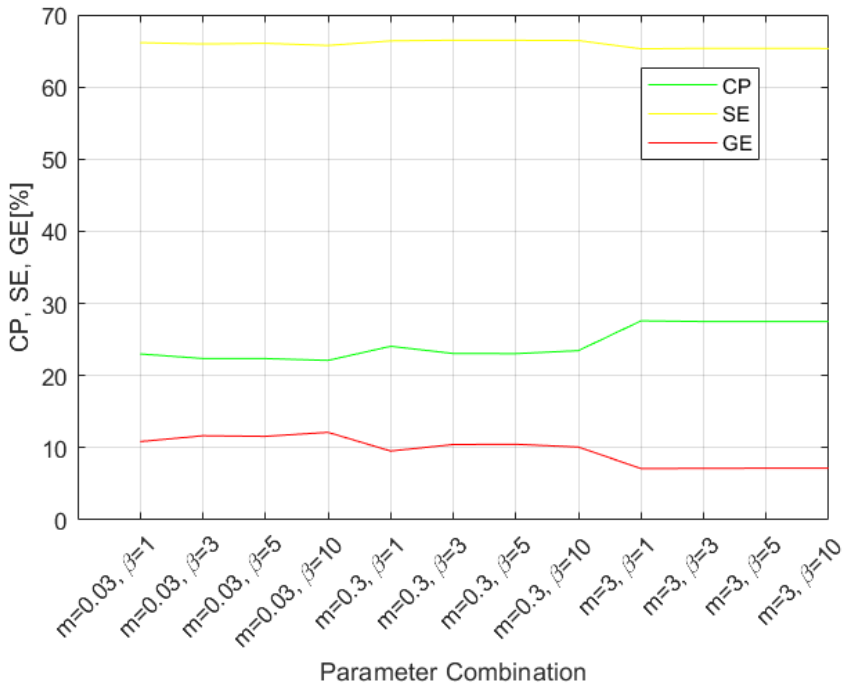


Figure 4.17: Average CP, SE and GE percentages per parameter combination for the DP method.

It is clear in Figure 4.17 that the number of gross errors obtained with the DP method is significantly smaller than the obtained with SBF. Figure 4.17 also shows that the GE percentage is reduced as m increases while the correct points percentage has the opposite evolution. The influence of β is residual since the curves are approximately constant when m is fixed. Moreover, as opposed to the other measures, the number of small errors remains unchanged for the different parameter combinations. These results show that smoother transitions defined by a large cost slope m led to a more accurate contour detection because the stone's boundary changes slowly. It can also be noted that the method with the most permissive cost parameter, $m = 0.03$ and $\beta = 10$, produced the greatest number of gross errors.

4.4.3 Methods comparison

The mean value and standard deviation of the evaluation metrics obtained with the parameters that minimized the GE percentage are shown in Table 4.3, for each method. The Dynamic Programming method achieved the best performance, with a percentage of gross errors of 7.12%, obtaining a reduction of about 10% when compared to the SBF approach. It is also noticeable that the DP method outperformed the SBF method for the majority of the 24 test images, given the smaller standard deviation. Furthermore, the number of correct contour points was higher for the DP method and this increase

in performance comes at the expense of a larger complexity: the DP method's computational time per test image is about 10 times the one taken by the inferior approach.

Table 4.3: Optimal performance measures of SBF and DP methods: mean and standard deviation computed for 24 stone circles.

Method	CP [%]	SE [%]	GE [%]	Time [s]	Parameters
SBF	21.35 (9.37)	**60.37 (11.69)**	18.28 (18.76)	**5.53 (1.84)**	$N = 360$, $g_{min} = 0$
DP	**27.59 (9.78)**	65.29 (10.58)	**7.12 (12.31)**	50.18 (14.27)	$N = 360, m = 3, \beta = 1$

Table 4.4 shows the optimal performance measures computed for $N = 64$. Note that the minimum gross error percentage for the DP with $N = 360$ was achieved with a larger cost slope m than the optimal m for a smaller number of directions, just like in the detection problem. This results from the fact that a larger angle interval leads to more flexibility in the choice of the radius while for a smaller angle interval, the displacement between consecutive radius values should be greatly penalized. Surprisingly, the average time taken by the DP method is inferior than the one taken by SBF for $N = 64$. Changing $N = 360$ to $N = 64$ led to a reduction of the average computational time of about 15 times in the DP method, without compromising the GE percentage. On the other hand, for the SBF, the complexity reduction is not as significant as in the DP case and the performance is also maintained.

Table 4.4: Performance measures of SBF and DP methods for sub-optimal parameters: mean and standard deviation computed for 24 stone circles.

Method	CP [%]	SE [%]	GE [%]	Time [s]	Parameters
SBF	18.49 (8.49)	**63.48 (12.98)**	18.03 (18.83)	4.40 (1.30)	$N = 64$, $g_{min} = 0$
DP	**24.61 (10.34)**	68.23 (13.01)	**7.16 (13.73)**	**3.17 (0.81)**	$N = 64, m = 0.3, \beta = 1$

Figure 4.18 shows the performance of the SBF (left) and DP (right) methods for three stone circles from different sites. The examples presented illustrate some of the natural variability regarding the circular shape of the structures. In each image it is represented the ground truth information (green), the detected contour points (blue) and the CP (green circle), SE (yellow circle) and GE (red circle). For visualization purposes, the experiments displayed were obtained with $N = 64$ directions and for the corresponding best parameter configuration ($m = 0.3$, $\beta = 1$).

From observing Figure 4.18 one can confirm the dependency between consecutive radius imposed in the DP method, given the smooth contours obtained. On the other hand, the contour points computed with the SBF method present a larger variation for neighboring directions as there is no global constraint connecting the contour detections and preventing large discontinuities. Moreover, these experiments show that the DP method is a more gradient-noise robust solution since the number of outliers is significantly reduced.

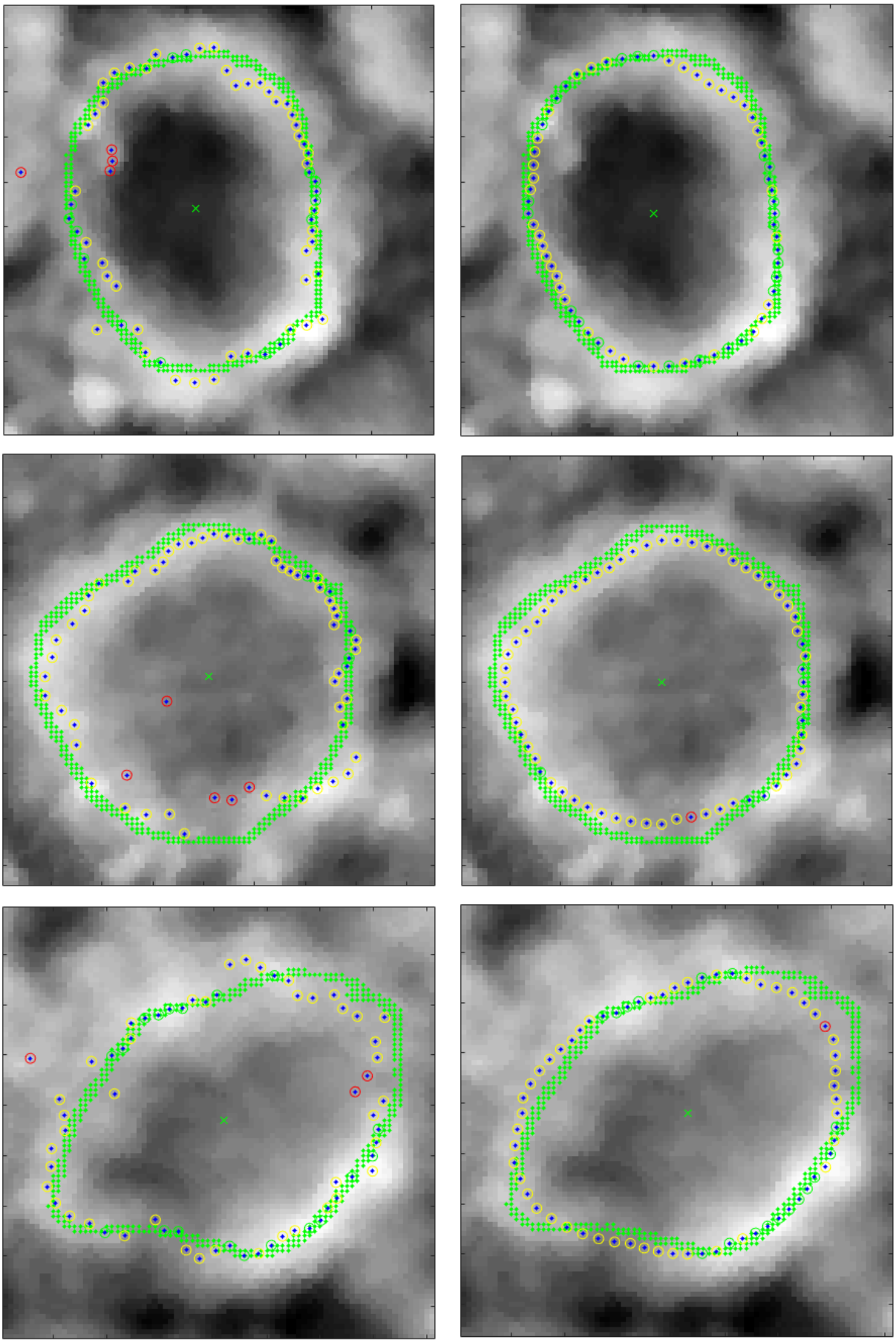
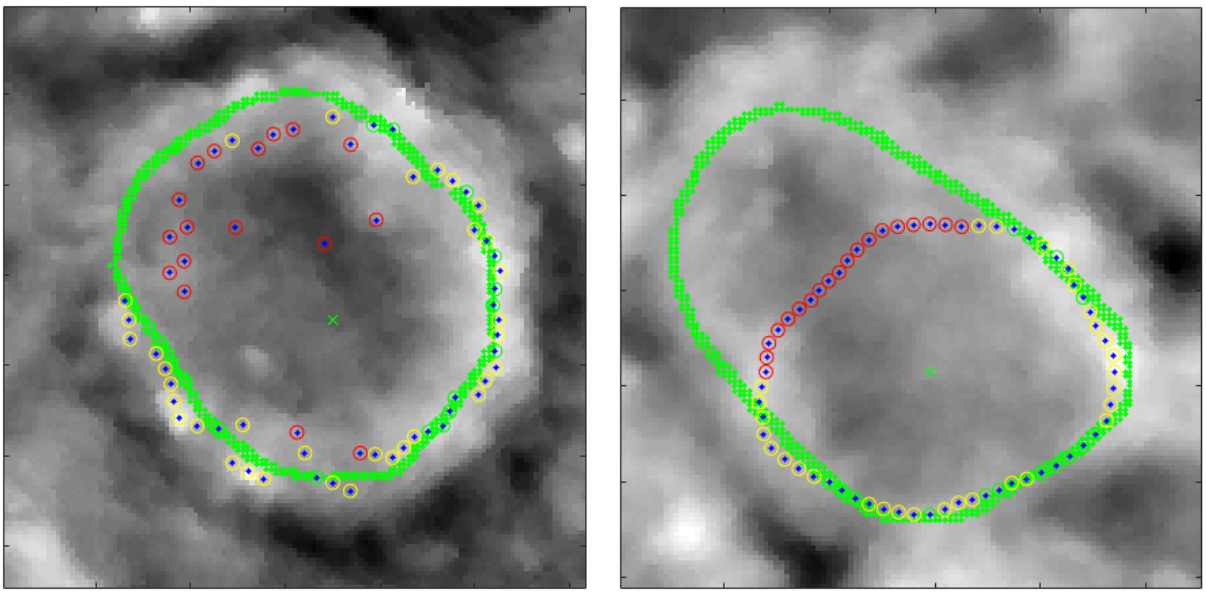


Figure 4.18: Examples of contour extraction for 3 circles with $N = 64$: outputs of SBF (left) and DP (right). Colour code: correct point (green), small error (yellow) and gross error (red).

A negative example obtained with the SBF method is shown in Figure 4.19(a), where several outlier contour points were detected. It can also be argued that for irregular and more deformed stone circles the DP method may be too rigid to accommodate the variations (see Figure 4.19(b)) yet, it still corresponds to the most suitable method for the delineation task.



(a) SBF detected several outlier contour points.

(b) DP computed a rigid contour for a deformed circle.

Figure 4.19: Negative examples of the circle delineation obtained with SBF (left) and DP (right) methods.

Chapter 5

Conclusions

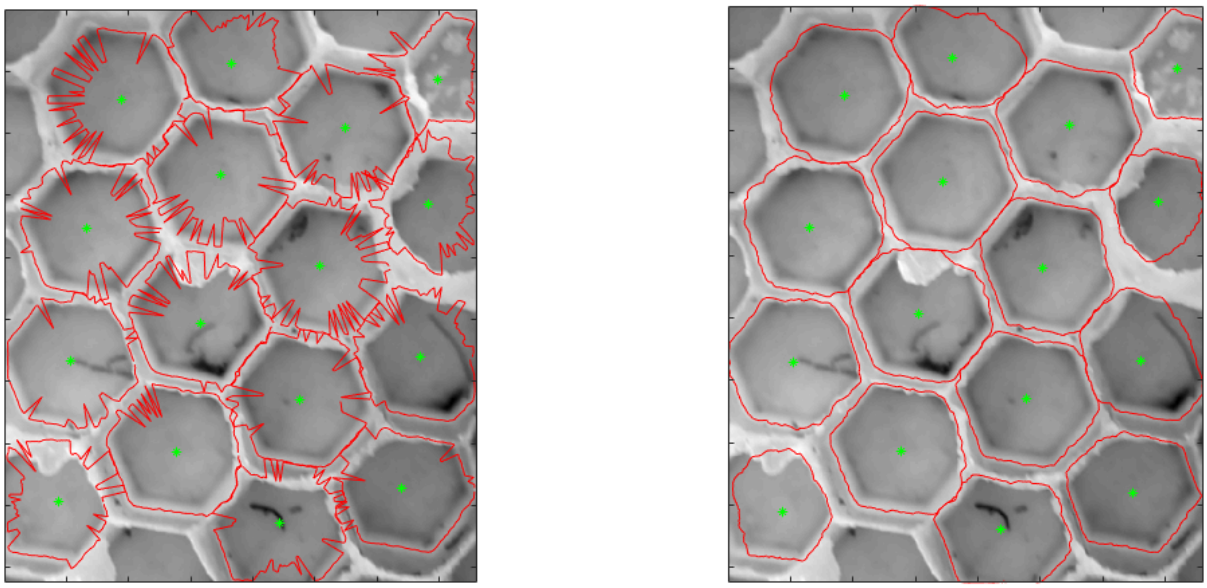
The work described in this thesis focuses on the unexplored subject of automated detection and delineation of stone circles in periglacial terrains, by analysis of Digital Elevation Models with centimetric resolution obtained after processing of image data captured by a UAV in Antarctica. The shape of these unique structures resembles a half-torus yet, their extremely regular geometry also exhibits some natural deformations.

The algorithms developed are based on Template Matching, the Sliding Band Filter and Dynamic Programming and explore different image properties, together with the stone's shape, to produce their location in the image. The last two methods also estimate the contour for each detected stone.

The detection methods were tested in 20 elevation images (1000×1000 pixels) where the variability of the structures is evident. In order to enhance the stones contained in the input image, a pre-processing stage was implemented to remove the brightness difference caused by a natural slope of the terrain. The DP approach achieved the best compromise between performance and complexity, reaching an average F-score of 84.3%, outperforming SBF (81.3%) and TM by a significant amount (66.7%). The rigid assumptions taken by TM limit the method's performance while SBF allows some flexibility and accommodates deformations and scale changes. The SBF corresponds to a particular case of the DP (with cost parameters $m = 0$ and $\beta = \infty$) yet, the latter's superior performance is due to an additional smoothness constraint that improves the gradient noise robustness. This constraint also led to DP's top performance in the contour delineation task: for the 24 circles evaluated, it reached an average percentage of gross errors of 7.1%, a significant reduction in comparison with the 18.3% obtained by the SBF.

Regarding future research, there is room for improvement. The flexibility presented by the Dynamic Programming based technique and its robustness to the gradient noise may be combined with additional methods that exploit other features of the stone circles images. The application of deformable models for delineation can also be addressed, where the model initialization is given by the output of the detection algorithms. The proposed methods can also be used in the segmentation of other regular polygons with at least five sides and a similar gradient radial structure. Figure 5.1 shows an example of the output of the SBF and DP methods for the inspection of beehive cell images. Both methods locate accurately the

center of the hexagon-like shapes and the smoothing criteria of DP improves the delineation task.



(a) Detection and delineation with SBF.

(b) Detection and delineation with DP.

Figure 5.1: Example of the application of the SBF and DP methods for detection of hexagon-like shapes in bee hive cell images.

Bibliography

[1] Uxa, T., Mida, P., and Křížek, M. Effect of climate on morphology and development of sorted circles and polygons. *Permafrost and Periglacial Processes*, 28(4):663–674, 2017.

[2] Hallet, B. Stone circles: form and soil kinematics. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371(2004):20120357, 2013.

[3] Pina, P., Heleno, S., Vieira, G., Mora, C., Miranda, V., and Hong, S. G. Extensive mapping of sorted stone circles with ultra-high resolution imagery: Preliminary results from the 2018 field campaign in Barton peninsula, King George island. In *Abstract Book of the 10th Portuguese Conference on Polar Sciences, University of Aveiro*, 10 2018.

[4] Kääb, A., Girod, L. M. R., and Berthling, I. T. Surface kinematics of periglacial sorted circles using structure-from-motion technology. *The Cryosphere*, 8:1041–1056, 2014.

[5] Pina, P., Pereira, F., Marques, J. S., and Heleno, S. Detection of stone circles in periglacial regions of Antarctica in UAV datasets. *Iberian Conference on Pattern Recognition and Image Analysis*, Lecture Notes in Computer Science, Vol. 11687, 279-288. Springer, 2019.

[6] Quelhas, P., Marcuzzo, M., Mendonça, A. M., and Campilho, A. Cell nuclei and cytoplasm joint segmentation using the sliding band filter. *IEEE Transactions on Medical Imaging*, 29(8):1463–1473, 2010.

[7] Marques, J. S. and Pina, P. Crater delineation by dynamic programming. *IEEE Geoscience and Remote Sensing Letters*, 12(7):1581–1585, 2015.

[8] Garrido, A. and De La Blanca, N. P. Applying deformable templates for cell image segmentation. *Pattern Recognition*, 33(5):821–832, 2000.

[9] Hansen, D. W. and Ji, Q. In the eye of the beholder: A survey of models for eyes and gaze. *IEEE transactions on pattern analysis and machine intelligence*, 32(3):478–500, 2009.

[10] Marques, J. S. and Pina, P. An algorithm for the delineation of craters in very high resolution images of mars surface. In *Iberian Conference on Pattern Recognition and Image Analysis*, Lecture Notes in Computer Science, Vol. 7887, 213-220. Springer, 2013.

[11] Bowyer, K. W., Hollingsworth, K., and Flynn, P. J. Image understanding for iris biometrics: A survey. *Computer vision and image understanding*, 110(2):281–307, 2008.

[12] Ballard, D. H. Generalizing the hough transform to detect arbitrary shapes. *Pattern recognition*, 13 (2):111–122, 1981.

[13] Rad, A. A., Faez, K., and Qaragozlou, N. Fast circle detection using gradient pair vectors. In *Dicta*, volume 2003, pages 879–887, 2003.

[14] Silveira, M. An algorithm for the detection of multiple concentric circles. *Iberian Conference on Pattern Recognition and Image Analysis*, Lecture Notes in Computer Science, 271-278. Springer, 2005.

[15] Perveen, N., Kumar, D., and Bhardwaj, I. An overview on template matching methodologies and its applications. *International Journal of Research in Computer and Communication Technology*, 2 (10):988–995, 2013.

[16] Bandeira, L., Saraiva, J., and Pina, P. Impact crater recognition on mars based on a probability volume created by template matching. *IEEE Transactions on Geoscience and Remote Sensing*, 45 (12):4008–4015, 2007.

[17] Kim, J. R., Muller, J.-P., van Gasselt, S., Morley, J. G., and Neukum, G. Automated crater detection, a new tool for mars cartography and chronology. *Photogrammetric Engineering & Remote Sensing*, 71(10):1205–1217, 2005.

[18] Lamiroy, B., Fritz, L., and Gaucher, O. Robust circle detection. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, volume 1, pages 526–530. IEEE, 2007.

[19] Stache, N. C. and Zimmer, H. Robust circle fitting in industrial vision for process control of laser welding. In *Proceedings of the 11th International Student Conference on Electrical Engineering, Prague MAY*, 2007.

[20] Cantzler, H. Random sample consensus (ransac). *Institute for Perception, Action and Behaviour, Division of Informatics, University of Edinburgh*, 1981.

[21] Yuille, A. L., Hallinan, P. W., and Cohen, D. S. Feature extraction from faces using deformable templates. *International journal of computer vision*, 8(2):99–111, 1992.

[22] Daugman, J. The importance of being random: statistical principles of iris recognition. *Pattern recognition*, 36(2):279–291, 2003.

[23] Martins, R., Pina, P., Marques, J. S., and Silveira, M. Crater detection by a boosting approach. *IEEE Geoscience and Remote Sensing Letters*, 6(1):127–131, 2008.

[24] Dietterich, T. G. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

[25] Falk, T., Mai, D., Bensch, R., Çiçek, Ö., Abdulkadir, A., Marrakchi, Y., Böhm, A., Deubner, J., Jäckel, Z., Seiwald, K., et al. U-net: deep learning for cell counting, detection, and morphometry. *Nature methods*, 16(1):67, 2019.

[26] Maggiori, E., Tarabalka, Y., Charpiat, G., and Alliez, P. Convolutional neural networks for large-scale remote-sensing image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55 (2):645–657, 2016.

[27] Filev, P., Hadjiiski, L., Sahiner, B., Chan, H.-P., and Helvie, M. A. Comparison of similarity measures for the task of template matching of masses on serial mammograms. *Medical physics*, 32(2):515–529, 2005.

[28] Dreyfus, S. Richard bellman on the birth of dynamic programming. *Operations Research*, 50(1): 48–51, 2002.

[29] Bellman, R. E. *Eye of the Hurricane: an autobiography*. World Scientific Singapore, 1984.

[30] Bellman, R. E. and Roth, R. S. *The Bellman Continuum: A Collection of the Works of Richard E. Bellman*. World Scientific, 1986.

[31] Bellman, R. E. 1957. dynamic programming. *Princeton UniversityPress. BellmanDynamic programming1957*, page 151, 1957.

[32] Bellman, R. et al. The theory of dynamic programming. *Bulletin of the American Mathematical Society*, 60(6):503–515, 1954.

[33] Santiago, C., Nascimento, J. C., and Marques, J. S. Fast segmentation of the left ventricle in cardiac mri using dynamic programming. *Computer Methods and Programs in Biomedicine*, 154: 9–23, 2018.

[34] Liu, G. and Haralick, R. M. Optimal matching problem in detection and recognition performance evaluation. *Pattern Recognition*, 35(10):2125–2139, 2002.

[35] Pina, P. and Marques, J. S. Delineation of impact craters by a mathematical morphology based approach. In *International Conference Image Analysis and Recognition*, Lecture Notes in Computer Science, Vol. 7969, 717-725. Springer, 2013.

[36] Dollar, P., Wojek, C., Schiele, B., and Perona, P. Pedestrian detection: A benchmark. *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 0:304–311, 06 2009. doi: 10.1109/CVPR.2009.5206631.

[37] Nascimento, J. C. and Marques, J. S. Performance evaluation of object detection algorithms for video surveillance. *IEEE Transactions on Multimedia*, 8(4):761–774, 2006.

Appendix A

Matching method

The task of establishing an unequivocal correspondence between the elements of a ground-truth set and the output of an automatic detection method is commonly performed to assess the performance of object and pedestrian detection systems [36] [37]. This section describes the matching method used in this work. It is similar to a modified version of the Hungarian algorithm proposed in [34], which pairs elements from two sets in order to minimize a cost function. The two sets of elements may have different cardinalities and there are considered restrictions in the associations.

Since the matching criteria depends on the localization error, the first step consists in building a matrix of distances from all the ground-truth centers (GT) to all the detected points [5]. The distance that separates the closest ground-truth and detected elements is evaluated: if the localization error is smaller than the maximum allowed distance λ, the elements form a valid pair and are considered a TP. The distance matrix is updated by removing the corresponding line and column of the elements paired as there can not be multiple associations for the same GT or detection. This process is repeated until there are no distance matrix entries smaller than λ. The ground-truth points and detections that were not matched correspond to the FN and FP, respectively.