December 2017

# Threshold Free Detection of Elliptical Landmarks Using Machine Learning

Lifan Zhang
*University of Wisconsin-Milwaukee*

# THRESHOLD FREE DETECTION OF ELLIPTICAL LANDMARKS USING MACHINE LEARNING

by

Lifan Zhang

A Thesis Submitted in

Partial Fulfillment of the

Requirements for the Degree of

Master of Science

in Engineering

at

The University of Wisconsin-Milwaukee

December 2017

# ABSTRACT

THRESHOLD FREE DETECTION OF ELLIPTICAL LANDMARKS USING MACHINE LEARNING

by

Lifan Zhang

The University of Wisconsin-Milwaukee, 2017
Under the Supervision of Professor Brian Armstrong

Elliptical shape detection is widely used in practical applications. Nearly all classical ellipse detection algorithms require some form of threshold, which can be a major cause of detection failure, especially in the challenging case of Moire Phase Tracking (MPT) target images. To meet the challenge, a threshold free detection algorithm for elliptical landmarks is proposed in this thesis. The proposed Aligned Gradient and Unaligned Gradient (AGUG) algorithm is a Support Vector Machine (SVM)-based classification algorithm, original features are extracted from the gradient information corresponding to the sampled pixels. with proper selection of features, the proposed algorithm has a high accuracy and a stronger robustness in blurring and contrast variation. The thesis confirms that the removal of thresholds in ellipse detection algorithm improves robustness.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# ACKNOWLEDGMENTS

I would like to thank everybody who supported and helped and me over the past year. Especially I would like to thank my adviser, Professor Brian Armstrong, for providing me with a vast amount of knowledge and instructions during my research. Mark Halstrom and Zijian Cao have been incredibly helpful by providing assitance on the Linux operating system. Meiling He: for maintaining an encouraging learning environment. My parents: for everything. And my family and friends for emotional support through the year.

# 1 Introduction

As a computer-based method to perform operations on images, shape detection is playing an increasingly important role in many aspects of our daily life, as well as in a wide variety of science fields, especially in computer vision and computer graphics. Examples of application are: in medical/biological: image-guided surgery, interpretation of X-ray images; in robotics: calibration, object tracking and mobile robot navigation; in industrial applications: zip code, 2-D bar code recognition [1].

Figure 1: QR Code

In many of those cases, a landmark detection algorithm is essential to locate landmarks in the image. Examples of landmarks used for metrology can be seen in figures 1-2. Figure 1 shows a QR Code used in tracking and marketing. Figure 2 is a tracking target used in vehicle alignment system.

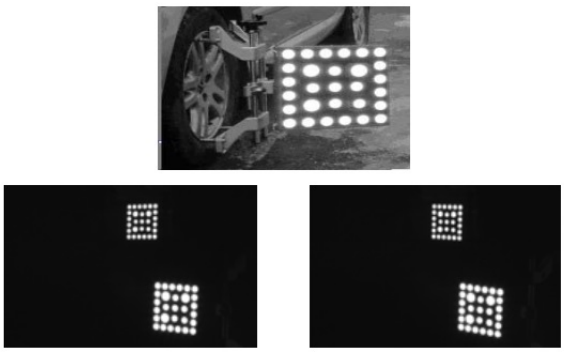


Figure 2: Landmark target in vehicle alignment system

## 1.1 Moire Phase Tracking (MPT) Target

Particularly, in this thesis, images of Moire Phase Tracking (MPT) targets are used as study objects. Moiré Phase Tracking (MPT) is a 3D motion tracking technology developed by Brian Armstrong and others [2]. MPT is used to perform 3D motion tracking of a person in motion by fixing multiple targets to a model who performs different tasks, like jumping, walking, running, etc. MPT uses a single camera as a sensor and tracks specially designed targets as shown in figure 3. The target relies on the detection of the starburst landmark in the center of the target, the location of the four circular landmarks, and the resolution of the periodic moiré patterns [3].

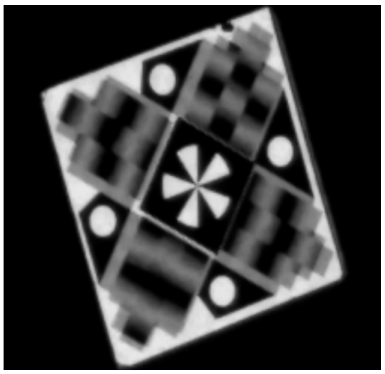


Figure 3: MTP target

## 1.2 Factors that Influence Landmarks' Condition

However, in real cases, circular landmarks always deform into elliptical shapes because of the tilt with respect to camera. Seven factors could change the condition of an elliptical landmark. They are: spatial location (x,y,z position), eccentricity, orientation, brightness, contrast, focus and noise. The first three can be reserved as internal factor, which are determined by landmarks themselves. The rest can be decided by external factors, like lighting condition, exposure and focus. This thesis mainly focuses on detecting elliptical landmarks influenced by external factors. In chapter 4, impacts of blurring and contrast on landmark detection are studied in detail.

## 1.3 Elliptical Landmark Detection

An elliptical landmark detection (ELD) algorithm is the process of taking image data and determining whether there is an elliptical landmark in the searched region. ELD is used to provide preknowledge for Precision landmark location (PLL) in MPT.

### 1.3.1 History of Ellipse Detection

Ellipses and elliptical shapes are evident in digital images, they carry useful statistical and geometrical information that can be used in a wide range of practical applications. In monitoring and surveillance field, ellipse detection plays a crucial role in eye tracking, lips reading, and face detection [4]; In biological industry, biological cells are segmented by ellipse fitting methods, multiple ellipse detection is used in 3D objects shape reconstruction [5]; In engineering, strain and curvature analysis, and sun spot center orientation relies on ellipse detection [6].

In the past three decades, ellipse detection methods have gained maturity [7]. Most of them rely on feature extraction, which is a field in image processing. Simple features, like edges, blobs, corners, lines, curvatures and ridges can be detected by specific operators [7]. To detect extracting shapes like rectangles and circles, more complex algorithms are needed.

Principle ellipse detection algorithms rely on unique mathematical properties of an ellipse, they can be sorted into three main categories.

**Least square method** The first category is based on traditional least-square fitting method. Typical examples are Direct Least-Square (DLS) proposed by Fitzgibbon [8], Enhance LS method proposed by Maini [9]. LS methods are suitable candidates for real-time applications, because of their linearity. Meanwhile, LS methods are extremely sensitive to outliner noise, and perform poorly on deformed or degraded images [10].

**Hough-Transform-based Method** The second type derives from Hough transform method. Hough transform is a feature extraction technique widely used in image processing and computer vision. It's first introduced by P.V.C Hough in 1962, and gained popularity in the past forty years because HT is relatively insensitive to image noise and tolerant of gaps in feature boundary descriptions. The purpose of this technique is to find certain object by a voting procedure carried out in a parameter space. Early Hough transform methods were concerned with detection of lines in the images [11], researchers later proposed Circle Hough Transform (CHT) [12] for circle and ellipse detection, and further generalized HT to objects with arbitrary shape detection [13]. New Hough Transform modifications are created every few years, for diverse applications or to remove different constraints [14]. The main disadvantages of HT based algorithms are large computational cost and thresholds.

**Statistical or combined techniques** The rest are reserved for statistical, heuristic, or combined techniques. RANdom Sample Consensus (RANSAC) [15] and Genetic Algorithm [16] are two typical statistical methods applied on ellipse detection problem. Combined methods are introduced in [17, 18, 19]. There are also original algorithms for ellipse detection on vehicle alignment planer target [20, 21]. Algorithms in this category are based on different techniques, and thus have diverse advantages and disadvantages, and are not addressed further in this thesis.

As computer-based technology is permeating in our daily life, the development of ellipse detection algorithm has hit a bottleneck of robustness and computational efficiency. For example, it is no longer practical to apply the same scheme suitable for single images, to elliptical faces detection in a sequence of video frames [22]. And as for some challenging situations, where sample images are exceptionally small, polluted, and poorly focused, traditional ellipse detection algorithms perform often poorly [7].

To achieve different requirements posed by diverse application goals, innovative solutions

are proposed from time to time. In general, all of those solutions are to reach a balance between robustness and computational efficiency.

### 1.3.2 Investigations into Threshold

A threshold is a level of a continuously varying quantity at which a binary decision is made. For example, an image intensity (brightness) is compared with a threshold to determine if a pixel is to be labeled black or white.

One classic intensity-threshold-based ellipse detection algorithm is region growing method, which is also known as a pixel-based image segmentation method. This approach starts from selecting initial seed points, then examines neighbor pixel of those initial seed points, and adds the neighbor pixel to the region if its intensity $I_{i,j} \leq T$, Where $T$ represents intensity threshold.

Intensity threshold could be either constant or adaptive. Constant intensity threshold is normally set as the mean intensity of a searching area. Adaptive intensity threshold can be calculated by automatic threshold selection algorithm, like Otsu's method, Histogram modeling by Gaussian distributions, or intelligent algorithm like Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) [23].

Other types of thresholds are used in traditional ellipse detection algorithms, such as the case of HT-based algorithms, where the threshold used is vote threshold. In HT, object candidates with votes larger than threshold V are labeled as object detected.

The inclusion of thresholds reduces the robustness of ellipse detection algorithms. For example, if illumination is non-uniform, the intensity threshold appropriate for one image region may lead to failed detection in another. Such cases are examined in detail in the following section.

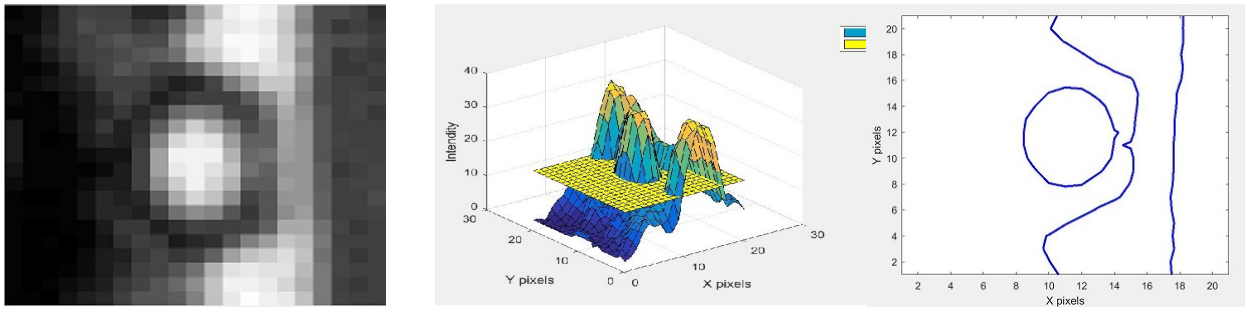
## 1.4 Threshold Failure

Defined as detection failure caused by threshold, threshold failure is discussed in this section. Sections 1.4.1 and 1.4.2 discuss the cause, mechanism and impact of threshold failure with region growing algorithm referring to intensity threshold failure and Circle Hough Transform (CHT) representing vote threshold failure respectively.

### 1.4.1 Intensity Threshold Failure

We use a region growing algorithm as an example to discuss intensity threshold failure. Region growing algorithm starts from a seed pixel inside of the ellipse, filling pixels around it outwards until the edge, where pixel has a larger intensity than threshold [24].

Figure 4 illustrates one detection, where threshold is set as 16. Figure 4 show scanning region from a real MPT target image, the corresponding intensity and threshold mesh, and the intersection line of two meshes respectively. The clear elliptical shape in figure 4-c indicates the success of region growing algorithm.

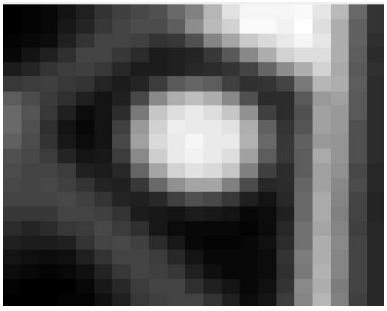



a): elliptical landmark b): intensity and threshold meshes c): intersection line of the two meshes

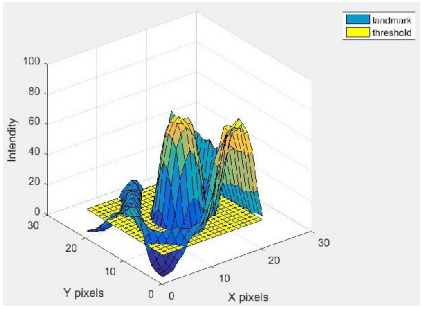
Figure 4: Region growing algorithm detection case 1 (threshold=16)

When region growing algorithm is applied to another case shown in figure 5, there is no clear elliptical region in figure 5-c, indicating the algorithm fails in this case.

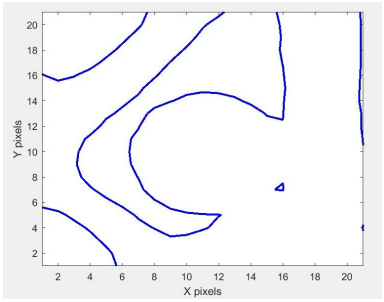
Figure 5 shows an example of threshold failure in MPT image.



a): elliptical landmark




b): intensity and threshold meshes




c): intersection line of the two meshes

Figure 5: Region growing algorithm detection case 2 (threshold=16)

### 1.4.2 Vote Threshold Failure

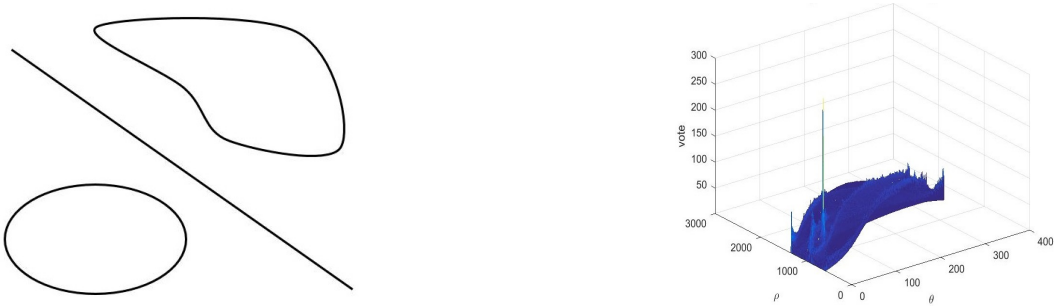
In HT-based algorithms, threshold failure happens in vote space. For an easy and brief illustration, the Standard Hough Transform (SHT) is used as an example of HT-based algorithms to illustrate the concept of vote threshold. In practice, SHT is used to detect straight lines. A straight line can be represented as Hesse normal form [25]:

$$r = xcos\theta + ysin\theta \tag{1}$$

where $r$ is the distance from the origin to the closest point on the straight line, $\theta$ is the angle between $x$ axis and the line connecting the origin with that closest point. Obviously, each pair of $(r, \theta)$ associates with each line. The $(r, \theta)$ plane is referred to as Hough space. SHT uses an accumulator, which is a two-dimensional array, to record votes for each $(r, \theta)$. Pairs of $(r, \theta)$ with votes higher than a threshold $V$ are selected as parameters corresponding to detected lines.

Figure 6 displays one SHT detection. SHT is used to detect the line of some arbitrary shapes shown in figure 6-a. Figure 6-b shows the corresponding vote space, where the peak refers to the line detected. The algorithm will succeed if the votes threshold lies between 98 and 297, but will fail otherwise. We can see from figure 6-b that if the vote threshold is set less than 98, then more than one line will be detected, while if the vote threshold is set

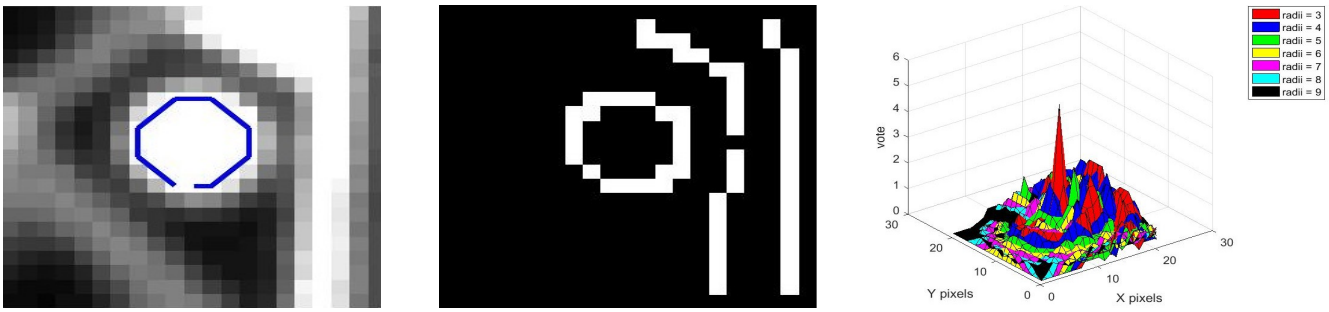
greater than 297, then no line can be detected.




a): arbitrary shapes for detection b): vote space

Figure 6: SHT detection

Applying the same scheme, ellipse detection is conducted on real MPT target images by Circle Hough Transform (CHT). Figures 7 and 8 show two detections in a real MPT target image.

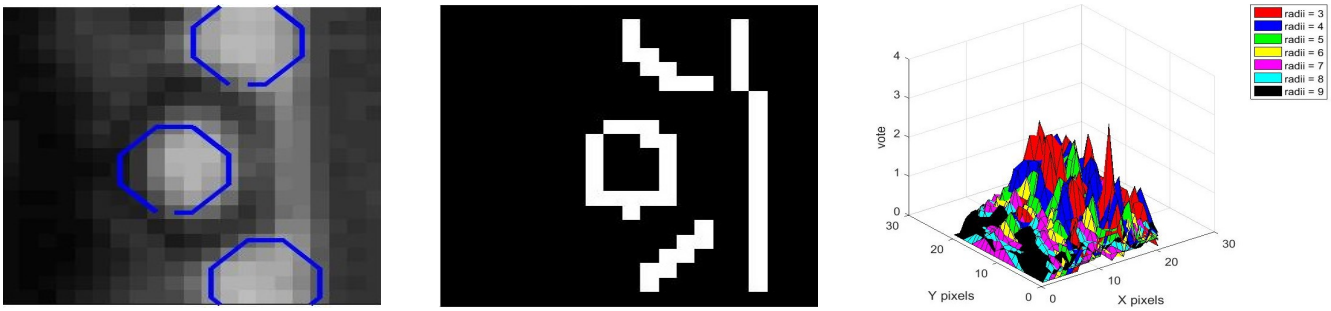



a): elliptical landmark & detected circle b): edge image c): vote space

Figure 7: CHT detection case 1

Figures 7-a and 7-b show the elliptical landmark and edge image respectively, figure 7-c shows the corresponding vote space, in the plot, meshes with different colors represent results with different radius. Blue circle in figure 7-a indicates the success of this case, where threshold $V$ is set as $0.9 * max(vote)$.

Then CHT with the same threshold $V$ is conducted on a different patch shown in figure 8-a. Vote threshold failure can be observed from the three blue circles in figure 8-a.




a): elliptical landmark & detected circles

b): edge image

c): vote space

Figure 8: CHT detection case 2

More generally speaking, MPT target images pose challenges to any edge-information-based and threshold-required ellipse detection algorithms, because of the following features:

- Elliptical landmarks in MPT Target are small (around 10 to 12 pixels in diameter), poorly focused, and occasionally merge with neighboring elements, such that, qualified edge information is only partially detected;
- Uneven illumination and diverse contrast cause threshold failure, as well as reduce algorithm robustness;

And removing threshold from the detection procedure is suppose to be an effective method to meet with those challenges.

## 1.5 Purpose of This Thesis

To eradicate failures caused by threshold in traditional ellipse detection algorithms, a threshold free elliptical shape detection algorithm is proposed in this thesis. Two main goals are:

1. To establish a threshold free algorithm for ellipse detection, with special application on MPT target images.

2. To obtain a good overall performance, that is high sensitivity and specificity, as well as strong robustness against blurring and low contrast.

In this thesis, one classical ellipse detection algorithm Randomized Hough Transform (RHT) is applied on the same set of test images as comparison to the proposed algorithm. All experiments are conducted on real digital MPT images.

In addition, this thesis not only finds a robust new method for ellipse detection, but also convies the exploitation of elliptical landmarks' properties, and provides inspiration for creative thinking about ellipse detection.

# 2 Randomized Hough Transform (RHT)

In order to emphasize the advantage of the proposed AGUG algorithm, a classical ellipse detection algorithm, the Randomized Hough Transform (RHT), is implemented as comparison [26]. In this chapter, basic concepts and principles of RHT are introduced in section 2.1, section 2.2 describes the method for generation of the testing pool, section 2.3 shows the testing results, and section 2.4 discusses the influence of the thresholds in RHT.

## 2.1 Basics of the Randomized Hough Transform

Five parameters are required to define an ellipse in the Hough Transform (HT) algorithm. Due to the computational complexity of HT, Randomized Hough Transform (RHT) is used in this thesis. The traditional approach for ellipse detection using the Hough technique is similar to line or circle detection [27]. The parametric equation of an ellipse is written as follows:

$$ax^2 + 2bxy + cy^2 + 2dx + 2ey + f = 0 \tag{2}$$

Using the Chellali, Fremont and Czervinski algorithm [28], to detect all five parameters of an ellipse, only three points are needed, among which two are considered to be the ellipses vertices. The basics of this theorem is described below. Given the two vertices of ellipse denoted as $(x_1, y_1)$, $(x_2, y_2)$, four of the five ellipse parameters can be determined by the following formulas:

$$x_0 = \frac{x_1 + x_2}{2} \tag{3}$$

$$y_0 = \frac{y_1 + y_2}{2} \tag{4}$$

$$a = \frac{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}{2} \tag{5}$$

$$\alpha = \tan(\frac{y_2 - y_1}{x_2 - x_1}) \tag{6}$$

where $(x_0, y_0)$ is the center of the assumed ellipse, term $a$ indicates the half-length of the major axis, parameter $\alpha$ stands for the orientation of the ellipse. In order to determine the fifth parameter in the ellipse, a third point on the ellipse is needed, which is denoted as $(x, y)$.

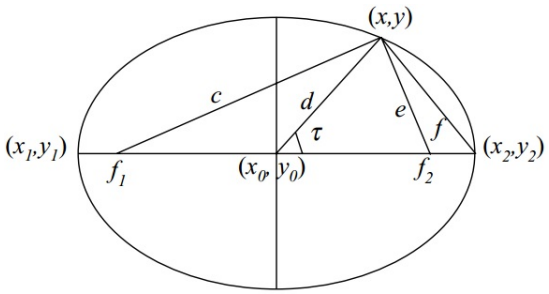



Figure 9: Illustration of ellipse

Figure 9 shows the geometry of an ellipse. $f_1$, $f_2$ are focus of the ellipse. Term $b$ is the half length of the minor axis. Terms $d$ and $\tau$ can be observed in figure 9. Term $b$ and $\cos\tau$ can be calculated by the following approximate formulas:

$$b = \sqrt{\frac{a^2 d^2 \sin\tau^2}{a^2 - d^2 \cos\tau^2}} \tag{7}$$

$$\cos\tau = \frac{a^2 + d^2 - f^2}{2ad} \tag{8}$$

Using equation (2)-(8), all ellipses in the image can be detected. This process has a complexity of $O(n^3)$.

Steps of RHT are as follows [29]:

- Apply Canny edge detector [30] to get binary edge image for each sample;

- For each edge-pixel pair $(x_1, y_1)$, $(x_2, y_2)$, apply equations (2)-(6) to calculate the center $(x_0, y_0)$, orientation $\alpha$ and $a$ for the assumed ellipse;
- For each third pixel $(x, y)$ on the edge, apply equation (7) to calculate $b$ ;
- Increment votes for the assumed ellipse by one;
- Loop until all pairs of pixels on edge are computed;
- As results, output all ellipses with votes higher than the threshold.

Further, by adding a random edge pixels selecting process, the complexity can be reduced from $O(n^3)$ to $O\left(\frac{n^2}{C}\right)$, where $C$ is a manually set constant. Only $m$ pairs of randomly picked points are selected. That is, instead of all points detected by the edge detector, only $m = n * C$ are selected, where $C = 1, 2, 3 \ldots$ .

## 2.2 Testing Pool Generation

Before testing RHT on real digital images, the method for generating the testing pool is introduced. Figure 10 shows a digital image of a model equipped with MPT targets. 15 * 15 pixel patches are selected from the image as samples. Samples are either true positive samples, containing elliptical landmarks, or true negative samples, indicating no elliptical landmark can be found. In the following experiments, 300 true positive samples and 300 true negative samples are selected from figure 10 as testing pool.

## 2.3 RHT on MPT Elliptical Landmark Detection

RHT is then applied to the testing pool generated in section 2.2, results are observed and processed in Matlab. Note that, the correctness of each detection result is determined by a human eye observation.

Figure 11 gives a glimpse of RHT detection results, where detected results are indicated by blue ellipses with red crosses as center. Three types of detection results can be seen

Figure 10: Source image for testing pool

Figure 11: A glimpse of RHT detection results on real MPT target images

from figure 10, they are correct detection, missing out (the algorithm misses a true positive sample) and wrong detection (the algorithm indicates a false positive ellipse). In the testing, the vote threshold is set as 0.12.

Accuracy, sensitivity and specificity are three standards used to analyze the results of detection algorithms. Term explanations are as follows:

$$\text{sensitivity} = \frac{\#\text{correct detection of true positive samples}}{\#\text{true positive samples}}$$

$$\text{specificity} = \frac{\#\text{correct detection of true negative samples}}{\#\text{true negative samples}}$$

$$\text{accuracy} = \frac{\#\text{correct detection of all samples}}{\#\text{all samples}}$$

where "#" stands for "the number of".

Detection result of RHT on MPT testing pool is listed in table 1:

| | accuracy | sensitivity | specificity |
|---|---|---|---|
| RHT (threshold:0.12) | 74.50% | 84.00% | 65.00% |

Table 1: Results of RHT on original MPT target image set

## 2.4 Influence of Thresholds

This section addresses the influence of thresholds on the Randomized Hough Transform (RHT). Note that, besides vote threshold in RHT, Canny edge detector brings in two intensity thresholds. The impacts of both of the two thresholds are discussed in section 2.4.1 and 2.4.2 respectively.

### 2.4.1 Impact of Threshold in Canny Detector

The Canny method applies two thresholds: a high threshold for low edge sensitivity and a low threshold for high edge sensitivity. Edge starts with the low sensitivity result and then grows it to include connected edge pixels from the high sensitivity result, filling in gaps in the detected edges. In the test in section 2.3, the thresholds in the Canny detector are automatically set as $[0.2, 0.52]$. To study the impact of the thresholds, the experiment in section 2.3 with different Canny detector thresholds is conducted another 10 times. We fix the high threshold as 0.52, increase the low threshold from 0.08 to 0.28 at an interval of 0.02. Results are shown in figure 12.

We can see from figure 12 that, RHT reaches the best performance when the low threshold equals 0.2, which is selected automatically by Matlab. Both of the sensitivity and specificity drop as threshold moves away from 0.2.

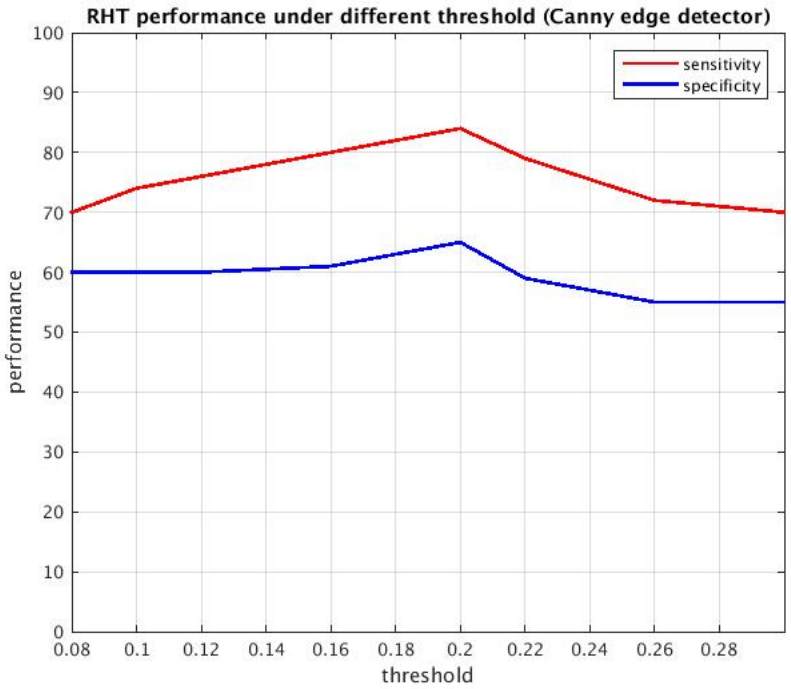



Figure 12: RHT performance with different Canny detector threshold

### 2.4.2 Impact of Vote Threshold

We conduct the same experiment for another 13 times, each with a different vote threshold. The threshold is increasing from 0.09 to 0.21 with the interval of 0.01 through all 13 experiments.

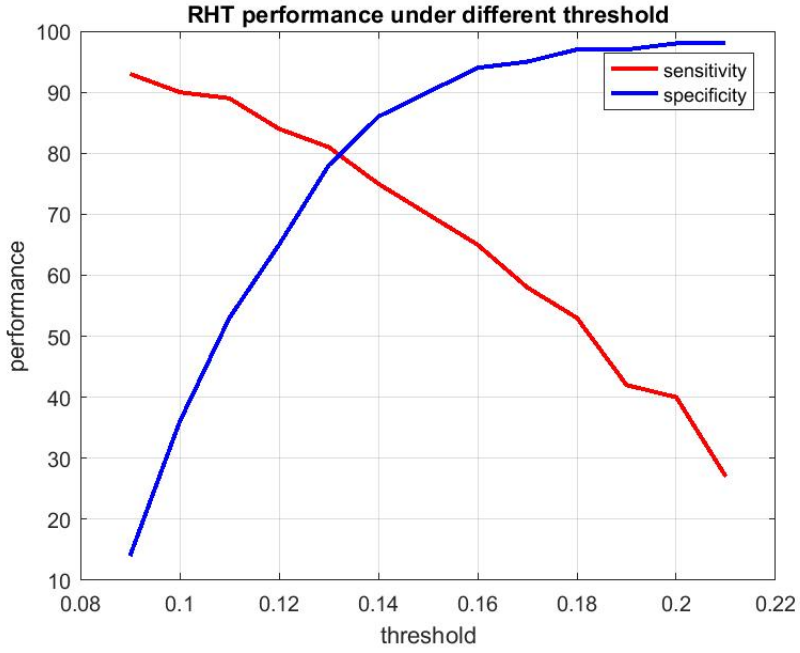



Figure 13: RHT detection results with different vote thresholds

As we can see from figure 13, the sensitivity reaches the maximum of 92.5% when the threshold gets 0.09, then drops to less than 30% as the threshold increases to 0.21, presenting the monotone decreasing tendency. Meanwhile, the specificity increases from 11% to 98%. The sensitivity and specificity intersects around a threshold of 0.13. A trade off between the sensitivity and the specificity can be observed.

## 2.5 Summary

In this chapter, a classical ellipse detection algorithm is introduced and applied on real digital images. The impact of both the threshold in Canny detector and RHT are studied with experiments. Experiment results confirm that threshold failures can occur, as well as address the influence of thresholds in RHT.

# 3 AGUG: A Threshold Free Elliptical Landmark Detection Algorithm

A threshold free elliptical landmark detection algorithm (AGUG) is proposed in this chapter. The proposed method is a Support Vector Machine (SVM)-based classification algorithm [31], which determines an input pixel to be inside elliptical landmarks or not. Aligned and unaligned gradient values of eight-way spokes corresponding to sample pixels are collected to generate features. Section 3.1 describes the technology used, section 3.2-3.4 illustrate the methodology of the proposed algorithm in detail. Results of the proposed algorithm on real MPT target images are discussed in chapter 4.

## 3.1 Support Vector Machine (SVM)

To illustrate the proposed algorithm (AGUG), a basic introduction to the SVM is necessary. Basics of classification procedure is introduced first. In a classification task, samples are separated into two non-overlapping sets, namely the training set and the testing set. Each sample in training set consists of a "label" and some "features". Samples in testing sets only contain "features". "Label" is an integer indicating the category to which the sample belongs. "Features" are vectors of numerical values that represent the sample.

Given a training set, a SVM training algorithm builds a classifier by a supervised learning algorithm. SVMs map inputs from original space into high-dimensional feature spaces, where samples are separated into different categories by a decision boundary [32].

Given a training set of instance-label pairs$(X_i, y_i), i = 1, \cdots, n_p$ where $X_i \in R^{n_d}$ and $y \in \{1, -1\}^{n_p}$, terms $n_d$ and $n_p$ indicate the number of dimensions and the number of samples respectively.

Training a SVM involves the maximization of the error function:

$$\min_{w,b,\xi} \frac{1}{2} W^T W + C \sum_{i=1}^{n} \xi_i \tag{9}$$

subject to the constraints:

$$y_i \left( W^T \phi \left( x_i \right) + b \right) \geq 1 - \xi_i, \xi_i \geq 0 \tag{10}$$

where $C$ is the capacity constant, $W$ is the vector of coefficients, $b$ is a constant, representing parameters for handling non-separable data in inputs. The index $i$ labels the $n_p$ training cases. Function $\phi$ is the kernel function mapping input data to the feature space. By using kernel function, SVMs can perform both linear and non-linear classification. The RBF (Radial Basis Function) kernel is used in this thesis:

$$K \left( x_i, x_j \right) = \exp \left( -\gamma \left\| x_i - x_j \right\|^2 \right), \gamma > 0 \tag{11}$$

where $\gamma$ is the kernel parameter.

## 3.2 Target-Pixel-Based Spoke Information Gathering

The proposed AGUG method is a three step procedure. The first step is to gather gradient information corresponding to sample pixels by a spoke-grabbing strategy.

The scheme of gradient information gathering is shown in figure 14. In the figure, an eight-spoke operator is reaching out from the target pixel in the center, collecting corresponding aligned gradient, unaligned gradient and intensity of each pixel on each spoke. The raw data is then stored in three matrices, with matrix $AG$ for aligned gradient, matrix $UG$ for unaligned gradient, matrix $I$ for intensity. Each of the matrices are of size $m * 8$, where m indicates the length of spoke, 8 indicates the number of spokes.

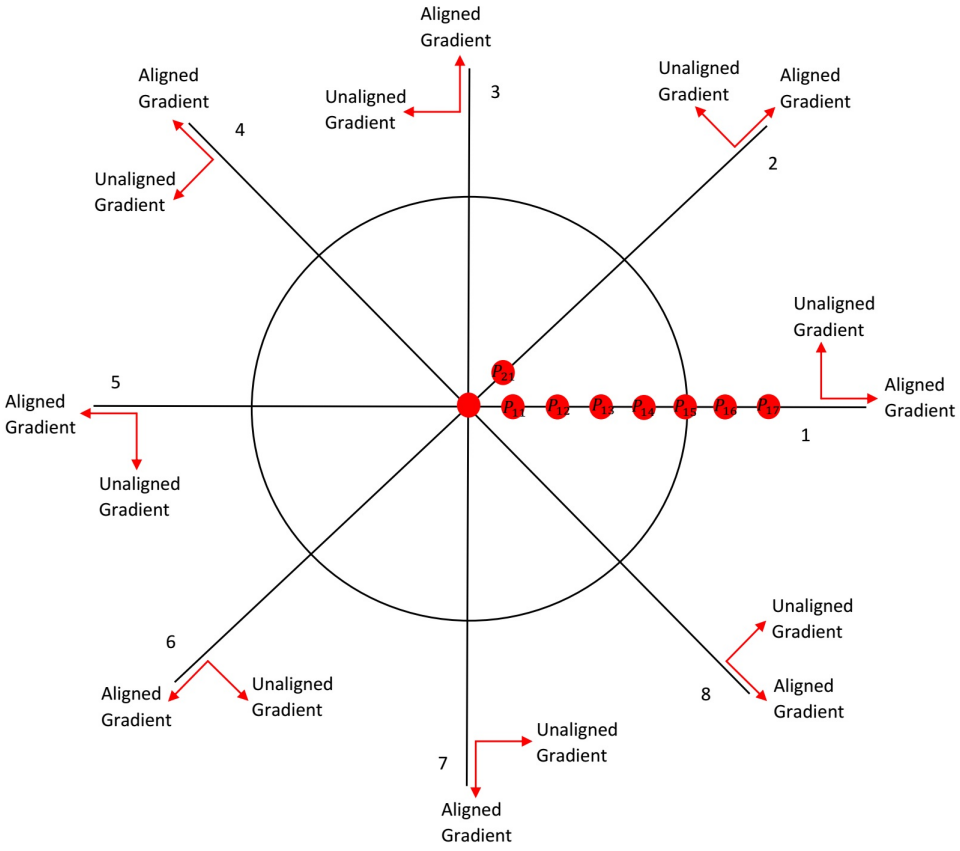



Figure 14: The scheme of target-pixel-based spoke information gathering

As for target pixel $p$, the corresponding $AG$ can be described in the following form:

$$AG = \begin{bmatrix} ag_{1,1} & \cdots & ag_{1,j} & \cdots & ag_{1,8} \\ \cdots & & & & \cdots \\ ag_{i,1} & \cdots & ag_{i,j} & \cdots & ag_{i,8} \\ \cdots & & & & \cdots \\ ag_{7,1} & \cdots & ag_{7,j} & \cdots & ag_{7,8} \end{bmatrix}_{7X8} \tag{12}$$

Matrices $UG$ and $I$ are created in similar way, with $ug_{i,j}$ and $I_{i,j}$ taking the place of $ag_{i,j}$ respectively. The index $i$ stands for the ith pixel on the spoke outward, index $j$ stands for the jth spoke. Values $ag_{i,j}$, $ug_{i,j}$ can be calculated from the following equation (13):

$$ag_{i,1} = Gx, \quad ag_{i,2} = \frac{Gx + Gy}{\sqrt{2}}, \quad ag_{i,3} = Gy, \quad ag_{i,4} = \frac{-Gx + Gy}{\sqrt{2}},$$

$$ag_{i,5} = -Gx, \quad ag_{i,6} = \frac{-Gx - Gy}{\sqrt{2}}, \quad ag_{i,7} = -Gy, \quad ag_{i,8} = \frac{Gx - Gy}{\sqrt{2}}$$

$$ug_{i,1} = Gy, \quad ug_{i,2} = \frac{Gy - Gx}{\sqrt{2}}, \quad ug_{i,3} = -Gx, \quad ug_{i,4} = \frac{-Gx - Gy}{\sqrt{2}},$$

$$ug_{i,5} = -Gy, \quad ug_{i,6} = \frac{Gx - Gy}{\sqrt{2}}, \quad ug_{i,7} = Gx, \quad ug_{i,8} = \frac{Gx + Gy}{\sqrt{2}} \tag{13}$$

where Gx, Gy represent the gradient along x axis and the gradient along y axis respectively. Note that, the aligned gradient describes the gradient along each spoke, while the unaligned gradient describes the gradient orthogonal to each spoke.

## 3.3 Feature Extraction

The second step of the AGUG algorithm is feature extraction. The goal is to extract numerical representations for samples from $AG$, $UG$ and $I$. To create a robust ellipse detection algorithm, features should be informative, non-redundant, and maximize dimensionality [33, 34].

### 3.3.1 Intuition and Exploration

Like any other scientific exploration, feature extraction should be guided by intuition and logic. This section shows a series of experiences, through which the logic behind feature construction is formed.

**Ideal Case** We start from a simple and ideal case shown in figure 15.

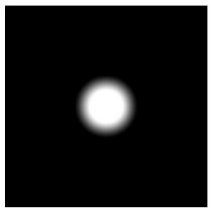


Figure 15: Simulated image

A simulated circle can be seen at the center of figure 15. The pixels inside of the circle receive an intensity value of 1 (white), the pixels outside receive a value of 0 (black). To mimic the blurring in real digital images, a 2D Gaussian Kernel is convolved with the binary image to create the illuminance function. The Gaussian Kernel is defined in equation (14):

$$G_{2D}(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{14}$$

where x is the distance from the origin in the horizontal axis, y is the distance from the origin in the vertical axis, and $\sigma$ is the standard deviation of Gaussian distribution.

Take the center of the circle in figure 15 as example. The corresponding $AG$, $UG$, $I$ are

gathered by the procedure introduced in section 3.2. Figures 16-18 show the $AG$, $UG$, $I$ respectively, where x axis represents pixels on each spoke, lines of different color represent results of different spokes. Note that, lines in figure 16 and 17 are overlapping.

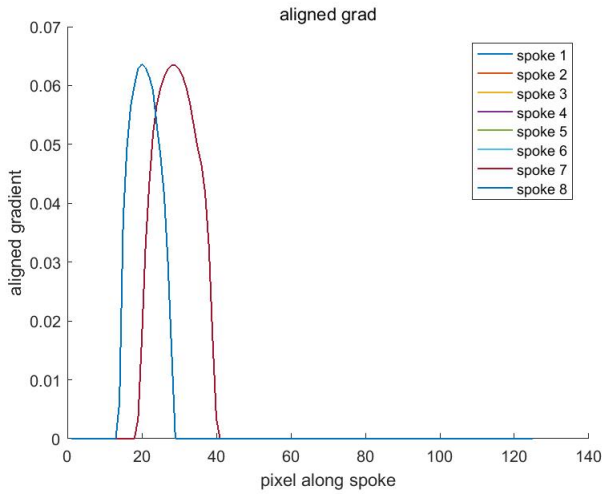



Figure 16: AG of the center of the circle in figure 15

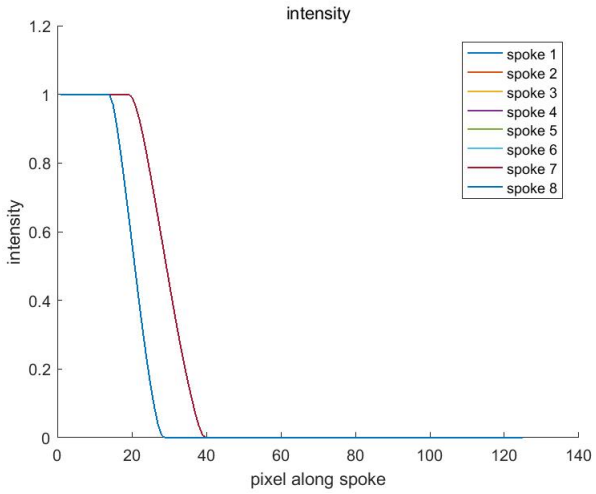



Figure 17: I of the center of the circle in figure 15

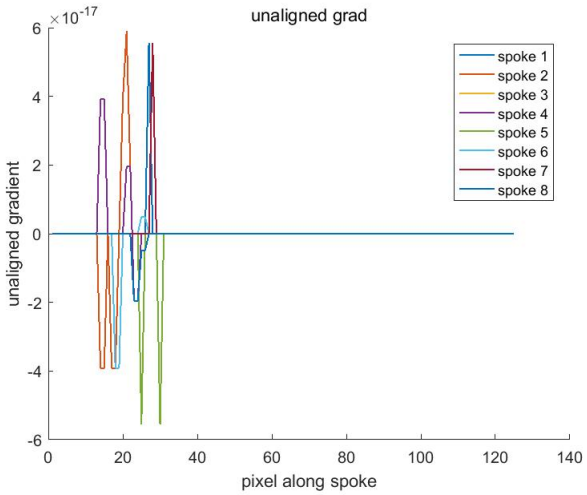



Figure 18: UG of the center of the circle in figure 15

We can conclude from figures 16-18 that:

- On each spoke, the pixels near the center or far away from the circle have aligned gradient equal to zero, while the pixels on the edge get the maximum of aligned gradient.
- Intensity drops from 1 to 0 from center outward along each spoke.
- For all pixels on eight spokes, unaligned gradient is a random value around $10^{-17}$, which can be considered as noise.

**Multiple cases** A more comprehensive experiment is designed to generalize the above experiment to multiple cases. The idea is shown in figure 19.

In figure 19, we can see a simulated circle with 16 red crosses labeled with numbers. Each of the cross represents a case belonging to different sets. Pixels in Set 1 (Case 1-4) are inside the circle. Pixels in Set 2 (Case 5-8) locate on the blurring edge. Pixels in set 3 (Case 9-16) are outside the circle.

$AG$, $UG$, $I$ along all eight spokes of cases 1-16 are gathered and plotted in Matlab. For simplification, case 1, case 5 and case 9 are taken as representatives to illustrate different patterns corresponding to each set.

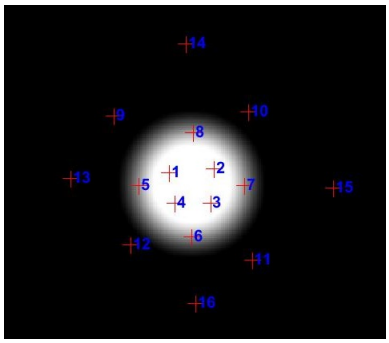



Figure 19: Simulated circle with labeled samples

**SET 1: inside the circle** Figure 20 shows the $AG$, $UG$, $I$ of pixel 1 in figure 19. Each subplot displays the aligned gradient, unaligned gradient and intensity on one of the eight spokes. Following observations can be obtained from figure 20:

- For each spoke, the intensity drops from 1 to 0 from center outward, the aligned gradient forms a bump as intensity drops, the unaligned gradient gets relatively small value all along the spoke.
- All eight spokes display the same pattern as described above.

We can further state that: for the pixel inside the circle, the mean of the aligned gradient peak of eight spokes should be relatively large, while the mean of the unaligned gradient peak of eight spokes should close to zero.

**SET 2: on the edge** Figure 21 shows the $AG$, $UG$, $I$ of pixel 5 in figure 19. Following patterns can be observed:

- For spokes 1-5, the intensity drops from some value near 0.6 to 0 from center outward, both the aligned gradient and the unaligned gradient form a bump as intensity drops;
- For spokes 6-8, the intensity first rises from some value near 0.6 to 1 then drops to 0, the aligned gradient and the unaligned gradient form two bumps as intensity changes, aligned gradient and unaligned gradient are of similar peak value;

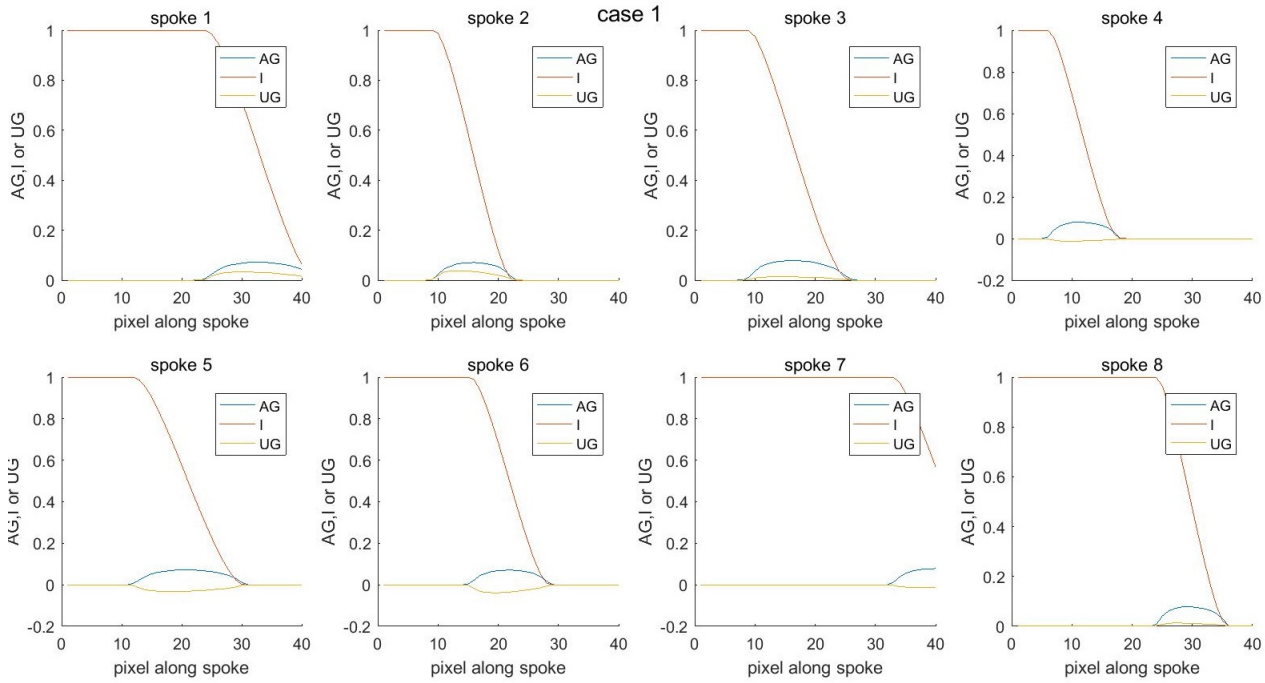



Figure 20: AU, UG, I of pixel 1 in figure 14

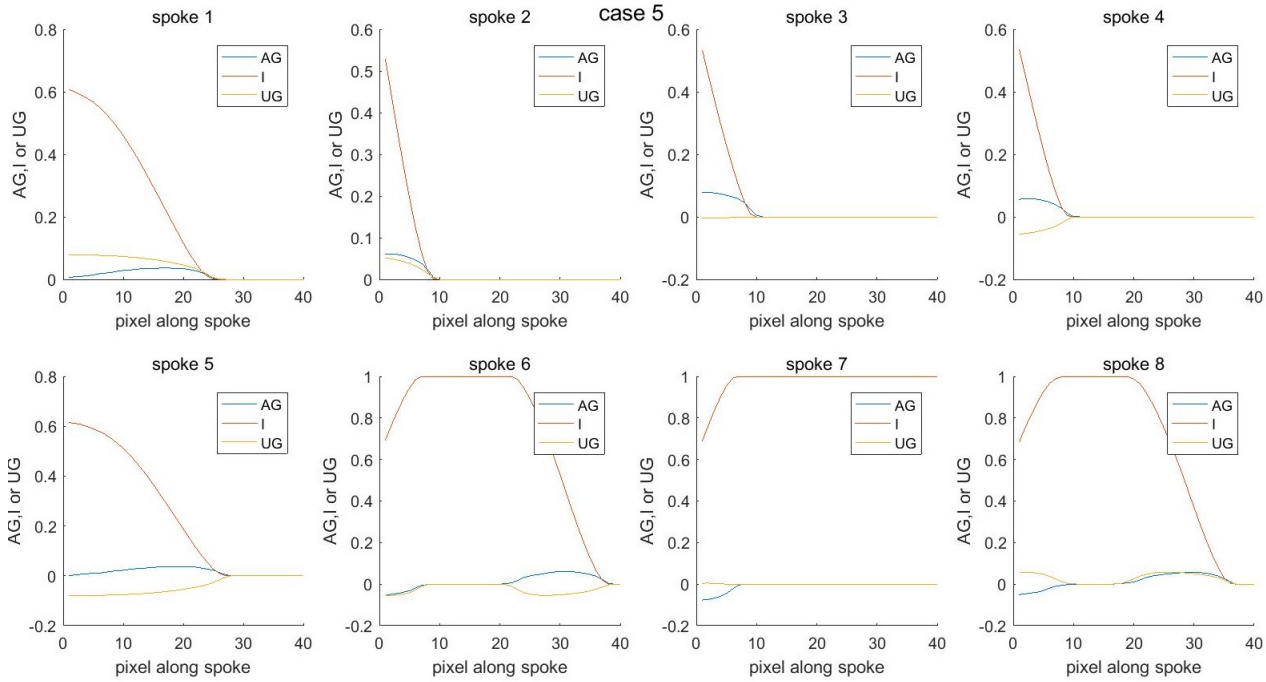



Figure 21: AU, UG, I of pixel 5 in figure 14

**SET 3: outside the circle** Figure 22 shows the $AG$, $UG$, $I$ of pixel 9 representing pixels outside the circle in figure 19. As can be observed from figure 22: most of the spokes (spokes 1-5) fail to hit the circle, while the rest display various situations.

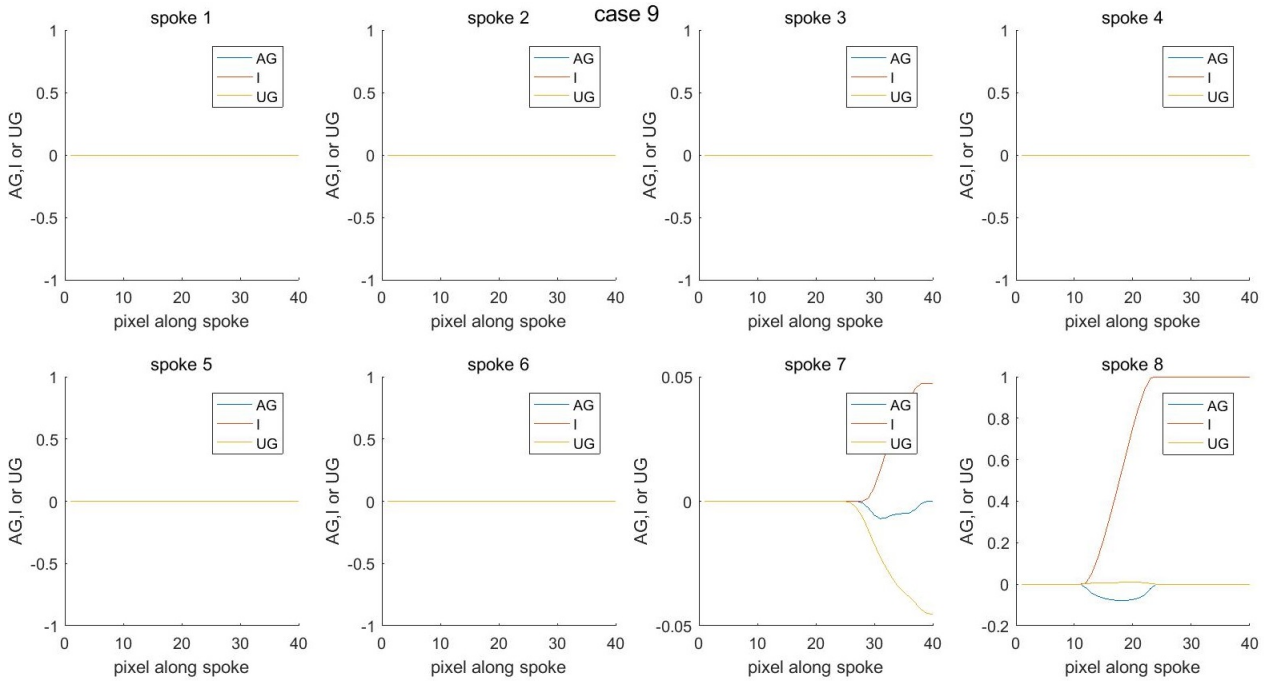



Figure 22: AU, UG, I of pixel 9 in figure 14

From above we can conclude that, different patterns of the $AG$, $UG$ and $I$ can be observed from different sets (inside the circle, on the edge and outside the circle) , which indicates useful information is buried in the $AG$, $UG$ and $I$, and signals lead to the feature construction in the following section.

### 3.3.2 Feature Construction

In this section, two sets of features, named Var-max and AG-I, are constructed, and tested on the testing pool generated in section 2.2. Libsvm developed by Chih-Chung Chang and Chih-Jen Lin is used to train SVM [32], necessary initial processing, raw data collection and feature extraction are performed in MATLAB.

**Var-max** Var-max is created based on observations obtained from experiments in section 3.3.1, which are indicated in table 2:

| | variance of max AG | mean of max AG | variance of max UG | mean of max UG |
|---|---|---|---|---|
| inside | small | big | small | small |
| outside | big | small | big | big |

| | variance of min AG | mean of min AG | variance of min UG | mean of min UG |
|---|---|---|---|---|
| inside | small | small (further away 0) | small | big (closer to 0) |
| outside | big | big | big | small |

Table 2: Logic behind Var-max construction

To get the mathematical form of Var-max, following operators are defined:

$$X = \begin{Bmatrix} x_1 & x_2 & \cdots & x_i & \cdots & x_{n_p} \end{Bmatrix}$$

where $X$ is a matrix, $x_1, \cdots, x_n$ are column vectors.

$$\overline{X} = \begin{Bmatrix} max(x_1) & max(x_2) & \cdots & max(x_i) & \cdots & max(x_{n_p}) \end{Bmatrix} \tag{15}$$

$$\underline{X} = \begin{Bmatrix} min(x_1) & min(x_2) & \cdots & min(x_i) & \cdots & min(x_{n_p}) \end{Bmatrix} \tag{16}$$

where var($\overline{X}$) refers to the variance of $\overline{X}$, mean($\overline{X}$) refers to the mean of $\overline{X}$.

We construct Var-max as a row vector containing eight elements, it can be written as :

$$[\ \text{var}(\overline{AG}) \quad \text{var}(\overline{UG}) \quad \text{var}(\underline{AG}) \quad \text{var}(\underline{UG}) \quad \text{mean}(\overline{AG}) \quad \text{mean}(\overline{UG}) \quad \text{mean}(\underline{AG}) \quad \text{mean}(\underline{UG})\ ]$$

Var-max is then calculated for each sample in the testing pool. Results of each element in Var-max are shown in figures 23-30 respectively.

Figures 23-24 show the results of var($\overline{AG}$) and var($\overline{UG}$). In the figures, x axis represents feature value, double y axis represent true positive sample and true negative sample. We

use blue and red to indicate results of true positive samples and true negative samples respectively.

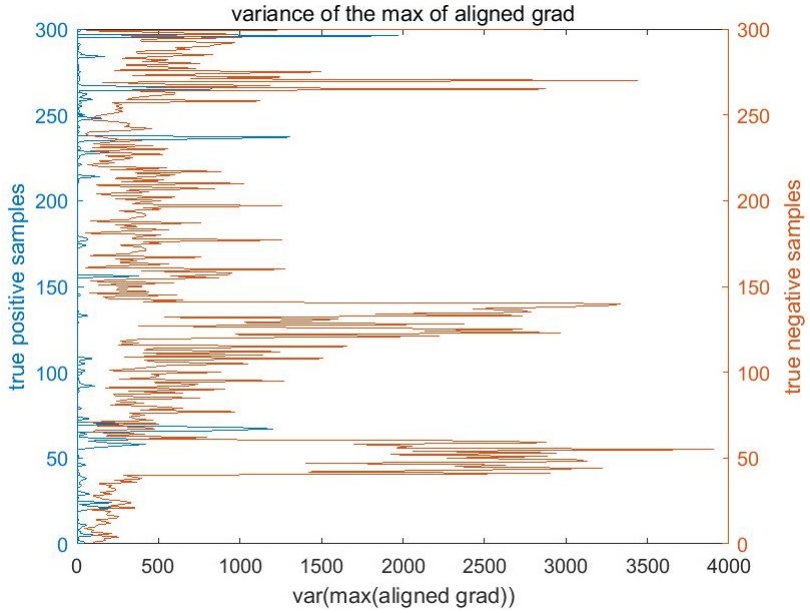



Figure 23: Testing results of var(max(AG))

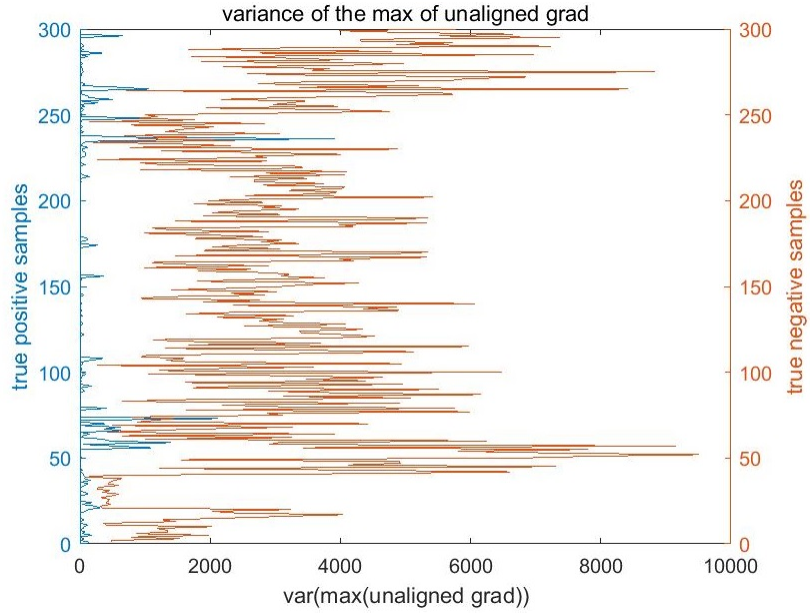



Figure 24: Testing results of var(max(UG))

From figure 23 we can see that, most of the true positive samples have var($\overline{AG}$)s less than 100, while var($\overline{AG}$)s of the most true negative samples are larger than 100 and vary in

a large region. An obvious margin can be observed between the blue line and the red line, suggesting var($\overline{AG}$) has a good separability. Similar conclusion can be concluded from figure 24.

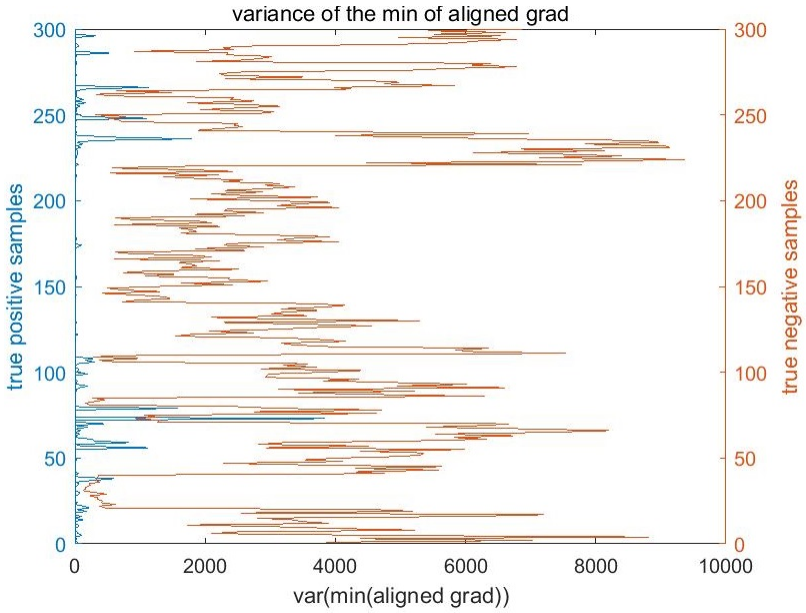



Figure 25: Testing results of var(min(AG))

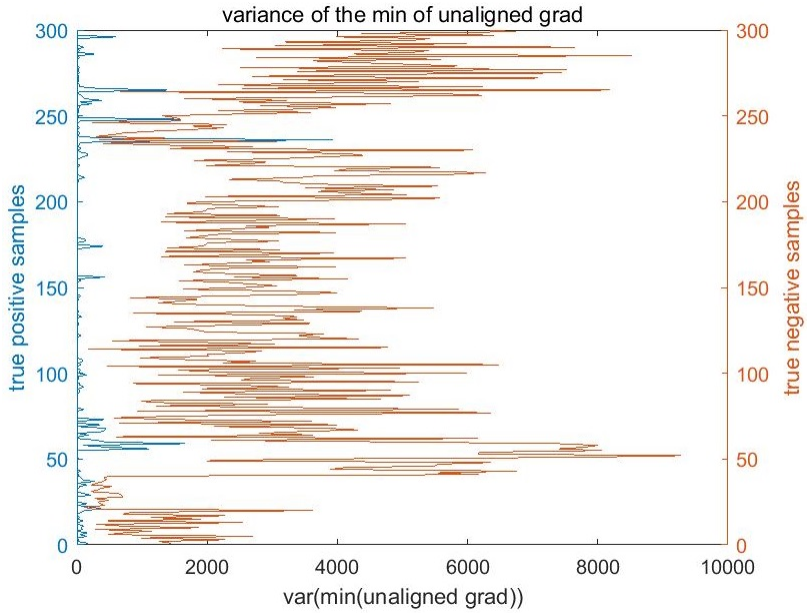



Figure 26: Testing results of var(min(UG))

Figures 25-26 show results of var($\underline{AG}$) and var($\underline{UG}$) in Var-max respectively. Because of

symmetry principle of positive values and negative values, similar pattern as that of var($\overline{AG}$) and var($\overline{UG}$) can be seen from figures 25-26.

Figures 27-28 show results of mean($\overline{AG}$) and mean($\overline{UG}$) in Var-max. From figure 27 we can see that, true positive samples have a much larger mean($\overline{AG}$) than true negative samples. Mean($\overline{UG}$) of true positive samples are relatively smaller than that of true negative samples. Also, in both cases of the mean($\overline{AG}$) and the mean($\overline{UG}$), the gap between true positive samples and true negative samples are pretty clear, suggesting good separability of mean($\overline{AG}$) and mean($\overline{UG}$).

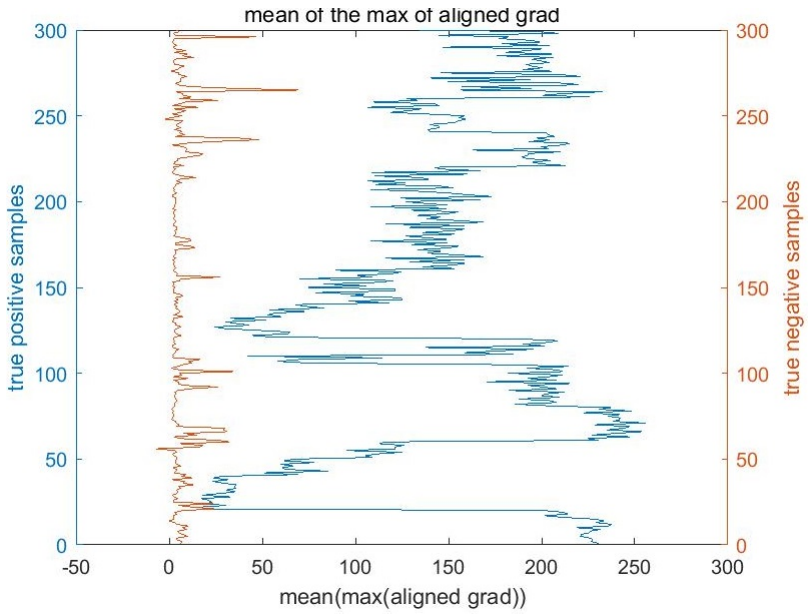



Figure 27: Testing results of mean(max(AG))

Figures 29-30 show results of mean($\underline{AG}$) and mean($\underline{UG}$). Similar observations as obtained from mean($\overline{AG}$) and mean($\overline{UG}$) can be seen from figures 29-30.

Results for Var-max are also in accordance with conclusions listed in table 2, indicating Var-max is a qualified set of features with good separability.

**AG-I** Another set of features, named AG-I, is constructed in this section. Figure 31 illustrates the logic behind AG-I construction. Figure 31 is one subplot obtained from figure 20, which shows the $AG$, $UG$ and $I$ on one spoke of a pixel inside the ellipse. We can see

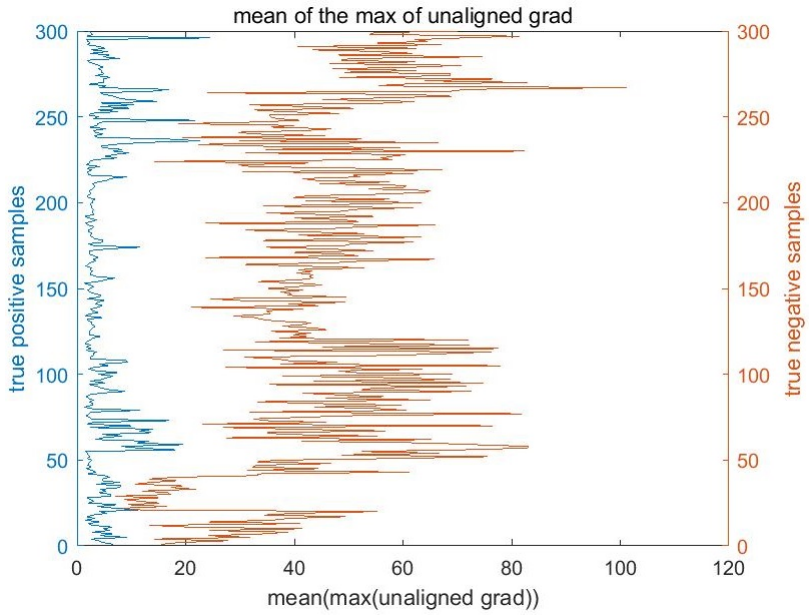



Figure 28: Testing results of mean(max(UG))

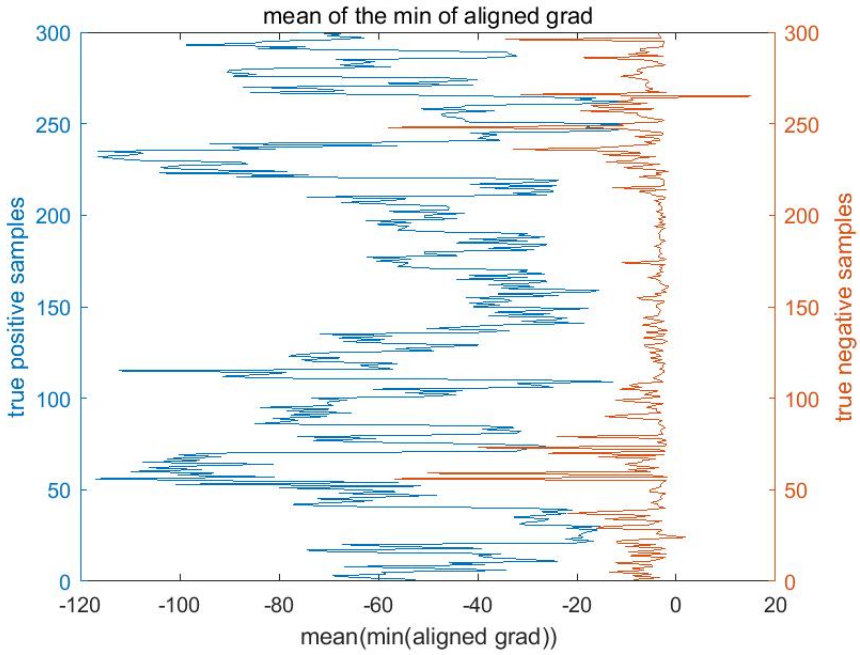



Figure 29: Testing results of mean(min(AG))

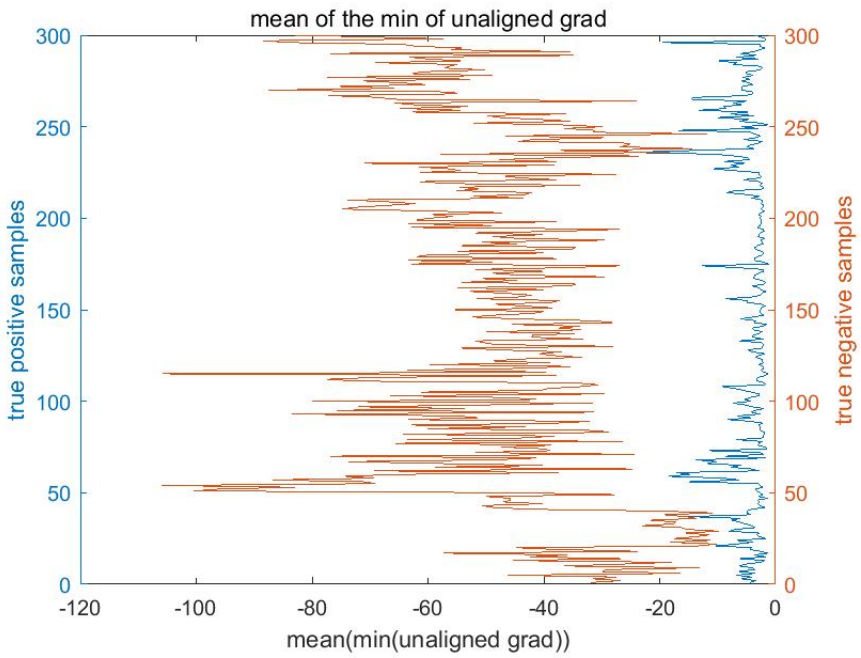



Figure 30: Testing results of mean(min(UG))

from figure 31 that, the peak of the aligned gradient indicates a turning pixel $P_n$, which is on the edge of the ellipse, separating pixels inside the ellipse $(P_0, \cdots, P_{n-1})$ from pixels outside the ellipse $(P_{n+1}, \cdots, P_m)$. The index m represents the length of the spoke. $I_{Pn}$ denotes the intensity of $P_n$.

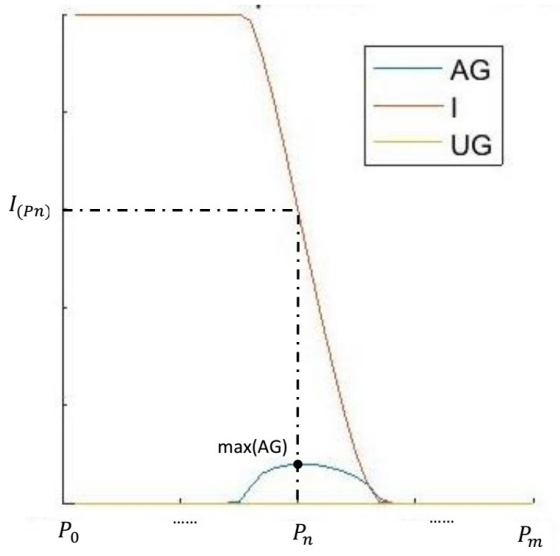



Figure 31: Illustration of generation of AG-I

It's obvious that the average intensity of $P_0, \cdots, P_{n-1}$ is larger than that of $P_{n+1}, \cdots, P_m$. Define:

$$v_{i1} = \frac{I_{P_0} + \cdots + I_{P_{n-1}}}{n+1} \tag{17}$$

$$v_{i2} = \frac{I_{P_{n+1}} + \cdots + I_{P_m}}{m-n+1} \tag{18}$$

where $v_{i1}$ represents the average intensity of $P_0, \cdots, P_{n-1}$, term $v_{i2}$ indicates the average intensity of $P_{n+1}, \cdots, P_m$. The index $i$ refers to the ith spoke. For true positive samples, all eight spokes should satisfy the following equation (19):

$$\frac{v_{i1}}{v_{i2}} > 1 \tag{19}$$

Such that, AG-I is defined as follows:

$$AG - I = [\, mean(V) \qquad var(V) \,] \tag{20}$$

where,

$$V = \left[ \begin{array}{ccccc} \frac{v_{11}}{v_{12}} & \cdots & \frac{v_{i1}}{v_{i2}} & \cdots & \frac{v_{81}}{v_{82}} \end{array} \right] \tag{21}$$

We then calculate AG-I for each sample in the testing pool. Results are plotted in figures 32-33, presenting var($V$) and mean($V$) respectively.

In figure 32, true positive samples have generally smaller var($V$) than true negative samples. Results of mean($V$) shown in figure 32 accords with our expectation described in equation (19), that is, true positive samples have mean($V$)s larger than 1, while mean($V$)s of true negative samples distribute evenly near 1.

Observations from figures 32-33, indicate AG-I provides an efficient set of features.

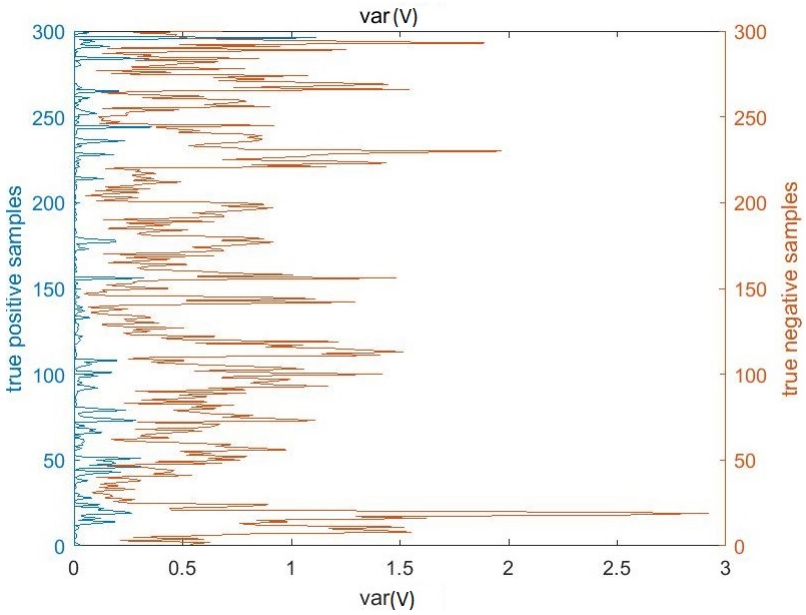



Figure 32: Testing results of var($V$)

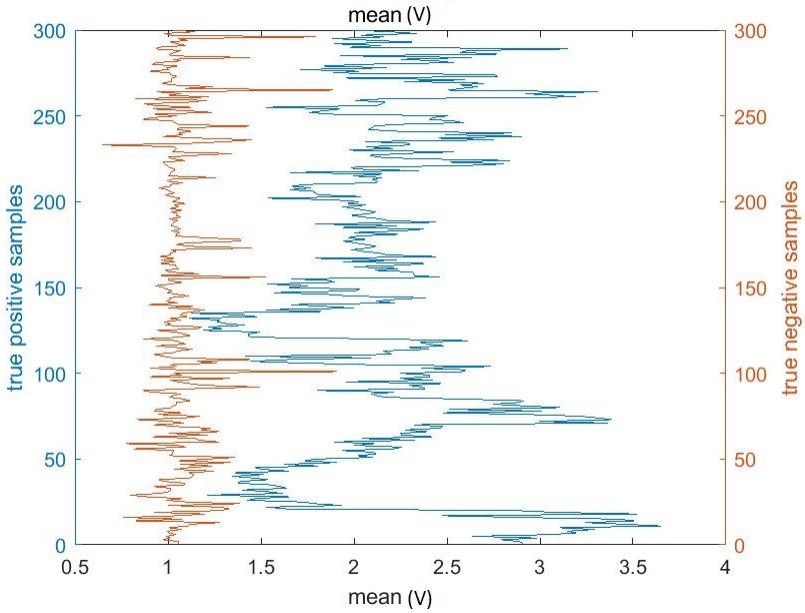



Figure 33: Testing results of mean($V$)

## 3.4 Feature Nondimensionalization and Scaling

Feature nondimensionalization and scaling are two processes conducted before training in the proposed algorithm.

### 3.4.1 Feature Nondimensionalization

Nondimensionalization is partial or full removal of units from variables. By doing so, the impact of the unit will be eliminated. One simple way of nondimensionalization is to divide the dimensional variables by another variable with the same or equivalent unit. In our cases, AG-I is dimensionless itself. Table 3 explains how Var-max is nondimensionalized.

| dimensional features | nondimensionalized features |
|---|---|
| $var(\overline{AG})$ | $\frac{var(\overline{AG})}{mean(\overline{I})^2}$ |
| $var(\overline{UG})$ | $\frac{var(\overline{UG})}{mean(\overline{I})^2}$ |
| $var(\underline{AG})$ | $\frac{var(\underline{AG})}{mean(\overline{I})^2}$ |
| $var(\underline{UG})$ | $\frac{var(\underline{UG})}{mean(\overline{I})^2}$ |
| $mean(\overline{AG})$ | $\frac{mean(\overline{AG})}{mean(\overline{I})}$ |
| $mean(\overline{UG})$ | $\frac{mean(\overline{UG})}{mean(\overline{I})}$ |
| $mean(\underline{AG})$ | $\frac{mean(\underline{AG})}{mean(\overline{I})}$ |
| $mean(\underline{UG})$ | $\frac{mean(\underline{UG})}{mean(\overline{I})}$ |

Table 3: Feature nondimensionalization

### 3.4.2 Feature Scaling

Features are scaled before they are put into SVM classifier. The main advantage of scaling is to avoid attributes in greater numeric ranges dominating those in smaller numeric ranges. Feature scaling can also avoid numerical difficulties during calculation. In SVM, each attribute is suggested to be scaled to the range $[-1, 1]$ or $[0, 1]$, and it is known that different feature scaling methods can lead to different results [35]. Investigation on best feature scaling and shaping techniques can be found in references [36, 37]. In general, a zero mean-unit feature normalization will yield better results with SVM.

Z-score normalization is a classical type of zero mean-unit feature normalization, which is also used to scale features in this thesis [35]. The result of z-score normalization is that features will have the properties of a standard normal distribution with $\mu = 0$ and $\sigma_z = 1$, where $\mu$ is the mean and $\sigma_z$ is the standard deviation from the mean; standard scores (z scores) of the samples are calculated as follows:

$$z = \frac{x - \mu}{\sigma_z} \tag{22}$$

## 3.5 Algorithm Overview

As a SVM-based classification algorithm, the proposed AGUG algorithm comes into two stages, namely training and testing. The training process is performed in following steps:

- Step 1: Label samples as true positive and true negative manually;
- Step 2: Gather raw data ($AG, UG$ and $I$) of all samples according to equations (12)-(13);
- Step 3: Calculate features (Var-max or AG-I) from corresponding equations (equations (15)-(16) or equations (17)-(21));
- Step 4: Nondimensionalize features according to table 3, if needed;
- Step 5: Scale features according to equation (22), and train SVM with processed features;

Figure 34 shows the flow chart of the proposed AGUG algorithm.

Note that, as for SVM, different features lead to different results. For comparison, SVM are trained with three different sets of features (dimensional Var-max, dimensional Var-max and AG-I, dimensionless Var-max and AG-I) and tested on the same testing pools in the following chapter. The three AGUG algorithms with different features are named in table 4.

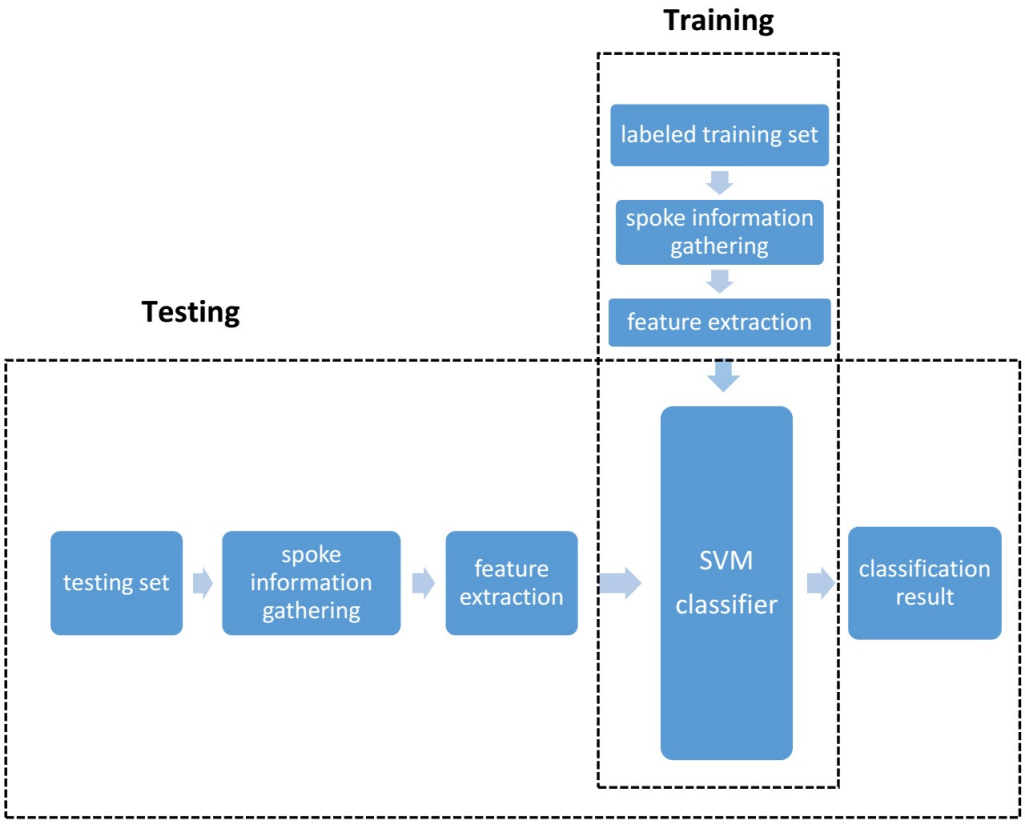



Figure 34: Flow chart of the proposed AGUG algorithm

| Name | Characteristic |
|---|---|
| AGUG (8 dimensional features) | Var-max (dimensional) |
| AGUG (10 dimensional features) | Var-max (dimensional), AG-I |
| AGUG (10 dimensionless features) | Var-max (dimensionless), AG-I |

Table 4: Name clarification for further experiments

# 4 Results and Comparison

In this chapter, comparing experiments are conducted between algorithms, which are Randomized Hough Transform (RHT) and three variations of AGUG algorithm. All involved algorithms are tested on the original image set, blurred image set and low-contrast image set respectively. Section 4.1 talks about the method of sample generation; section 4.2 introduces technology behind image modification, to test focus and contrast; section 4.3 investigates the influence of blurring and contrast on AGUG algorithm feature by feature; results of AGUG algorithms with different features and Randomized Hough Transform (RHT) on original image set, blurred image set and low-contrast image set are analyzed and compared in section 4.4.

## 4.1 Sample Generation

Figures 35-36 are two source images for sample generation in training sets and testing sets respectively. Patches with the size of $15 * 15$ pixels are picked from those two images. Each patch can be either a true positive sample (there is an elliptical landmark) or true negative sample (there is no landmark). Examples can be seen in figure 37. We generate a training set with 1791 samples, among which 792 are true positive, 999 are true negative, and a testing set with 600 samples, half of which are true positive, the rest are true negative.

Figure 35: Source image for training set

Figure 36: Source image for testing set

Figure 37: Example of true positive sample(left) and true negative sample(right)

## 4.2 Image Simulation

Image processing is used to generate blurred images and low-contrast images. By adjusting blurring or contrast parameters, images with different blur degree or contrast are generated.

### 4.2.1 Blurred Image Generation

To create blurred images, a 2D Gaussian Kernel is used to create the blurring operator. The formula for Gaussian Kernel is shown in equation (14) section 3.3.1. We define $\sigma$ as blurring parameter. By adjusting blurring parameter from 0.5 to 9.5 with interval of 0.5, 10 different blurring Gaussian operator are generated, which are then used to generate 10 different blurred images, making up the blurring image set. Samples are then selected from the generated blurred images.

### 4.2.2 Low-Contrast Images Generation

To create low-contrast images, a function is introduced to map intensity from $[0, 1]$ to $[I_1, I_2]$. For pixel $p$, contrast adjustment can be conducted using the following formula:

$$p_2 = p_1(I_2 - I_1) + I_1 \tag{23}$$

where $p_1$ represents initial intensity value, $p_2$ refers to the adjusted intensity value. By setting $I_2$ as 1, increasing $I_1$ from 0.1 to 0.9 at interval of 0.05, 17 degraded images with different levels of contrast are generated. Samples are then selected from the generated low-contrast images.

## 4.3 Impact of Blurring and Contrast Variation on Features

Impact of blurring and contrast variation on features are investigated in this section. Features are calculated by the same scheme introduced in section 3.3.2. All 10 features (8 in Var-max, 2 in AG-I) generated in section 3.3 are tested on a series of blurring and low-contrast images from section 4.2. In the following section, mean($V$) is taken as an example to illustrate the conclusion.

### 4.3.1 Blurred Image Test

Figure 38 displays results of mean($V$) on blurred images with various blurring parameter $\sigma$ values. Six subplots show cases where $\sigma = 0.1$, $\sigma = 1$, $\sigma = 2$, $\sigma = 3$, $\sigma = 4$, $\sigma = 5$ respectively. We can see that, as $\sigma$ gets larger, the gap between true positive samples and true negative samples gets narrower, and almost disappears when $\sigma = 5$.

### 4.3.2 Low-Contrast Image Test

Results of mean($V$) on images with different contrast can be seen in figure 39. In the figure, subplots represent cases with different contrast parameter $I_1$. A larger $I_1$ indicates a lower

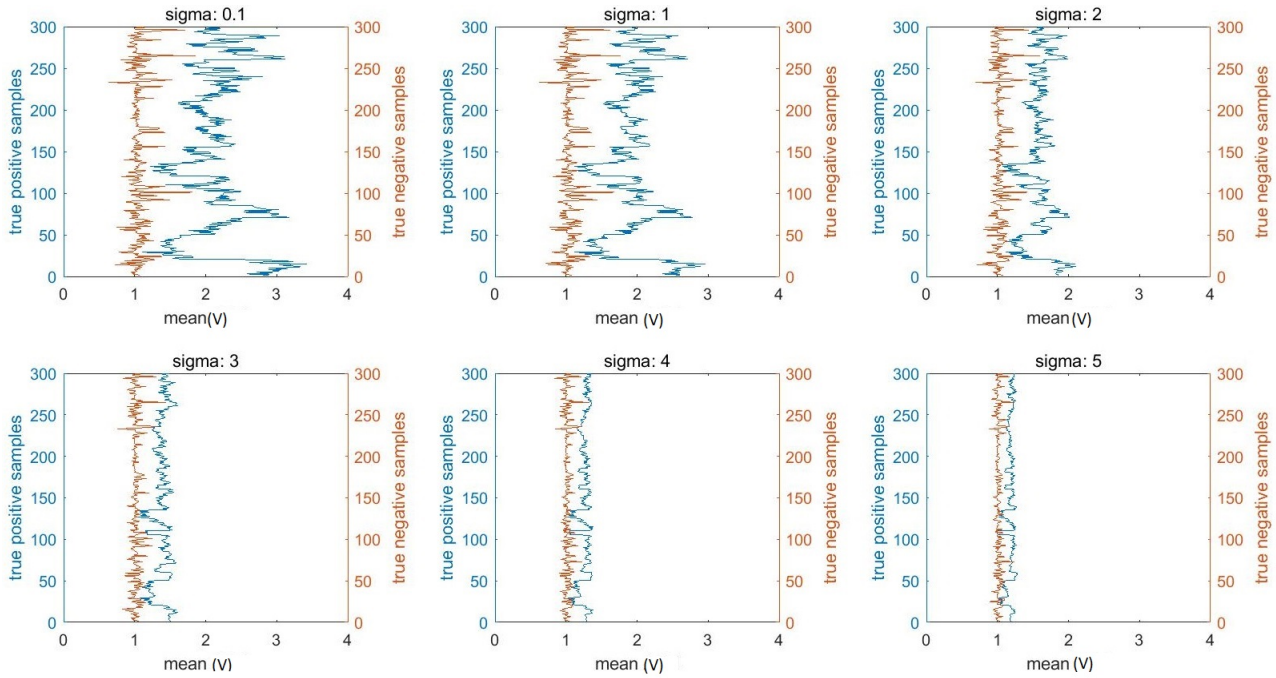



Figure 38: Results of mean($V$) on blurred images with different blurring parameters

contrast.

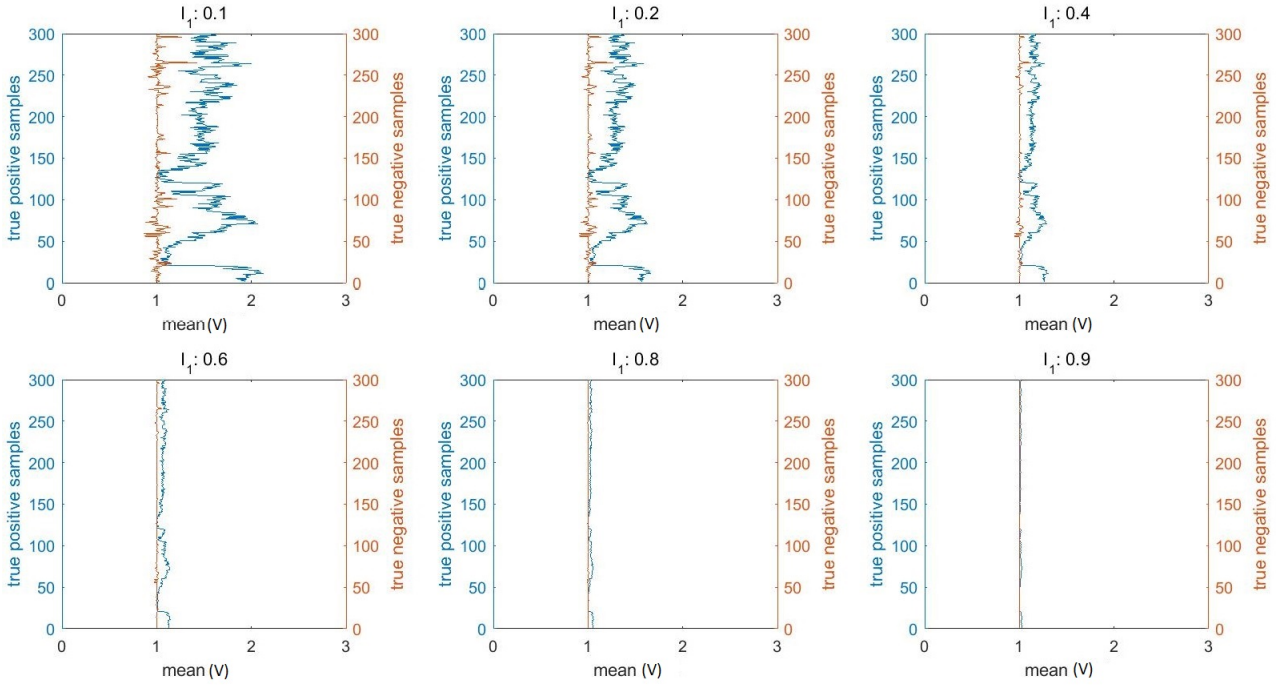



Figure 39: Results of mean($V$) on images with different contrasts

As we can see from figure 39, true positive samples and true negative samples are getting

harder to separate as $I_1$ increases from 0.1 to 0.9, and almost inseparable when $I_1 = 0.8$, indicating the failure of detection.

The observations from figures 38-39 are consistent with the logic that, the more degraded the image is, the harder to tell true positive samples from true negative samples, and the algorithm will totally fail when the image is degraded beyond some degree.

## 4.4 Results and Comparison

In this section, the performance of the proposed AGUG algorithm with 8 dimensional features, 10 dimensional features, 10 dimensionless features (name clarification in table 4) and Randomized Hough Transform (RHT) are analyzed and compared. Experiments are conducted on three sets: original image set selected from figure 36, blurred and low-contrast image sets generated in section 4.3. Performance of all four algorithms on the three testing sets are displayed and discussed in sections 4.4.1-4.4.4. Sections 4.4.5-4.4.7 compare the results from sections 4.4.1-4.4.4 by sensitivity, specificity and accuracy. Note that, all following experiments use the same Support Vector Machine (SVM) trained from the training set created in section 4.1.

### 4.4.1 Results on Original Image

Results of all four algorithms on original image set are listed in table 5:

| | accuracy | sensitivity | specificity |
|---|---|---|---|
| RHT (threshold:0.12) | 74.50% | 84.00% | 65.00% |
| AGUG (8 dimensional features) | 96.00% | 93.67% | 98.33% |
| AGUG (10 dimensional features) | 98.00% | 97.33% | 98.67% |
| AGUG (10 dimensionless features) | 99.00% | 98.33% | 99.67% |

Table 5: Results comparison on the original image set

For discussion, table 6 is rearranged in terms of error rates.

| | total error rate | type II error rate* | type I error rate** |
|---|---|---|---|
| RHT (threshold:0.12) | 25.50% | 16.00% | 35.00% |
| AGUG (8 dimensional features) | 4.00% | 6.33% | 1.67% |
| AGUG (10 dimensional features) | 2.00% | 2.67% | 1.33% |
| AGUG (10 dimensionless features) | 1.00% | 1.67% | 0.33% |

* type II error is the failure to retain a true positive sample (false negative)
** type I error is the failure to reject a true negative sample (false positive)

Table 6: Error rates comparison on the original image set

We can see from table 5 and 6 that:

- AGUG algorithms (regardless of feature difference) have type II error rates from 1.67% to 6.33%, which are 2 to 8 times smaller than the type II error rate of RHT. This observation indicates that, AGUG algorithms are almost 10 times less likely to falsely infer the absence of true positive samples than RHT.

- In average, AGUG algorithms have 30 times smaller type I error rates than RHT, which illustrates the huge advantage of AGUG algorithms over RHT in detecting true negative samples.

- The type II error rate decrease 4.66% from AGUG (8 dimensional features) to AGUG (10 dimensional features), indicating AG-I (two more features added) a powerful set of features in detecting true positive samples.

- Both the type I and II error rates drop 1% from AGUG (10 dimensional features) to AGUG (10 dimensionless features), which indicates that nondimensionalization is a powerful process to improve the algorithm.

### 4.4.2 Results of RHT on Blurred and Low-Contrast Image Sets

Figure 40 shows the performance of RHT on blurred image set. As we can see, both of sensitivity and specificity drop as $\sigma$ increases. Sensitivity drops from 92.5% to nearly 10%

as $\sigma$ increases to 6. Specificity stops near 50%. RHT's performance on low-contrast images is shown in figure 41. Sensitivity and specificity vary in similar trend as in figure 40.

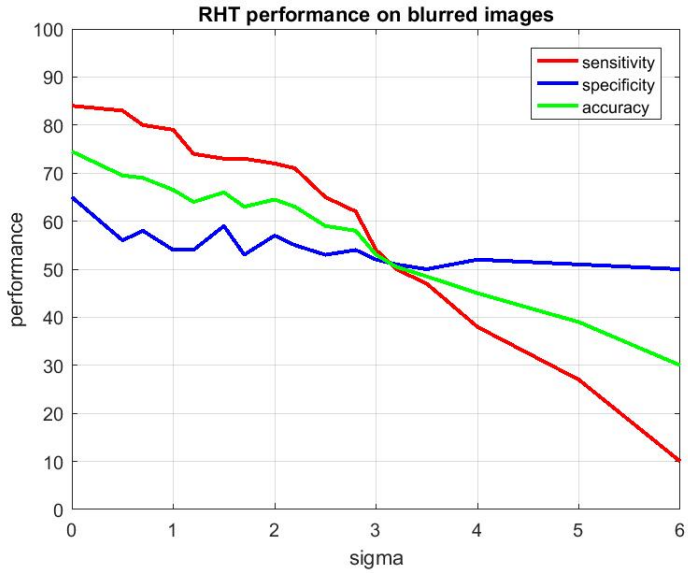



Figure 40: Performance of RHT on blurring image set

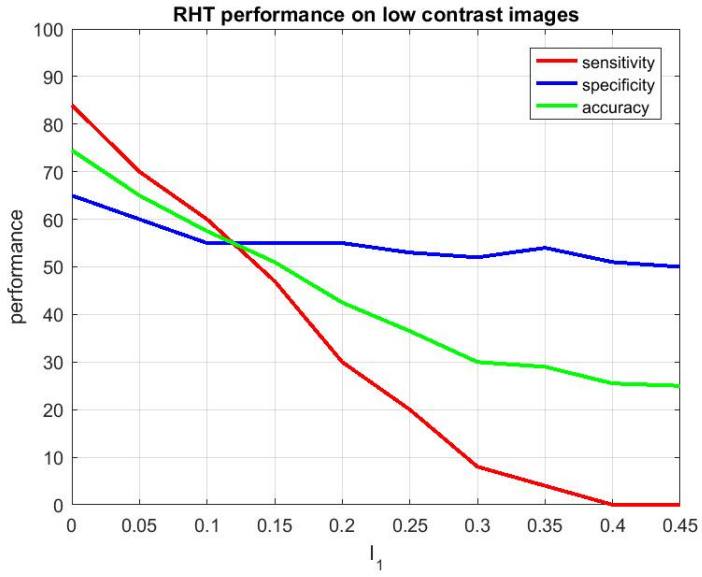



Figure 41: Performance of RHT on images with different contrasts

### 4.4.3 Results of AGUG (8 dimensional features) on Blurred and Low-Contrast Image Sets

Figure 42 shows the performance of AGUG with 8 dimensional features (Var-max only) on blurred image set. In figure 42, sensitivity decreases as $\sigma$ increases, while specificity stays at some value near 100% after $\sigma = 1$.

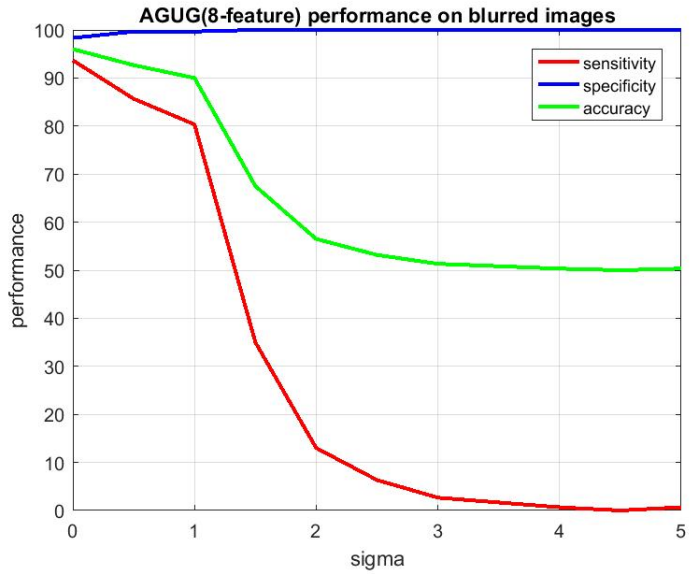



Figure 42: Performance of AGUG(8 features) on blurring images

Performance of AGUG (8 features) on low-contrast images is shown in figure 43. Sensitivity and specificity follow the same pattern as in blurred image set in the region $I_1 : [0.1, 0.8]$. Specificity drops from 100% to 85% after $I_1 > 0.8$. The algorithm still has a sensitivity of 20% when $I_1 = 0.9$.

### 4.4.4 Results of AGUG (10 dimensional features) on Blurred and Low-Contrast Image Sets

Figures 44-45 shows performance of AGUG with 10 dimensional features on blurred and low-contrast image sets respectively. In both tests, sensitivity decreases as $\sigma$ or $I_1$ increases, specificity stays around 98%. Algorithm fails as $\sigma > 9$ in blurred image set, $I_1 = 0.45$ in low-contrast image set.

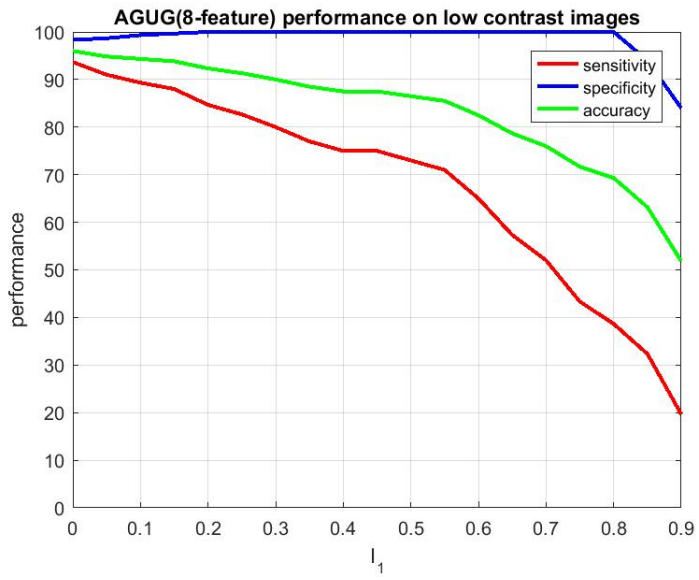



Figure 43: Performance of AGUG(8 features) on images with different contrasts

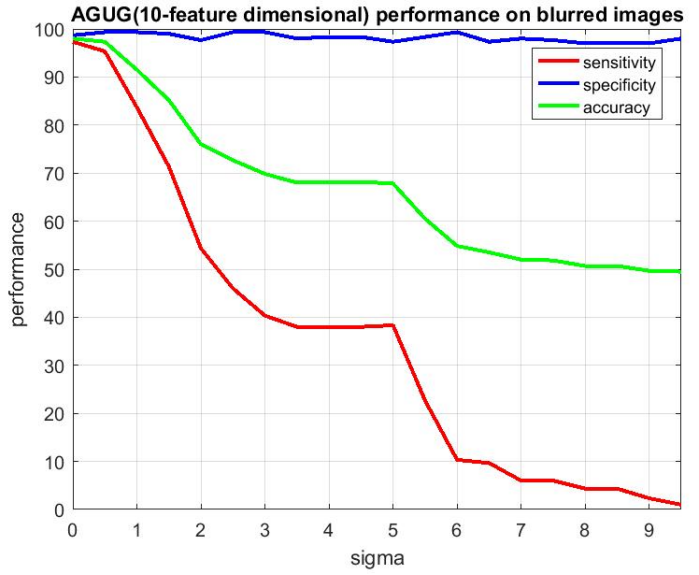



Figure 44: Performance of AGUG with 10 dimensional features on blurred images

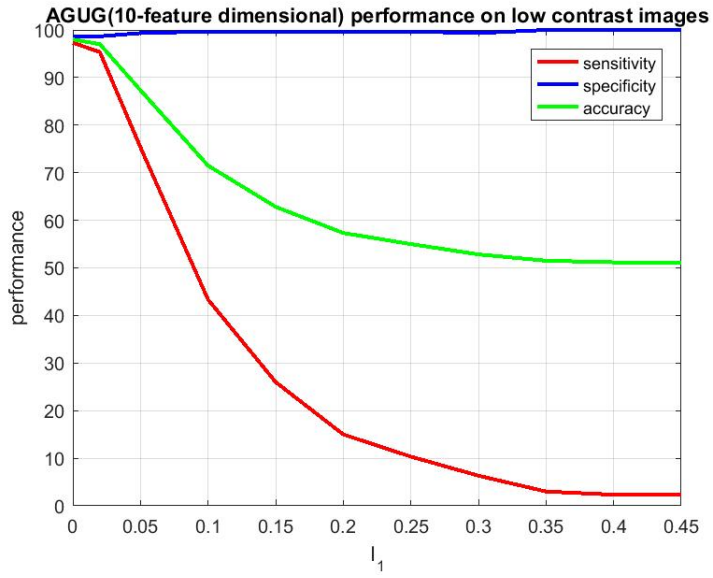



Figure 45: Performance of AGUG with 10 dimensional features on images with different contrasts

### 4.4.5 Results of AGUG (10 dimensionless features) on Blurred and Low-Contrast Image sets

Results of feature-nondimensionalized AGUG are shown in figures 46-47. For blurred image set, sensitivity decreases to less than 10% as $\sigma$ increases to 14, while specificity keeps above 93%. In low-contrast images testing, algorithm fails as $I_1 = 0.45$.

We can conclude from results in sections 4.4.1-4.4.4 that, for Randomized Hough Transform (RHT):

- Blurring and contrast variation cause a reduction in both sensitivity and specificity.
- As image degrades to some extent, no true positive samples and only 50% of true negative samples can be detected.

for the proposed AGUG algorithm in general (regardless of feature difference):

- Sensitivity drops as image degradation gets worse, while specificity is hardly influenced by blurring or contrast variation.
- As image degrades to some degree, the algorithm will mark all samples as true negative.

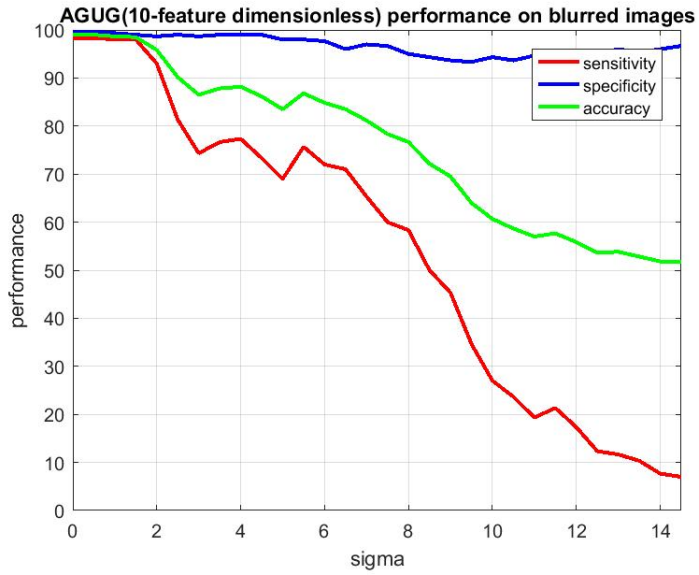



Figure 46: Performance of AGUG with 10 dimensionless features on blurred images

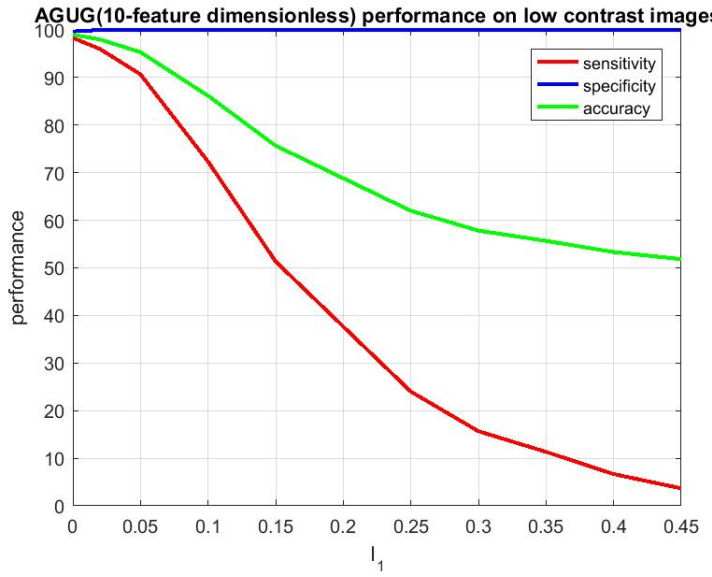



Figure 47: Performance of AGUG with 10 dimensionless features on images with different contrasts

### 4.4.6 Sensitivity Comparison

Comparison of sensitivity from figures 40, 42, 44, 46 are shown in figures 48-49. We can see from figure 48 with blurred images that, sensitivity of all four algorithms drop as $\sigma$ increases, AGUG with 10 dimensionless features has the highest sensitivity in general, AGUG with 8 features is the worst. All AGUG algorithms have better performance than RHT when $\sigma$ is less than 1.

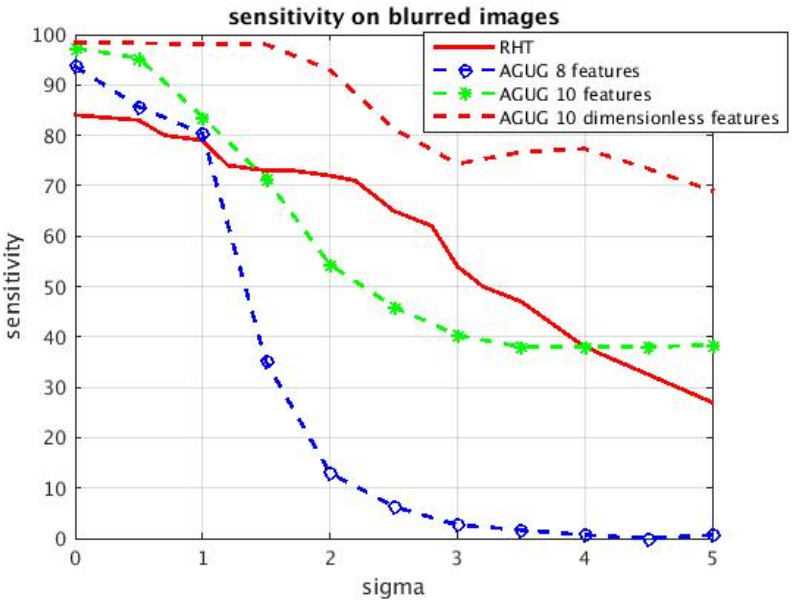



Figure 48: Sensitivity comparison of all four algorithms on blurred images

Figure 49 shows the sensitivity comparison on low-contrast images. We can see that, AGUG with 8 features has the overall highest sensitivity, RHT, AGUG with 10 features and AGUG with 10 dimensionless features lose efficacy at similar rate as $\sigma$ increases. The latter three algorithms all fail as $I_1 = 0.45$. Note that, AGUG (10 dimensional features) has higher sensitivity than AGUG (10 dimensionless features) in general on both figure 48 and figure 49, indicating the advantage of feature nondimensionalization in improving the algorithm's robustness.

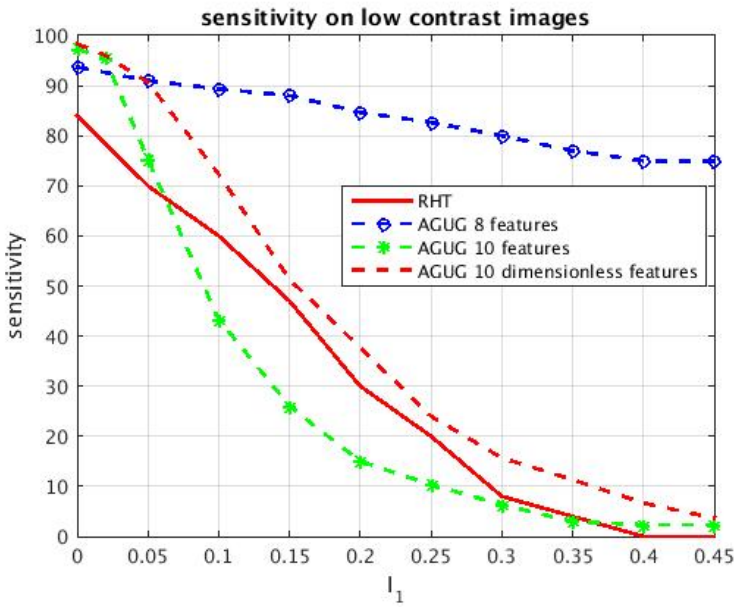



Figure 49: Sensitivity comparison of all four algorithms on images with different contrasts

From the above observations we can get following conclusion that,

- Feature selection has significant influence on detection results. With properly selected features, eg. nondimensionalized Var-max and AG-I, the proposed AGUG algorithm has a stronger robustness in sensitivity and specificity than RHT.
- AG-I (additional feature set in AGUG (10 features) than AGUG (8 features)) has a strong robustness against blurring, while is pretty sensitive to contrast variation.
- Feature nondimensionalization is powerful in improving algorithm's robustness.

### 4.4.7 Specificity Comparison

Figures 50-51 compare the specificity of all four algorithms on blurred image set and low-contrast image set respectively. In both cases, all AGUG algorithms (regardless of feature difference) have specificity close to 100% regardless of the condition of the image, while the specificity of RHT drops from 65% as the image degrades and stays at 50%.

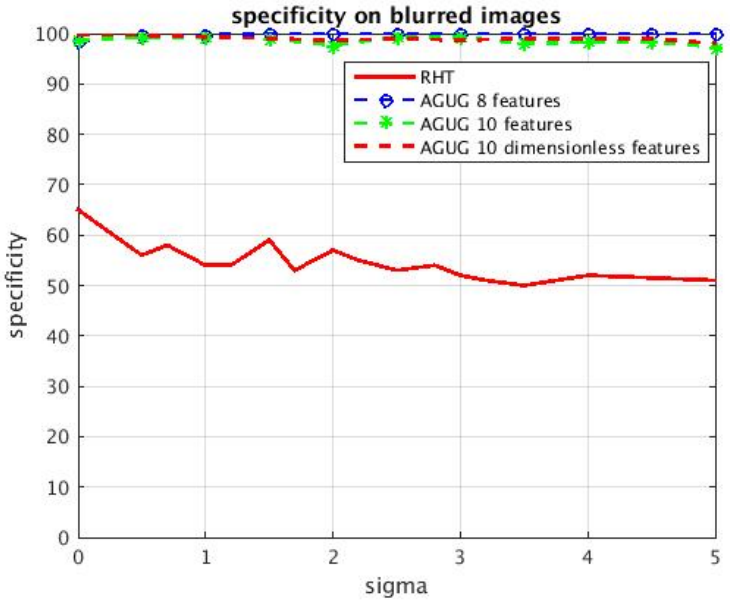



Figure 50: Specificity comparison of all four algorithms on blurred images

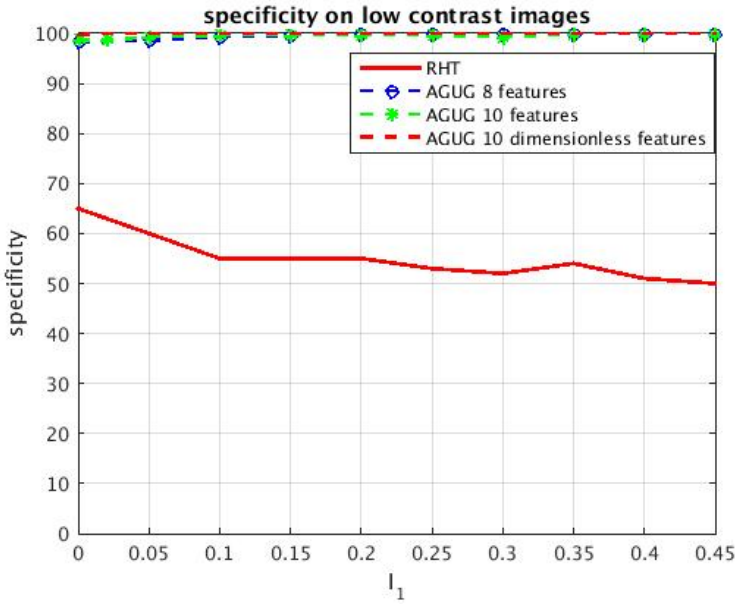



Figure 51: Specificity comparison of all four algorithms on images with different contrasts

This observation suggests the proposed AGUG algorithms have strong robustness in detecting true negative samples against blurring and contrast variation, while RHT is sensitive to blurring and contrast variation.

### 4.4.8 Accuracy Comparison

Finally, accuracy of all four algorithms on blurred and low-contrast image set are compared. As the average of sensitivity and specificity, accuracy reflects the overall performance of an algorithm. From figure 52 we can see that, AGUG with 10 dimensionless features has the highest accuracy, which is nearly 25% higher in average than that of RHT.

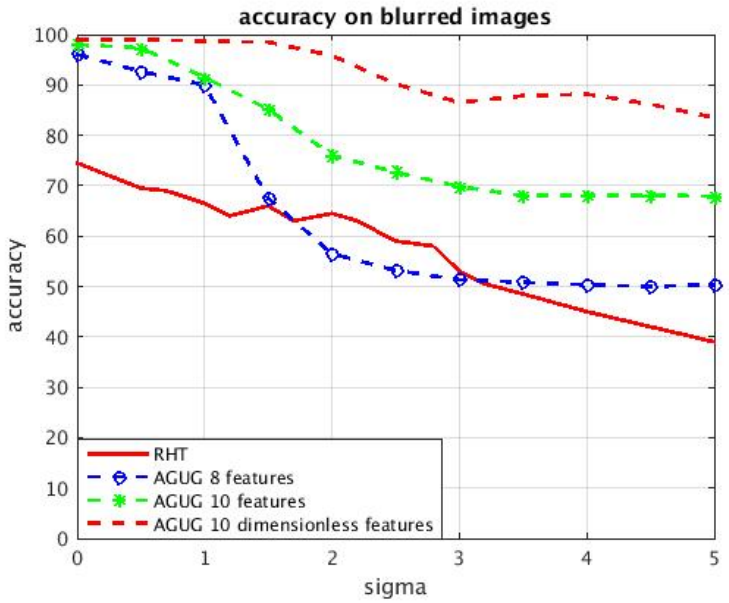



Figure 52: Accuracy comparison of all four algorithms on blurred images

In low-contrast image testing, as we can see in figure 53, when $I_1 < 0.05$, AGUG (10 dimensionless features) has the highest accuracy, while AGUG (8 features) rises to the highest of near 90% after $I_1 = 0.05$. RHT has the worst accuracy among all four algorithms in this test.

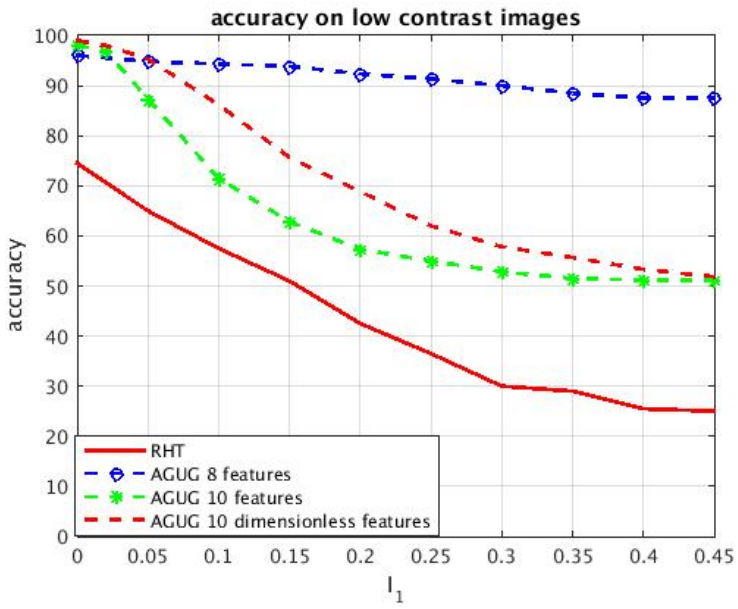



Figure 53: Accuracy comparison of all four algorithms on images with different contrasts

## 4.5 Summary

From comparison experiences in this chapter, we can conclude that, compared to Randomized Hough Transform (RHT), the proposed AGUG algorithm:

- has better performance, which indicates both a higher sensitivity and a higher specificity.
- has a stronger robustness in blurring and contrast variation with properly selected features.

# 5 Conclusion and Discussion

## 5.1 Conclusion

As a widely used process, ellipse detection has been studied for decades. All traditional ellipse detection algorithms, such as the Hough Transform (HT) variants reviewed in this thesis, require some form of threshold, which is a major cause of detection failure, especially in the challenging case of Moire Phase Tracking (MPT) target images.

The AGUG algorithm proposed in this thesis requires no threshold in the ellipse detection procedure. The proposed threshold free detection algorithm detects ellipses by training a Support Vector Machine (SVM) classifier, which predicts whether the input pixels are inside of ellipses or not. Features are extracted from aligned and unaligned gradient values corresponding to the input pixels.

Tested on 600 samples generated from real digital images, the proposed AGUG algorithm has the type II and I error rates 10 times and 100 times smaller than those of a classical ellipse detection algorithm, Randomized Hough Transform (RHT). Experiments on series of blurred and low-contrast images also verify that with proper selection of features, the proposed AGUG algorithm is also robust to blurring and contrast variation.

## 5.2 Impact of this Thesis

This thesis sheds lights on threshold free ellipse detection algorithms. Prior to this, known ellipse detection algorithms relied on thresholds, which would lead to failure in low-quality images. This thesis suggests a possible way to remove thresholds from the ellipse detection procedure, and verifies that the removal of threshold improves the algorithm's robustness. The improved robustness can handle lower-quality images in a wide range of applications. Any technique requires ellipse detection can benefit from the findings of this thesis.

## 5.3 Future Work

Two investigative paths could be further pursued based on the findings of this thesis.

### 5.3.1 Further Investigation into Feature Selection

An improvement could potentially be achieved by investigation into feature selection. As comparing experiment in sections 4.4.5-4.4.7 indicates, different features generate different results in machine learning scheme, and thus, feature selection should fit for specific goals, and be carried out with caution. It's highly convincing that, features with better separability could be generated, and consequently, the performance of AGUG algorithm will be improved.

### 5.3.2 Modifications to Other Shape Detection Problems

The proposed algorithm in this thesis is especially designed for elliptical landmark detection. This detection scheme could be modified to other shape detection problems by constructing corresponding features.

# References


[1] J. A. Gutierrez and B. S. Armstrong, *Precision Landmark Location for Machine Vision and Photogrammetry: Finding and Achieving the Maximum Possible Accuracy*. Springer Science & Business Media, 2007.

[2] A. Nikhanj, *Adaptation of Moiré phase tracking to a mobile device for field 3D data collections*. PhD thesis, The University of Wisconsin-Milwaukee, 2015.

[3] C. Foster, *Discovery and correction of bias in Precision Landmark Location*. PhD thesis, The University of Wisconsin-Milwaukee, 2012.

[4] W. W. M. Khairosfaizal and A. Nor'aini, "Eyes detection in facial images using circular hough transform," in *Signal Processing & Its Applications, 2009. CSPA 2009. 5th International Colloquium on*, pp. 238–242, IEEE, 2009.

[5] J. Pu, B. Zheng, J. K. Leader, and D. Gur, "An ellipse-fitting based method for efficient registration of breast masses on two mammographic views," *Medical physics*, vol. 35, no. 2, pp. 487–494, 2008.

[6] S. Xiaofang and G. Wencheng, "A sun spot center orientation method based on ellipse fitting in the application of cpv solar tracker," in *Intelligent System Design and Engineering Application (ISDEA), 2010 International Conference on*, vol. 1, pp. 175–178, IEEE, 2010.

[7] C. Wong, S. Lin, T. Ren, and N. Kwok, "A survey on ellipse detection methods," in *Industrial Electronics (ISIE), 2012 IEEE International Symposium on*, pp. 1105–1110, IEEE, 2012.

[8] A. Fitzgibbon, M. Pilu, and R. B. Fisher, "Direct least square fitting of ellipses," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 21, no. 5, pp. 476–480, 1999.

[9] E. S. Maini, "Enhanced direct least square fitting of ellipses," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 20, no. 06, pp. 939–953, 2006.

[10] P. L. Rosin, "A note on the least squares fitting of ellipses," *Pattern Recognition Letters*, vol. 14, no. 10, pp. 799–808, 1993.

[11] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, no. 1, pp. 11–15, 1972.

[12] H. Yuen, J. Princen, J. Illingworth, and J. Kittler, "Comparative study of hough transform methods for circle finding," *Image and vision computing*, vol. 8, no. 1, pp. 71–77, 1990.

[13] D. H. Ballard, "Generalizing the hough transform to detect arbitrary shapes," *Pattern recognition*, vol. 13, no. 2, pp. 111–122, 1981.

[14] J. Illingworth and J. Kittler, "A survey of the hough transform," *Computer vision, graphics, and image processing*, vol. 44, no. 1, pp. 87–116, 1988.

[15] R. C. Bolles and M. A. Fischler, "A ransac-based approach to model fitting and its application to finding cylinders in range data.," in *IJCAI*, vol. 1981, pp. 637–643, 1981.

[16] P.-Y. Yin, "A new circle/ellipse detector using genetic algorithms," *Pattern Recognition Letters*, vol. 20, no. 7, pp. 731–740, 1999.

[17] H. I. Cakir, B. Benligiray, and C. Topal, "Combining feature-based and model-based approaches for robust ellipse detection," in *Signal Processing Conference (EUSIPCO), 2016 24th European*, pp. 2430–2434, IEEE, 2016.

[18] Y. Qiao and S. Ong, "Arc-based evaluation and detection of ellipses," *Pattern recognition*, vol. 40, no. 7, pp. 1990–2003, 2007.

[19] F. Mai, Y. Hung, H. Zhong, and W. Sze, "A hierarchical approach for fast and robust ellipse extraction," *Pattern Recognition*, vol. 41, no. 8, pp. 2512–2524, 2008.

[20] W. Yuan and S. Zhang, "Fast ellipse detection and automatic marking in planar target image sequences.," *JCP*, vol. 9, no. 10, pp. 2379–2386, 2014.

[21] Z. Shengnan, Y. Shuang, N. Lianqiang, and Y. Weiqi, "A fast ellipse detection method in planar target image," in *Digital Manufacturing and Automation (ICDMA), 2012 Third International Conference on*, pp. 42–45, IEEE, 2012.

[22] D. Maio and D. Maltoni, "Real-time face location on gray-scale static images," *Pattern Recognition*, vol. 33, no. 9, pp. 1525–1539, 2000.

[23] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE transactions on systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[24] B. Jähne, "Digital image processing," 2002.

[25] M. Bôcher, *Plane analytic geometry: with introductory chapters on the differential calculus*. H. Holt, 1915.

[26] R. A. McLaughlin, "Randomized hough transform: improved ellipse detection with comparison," *Pattern Recognition Letters*, vol. 19, no. 3, pp. 299–305, 1998.

[27] C. A. Basca, M. Talos, and R. Brad, "Randomized hough transform for ellipse detection with result clustering," in *Computer as a Tool, 2005. EUROCON 2005. The International Conference on*, vol. 2, pp. 1397–1400, IEEE, 2005.

[28] L. Xu, E. Oja, and P. Kultanen, "A new curve detection method: randomized hough transform (rht)," *Pattern recognition letters*, vol. 11, no. 5, pp. 331–338, 1990.

[29] Y. Xie and Q. Ji, "A new efficient ellipse detection method," in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 2, pp. 957–960, IEEE, 2002.

[30] J. Canny, "A computational approach to edge detection," *IEEE Transactions on pattern analysis and machine intelligence*, no. 6, pp. 679–698, 1986.

[31] C. Cortes and V. Vapnik, "Support vector machine," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[32] C.-W. Hsu, C.-C. Chang, C.-J. Lin, *et al.*, "A practical guide to support vector classification," 2003.

[33] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of machine learning research*, vol. 3, no. Mar, pp. 1157–1182, 2003.

[34] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Computers & Electrical Engineering*, vol. 40, no. 1, pp. 16–28, 2014.

[35] P. Juszczak, D. Tax, and R. P. Duin, "Feature scaling in support vector data description," in *Proc. ASCI*, pp. 95–102, Citeseer, 2002.

[36] G. Forman, M. Scholz, and S. Rajaram, "Feature shaping for linear svm classifiers," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 299–308, ACM, 2009.

[37] Y.-W. Chen and C.-J. Lin, "Combining svms with various feature selection strategies," *Feature extraction*, pp. 315–324, 2006.