A Parallel Windowing Approach to the Hough Transform for Line Segment Detection

by
Payam Salmanpour Rahmdel

School of Science and Technology
Middlesex University London

A dissertation submitted to Middlesex University London
in partial fulfilment of the requirements for the degree of
Doctor of Philosophy

September 2013

This doctoral dissertation is dedicated to
my grandparents Mansoureh & Hasan, Forough & Masoud
my parents Farshideh & Ali
my loving wife Shayandokht

Declaration

This thesis is an account of research undertaken between November 2010 and September 2013 at The School of Science and Technology, Middlesex University London, United Kingdom.

Except where acknowledged in the customary manner, the material presented in this thesis is, to the best of my knowledge, original and has not been submitted in whole or part for a degree in any university.

Payam S. Rahmdel
September, 2013

Abstract

In the wide range of image processing and computer vision problems, line segment detection has always been among the most critical headlines. Detection of primitives such as linear features and straight edges has diverse applications in many image understanding and perception tasks. The research presented in this dissertation is a contribution to the detection of straight-line segments by identifying the location of their endpoints within a two-dimensional digital image. The proposed method is based on a unique domain-crossing approach that takes both image and parameter domain information into consideration. First, the straight-line parameters, i.e. location and orientation, have been identified using an advanced Fourier-based Hough transform. As well as producing more accurate and robust detection of straight-lines, this method has been proven to have better efficiency in terms of computational time in comparison with the standard Hough transform. Second, for each straight-line a window-of-interest is designed in the image domain and the disturbance caused by the other neighbouring segments is removed to capture the Hough transform butterfly of the target segment. In this way, for each straight-line a separate butterfly is constructed. The boundary of the butterfly wings are further smoothed and approximated by a curve fitting approach. Finally, segments endpoints were identified using butterfly boundary points and the Hough transform peak. Experimental results on synthetic and real images have shown that the proposed method enjoys a superior performance compared with the existing similar representative works.

Acknowledgement

Studying towards achieving a doctorate degree was a journey that my beloved God put me through. It was full of ups and downs, the frustration of repetitive failures and the excitement of new findings. Successfully completing this journey would have not been feasible without the presence of a number of incredible human beings.

I would like to express my heartfelt gratitude to my Director of Studies, Dr. Daming Shi who provided the vision, encouragement and advice necessary throughout this journey. No doubt, meeting with him, four years ago in South Korea, was a life-changing moment for me. He had a strong influence on my professional life. I see him as a role model of persistence and creativity. He has been a strong and supportive adviser throughout. Without his continuous and careful supervision, it would not have been possible to reach this quality of work.

I would like to thank my supervisor Prof. Richard Comley for his significant influence on forming my academic and research attitude. He always had a strong believe in my work and abilities that gave me the courage to move towards the target. I admire his punctilious comments and flawless suggestions throughout my studies and more importantly on this thesis. He is a model of a modest academic and I shall always remember him as a gentleman with huge respect.

I would like to acknowledge the Middlesex University Research Committee for the recognition of my research proposal by awarding me the Research Student Bursary, the fully funded scholarship that provided the financial support for this research.

Alongside the research towards my PhD was the invaluable experience of sharing the knowledge with others by teaching. I would like to thank Prof. Mehmet Karamanoglu who trusted and welcomed me to his team at the Design Engineering and Mathematics Department. I am thankful for his support and inspiration that opened a lot of opportunities for me during my

studies.

I am grateful to all my friends, fellow PhD students, and other researchers in our research centre in the School of Science and Technology for creating a fantastic environment in which to work and enjoy. I am thankful to Juan Martinez, Application Support Engineer at The MathWorks, for his great help and suggestions during the MATLAB implementation and programming of the proposed algorithm. I am also grateful to Leonard Miraziz for his continuous IT and technical support.

I would like to thank my uncle Masoud Rahmdel who has always been a symbol of enthusiasm and self-motivation for me. He has been a powerful support and truly a reliable source to consult with since my very early steps towards higher education, when I started to learn English. I am thankful for his positive and ever encouraging quotes.

No words can ever describe my gratitude to my wonderful family for their lifelong gracious love. "Thank" is such a naive word to express my highest appreciation and gratefulness to my parents, Farshideh and Ali for devoting their lives to me and my brothers' happiness. They are a symbol of pure love and mutual respect. I shall always be grateful to my parents for letting me openly choose my way and supporting me with the best that any parents could ever do. I would like to take this opportunity and thank my two amazing brothers, Navid and Vedad, for all the fun and fights we had together.

I would like to extend my deepest appreciation to my dear grandfathers Haj Hasan Rahmdel and Dr. Masoud Ebrahimzadeh, and to my loving grandmothers Mansoureh Mesgarian and Forough Nourbakhsh for all the invaluable lessons they taught me. I am thoroughly indebted to my grandmother Mansoureh's prayers, whose dream was to see my success. Losing her right in the middle of this journey was so unfortunate, but I hope I have finally made her dream come true, and may she rest in peace.

It is worth mentioning my great grandfather Haj Seyyed Sadegh Nour-

bakhsh. Unfortunately, I missed the chance of meeting him, but tales about his selfless attitude and modesty told by other people have always fascinated and inspired me. He was a self-motivated and enthusiastic young man who, despite the loss of his parents managed to become one the wealthiest and well-known traders in our province, Mazandaran, Iran.

Finally, and of course importantly, to my beloved wife Shayandokht who patiently shared every moment of this journey with me. I could never ever have asked for a better companion. She motivated me with her incredible support and strong believe on my abilities for the last eight years, since the day I firmly decided to leave basketball as my main profession and begin a remarkable journey of education. She is wonderfully patient and calm but motivating and enthusiastic, and more importantly a role model of a courageous and successful woman. I am thankful to her for preparing a fantastic mental and physical environment for my studies.

May God bless you all

Payam S. Rahmdel
London, September 2013

List of Publications

Journal articles

1. Payam S. Rahmdel, Daming Shi, and Richard Comley, "Comment on: Collinear Segment Detection Using HT Neighbourhoods," *IEEE Transactions on Image Processing.* (Accepted for publication)

2. Payam S. Rahmdel, Daming Shi, and Richard Comley, "Sinogram separation approach to generalized interpolated Fourier transform for line segmentation," *IET Image Processing.* (under review)

3. Daming Shi, Junbin Gao, Payam S. Rahmdel, Michael Antolovich, and Tony Clark, "UND: Unite-and-Divide Method in Fourier and Radon Domains for Line Segment Detection," *IEEE Transactions on Image Processing*, vol.22, no.6, pp.2500-2505, June 2013.

Conference proceedings

1. Payam S. Rahmdel, Daming Shi, and Richard Comley, "Radon Sinogram Decomposition for Line Segmentation," in *Proceedings of IEEE 56th International Midwest Symposium on Circuits and Systems*, Columbus, Ohio, USA, August 2013.

2. Payam S. Rahmdel, Daming Shi, and Richard Comley, "Lane Detection Using Fourier-Based Line Detector," in *Proceedings of IEEE 56th International Midwest Symposium on Circuits and Systems*, Columbus, Ohio, USA, August 2013.

Honours and awards

"Honourable Mention Award" for the quality of the paper and excellent presentation from *IEEE 56th International Midwest Symposium on Circuits and Systems*, 4-7 August 2013, Columbus, Ohio, United States of America.

Article: "Radon Sinogram Decomposition for Line Segmentation"

Contents

List of Figures

Note

The axes for all images are in units of pixels.

The axes for all Hough spaces (butterflies) are in units of 0.5 pixels for distance and degrees for angle.

List of Tables

Chapter 1

Introduction

1.1 Motivation

Line segments are everywhere around us. Wherever you look there is an object that can be easily distinguished from its surrounding environment by its sharp and straight edges. Your laptop computer, mobile phone, the room you are sitting in and the thesis that you are reading are all made of a number of line segments with a certain length. In fact, most man-made objects can be recognised by straight edges and boarders. In terms of our visual perception, the length, location and angle of these line segments can vary with respect to our point of view of the object.

In the field of image processing and computer vision, straight-line detection and segmentation are the techniques that allow a machine or intelligent system to extract linear features and structures from a digital image or sequence of images. Extracted features, e.g. straight-lines and segments, can later be used either by human users or intelligent computer systems to make sense or decisions based on their initial desire.

This research is an attempt to highlight the problems and obstacles that have challenged researchers in the past and recent years and to propose a practical solution to alleviate some of the shortcomings.

To begin, a brief review of the vast scope of research in the area is presented, starting from image processing to computer vision, showing how the research presented in this thesis fits into this diversity.

1.2 Image processing and computer vision in action

It has always been the researchers' aim to make a machine that can see, analyse, understand and respond to its surrounding environment based on its visual perception. Computer vision and image understanding is the science and technology that tries to fulfil these needs with the help of numerous mathematical, statistical, geometrical, as well as other analytical techniques and tools, to extract information from digital images and use that information to solve the problems for which the system has been designed.

For a typical image processing or computer vision system, images play the role of input data, streaming through the vision sensor, e.g. digital camera or scanner. For a computer system, a digital image can be seen as a two-dimensional (2D) discrete function $f(x, y)$ or an array of digits, where x and y represent rows and columns respectively.

Diversity of applications such as in medicine, astronomy, law enforcement, and the film industry. has opened an enormous ground for researchers, scientists, and engineers to work in. The continuum of research from image processing to computer vision can be broken down into three major levels of process:

1. **Low-level process.** This is a very early stage in processing an image and usually is referred to as a pre-processing step. In a typical low-level stage the input data is a digital image that goes through a series of operations that result in an output image. In a system that is designed for such operations both input and output are images.

Examples: Noise removal, image sharpening/blurring, image enhancement, image restoration, and edge detection.

2. **Mid-level process.** In a mid-level process, the image that has gone through the low-level operations and been prepared according to the requirements of the task will be used as the input for further investigations. In this stage, the main interest is to find attributes, correlations, similarities and special features based on the purpose of the algorithm. Thus the input is an image and the output will be the attributes.

   Examples: Object detection, image segmentation, *line detection and segmentation*, face detection, image registration, feature extraction, and geometrical or arbitrary shape detection.

3. **High-level process.** This is where the perception and understanding comes into the play. The attributes and features that have been produced in a mid-level process will be gathered, categorised, and utilised as input data for the system. The expected output is a level of understanding that can be used for sense and decision making.

   Examples: Face recognition, gesture recognition, scene understanding, autonomous vehicle or robot navigation, text understanding, and human-computer interaction.

Note that, here the level of process does not represent the level of complexity of the algorithm but just the level that the operation is being performed. Meaning that, a low-level process such as image enhancement can be as complex as a perception task operated in a high-level stage.

As mentioned above, extracting linear features such as straight-lines and segments can be considered as a mid-level process. Accordingly, the type of operations conducted in this research and the algorithm proposed in this thesis can be fitted into the second category of the processing levels.

1.3 Line segmentation problem statement

Detection of line profiles and segments is one of the most important and, in fact, primitive tasks in image analysis and has its application in biomedical image processing, path detection for robot navigation, lane marker detection, and road and stream detection in satellite images. For more than thirty years researchers have contributed to build more robust and accurate algorithms with faster performance. In fact, there is always a trade-off between these two parameters, i.e. accuracy and speed, as they are two sides of the same coin.

The Hough transform (HT) [1] and its extensions are among the most well-known techniques for detection of straight lines in a digital image. It has shown great performance in dealing with noisy and cluttered images. One of the recent improvements on the standard HT is a unique Fourier-based approach [2] that results in higher accuracy of line detection in comparison with the representative state-of-the-art techniques. It produces a high-resolution parameter space through a, so called, multi-layer fractional Fourier transform (MLFRFT). Multiple instances of fractional Fourier transform of the same image provide more frequency samples that lead to higher accuracy and better performance. However this technique is incapable of detecting line segments and their endpoints, as it can identify only the location and orientation of the straight lines passing through the segments. That has motivated the author of this thesis to build upon the foundation of the MLFRFT approach and extend its capabilities from line detection to an accurate and robust line segmentation technique.

Primary research question:

How does a system detect segment endpoints of the straight-lines captured via the MLFRFT-based HT?

Secondary research questions:

1. How does the MLFRFT perform if implemented in parallel?
2. How fast is the process of line detection using a MLFRFT-based HT?
3. How accurately can the boundary sinusoids of a HT butterfly be approximated?
4. How can the intersections of butterfly boundary points and the HT peaks that localise the endpoints of a line segment be found?
5. How can overlapping butterflies be isolated when they appear in separate Hough spaces?
6. How can the disturbance caused by non-collinear segments intersecting with the target segments be eliminated?
7. Can an optimum solution to the line segmentation problem be found?

1.4 Thesis layout

This thesis is structured as follows:

- **Chapter 2** provides an extensive literature survey by reviewing numerous attempts to solve the line segment detection problem, with a special focus on HT-based techniques. The main directions of the HT research have been identified and the main problems have been highlighted. In the second part of this chapter three major categories of the existing line segmentation approaches, i.e. bottom-up, top-down, and domain-crossing, have been introduced and their specifications have been evaluated. This provides an appropriate insight to the advantages and disadvantages of each approach.

- **Chapter 3** starts with the fundamental idea of Fourier-based HT techniques and the concept of the central-slice theorem in the frequency domain. It highlights the problem of zero-padding and introduces the multi-layer fractional Fourier transform (MLFRFT) as a solution to that problem. The idea of having a non-integer frequency grid has also been explained in detail using figures and descriptions. Parallel implementation of the MLFRFT is investigated in this chapter together with a comparison of the computational time required for line detection using the standard HT and the MLFRFT-based HT. Finally a potential real-world application of the MLFRFT-based HT in lane marker detection is introduced at the end of the chapter.

- **Chapter 4** contains the main contributions of this research. It starts from the concept of HT butterfly formation and the relationship between its boundary sinusoidal curves and each of the endpoints of a line segment. A unique line segmentation method based on butterfly boundary points (LSBB) has been proposed that has two main characteristics: 1) the use of Robust Least Squares curve fitting for precise approximation of boundary sinusoids; 2) using the intersection of three points and the centroid of a triangle to localise the endpoint. In the second part of this chapter, the algorithm has been extended to take into account multiple line segments. Using a parallel windowing approach (PWA), the complex and overlapping butterflies have been isolated to a number of single butterflies by crossing over to the image domain and applying a number of windows-of-interest. The effect of non-collinear segments on the butterfly of a target segment has been removed with the help of the disturbance elimination algorithm proposed in this chapter.

- **Chapter 5** includes the results of experiments conducted to evaluate the performance of the proposed algorithm in comparison with the

representative approaches. Detection accuracy and robustness to noise have been studied and compared using ground-truth[1] data. These experiments were performed on both synthetic and real-word images reported in the literature. Observations and findings are discussed at the end of the chapter.

- **Chapter 6** concludes the thesis by giving a summary of the work and highlighting the main contributions of this research. It also underlines the limitations of the method and provides a number of suggestions for future investigation.

[1]For the purposes of this thesis, the term Ground Truth refers to an actual datum (or pixel) in an image (e.g. end point of a line) as established by my direct observation and measurement.

Chapter 2

Literature Review

2.1 Introduction

Extraction of straight-line segments from a digital image has always been a challenging and yet critical pre-processing stage for researchers in the field of image understanding and computer vision. As a mid-level image processing task, line segmentation has extensive applications in object recognition [3, 4, 5], shape detection [6, 7, 8], power line detection [9, 10, 11], image compression [12] and road or lane detection [13, 14, 15, 16]. Despite the significant progress in the past two decades, there is still a rising demand for methods that are swift in computation and precise in segment detection. In fact, these criteria, i.e. speed and accuracy, are two sides of one coin in practical applications such as medical [17], astronomical [18], surveillance [19], and target tracking [20].

Numerous attempts from fundamentally distinctive views have been made to tackle the problem. Representative works include line detection methods based on the Hough transform [21, 1], subspace [22], wavelet transform [23], ridgelet transform [24], chain code detection [25], hidden Markov models [26], feature-adapted beamlet transform [27], and machine learning approaches such as the Bayesian approach [28], Bayesian Ying-Yang harmony learning

[29] and Principal Component Analysis (PCA) [30].

Among this diversity one method has captured the most attention and established a strong foundation for many linear and curvilinear object detection algorithms over the past fifty years.

Proposed by Paul Hough [21] and further refined by Richard Duda and Peter Hart [1], the Hough transform (HT) is indeed one of the most popular methods for the detection of linear and curvilinear structures in 2-dimensional (2D) images. It has demonstrated a robust behaviour to variant noise and degraded environment. The HT has been used for many object recognition applications such as the detection of straight-lines [31, 2, 32], circles [6, 33], ellipses [4], triangles [34], rectangles [7] and predefined arbitrary shapes [8]. As for the purpose of this research, which is to propose a line segmentation algorithm using a unique HT-based approach, only the first application will be considered.

2.2 Hough transform for line detection

Line detection plays a key role in object recognition by detecting straight edges. Therefore, much attention has been paid to straight-line detection using the HT. In general, the HT maps every pixel in a binary image from its 2D Cartesian coordinates (x, y) to a new 2D coordinate space (ρ, θ) using the mapping function

$$\rho = x \cos\theta + y \sin\theta \tag{2.1}$$

The new coordinate space is called Hough space, also known as parameter space, and the process is called mapping from image or feature space to data or parameter space. The Hough space consists of θ as the angle of the normal from the origin perpendicular to a straight-line and ρ as the length of this normal. θ and ρ are typically restricted to $[0, \pi)$ and $[-\frac{R}{2}, \frac{R}{2}]$ respectively, where $R = \sqrt{m^2 + n^2}$ for an $(m \times n)$ pixels image.

For each pixel (x, y) in the image domain that goes through the mapping

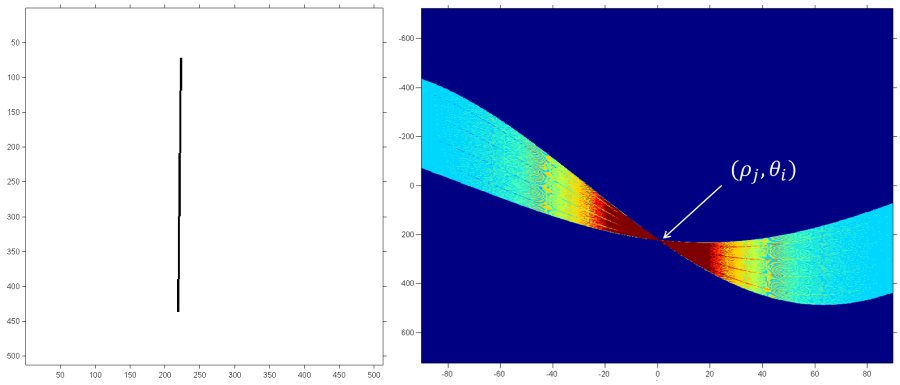
procedure, the HT calculates the value of ρ for every discretised value of θ using Equation 2.1. Each iteration of Equation 2.1 generates a vote for a particular cell in the (ρ, θ) space. An accumulator is allocated to count the number of votes each cell receives. As a result, each pixel in the image domain generates a sinusoidal curve in Hough space. Accordingly, the combination of a number of collinear pixels in the image domain produces a number of sinusoidal curves in the Hough space that have a common intersection point at (ρ_j, θ_i). Detecting this common intersection point, i.e. the HT peak, we can eventually detect the straight-line that represents that peak. A peak detection needs to be performed to separate the potential candidates.

Therefore, the line detection problem in image space turns into a much simpler task of peak detection in the Hough transform parameter space. It is because of this characteristic that the HT shows a reasonable robustness to noise and complex backgrounds. The HT process explained above is summarised in Algorithm 1.

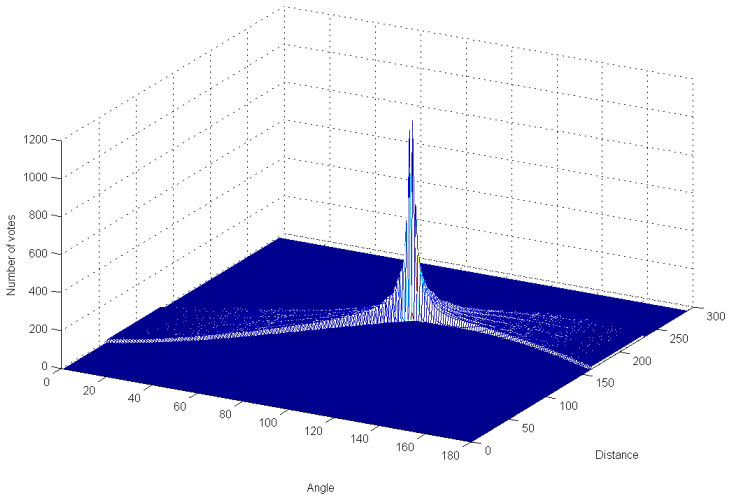
Algorithm 1 Standard Hough transform (SHT).

Require: : $I(x, y)$ /* A binary image
Require: : δ /* Specify the resolution in θ axis, i.e, discretisation step
1: $A \leftarrow 0$ /* Initialising the accumulator array
2: **for all** feature points in $I(x, y)$ **do**
3: **for all** θ_i , $i = 0 \rightarrow \pi$ **do** /* given the δ
4: $\rho_j \leftarrow x \cos\theta_i + y \sin\theta_i$
5: $A(\rho_j, \theta_i) \leftarrow A(\rho_j, \theta_i) + 1$
6: **end for**
7: **end for**

Figure 2.1 shows a single straight-line segment and a 2D and 3D representations of its Hough space where the peak is clearly distinguishable. The combination of the sinusoids and the common intersection point (ρ_j, θ_i), i.e. the HT peak, is highlighted by the arrow in Figure 2.1(b).



(a) (b)



(c)

Figure 2.1: 2D and 3D representation of the HT of a line segment. (a) 512×512 image of a single straight-line segment. (b) 2D HT representation with a distinguishable peak at (ρ_j, θ_i). (c) 3D HT representation to highlight the peak; the angle axis is in unit of degree and the distance axis is in unit of 0.5 pixels.

2.2.1 Research in Hough transform

Three major questions arise in respect of the HT algorithm explained above, which are:

1. How to determine optimal $\rho - \theta$ resolution?
2. How to enhance the computational efficiency and memory storage?
3. How to isolate and detect the HT peaks accurately with minimal false detection?

Enhancements on $\rho - \theta$ resolution

The first question is related to discretisation of the Hough space. It is important for both the accuracy and computational efficiency to determine a sufficient resolution for discretising the Hough space. The higher the resolution the slower the process. Numerous works have tackled the problem such as using gradient direction [35], interpolation in the Hough space [36], dynamically quantised spaces [37], trial and error [36], sensitivity function [38], using information from the discretisation of image space [39], error propagation [40], and weight of accumulation [41]. Instead of using uniform discretisation, Duan et al. [42] introduced a non-uniform discretisation of the Hough space using angular and distance dependency of the minimum $\rho - \theta$ spacing of neighbouring lines.

Enhancements on computational time and memory requirements

A large number of distinct variations of the HT were introduced to advance its performance, mainly focusing on computational complexity and memory efficiency of the standard Hough transform[43, 33, 44, 45]. Representative works that appear in most of the related literature include Stephens's probabilistic Hough transform (PHT) [46], progressive probabilistic Hough transform

(PPHT) [47, 48, 49], randomized Hough transform (RHT) [50, 51], hierarchical Hough transform (HHT) [52], elliptical Gaussian kernel-based Hough transform (KHT) [45], and regularized Hough transform [31].

Stevens proposed the PHT [46] as a form of likelihood function and showed a strong relationship between the HT and maximum likelihood method [53, 54]. Quite different from the other variations of the HT that operate on a discrete space, Stevens's PHT is defined as a continuous mathematical function. Despite its accuracy PHT does not suggest a significant improvement on the computational cost.

The RHT introduced a fast algorithm by its unique mapping strategy. Unlike the SHT that maps each pixel to a number of curves, i.e. *one-to-many* approach, the RHT randomly selects pairs of collinear pixels that have a higher probability of being part of a line and maps them into one (ρ, θ) coordinates in the Hough space, i.e. *many-to-one* approach. The RHT has lower computational cost, smaller storage requirement and higher robustness to noise than the traditional HT, however it cannot provide an effective solution to optimally select a subset of pixels to work with. This is particularly important when the number of edge pixels increases.

The progressive probabilistic HT (PPHT) was introduced in [48] as an improvement over Kiryati et al.'s probabilistic HT [55], which similar to the RHT is based on random selection of a small subset of edge pixels. The PPHT randomly selects pixels from the subset and updates the corresponding accumulator cells. When a cell reaches a given threshold its corresponding corridor in image space will be searched for the longest possible segment. The PPHT is fast and suitable for real-time applications, however not a proper choice for short segments. Galamhos et al. [56] extended this work by taking gradient information into consideration to control the voting process that enhanced both the accuracy and the efficiency of the PPHT.

The HHT suggested a method that minimises the likelihood of irrelevant pixels selection by highlighting the actual distribution of feature points along

a line. It splits the image into a number of sub-images through a quad-tree decomposition process, referred to as pyramid structure, and performs the HT on every low-level sub-image to find the line segments. Afterwards, the identified line segments will be propagated upwards to be grouped based on their hierarchy. Due to independence of every sub-image the algorithm can be implemented in parallel. However, the HHT does not suggest an adaptive parameter tuning and the constraints are subject to being predefined.

Enhancements on peak detection

A peak isolation algorithm can simply detect the local maxima in an $N \times N$ neighbourhood in the Hough space [57], given an odd number for N. But determining the optimal value of N can be problematic. The larger the value of N the higher the probability of missing adjacent lines, and the smaller the value of N the higher the probability of repetitious detection. A global peak detection can be used instead of a local neighbourhood approach. In [52] Princen et al. used an iterative global peak detection approach where the globally highest peak is detected first and its corresponding feature points get eliminated. A HT is then applied and the process repeats iteratively until all the peaks are found. In [58] an iterative "identify and remove algorithm" was employed where each peak is removed from the Hough space after it is identified. There are two cases: 1) when we have a desired number of lines to be detected, the algorithm terminates after a certain number of iterations; 2) when the number of expected lines is unknown so that iterations will continue for every single peak above the threshold. Although the global peak detection is more accurate and robust than the local method, its iterative accumulation process is computationally expensive and it is only suitable for small-size images. Peak detection using self-organising maps were suggested to reduce the memory requirement of the HT [59]. More recent works focused on generating solid and distinguishable peaks and the ways to search and identify the true peaks with minimal false detection [31, 44, 45, 60]. These have major

problems, in particular memory inefficiency and computational complexity. Detection accuracy is directly proportional to the memory capacity. That means in order to have an accumulator array with a high resolution in ρ and θ coordinates, a higher capacity of memory storage is required.

HT butterfly analysis

One significant direction in HT research is *HT butterfly analysis* that considers both the peak and the area surrounding the peak. This is particularly important for segment endpoint detection. The term butterfly refers to the shape of the peak and its associated sinusoidal curves. As shown in Figure 2.1(b), the combination of sinusoids that represent the line segment form a butterfly shape around the peak. In fact, such butterflies contain valuable information about a segment's length, endpoint and thickness. Because every single pixel of a line contributes to create its HT butterfly in a one to one mapping process (refer to Equation 2.1), the resulting butterfly is capable of providing highly accurate line-segment parameters.

Efforts have been made to parameterise the HT butterfly. Representative works include line segment descriptors [61, 62], multi-segmentation [63, 64], peak enhancement using butterfly features [32, 65], HT neighbourhood [66, 67], butterfly self-similarity [68], and butterfly symmetry [69]. Butterfly analysis for segment endpoint extraction forms a major part of the methodology presented in this thesis and will be discussed in more detail in Chapter 4.

2.2.2 Hough transform limitations

In general, there are three main problems associated with the existing HT-based methods:

1. **Computational complexity**. Despite the efforts to improve the efficiency of the HT, its heavy computational cost is still a major concern

in real-time applications.

2. **Edge detection**. The HT is applicable only to binary images, hence accurate edge detection is an important prerequisite. In addition to imposing extra computational burden, edge detection may increase the false detection error; when the noise-level is high, some of the true feature points in an image can be ignored as noisy pixels and also some false points may be seen as true feature points.

3. **Detection of line segments**. Despite the fact that the HT is an accurate and robust technique for line profile extraction in noisy or cluttered images, it is incapable of specifying the endpoints for a line. It can identify only line-segments that pass through the entire image. Therefore segmentation procedures have to be adapted in order for the HT to identify the endpoints of a segment.

A number of works addressed the edge detection problem of the HT using Fourier-based HT techniques [70, 2, 71]. These methods rely on a unique mathematical relationship between the Hough and the forward Radon transform and will shape a significant portion of the algorithm proposed in this research.

2.2.3 The Radon transform

Equivalently and due to the mathematical identity, the HT can be viewed as special case of the forward Radon transform (RT) [72]. The Radon transform is a process of iterative mapping of a 2D function onto its projections [73], (translated article in [74]). Given a 2D function or image $f(x, y)$, its 1D projection is obtained by integrations along lines perpendicular to θ, where θ is the projection angle relative to the x-axis and $0 < \theta \leqslant \pi$. A complete set of projections can be obtained by applying different projection angles varying from 0 to π. Combination of 1D projections forms a complete parameter

space called as sinogram, also known as the Radon space. As a result of this iterative process each point in image space maps to a sinusoidal curve in a new parameter space. This is similar to the process of mapping each feature point from Cartesian space to Hough space in the standard Hough transform (SHT).

Note that the integrations are performed over lines parallel to the y' axis in a (x', y') coordinate system that is rotated at angel θ using rotation operator

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{2.2}$$

Thus for a point (x, y) that is located at a distance ρ along the x' axis Equation 2.1 holds, and can also be written in vector notation:

$$\rho = r\cos\alpha\cos\theta + r\sin\alpha\sin\theta = r\cos(\alpha - \theta) = \mathbf{r}.\hat{\mathbf{n}} \tag{2.3}$$

where $\mathbf{r} = (x, y) = |\mathbf{r}|\angle\alpha = [r\cos\alpha, r\sin\alpha]$ is the position vector of a point in rectangular coordinates and $\hat{\mathbf{n}} = 1\angle\theta = [\cos\theta, \sin\theta]$ is the unit vector perpendicular to the projection angle [70].

In theory, the RT obtains the projections by performing line integrals along the variant angles of θ using an equation

$$\mathbf{R}_\theta[f(x,y)] = \lambda(\rho,\theta) = \int\int f(x,y)\ \delta(\rho - x\cos\theta - y\sin\theta)\,\mathrm{d}x\mathrm{d}y \tag{2.4}$$

where $\lambda(\rho, \theta)$ is a one-dimensional Radon projection of $f(x, y)$ at angle θ, $\mathbf{R}$ is the Radon transform operator, and δ is the Dirac delta function.

In practice, the RT can be computed efficiently using the fast implementation of the Fourier transform to reduce the computational complexity. That is based on a Fourier slice theorem and will be discussed in Chapter. 3.

Relationship between the RT and HT

Despite the identity of the final result, these two linear transformations, i.e. the HT and the RT, look at the same problem from slightly different perspectives. The HT is the question of how a pixel (x, y) in image space is mapped to a sinusoid in parameter space using Equation 2.1, i.e. the *writing paradigm*. Having x and y as constants, vary θ and compute ρ. The RT is the question of how a data point (ρ, θ) in parameter space is obtained from line integrals in image space using Equation 2.4, i.e. the *reading paradigm*, i.e. having ρ and θ, apply line integrals (compute the projections) along the corresponding line. Therefore, the Hough transform can be seen as a discretised version of the continuous Radon transform [75, 76].

Aim of the proposed method

The method introduced in this thesis aims to focus on the second and third problems in Section 2.2.2 without adding to current computational complexity. To address the problem of binarisation and edge detection an advanced Fourier-based HT has been adapted for use in this research. A full discussion is given in Chapter 3. To extract the segment endpoints a unique image-domain windowing approach has been introduced. It requires crossing over from Hough space to image space and applying an image domain filtering in the direction of a detected line. Butterfly boundary analysis has been adapted to isolated HT peaks. Chapter 4 explains the algorithm.

As mentioned earlier, the HT is a powerful tool to detect straight lines but not line segments. An additional segmentation procedure needs to be performed to extract the segments from the detected lines. To classify the existing segmentation methods an extensive survey of various approaches is given in the following sections.

2.3 Segmentation approaches

Popular line segmentation methods used in practical applications can be classified into three major groups, namely, bottom-up, top-down, and domain-crossing approaches. The bottom-up approach starts with single pixels, which grow to segments; whereas the top-down approach extracts straight lines followed by segmentation. Domain-crossing methods take into account both local features, from a bottom-up, and global features, from a top-down approach. In the following, different characteristics of these approaches will be surveyed.

2.3.1 Bottom-up approach

A bottom-up approach, also known as local approach, typically begins from the pixel level and the line grows pixel by pixel to reach the requirement of a true line segment defined by the algorithm. These approaches mainly use gradient information to draw the line. State-of-the-art methods include Gioi et al.'s line segment detector (LSD) [77], Akinlar and Topal's Edge Drawing (EDLines) [78], and Yang et al.'s two-orthogonal direction image scanning (TODIS) [79]. Other works use a small matrix of eigenvalues [80, 81].

One of the first bottom-up frameworks was introduced by Nevatia and Ramesh Babu [82]. The algorithm starts with a convolution-based edge detection followed by a line thinning and threshold. Afterwards, a linear approximation is used to link the edge points based on their gradient orientations. Khan et al. extended the concept using a connected component algorithm (CCA) [83]. A CCA groups together adjacent feature points with similar gradient orientation into line support regions. The algorithm was simple and fast, and hence later became the core of some of the well-known bottom-up algorithms such as [84] and [77].

The LSD utilises Burns et al.'s iterative *region-growing* process [84]. Using image gradient magnitude and angle, each region starts from a pixel by

setting the region's angle to the pixel's gradient direction. In the next iteration the algorithm compares the gradient direction of adjacent pixels (in an eight-pixel neighbourhood) with the region angle. If any neighbouring pixel shares a similar angle within a certain tolerance it will be added to the growing region. Pixels belonging to a particular region will be labelled to avoid revisiting. In the second phase of the LSD, Desolneux et al.'s [85] approach was used to validate the result.

LSD has proven to be a breakthrough in line segmentation by outperforming other representative works such as Etemadi [86], Burns [84], the progressive probabilistic Hough transform [49], and Desolneux [85]. It is fast as it can process a (512×512) grayscale natural image in less than 0.30 seconds using an Apple PowerBook G4 1.5 GHz [77]. However, there are two major problems involved with the LSD's performance: 1) it is highly sensitive to noise that misleads the region-growing process in connecting the true segments; 2) it loses accuracy when facing dense intersecting straight lines.

Yang et al. partially addressed the LSD's shortcomings by introducing *two-orthogonal direction image scanning* (TODIS) [79] but paid the price of computational time. Unlike the LSD, TODIS works with binary images, thus edge detection has to be applied in the first place. An image is examined in both horizontal and vertical directions. A multi-scale scanning approach was taken to label each candidate line segment. There are some issues due to the multi-scale nature of TODIS. In smaller scales a long segment will mistakenly appear as a number of short segments. Similarly, in larger scales a number of collinear short segments may appear as one continuous long segment. In addition, unlike the LSD, in TODIS each pixel may be visited more than once and that has a negative effect on the computational time.

The EDLines [78] suggests a faster algorithm for line segment detection. In terms of accuracy in segmentation, the results of EDLines are similar to the LSD's; however, the EDLines processes the same image ten times faster

than the LSD. That makes the EDLines a perfect candidate for real-time applications. In their algorithm Akinlar and Topal used the concept of *Edge Drawing* (ED) [87, 88] to produce an accurate edge map. The underlying idea is to use image gradient information to connect the edge pixels that belong to the same segment. Unlike the other edge detection approaches such as the Canny edge detector [89] that generates a binary edge map consisting of arbitrary pixels, ED results in a number of related and joint edge pixels in a shape of edge segments. This pre-processing step plays a key role in EDLines' success by reducing the error caused by broken segments. However, for the next step, i.e. line segment extraction they applied a simple Least Square Line Fitting approach that causes inaccuracy in noisy situations and breaks down the long segments.

In general, bottom-up approaches are computationally simple and easy to implement. Plus, their local nature is well-suited for taking short line segments into account. However, their local characteristic fails to maintain the robustness in challenging situations such as when line segments intersect or when there are rather long segments in an image. Due to the effect of noise and image resolution such long segments appear as a series of disconnected short segments. This sensitive behaviour in response to noise appears to be the main drawback of the bottom-up approaches. Using global information can help solve such problems. The top-down approach looks at the problem from a rather different perspective.

2.3.2 Top-down approach

In the top-down approach, also known as the global approach, the true straight-lines are firstly extracted before they are broken down into segments. The Hough transform (HT) is one of the most well-known top-down techniques for detection of line profiles [21, 1].

Most of the proposed HT-based line segmentation methods use the parameter space information during and after the HT voting process. For

instance, the connective HT (CHT) utilises a probabilistic approach to investigate the connectively of the feature points [90]. After applying the HT, a fixation point is selected using the information obtained during the accumulation to vote for two 1D accumulators. The CHT suggest a faster computation than the standard HT. To capture segment coordinates using the fast Hough transform (FHT), Guil et al. [91] kept aside those feature points that collaborate during the voting process. By arranging the image points belonging to the same straight-line in a respective order, coordinates of the segment endpoint can be captured by selecting the points that are further away from each other. Nevertheless, the method is not perfectly designed for line segments with a negative slope smaller than one or line segments that are collinear.

The progressive probabilistic Hough transform (PPHT) suggested a fast and efficient algorithm in [47, 48, 49]. However, there is a trade-off between the speed and accuracy. The PPHT outperforms the standard HT in terms of speed but results in lower accuracy and a large number of missing lines (i.e. false negative). Moreover, it requires a large set of parameter adjustments such as threshold that have to be delicately tuned. The problem with the accuracy of the PPHT was addressed later in [41] at the expense of a high computation and memory requirement.

The extended Hough transform (EHT) optimized the traditional 2D HT with a third parameter [92]. In this 3D representation of the HT, each individual column or row of the image space is plotted to a unique dual 2D HT. Later in [93], an optimized algorithm was proposed to reduce the execution time during the voting process. However, similar to all other HT-based methods, it requires prior edge detection to enhance the linear features that brings additional computational time. In addition, edge detection may cause inaccuracy in segmentation performance by neglecting true segments due to inappropriate threshold selection.

Global searching algorithm embedded in the HT-based approaches showed

a better robustness to variant noise. Furthermore, in the case of long segments, a global view helps to detect long lines as a single continuous segment, unlike the local approaches that break them down. As a more tangible example, this is similar to looking through a window with scratched glass from a distance without noticing the blurriness caused by the scratches. However if you get closer to the glass, the scratches will become more noticeable. Let us compare the performance of the bottom-up and top-down approaches using natural images to better understand their differences. As a significant limitation, due to its global nature, the HT usually fails to detect short segments. Because short lines generate smaller clusters in the accumulator array and, therefore, it is more difficult to distinguish these small peaks from the surrounding peaks.

Bottom-up vs. top-down

Performance of the above two approaches can be better explained in Figure 2.2. The first row shows an example where the global approach outperforms the local method. The image of an arrow degraded with Gaussian noise has been tested with the EHT, as a representative of the top-down approach, and the LSD, as a representative of the bottom-up approach in Figure 2.2(b) and (c), respectively. Although not flawless, the EHT detects the segments, especially the long lines, with an acceptable accuracy. In contrast, due to the background noise the LSD fails to detect the true long segments despite the simplicity of the image. Instead, it produces a number of collinear and disjointed segments. The additive noise stops the growing region from expansion before reaching the true endpoint. However, in an image with rather complex texture, without additive noise, in the second row, i.e. Figure 2.2(d), the LSD performs better by detecting most of the segments with a greater detail, see Figure 2.2(f). In contrast, the global nature of the HT-based approach is incapable of detecting many of the short segments as shown in Figure 2.2(e). This is when the LSD outperforms the HT-based approach.

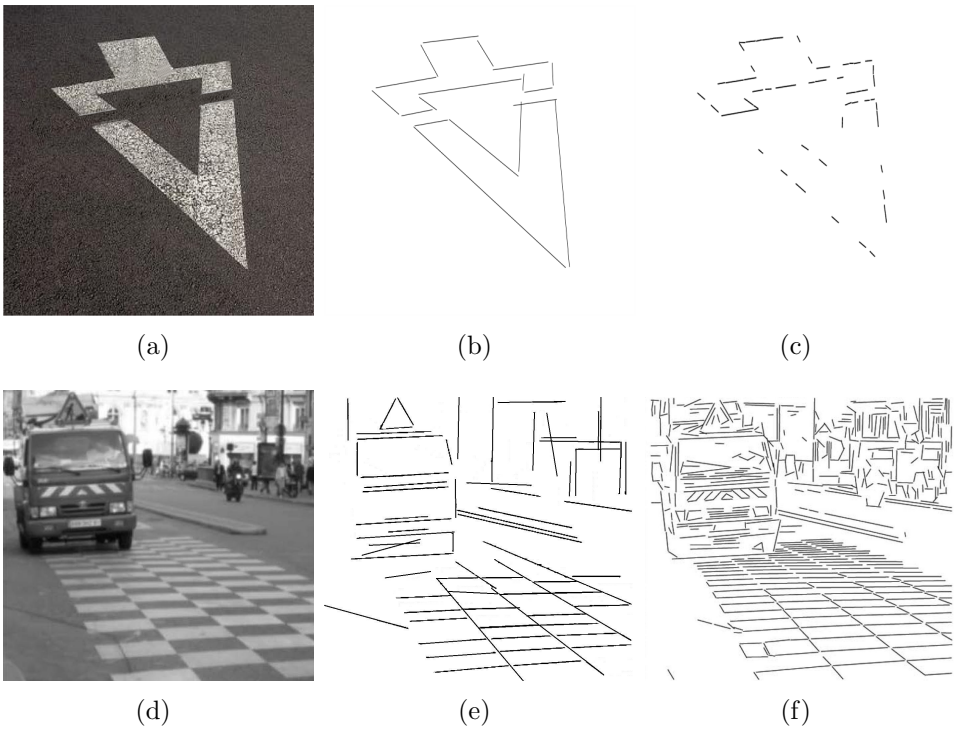


Figure 2.2: Performance of the bottom-up versus top-down method. (a) Original image of arrow (512×512). (b) Result of the EHT where long segments are correctly detected. (c) Result of the LSD shows disconnections in the segments due to background noise. (d) Original image of truck (911×850). (e) Result of the EHT when it fails to detect many of the short segments. (f) The LSD successfully detects segments as small as a few pixels length.

From the machine learning viewpoint, the top-down method is a model selection technique from parameter space to data or feature space, whereas the bottom-up method is a regularisation technique from data or feature space to parameter space. The optimal solution can be obtained using a two pathway learning process, in which both top-down model selection and bottom-up regularisation are considered. This leads us to the third category of the methods which is the domain-crossing approach.

2.3.3 Domain-crossing approach

The domain-crossing approach utilises the information from both the image and parameter space. It starts by mapping the feature points from the image to the parameter domain and again using the image-domain data combined with the parameter domain data to capture the line segments.

Song and Lyu speed up the HT voting process using image gradient prediction [5]. After peak detection in Hough space, a unique line verification method is used by crossing from Hough space over to the image space. In this way, line thickness can also be distinguished as well as achieving a reduction in false detection. Despite its accuracy the method is computationally expensive.

Bandera et al. suggest a more efficient algorithm in [94]. A random window randomised Hough transform is used in the global phase to construct the Hough space and capture the line parameters, i.e. the peaks. A unique mean shift clustering technique is utilised afterwards to highlight the peaks and find the potential straight-lines. Crossing over to the image space, edge pixels that are aligned with the detected lines are projected onto the lines to approximate the true line segments in a local merging procedure. The idea was further improved in slice sampling weighted mean shift (SSWMS) analysis [95]. Sequential sampling of the Hough space was suggested to enhance the random sampling approach as well as a new likelihood function for the local clustering. The SSWMS shows a better accuracy than the PPHT with

faster execution time.

In [27], the beamlet transform has been viewed as a special case of the Radon transform (RT). The underlying idea is to apply a feature-adapted RT in a quadtree decomposition manner to obtain a feature-adapted beamlet transform (FABT). The term *feature-adapted* comes from the convolution of basis filters (i.e. second- or third-order edge and ridge detectors [96]) with the corresponding image scale prior to RT computation. This filtering highlights linear structures of the image and enhances the accuracy and performance of the task. However, the FABT is also computationally expensive because all of the convolutions must be applied in the time domain due to the indefinability of the convolution theorem for the pseudo-polar Fourier transform.

Recently, a novel geometrical technique based on the neighbourhood of straight-line segments in both the spatial and the parameter domains was introduced [66, 67]. The neighbourhood of a line segment in the parameter domain is defined in a lozenge-like quadrangle to approximate the neighbourhood of the segment in the image domain. Instead of two endpoints, Du et al. used coordinates of the centre point to find the location of the segment in the image. However, this method suffers from the traditional problem of the HT-based techniques, i.e. detecting short line segments, where we face a trade-off in the neighbourhood radius selection. That means segments with shorter length have bigger approximation errors in comparison to those with longer length.

2.4 Discussion

Table 2.1 gives an overview of the different approaches to line segment detection discussed in this chapter.

The traditional point-to-segment extraction methods are computationally efficient and simple to implement. That makes such methods suitable for real-time object detection and tracking applications. Nevertheless, their focus is

on local information such as gradient magnitude and angle at pixel level. That makes noise an effective parameter in degrading their performance. In addition, failing to consider global information results in discontinuity in the case of elongated segments.

The traditional line-to-segment extraction methods work the other way around. Their global nature reduces the effect of noise and increases their robustness. That is also the reason that global approaches can detect large objects more accurately. However, neglecting the local gradient information in an image causes inaccuracy in extracting short line segments. In addition to that, the computational burden of the global methods is considerable and such methods are more appropriate for off-line image processing.

The domain-crossing approaches benefit from different properties in both image space and transformed space to identify the line segments. They are more robust to noise and occlusion due to their global nature as well as having higher accuracy for short line segmentation because of their image-domain line segmentation algorithms. However, they require a delicate design of algorithm to speed up the line extraction process, and it is not well-suited for real-time applications.

The proposed methodology

The research in this thesis aims to introduce a novel domain-crossing approach to line segment detection problems. The proposed methodology is based on an advanced HT line detector, i.e. an HT using a multi-layer fractional Fourier transform for the global image analysis, discussed in Chapter 3. In the second phase, i.e. crossing over to the image domain, the proposed method adapts an accurate image-domain window-of-interest to detect the true segments and is discussed in Chapter 4.

Table 2.1: An overview of the different approaches to line segment detection.

Categories	Description	Advantages	Disadvantages
Bottom-up	Region growth from pixel to segment. Representative works include Nevatia and Babu [82], Kahn et al. [83], Burns et al. [84], Gioi et al. [77], Akinlar and Topal [78], and Yang et al. [79]	Fast and simple, appropriate for real-time applications	Sensitive to noise, breaks down lengthy segments
Top-Down	Straight-line detection, followed by segment extraction. Representative works include Hough transform family [1], Yuen et al. [90], Guil et al. [91], Matas et al. [49], Nguyen et al. [41], Cha et al. [92], and Chung et al. [93].	Robust to noise, suitable for detecting large objects	Slow, inaccuracy in extracting short lines
Domain-Crossing	Decomposition of lines through transformed space. Representative works include Song and Lyu [5], Bandera et al. [94], Nieto et al. [95], Berlemont and Olivo-Marin [27], and Du et al. [66, 67].	Robust to noise, more accurate for short segments	Slow

Chapter 3

Advanced Fourier-based Hough transform

In this chapter an advanced Fourier-based Hough transform, i.e. Hough transform using multi-layer fractional Fourier transform (MLFRFT) is introduced as the first stage in the line segmentation algorithm proposed in this research. The aim is to, first, extract the potential straight-lines in the image and detect the set of line parameters ρ and θ for every possible straight-line. Each set of line parameters is used in the next stage (Chapter 4) to create an image-domain window-of-interest to determine the endpoints of the line segment. Therefore, accurate line detection is highly significant to assure the validity of the segment detection. The computational complexity of the MLFRFT has been compared with the traditional zero-padding method and the standard Hough transform. In addition, to test the performance of the MLFRFT approach, a new lane marker detection method is suggested at the end of the chapter.

3.1 Hough transform using Fourier method

As mentioned in Section. 2.2.3, Fourier-based HT methods rely on the mathematical identity of the Hough transform and the Radon transform (RT). One of the main motivations is the use of greyscale image instead of binary edge maps as well as benefiting from the fast implementation of the Fourier transform. This will significantly improve the computational efficiency of the HT. Experimental results conducted in this research show that the MLFRFT is approximately three times faster than the SHT. For instance, for a greyscale natural image of size (512×512), the MLFRFT-based HT required approximately 0.125 seconds to produce the Hough peaks whilst the SHT required 0.373 seconds, (on a Dell machine, Intel Core i5, 2.4 GHz running MATLAB. 8).

Central-slice theorem

In practice, the RT enjoys the unique property of the *central-slice theorem* in the frequency domain for a fast and efficient implementation of the algorithm [97, 98]. This is one of the most important characteristics of the RT because it implies that the RT can be computed using the fast Fourier transform with computational complexity of $O(N^2 \log_2 N)$ for an image of $(N \times N)$ pixels. Given a 2D discrete function $f(x, y)$, the central-slice theorem shows the equality of the result of the following two procedures.

- Project the function onto a line (1-dimensional space) of a certain angle θ, and compute a 1D discrete Fourier transform (1D-DFT) of that projection.
- Compute a 2D discrete Fourier transform (2D-DFT) of the function, then take out a slice passing through its origin from the same angle θ.

Projection of a 2D function onto a line through θ is, in fact, the forward RT of that function, i.e. $\mathbf{R}_\theta[f(x, y)]$ in Equation 2.4. Therefore, the first

statement means applying the 1D Fourier transform to the RT of that function. Comparing the two statements, it can be concluded that a slice through the origin of the 2D-DFT spectrum of $f(x, y)$ at angle θ is equivalent to the 1D-DFT of the Radon projection of $f(x, y)$ at θ. Consequently, to obtain a RT of an image $f(x, y)$ in a fast and efficient manner it is easy to, first, compute a 2D-DFT of the image and then compute a 1D inverse DFTs over its central slices.

$$f(x,y) \xrightarrow{2D-DFT} F(\mu,\eta) \rightarrow Central-slice\ theorem \xrightarrow{1D-DFT^{-1}} \mathbf{R}f(x,y)$$

To better explain the central-slice theorem in a mathematical notation, let us consider $\Lambda(v, \theta)$ as a 1D-DFT of the Radon projection $\lambda(\rho, \theta)$, given the Equation 2.4, defined by

$$\begin{aligned}
\Lambda(v,\theta) &= \int \left[\int \int f(x,y)\ \delta(\rho - x\cos\theta - y\sin\theta)\ \mathrm{d}x\mathrm{d}y \right] e^{-j2\pi\rho v}\mathrm{d}\rho \\
&= \int \int f(x,y) \left[\int \delta(\rho - x\cos\theta - y\sin\theta)\ e^{-j2\pi\rho v}\ \mathrm{d}\rho \right] \mathrm{d}x\mathrm{d}y \\
&= \int \int f(x,y) \left[\int \delta(\rho - (x\cos\theta + y\sin\theta))\ e^{-j2\pi\rho v}\ \mathrm{d}\rho \right] \mathrm{d}x\mathrm{d}y \quad (3.1)
\end{aligned}$$

Recalling one of the unique properties of the Dirac delta function, i.e. $\int \delta(\rho - a)\ e^{-k\rho}\ \mathrm{d}\rho = e^{-ka}$, Equations 3.1 can be simplified as

$$\Lambda(v,\theta) = \int \int f(x,y)\ e^{-j2\pi(xv\cos\theta + yv\sin\theta)}\ \mathrm{d}x\mathrm{d}y \quad (3.2)$$

Equation 3.2 is equivalent to a central-slice of the 2D-DFT of function $f(x, y)$ at angle θ. In other words, if we denote the slicing operator S_θ as

$$S_\theta[f(x,y)](x') = f(x'\cos\theta, y'\sin\theta) \quad (3.3)$$

and let $\mathbf{F}_1$ and $\mathbf{F}_2$ be 1D and 2D Fourier transform operators, respectively, then the central-slice theorem states

$$\mathbf{F}_1\{\mathbf{R}_\theta[f(x,y)]\}(v) = S_\theta\{\mathbf{F}_2[f(x,y)](\mu,\eta)\}(v) \tag{3.4}$$

where $\mathbf{R}_\theta$ is Radon projection at angle θ, and $\mu-\eta$ are horizontal and vertical axes in 2D Cartesian Fourier space. Equation 3.4 demonstrates that the 1D Fourier transform of the Radon integral projection at angle θ is equal to the slice taken from the origin of the 2D Fourier transform at the same angle.

It should be emphasised that the 2D-DFT is a linear one-to-one transformation from Cartesian spatial space to the Cartesian frequency or Fourier space. The DFT functions implemented in computer programs produce a Cartesian grid of rows and columns. As a result, taking out the slices through the centre of such a spectrum would be problematic. Therefore to apply the central-slice theorem, the Cartesian coordinate system of the Fourier space has to be converted to a polar or circular coordinate system using interpolation techniques such as the bilinear or nearest neighbour interpolation. Then it would be easy to apply the 1D-DFT^{-1} to the rows of the polar grid.

In summary, instead of computing projections along all of the θ angles in the spatial domain to obtain the RT, it is feasible to transfer the image to the corresponding frequency domain by performing a 2D-DFT and interpolating its Cartesian coordinates to polar. Parameter space is then computed utilizing a 1D-DFT^{-1} along the distance axis (i.e. ρ direction) of the polar frequency grid.

The above procedure is swift because in practice the fast Fourier transform (FFT) reduces the computational cost from $O((N^2)^2)$ operations to $O(N^2 \log_2 N)$ for an $(N \times N)$ image. For a detailed description of the FFT algorithm, please refer to Appendix A. Nevertheless, the FFT only produces $(N \times N)$ frequency samples which may not be a sufficient sampling rate to guarantee the required accuracy. In other words, frequencies of interest may not fall into an $(N \times N)$ discretised plane. Such a problem, i.e. restriction

in the number of frequency samples in the Fourier spectrum causes aliasing effects and increases the probability of false detection. Therefore a higher number of frequencies is required to assure full coverage of all possible frequencies.

Zero-padding was suggested to increase the number of frequency samples in the Fourier spectrum by upsizing the original image with zero-valued pixels [70]. For instance, a zero-padded image will have the size of $(bN \times bN)$, where $b > 1$ and typically is an integer of power of 2. This will result in $(bN \times bN)$ samples in the Fourier spectrum. In other words, having more pixels in an image means having more frequency samples in the Fourier spectrum. Since the additional pixels are zero, they do not interfere with the original signal and have no effect but increasing the number of frequency samples. However, there are two major challenges associated with the zero-padding:

1. Imposing extra computation and memory requirement as a result of upsizing the original image.

2. Interpolation error during the Cartesian to polar mapping. Note that to apply the central-slice theorem, the rectangular coordinates are mapped to circular or polar coordinates via an interpolation process, i.e. bilinear or nearest neighbour interpolation.

The multi-layer fractional Fourier transform (MLFRFT) addressed the above mentioned deficiencies with its unique frequency sampling approach. Instead of adding zeros to the original image, the Fourier spectrum is captured via a union of multiple layers of the DFT. Moreover, Frequency samples of the MLFRFT grid are naturally closer to the polar grid, hence the interpolation error is reduced. The MLFRFT is the basis of the proposed segmentation algorithm and has been discussed in the following.

3.2 Multi-layer fractional Fourier transform

FFT-based algorithms are incapable of performing polar or log-polar Fourier transforms in an accurate and efficient manner. This causes a major problem for the Fourier-based HT as explained in the previous section. Efforts have been made to tackle the problem. Basically, estimation of the Cartesian Fourier spectrum on a polar or log-polar grid can be achieved in two ways: apply image warping [99] followed by the FFT computation [100] or direct interpolation of the Cartesian samples into polar samples [101]. Interpolation-based methods received much attention due to accuracy and small interpolation error of the non-uniformly sampling approaches such as in [102, 103, 104, 105, 106]. Among them, the pseudo-polar Fourier transform (PPFT) demonstrated significantly lower interpolation error with almost the same computational complexity as the 2D-FFT [105, 107]. However, the PPFT is not well-suited for images with large scale factor [108] as well as hand-taken images. Pan et al. [109] addressed the shortcomings of the PPFT-based approaches with a novel multi-layer Fourier approach as explained in the following.

3.2.1 1D fractional Fourier transform

Let us start from the traditional discrete Fourier transform and extend the concept to an adaptable fractional Fourier transform. Given a discrete signal $\{f(x)| -\frac{N}{2} \leqslant x \leqslant \frac{N}{2} - 1\}$ with N discrete samples, where N is an even number, a 1D discrete Fourier transform (DFT) can be defined as

$$F(k) = \sum_{x=-\frac{N}{2}}^{\frac{N}{2}-1} f(x)\ e^{-j2\pi xk/N} \tag{3.5}$$

where frequency samples are uniformly distributed in a $[-\pi, \pi]$ plane. By adding an adjustable parameter to the exponential component, a fractional

Fourier transform (FRFT) can then be defined as

$$F^{\gamma}(k) = \sum_{x=-\frac{N}{2}}^{\frac{N}{2}-1} f(x)\ e^{-j2\pi xk\gamma/N} \tag{3.6}$$

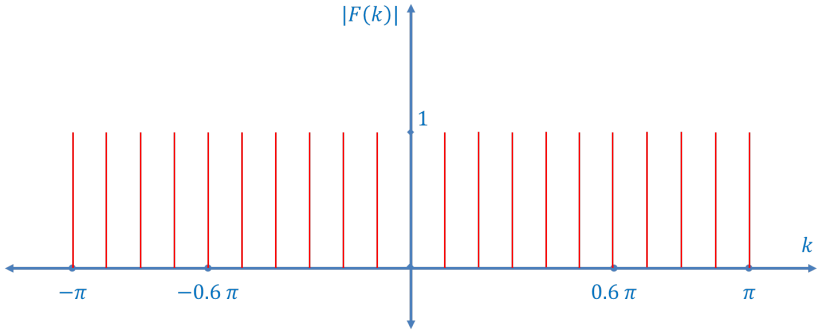
where the additional parameter γ is a non-integer predefined fractional scaler, and $0 < \gamma \leqslant 1$. It can be seen that frequency samples in the fractional Fourier spectrum are uniformly distributed in a $[-\gamma\pi, \gamma\pi]$ plane. When $\gamma = 1$ the fractional Fourier transform acts similarly to the traditional DFT and the frequency domain is scattered in the $[-\pi, \pi]$ plane. This is a unique property of the fractional Fourier transform that indicates the feasibility of having non-integer frequency samples. Having said that $F(\gamma k)$ can be defined as

$$F(\gamma k) = \sum_{x=-\frac{N}{2}}^{\frac{N}{2}-1} f(x)\ e^{-j2\pi xke\gamma/N}. \tag{3.7}$$

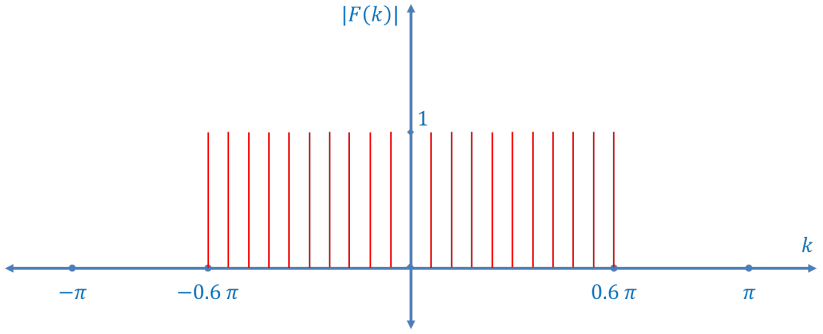
Hence given the right hand sides of Equations 3.6 and 3.7 are identical, it is true to say

$$F^{\gamma}(k) = F(\gamma k). \tag{3.8}$$

To better understand the concept of integer and non-integer Fourier spectrum let us consider the magnitude plot of $F(k)$ as depicted in Figure 3.1. Here $N = 20$, i.e. the number of discrete samples. Figure 3.1(a) shows the distribution of the samples when $\gamma = 1$. This is the case when the DFT and FRFT of a signal are identical and the Fourier spectrum is scattered in the $[-\pi, \pi]$. By changing the γ to 0.6 we will have the same number of samples but distributed in the $[-0.6\pi, 0.6\pi]$ plane, as shown in Figure 3.1(b). That implies a higher resolution frequency sampling with the same N. In Figure 3.1(a) only 12 samples represent the frequency response of the signal from -0.6π to 0.6π. However, using the fractional scaler it is possible to have all 20 samples allocated to the $[-0.6\pi, 0.6\pi]$ interval.



(a)



(b)

Figure 3.1: The concept of integer and non-integer spectrum using 1D fractional Fourier transform. (a) $\gamma = 1$ in Equation 3.6, the FRFT and DFT of the signal are equal and the distribution is in $[-\pi, \pi]$. (b) $\gamma = 0.6$, the FRFT provides higher resolution frequency sampling in $[-0.6\pi, 0.6\pi]$.

Note that the number of possible samples is fixed at N and higher accuracy cannot be achieved by simply increasing N, because N is the number of elements in the discrete input signal/function $f(x)$. However a higher resolution Fourier spectrum can be obtained by computing a number of FRFTs of the original signal, each with different fractional scalers, and combining the result. This is the fundamental concept behind the multi-layer fractional Fourier transform and will be explained in Section 3.2.3.

3.2.2 2D fractional Fourier transform

Extending Equation 3.6, a 2D fractional Fourier transform operator is defined by

$$F^{\gamma}(k_1, k_2) = \sum_{x=-\frac{N}{2}}^{\frac{N}{2}-1} \sum_{y=-\frac{N}{2}}^{\frac{N}{2}-1} f(x, y)\ e^{-j2\pi(x\gamma k_1 + y\gamma k_2)/N} \tag{3.9}$$

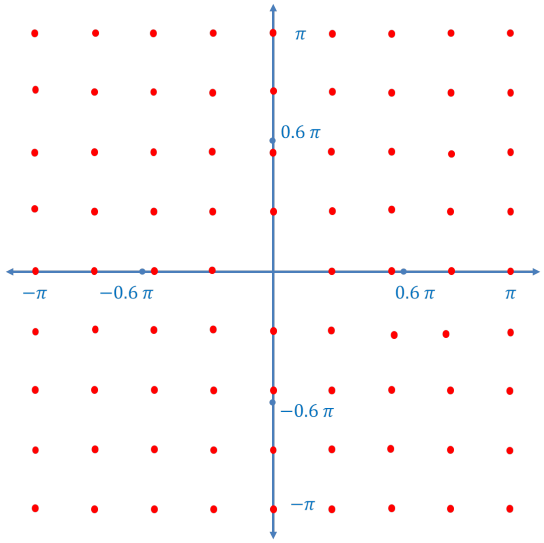
where $\{f(x, y)| - \frac{N}{2} \leqslant x, y \leqslant \frac{N}{2} - 1\}$. Here $f(x, y)$ is a 2D discrete signal which, in our case, is a 2D digital image with $N \times N$ pixels. When $\gamma = 1$, the frequency response of the signal is scattered in a $[-\pi, \pi] \times [-\pi, \pi]$ grid which is the same as the response to the traditional 2D-DFT. For any other values of γ between 0 and 1 the frequency response is in a $[-\gamma\pi, \gamma\pi] \times [-\gamma\pi, \gamma\pi]$ grid. Similar to the previous example, Figure 3.2 shows an integer and non-integer Fourier spectrum using a 2D-FRFT. The result for $\gamma = 1$ and $\gamma = 0.6$ is depicted in Figure 3.2(a) and (b), respectively.

In addition, Equation 3.9 implies that the 2D fractional Fourier transform can be performed in an analogous manner as the traditional 2D-DFT; computing the 1D transforms for every row and then computing the 1D transforms for every column. Therefore, the FFT (fast Fourier transform) algorithm can be used to reduce the computational cost from $O((N^2)^2)$ operations to $O(N^2 \log_2 N)$ for an $(N \times N)$ image. It is therefore easy to extend Equation 3.8 to a 2D version defined by

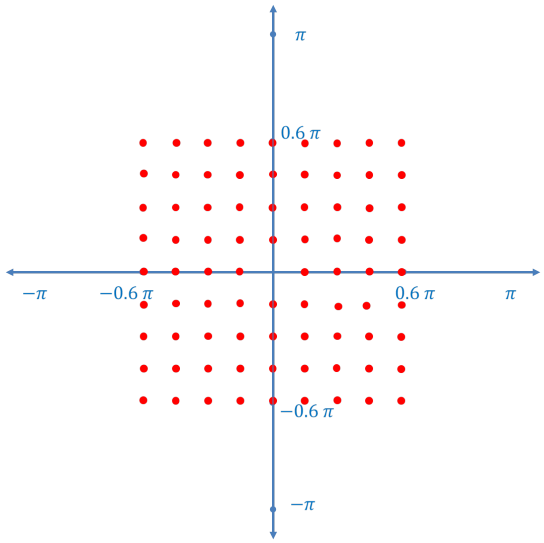
$$F^{\gamma}(k_1, k_2) = F(\gamma k_1, \gamma k_2). \tag{3.10}$$

3.2.3 2D multi-layer fractional Fourier transform (ML-FRFT)

The idea of a multi-layer fractional Fourier transform (MLFRFT) is to apply a number of FRFT, each with a different γ, and combine the results to make a unified Fourier spectrum. If applied to an image, the resulting frequency



(a)



(b)

Figure 3.2: Integer and non-integer spectrum using 2D fractional Fourier transform. (a) $\gamma = 1$ in Equation 3.6, the 2D-FRFT and 2D-DFT of the signal are equal and the distribution is in $[-\pi, \pi] \times [-\pi, \pi]$. (b) $\gamma = 0.6$, the 2D-FRFT provides higher resolution frequency sampling in $[-0.6\pi, 0.6\pi] \times [-0.6\pi, 0.6\pi]$.

response contains more frequency samples compared with the traditional one-layer 2D-DFT. Therefore, one can obtain more information of pixel intensity variation to increase both the performance and the accuracy. Moreover, the resulting frequency samples are closer to the polar coordinates [109]. Hence, the interpolation error, caused by the mapping from Cartesian to a polar grid, is minimized.

Let us define L as the number of FRFT layers and $i = 1, 2, ..., L$. Given the fractional coefficient γ_i, a MLFRFT grid is defined as a set of P_i when

$$P_i = \{\mathbf{F}(\gamma_i k_1, \gamma_i k_2)| - \frac{N}{2} \leqslant k_1, k_2 \leqslant \frac{N}{2} - 1\} \tag{3.11}$$

A union of L layers results in a complete set of frequency grids P by

$$P = \bigcup_{i=1}^{L} P_i \tag{3.12}$$

Equation 3.12 indicates that P, which is the resulting MLFRFT grid, has L times more frequency samples compared with the traditional 2D-DFT. It is clear that the higher the number of layers, the higher the resolution of the resulting Fourier spectrum, i.e. $(L \times N \times N)$ instead of $(N \times N)$ samples. Figure 3.3 is a depiction of MLFRFT grid with three fractional layers. The fractional coefficients are $\gamma_1 = 0.5$, $\gamma_2 = 0.8$, and $\gamma_3 = 1$ in blue, green, and red, respectively. A block diagram of the Radon transform using multi-layer fractional Fourier transform is depicted in Figure 3.4.

Note that the MLFRFT takes an image from the spatial domain and generates a pure frequency domain grid. The only difference between the MLFRFT and the traditional DFT, in terms of their frequency-domain grid, is the finer sampling of the MLFRFT as a result of the multi-layer approach. Therefore, the central slice theorem still holds for the MLFRFT.

It has been shown in [109] that the multi-layer approach outperforms other Fourier-based methods including the pseudo-polar Fourier transform

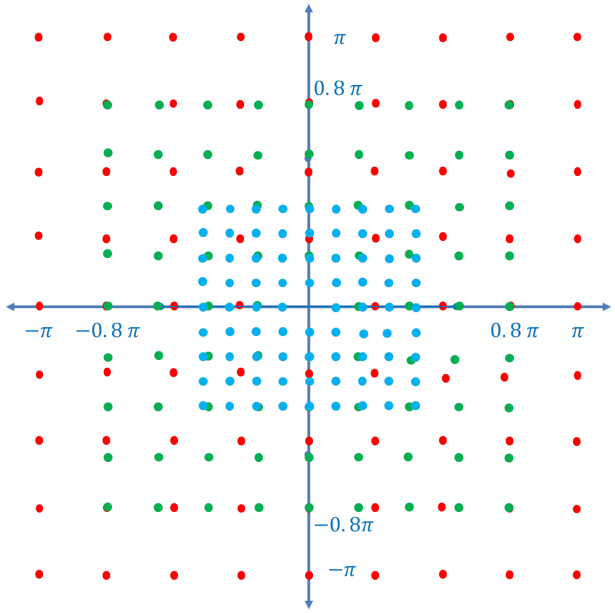


Figure 3.3: Multi-layer fractional Fourier transform grid with three fractional layers. $\gamma_1 = 0.5$ (blue), $\gamma_2 = 0.8$ (green), and $\gamma_3 = 1$ (red).

[105]. In addition to its finer Fourier spectrum that improves the accuracy, location of frequency samples in the MLFRFT grid are closer to the polar coordinates. Such closeness will help reduce the interpolation error caused during the Cartesian to polar mapping of the samples.

It should be emphasised that computation of each fractional layer is independent of other layers. Therefore, calculating the whole set of layers can be carried out in parallel using multi-core CPU systems.

To obtain an even more precise spectrum, the MLFRFT can be extended to have two adjustable parameters for each layer, instead of one. A different set of scale factors can be used for the x and y axes for each fractional layer L, defined as the generalised interpolated Fourier transform (GIFT) [71]:

$$F^{\gamma_1,\gamma_2}(k_1,k_2) = \sum_{x=-\frac{N}{2}}^{\frac{N}{2}-1} \sum_{y=-\frac{N}{2}}^{\frac{N}{2}-1} f(x,y)\; e^{-j2\pi(x\gamma_1 k_1 + y\gamma_2 k_2)/N} \tag{3.13}$$

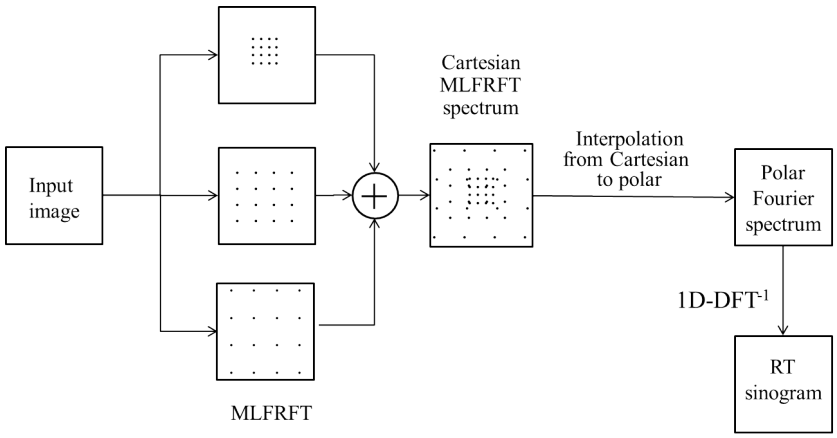


Figure 3.4: Block diagram of the MLFRFT-based HT.

where $0 < \gamma_1, \gamma_2 \leqslant 1$ are predefined fractional scalers for the x and y dimensions, respectively. The resulting non-integer grid has specifically designed frequency samples distributed in a $[-\gamma_1\pi, \gamma_2\pi]\times[-\gamma_1\pi, \gamma_2\pi]$. Figure 3.5 shows and example of such a grid with $\gamma_1 = 0.6$ and $\gamma_2 = 0.8$.

Although the GIFT suggests a more flexible frequency grid than the MLFRFT it does not provide a finer spectrum, but a type of customised grid for specific application dependent tasks. Furthermore, the GIFT suffers from tedious parameter tuning without a proper adaptable tuning method. Therefore adjusting two different parameters for each layer based on trial and error can be problematic. As a result MLFRFT has been adapted in this research to achieve a high resolution Fourier spectrum that leads to higher a accuracy Radon sinogram.

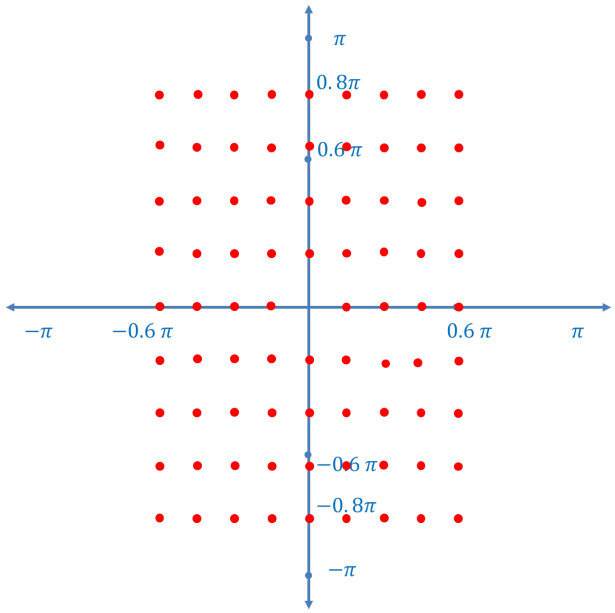


Figure 3.5: 2D non-integer GIFT grid with two fractional scalers γ_1 and γ_2. $\gamma_1 = 0.6$ is the scale factor in the x axis and $\gamma_2 = 0.8$ is a scale factor in the y axis.

3.3 Line detection using MLFRFT

Now that a high resolution Fourier spectrum has been generated thanks to the MLFRFT approach, it can be utilised for detection of straight-lines in a greyscale image. The aim is to obtain a set of distinguishable peaks in a sinogram using the MLFRFT. Consider an input greyscale image $f(x, y)$ with $(N \times N)$ pixels. Applying a MLFRFT with L layers results in a $(n \times n)$ Fourier spectrum where n is determined by the size of fractional layers. Note that, to guarantee a full coverage of border frequencies in the Fourier spectrum, $\gamma = 1$ will always be used as one of the fractional layers. The following steps have to be performed to construct the sinogram.

3.3.1 Step 1. Interpolation of top-half grid

One of the interesting characteristics of the Fourier spectrum is the symmetry of frequency samples to the origin of the grid. This property is particularly useful for Cartesian to polar mapping of the samples and means it is sufficient to map only the top-half of the Cartesian frequency grid from $\mu - \eta$ to polar grid $v - \theta$ and then duplicate the result with a conjugate mirror to compensate for the missing data. To discretise the θ from 0 to π, a predefined resolution $\Delta\theta$ is defined as

$$\Delta\theta = \frac{\pi}{h-1} \tag{3.14}$$

where h denotes a user-defined number of desired angles uniformly distributed in $[0, \pi]$, i.e. $\theta = 0, \Delta\theta, 2\Delta\theta, ..., \pi$. For instance, $h = 256$ means $0.012\ rad$ (0.7°) for each value of θ. h has to be sufficiently high to guarantee optimum pixel coverage. Each element of the polar $v - \theta$ grid at $(v = \sqrt{\mu^2 + \eta^2}, \theta = \tan^{-1}\frac{\eta}{\mu})$ is computed at the location of its Cartesian coordinate and interpolation of its surrounding elements in the $\mu - \eta$ grid.

3.3.2 Step 2. Peak enhancement

It is a common practice to perform edge enhancement to the input image prior to the Radon transform computation using a 2D convolutional bandpass filter. Such filtering enhances the high frequency components of the Fourier spectrum and therefore highlights the sinogram peaks. Due to the convolution property of the Fourier transform, filtering can be applied in the frequency domain by a simple multiplication. Benefiting from the filter theorem suggested in [98], the 2D filtering operation can be reduced to a 1D filtering. Accordingly, the Radon projection of a 2D convolution of two 2D functions f_1 and f_2 is equivalent to a 1D convolution of the Radon projections of each of the functions. Figure 3.6 illustrates this concept.

Therefore, after computing the polar spectrum of the top-half of the ML-FRFT spectrum, a 1D difference of Gaussian (DoG) filter is applied to enhance the peak structure of the sinogram. Because the Fourier spectrum of the 2D-DoG is circular symmetric to its origin, a 1D-DoG is obtained via taking a projection over angular slices about the origin of the 2D-DoG. The 2D-DoG is a second order bandpass filter defined by function $h(x, y)$ as

$$h(x,y) = d_e^2 \ \exp\left(\frac{-d_e^2}{x^2+y^2}\right) - d_i^2 \ \exp\left(\frac{-d_i^2}{x^2+y^2}\right) \tag{3.15}$$

where

$$d_{e,i} = \frac{1}{\sigma_{e,i}\sqrt{2\pi}} \tag{3.16}$$

σ_e and σ_i are excitatory and inhibitory standard deviation of the Gaussian filters. The best approximation occurs when the ratio between the σ_e and σ_i is 1.6, suggested by Marr and Hildreth [110]. After simplification, the Fourier transform of the $h(x, y)$ is obtained as:

$$\Lambda_h(v,\theta) = \exp\left(\frac{-v^2}{d_e^2}\right) - \exp\left(\frac{-v^2}{d_i^2}\right) \tag{3.17}$$

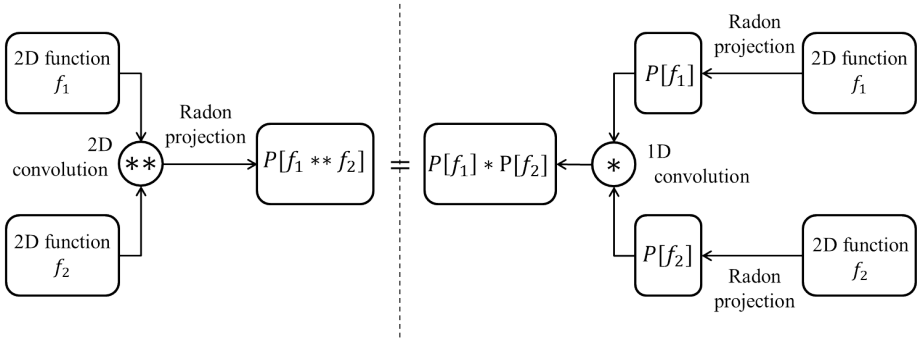


Figure 3.6: The Radon projection of 2D convolution of 2D functions f_1 and f_2 is equivalent to the 1D convolution of the Radon projections of each of the functions.

The proof of $\Lambda_h(v,\theta)$ is given in Appendix B. It can be seen that $\Lambda_h(v,\theta)$ is independent of θ. That implies 1D operations along the distance axis, i.e. v, can be applied instead of 2D filtering. Therefore, the original signal $\Lambda(v,\theta)$ turns into a filtered signal using multiplication in the Fourier domain, given as

$$\Lambda_g(v,\theta) = \Lambda(v,\theta) \times \Lambda_h(v,\theta) \tag{3.18}$$

3.3.3 Step 3. Conjugate mirror

To recreate the missing data in the bottom-half grid that was removed before Cartesian-polar mapping, it is easy to duplicate the result after 1D filtering using a conjugate mirror. Note that because the elements in the frequency domain are complex numbers, a conjugate of the top-half frequencies is obtained and then frequencies are reflected using a simple matrix reflection.

3.3.4 Step 4. 1D-DFT^{-1}

Finally to obtain the Radon transform a 1D-DFT^{-1} is applied to the conjugate mirrored $\Lambda_g(v,\theta)$ along its v axis, as explained in Section 3.1 and

suggested by Equations 3.1 and 3.2. The result produces a sinogram with a number of overlapping sinusoidal curves in a shape of butterfly, having intersection at the peaks. Such peak structures are much more consistent due to the MLFRFT frequency sampling and 1D-DoG filter for peak enhancement. A peak detection is performed to identify the prominent peaks. Each peak corresponds to a potential straight-line in the image.

3.4 Evaluation on computational complexity

3.4.1 Parallel implementation of the MLFRFT

As noted in Section. 3.2.3, computing each of the fractional layers of the MLFRFT is an independent process and can be carried out in parallel with the other layers using multi-core CPUs. Although parallel computation of the MLFRFT is theoretically feasible and has been highlighted as one of the notable strengths of this method in both [109] and [2], empirical studies conducted in this research show that the parallel implementation could increase the computational time significantly.

Given the pseudo-code for the MLFRFT in Algorithm. 2, operations inside the **for** loop can be carried out in parallel using a multi-core computer system. Depending on the number of available cores, the client splits the task within the loop and allocates a number of parallel workers to take over the iterations. Each iteration is carried out by one worker that receives the required data from the client. Here the number of iterations depends on the number of fractional layers, i.e. L, that is usually 3 or 4. Using MATLAB's *Parallel Computing Toolbox* and running the code for a (512$\times$ 512) greyscale image on a CPU with two physical cores, a four-layer MLFRFT requires 0.194 seconds. This is when the serial implementation for the similar configuration requires 0.051 seconds. The difference is due to the following two reasons: 1) having a small number of iterations inside the **for** loop, e.g. four iterations in this case; 2) the code executed within the **for** loop is not

computationally heavy enough to overcome the communication overhead between the client and the workers. We should keep in mind that data transfer from the client to the workers normally has a high computational cost. To observe a real difference between parallel and serial computing, either the number of iterations or the complexity of the task inside the loop has to be increased. In our case, the complexity of the code cannot be increased as it is a Fourier transform operation. Only by significantly increasing L, e.g. $L > 100$, can we start observing improvements of parallel computing over serial computing. However, it is unnecessary for the MLFRFT to have more than 100 layers of FRFT.

Algorithm 2 Pseudo-code for the MLFRFT.

Require: : $I(x, y)$ /* An input image

Require: : $\gamma = [\gamma_1, ..., \gamma_L]$ /* L = maximum number of layers

1: **for all** i , $i = 1 \rightarrow L$ **do** /* given γ

2: $F^{\gamma_i}(k_1, k_2) \leftarrow \sum_{x=-\frac{N}{2}}^{\frac{N}{2}-1} \sum_{y=-\frac{N}{2}}^{\frac{N}{2}-1} f(x, y)\ e^{-j2\pi(x\gamma_i k_1 + y\gamma_i k_2)/N}$

3: $P_i \leftarrow F^{\gamma_i}$

4: **end for**

5: $P \leftarrow \bigcup_{i=1}^{L} P_i$ /* MLFRFT frequency grid

It is easy to observe the communication overhead between a client and the workers in MATLAB by running a simple experiment and comparing the computational time in serial and parallel mode. One of the most fundamental operations for a compiler is adding numbers in a specified range. Suppose we want to run an experiment of adding all of the natural numbers from a range x_1 to x_n. For instance, running this experiment for $x_1 = 1$ to $x_n = 1,000,000$ requires 0.017 seconds in serial mode and 0.67 seconds in parallel mode with two workers. It can be seen that for such simple operations, there is a considerable communication cost between the client and the workers. This is a measure that denotes how complicated the code inside the loop has to

be in order to take advantage of the parallel mode.

Based on the above discussion, parallel computing cannot be suggested for the MLFRFT in practical applications, due to the small number of required FRFT layers, which is typically three or four.

3.4.2 Computational time comparison

Empirical studies have shown that the MLFRFT-based HT outperforms the SHT in both detection accuracy and computational time. Evaluation on detection accuracy of the 3-layer MLFRFT was reported in [2]. Here the MLFRFT method has been tested against the SHT in terms of the processing time. A set of Twenty-five greyscale images with equal size of (512×512) has been used for this experiment. The computing configuration for the experiments is as follows: MATLAB 8 running on Intel Core i5 CPU, 2.4 GHz with 8 GB RAM, Microsoft Windows 7. A 4-layer MLFRFT was used rather than 3-layer, with $\gamma_1 = 0.5$, $\gamma_2 = 0.7$, $\gamma_3 = 0.9$, and $\gamma_4 = 1$, to achieve higher accuracy in the sinogram. Table 3.1 shows the results.

It can be seen that the whole process of a typical 4-layer MLFRFT-based HT requires approximately 0.125 seconds, including 0.059 seconds to compute four fractional layers, 0.065 seconds for Cartesian to polar mapping plus conjugate mirror, and 0.0012 seconds for the 1D-DFT^{-1}. The SHT, by comparison, requires approximately 0.373 seconds, including 0.31 seconds to detect the edge map based on a Canny edge detector plus 0.063 seconds for the HT voting process. The results indicate that the MLFRFT is approximately three times faster than the SHT. This to be added to the four times greater accuracy than the SHT due to the 4-layer MLFRFT. The main cause of difference in computing time is the compulsory edge detection required prior to the SHT that takes the majority of computation. In addition to imposing a significant computational cost to the system, edge detection may increase false detections. When the noise-level in an image is high, some of the true feature points in the image can be ignored as noise and also some

Table 3.1: Computational time comparison between the MLFRFT-based HT (with four layers) and the SHT. Twenty-five greyscale image of size (512×512) has been tested on MATLAB. The MLFRFT-based HT is approximately 3 times faster than the SHT. The time unit is in seconds.

	MLFRFT				SHT		
Image	4 × 2D-FRFT	$x-y$ to $v-\varphi$ conjugate mirror	1D-DFT-1	Total	Edge detection (Canny)	HT	Total
I-1	0.052	0.067	0.0011	0.120	0.290	0.095	0.385
I-2	0.056	0.067	0.0014	0.124	0.313	0.078	0.391
I-3	0.060	0.064	0.0012	0.125	0.345	0.042	0.387
I-4	0.060	0.063	0.0013	0.124	0.320	0.085	0.405
I-5	0.076	0.066	0.0010	0.143	0.326	0.058	0.384
I-6	0.060	0.061	0.0013	0.122	0.337	0.075	0.412
I-7	0.048	0.065	0.0014	0.114	0.310	0.038	0.348
I-8	0.076	0.064	0.0010	0.141	0.288	0.080	0.368
I-9	0.056	0.066	0.0011	0.123	0.311	0.028	0.339
I-10	0.052	0.062	0.0014	0.115	0.306	0.062	0.368
I-11	0.060	0.064	0.0013	0.125	0.282	0.097	0.379
I-12	0.068	0.066	0.0013	0.135	0.326	0.091	0.417
I-13	0.064	0.067	0.0010	0.132	0.301	0.087	0.388
I-14	0.048	0.064	0.0012	0.113	0.296	0.029	0.325
I-15	0.060	0.068	0.0013	0.129	0.284	0.059	0.343
I-16	0.044	0.065	0.0011	0.110	0.302	0.066	0.368
I-17	0.072	0.062	0.0011	0.135	0.312	0.048	0.360
I-18	0.048	0.063	0.0014	0.112	0.322	0.086	0.408
I-19	0.076	0.066	0.0012	0.143	0.285	0.080	0.365
I-20	0.076	0.064	0.0014	0.141	0.320	0.064	0.384
I-21	0.056	0.063	0.0011	0.120	0.347	0.021	0.368
I-22	0.060	0.061	0.0012	0.122	0.350	0.063	0.413
I-23	0.052	0.065	0.0013	0.118	0.307	0.023	0.330
I-24	0.044	0.068	0.0010	0.113	0.283	0.094	0.377
I-25	0.056	0.062	0.0014	0.119	0.286	0.034	0.320
Average	0.059	0.065	0.0012	0.125	0.310	0.063	0.373

false points may be viewed as true feature points. Thus, it can be concluded that a MLFRFT-based HT with a sufficient resolution is more efficient than the SHT.

Although the HT-based line detection is mostly reported in off-line image understanding problems, it has also been suggested in many real-time applications. Lane marker detection in driver assistance systems is one of the real-time applications of the HT. The MLFRFT-based HT has been applied to benchmark road images [13] to test its performance in response to real-world challenges.

3.5 Application to lane marker detection

For a human driver, road boundaries, lane marks, road colour and texture are key navigational tools in any weather condition. Thus, one would expect an autonomous system to benefit from the same guidelines and follow the same principles as humans. That is why vision-based imaging has always been one of the prominent sensing modalities for such systems. All of the existing system perception problems can be categorised into two groups: namely, lane and road detection and obstacle detection [15]. To incorporate line detection with the MLFRFT-based HT method, only the former task has been considered.

Lane marker detection

Lane detection and perception prevents unintentional departure from the lanes by simultaneous detection and tracking of the position and orientation of the lane markers and sending a warning messages to the driver. Therefore, precise line detection and segmentation plays a vital role in the overall performance of the system [13, 14, 15, 16]. Several works have addressed the problem using HT-based methods [111, 16, 112, 113, 114, 13]. Among them, the *Advanced Lane Detector 2.0* (ALD 2.0) [13] showed a superior perfor-

mance in dealing with real-time video sequences. The ALD is robust and accurate, however, it is based on the SHT to extract the linear structures. An image (frame taken from a video sequence) undergoes a three-step pre-processing, and then binarisation prior to a low-resolution SHT and peak detection. A two-step post-processing is performed afterwards to validate the true lane markers. Apart from the pre and post-processing operations to assure the quality and reliability of the lane detection, the SHT takes the major responsibility in lane marker detection and that is straight-line detection.

Therefore, the MLFRFT-based HT has been suggested to replace the SHT. The principal aim is to incorporate the MLFRFT-based HT in the lane detection to avoid the binarisation of greyscale images and improve the computational time whilst guarantee the detection accuracy.

The lane detection consists of the following steps:

1. *Region of Interest selection (ROI).* This area is the portion of an image that has the highest probability of encompassing the lane markers. The Region of Interest is typically selected between the vanishing point and the vehicle's hood.

2. *Temporal blurring.* This is to assure that the SHT can detect the short dashed lane markers when they appear. Temporal blurring connects the broken lane markers by producing an average image of current frame and a number of predecessor frames.

3. *MLFRFT-based HT.* The image is split into halves and the MLFRFT-based HT is applied to each half separately.

4. *Peak detection.* The HT peak detection determines the potential straight-line candidates for the lane markers.

Experiments were performed using the publicly accessible database provided by the authors in [13]. Diverse driving scenes and road conditions were

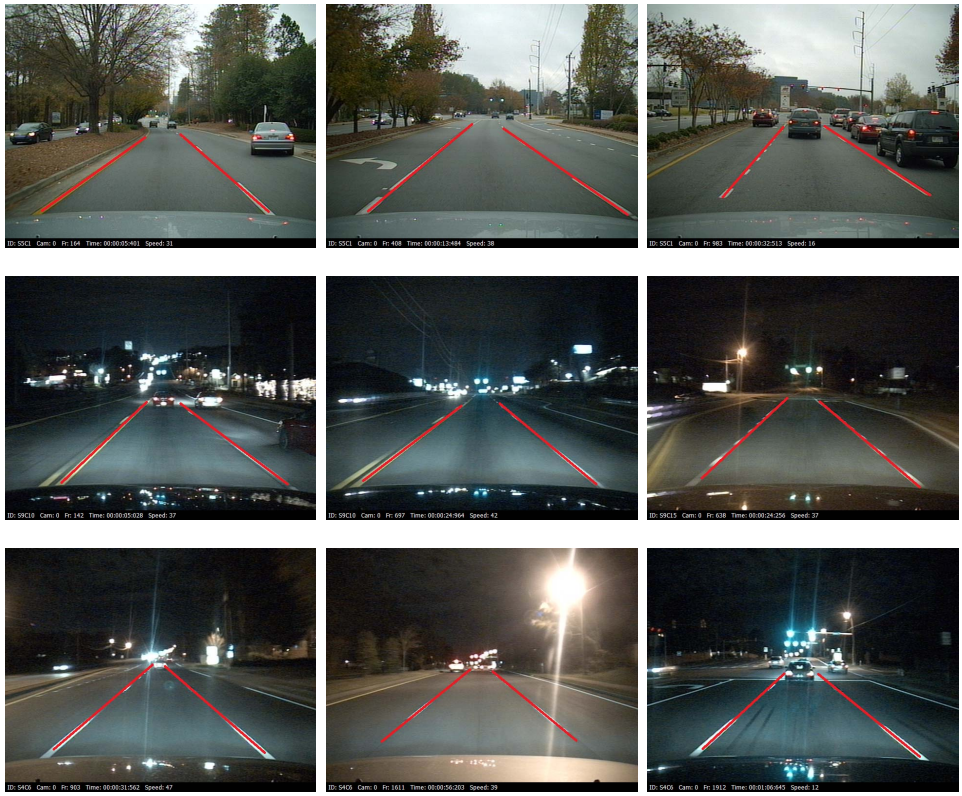


Figure 3.7: Lane detection using MLFRFT-based HT in diverse driving scenes and road conditions

chosen with distinct illumination levels. Figure 3.7 illustrates a number of detection results obtain via the MLFRFT-based HT method.

The computational time is approximately 0.14 seconds for each frame. In theory, 7 frames can be processed, however in some driving scenes, especially in daytime where the level of illumination varies between the surrounding objects, the processing time can go up to 0.2 seconds at the highest. As a result, in average, three to four frames per second can be guaranteed to be processed. Edge detection itself could at least double the amount of computational time. As reported in [13], the ALD 2.0 processes each frame in 0.8 seconds including the post processing steps.

3.6 Summary

To perform line segmentation it is necessary to detect the position and orientation (ρ, θ) of the potential straight-line segment before identifying its endpoints. As the first step for the proposed domain-crossing approach to line segmentation, i.e. the global phase, a MLFRFT-based HT was suggested in this chapter due to the following two superior capabilities:

1. High-resolution frequency sampling as a result of the multiple instances of non-integer frequency grid that leads to higher precision in line parameterisation than the SHT.

2. More efficient computational complexity, (three times faster than the SHT). The MLFRFT can be computed in a similar fashion to the FFT by simply adding the fractional scaler γ to the exponential component of the Fourier transform. Thus complexity of $O(N^2 \log_2 N)$ still holds for the MLFRFT.

Utilising the MLFRFT was preferred instead of GIFT (generalised interpolated Fourier transform), despite the more flexible design of GIFT's frequency samples. GIFT cannot deliver higher precision but only delivers a customised frequency grid with tedious parameter tuning.

Due to the popularity of HT-based methods in lane marker detection in driver assistance systems the MLFRFT was suggested as a replacement for the traditional SHT.

Despite the fact that the MLFRFT is an accurate and efficient technique to extract linear profiles in a noisy or degraded image, it is unable to distinguish between straight-lines and straight-line segments. It can only identify straight-line parameters (ρ, θ) resulting in lines passing through the entire image. To identify the segment endpoints a domain-crossing approach has been introduced in the following chapter.

Chapter 4

Parallel butterfly decomposition for line segmentation

The MLFRFT-based HT yields straight-line parameters (ρ, θ) for each potential line segment. These lines have to be further processed to extract segment endpoint information. In this chapter the notion of line segmentation using butterfly boundary information is discussed and an accurate algorithm (LSBB) is proposed based on Least Square curve fitting and the intersection of three points. The idea has been further extended for multiple segments in an image by a parallel windowing approach (PWA-HT) and a robust disturbance elimination technique to remove the effect of spurious feature points. Experimental results are given in the next chapter to verify the performance of the proposed algorithm and compare it with state-of-the-art representative works.

4.1 Line segmentation using butterfly boundary (LSBB)

4.1.1 Fundamentals

From the HT point of view, each feature point in the image space corresponds to a sinusoidal curve in the Hough space. Consequently, all of the points on a straight-line represented by $\rho_0 = x\cos\theta_0 + y\sin\theta_0$ in image space form a butterfly shape illustrating a collection of sinusoids in the Hough space. These sinusoids have one communal intersection point at the peak (ρ_0, θ_0). Accordingly, the first and the last sinusoids trajecting through the boundary of the butterfly represent the two segment endpoints. Therefore, the problem of finding the endpoints in image space turns into the problem of finding two boundary sinusoids in the Hough space. This is the principal idea behind line segment detection using butterfly boundary points. Let us clarify the concept of butterfly boundary sinusoids and explain the reason why finding these sinusoids can lead us to detect the segment endpoints.

From image to Hough space

Consider the process of butterfly formation, starting from a single pixel and then expanding the notion to a number of collinear pixels forming a continuous line segment. Figure 4.1 shows a step by step generation of a line segment from a single pixel p_1, (top-left), to a line segment L, (bottom-left). The HT of each step is depicted in the right-hand side column of the figure, where each newly generated sinusoid is labelled. The index of each feature point p_i is related to its sinusoid s_i. Accordingly, p_1 generates the sinusoid s_1, p_2 generates the sinusoid s_2, and so on. In fact, the two extremes, p_1 and p_2 represent the two endpoints of the segment L. It can be seen how the sinusoids of middle points, p_3, p_4 and p_5 and the rest, fill in between the boundary sinusoids s_1 and s_2 in the Hough space. Therefore, these sinusoids

lying on the boundary of the butterfly contain significant information, useful for endpoint detection.

From Hough to image space

Taking s_1 in Figure 4.1 into consideration (top-right), each point (ρ_i, θ_i) along this sinusoid represents a straight-line $\rho_i = x\cos\theta_i + y\sin\theta_i$ in image space, where θ_i is the angle of the normal to the line and ρ_i is the distance of the normal from the origin to the line. Therefore, the combination of all lines represented by s_1 have one common intersection point in image space where they all cross over each other, which is, in fact, p_1.

Theoretically, the intersection of two points captured on a sinusoid are sufficient to find their corresponding feature point in image space [62], however, this may not satisfy the required accuracy. Let (ρ_1, θ_1) and (ρ_2, θ_2) be the coordinates of two points lying on s_1, as shown in the top-right of Figure 4.1. The intersection of the straight-lines represented by these points, i.e. l_1 and l_2, in image space is obtained by solving the following two simultaneous equations with two unknowns:

$$\rho_1 = x_1\cos\theta_1 + y_1\sin\theta_1 \tag{4.1}$$

$$\rho_2 = x_1\cos\theta_2 + y_1\sin\theta_2 \tag{4.2}$$

where (x_1, y_1), the solution of the equations, is the coordinate of the target feature point p_1. Simply speaking, to detect p_1 in image space it is sufficient to pick up two points such as (ρ_1, θ_1) and (ρ_2, θ_2) along its sinusoid s_1 and find the intersection of the lines represented by them, l_1 and l_2, in image space. The intersection of l_1 and l_2 occurs at the location of p_1.

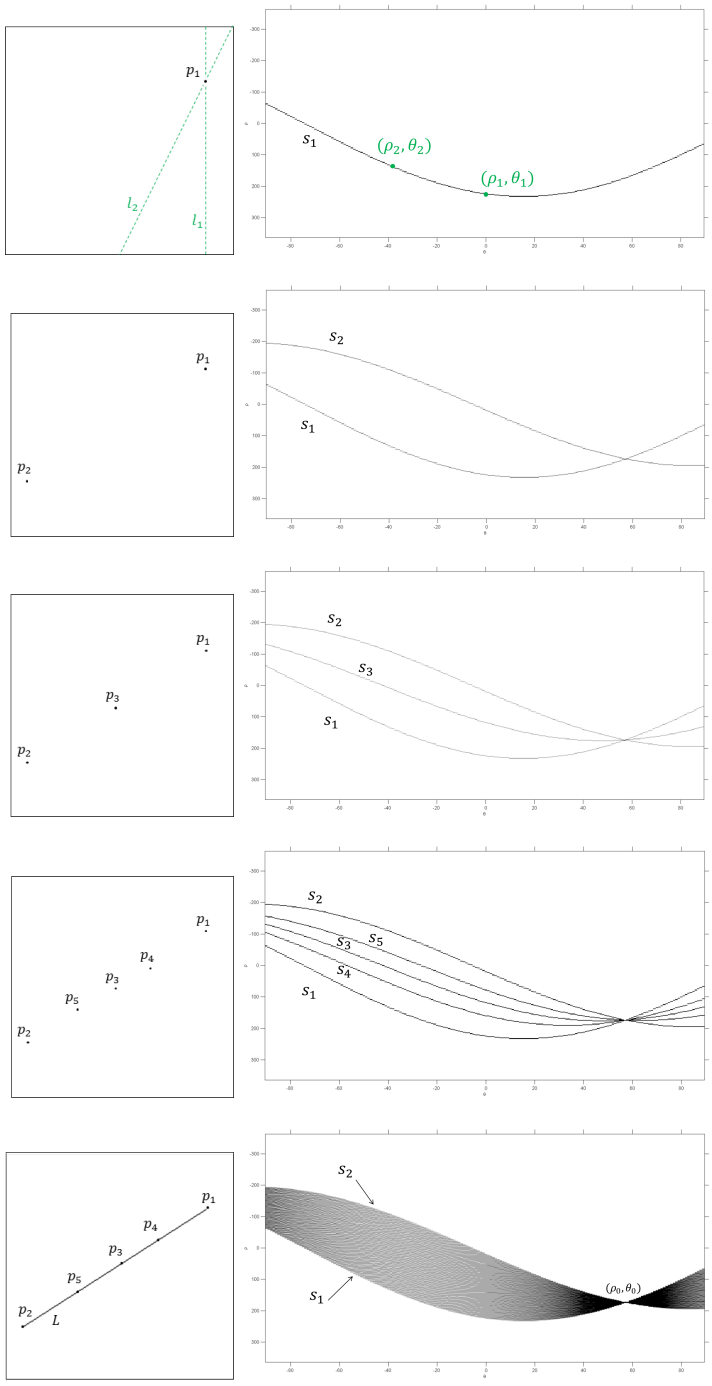


Figure 4.1: Step by step generation of a butterfly.

4.1.2 Endpoint detection

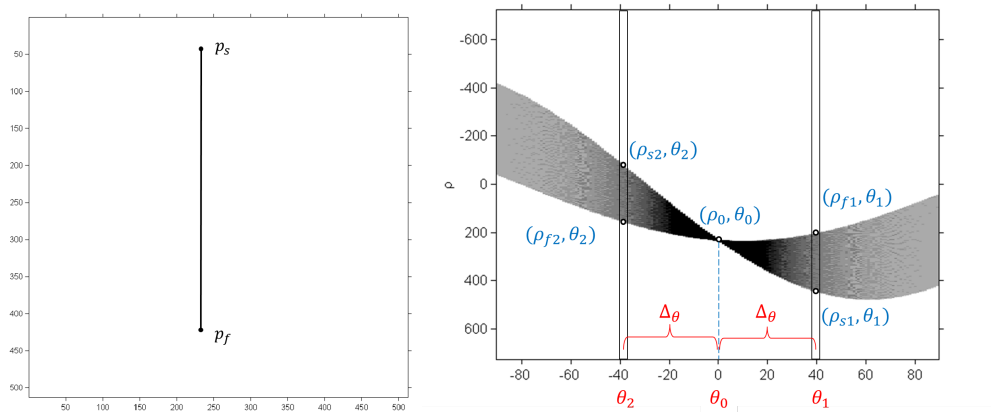
The concept of two points intersecting can be extended to every other sinusoid of a butterfly. However, since we are interested only in identifying the two endpoints, p_1 and p_2, two points have to be captured along each trajectory s_1 and s_2, i.e. two sinusoids that lie on the boundary of the butterfly wings (bottom-right image of Figure 4.1). Afterwards, the intersection of each two points can be found in the image domain to detect the endpoints.

Consider a vertical line segment and its corresponding HT butterfly shown in Figure 4.2 (a) and (b), respectively. (ρ_0, θ_0) represents the peak where all of the sinusoids cross over each other. Let us define two column vectors θ_1 and θ_2 on either side of the peak along the θ axis with $\Delta\theta$ as an interval between θ_0 and each of the columns. Boundary points on the butterfly wings can then be identified by seeking the *rising* and *falling* edges, i.e. the first and last non-zero elements, along each of the column vectors. In our notation, (ρ_{f1}, θ_1) and (ρ_{f2}, θ_2) are the two points corresponding to the **first** endpoint, hence the subscript f. Similarly, (ρ_{s1}, θ_1) and (ρ_{s2}, θ_2) are the two points corresponding to the **second** endpoint in the image domain, hence the subscript s. Once boundary points are captured, simultaneous equations 4.1 and 4.2 can be solved to localise the endpoint in image space.

4.1.3 Robust Least Squares curve fitting

In practice, acquiring such a clear butterfly is nearly impossible due to noise and disturbance. This may result in having a number of adjacent sinusoids at the location of the butterfly boundary and therefore causing confusion in choosing the true boundary sinusoids. As a result, a Least Squares curve fitting method is suggested in this research to approximate the true boundary sinusoids when having a number of adjacent sinusoids.

Having multiple columns available on either sides of the peak, a number



(a) (b)

Figure 4.2: (a) Perpendicular line segment. (b) The HT butterfly. Sinusoids trajecting through the boundary correspond to segment endpoints.

of boundary points can be obtained. Let us define

$$(\rho_1, \theta_1), (\rho_2, \theta_2), ..., (\rho_k, \theta_k) \tag{4.3}$$

as a number of data points captured along each boundary sinusoid, where k is the number of specified columns. These data points can be fitted into a curve using a Least Squares curve fitting method. However, one of the main disadvantages of least squares approaches, such as Linear Least Squares and Weighted Least Squares, is their sensitivity to outliers. Outliers are the extreme data points that should not be taken into consideration. As for a butterfly, noise and disturbance can cause outliers to appear. The problem is when the effect of these outliers gets magnified by squaring the residuals and this degrades the fitted curve. As a result, to minimise their influence a Robust Least Squares curve fitting based on bisquare weights [115, 116] has been used in this research. This approach minimises a weighted sum of squares, where the weight given to each data point depends on its distance to the fitted curve. Points that are closer to the curve receive higher

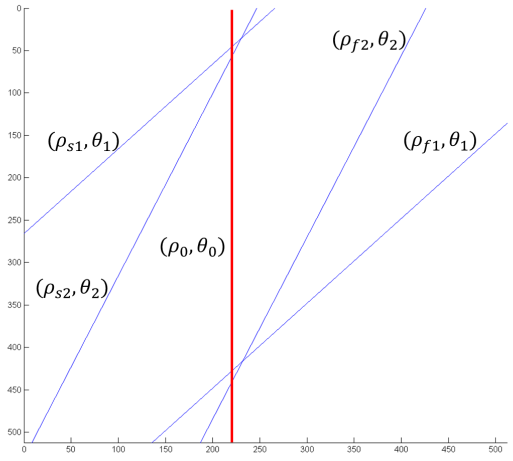


Figure 4.3: Using the intersection point of three line. Intersections between the line represented by the boundary points (ρ_{f1}, θ_1), (ρ_{f2}, θ_2), (ρ_{s1}, θ_1), (ρ_{s2}, θ_2) and the target straight-line (ρ_0, θ_0) in the image domain represent the endpoints.

weight, whereas points that are further away from the curve receive lower weight [117]. Therefore, any two points selected on the approximated curves (the first and last curves representing the boundary) would be sufficient to represent the endpoints.

4.1.4 Intersection of three points

Detection accuracy can be further improved by using intersection points of three lines for each endpoint. This is done by utilising the peak coordinates (ρ_0, θ_0), without imposing any extra computational burden. Note that peak information should not be neglected but used alongside the butterfly boundary information. It is easy to further increase the reliability of the result by introducing a third straight line. Because the peak corresponds to the line passing through both of the endpoints, it can be considered as a valuable criterion. Note that, as a part of the MLFRFT processing steps, this peak will

be captured via the peak detection process; therefore, no extra computation will be imposed to the system.

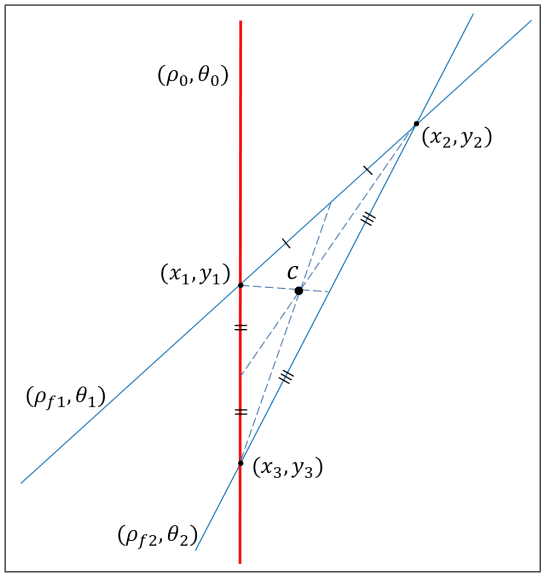


Figure 4.4: Figure 4.3 is cropped and the intersection area for p_f is enlarged. The intersection area forms a triangular shape. The true endpoint is defined as the geometrical centre of the triangle.

Figure 4.3 shows the intersection of the lines corresponding to the boundary points and the peak (coloured in red) in the image domain. In theory, straight-lines represented by (ρ_{f1}, θ_1), (ρ_{f2}, θ_2) and (ρ_0, θ_0) have one communal intersection which is the first endpoint (p_f in Figure 4.2(a)). The same scenario applies for the straight-lines represented by (ρ_{s1}, θ_1), (ρ_{s2}, θ_2) and (ρ_0, θ_0) to detect the second endpoint (p_s in Figure 4.2(a)). However, in practice, these lines do not meet at an exact point and a triangular shape forms in the intersection area. This is due to rounding errors caused by discretisation of the Hough space.

For instance, if we crop the Figure 4.3 and enlarge the intersection area of p_f, a triangular shape can be seen as shown in Figure 4.4. A fair approximation would be to choose the centroid of the triangle as the detected

endpoint. Geometric barycentre or centroid of a triangle is the average (arithmetic mean) position of the three vertices. Simply speaking, it is a single point where the three medians[1] of a triangle intersect. If (x_1, y_1), (x_2, y_2) and (x_3, y_3) are three vertices of the triangle then the centroid is given by

$$c = \left(\frac{(x_1 + x_2 + x_3)}{3}, \frac{(y_1 + y_2 + y_3)}{3} \right) \tag{4.4}$$

A formal statement for the LSBB algorithm is given as follows:

Step 1) Curve fitting. A number of adjacent sinusoids on each boundary will be fitted into a single curve using a Robust Least Squares method. This is done by choosing multiple columns along θ. Empirical studies showed twenty columns on each side of the peak can provide sufficient accuracy for curve fitting.

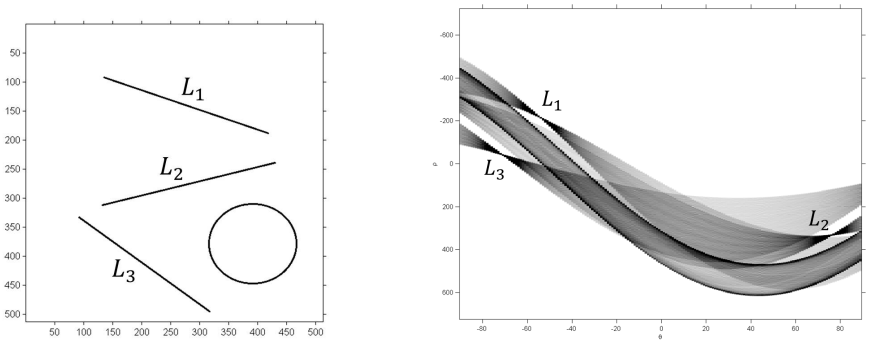
Step 2) Boundary point selection. Define the Δ_θ and seek the rising and falling edges along the two column vectors θ_1 and θ_2 and extract (ρ_{f1}, θ_1), (ρ_{f2}, θ_2), (ρ_{s1}, θ_1) and (ρ_{s2}, θ_2).

Step 3) Solving simultaneous equations. Find the line intersections of straight-lines represented by (ρ_{f1}, θ_1), (ρ_{f2}, θ_2), and (ρ_0, θ_0) in the image domain. A triangular shape appears in the intersection area. The target endpoint is found by locating the centroid of the triangle using Equation 4.4.

Step 4) Repeat Step 3 for the second endpoint. Solving the simultaneous equations to find the intersections of (ρ_{s1}, θ_1), (ρ_{s2}, θ_2), and (ρ_0, θ_0).

Step 5) Draw the segment. Connect the two endpoints to identify the straight-line segment.

[1]Median of a triangle is a straight line cutting through a vertex and the midpoint of the opposite side, and splits the triangle into two equal areas



(a) (b)

Figure 4.5: (a) Mixed shape image. (b) Its overlapping sinogram.

Problem

Despite its accuracy, the LSBB algorithm is highly dependent on the number of lines in the image space and detection precision can be unfavourably affected when the number of segments increases. This will lead to a complex Hough space with superimposed butterfly wings. That is a recurrent problem in image processing applications because there are typically many line segments in an image. For instance, Figure 4.5(a) shows a test image reported in [70, 2, 71], consisting of a mixture of three distinguishable straight-line segments and a circle, with its corresponding superimposed butterflies in Figure 4.5(b). It can be seen, that the sinusoids of L_2 are overlapped with those of L_1, L_3, and the circle. Accordingly, seeking boundary points for each peak will be a problematic task and will eventually yield false endpoints. In fact, the essential condition for the LSBB algorithm is to have a transparent single butterfly when the impressions of the other peaks are eliminated. This suggests decomposing the Hough space in a way that, each butterfly appears in a separate Hough space.

Solution

Basically, each butterfly has to be treated **separately**. There are two ways to do this: either operations in Hough space or enhancements in image space. To apply operations in Hough space a typical solution could be to define a "window-of-interest" around each peak and its surrounding sinusoids [63, 64]; however, it would not guarantee the complete elimination of adjacent peaks and their overlapping sinusoids. Instead for each peak a window can be defined in image space around the straight-line representing that peak. The concept of neighbourhood of a straight-line can be used to construct such a window in the image plane [67]. However, one must go through the entire process of building a proper neighbourhood and compromise on the radius selection for each segment. The neighbourhood approach also requires tedious parameter tuning. In the following, a new window-of-interest in the image domain is defined and the butterfly of each segment is isolated without neighbourhood identification. Because computation of each window and its corresponding butterfly is independent the algorithm can be implemented in parallel.

4.2 Parallel windowing approach (PWA)

The aim is to isolate the butterflies in a way that each of them appears in a separate sub-Hough space. Having the straight-line parameters (ρ_i, θ_i) obtained via the MLFRFT-based HT, a set of windows can be defined in the *spatial domain* to encompass the area surrounding the lines. Note that the number of windows is equal to the number of peaks, i.e. one window for each detected straight line.

Let us define L_0 with the parameters (ρ_0, θ_0) as a straight-line passing through the entire image. To form a window around L_0, a predefined parameter μ is used as the width of this window. We can select μ as $t + 2$ pixels where t is the maximum line thickness in pixels. The length of the window

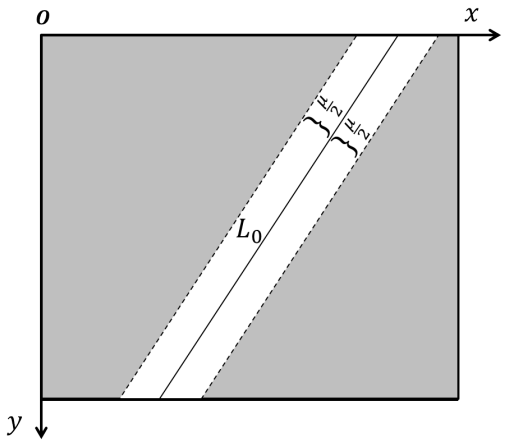


Figure 4.6: Illustration of window-of-interest in image plane for straight-line L_0. μ is the width of the window

is restricted to the image boundary. Figure. 4.6 illustrates the definition of the window-of-interest in the image plane. It can be seen that the proposed window forms a parallelogram encompassing the entire straight-line. To acquire a sub-HT, only feature points inside the window will be mapped into the Hough space, meaning that any other feature points outside the window will be neglected in butterfly formation. The resulting Hough space has the same size as the original HT, but it contains only the sinusoids that belong to the feature points falling inside the parallelogram. A block diagram of the PWA-HT process is drawn in Figure. 4.7.

It should be emphasized that, because computing each decomposed Hough space is an independent process, it is easy to calculate the whole set of sub-Hough spaces in parallel using multi-core CPU systems. Parallel implementation of the algorithm can help reduce the computational cost significantly. This ability is one of the important characteristics of the proposed method.

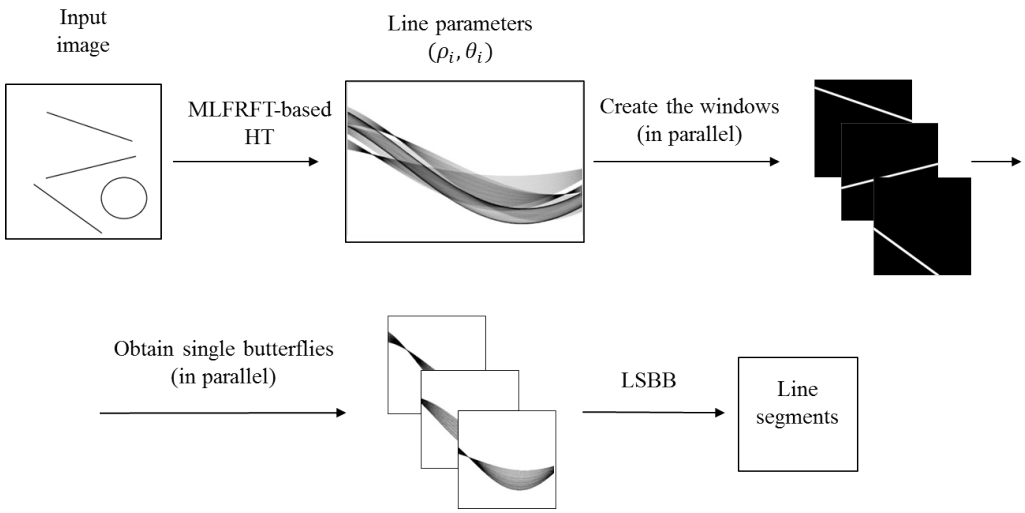


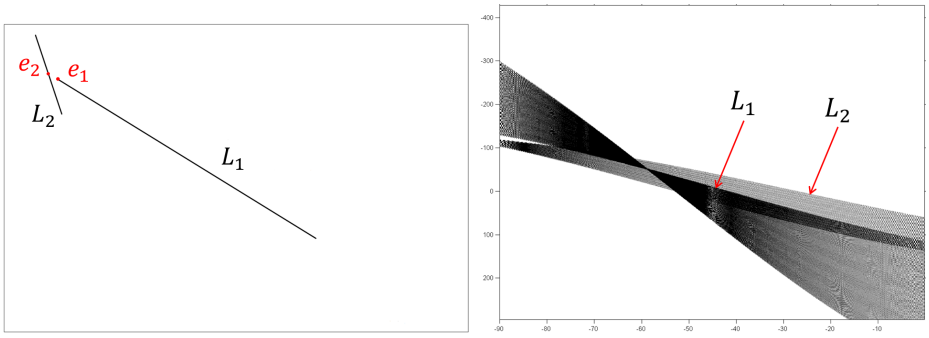
Figure 4.7: Block diagram of the proposed parallel windowing approach to HT butterfly decomposition (PWA-HT) for line segment detection.

4.3 Disturbance elimination

Although the windowing approach is capable of removing the disturbing sinusoids caused by the feature points outside the window, it cannot guarantee that all of the remaining feature points inside the window belong to the same segment or to a number of collinear segments. In other words, the remaining feature points inside the window may be either collinear or non-collinear. Collinear feature points will contribute to the formation of the butterfly. However, non-collinear feature points inside the window will cause disturbing sinusoids. Such disturbance will degrade the detection accuracy and lead to a false detection and therefore has to be excluded.

To clarify the problem, let us consider a test image and its corresponding sub-HT as shown in Figure. 4.8. Figure. 4.8(a) shows a synthetic image with segment L_1 as the target straight-line segment, to be detected. Figure. 4.8(b) shows the corresponding sub-HT when the effect of disturbing feature points outside the window is eliminated thanks to the windowing approach.

It can be seen from Figure. 4.8(b) that, even after applying the image-



(a) (b)

Figure 4.8: Problem of disturbing feature points inside the window. (a) Original image with target segment L_1. (b) Butterfly captured using windowing approach. Sinusoids of L_2 are still superimposing with those of L_1.

domain window, sinusoids of L_2, (the line which intersects with L_1's extension), still superimpose with those of L_1. Consequently, seeking the rising edge will mislead us to admit e_2 (in Figure. 4.8(a)) as L_1's first endpoint rather than the true endpoint which is e_1. The problem occurs when the target segment, or its extension, intersects with one or more non-collinear segments within the image boundary. As a result, any non-collinear feature points inside the window will also get voted, alongside the collinear feature points, during the sub-HT process (see the shadow of L_2 in Figure. 4.8(b)). Therefore, the essential condition is to have the effect of any disturbing feature point inside the window excluded. This suggests an essential enhancement and filtering technique to exclude the remaining non-collinear feature points. This enhancement has to be performed after the MLFRFT-based HT, thus, line parameters, i.e. ρ_i and θ_i are known. Basically, the algorithm can be explained in two major steps:

Step 1) Seek the lines which intersect with the target segment or its extension. The intersection has to occur within the image coordinates. This is straightforward because line parameters, i.e. $(\vec{\rho}, \vec{\theta})$, are known, thanks to the MLFRFT-based HT. Having the parameters, straight-line

equations can be constructed by $\rho_i = x\cos\theta_i + y\sin\theta_i$. Thus, intersecting lines can be found by solving a series of simultaneous equations. Algorithm 3 synthesizes the pseudo-code of how to find and store the lines having intersection within image.

Step 2) For each intersecting line of every target segment construct a mask window W_m with width r_m. This time mask the feature points inside the window by assigning every pixel to zero. Note that, such a window plays the role of a mask-filter therefore the r_m can be as small as 3 to 5 pixels. Algorithm 4 summarizes the procedure.

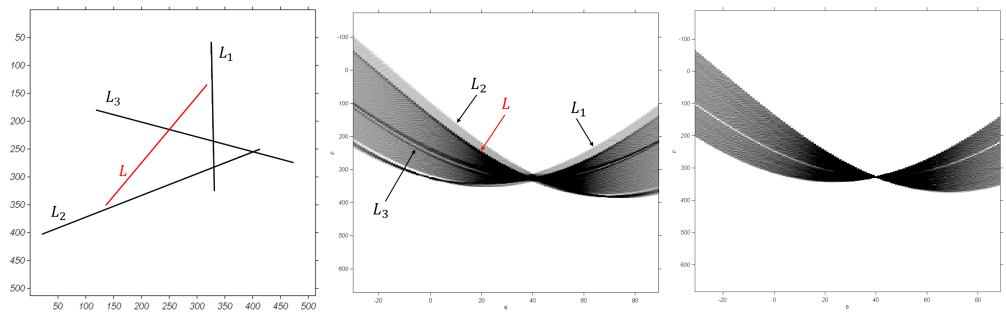
Algorithm 3 Find and store the lines having intersection within image

Input: $\vec{\rho}, \vec{\theta}$ /* From the MLFRFT-based HT /*
$M = [0]_{n\times n}$ /* n is the total number of lines /*
for all i such that $1 \leq i \leq (n-1)$ **do**
    for all j such that $(i+1) \leq j \leq n$ **do**
        $x \leftarrow (\rho_i \sin\theta_j - \rho_j \sin\theta_i)/(\cos\theta_i \sin\theta_j - \cos\theta_j \sin\theta_i)$
        $y \leftarrow (-\rho_i \cos\theta_j + \rho_j \cos\theta_i)/(\cos\theta_i \sin\theta_j - \cos\theta_j \sin\theta_i)$
        /* Gets the coordinates of the intersection point /*
        if (x, y) is inside image coordinates **then**
            $M[i, j] \leftarrow$ True
        else
            $M[i, j] \leftarrow$ False
        end if
    end for
end for
$M \leftarrow M + M^T$
Output: M /* intersection matrix /*

Let us consider the target straight-line segment L (coloured in red) as shown in Figure 4.9(a) and non-collinear segments L_1, L_2, and L_3. L_3 has intersection with L, whereas, L_2, and L_3 are have intersections with extensions of L. Figure 4.9(b) shows the result of the windowing approach without disturbance elimination. It can be seen that L_1 and L_2 affect the

Algorithm 4 Create the window (mask-filter) for each line and compute the sub-HT.

Input: Original image I,
$P = 0, H = 0$
for all i such that $1 \leq i \leq n$ **do** /* for every straight-line/*
    /* Construct the window W_i for straight-line (ρ_i, θ_i) /*
    /* and exclude the feature points outside the window /*
    /* I_i is the original image when masked with W_i
    Compute $I_i = I \ \& \ W_i$
    $P \leftarrow \texttt{find}(M[i])$ /* Returns indexes of intersecting lines for ith line /*
    if $P \neq 0$ **then**
        for all j such that $1 \leq j \leq$ length of P **do**
            /* Construct the mask window W_m /*
            Compute $I_i = I_i \ \& \ \overline{W_m}$
        end for
    end if
    Compute $H_i = \texttt{hough}(I_i)$ /* Returns the sub-HT for the ith target segment /*
end for
Output: H



(a) (b) (c)

Figure 4.9: Comparison between the sub-HT butterflies. (a) Synthetic image of 512×512 pixels. (b) Butterfly captured without disturbance elimination. (c) Butterfly captured via improved windowing approach.

butterfly's significant edge. That will eventually lead to a large error. In Figure 4.9(c) these sinusoids have been completely removed thanks to the disturbance elimination algorithm explained above.

Chapter 5

Experimental results and analysis

In this chapter the performance of the proposed algorithm is verified in terms of endpoint detection accuracy and robustness in the presence of noise. The experiments were conducted on synthetic as well as real-word images reported in the recent literature.

5.1 Endpoint detection accuracy

This experiment aims to verify the segmentation accuracy and compare the results with the ground truth location of the endpoints. An image of four line segments with different thickness, i.e. 2 and 8 pixels, mixed with two curvilinear objects has been chosen. Figure 5.1(a) shows the test image of size (489×609) with its corresponding butterflies in Figure 5.1(b). It can be seen that the sinusoids of line segments overlap with each other and with those of the curvilinear shapes, making the detection of boundary points a problematic task. The result of line detection using a MLFRFT-based HT is shown in Figure 5.1(c) where all of the potential straight-lines are found. Using the proposed parallel windowing approach, the butterflies are

divided into four isolated and single butterfly Hough spaces. The four isolated butterflies of S_1, S_2, S_3, and S_4, are depicted in Figures. 5.1(d), (e), (f), and (g), respectively. Note that the sinusoids corresponding to the curvilinear structures do not appear in the isolated butterflies, because these sinusoids did not contribute to any peak. The result of endpoint detection using a PWA-HT is shown in Figure 5.1(h).

Qualitative analysis given in Table 5.1 provides a better insight into the accuracy of PWA-HT in endpoint detection. Obtained via the MLFRFT-based HT, (ρ_i, θ_i) refers to the peaks in the Hough space, where $-780 \leq \rho_i \leq 780$ and $-\pi/2 \leq \theta_i < \pi/2$. Each of these peaks correspond to a straight-line in the image domain as shown in Figure 5.1(c) . The location of the endpoints for each segment is denoted in (x, y) coordinates where the origin is set to the top-left corner of the image. The detected location of the endpoints has been compared with the ground-truth data. Euclidean distance from the detected endpoint to its ground-truth position has been used to measure the error, i.e. deviation from ground-truth.

Based on the observations and keeping the size of the image and the length of the segments in mind, the PWA-HT shows the average deviation from ground truth data to be 2-3 pixels error for lines with thickness of 2 pixels and 5-6 pixels error for the lines with thickness of 8 pixels. Minor deviation of the detected endpoints from the ground-truth coordinates illustrates an acceptable accuracy of the proposed butterfly isolation method regardless of line thickness and having curvilinear structure. However, it is worth drawing the reader's attention to the sensitivity of accuracy to segments with different thickness. Detection accuracy degrades as the width of the segment grows. For the segments S_1 and S_2 with thicker width, deviation from the ground-truth data is larger than the other two. This is indeed expected because there does not exist one single pixel that represents an endpoint for a thick segment but, instead, a number of pixels. Since the central pixel has been chosen as the ground-truth position of each endpoint, the detected endpoint

may not lie on the same pixel.

Table 5.1: Evaluation on endpoint detection accuracy. Qualitative results of Figure 5.1

Line segment			S_1	S_2	S_3	S_4
Peak (ρ_i, θ_i)			(-70,-41)	(262,58)	(-4,-61)	(438,43)
Ground-truth data	First endpoint		(49,160)	(37,288)	(41,27)	(78,563)
	Second endpoint		(415,583)	(449,29)	(448,253)	(448,160)
Detection results	First endpoint	Detected coordinates	(53,157)	(34,283)	(43,28)	(80,561)
		Deviation from ground-truth	5	5.8	2.2	2.8
	Second endpoint	Detected coordinates	(412,580)	(446,33)	(445,254)	(447,158)
		Deviation from ground-truth	4.2	6.4	3.1	2.2

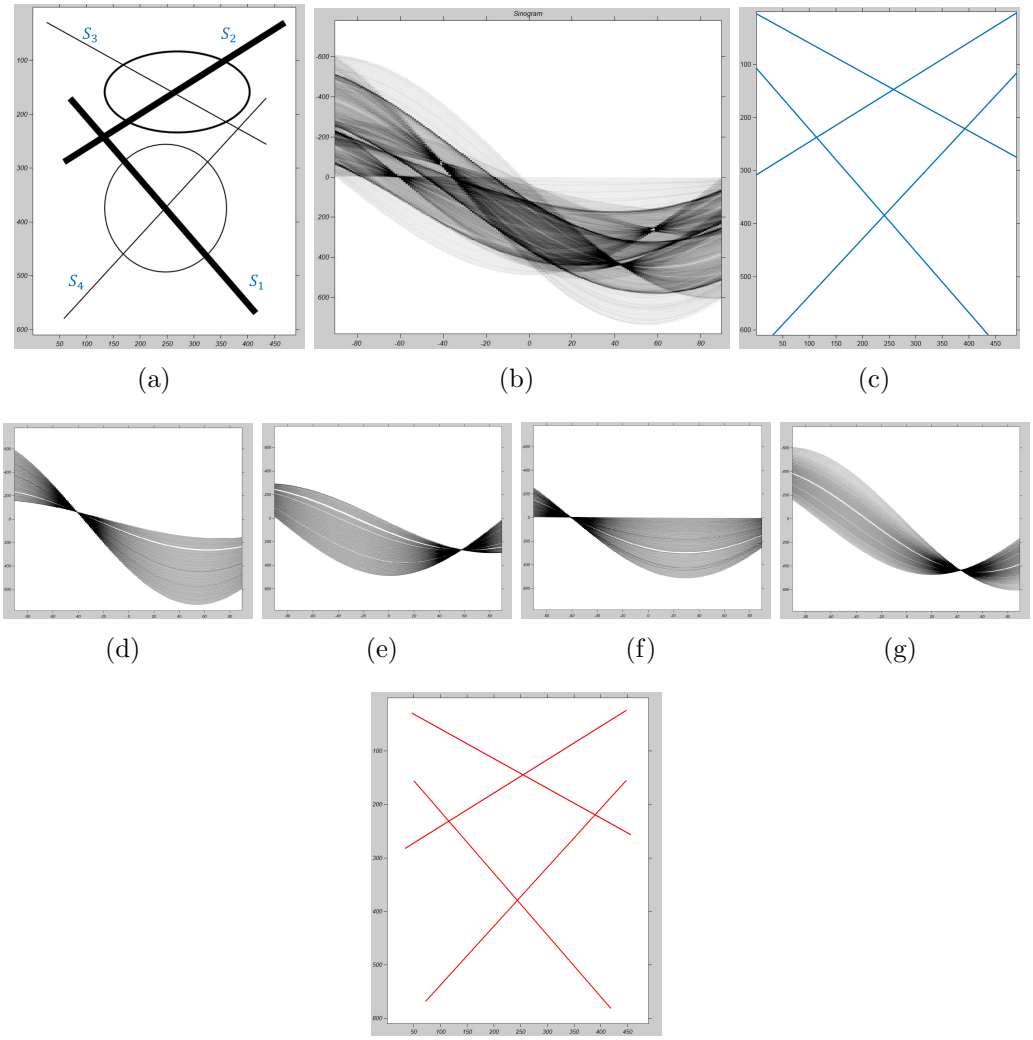


Figure 5.1: (a) Mixed shape and mixed thickness image of size (489 × 609). (b) The overlapping butterflies. (c) Result of MLFRFT-based HT. (d)-(g) Isolated butterflies of S_1, S_2, S_3, and S_4, respectively. (g) Detected segments PWA-HT.

5.2 Disturbance elimination

To evaluate the performance of the proposed algorithm with and without the disturbance elimination technique explained in Section 4.3, a detailed qualitative analysis has been performed in this experiment. A real greyscale image of size (512×512) with ground-truth information of the number of segments and the coordinates of the endpoints has been used, as shown in Figure 5.2(a). Target segments S_1 to S_6 have been labelled for the sake of clarity. The result of an MLFRFT-based HT is depicted in Figure 5.2(c), where all of the straight-lines have been detected. L_1 to L_6 are the corresponding straight-lines passing through S_1 to S_6, respectively. Algorithm 3 generates the line intersection matrix M given in Table 5.2, where lines that are intersecting with each target segment are identified for further processing.

Table 5.2: Intersection matrix M as a result of Algorithm3. Lines that are intersecting with each target segment are identified.

	L_1	L_2	L_3	L_4	L_5	L_6
L_1	0	0	1	1	0	0
L_2	0	0	1	1	0	0
L_3	1	1	0	0	1	1
L_4	1	1	0	0	1	1
L_5	0	0	1	1	0	0
L_6	0	0	1	1	0	0

Figure 5.2(d)-(i) illustrates the result of the windowing approach without applying the disturbance elimination method. Shadows of disturbing feature points that are visible around the butterfly wings will eventually lead to a large deviation from the ground-truth endpoints. Taking the butterfly of S_1 in Figure 5.2(d) as an instance, the shadows surrounding the butterfly peak and the wings are the consequence of feature points of S_3 and S_4 which intersect with S_1, (refer to Table 5.2). The results of line segmentation without disturbance elimination is shown in Figure 5.2(p) where the considerable effect of S_3 and S_4 on the endpoints of S_1 and S_2 is visible. Fig-

ure 5.2(j)-(o) represent the result of the improved windowing approach using the disturbance elimination algorithm, where the disturbing feature points have been removed using Algorithm 4. The segmentation result is depicted in Figure 5.2(q).

Table 5.3 provides a detailed comparison between the ground-truth information and the detected endpoint. Euclidean distance from the detected endpoint to the true endpoint has been measured as the deviation criterion. When the disturbance elimination is applied, the error is as small as two to three pixels distance.

Table 5.3: Detailed comparison of the method with the ground-truth data. The PWA-HT with disturbance elimination has a lower deviation from the ground-truth (GT) data.

Line segment			S_1	S_2	S_3	S_4	S_5	S_6
Ground-truth data	First endpoint		(46,452)	(80,502)	(315,206)	(234,266)	(233,264)	(302,352)
	Second endpoint		(243,282)	(290,335)	(378,287)	(301,352)	(310,204)	(379,290)
Without disturbance elimination	First endpoint	Detected coordinates	(51,448)	(84,497)	(308,200)	(233,260)	(232,266)	(300,353)
		Deviation from GT	6.4	6.4	9.2	6.0	2.2	2.2
	Second endpoint	Detected coordinates	(332,218)	(336,272)	(383,294)	(299,359)	(316,200)	(382,285)
		Deviation from GT	109.6	78.0	8.6	7.2	7.2	5.8
With disturbance elimination	First endpoint	Detected coordinates	(48,451)	(79,502)	(313,205)	(235,265)	(234,265)	(301,354)
		Deviation from GT	2.2	1.0	2.2	1.4	1.4	2.2
	Second endpoint	Detected coordinates	(245,281)	(289,337)	(378,290)	(301,351)	(312,205)	(378,291)
		Deviation from GT	2.2	2.2	3.0	1.0	2.2	1.4

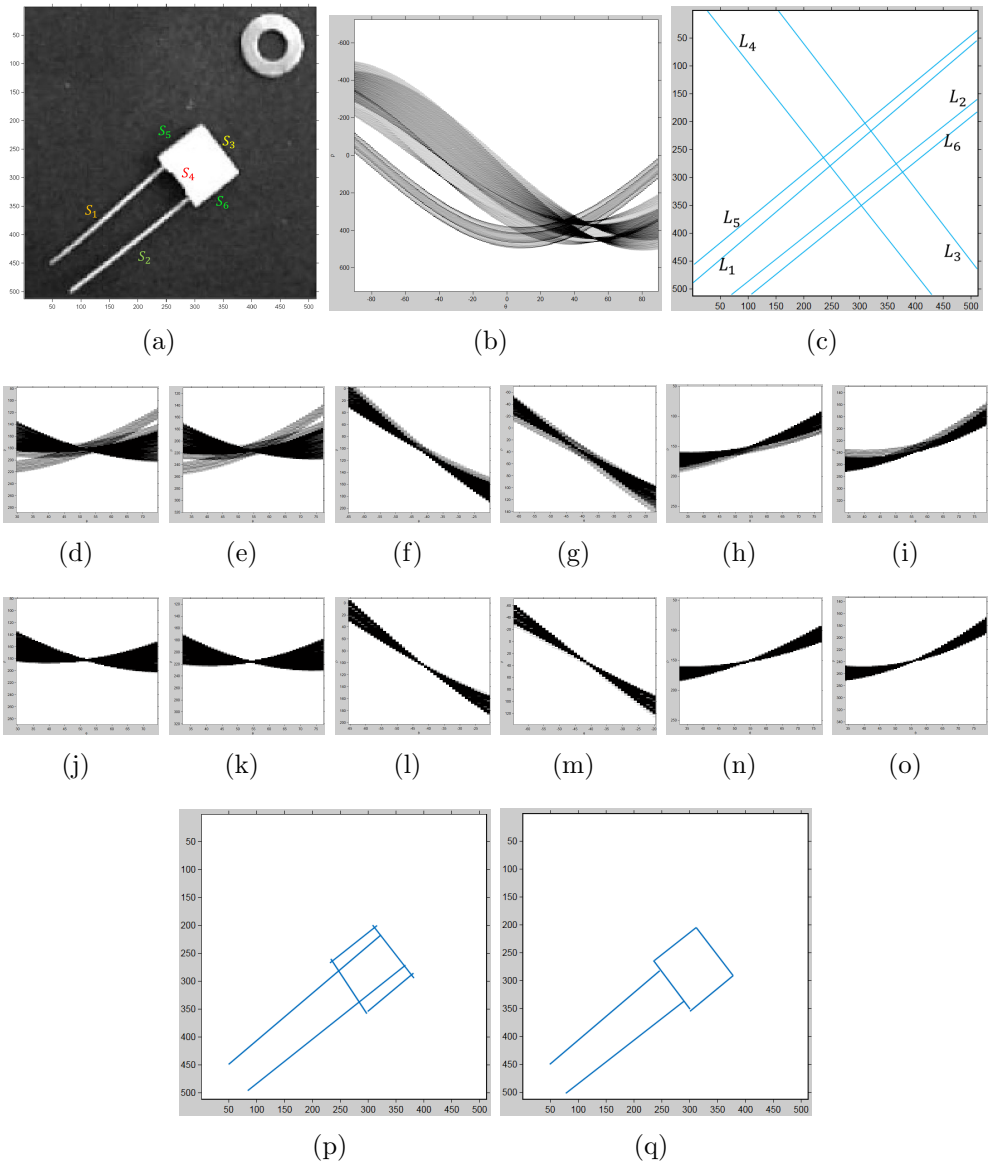


Figure 5.2: Evaluation on disturbance elimination.(a) Original greyscale image. (b) Overlapping butterflies. (c) Result of MLFRFT-based HT. (d)-(i) Result of PWA-HT without disturbance elimination. (j)-(o) Result of the PWA-HT with disturbance elimination. (p) Segmentation result before disturbance elimination. (q) Segmentation result after disturbance elimination.

5.3 Robustness in presence of noise

In this experiment, the robustness of the proposed method to different levels of Gaussian white noise has been evaluated. An edge image reported in [77, 78] with one step-change in the middle is used, and a comparison has been made with three state-of-the-art line segmentation techniques, namely, the LSD [77], EDLines [78], and Du et al.'s HT neighbourhood approach [67]. To generate the results produced by the LSD, a freely available package in IPOL provided by the authors of the LSD has been used [118]. A similar freely available package has been used for the EDLines algorithm obtained from [119]. Unfortunately, the original implementation of the neighbourhood approach were not freely available and could not be provided via the authors in [67, 66] either. Therefore, the algorithm was implemented by the author of this thesis using MATLAB. Implementation was based on the HT neighbourhood definition explained in [66] and the segmentation method explained in [67].

Since the LSD and EDLines are parameter-less algorithms, default settings were used without any changes. For the neighbourhood approach, the following parameter tuning has been chosen for this and the rest of the experiments presented in this chapter; neighbourhood radius r is set to 6 (as suggested in [66]), minimum length of the segment $l = 20$ pixels, and collinear segments distance $d = 30$ to compute the collinear segments resolution δ_θ based on the formula in [67]. As for the proposed segmentation algorithm, a four-layer MLFRFT with $\gamma_1 = 0.5$, $\gamma_2 = 0.7$, $\gamma_3 = 0.9$, and $\gamma_4 = 1$ was chosen; and for the window, $t = 2$ as the maximum line thickness and $\Delta_\theta = 35^\circ$ from θ_0.

Figure 5.3 illustrates the original image of size (512×512) pixels mixed with Gaussian white noise of *mean* = -0.2 and *variance* = 0.4, in the first row, and *mean* = -0.9 and *variance* = 1.1, in the second row. In the first row with a lower-density of noise, the LSD produced a number of disconnected segments instead of one single continuous segment. This is indeed expected;

the presence of noise on the vertical strip prevents the line-support regions of the LSD to iteratively grow to a continuous segment. Therefore, only a number of short segments can be generated. By increasing the noise density, as shown in the second row, the LSD fails to detect even a single portion of the segment; however, no false positive detection occurred.

In low-density noise, the EDLines produced a better result than the LSD but still suffers from disjointed segments. That is due to the use of a default 5×5 Gaussian smoothing kernel in its first phase that generates misleading gradient information for the pixels lying on the vertical strip. In high-density noise, the EDLines detects some portions of the segment in a form of a number of short segments; some with a slight deviation from the actual angle. This is again because of the very high rate of pixel intensity changes along the strip that cause wrong gradient information that adversely effect the edge drawing procedure in EDLines. However, both the LSD and EDLines have suggested the image resolution is cut in half in order to cope with the noise sensitivity problem [77, 78].

The neighbourhood approach detects the segment without breaking it down into short lines, in the first row. The global nature of the SHT, embedded in this method ignores the effect of noisy pixels in the local view. However, in high-density noise, it produces disconnected and disoriented segments with a few false positives. That is, in fact, due to the edge detection prior to applying the SHT. When the noise-level is high, some of the true feature points are ignored as noisy pixels and also some false points can be seen as true feature points. Therefore, the result cannot be reliable for the SHT to detect the correct peak.

The proposed method based on PWA-HT generates an accurate result in low-density noise by detecting the entire segment as a single vertical strip without producing any false positives. This is mainly due to being independent of edge detection by using high resolution frequency sampling of the MLFRFT-based HT to accurately detect the location and the orientation of

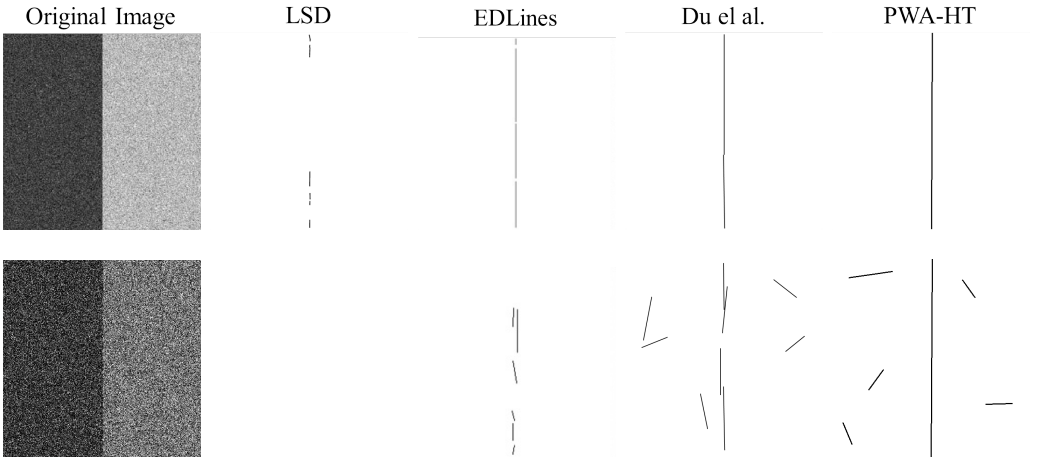
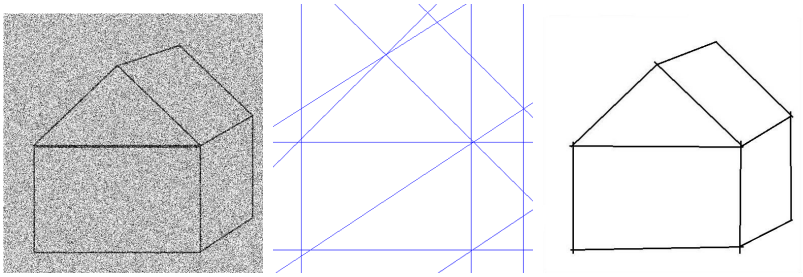


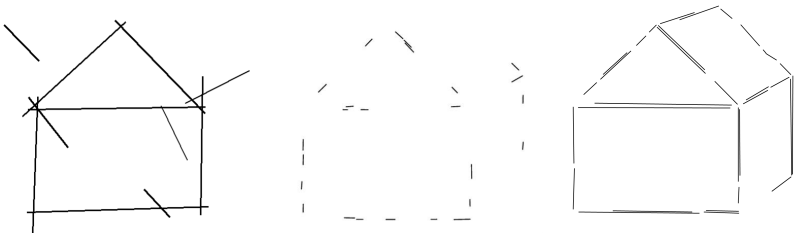
Figure 5.3: Verification on noisy image. From left, original noisy image of a vertical strip of size (512 × 512) with two levels of Gaussian white noise, results produced by the LSD, the EDLines, Du et al.'s HT neighbourhood approach, and the proposed PWA-HT, respectively.

the segment in the first place. The result in the second row also shows the robustness of the algorithm to high-density noise. The vertical strip is fully captured without losing the connectivity of the pixels. However, a few false positive segments are found in the background. To alleviate the problem line validation used in [77, 78] can be adapted. This can be considered in the future work of this research. Overall, the proposed method demonstrates a good precision, in terms of identifying the true segment accurately, in less noisy data (see the top-right of Figure 5.3), whilst preserving the immutability of precision in highly cluttered image by detecting the true segment.

A similar comparison has been conducted on the House image as reported in [2] to examine the robustness the PWA-HT to noise. Figure 5.4(a) shows the image of a house consisting of 11 straight lines occluded with white Gaussian white noise of *mean* = -0.1 and *variance* = 0.3. Figure 5.4(b-f) shows the result of line detection using the MLFRFT-based HT, line segmentation using the PWA-HT, SHT, LSD, and EDLines, respectively. To obtain the re-



(a) Original (b) MLFRFT-HT (c) PWA-HT



(d) SHT (e) LSD (f) EDLines

Figure 5.4: Robustness in presence of noise. (a) Image of House. (b) Result of MLFRFT-based HT. (c)-(f) Line segmentation using the PWA-HT, SHT, LSD, and EDLines, respectively.

sult of the SHT, MATLAB's built-in functions `hough`, `houghpeaks` with the maximum number of peaks set to 11 and `houghlines` with minimum segment length of 50 has been used. These parameters were determined based on trial-and-error to obtain the perceptually best consequence.

From Figure 5.4, it can be seen that the PWA-HT is more robust to noise than the other representative methods. Since the global information is considered and accurate frequency sampling is applied, the PWA-HT is capable of detecting noisy segments. When true line segments are corrupted by noise, local information, such as gradient orientation, cannot be reliable. That is why the LSD and the EDLines fail to detect the true endpoints of the long segments. In addition, their thresholds are not adaptive but predetermined. Although the SHT detects seven segments with only a small

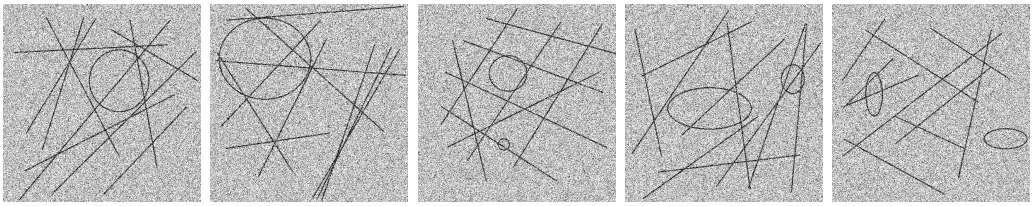


Figure 5.5: A set of five selected samples from dataset of 175 image. $J = 10$ and $d = 0.50$

difference (few pixels) from the true location or coordinates of the endpoints, the remaining five segments are considered as false positives.

To conduct a quantitative study on noise performance, a dataset of 175 synthetic images each containing J lines mixed with one or two curvilinear structures where J = 8, 10, 13, 15, 20, 25, and 30 has been built. For each value of J, 5 distinct manners of object occurrence have been randomly chosen in terms of orientation and position. Then different levels of noise density d have been added to evaluate the detection accuracy. d varies from 0 to 0.5 for the *salt and paper* noise and affect approximately $d \times$ *the number of pixels*. Figure 5.5 shows a subset of five selected images from the dataset with $J = 10$ and noise density $d = 0.50$.

A comparison with the SHT, LSD, and PWA-HT is indicated in Figure 5.6. The results exhibit a moderate performance for the SHT. This fairly consistent behaviour to variant noise is due to the HT's voting process and the minimum threshold value to distinguish the potential straight lines from other image components. Nevertheless, edge detection prior to the SHT causes a major inaccuracy in the final result by neglecting the true feature points. LSD shows an absolute precision in less noisy data; however, it is unable to preserve the accuracy in higher-density noise levels. These noises affect the region growing process of the LSD. The proposed segment detector based on the PWA-HT demonstrates an adequate precision in less noisy data while it preserves the precision immutability in even highly cluttered

images. The reason can be found in the use of 1) The MLFRFT-based HT to accurately detect the location and orientation of possible line segments in the first place; 2) the parallel windowing approach to precisely remove the effect of disturbing feature points in the image plane; and 3) The segment detection algorithm based on the butterfly boundary analysis.

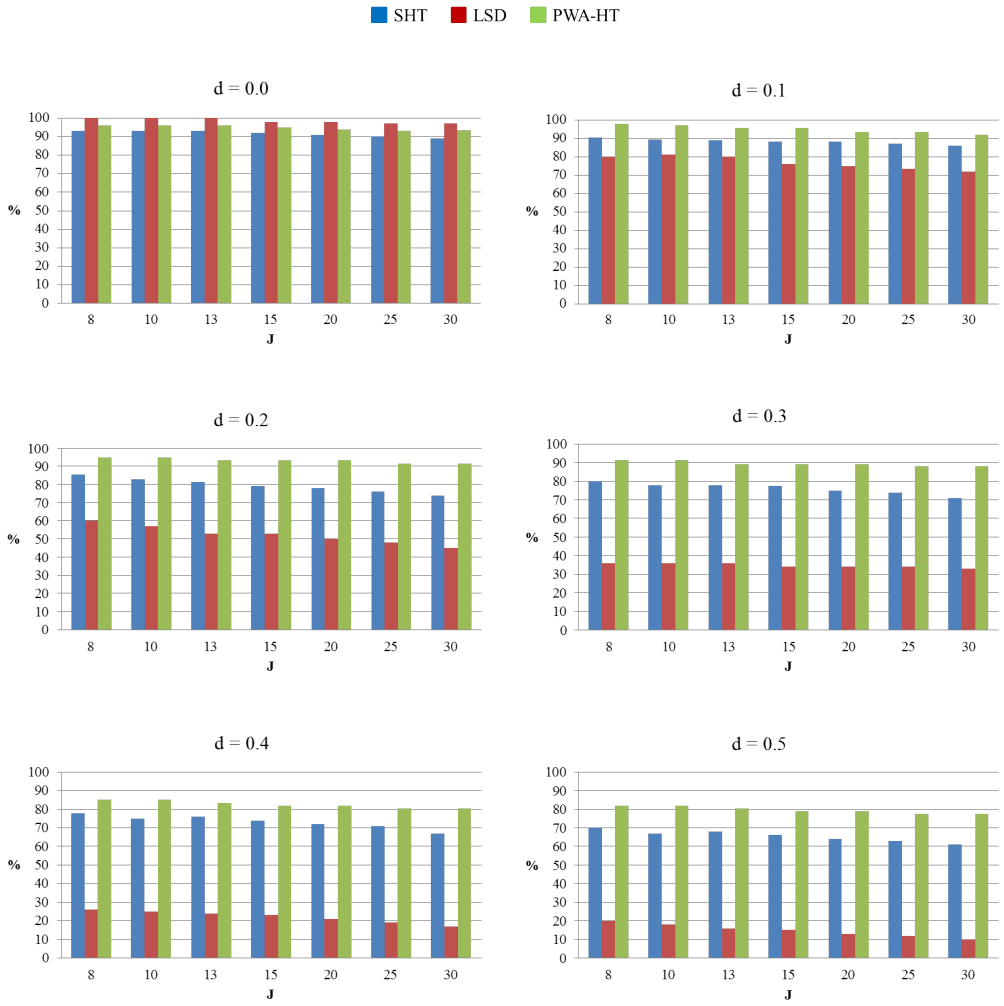


Figure 5.6: Comparison between the average rate of line segment detection in presence of noise using SHT, LSD, and PWA-HT for a dataset of 175 synthetic images. Results of the SHT, LSD, and PWA-HT are coloured in blue, red, and green, respectively. d represents variant noise densities for the salt and pepper noise.

5.4 Evaluation on natural images

It this experiment, performance of the PWA-HT on a number of real-world test images, reported in the literature, has been evaluated. Results are depicted in Figure 5.7 and Figure 5.8. The first column shows the original images of a Truck, a Zebra, and an Office reported in [77, 78], a House [65], two arrows [67], geometrical objects [70], and microscopic elements [120]. Segmentation results for the LSD, EDLines, Du et al.'s HT neighbourhood approach, and the proposed PWA-HT are shown in the second, third, fourth, and fifth columns, respectively. Details of this experiment such as the number of detected segments by each method and processing time are summarised in Table 5.1.

Figure 5.7 shows more complex texture images with significantly more segments to be detected, whereas Figure 5.8 shows images with simple backgrounds and one main feature.

Each of the selected images has unique characteristics to test the performance of the representative methods. The Truck image in Figure 5.7 is a test image with a chessboard pattern and geometrical structures such as squares, rectangles, and other polygons. The Zebra has non-geometrical structures with curved lines, and most of the detected lines correspond to straight-edges but not real straight-lines. Similar to the Truck, the Office image also has geometrical structure, but with bold lines and variable illumination levels. The House image is a classical test image with a strong outline and sharp edges that has an appropriate mixture of short and long segments.

The Arrow1 in Figure 5.8 is an image with a simple background and clear and complete edges. The Arrow2 shows the case where the edges of the arrow on a public road are fragmented because of physical degradation. The Objects image is an appropriate example to distinguish between the true straight-lines and curvilinear structures in an image. It is expected that a straight-line segmentation method will ignore any non-linear features in an image, unlike the results of the LSD and EDLines shown in this example.

And finally, the last image shows an electronic microscopic image of viruses magnified by 150,000 times. This image particularly indicates that, despite having a simple and unified background, any noise-like patterns and roughness in the surface can degrade the detection results in local approaches.

The results show, in most of the cases, that the LSD generates more segments than the other three methods. However, many of the identified short segments are actually part of a single continuous segment that should not be broken down into several segments. This effect is more visible in the House, Arrow1, Objects, and Microscopic images. EDLines has a behaviour very similar to the LSD, but with a considerably lower computational burden.

In contrast, the HT neighbourhood approach fails to detect many of the true short segments (see the results for the Truck, Zebra, and Office). That is because of the trade-off in selecting the neighbourhood radius r. A small radius will not guarantee complete coverage of the neighbourhood, while a large radius fails to produce the short segments. Segments with shorter lengths have bigger approximation errors in comparison to those with longer lengths [66]. The HT neighbourhood approach also has the highest computational complexity among the rest of the tested methods.

However, the PWA-HT showed a superior performance in terms of both the number of detected segments and computational time, when compared to Du et al.'s HT neighbourhood approach. Results show more short segments can be detected in a lower processing time. That is due to the high resolution frequency sampling of the MLFRFT used to generate high resolution peaks. Also, note that the accuracy of the detected endpoints in the PWA-HT are also higher than Du et al.'s approach (see the results for the House, Arrow1, Object, and Microscopic images).

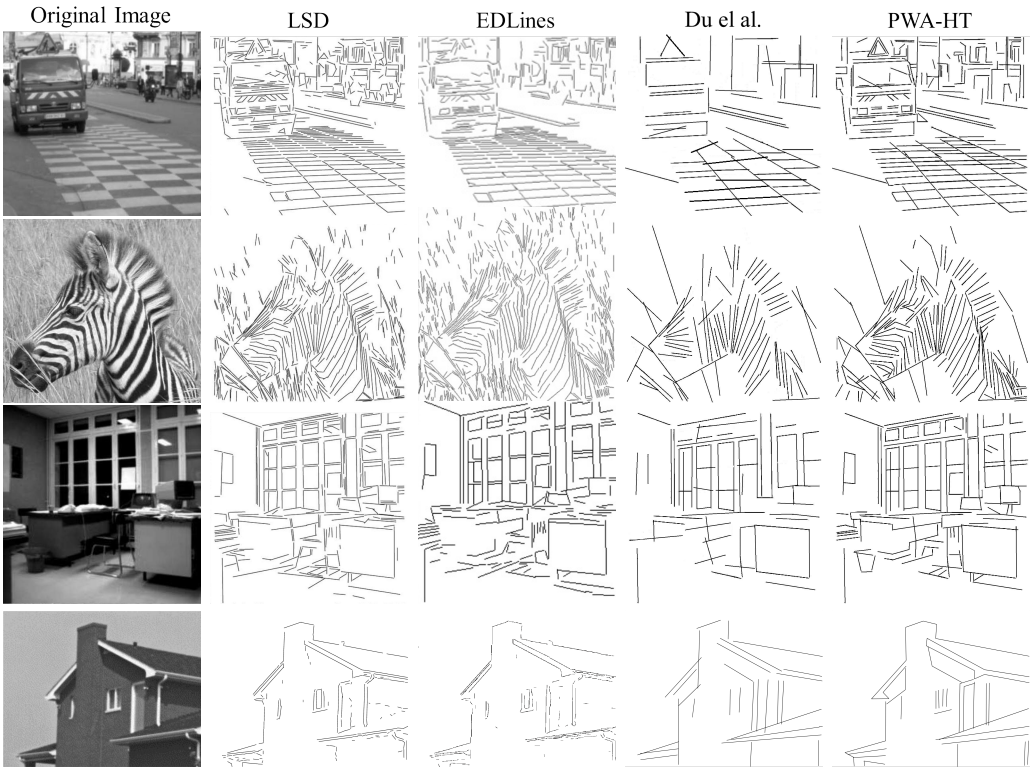


Figure 5.7: Line segmentation on natural images I. From left column, original images, results produced by the LSD, EDLines, Du et al.'s HT neighbourhood approach, and the proposed PWA-HT, respectively.

Table 5.4: Time cost and number of segments.

Image informaiton		LSD		EDLines		Du et al.		PWA-HT	
Description	size	No. of Seg.	Time (sec)	No. of Seg.	Time (sec)	No. of Seg.	Time (sec)	No. of Seg.	Time (sec)
Truck	911×850	799	0.42	587	0.03	70	1.71	227	1.53
Zebra	912×851	1704	0.66	1579	0.06	82	2.34	261	1.88
Office	909×914	613	0.46	616	0.05	72	1.84	155	1.36
House	888×701	231	0.37	190	0.03	29	1.56	47	1.31
Arrow1	512×512	66	0.18	48	0.03	18	1.29	18	0.98
Arrow2	512×512	176	0.16	152	0.02	14	1.05	23	0.72
Objects	512×512	88	0.16	50	0.02	6	0.87	10	0.64
Microscope	512×512	157	0.14	163	0.02	31	1.22	42	0.81

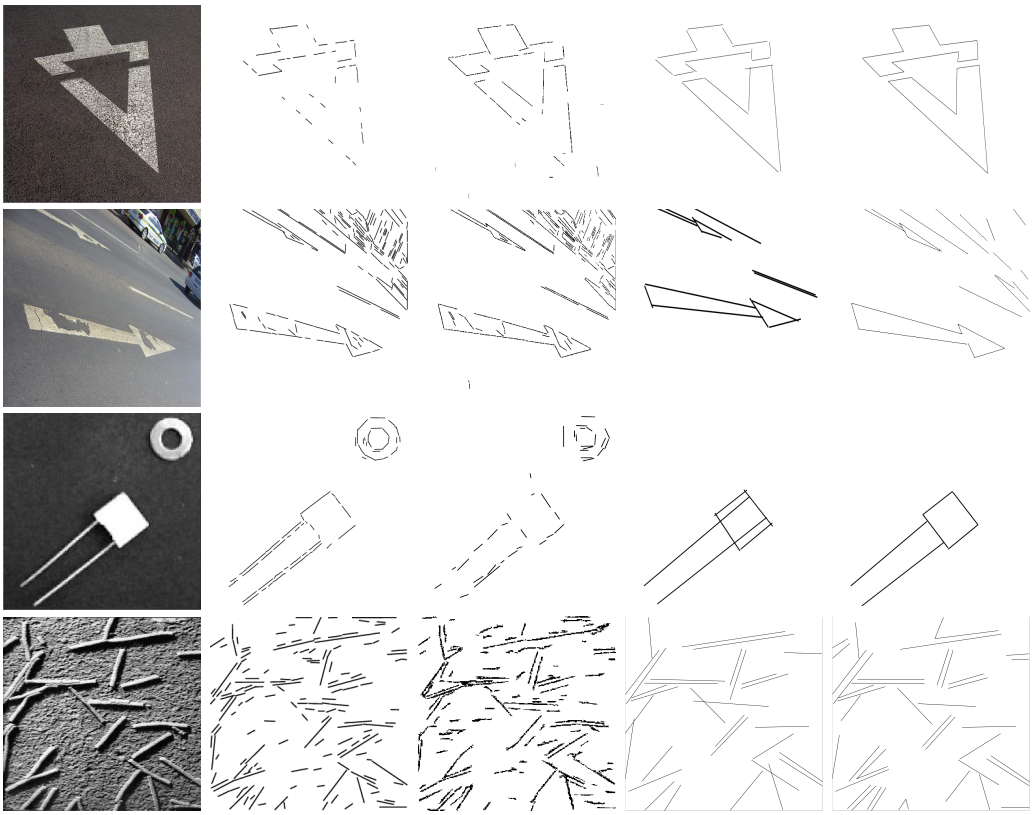


Figure 5.8: Line segmentation on natural images II. From left column, original images, results produced by the LSD, EDLines, Du et al.'s HT neighbourhood approach, and the proposed PWA-HT, respectively.

5.5 Discussion and analysis of findings

In this chapter, performance of the proposed parallel windowing approach to the Hough transform using a multilayer fractional Fourier transform (PWA-HT) was tested in terms of endpoint detection accuracy, robustness to noise and disturbance, and detecting short segments in real-world images. The results of these experiments show some notable advantages as well as disadvantages of the PWA-HT over the representative works that can be interpreted in the following.

The main advantage of the proposed the PWA-HT over the representative approaches is robustness to noise and disturbance in the image. PWA-HT inherits its robustness to noise from the advanced MLFRFT-based HT. Use of the multi-layer Fourier transform also helps reduce the computational burden of the SHT suggested in Du et al.'s approach, due to being independent of prior edge detection and use of the fast Fourier transform algorithm. Furthermore, the disturbance elimination algorithm removes the effect of other non-collinear segments by masking the feature points inside each window.

However, experiments on the step image (Figure 5.3) and the dataset of 175 images (Figure 5.6) show that the PWA-HT is incapable of maintaining accuracy when we face a high rate of occlusion in the pixels. This yields to false positive detection as observed in Figure 5.3. Results in Figure 5.6 demonstrates that when noise density is more than 0.4, i.e. more than 40 per cent of the pixels are occluded with salt and pepper noise, the PWA-HT fails to maintain its robustness. Less than 80 per cent of the segments could have been captured for the images of 30 lines with $d = 0.5$. Nevertheless, the PWA-HT still outperforms the representative approaches such as the LSD and EDLines, due to its global nature.

One of the main advantages of the local approaches, such as the LSD and EDLines, over the PWA-HT and Du et al.'s neighbourhood approach is the line segment validation procedure that is embedded inside their algorithm [77, 78]. In the LSD and EDLines line segment validation is applied after

identifying the potential candidate lines to avoid false detections. In fact, this is the reason why the PWA-HT and the neighbourhood approach produces a few false positives in Figure 5.3. To alleviate the problem a line validation procedure has been suggested in the future work of this research.

Results on natural images revealed that despite the PWA-HT's superior performance over the HT neighbourhood approach it still fails to detect many short segment in comparison to the LSD and EDLines. As discussed in Section 2.3.2 this is, in fact, a very classical drawback of the HT-based methods and is due to their natural global behaviour. A local approach alone is sensitive to noise and produces disjointed segments. On the other hand, a global approach fails to identify the short segments. The optimum solution could be to take advantage of both approaches by combining them. This idea has been further discussed in the future work section.

Chapter 6

Conclusion and future work

6.1 Summary of the work done

In this research, a novel line segment detection method based on the information in both image space and transformed space was proposed.

In the first phase, an advanced Fourier-based Hough transform was used to identify every potential straight-line. That was done by utilising a 2D multilayer fractional Fourier transform (MLFRFT), Cartesian to polar mapping of the frequency samples, and applying the central slice theorem. The parameter space and the peak structure were further enhanced by a 1D-DoG filter. Having a high resolution frequency grid yielded a high resolution parameter space with distinguishable peaks and butterfly boundary sinusoids. The feasibility of parallel implementation of the MLFRFT was investigated and the results revealed an interesting fact. For a typical four-layer MLFRFT, parallel computation of fractional layers increased the computational time instead of helping to reduce it. That is due to the fact that the communication overhead between the client and the parallel workers is higher than the complexity of the code inside the **for** loop (Section. 3.4.1). The execution time of the MLFRFT-based HT was compared with the SHT, which was nearly three times faster in line detection when using the MLFRFT-based

HT. A real-world application of line detection using the MLFRFT was suggested in lane marker detection that has also shown a faster performance in comparison with a traditional SHT-based method.

In the second phase, the identified straight-lines were further analysed for segmentation. A line segmentation method using the HT butterfly boundary (LSBB) was suggested in Section. 4.1. Each straight-line in the image space corresponds to a butterfly in Hough space and sinusoids lying on the boundary of the butterfly wings represent the endpoints of the segments. To approximate the boundary sinusoids, a Robust Least Squares curve fitting was used by obtaining multiple points along each boundary sinusoid. Afterwards, intersections of two boundary points and the peak were found in the image space. These points form a triangular shape and the centroid of the triangle was approximated as the location of the target endpoint. To apply this concept to images with multiple line segments where butterfly wings overlap with each other, a parallel image-domain windowing approach (PWA-HT) was adapted. For each straight-line, identified in the first phase, a window is created with a predefined width that surrounds the target segment and neglects the other pixels outside the window. To further remove the effect of non-collinear feature points inside the window a disturbance elimination algorithm was proposed. Segments that intersect with the target segment or its extension were identified and filtered out from the window. Afterwards a sub-HT was applied to map only the feature points inside the window to achieve a single transparent butterfly.

Experimental results revealed a superior robustness to noise in comparison with the other representative line segmentation techniques. The results also showed the PWA-HT is faster than the state-of-the-art HT-based technique.

6.2 Contribution of the thesis

In this work a novel domain-crossing approach to line segmentation was proposed. The main contribution of this research can be summarised as follows:

1. Extending the MLFRFT-based HT from line detection to line segmentation.
2. Investigation on parallel implementation of the MLFRFT-based HT.
3. Computational time comparison between the SHT and MLFRFT-based HT.
4. Real-world application of the MLFRFT-based HT in lane marker detection.
5. Utilising Least Squares curve fitting to approximate the HT butterfly boundary sinusoids.
6. Intersection of three lines and centroid of the resulting triangle to localise segment endpoints.
7. Windowing approach to isolate the HT butterflies to each appear as a single transparent butterfly.
8. Disturbance elimination algorithm to further enhance the resulting butterfly.

6.3 Limitations

The line segment detection method proposed in this thesis has shown a better robustness to noise than the LSD and EDLines methods and performed better, in terms of the detection of short lines, than Du et al.'s neighbourhood approach. However, it has to be underlined that the presented method

is novel but some issues remain unresolved. Two major limitations of the PWA-HT are as follows:

1. Highlighted by the experimental results, the proposed method fails to avoid false detection caused by high density noise.

2. Unlike the LSD and EDLines it cannot be considered as a high speed real-time algorithm but an accurate and robust line segmentation tool worthy of serious consideration.

6.4 Potential applications

According to the experiments and the performance comparison conducted in this research a number of potential application areas can be identified for the proposed algorithm. The PWA-HT would be a preferred method over local approaches for segment detection in noisy and occluded environments because of its robustness and reliability. It is also suitable for use in inexpensive systems, e.g. with low resolution cameras, and limited computational sophistication.

Depending on the processing time required by the task, the PWA-HT can be used in slow real-time applications. For example, when a processing rate of one frame per second or less is acceptable, the PWA-HT can be a suitable candidate. Slow-movement robots used in rescue missions and still medical images such as in magnetic resonance imaging (MRI) or positron emission tomography (PET) are among the potential applications of the proposed method.

6.5 Suggestions for future work

The future work of this research can be categorised as the following:

1. **Parallel Least Squares.** In the current algorithm a Robust Least Squares curve fitting approach is used due to its insensitivity to outliers and optimised implementation on MATLAB's *Curve Fitting Toolbox*. Forty points were used to approximate the curve (twenty points on each side of the peak). For the future work of this research, more recent developments on such curve fitting approaches can be considered including Parallel Least Squares curve fitting [121]. Using parallel methods more boundary points can be taken into consideration without increasing the computational cost. This will increase the accuracy of the fitted curve.

2. **Line validation.** As shown in the experiments, a high degree of noise can cause false positive segments to be generated. One of the major extensions to this work could be to avoid false segments being detected. A number of works have addressed this problem using line segment validation and verification techniques, such as in [77, 78, 5]. Line segment validation can be performed after identifying the potential segments via the PWA-HT.

3. **Combining local and global approaches.** Although more short lines can be detected with the PWA-HT than the HT neighbourhood approach, there is still a considerable difference between the proposed method and the representative local approaches in terms of the number of detected short segments. Local methods can easily detect short segments without false detection. A new direction to this research can be to make use of local image scanning suggested in [77] and incorporating the region growing technique to seek for the short segments outside the windows. This can be done after the PWA-HT, to detect every other short segment that could not be identified during the process. Furthermore, in the disturbance elimination stage, the region growing can be applied inside the window to detect and filter out any

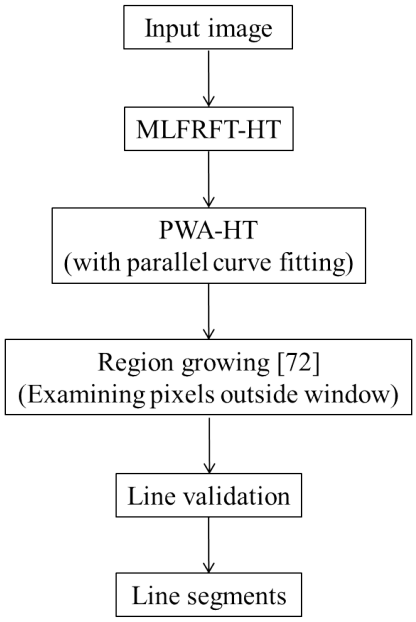


Figure 6.1: Block diagram of the future work and extensions to this research.

disturbances caused by the short segments that are not distinguishable by the MLFRFT-based HT. Note that the local approaches are relatively fast and considered as real-time algorithms. Therefore, the extra computational cost would be bearable.

4. **Implementation in a lower-level programming language.** Last but not least, computational time might be further improved by translating the source code of the proposed algorithm from MATLAB, which is a high-level programming language, to a low or intermediate-level programming language such as C or C++ . Note that, the LSD and EDLines are based on C (low-level) and C++ (intermediate-level) languages respectively. The C++ implementation of the current algorithm is on progress at present and experiments on the MLFRFT have already shown a promising result.

Figure 6.1 illustrates a block diagram of the suggested future work and extensions to this research.

6.6 Epilogue

With the fast progress of research and applications in image understanding and computer vision, the need for accurate line detection and segmentation is increasing. The work presented in this thesis has been an attempt to contribute to one of the primitive and yet important steps in many object recognition and feature detection tasks, i.e. line detection and segmentation. On one hand, the proposed algorithm illustrated promising outcomes with good accuracy. On the other hand, it highlighted the gaps and limitation that are worthy of future investigation. It also provided grounds for improvement and expansion of this work.

Bibliography

[1] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Graphics and Image Processing*, vol. 15, pp. 11–15, January 1972.

[2] D. Shi, L. Zheng, and J. Liu, "Advanced Hough transform using a multilayer fractional Fourier method," *IEEE Transactions on Image Processing*, vol. 19, pp. 1558–1566, June 2010.

[3] C. F. Olson, "A general method for geometric feature matching and model extraction," *International Journal of Computer Vision*, vol. 45, pp. 39–54, 2001.

[4] C.-T. Ho and L.-H. Chen, "A high-speed algorithm for elliptical object detection," *IEEE Transactions on Image Processing*, vol. 5, pp. 547–550, March 1996.

[5] J. Song and M. R. Lyu, "A Hough transform based line recognition method utilizing both parameter space and image space," *Pattern Recognition*, vol. 38, no. 4, pp. 539–552, 2005.

[6] V. Ayala-Ramirez, C. H. Garcia-Capulin, A. Perez-Garcia, and R. E. Sanchez-Yanez, "Circle detection on images using genetic algorithms," *Pattern Recognition Letters*, vol. 27, no. 6, pp. 652–657, 2006.

[7] Y. Liu, T. Ikenaga, and S. Goto, "An MRF model-based approach to the detection of rectangular shape objects in color images," *Signal Processing*, vol. 87, no. 11, pp. 2649–2658, 2007.

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

[9] Z. Li, Y. Liu, R. Hayward, J. Zhang, and J. Cai, "Knowledge-based power line detection for UAV surveillance and inspection systems," in *23rd International ConferenceImage and Vision Computing New Zealand, IVCNZ 2008*, pp. 1–6, 2008.

[10] J. Zhang, L. Liu, B. Wang, X. Chen, Q. Wang, and T. Zheng, "High speed automatic power line detection and tracking for a UAV-based inspection," in *2012 International Conference on Industrial Control and Electronics Engineering (ICICEE)*, pp. 266–269, 2012.

[11] S. Du and C. Tu, "Power line inspection using segment measurement based on HT butterfly," in *2011 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, pp. 1–4, 2011.

[12] P. Frnti, E. I. Ageenko, H. Klviinen, and S. Kukkonen, "Compression of line drawing images using Hough transform for exploiting global dependencies," in *International Conference on Information Sciences*, 1998.

[13] A. Borkar, M. Hayes, and M. Smith, "A novel lane detection system with efficient ground truth generation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, pp. 365–374, March 2012.

[14] J. McCall and M. Trivedi, "Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation," *IEEE Transac-*

tions on Intelligent Transportation Systems, vol. 7, pp. 20–37, March 2006.

[15] A. Bar Hillel, R. Lerner, D. Levi, and G. Raz, "Recent progress in road and lane detection: a survey," *Machine Vision and Applications*, pp. 1–19, 2012.

[16] G. Zhang, N. Zheng, C. Cui, Y. Yan, and Z. Yuan, "An efficient road detection method in noisy urban environment," in *IEEE Intelligent Vehicles Symposium*, pp. 556–561, June 2009.

[17] J.-S. Hong, T. Dohi, M. Hasizume, K. Konishi, and N. Hata, "A motion adaptable needle placement instrument based on tumor specific ultrasonic image segmentation," in *Medical Image Computing and Computer-Assisted Intervention*, vol. 2488 of *Lecture Notes in Computer Science*, pp. 122–129, Springer Berlin/Heidelberg, 2002.

[18] P. Ballester, "Hough transform and astronomical data analysis," *Vistas in Astronomy*, vol. 40, no. 4, pp. 479–485, 1996.

[19] Z. Li, Y. Liu, R. Walker, R. Hayward, and J. Zhang, "Towards automatic power line detection for a UAV surveillance system using pulse coupled neural filter and an improved Hough transform," *Machine Vision and Applications*, vol. 21, pp. 677–686, 2010.

[20] A. Moqiseh and M. Nayebi, "Combinational Hough transform for surveillance radar target detection in a 3-D data map," in *IEEE Radar Conference, 2008. RADAR '08*, pp. 1–6, May 2008.

[21] P. Hough, "Method and means for recognizing complex patterns." U.S. Patent 3.069.654, December 1962.

[22] H. Aghajan and T. Kailath, "SLIDE: subspace-based line detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 11, pp. 1057–1073, 1994.

[23] J.-C. Chien and C.-C. Li, "Wavelet-based line detection in gray-scale images," in *IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, vol. 4, pp. 3670–3673 vol.4, 1997.

[24] B. Hou, F. Liu, and L. Jiao, "Linear feature detection based on ridgelet," *Science in China Series E: Technological Sciences*, vol. 46, no. 2, pp. 141–152, 2003.

[25] G. quan Lu, X. Hong-guo, and Y. bing Li, "Line detection based on chain code detection," in *IEEE International Conference on Vehicular Electronics and Safety*, pp. 98–103, 2005.

[26] Y. Zheng, H. Li, and D. Doermann, "A parallel-line detection algorithm based on HMM decoding," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 5, pp. 777–792, 2005.

[27] S. Berlemont and J.-C. Olivo-Marin, "Combining local filtering and multiscale analysis for edge, ridge, and curvilinear objects detection," *IEEE Transactions on Image Processing*, vol. 19, pp. 74–84, January 2010.

[28] A. Bonci, T. Leo, and S. Longhi, "A Bayesian approach to the Hough transform for line detection," *IEEE Transactions on Systems, Man and Cybernetics, Part A*, vol. 35, no. 6, pp. 945–955, 2005.

[29] J. Ma and L. Li, "Automatic straight line detection through fixed-point BYY harmony learning," in *Advanced Intelligent Computing Theories and Applications. With Aspects of Theoretical and Methodological Issues*, vol. 5226 of *Lecture Notes in Computer Science*, pp. 569–576, Springer Berlin Heidelberg, 2008.

[30] Z.-Y. Liu, K.-C. Chiu, and L. Xu, "Strip line detection and thinning by RPCL-based local PCA," *Pattern Recognition Letters*, vol. 24, no. 14, pp. 2335 – 2344, 2003.

[31] N. Aggarwal and W. Karl, "Line detection in images through regularized Hough transform," *IEEE Transactions on Image Processing*, vol. 15, pp. 582–591, March 2006.

[32] J. Ji, G. Chen, and L. Sun, "A novel Hough transform method for line detection by enhancing accumulator array," *Pattern Recognition Letters*, vol. 32, no. 11, pp. 1503 – 1510, 2011.

[33] D. Walsh and A. E. Raftery, "Accurate and efficient curve detection in images: the importance sampling Hough transform," *Pattern Recognition*, vol. 35, no. 7, pp. 1421 – 1431, 2002.

[34] L. Hai-Bin and Y. Wei-Dong, "An effective algorithm to detect triangles in image," *Journal of Image and Graphics*, vol. 13, no. 3, pp. 456–460, 2008.

[35] T. van Veen and F. Groen, "Discretization errors in the Hough transform," *Pattern Recognition*, vol. 14, no. 1-6, pp. 137 – 145, 1981.

[36] W. Niblack and D. Petkovic, "On improving the accuracy of the Hough transform: theory, simulations, and experiments," in *Computer Society Conference on Computer Vision and Pattern Recognition, 1988. Proceedings CVPR '88*, pp. 574–579, 1988.

[37] J. O'Rourke, "Dynamically quantized spaces for focusing the Hough transform," in *Proceedings of the 7th international joint conference on Artificial intelligence*, IJCAI'81, (San Francisco, CA, USA), pp. 737–739, Morgan Kaufmann Publishers Inc., 1981.

[38] I. Svalbe, "Natural representations for straight lines and the Hough transform on discrete arrays," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 9, pp. 941–950, 1989.

[39] M. Zhang, "On the discretization of parameter domain in Hough transformation," in *Proceedings of the 13th International Conference on Pattern Recognition*, vol. 2, pp. 527–531, 1996.

[40] Q. Ji and R. M. Haralick, "Error propagation for the Hough transform," *Pattern Recognition Letters*, vol. 22, no. 6-7, pp. 813 – 823, 2001.

[41] T. T. Nguyen, X. D. Pham, and J. Jeon, "An improvement of the standard Hough transform to detect line segments," in *IEEE International Conference on Industrial Technology, ICIT 2008.*, pp. 1–6, 2008.

[42] H. Duan, X. Liu, and H. Liu, "A nonuniform quantization of Hough space for the detection of straight line segments," in *2nd International Conference on Pervasive Computing and Applications, ICPCA 2007*, pp. 149–153, 2007.

[43] N. Kiryati and A. Bruckstein, "Antialiasing the Hough transform," *CVGIP: Graphical Models and Image Processing*, vol. 53, no. 3, pp. 213 – 222, 1991.

[44] V. Shapiro, "Accuracy of the straight line Hough transform: The non-voting approach," *Computer Vision and Image Understanding*, vol. 103, no. 1, pp. 1 – 21, 2006.

[45] L. A. Fernandes and M. M. Oliveira, "Real-time line detection through an improved Hough transform voting scheme," *Pattern Recognition*, vol. 41, no. 1, pp. 299 – 314, 2008.

[46] R. Stephens, "Probabilistic approach to the Hough transform," *Image and Vision Computing*, vol. 9, no. 1, pp. 66–71, 1991.

[47] J. Matas, C. Galambos, and J. Kittler, "Progressive probabilistic Hough transform," in *Proceedings of the British Machine Vision Conference*, pp. 26.1–26.10, BMVA Press, 1998.

[48] C. Galambos, J. Matas, and J. Kittler, "Progressive probabilistic Hough transform for line detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. –560 Vol. 1, 1999.

[49] J. Matas, C. Galambos, and J. Kittler, "Robust detection of lines using the progressive probabilistic Hough transform," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 119 – 137, 2000.

[50] 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.

[51] L. Xu and E. Oja, "Randomized Hough transform (RHT): basic mechanisms, algorithms, and computational complexities," *Computer Vision and Image Understanding*, vol. 57, pp. 131–154, March 1993.

[52] J. Princen, J. Illingworth, and J. Kittler, "A hierarchical approach to line extraction based on the Hough transform," *Computer Vision, Graphics, and Image Processing*, vol. 52, no. 1, pp. 57 – 77, 1990.

[53] R. A. Fisher, "The maximum likelihood method," *Messenger Math*, vol. 41, pp. 155 – 160, 1912.

[54] R. V. Hogg and A. Craig, *Introduction to Mathematical Statistics*. Prentice Hall, 5th ed., 1994.

[55] N. Kiryati, Y. Eldar, and A. Bruckstein, "A probabilistic Hough transform," *Pattern Recognition*, vol. 24, no. 4, pp. 303 – 316, 1991.

[56] C. Galambos, J. Kittler, and J. Matas, "Using gradient information to enhance the progressive probabilistic Hough transform," in *Proceedings of 15th International Conference on Pattern Recognition*, vol. 3, pp. 560–563 vol.3, 2000.

[57] P. R. Thrift and S. M. Dunn, "Approximating point-set images by line segments using a variation of the Hough transform," *Computer Vision, Graphics, and Image Processing*, vol. 21, no. 3, pp. 383 – 394, 1983.

[58] M. Fiala, "Identify and remove Hough transform method," in *Proceedings of the Vision Interface*, pp. 184–187, 2003.

[59] C.-T. Choy, P.-K. Ser, and W.-C. Siu, "Peak detection in Hough transform via self-organizing learning," in *IEEE International Symposium on Circuits and Systems, ISCAS '95.*, vol. 1, pp. 139–142 vol.1, 1995.

[60] L. Chao, W. Zhong, and L. Lin, "An improved HT algorithm on straight line detection based on Freeman chain code," in *2nd International Congress on Image and Signal Processing*, pp. 1–4, 2009.

[61] M. Atiquzzaman and M. W. Akhtar, "Complete line segment description using the Hough transform," *Image and Vision Computing*, vol. 12, no. 5, pp. 267 – 273, 1994.

[62] M. Atiquzzaman and M. Akhtar, "A robust Hough transform technique for complete line segment description," *Real-Time Imaging*, vol. 1, no. 6, pp. 419–426, 1995.

[63] V. Kamat and S. Ganesan, "A robust Hough transform technique for description of multiple line segments in an image," in *Proceedings of IEEE International Conference on Image Processing*, vol. 1, pp. 216–220, October 1998.

[64] V. Kamat-Sadekar and S. Ganesan, "Complete description of multiple line segments using the Hough transform," *Image and Vision Computing*, vol. 16, no. 910, pp. 597 – 613, 1998.

[65] Y. Furukawa and Y. Shinagawa, "Accurate and robust line segment extraction by analyzing distribution around peaks in Hough space," *Computer Vision and Image Understanding*, vol. 92, no. 1, pp. 1–25, 2003.

[66] S. Du, B. van Wyk, C. Tu, and X. Zhang, "An improved Hough transform neighborhood map for straight line segments," *IEEE Transactions on Image Processing*, vol. 19, pp. 573–585, March 2010.

[67] S. Du, C. Tu, B. J. van Wyk, and Z. Chen, "Collinear segment detection using HT neighborhoods," *IEEE Transactions on Image Processing*, vol. 20, pp. 3612–3620, December 2011.

[68] C. Tu, S. Du, B. van Wyk, K. Djouani, and Y. Hamam, "High resolution Hough transform based on butterfly self-similarity," *Electronics Letters*, vol. 47, no. 25, pp. 1360–1361, 2011.

[69] S. Du, C. Tu, and M. Sun, "High accuracy Hough transform based on butterfly symmetry," *Electronics Letters*, vol. 48, no. 4, pp. 199–201, 2012.

[70] C. G. Ho, R. C. D. Young, C. D. Bradfield, and C. R. Chatwin, "A fast Hough transform for the parametrisation of straight lines using fourier methods," *Real-Time Imaging*, vol. 6, no. 2, pp. 113–127, 2000.

[71] L. Zheng and D. Shi, "Advanced Radon transform using generalized interpolated Fourier method for straight line detection," *Computer Vision and Image Understanding*, vol. 115, pp. 152–160, February 2011.

[72] S. R. Deans, "Hough transform from the Radon transform," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 3, pp. 185–188, March 1981.

[73] J. Radon, "ber die bestimmung von funktionen durch ihre integralwerte lngs gewisser mannigfaltigkeiten," *Berichte der Sachsischen Akadamie der Wissenschaft,*, vol. 69, pp. 262–277, April 1917.

[74] J. Radon, "On the determination of functions from their integral values along certain manifolds," *IEEE Transactions on Medical Imaging*, vol. 5, no. 4, pp. 170–176, 1986.

[75] P. Toft, "The Radon transform - theory and implementation," *PhD thesis, Department of Mathematical Modelling, Technical University of Denmark*, June 1996.

[76] M. van Ginkel, C. L. Hendriks, and L. van Vliet, "A short introduction to the Radon and Hough transforms and how they relate to each other," *in the Quantitative Imaging Group Technical Report Series*, no. QI-2004-01, 2004.

[77] R. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "LSD: A fast line segment detector with a false detection control," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 722–732, April 2010.

[78] C. Akinlar and C. Topal, "EDLines: A real-time line segment detector with a false detection control," *Pattern Recognition Letters*, vol. 32, no. 13, pp. 1633–1642, 2011.

[79] K. Yang, S. Sam Ge, and H. He, "Robust line detection using two-orthogonal direction image scanning," *Computer Visison and Image Understanding*, vol. 115, no. 8, pp. 1207–1222, 2011.

[80] D. Guru, B. Shekar, and P. Nagabhushan, "A simple and robust line detection algorithm based on small eigenvalue analysis," *Pattern Recognition Letters*, vol. 25, no. 1, pp. 1 – 13, 2004.

[81] J. Koeck and W. Zhang, "Video compass," in *Computer Vision - ECCV 2002*, vol. 2353 of *Lecture Notes in Computer Science*, pp. 476–490, Springer Berlin Heidelberg, 2002.

[82] R. Nevatia and K. R. Babu, "Linear feature extraction and description," *Computer Graphics and Image Processing*, vol. 13, no. 3, pp. 257 – 269, 1980.

[83] P. Kahn, L. Kitchen, and E. M. Riseman, "A fast line finder for vision-guided robot navigation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 11, pp. 1098–1102, 1990.

[84] J. B. Burns, A. R. Hanson, and E. M. Riseman, "Extracting straight lines," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, pp. 425–455, July 1986.

[85] A. Desolneux, L. Moisan, and J.-M. Morel, "Meaningful alignments," *International Journal of Computer Vision*, vol. 40, pp. 7–23, 2000.

[86] A. Etemadi, "Robust segmentation of edge data," in *International Conference on Image Processing and its Applications*, pp. 311–314, 1992.

[87] C. Topal, C. Akinlar, and Y. Gen, "Edge Drawing: A heuristic approach to robust real-time edge detection," in *20th International Conference on Pattern Recognition (ICPR)*, pp. 2424–2427, 2010.

[88] C. Topal and C. Akinlar, "Edge Drawing: A combined real-time edge and segment detector," *Journal of Visual Communication and Image Representation*, vol. 23, no. 6, pp. 862 – 872, 2012.

[89] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[90] S. Y. K. Yuen, T. S. L. Lam, and N. K. D. Leung, "Connective Hough transform," *Image and Vision Computing*, vol. 11, no. 5, pp. 295 – 301, 1993.

[91] N. Guil, J. Villalba, and E. Zapata, "A fast Hough transform for segment detection," *IEEE Transactions on Image Processing*, vol. 4, pp. 1541–1548, Nov. 1995.

[92] J. Cha, R. Cofer, and S. Kozaitis, "Extended Hough transform for linear feature detection," *Pattern Recognition*, vol. 39, no. 6, pp. 1034–1043, 2006.

[93] K.-L. Chung, T.-C. Chang, and Y.-H. Huang, "Comment on: Extended Hough transform for linear feature detection," *Pattern Recognition*, vol. 42, no. 7, pp. 1612–1614, 2009.

[94] A. Bandera, J. Prez-Lorenzo, J. Bandera, and F. Sandoval, "Mean shift based clustering of hough domain for fast line segment detection," *Pattern Recognition Letters*, vol. 27, no. 6, pp. 578 – 586, 2006.

[95] M. Nieto, C. Cuevas, L. Salgado, and N. Garca, "Line segment detection using weighted mean shift procedures on a 2D slice sampling strategy," *Pattern Analysis and Applications*, vol. 14, no. 2, pp. 149–163, 2011.

[96] M. Jacob and M. Unser, "Design of steerable filters for feature detection using Canny-like criteria," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, pp. 1007–1019, August 2004.

[97] R. Mersereau and A. Oppenheim, "Digital reconstruction of multidimensional signals from their projections," *Proceedings of the IEEE*, vol. 62, no. 10, pp. 1319–1338, 1974.

[98] R. Easton Jr. and H. Barrett, "Tomographic transformations in optical signal processing," *Optical Signal Processing*, pp. 335 – 386, 1987.

[99] G. Wolberg and S. Zokai, "Robust image registration using log-polar transform," in *Proceedings of IEEE International Conference on Image Processing*, vol. 1, pp. 493–496 vol.1, 2000.

[100] S. Derrode and F. Ghorbel, "Robust and efficient Fourier-Mellin transform approximations for gray-level image reconstruction and complete invariant description," *Computer Vision and Image Understanding*, vol. 83, no. 1, pp. 57 – 78, 2001.

[101] B. Reddy and B. N. Chatterji, "An FFT-based technique for translation, rotation, and scale-invariant image registration," *IEEE Transactions on Image Processing*, vol. 5, no. 8, pp. 1266–1271, 1996.

[102] A. Dutt and V. Rokhlin, "Fast Fourier transforms for nonequispaced data, II," *Applied and Computational Harmonic Analysis*, vol. 2, no. 1, pp. 85 – 100, 1995.

[103] N. Nguyen and Q. H. Liu, "The regular Fourier matrices and nonuniform fast Fourier transforms," *SIAM Journal on Scientific Computing*, vol. 21, pp. 283–293, August 1999.

[104] J. Fessler and B. Sutton, "Nonuniform fast Fourier transforms using min-max interpolation," *IEEE Transactions on Signal Processing*, vol. 51, no. 2, pp. 560–574, 2003.

[105] A. Averbuch, R. Coifman, D. Donoho, M. Elad, and M. Israeli, "Fast and accurate polar Fourier transform," *Applied and Computational Harmonic Analysis*, vol. 21, no. 2, pp. 145–167, 2006.

[106] L. Liang, G. Shi, and X. Xie, "Nonuniform directional filter banks with arbitrary frequency partitioning," *IEEE Transactions on Image Processing*, vol. 20, no. 1, pp. 283–288, 2011.

[107] A. Averbuch, R. Coifman, D. Donoho, M. Israeli, and Y. Shkolnisky, "A framework for discrete integral transformations i - the pseudopolar Fourier transform," *SIAM Journal on Scientific Computing*, vol. 30, no. 2, pp. 764–784, 2008.

[108] Y. Keller, A. Averbuch, and M. Israeli, "Pseudopolar-based estimation of large translations, rotations, and scalings in images," *IEEE Transactions on Image Processing*, vol. 14, no. 1, pp. 12–22, 2005.

[109] W. Pan, K. Qin, and Y. Chen, "An adaptable-multilayer fractional Fourier transform approach for image registration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, pp. 400–414, March 2009.

[110] D. Marr and E. Hildreth, "Theory of edge detection," *Proceedings of the Royal Society of London. Series B. Biological Sciences*, vol. 207, no. 1167, pp. 187–217, 1980.

[111] A. Borkar, M. Hayes, M. Smith, and S. Pankanti, "A layered approach to robust lane detection at night," in *IEEE Workshop on Computational Intelligence in Vehicles and Vehicular Systems. CIVVS '09*, pp. 51–57, April 2009.

[112] Y. Jiang, F. Gao, and G. Xu, "Computer vision-based multiple-lane detection on straight road and in a curve," in *International Conference on Image Analysis and Signal Processing (IASP)*, pp. 114–117, April 2010.

[113] X. Shi, B. Kong, and F. Zheng, "A new lane detection method based on feature pattern," in *2nd International Congress on Image and Signal Processing. CISP '09*, pp. 1–5, October 2009.

[114] A. Borkar, M. Hayes, and M. Smith, "Robust lane detection and tracking with ransac and kalman filter," in *16th IEEE International Conference on Image Processing (ICIP)*, pp. 3261–3264, November 2009.

[115] P. Gravel, G. Beaudoin, and J. De Guise, "A method for modeling noise in medical images," *IEEE Transactions on Medical Imaging*, vol. 23, no. 10, pp. 1221–1232, 2004.

[116] MATLAB and Curve Fitting Toolbox Release 2013a, The MathWorks, Inc., Natick, Massachusetts, United States.

[117] D. C. Hoaglin, F. Mosteller, and J. W. Tukey, *Understanding robust and exploratory data analysis*, vol. 3. Wiley New York, 1983.

[118] LSD: a Line Segment Detector, Website. http://www.ipol.im/pub/art/2012/gjmr-lsd/.

[119] EDLines: Edge Drawing (ED)-Based Real-Time Line Segment Detection, Website. http://ceng.anadolu.edu.tr/CV/EDLines/demo.aspx.

[120] "Electron microscope image website, http://cosmicastronomy.com/3d-learn.htm," 2012.

[121] A. Yang, W. Ai-ling, and J. Chang, "The research on parallel least squares curve fitting algorithm," in *International Conference on Test and Measurement, ICTM 09*, vol. 2, pp. 201–204, 2009.

[122] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Mathematics of computation*, vol. 19, no. 90, pp. 297–301, 1965.

[123] M. Frigo and S. Johnson, "The design and implementation of FFTW3," *Proceedings of the IEEE*, vol. 93, no. 2, pp. 216–231, 2005.

Appendices

Appendix A

Fast Fourier transform

A.1 Forward transform

The fast Fourier transform (FFT) is a computationally optimised algorithm for calculating the discrete Fourier transform (DFT). Let us consider the continuous Fourier transform as a form of integral given by

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-j2\pi ux} \mathrm{d}x \tag{A.1}$$

where $f(x)$ is a continuous function in the time domain. As a result of an imaginary component j the transform yields to a complex domain, refereed as the Fourier or frequency domain. Recalling that the imaginary exponent could be written as:

$$e^{j\theta} = \cos\theta + j\sin\theta \tag{A.2}$$

For a digital signal that has been sampled from its continuous form $f(x)$ into a discrete form $f(n)$, a DFT is defied as:

$$F(k) = \sum_{n=0}^{N-1} f(n)\ e^{-j2\pi nk/N} \tag{A.3}$$

Here $\{f_0, f_1, ..., f_{N-1}\}$ are discrete samples of input signal $f(n)$ and $\{F_0, F_1, ..., F_{N-1}\}$ are the corresponding result of the DFT. N is the total number of available discrete components in $f(n)$ and is usually a power of 2.

Therefore, to implement the algorithm using a computer program it is sufficient to write a double loop code and calculate the sums of the products of input samples and imaginary exponents. The complexity of such operation is of $O(N^2)$ order. However, the complexity can be reduced to the order of $O(N \log_2 N)$ using the algorithm explained in the following.

Let us consider the DFT of a signal with $N = 8$ samples given as:

$$\begin{aligned} F(k) = f(0) + f(1)\ e^{-j2\pi k/8} + f(2)\ e^{-j2\pi 2k/8} + f(3)\ e^{-j2\pi 3k/8} \\ + f(4)\ e^{-j2\pi 4k/8} + f(5)\ e^{-j2\pi 5k/8} + f(6)\ e^{-j2\pi 6k/8} + f(7)\ e^{-j2\pi 7k/8} \end{aligned} \quad \text{(A.4)}$$

Equation. A.4 can be split into two similar sums by separating the odd and even elements and factoring out the $e^{-j2\pi k/8}$ from $f(1)$ component.

$$\begin{aligned} F(k) = \Big[f(0) + f(2)\ e^{-j2\pi 2k/8} + f(4)\ e^{-j2\pi 4k/8} + f(6)\ e^{-j2\pi 6k/8} \Big] \\ + e^{-j2\pi k/8} \Big[f(1) + f(3)\ e^{-j2\pi 2k/8} + f(5)\ e^{-j2\pi 4k/8} + f(7)\ e^{-j2\pi 6k/8} \Big] \end{aligned} \quad \text{(A.5)}$$

We can repeat the simplification by factoring out $e^{-j2\pi 2k/8}$:

$$F(k) = \left[\left(f(0) + f(4)\ e^{-j2\pi 4k/8}\right) + e^{-j2\pi 2k/8}\left(f(2) + f(6)\ e^{-j2\pi 4k/8}\right)\right] + e^{-j2\pi k/8}\left[\left(f(1) + f(5)\ e^{-j2\pi 4k/8}\right) + e^{-j2\pi 2k/8}\left(f(3) + f(7)\ e^{-j2\pi 4k/8}\right)\right] \tag{A.6}$$

Equation. A.6 shows there are $\log_2 8 = 3$ levels of summation, i.e. the deepest level in parenthesis, the middle level in brackets and the outer or the last level. Also, for every level the exponential component is the same. Equation. A.6 can be further simplified as:

$$F(k) = \left[\left(f(0) + f(4)\ e^{-j\pi k}\right) + e^{-j\pi k/2}\left(f(2) + f(6)\ e^{-j\pi k}\right)\right] + e^{-j\pi k/4}\left[\left(f(1) + f(5)\ e^{-j\pi k}\right) + e^{-j\pi k/2}\left(f(3) + f(7)\ e^{-j\pi k}\right)\right] \tag{A.7}$$

Note that one of the interesting properties of a complex number $e^{j\theta}$ is the periodicity at every 2π radian and can be used to speed-up the computation. Given that

$$e^{j\theta} = e^{j(\theta+2\pi)} \tag{A.8}$$

Period for each of the exponential terms inside the parenthesis in Equation. A.7 is $k = 2$. Meaning that summations inside the parenthesis are the same for $k = 0, 2, 4, 6$ and for $k = 1, 3, 5, 7$. That means on the deepest level, inside the parenthesis, $4 \times 2 = 8$, i.e. number of summations times the period, operations are required. Also, as another characteristic of complex numbers, since $k = 1, 3, 5, 7$ corresponds to half of the period π, exponent

multiplier is the same as for $k = 0, 2, 4, 6$ but with the opposite sign (positive sign for even terms and negative sign for odd terms).

$$-e^{j\theta} = e^{j(\theta+\pi)} \tag{A.9}$$

Similarly, the period for each of the exponential terms inside the brackets is $k = 4$. Meaning that summations inside the brackets are the same for the pairs $k = 0, 4$; $k = 1, 5$; $k = 2, 6$ and $k = 3, 7$. This implies $2 \times 4 = 8$ operations are needed for the middle level, inside the brackets. Note that the second half, i.e. odd terms, can be calculated by changing the signs of the exponent multiplier in first half, because the distance between k and $k + 2$ is π. Therefore, the factor for $k = 0, 4$ is +1; for $k = 2, 6$ is -1; for $k = 1, 5$ is $-j$ and for $k = 3, 7$ is j.

Finally, for the outer level there is one summation for every $f(k)$ component, and the period of the exponent multiplier is 8. Which implies $1 \times 8 = 8$ operations where the second half of them is achieved by changing the signs in the first half.

As a result, on every level of computation there are 8 summations. Meaning that for N samples there are $\log_2 N$ levels and N summations required on each level. That yields to $O(N \log_2 N)$ order of number of operations. On the other hand, having a constant number of summations on every level means that the data can be processed in-place. This is the underlying idea of the fast implementation of the DFT algorithm which is called the FFT.

Given the above explanation, a formal statement of the FFT algorithm is as follows:

Let us study the steps involved in the algorithm for the previous example when $N = 8$. The first step is to reorder the discrete elements of the input function from their natural order $\{f(0), f(1), f(2), f(3), f(4), f(5), f(6), f(7)\}$ into $\{f(0), f(4), f(2), f(6), f(1), f(5), f(3), f(7)\}$.

In practice it is easy to reorder the elements by swapping the bits of the binary numbers. This is similar to applying an arithmetic mirror.

Algorithm 5 Formal statement of the FFT algorithm

```
1: Prepare the input discrete function for summations by rearranging its elements
2: for every summation level do
3:     for every exponential factor of the half-period do
4:         Calculate the exponential factor
5:         for every summation of this factor do
6:             Calculate product of the factor and the second term of the summation
7:             Calculate the summation
8:             Calculate the difference
9:         end for
10:    end for
11: end for
```

Table A.1: Reordering the numbers in binary by swapping, (mirroring), the bits.

original order	binary form	after mirroring	new order
0	000	000	0
1	001	100	4
2	010	010	2
3	011	110	6
4	100	001	1
5	101	101	5
6	110	011	3
7	111	111	7

Summation levels in the first **for** loop include parenthesis, brackets and outer level. In general, this leads to iterations on pairs, quads, octets and so on.

The second **for** loop refers to the iterations on components of the first half-period and the second half-period, (taking differences instead of summations for the first half). Note that the period of the deepest level is 2, hence the half-period is 1, which means this cycle will be executed only once. Similarly, the period for the second level is 4, hence the half-period is 2 and the cycle will be executed 2 times.

The next step is to calculate the exponential factor which is the imaginary exponent.

The innermost loop calculates the product of the exponential factor and the second term of the summation, where the exponential term changes its sign but not its absolute value, i.e. magnitude, according to Equation. A.9. To perform an in-place processing a butterfly scheme can be utilised as shown in Figure A.1

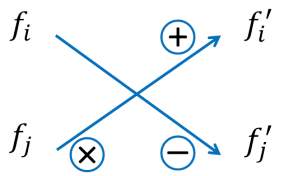


Figure A.1: The FFT butterfly scheme.

The bottom term is multiplied by an imaginary exponent and then the sum of the terms is stored in place of the upper term and the difference is stored in place of the bottom term. For our example with $N = 8$ samples the butterfly is depicted in Figure A.2.

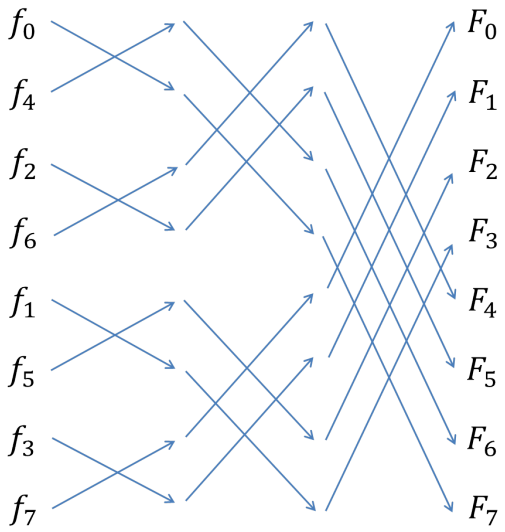


Figure A.2: Butterfly scheme for $N = 8$ samples.

A.2 Inverse transform

The inverse Fourier transform of a continuous function is defined as:

$$f(x) = \int_{-\infty}^{\infty} F(u) e^{j2\pi ux} \mathrm{d}x \tag{A.10}$$

and its discrete counterpart is

$$f(n) = \frac{1}{N} \sum_{k=0}^{N-1} F(k)\ e^{j2\pi nk/N} \tag{A.11}$$

Therefore, the difference between the forward and the inverse transforms, i.e. Equation. A.1 and Equation. A.10, is just a change in sign of the exponential factor. That means, with a slight modification, the algorithm for the forward transform can also be used for the inverse transform.

A.3 Remark

Most of the FFT algorithms that are being used in computer programs today as well as the above implementation are based on the "Cooley-Tukey" algorithm that was proposed in 1965 and became popular several years after that [122]. Cooley and Tukey reinvented the original algorithm, proposed by Gauss in 1805, and described a way to perform it efficiently on a computer system. One of the latest implementations of the FFT based on this algorithm is the FFTW [123] which is a freely available C subroutine library and can be found at http://www.fftw.org/.

Appendix B

Proof of DoG filter

A 2D difference of Gaussian filter in spatial domain is defined as:

$$h(x,y) = d_e^2 \ \exp\left(\frac{-d_e^2}{x^2+y^2}\right) - d_i^2 \ \exp\left(\frac{-d_i^2}{x^2+y^2}\right) \tag{B.1}$$

where

$$d_{e,i} = \frac{1}{\sigma_{e,i}\sqrt{2\pi}} \tag{B.2}$$

, σ_e and σ_i are excitatory and inhibitory standard deviation of the Gaussian filters. A 2D Fourier transform on $h(x,y)$ can be expressed as:

$$\mathbf{F}_2\Big[h(x,y)\Big] = H(X,Y) = \ \exp\left(\frac{-P_x^2+P_y^2}{d_e^2}\right) - \ \exp\left(\frac{-P_x^2+P_y^2}{d_i^2}\right) \tag{B.3}$$

To take angular slice in polar grid, $H(X,Y)$ can be substitute by $\Lambda_h(v,\theta)$:

$$\Lambda_h(v,\theta) = H(X,Y)|_{in\ polar\ grid}$$
$$= \exp\left(\frac{-(v\cos\theta)^2 + (v\sin\theta)^2}{d_e^2}\right) - \exp\left(\frac{-(v\cos\theta)^2 + (v\sin\theta)^2}{d_i^2}\right)$$
$$= \exp\left(\frac{-v^2}{d_e^2}\right) - \exp\left(\frac{-v^2}{d_i^2}\right) \quad \text{(B.4)}$$

Appendix C

Publications

C.1 Published journal article

UND: Unite-and-Divide Method in Fourier and Radon Domains for Line Segment Detection

Daming Shi *Senior Member IEEE*, Junbin Gao, Payam S. Rahmdel *Student Member IEEE*, Michael Antolovich, and Tony Clark

***Abstract*—In this correspondence paper, we extend our previously proposed line detection method [1] to line segmentation using a so-called "unite-and-divide" (UND) approach. The methodology includes two phases, namely the union of spectra in the frequency domain, and the division of the sinogram in Radon space. In the union phase, given an image, its sinogram is obtained by parallel 2D multilayer Fourier transforms, Cartesian-to-polar mapping and 1D inverse Fourier transform. In the division phase, the edges of butterfly wings in the neighborhood of every sinogram peak are firstly specified, with each neighborhood area corresponding to a window in image space. By applying the separated sinogram of each such windowed image, we can extract the line segments. The Division Phase identifies the edges of butterfly wings in the neighborhood of every sinogram peak such that each neighborhood area corresponds to a window in image space. Line segments are extracted by applying the separated sinogram of each windowed image. Our experiments are conducted on benchmark images and the results reveal that the UND method yields higher accuracy, has lower computational cost and is more robust to noise, compared to existing state-of-the-art methods.**

***Index Terms*—Line Segment Detection, Radon Transform, Hough Transform, Fourier Transform.**

I. Introduction

In our daily lives, we localize and recognize objects based on a variety of line segments, such as corridor guidelines and road edges. In image processing and computer vision, line segments are the most important primitive features for object localization and recognition and are used to solve many problems, such as the detection of cracks in materials and searching satellite images [2], [3]. However, in many practical applications, such as image-guided surgery and military target tracking [4], [5], the requirement of both speed and accuracy means that there is a need for more efficient and effective line segment detection methods. Hence, developing new methods to extract line segments quickly and accurately is still a challenging topic in computer vision [6].

Existing line segment detection methods fall into three categories, namely, the top-down approach, the bottom-up approach and the domain-crossing approach. The top-down approach extracts straight lines followed by segmentation; whereas the bottom-up approach starts with single pixels, which grow to segments. The domain-crossing approach takes advantage of different properties in both image space and transformed space to detect line segments.

The most commonly used top-down method is the extraction of straight lines through the Hough transform [7], [8], followed by detecting the position of endpoints in a straight line. Guil et al [9] use the fast Hough transform to obtain a straight line, order the feature points of each straight line, then determine the line segment endpoints based on the distance of feature points after a clustering procedure. However, this method cannot deal with a line whose negative slope is less than -1 and multi-segment lines. Also, if the parameter threshold of the Hough transform is not chosen properly, the algorithm will result in a pseudo-line or missing segments.

Cha et al. [10] extend the traditional 2D Hough transform to 3D in order to find segment endpoints and lengths. In this method, a Hough transform is used for each individual column or row in image space to obtain a double Hough space, resulting in the traditional Hough transform being converted from finding lines to finding line segments using multi-point voting. This method still has the problems inherent in the Hough transform, such as requiring a binary image and the difficulty in selecting an appropriate parameter threshold. In addition, after extending to 3D, the time cost to detect line segments increases dramatically [11].

Burns [12] presents a linear-time segment detection method based on the gradient direction of boundary points instead of gradient values, which is a bottom-up methodology. In a similar fashion, Desolneux et al [13] address the problem of calculating a local orientation map that is robust with respect to quantization noise, and then using local gradient information to find the meaningful alignment of possible line segments. Unfortunately, the Desolneux method is very time-consuming, as it tests every possible line segment in the image. To present a fast algorithm, Grompone von Gioi et al. [6] developed a practical and fast line segment detection program that cumulates most of the advantages of the Desolneux method by controlling pseudo-segments based on the Helmholtz principle [14].

Domain-crossing approach to line segmentation. From the point of view of machine learning, out of the three line segment detection approaches, the top-down method is a model selection technique from parameter space to data/feature space, whereas the bottom-up method is a regularization technique from data/feature space to parameter space. The optimal solution can be obtained using a two-pathway learning process, in which both top-down model selection and bottom-up regularization are considered. This indicates that the domain-crossing approach is capable of obtaining the optimal solution by using both model selection and regularization crossing different spaces.

Research results relating to the domain-crossing approach include feature-adapted beamlet transform (FABT) [15], and the most recent Du et al's neighborhood method [16], [17]. FABT is used for line segment detection and extraction of the curve structure using multi-scale analysis. Berlemont et al. [15] consider the beamlet transform as a special case of Radon transform, because they both calculate line integrals in one image. The basic idea is that the Radon transform is applied to the multi-scale method to calculate the FABT. Adaptive feature refers to the application of local filters (such as the Canny filter) before the Radon transform in order to highlight line structure in an image and to improve the accuracy and effectiveness. However, empirical studies show that the Radon transform cannot detect lines accurately in smaller scaled images, especially, when the image size is smaller than 32×32. Du et al. [16] have investigated the neighborhood between image space and parameter space in order to improve the traditional Hough transform. A neighborhood in parameter space corresponds to a diamond-shaped quadrilateral in image space. The difficulties of the method arise from the choice of

Daming Shi, Tony Clark and Payam S. Rahmdel are with the School of Science and Technology, Middlesex University, London NW4 4BT, UK (e-mail:{d.shi, t.n.clark, p.rahmdel}@mdx.ac.uk)

Junbin Gao and Michael Antolovich are with the School of Computing and Mathematics, Charles Sturt University, Bathurst, NSW 2795, Australia (e-mail:{jbgao, mantolovich}@csu.edu.au)

the radius of the neighborhood, which has a greater impact on short line segment detection.

The contribution of this paper is to propose a novel domain-crossing approach for efficient and accurate line segment extraction. The approach detects lines using the union of spectra in the Fourier domain. Line segments are detected by analyzing the division of sinogram in Radon space. The remainder of this paper is organized as follows. In Section II, our Unite-and Divide (UND) method is described in detail. Section III reports the experimental results, followed by our conclusions in Section IV.

II. Unite-and-Divide Algorithm

The frequency space of an image can be obtained with a Fourier transform, but there is no spatial information kept in frequency space. We can combine frequency space and image space according to the needs of specific features, by using another transformation as a medium. In detecting line segment features, the Radon transform can be used as this medium based on the projection-slice theorem [18].

There are two problems with the domain-crossing approach to line segment detection. The first, how to increase the number of frequencies for analysis in the Fourier domain without substantially increasing computational cost; the second, how to segment lines quickly and correctly. A unite-and-divide method (UND) is proposed to address these two problems.

A. Union of Spectra

Given an $N \times N$ image, its multilayer fractional Fourier transform can be performed by, see [19],

$$F^{\alpha}(k_1,k_2) = \sum_{n_1=-\frac{N}{2}}^{\frac{N}{2}-1} \sum_{n_2=-\frac{N}{2}}^{\frac{N}{2}-1} f(n_1,n_2) \exp\left\{-j\frac{2\pi}{N}\left(n_1\alpha k_1 + n_2\alpha k_2\right)\right\} \quad (1)$$

where $\{f(n_1,n_2)| -N/2 \le n_1,n_2 \le N/2\}$ is a 2D image in the case of line detection, and $0 < \alpha \le 1$ is a real number. The $N \times N$ frequencies of $F^{\alpha}(k_1,k_2)$ are scattered in $[-\alpha\pi,\alpha\pi] \times [-\alpha\pi,\alpha\pi]$. According to Equation (1), the combination of MLFRFTs with different α values results in a high-resolution grid. In other words, one can get the Fourier transform of an input image by combining multiple F^{α}s. Note that the computation of each Fourier layer is an independent operation and can be carried out in parallel.

Sinogram can be generated via crossing from Fourier domain to the Radon domain by the following steps: 1) mapping the Cartesian frequency samples to Polar grid 2) apply 1D inverse Fourier transform to the Polar grid. Each peak in the Radon space corresponds to a potential line in image domain. To detect the maximum peaks, an iterative *identify and remove* algorithm has been employed [23] where each peak is removed from the sinogram after it is identified. There are two circumstances involved in this method: 1) when we have a desired number of lines to be detected, algorithm terminates after certain number of iterations 2) when the number of lines is unknown so that iterations will continue for every single peak above the threshold.

B. Division of Sinogram

The sinogram detection peaks are obtained in the previous phase of Spectrum Union. These peaks in Radon space may identify straight-lines at certain angles and distances to the origin, however they cannot detect the endpoints of the line segments. In fact, the sinogram is composed of a number of butterfly-shaped waveforms centered at each peak. Segment endpoint information is contained in the butterfly edges on both sides of the peak i.e. the boundary of the butterfly wing.

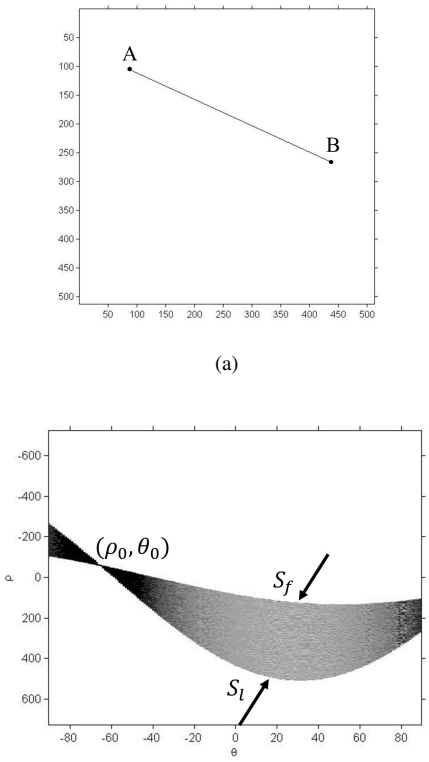


Fig. 2: Illustration of a line segment and its corresponding sinogram. (a) a segment with two endpoints. (b) Sinogram with peak corresponding to line position and butterfly-wings' boundaries corresponding to the two endpoints.

To clarify this concept, let us consider an example of single segment image and its sinogram as shown in Figure 2. From the Radon transform point of view, each feature point in the image plane corresponds to a sinusoidal curve in the Radon space. Consequently, all of the points along the straight line $\rho_0 = x\cos\theta_0 + y\sin\theta_0$ form a butterfly shape in the Radon space, intersected at peak (ρ_0,θ_0) as shown in Figure 2. In other words, the Radon transform maps every single feature point from its image plane representation into a waveform representation in a respective order. This means the first and the second endpoints of the segment (i.e., points A and B in Figure 2a) produce the first and the last boundary sinusoids of the butterfly (i.e., S_f and S_l in Figure 2b) respectively. The two arrows in Figure 2b are emphasizing on the two corresponding boundary sinusoids.

However, there are usually tens of hundreds of interwoven segments in real world images that result in overlapping butterfly-shaped waves in Radon space, making it impossible to find the boundary sinusoids for each butterfly-shaped wave. This is where it comes the need for dividing the overlapping butterfly wings into a number of single butterfly sinogram so that finding the boundary sinusoids will be an easy task. This is achievable by introducing a spatial domain window to divide the butterflies.

Spatial domain window for sinogram division: Spectra-union method generates accurate straight lines but without endpoint identification. Having the precise line parameters i.e. distance ρ and angle θ a rectangular image domain window can be designed where the target straight line plays the role of diameter for the window.

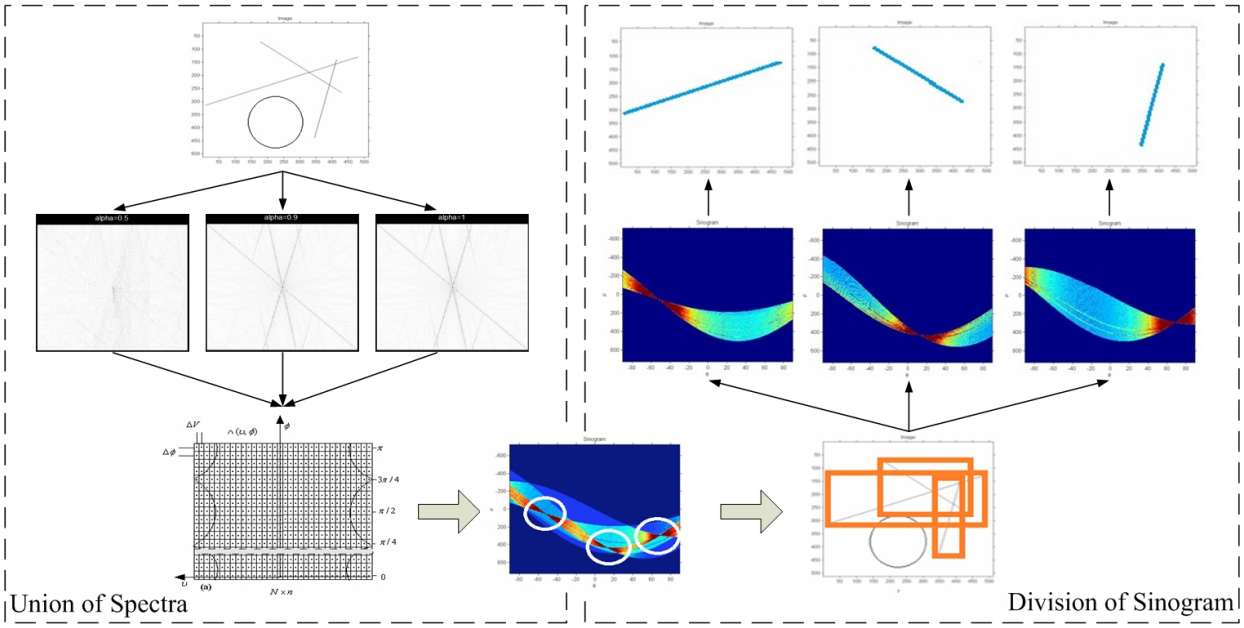


Fig. 1: The Unite-and-divide method

Size of the window is bounded by the line's outermost endpoints. A Gaussian filter can be conducted afterwards to emphasize on the linear structures and identify pixels align the line direction while masking the surrounding regions inside the window. Such an image domain filter in line direction will significantly improve the sinograms quality. Finally, a subsequent Radon transform can be obtained for the window to generate the desired single butterfly sinogram, from which boundary sinusoids are easily identifiable and capable of leading us towards the segment endpoint extraction.

C. The UND algorithm

A formal statement for the UND algorithm is given as follows:

1) **Union of Spectra.** Compute a number of Fourier transforms with different α using Equation (1), and then combine them to form a united spectrum.
2) **Crossing from Fourier domain to Radon domain.** Map the united spectrum from a Cartesian grid to a polar grid using the linear interpolation method, and then apply 1D inverse FFTs to the spectrum. Each of the 1D inverse FFTs corresponds to a sinusoid in the Radon domain according to the central slice theorem.
3) **Division of Sinogram.** Detect the maximum peaks in the sinogram, which correspond to the straight lines in the image. Each straight line may contain a multiple number of segments.
4) **Crossing from Radon domain back to image domain.** Each peak in the sinogram corresponds to a straight line in the image domain, which is the diagonal line of a window. Apply Gaussian filters along the diagonal to highlight the multi-segment line but mask the areas on its both sides.
5) **Segment detection in single-line sinogram.** Obtain the sinogram for each sub-window in step 4). Segments can be detected by analyzing the discontinuity of the butterfly wings. Each edge in the butterfly wing corresponds to the endpoint of a segment.
6) Repeat step 4) and step 5) until all the butterflies are processed.
7) End.

III. Experimental Results

In this section, the accuracy, robustness and computational time of our proposed method is verified with experiments on mixed-thickness lines [1], noisy images [20], and a natural image [6].

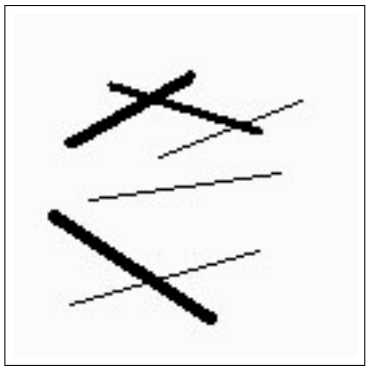


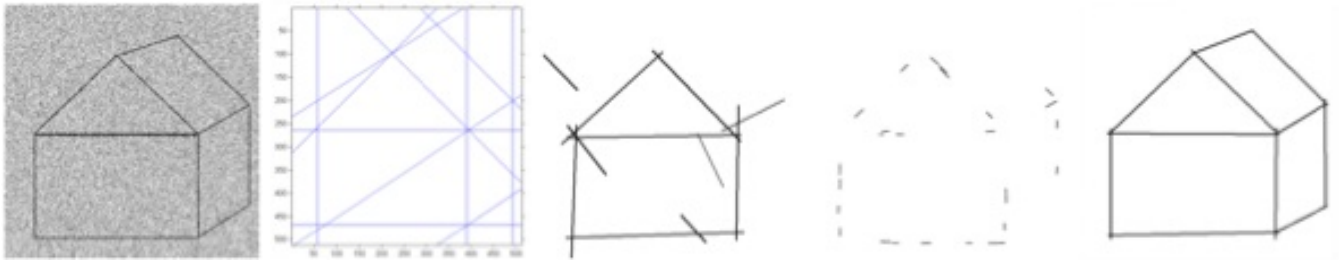
Fig. 3: Image of lines with different thicknesses

A. Verification of accuracy

The objective of this experiment is to demonstrate that the UND method can detect line segments with thicknesses greater than one pixel without pre-requisite edge detection. The test image is shown in Figure 3; and Table I gives the positions of the detected line segments which are specified by the distance ρ, the angle ϕ to the origin and the two endpoints (x_0, y_0), (x_1, y_1). The origin is set to the bottom left the image. The last row in the table gives the detection errors, which are the absolute values of the difference between the true endpoint coordinates and their detected counterparts. The results reveal that UND is capable of detecting line segments accurately from the images with insensitivity to the thicknesses of lines.

TABLE I: Positions of the Segments Detected by the UND

Lines	1	2	3	4	5	6
ρ	1.41	22.62	31.11	12.72	28.28	53.74
ϕ	-82.27	56.25	106.17	56.25	-68.21	-107.58
(x_0, y_0)	(20,19)	(14,55)	(28,59)	(35,102)	(54,74)	(19,80)
(x_1, y_1)	(91,40)	(75,14)	(99,69)	(92,84)	(107,96)	(67,106)
$errors$	(1,1)	(1,1)	(0,0)	(0,0)	(0,1)	(0,0)
	(0,0)	(0,1)	(0,0)	(1,0)	(0,0)	(0,1)



(a) (b) (c) (d) (e)

Fig. 4: Segment detection from an 11-line image with noise. (a) Original image. (b) Lines detected by HT-MLFRFT. (c) Segments detected by SHT. (d) Segments detected by LSD, (e) Segments detected by UND.

B. Robustness to noise

In this section, we examine the robustness of our line detection method to noise. We have performed the experiment as reported in [1]. Figure 4(a) shows the image of a house consisting of 11 straight lines with white noise. Figure 4(b) shows line detection, but not segmented, from our previous work. Figures 4(c-e) compare the segment detection results among the SHT, LSD, and our proposed UND.

From Figure 4, one can see UND is more robust to noise than all the other methods. When true line segments are disconnected by noise, local information, such as gradient orientation, is not capable of detecting segments properly. Since sampling in the frequency domain is not accurate enough, SHT cannot fully retrieve the connection information in image space. LSD does not perform well, because its thresholds, such as the gradient magnitude, are not adaptive but pre-determined. Since the global information is considered and accurate sampling is applied, UND is capable of detecting broken segments.

C. Line segment detection from natural images

A number of natural images reported in [6] are used to verify the performance of our proposed UND for line segment detection. In these experiment, the peak threshold for SHT and UND is set to $0.8 * p\text{Max}$, where $p\text{Max}$ is the maximum peak value in the sinogram. The main purpose of this experiment is to compare the performance of line segment detection among the representative methods such as SHT [21], RHT [22], LSD [6], HT neighborhood approach and UND in terms of computation time and line detection results, as shown in Figure 5 and Figure 6. In SHT and RHT we simply cut the lines to segments based on connected-ness.

Figure 5 shows that LSD enjoys a level of performance superior to all the other methods, and that our proposed UND is the best among the Hough transform family. From time to time, RHT cannot detect short lines because of its random sampling strategy. SHT often produces false detections, since edge orientation information is not employed in line detection. The advantage of robustness to noise can have problems. In the experiment with noisy images, UND performs very well because it is not sensitive to abrupt connections/disconnections. However, the noise-like short lines in this experiment are easily influenced by the other foreground pixels. Accordingly, short lines may not form peaks in the sinogram, which is why UND cannot detect some short segments.

However, Figure 6 shows that the UND has a better performance in detection of short segments when comparing to the recently proposed HT neighborhood approach [16]. This is due to the trade-off in selecting the neighborhood radius. Small radius will not guarantee the complete coverage of the segment while large radius fails to undertake the short segments. Segments with shorter lengths have bigger approximation errors in comparison to those with longer lengths. Effects of this compromise can be observed in the result generated from the images of truck and zebra. In addition, despite the fact that the LSD generates more segments than the UND, sometimes it breaks down a true long segment and fails to retain the continuity of the line.

In terms of speed, RHT is the fastest in the Hough transform family, but our UND is faster than SHT. Although the speed of LSD is reported elsewhere with a different computer configuration (Apple PowerBook G4 1.5GHz [16]), it can be deduced that UND should be slower than LSD.

IV. Conclusions

In this research we propose a domain-crossing methodology called UND, in which Radon transform is carried out after the union of multiple sets of spectra; and the segments are detected in the divided sinogram. Since both the union and division phases can be implemented in parallel, this method enjoys high accuracy but low computational cost. We tested the UND-based line segment detection method with noisy and real images. The results show that our method outperforms the representative line segmentation methods such as SHT, RHT and LSD.

Acknowledgment

The work of the second and third authors is partially supported by Charles Sturt University (CSU) Competitive Research Grant (OPA4818), Research Compact Grant from the Faculty of Business at CSU and Newcrest Mining Grant.

References

[1] D. Shi, L. Zheng, and J. Liu, "Advanced Hough transform using a multilayer fractional Fourier method," *IEEE Transactions on Image Processing*, vol. 19, no. 6, pp. 1558–1566, 2010.

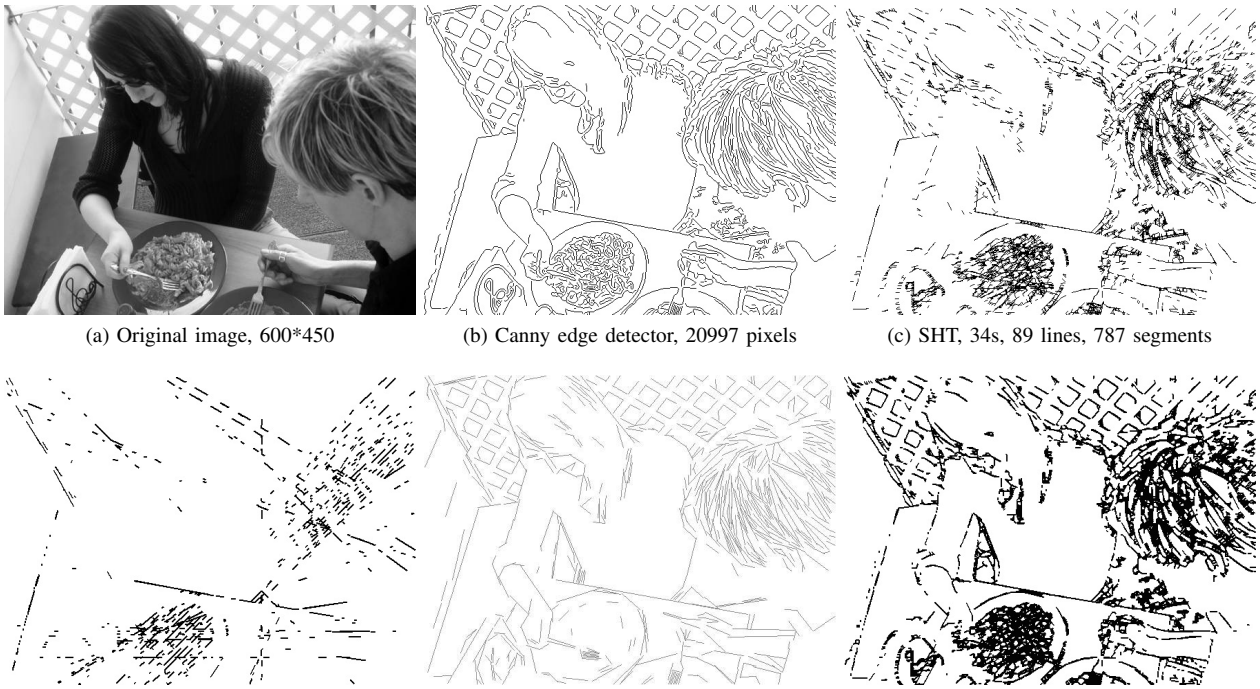
[2] T. M. Meksen, M. Boudraa, and R. Drai, "Detection of cracks in materials using the randomized Hough transform on ultrasonic images," in *Proceedings of the 6th WSEAS International Conference on Signal Processing, Computational Geometry & Artificial Vision*, 2006.

[3] D. K. San and M. Turker, "Building extraction from high resolution satellite images using Hough transform," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Science*, vol. 38, no. 8, pp. 1063–1068, 2010.

[4] M. Ding and A. Fenster, "A real-time biopsy needle segmentation technique using Hough transform," *Medical Physics*, vol. 30, no. 8, pp. 2222–2233, 2003.

[5] H. Leung, Z. Hu, and M. Blanchette, "Evaluation of multiple target track initiation techniques in real radar tracking environments," *IEE Proceedings*, vol. 143, pp. 246–254, 1996.

[6] R. Grompone von Gioi, J. Jakubowicz, J. M. Morel, and G. Randall, "LSD: A fast line segment detector with a false detection control," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 4, pp. 722–732, 2010.



(a) Original image, 600*450 (b) Canny edge detector, 20997 pixels (c) SHT, 34s, 89 lines, 787 segments

(d) RHT, 1.0s, 53 lines, 552 segments (e) LSD, 0.7s, 716 line segments [6] (f) UND, 1.5s, 121 lines, 893 segments

Fig. 5: Performance comparison of representative approaches to line detection from a natural image.

[7] P. Hough, "Method and means for recognizing complex patterns," U.S. Patent 3.069.654, Tech. Rep., 1962.

[8] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Communications of ACM*, vol. 15, pp. 11–15, 1972.

[9] N. Guil, J. Villalba, and E. Zapata, "A fast Hough transform for segment detection," *IEEE Transactions on Image Processing,*, vol. 4, no. 11, pp. 1541–1548, 1995.

[10] J. Cha, R. Cofer, and S. Kozaitis, "Extended Hough transform for linear feature detection," *Pattern Recognition*, vol. 39, no. 6, pp. 1034–1043, 2006.

[11] K. L. Chung, T. C. Chang, and Y. H. Huang, "Comment on: Extended Hough transform for linear feature detection," *Pattern Recognition*, vol. 42, no. 7, pp. 1612 –1614, 2009.

[12] J. B. Burns, A. R. Hanson, and E. M. Riseman, "Extracting straight lines," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 4, pp. 425–455, 1986.

[13] A. Desolneux, S. Ladjal, L. Moisan, and J. Morel, "Dequantizing image orientation," *IEEE Transactions on Image Processing*, vol. 11, no. 10, p. 11291140, 2002.

[14] R. L. Gregory, "Helmholtzs principle," *Perception*, vol. 36, no. 6, pp. 795–796, 2007.

[15] S. Berlemont and J. C. Olivo-Marin, "Combining local filtering and multiscale analysis for edge, ridge, and curvilinear objects detection," *IEEE Transactions on Image Processing*, vol. 19, no. 1, pp. 74–84, 2010.

[16] S. Du, B. van Wyk, C. Tu, and X. Zhang, "An improved Hough transform neighborhood map for straight line segments," *IEEE Transactions on Image Processing*, vol. 19, no. 3, pp. 573–585, 2010.

[17] S. Du, C. Tu, B. J. van Wyk, and Z. Chen, "Collinear segment detection using HT neighborhoods," *IEEE Transactions on Image Processing*, vol. 20, no. 12, pp. 3612–3620, 2011.

[18] C. G. Ho, R. C. D. Young, C. D. Bradfield, and C. R. Chatwin, "A fast Hough transform for the parametrisation of straight lines using Fourier methods," *Real-Time Imaging*, vol. 6, no. 2, pp. 113–127, 2000.

[19] W. Pan, K. Qin, and Y. Chen, "An adaptable-multilayer fractional Fourier transform approach for image registration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 3, pp. 400–414, 2009.

[20] L. Zheng and D. Shi, "Advanced Radon transform using generalized interpolated fourier method for straight line detection," *Computer Vision and Image Understanding*, vol. 115, pp. 152–160, 2011.

[21] M. Atiquzzaman and M. Akhtar, "A robust Hough transform technique for complete line segment description," *Real-Time Imaging*, vol. 1, no. 6, pp. 419–426, 1995.

[22] L. Xu and E. Oja, "Randomized Hough transform - basic mechanisms, algorithms, and computational complexities," *CVGIP: Image Understanding*, vol. 57, no. 2, pp. 131–154, 1993.

[23] M. Fiala, "Identify and remove Hough transform method," in *Proceedings of the Vision Interface*, pp. 184187, 2003.

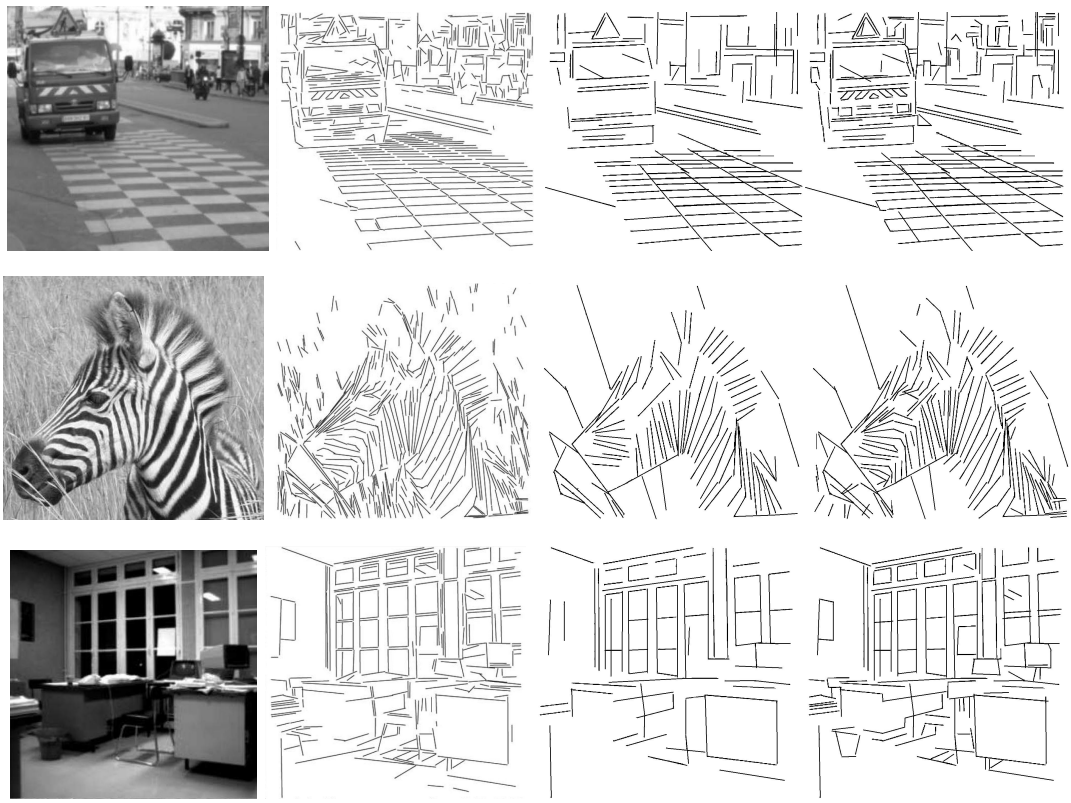


Fig. 6: Experiments on real images. From left to right are original images, results of line segmentation using LSD, HT neighborhoud and UND respectively.

C.2 Accepted for publication

Comment on "Collinear Segment Detection Using HT Neighborhoods"

Payam S. Rahmdel, Daming Shi, *Senior Member, IEEE,* and Richard Comley

***Abstract*—A novel application of the Hough transform (HT) neighborhood approach to collinear segment detection was proposed in [1]. It, however, suffered from one major weakness in that it could not provide an effective solution to the case of segment intersection. This paper analyzes a vital prerequisite step, disturbance elimination in the Hough space, and shows why, this method alone, is incapable of distinguishing the true segment endpoints. To address the problem, a unique HT butterfly separation method is proposed in this correspondence, as an essential complement to the above publication.**

***Index Terms*—Collinear segment detection, Hough Transform, disturbance elimination, segment intersection.**

I. Introduction

Extraction of straight line segments from a digital image is one of the most salient tasks in image understanding and computer vision. The Hough transform (HT) is a well-known technique for the detection of linear and curvilinear structures in an image [2], [3]. Despite its accuracy and robustness to variant noise, the HT is incapable of specifying the endpoints for a line segment. It can identify only straight lines but not segments.

Most of the proposed HT-based line segmentation methods use the Hough space information during and after the HT voting process. For instance, the connective HT (CHT) utilizes a probabilistic approach to investigate the connectively of the feature points [4]. After applying the HT, a fixation point is selected using the information obtained during the accumulation to vote for two 1-D accumulators. The progressive probabilistic Hough transform (PPHT) suggested a fast and efficient algorithm [5], [6]. The PPHT outperforms the standard HT (SHT) in terms of speed but results in lower accuracy and a large number of missing lines (i.e., false negative). Moreover, it requires a large set of parameter adjustments such as threshold that have to be delicately tuned. The problem with the accuracy was addressed in [7] but paid a high cost in terms of computation and memory requirement. An extended Hough transform (EHT) was introduced by optimizing the traditional 2-D HT with a third parameter [8]. In this 3-D representation of the HT, each individual column/row of the image space is plotted to a unique dual 2-D HT. Later in [9], an optimized algorithm was proposed to reduce the execution time during the voting process.

HT butterfly analysis considers both the peak and the area surrounding the peak. Efforts have been made to parameterize the HT butterfly [10], [11]. Recently, Du et al. proposed a set of new definitions for the HT neighborhood [12], which reveals a unique correlation between the neighborhood of straight-line in the image space and its corresponding neighborhood of the HT butterfly in the Hough space. Subsequently, the ideology was further developed and specialized for collinear line segmentation using the information concealed in the butterfly wings and seeking the rising and falling edges along the predefined resolution δ_θ [1]. However, as the authors in [1] declared, the essential prerequisite is to have a transparent butterfly which only contains the sinusoids of *collinear* feature-points, i.e., collinear segments. That implies any *non-collinear* feature-point inside the image-domain window will degrade the detection accuracy and lead to a false detection, and therefore has to be excluded.

It should be emphasized that the methodologies presented in both [1] and [12] are highly dependent on having a single-peak transparent butterfly (or sub-HT as it is called in [1]) for each segment. Therefore, eliminating occlusion caused by non-collinear segments is an inevitable task and failing to operate this step flawlessly will produce unreliable results, i.e., wrong endpoints.

Despite the authors' appreciable concern to address this problem, the method proposed in [1] fails to isolate the disturbing feature-points completely and therefore fails to detect the true endpoint coordinates. In the following, we further analyze this problem and propose a reliable solution to overcome this issue.

Payam S. Rahmdel, Daming Shi, and Richard Comely are with the School of Science and Technology, Middlesex University London, London NW4 4BT, UK (e-mail:{p.rahmdel, d.shi, r.comley,}@mdx.ac.uk)

II. Problem Statement

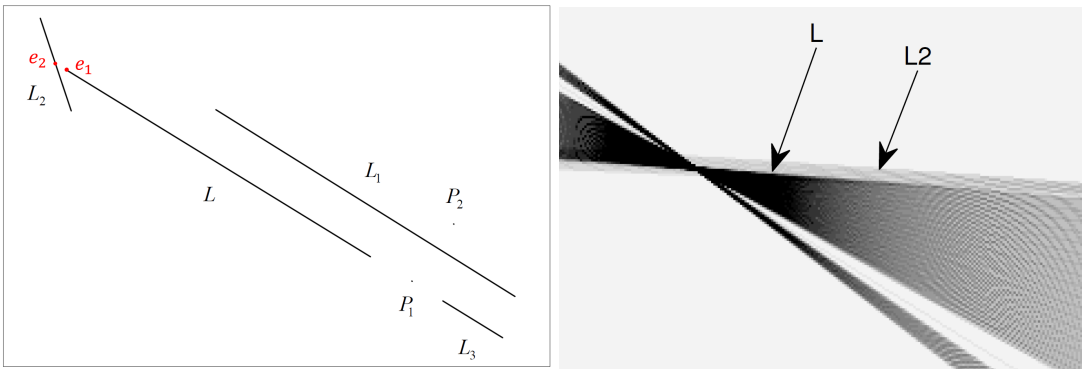
The disturbance elimination proposed in [1], i.e., *window of interest*, is capable of excluding the feature-points outside the window. However, this cannot guarantee that the remaining feature-points inside the window are all collinear. To clarify the problem, let us consider a test image and its corresponding sub-HT as shown in Fig. 1. Fig. 1(a) shows a synthetic image with segment L as the target straight-line segment, to be detected. Fig. 1(b) shows the corresponding sub-HT when the effect of disturbing feature-points is eliminated thanks to the windowing approach. Fig. 1(a) and (b) are duplicates of Fig. 10 and 11, reported in [1], respectively.

It can be seen from Fig. 1(b) that, even after applying the image-domain window, sinusoids of L_2, (the line which intersects with L's extension), still superimpose with those of L. Consequently, seeking the rising-edge along the predefined δ_θ will mistakenly lead us to admit e_2 (in Fig. 1(a)) as L's first endpoint rather than the true endpoint which is e_1. Reducing the neighbohood's radius r cannot alleviate the error either, because, there is always a trade-off in choosing a proper radius; a small radius leads to ignorance of short segments [12]. The problem occurs when the target segment, or its extension, intersects with one or more non-collinear segments within the image boundary. As a result, any non-collinear feature-points inside the window will also get voted, alongside the collinear feature-points, during the sub-HT process (see the shadow of L_2 in Fig. 1(b)). Therefore, the essential condition is to have the effect of any disturbing feature point inside the image-domain window excluded.

Despite the novelty of the windowing approach suggested in [1], it is unable to eliminate the disturbance when one or more segments intersect with the target segment or its extension. This suggests an essential enhancement and filtering technique to exclude the remaining non-collinear feature-points. That motivated us to propose an improved windowing approach, to be considered as an essential complement to Du et al.'s approach.

III. Improved windowing approach

Similar to [1], this enhancement has to be performed after the first HT, thus, line parameters, i.e., ρ_i and θ_i are known. Basically, the algorithm can be explained in two major steps:



(a) (b)

Fig. 1: Problem of windowing approach. (a) Original image with target segment L. (b) Sinogram captured using Du et al.'s windowing approach. Sinusoids of L_2 are still superimposing with those of L.

Step 1) Seek the lines which intersect with the target segment or its extension. The intersection has to occur within the image coordinates. This is straightforward because line parameters, i.e., $(\vec{\rho}, \vec{\theta})$, are known. Having the parameters, straight-line equations can be constructed by $\rho_i = x\cos\theta_i + y\sin\theta_i$. Thus, intersecting lines can be found by solving a series of simultaneous equations. Algorithm. 1 synthesizes the algorithm in pseudo-code.

Step 2) For each intersecting line of every target segment construct a mask window W_m with radius r_m. But this time mask the feature-points inside the window by assigning every pixel to zero. Note that, such a window plays the role of a mask-filter therefore the masking radius r_m can be as small as 3 to 5 pixels. Algorithm. 2 summarizes the procedure.

Algorithm 1 Find and store lines having intersection within image

Input: $\vec{\rho}, \vec{\theta}$ /* From the HT /*
$S = [0]_{n\times n}$ /* n is the total number of lines /*
for all i such that $1 \le i \le (n-1)$ **do**
  for all j such that $(i+1) \le j \le n$ **do**
    $x \leftarrow (\rho_i \sin\theta_j - \rho_j \sin\theta_i)/(\cos\theta_i \sin\theta_j - \cos\theta_j \sin\theta_i)$
    $y \leftarrow (-\rho_i \cos\theta_j + \rho_j \cos\theta_i)/(\cos\theta_i \sin\theta_j - \cos\theta_j \sin\theta_i)$
    /* Gets the coordinates of the intersection point /*
    if (x, y) is inside image coordinates **then**
      $S[i, j] \leftarrow$ True
    else
      $S[i, j] \leftarrow$ False
    end if
  end for
end for
$S \leftarrow S + S^T$
Output: S /* Intersection matrix /*

Algorithm 2 Create the window (mask-filter) for each line and apply the sub-HT.

Input: Original image I,
$P = 0, H = 0$
for all i such that $1 \le i \le n$ **do** /* for every straight-line/*
  /* Construct a window W as suggested in [1] /*
  /* and exclude the feature-points outside the window /*
  /* I_i is the original image when masked with W
  Compute $I_i = I \ \& \ W$
  $P \leftarrow find(S[i])$ /* Returns indexes of intersecting lines for ith line /*
  if $P \neq 0$ **then**
    for all j such that $1 \le j \le$ length of P **do**
      /* Construct the mask window W_m /*
      Compute $I_i = I_i \ \& \ \overline{W_m}$
    end for
  end if
  Compute $H_i = Hough(I_i)$ /* Returns the sub-HT for the ith target segment /*
end for
Output: H

IV. Location of Intersection

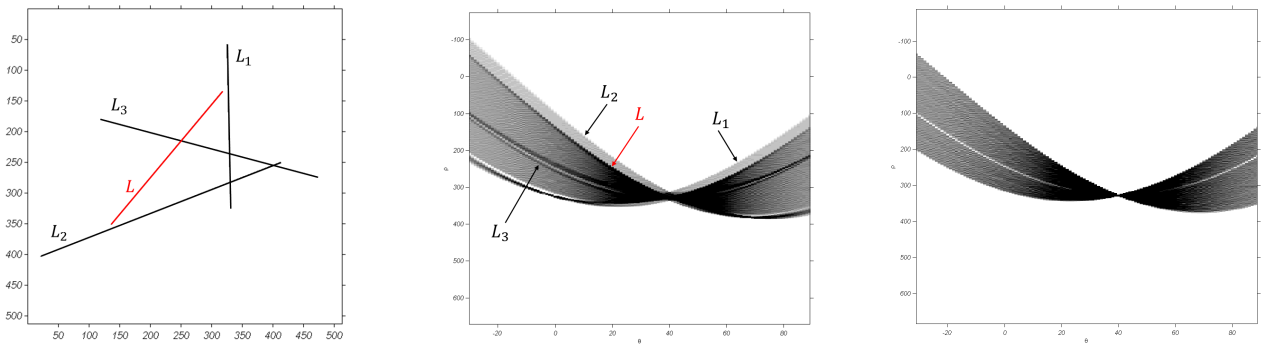
There will be two possibilities in terms of where the intersection occurs. Either the intersection occurs with the target segment itself or with its extension. Each of these scenarios has a different effect on the butterfly's shape.

A. Intersection with extension of the target line

In this case, the non-collinear lines can be easily eliminated using the mask-filter without affecting the feature-points of the target segment. Let us consider the target straight-line segment L shown in Fig. 2(a). L_1 and L_2 are two non-collinear segments having intersections with L's extension. Fig. 2(b) shows the result of Du et al.'s disturbance elimination method. It can be seen that L_1 and L_2 affect the butterfly's significant edge. That will eventually lead to a large error. In Fig. 2(c) these sinusoids have been completely removed thanks to the improved windowing approach explained above.

B. Intersection with the target line

Eliminating feature-points of a line which has an intersection with a target segment, (e.g. L_3 in Fig. 2(a)), may split a continuous target segment into two collinear segments. Accordingly, the resulting butterfly wing will have a discontinuity in its sinusoids, (e.g. a white gap in the butterfly wings in Fig. 2(c)). To avoid breaking the segment into two, a radius smaller than d_r, (the *collinear segment distinguishing distance* [1]) is chosen. If $r_m > d_r$, the result will be two collinear segments instead of one continuous segment. Therefore, choosing $r_m < d_r$ can guarantee the correct detection of a continuous segment.



(a) (b) (c)

Fig. 2: Comparison between the sub-HT butterflies. (a) Synthetic image of 512×512 pixels. (b) Butterfly captured via Du et al.'s method. (c) Butterfly captured via improved windowing method proposed in this paper.

V. Experiments

A detailed qualitative analysis has been made to compare the performance of the proposed algorithm with Du et al.'s method. A real gray-scale image of size (512×512) with ground-truth information of the number of segments and the coordinates of the endpoints has been used as shown in Fig. 3(a). Fig. 3(b) shows the edge map where the target segments S_1 to S_6 have been labelled for the sake of clarity. The result of an SHT using MATLAB's *hough()* function is depicted in Fig. 3(c). L_1 to L_6 are corresponding straight lines passing through S_1 to S_6, respectively. Algorithm.1 generates the line intersection matrix given in Table. I, where lines that are intersecting with each target segment are identified.

TABLE I: Result of Algorithm.1. Lines that intersect with each target segment are found.

	L_1	L_2	L_3	L_4	L_5	L_6
L_1	0	0	1	1	0	0
L_2	0	0	1	1	0	0
L_3	1	1	0	0	1	1
L_4	1	1	0	0	1	1
L_5	0	0	1	1	0	0
L_6	0	0	1	1	0	0

Fig. 3(d)-(i) illustrates the result of Du et al.'s disturbance elimination method with radius $r = 6$. Shadows of disturbing feature points are visible around the butterfly wings, that eventually lead to large deviation from the ground-truth endpoints. Taking the butterfly of S_1 as an instance (Fig. 3(d)), the shadows are the consequence of feature points of S_3 and S_4 which intersect with S_1, (refer to Table. I). The result of Du et al.'s method is shown in Fig. 3(p) where the considerable effect of S_3 and S_4 on the endpoints of S_1 and S_2 is visible. Fig. 3(j)-(o) represent the result of the improved windowing approach, where the disturbing feature points have been removed using Algorithm. 2. The segmentation result is depicted in Fig. 3(q).

Table. II provides a detailed comparison with the ground-truth. Euclidean distance from the detected endpoint to the true endpoint has been measured as the deviation criterion.

VI. Conclusion

This paper highlighted a major problem of two recently proposed interrelated publications of this Transactions, [1] and [12]. The enhancement proposed in this paper can be seen as an essential complement to the above publications. The MATLAB source code will be made available to all the readers upon request.

References

[1] S. Du, C. Tu, B. J. van Wyk, and Z. Chen, "Collinear segment detection using HT neighborhoods," *IEEE Transactions on Image Processing*, vol. 20, no. 12, pp. 3612–3620, December 2011.

[2] P. Hough, "Method and means for recognizing complex patterns," U.S. Patent 3.069.654, December 1962.

[3] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Graphics and Image Processing*, vol. 15, pp. 11–15, January 1972.

[4] S. Y. K. Yuen, T. S. L. Lam, and N. K. D. Leung, "Connective Hough transform," *Image and Vision Computing*, vol. 11, no. 5, pp. 295 – 301, 1993.

[5] C. Galamhos, J. Matas, and J. Kittler, "Progressive probabilistic hough transform for line detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 1999, pp. 554–560.

[6] J. Matas, C. Galambos, and J. Kittler, "Robust detection of lines using the progressive probabilistic Hough transform," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 119 – 137, 2000.

[7] T. T. Nguyen, X. D. Pham, and J. Jeon, "An improvement of the standard Hough transform to detect line segments," in *IEEE International Conference on Industrial Technology*, 2008, pp. 1–6.

[8] J. Cha, R. Cofer, and S. Kozaitis, "Extended Hough transform for linear feature detection," *Pattern Recognition*, vol. 39, no. 6, pp. 1034–1043, 2006.

[9] K.-L. Chung, T.-C. Chang, and Y.-H. Huang, "Comment on: Extended Hough transform for linear feature detection," *Pattern Recognition*, vol. 42, no. 7, pp. 1612–1614, 2009.

[10] J. Ji, G. Chen, and L. Sun, "A novel Hough transform method for line detection by enhancing accumulator array," *Pattern Recognition Letters*, vol. 32, no. 11, pp. 1503–1510, 2011.

[11] Y. Furukawa and Y. Shinagawa, "Accurate and robust line segment extraction by analyzing distribution around peaks in Hough space," *Computer Vision and Image Understanding*, vol. 92, no. 1, pp. 1–25, 2003.

[12] S. Du, B. van Wyk, C. Tu, and X. Zhang, "An improved Hough transform neighborhood map for straight line segments," *IEEE Transactions on Image Processing*, vol. 19, no. 3, pp. 573–585, March 2010.

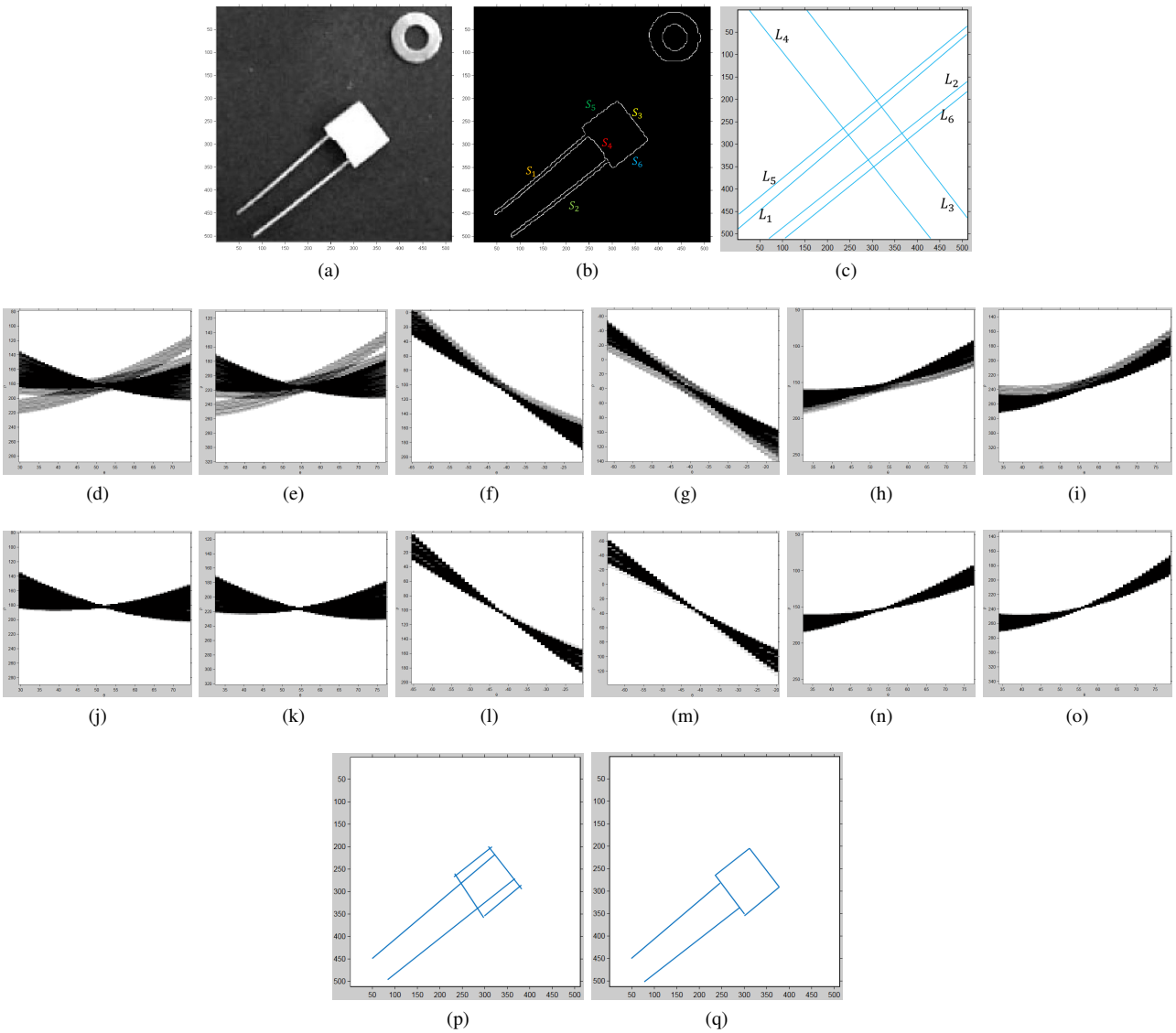


Fig. 3: (a) original gray-scale image. (b) edge map with labelled segments. (c) result of SHT. (d)-(i) result of Du et al.'s method. (j)-(k) result of the Algorithm.2. (p) Du et al.'s segmentation result. (q) segmentation using the proposed method.

TABLE II: Detailed detection comparison of the methods with the ground-truth data. The proposed method has a lower deviation from the ground-truth.

	Ground-truth		Du et al. [1]				Proposed method			
			First endpoint		Second endpoint		First endpoint		Second endpoint	
Line segment	First endpoint coordinates	Second endpoint coordinates	Detected coordinates	Deviation from ground-truth	Detected coordinates	Deviation from ground-truth	Detected coordinates	Deviation from ground-truth	Detected coordinates	Deviation from ground-truth
S_1	(46,452)	(243,282)	(51,448)	6.4	(332,218)	109.6	(48,451)	2.2	(245,281)	2.2
S_2	(80,502)	(290,335)	(84,497)	6.4	(336,272)	78.0	(79,502)	1.0	(289,337)	2.2
S_3	(315,206)	(378,287)	(308,200)	9.2	(383,294)	8.6	(313,205)	2.2	(378,290)	3.0
S_4	(234,266)	(301,352)	(233,260)	6.0	(299,359)	7.2	(235,265)	1.4	(301,351)	1.0
S_5	(233,264)	(310,204)	(232,266)	2.2	(316,200)	7.2	(234,265)	1.4	(312,205)	2.2
S_6	(302,352)	(379,290)	(300,353)	2.2	(382,285)	5.8	(301,354)	2.2	(378,291)	1.4

C.3 Conference proceedings

To be appear in *Proceedings of IEEE 56th International Midwest Symposium on Circuits and Systems*, Columbus, Ohio, USA, August 2013.

Radon Sinogram Decomposition for Line Segmentation

Payam S. Rahmdel, Daming Shi, and Richard Comley
School of Science and Technology
Middlesex University London
London NW4 4BT, United Kingdom
Email: {p.rahmdel, d.shi, r.comley}@mdx.ac.uk

***Abstract*—In this paper, a novel line segmentation algorithm is proposed as an extension to our previous works, i.e., line detection using multilayer fractional Fourier methods. First, a multilayer Fourier-based Radon transform (RT-MLFRFT) is applied to detect the location and the orientation of the straight lines. Afterwards, a set of windows-of-interest in spatial domain is designed to decompose the complex Radon space to a number of Radon spaces with a single transparent butterfly. This is to eliminate the effect of the neighbour RT peaks and guarantee the correct segment detection. Experimental results show that the proposed method enjoys superior performance compared with existing similar representative works.**

I. Introduction

Extraction of straight line segments from a digital image is one of the most salient tasks in image understanding and computer vision. A significant number of straight line detection techniques have been proposed by the computer vision community. All of the existing methods can be classified into two groups: namely, top-down and bottom-up approaches. The former extracts lines that are broken down to segments, whereas the latter accumulates pixel by pixel to form segments. Recently proposed line segment detector (LSD) is a decent example of bottom up approaches [1]. The Hough transform (HT), on the other hand, is indeed one of the most well-known top-down techniques for detection of line profiles in a typical image array [2]. It is based on transforming a given image from the spatial space to its corresponding parameter space. The so-called Hough space consists of θ as the angle of the normal from the origin to a straight line and ρ as the length of this normal. Using this definition, each line in the image space can be identified by $\rho = x\cos\theta + y\sin\theta$ where x and y are the coordinates of the corresponding feature point in the spatial domain.

Despite the fact that the HT is an accurate technique for line profile extraction in noisy or cluttered images, it is incapable of specifying the endpoints for a line. It can identify only lines that pass through the entire image.

Extended Hough transform (EHT) optimized the traditional 2-D HT with a third parameter [3]. In this 3-D representation of the HT, each individual column/row of the image space is plotted to a unique dual 2-D HT. Later in [4], optimized algorithm was proposed to reduce the execution time during the voting process. However, it requires a prior edge detection to enhance the linear features. Recently, a novel geometrical technique based on the neighborhood of straight line segments in both the spatial and parameter domain was introduced [5], [6]. The neighborhood of a line segment in the parameter domain is defined in a lozenge-like quadrangle to approximate the neighborhood of the segment in the image domain. Nevertheless, for the short segments accurate neighborhood radius selection is a tedious job. Segments with shorter lengths have bigger approximation error in comparison to those with longer lengths. Instead of using the HT in binary image, the Radon transform (RT) can be replaced in a grayscale image. Feature-adapted beamlet transform provided a practical solution to detect linear profiles using RT in a multi-scale manner [7]. However, linear filtering prior to RT requires a considerable computational expense.

Therefore, the principal aim of this research is to propose an accurate and reliable algorithm to extract the endpoints of a line segment after RT computation. The remainder of this paper is organized as follows: Section II is a review of the Fourier-based RT proposed in our previous work followed by our segment detection approach in Section III. Multi-segment detection using windows of interest in the spatial domain is shown in Section IV. Experimental results are given in Section V prior to our conclusion in Section VI.

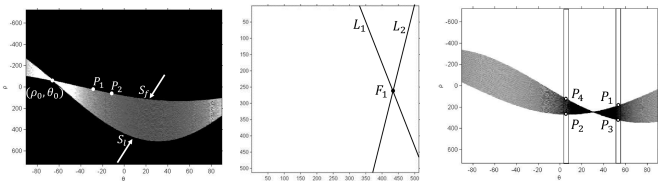
II. Radon Transform using Multilayer Fractional Fourier Transform

Equivalently and due to the mathematical identity, the HT can be obtained via the RT [8] because it performs a number of line integrals along the variant angles of θ using equation

$$Rf(x,y) = \int\int f(x,y)\delta(x\cos\theta + y\sin\theta - \rho)\,\mathrm{d}x\mathrm{d}y$$

where $Rf(x,y)$ is the 2-D forward Radon transform of the $f(x,y)$. Straight line detection using an advanced Fourier-based RT was introduced in [9],[10]. The Fourier spectrum of an image was constructed through a multilayer fractional Fourier transform (MLFFT) [11], defined as

$$F^{\gamma}(X,Y) = \sum_{x=-\frac{N}{2}}^{\frac{N}{2}-1}\sum_{y=-\frac{N}{2}}^{\frac{N}{2}-1} f(x,y) \times e^{-j2\pi(x\gamma X + y\gamma Y)/N}$$



(a) (b) (c)

Fig. 1. (a) Sinogram of a line segment captured via RT-MLFRFT. (b) Two representative straight lines $L1$ and $L2$ of two random boundary points P_1 and P_2. (c) Selection of the two columns along θ

where $\{f(x,y)| -\frac{N}{2} \leqslant x,y \leqslant \frac{N}{2}-1\}$ and $0 < \gamma \leqslant 1$ is a fractional scaler for each layer of DFT. The resulting frequency response of the image contains more frequency samples compared with conventional one-layer 2-D DFTs. Therefore, one can obtain more information of pixel intensity variation to increase both the performance and the accuracy of the RT. Furthermore, these its frequency samples are closer to Polar coordinates in comparison to those in a 2-D DFT. Thus, the interpolation error is reduced during the Cartesian to Polar mapping procedure.

Despite the fact that the RT using MLFRFT is an accurate and robust technique for extracting any line profiles in a noisy and cluttered image, it is unable to distinguish between the straight lines and straight line segments. This method can only identify lines passing through the entire image. In the following, we show how it is possible to identify the endpoints of the lines by seeking the boundary information of the Radon butterflies.

III. Radon boundary analysis

Each feature point in the image plane corresponds to a sinusoidal curve in the Radon space. Consequently, all of the points in a straight line $\rho_0 = x\cos\theta_0 + y\sin\theta_0$ form a butterfly shape in the Radon space, intersected at peak (ρ_0, θ_0) as shown in Fig. 1(a). Let us focus only on the boundary of the sinogram and investigate its ability to provide us with the required information to find endpoints of the line segment. In fact, this information is laid on the first and the last sinusoids (i.e., S_f and S_l) crossing the peak at (ρ_0, θ_0) in Fig. 1(a).

Similarly, each sinusoidal curve in Radon space represents a number of straight lines intersecting on a common feature point in the image plane. As a result, one can randomly choose at least two points along a sinusoid and draw the corresponding straight lines in the image plane. The intersection of the two lines would be the location of the target feature point. In other words, to capture the coordinates of a feature point in image space one must obtain at least two points from its corresponding sinusoid and, in image plane, find the intersection of the two lines representing the two sinusoid's points. In Fig. 1(a), P_1 and P_2 are two randomly selected points along the butterfly's boundary. Straight lines $L1$ and $L2$ in Fig. 1(b) are representatives of P_1 and P_2 in the image plane. Finally, intersection point F_1 is the target feature point, which in this case is one of the endpoints of line segment. This scenario can be applied to all other sinusoids in a sinogram. Since we are interested only in finding the endpoints of the line segment therefore, we need to consider those two sinusoids that are lain on the boundary of a butterfly wing. In fact, sinusoids S_f and S_l are representatives of this boundary. Thus, in this case, F_1 can be considered as one of the endpoints of our segment since P_1 and P_2 are located on the sinusoid S_f.

To extract the true parameters from the boundary sinusoids, we chose two columns on either side of the peak in the θ direction with $\Delta\theta$ as the interval between θ_0 and the column. Fig. 1(c) shows this concept. The aim is to find the coordinates of P_1, P_2, P_3, P_4. This is feasible by seeking the indexes of rising and falling edges along the columns.

In fact, P_1 and P_2 are responsible for two straight lines crossing the first endpoint; similarly, P_3 and P_4 are responsible for two straight lines crossing the second endpoint. We can then find the intersection of P_1,P_2 and P_3,P_4 to obtain the endpoints of the line segment. Algorithm 1 provides a summary of the algorithm.

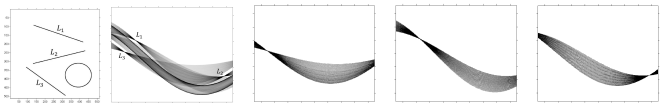
Algorithm 1 Segment detection using Radon boundary information

Input: Image I ($n \times n$ pixels), Δ_θ

$[C1, C2] = 0$, $[P_1, P_2, P_3, P_4] = 0$
Compute $S = R(I)$
/* S is the 2-D Radon transform sinogram /*
/* extract the line parameters (ρ_i, θ_i) /*
$C1 \leftarrow$column$(S, [\theta_1 = \theta_0 + \Delta_\theta])$
$C2 \leftarrow$column$([S, [\theta_2 = \theta_0 - \Delta_\theta)])$
/* Column(S, θ) returns the corresponding /*
/* column vector θ_i from the matrix S /*
/* index(C) returns the $[P_F, P_L]$, the vector indices /*
/* of the two boundary points /*
$[P_1, P_3] \leftarrow$ index$(C1)$
$[P_2, P_4] \leftarrow$ index$(C2)$
$(x_1, y_1) \leftarrow$ intersection $([P_1, \theta_1]), [P_2, \theta_2])$
$(x_2, y_2) \leftarrow$ intersection $([P_3, \theta_1]), [P_4, \theta_2])$
/* intersection(k_1, k_2) retunes the intersection /*
/* of k_1 and k_2 in the image plane /*

Output: (x_1, y_1) and (x_2, y_2), coordinates of segment the endpoints

Also, a set of four simultaneous equations can be solved to compute the coordinates of the endpoints from P_1,P_2 and P_3,P_4 [12]. However, the detection precision can be unfavourably affected when the number of segments increases. This will lead to having a complex sinogram and superimposed butterfly wings. This is a recurrent problem in the image processing applications since there are typically tens of line segments in an image. In fact, the essential condition prior to applying the segment detection algorithm would be to have a transparent single butterfly sinogram when the impressions of the other peaks are eliminated. In this way, a typical solution could be to define a "window-of-interest" around the HT peaks [13]; however, it would not guarantee the complete elimination of adjacent peaks and their overlapping sinusoids. To apply



(a) (b) (c) (d) (e)

Fig. 2. (a) Mixed shape image. (b) its overlapping sinogram. (c), (d) and (e) Three isolated butterflies

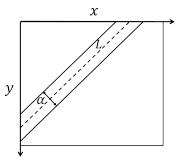


Fig. 3. Illustration of window-of-interest in image plane for straight line L. α is the width of the window.

this window in the image plane, the authors in [6] used the segment's neighborhood. However, to obtain such a window one must go through the entire process of building a proper neighborhood and compromise on the radius selection for each segment. In the following section, we will discuss how to choose an appropriate window-of-interest from the image plane without neighborhood identification.

IV. Windowing approach for butterfly break-up

A. Window definition

Fig. 2(a) shows a benchmark image, consisting of a mixture of three straight line segments and a circle [14], [9], [10], with its corresponding superimposed butterflies in Fig. 2(b). It can be seen, that the sinusoids of L_2 are overlapped with those of L_1, L_3, and the circle. Accordingly, seeking boundary information would be a problematic task. However, if we could decompose the sinogram in a way that, each butterfly appears in a separate Radon space, the precession of the Algorithm 1 can be guaranteed. In this way, we define a set of windows in spatial space pointing out that the number of windows are equal to the number of Radon peaks derived via the multilayer approach. Having the parameters of the straight line (i.e., orientation and location) after RT-MLFRFT, an image domain window can be defined as a rectangular area around the line. A predefined parameter α is the width of this window. We can select the α as $t+2$ pixels where t is the maximum line thickness in pixels. The length of the window is restricted to the image boundaries. Fig. 3 illustrates our definition of window-of-interest in the image plane.

B. Window implementation

For each Radon peak a window is created. By applying the RT to each windowed image a sub-RT is then computed. The resulting sub-RT has the same size as the original RT but it contains only the sinusoids that are belong to the feature points within the window. Fig. 2(c), 2(d) and 2(e) shows three transparent sinograms decomposed from their complex sinogram in Fig. 2(b) using the proposed window-of-interest. The butterfly corresponding to the circle would not appear in this decomposition because its sinusoids did not contribute in the peak detection process after RT-MLFRFT. A block diagram of the process is drawn in Fig. 4.

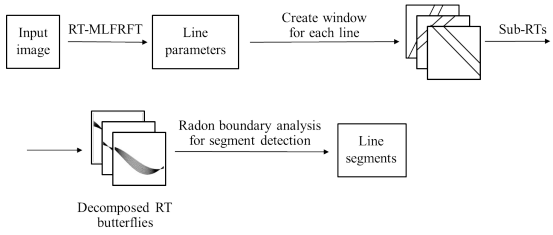


Fig. 4. Block diagram of the proposed line segment detection algorithm

It should be emphasized that, implementing the algorithm in a parallel way can reduce the computational time. Since computing each sub-RT is an independent process, it is easy to calculate the whole set of sub-RTs simultaneously using multi-core CPU systems.

V. Experimental Results

In this section the performance of the proposed algorithm is investigated in terms of segment detection accuracy and robustness in presence of noise. These experiments are based on our own data set and those reported in the literature.

A. Experiments on synthetic data set

Here we built a data set of 175 synthetic images each containing J lines mixed with one/two curvilinear structures where $J = 8$, 10, 13, 15, 20, 25, and 30. For each value of J, 5 distinct manners of object occurrence have been randomly chosen in terms of orientation and position. Then different levels of noise density d have been added to evaluate the detection accuracy. d varies from 0 to 0.5 for the *salt and paper* noise. Fig. 5 shows a subset of five selected images from the data set with $J = 10$ and noise density $d = 0.50$. A comparative study of this experiment with extended Hough transform (EHT) [4], line segment detector (LSD) [1], and the algorithm proposed in this paper is indicated in Table I. The results exhibit a moderate performance for the EHT. This fairly stable behaviour to variant noise is due to the HT's voting process and the minimum threshold value to distinguish the potential straight lines from other image components. Nevertheless, edge detection prior to EHT causes a major inaccuracy in the final result by neglecting the true feature points. LSD shows an absolute precision in less noisy data; however, it is unable to preserve the accuracy in higher noise levels. These noises affect the region growing process of the LSD. The proposed segment detector based on the RT-MLFRFT demonstrates an adequate precision in less noisy data while it preserves the precision immutability in even highly cluttered images. The reason can be found in the use of 1) RT-MLFRFT to accurately detect the location and orientation of possible line segments [9]; 2) Windowing approach described in Section IV to precisely remove the effect of redundant feature points in the image plane; and 3) Segment detection algorithm based on the butterfly boundary analysis presented in Section III.

TABLE I
AVERAGE RATE OF LINE SEGMENT DETECTION ACCURACY IN PRESENCE OF VARIANT NOISE (%).

J	8	10	13	15	20	25	30
			$d = 0$				
EHT	98	98	98	98	95.5	95.5	95.5
LSD	100	100	100	100	100	100	100
Proposed method	98	98	98	98	95.5	95.5	93.3
			$d = 0.30$				
EHT	93.3	93.3	93.3	91	91	88	88
LSD	75	73	73	73	70	70	69
Proposed method	97	97	95.5	95.5	95.5	93.3	93.3
			$d = 0.50$				
EHT	88	85	86	84	82	81	81
LSD	55	55	54	53	53	51	51
Proposed method	92	92	90.3	89	89	87.3	87.3

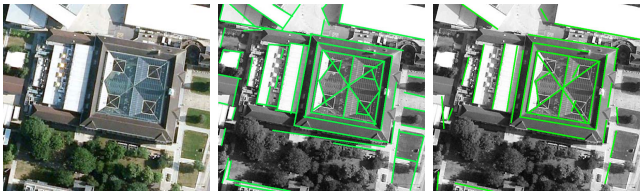
Fig. 5. A set of five selected samples from data set of 175 image. $J = 10$ and $d = 0.50$

B. Experiments on real data

For this experiment we applied our proposed segment detection algorithm to a satellite image shown in Fig. 6. The comparison has been made with Du's neighborhood approach. For the neighborhood's predefined parameters we chose radius $r = 3$ pixels, minimum length of the segment $l = 20$ pixels, and collinear segment distance $d = 30$ to compute the collinear segment resolution δ_θ. As for the proposed segment detector we chose the $\Delta_\theta = 20$ degree from θ_0. The result shows the proposed method has been able to detect most of the significant lines as well as short segments while the neighborhood approach could not consider the short lines (e.g., border of the small buildings around the main building). However, in the case of collinear segments, when two or three lines share an exact angle (e.g., Fig. 6(c), the three segments of the roof border on the right-hand side of the main building) the neighborhood approach performs better. The current algorithm is designed in a way that the segment with the largest number of pixels is captured and the effect of other collinear features is eliminated. This issue will be considered in our future work.

VI. CONCLUSION

In this paper, we presented a method to find line segments from the peaks generated via Radon transform. Regardless of undesirable extra parametrisation we sought boundary information of sinogram butterflies. In fact, the first and the last sinusoid of a peak lead us to the location of the endpoints. Accurate extraction of boundary information relies on having a transparent butterfly when the influence of other sinusoids are eliminated. This extraction will be negatively affected when we face increases in the number of segments. Therefore, a windowing approach in the image domain was conducted after the RT-MLFRFT for multi-segment images with complex parameter space. Since all the windowing operation are independent, parallel computing using multi-core CPUs can be used to reduce the computational time.



(a) (b) (c)

Fig. 6. Comparison between Hough transform neighborhood approach and the proposed method. (a) The original satellite image. (b) and (c) Results of the proposed method and neighborhood mapping respectively.

REFERENCES

[1] R. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "LSD: A fast line segment detector with a false detection control," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 722 –732, April 2010.

[2] P. Hough, "Method and means for recognizing complex patterns." U.S. Patent 3.069.654, December 1962.

[3] J. Cha, R. Cofer, and S. Kozaitis, "Extended Hough transform for linear feature detection," *Pattern Recognition*, vol. 39, no. 6, pp. 1034 – 1043, 2006.

[4] K.-L. Chung, T.-C. Chang, and Y.-H. Huang, "Comment on: Extended Hough transform for linear feature detection," *Pattern Recognition*, vol. 42, no. 7, pp. 1612 – 1614, 2009.

[5] S. Du, B. van Wyk, C. Tu, and X. Zhang, "An improved Hough transform neighborhood map for straight line segments," *IEEE Transactions on Image Processing*, vol. 19, pp. 573 –585, March 2010.

[6] S. Du, C. Tu, B. J. van Wyk, and Z. Chen, "Collinear segment detection using HT neighborhoods," *IEEE Transactions on Image Processing*, vol. 20, pp. 3612 –3620, December 2011.

[7] S. Berlemont and J.-C. Olivo-Marin, "Combining local filtering and multiscale analysis for edge, ridge, and curvilinear objects detection," *IEEE Transactions on Image Processing*, vol. 19, pp. 74 –84, January 2010.

[8] S. R. Deans, "Hough transform from the Radon transform," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 3, pp. 185 –188, March 1981.

[9] D. Shi, L. Zheng, and J. Liu, "Advanced Hough transform using a multilayer fractional Fourier method," *IEEE Transactions on Image Processing*, vol. 19, pp. 1558 –1566, June 2010.

[10] L. Zheng and D. Shi, "Advanced Radon transform using generalized interpolated Fourier method for straight line detection," *Computer Vision and Image Understanding*, vol. 115, pp. 152–160, February 2011.

[11] W. Pan, K. Qin, and Y. Chen, "An adaptable-multilayer fractional Fourier transform approach for image registration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, pp. 400 –414, March 2009.

[12] M. Atiquzzaman and M. Akhtar, "A robust Hough transform technique for complete line segment description," *Real-Time Imaging*, vol. 1, no. 6, pp. 419 – 426, 1995.

[13] V. Kamat and S. Ganesan, "A robust Hough transform technique for description of multiple line segments in an image," in *Proceedings of IEEE International Conference on Image Processing*, vol. 1, pp. 216 –220, October 1998.

[14] C. G. Ho, R. C. D. Young, C. D. Bradfield, and C. R. Chatwin, "A fast Hough transform for the parametrisation of straight lines using fourier methods," *Real-Time Imaging*, vol. 6, no. 2, pp. 113 – 127, 2000.

Lane detection using Fourier-based line detector

Payam S. Rahmdel, Daming Shi, and Richard Comley
School of Science and Technology
Middlesex University London
London NW4 4BT, United Kingdom
Email: {p.rahmdel, d.shi, r.comley}@mdx.ac.uk

***Abstract*—In this paper a new approach to the lane marker detection problem is introduced as a significant improvement for a semi/fully autonomous driver assistance system. The method incorporates advanced line detection using a multilayer fractional Fourier transform (MLFRFT) and a state-of-the-art advanced lane detector (ALD). Experimental results have shown a considerable reduction in computational complexity over ALD.**

***Index Terms*—Lane detection, lane tracking, multilayer fractional Fourier transform, Hough transform, line detection.**

I. Introduction

A. Background

It has always been a goal for engineers to get to the point of accident free driving. Developing a full autonomous navigation system to assist the driver by alerting him/her in hazardous situations or even take part of the driving task has been an active field of car accident research for the past two decades [1], [2]. Driver assistance systems help drivers to better prepare in unpredictable instances and react instantly when necessary. More and more semi-autonomous features are getting added together to push the boundaries towards full autonomous driver assistance systems. Various sensing modalities are adapted to these systems. Monocular and stereo cameras, light detection and ranging (LIDAR) [3], inertial measurement unit (IMU) and global positioning system (GPS) are the most dominant modalities.

For a human driver, road boundaries, lane markings, road colour and texture are key navigational aids in any weather condition. Thus, one would expect an autonomous system to benefit from the same guidelines and follow the same principles as humans. That is why vision-based imaging has been always a prominent sensing modalities for such systems. All of the existing system perception problems can be categorised into two groups: namely, lane/road detection and obstacle detection [1]. To incorporate line segment detection method, presented in this research, only the former task will be concerned.

B. Lane marker detection

Lane detection and perception prevents unintentional departure from the lanes by the simultaneous detection and tracking of the position and orientation of lane markers and the sending of warning messages to the driver. Therefore, precise line detection and segmentation plays a vital role in the overall performance of the system. Without a doubt, the Hough transform (HT) is a reliable candidate to fulfil this task. Several works have addressed the problem using Hough transform (HT) based methods [4], [5], [6], [7], [8], [9]. Among them, a recently proposed *Advanced Lane Detector* (ALD) [9] showed a superior performance when dealing with real-time video sequences. The proposed framework is robust and accurate, however, for its main job, i.e., line extraction, it relies on the standard Hough transform (SHT). Meaning that edge detection (binarization) is a significant prerequisite step for the procedure.

Problems of edge detection: 1) Edge detection requires a significant amount of computational time that is a major concern in a real-time application. 2) It may increases the false detection; when the noise-level is high, some of the true feature points in image can be ignored as noise and also some false points may be seen as true feature points.

Therefore, HT based on multilayer fractional Fourier transform (HT-MLFRFT) is adopted in this research [10] without the need of edge detection. The principal aim of this paper is to incorporate a HT-MLFRFT with the ALD to avoid the problems of edge detection whilst guaranteeing the detection accuracy.

The remainder of this paper is organized as follows: Section II gives an overview of lane detection framework. In Section III advance multilayer Fourier-based HT is introduced to improve the computational time. Section IV shows the experimental results followed by the conclusion in Section V.

II. Conventional lane detection framework

In the following, a brief description of each step is presented, (also shown in Fig. 1). A formal statement of the lane marker detection process is given as follows, (For more details see [9]):

Step 1) Pre-processing. It consists of three major tasks.

1) *Region of interest selection*. This area is the portion of image that has the higher probability of encompassing the lane markers. Region of interest is usually selected between the vanishing point and the vehicle's hood.
2) *Temporal blurring*. This is to assure that the HT can detect the short dashed lane markers when appear. Temporal blurring connects the broken lane markers by producing an average image of current frame and a number of predecessor frames.
3) *Bird's eye view*. Inverse perspective mapping (IPM) transforms an image from camera perspective view to bird's eye view. In camera view, the width of lane

markers changes depending on its distance to the camera. Also, instead of non-parallel lines, bird's eye view provides parallel line separated in specific distance.

Step 2) Hough transform.

1) *Edge detection*. Because the HT is only applicable on binary images, edge detection has to be performed in advance to obtain a binary image.
2) *Standard Hough transform*. The image is split into halves and HT will be applied to each half separately.
3) *Peak detection*. The HT peak detection determines the potential candidate for the lane marker.

Step 3) Post-processing. It consists of two major tasks.

1) *Sampling the detected lines*. Each candidate line has to be sampled by a predefined resolution.
2) *1D template matching*. At each sample point, 1D template matching will be applied to estimate the best matching pixel.

Step 4) Locate and track the lane markers.

1) *Least square estimation*. To minimize the error and locate the best matching straight line, a linear least square estimation (LSE) is applied.
2) *Kalman filter*. Kalman filter helps to predict the line estimation when lane marker disappears in some frames.

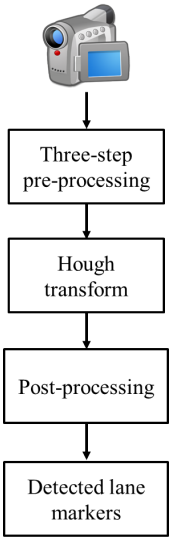


Fig. 1. Block digram of lane marker detection

Despite the accuracy of the presented framework its main step, i.e., Hough transform, is highly dependent on accurate edge detection. As explained in Section I, failing to operate this step in an appropriate manner will degrade the overall detection results.

However, a Hough transform based on multilayer fractional Fourier transform (HT-MLFRFT) does not require prior edge detection [10]. HT-MLFRFT showed a superior robustness in dealing with noisy images and can be applied directly to a greyscale image. This can decrease the computational complexity and improve the real-time performance of the system.

III. Hough Transform using Multilayer Fractional Fourier Transform

A. Forward Radon Transform

Equivalently and due to the mathematical identity, the Hough transform can be obtained via the Radon transform [11] as it perform line integrals along the angles of θ using equation

$$\begin{aligned} Rf(x,y) &= \lambda(\rho,\theta) \\ &= \int\int f(x,y)\delta(x\cos\theta + y\sin\theta - \rho)\,\mathrm{d}x\mathrm{d}y \end{aligned} \quad (1)$$

where $Rf(x,y)$ is a 2-D forward Radon transform of 2-D function f. This technique is based on the unique property of *Fourier slice theorem* in frequency domain for parameterisation of straight line [12]. Meaning that, the result of a 2-D discrete Fourier transform (DFT) of a given function (image) $f(x,y)$ is equivalent to a 1-D inverse Fourier transform of the Radon transform of the same function.

$$f(x,y) \xrightarrow{2D-DFT} F(X,Y) \rightarrow \text{Fourier slice theorem...}$$
$$\ldots \xrightarrow{1D-IDFT} Rf(x,y)$$

Therefore, instead of calculating line integrals along the θ in the spatial domain it is feasible to 1) transfer the image to its corresponding frequency domain by performing 2-D DFT; 2) interpolate the Cartesian coordinates to polar; 3) apply the 1-D DFT along each rows, i.e. in ρ direction, of the corresponding 2-D array. However the result of the DFT or so called one-layer DFT arises the problem of aliasing and therefore increases the problem of false detection.

B. Multilayer approach to Fourier transform

Based on the theory of multi-layer fractional Fourier transform introduced by Pan et al. [13] for image registration, Shi et al. proposed an advanced technique for the Hough transform based straight line detection [10], [14]. In their method the spectrum of an image consists of different layers of the Fourier transform. The MLFRFT of $f(x,y)$ is defined as

$$\begin{aligned} F^{\alpha}(X,Y) = &\sum_{x=-\frac{N}{2}}^{\frac{N}{2}-1}\sum_{y=-\frac{N}{2}}^{\frac{N}{2}-1} f(x,y) \\ &\times \exp(-j2\pi(x\alpha X + y\alpha Y)/N) \end{aligned} \quad (2)$$

where $\{f(x,y)| -\frac{N}{2} \leqslant x,y \leqslant \frac{N}{2}-1\}$ and $0 < \alpha \leqslant 1$ is a fractional scaler for each layer of transform. The resulting frequency response of the image has more frequency samples compared to the traditional one-layer 2-D DFT. Therefore, one can achieve more information of pixel intensity variation to increase both the performance and the accuracy of the HT. The process of HT using MLFRFT is depicted in Fig. 2.

Due to the natural behaviour of MLFRFT its frequency domain samples are closer to Polar coordinates in comparison to the 2-D DFT frequency samples. Thus, the interpolation error is reduced during the Cartesian to Polar mapping procedure.

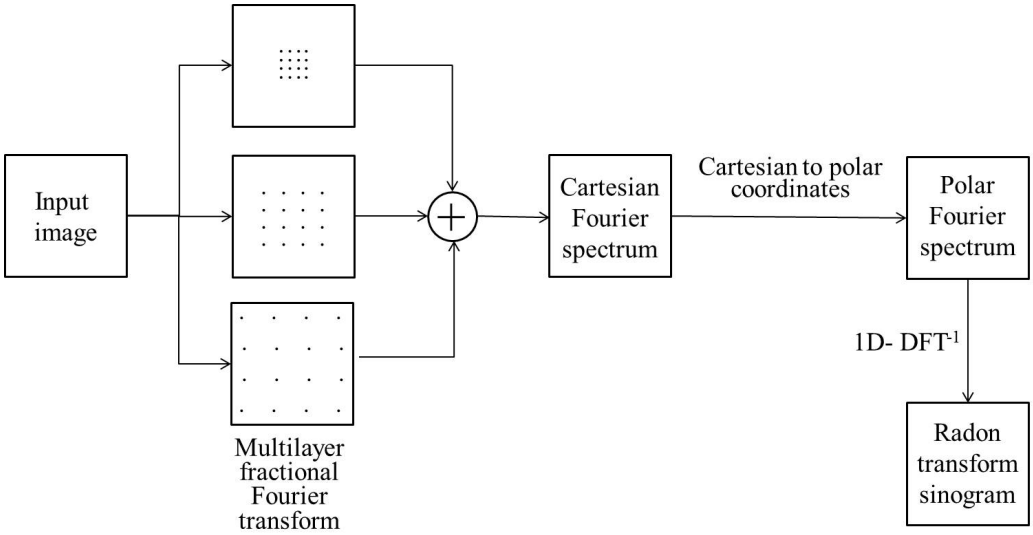


Fig. 2. Block diagram of lane detection using multi-layer Fourier-based Hough transform.

The method presented in [14] i.e. generalized interpolated Fourier transform (GIFT) is even more precise. As shown in Eq. 3, for each layer of Fourier transform different scale factor is used in x and y direction. GIFT is computed by

$$F^{\alpha,\beta}(X,Y) = \sum_{x=-\frac{N}{2}}^{\frac{N}{2}-1} \sum_{y=-\frac{N}{2}}^{\frac{N}{2}-1} f(x,y) \times \exp(-j2\pi(x\alpha X + y\beta Y)/N) \quad (3)$$

where $0 < \alpha, \beta \leqslant 1$ are predefined scalers for x and y dimension of the image. For more details please refer to [10] and [14].

As a result, HT-MLFRFT has been replaced in our research to address the problem of edge detection. Thus, the block digram depicted in Fig. 2 has been replaced with the Step 2, i.e., Hough transform, explained in Section II.

IV. Experimental results

To examine the efficiency of the proposed methodology and to make a reliable comparison, experiments were performed using the publicly accessible database provided by the authors in [9]. Diverse driving scenes and road conditions were chosen with distinct illumination levels. Fig. 3 illustrates a number of detection results obtain via the proposed methodology.

The computational time on a Dell machine, Intel Core i5, 2.4 GHz, were 0.12 seconds in average for each frame. Therefore, approximately eight frames per second can be processed. However, in some driving scenes, especially in daytime where the level of illumination varies between the surrounding objects, the processing time can go up to 0.2 seconds, at the highest. As a result, five frames per second can be guaranteed which will provide a sufficient accuracy in terms of video processing. If edge detection were included this would at least double the amount of computational time. Table I shows a comparison between the proposed lane detection method and the ALD.

TABLE I
Computational time comparison between the proposed lane detection and the ALD.

Proposed method	ALD
0.12 sec/frame	0.80 sec/frame

V. Discussion and Conclusion

We introduced a new approach to the lane detection problem based on our previously proposed Fourier-based line detection approach. The proposed approach shows a superior performance due to the following reasons; 1) being independent of prior edge detection; 2) the use of Fourier-based HT to detect accurately the location and the orientation of the potential lines. Because the standard HT is only applicable for binary images, an adaptive threshold algorithm has been used for ALD to convert the grayscale image to black and white and highlight the prominent edges. In addition to the extra computational time that the edge detection requires, some of the significant segment information may also get eliminated during the edge detection and threshold process. However, our proposed method is based on a multi-layer Fourier transform which can accept a grayscale image as its input. Therefore, prior edge detection is not required. In addition to its time efficiency, the high-resolution frequency response will guarantees a sufficient accuracy in detecting all the significant image features as well as line profiles.

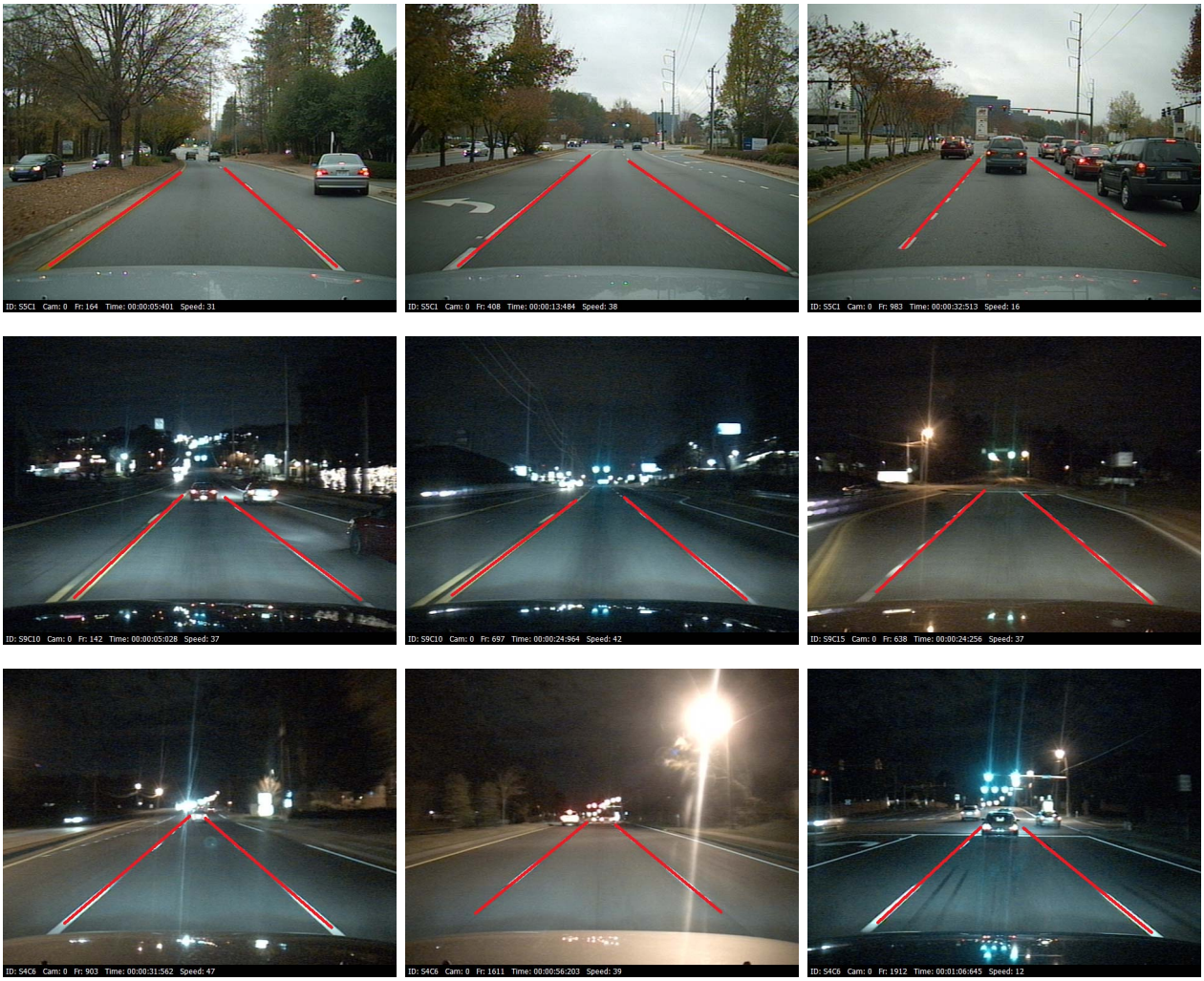


Fig. 3. Lane detection using HT-MLFRFT in diverse driving scenes and road conditions

REFERENCES

[1] A. Bar Hillel, R. Lerner, D. Levi, and G. Raz, "Recent progress in road and lane detection: a survey," *Machine Vision and Applications*, pp. 1–19, 2012.

[2] J. McCall and M. Trivedi, "Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, pp. 20–37, March 2006.

[3] J. Han, D. Kim, M. Lee, and M. Sunwoo, "Enhanced road boundary and obstacle detection using a downward-looking lidar sensor," *IEEE Transactions on Vehicular Technology*, vol. 61, pp. 971–985, March 2012.

[4] A. Borkar, M. Hayes, M. Smith, and S. Pankanti, "A layered approach to robust lane detection at night," in *IEEE Workshop on Computational Intelligence in Vehicles and Vehicular Systems. CIVVS '09*, pp. 51–57, April 2009.

[5] G. Zhang, N. Zheng, C. Cui, Y. Yan, and Z. Yuan, "An efficient road detection method in noisy urban environment," in *IEEE Intelligent Vehicles Symposium*, pp. 556–561, June 2009.

[6] Y. Jiang, F. Gao, and G. Xu, "Computer vision-based multiple-lane detection on straight road and in a curve," in *International Conference on Image Analysis and Signal Processing (IASP)*, pp. 114–117, April 2010.

[7] X. Shi, B. Kong, and F. Zheng, "A new lane detection method based on feature pattern," in *2nd International Congress on Image and Signal Processing. CISP '09*, pp. 1–5, October 2009.

[8] A. Borkar, M. Hayes, and M. Smith, "Robust lane detection and tracking with ransac and kalman filter," in *16th IEEE International Conference on Image Processing (ICIP)*, pp. 3261–3264, November 2009.

[9] A. Borkar, M. Hayes, and M. Smith, "A novel lane detection system with efficient ground truth generation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, pp. 365–374, March 2012.

[10] D. Shi, L. Zheng, and J. Liu, "Advanced Hough transform using a multilayer fractional Fourier method," *IEEE Transactions on Image Processing*, vol. 19, pp. 1558 –1566, June 2010.

[11] S. R. Deans, "Hough transform from the Radon transform," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 3, pp. 185–188, March 1981.

[12] C. G. Ho, R. C. D. Young, C. D. Bradfield, and C. R. Chatwin, "A fast Hough transform for the parametrisation of straight lines using fourier methods," *Real-Time Imaging*, vol. 6, no. 2, pp. 113–127, 2000.

[13] W. Pan, K. Qin, and Y. Chen, "An adaptable-multilayer fractional Fourier transform approach for image registration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, pp. 400–414, March 2009.

[14] L. Zheng and D. Shi, "Advanced Radon transform using generalized interpolated Fourier method for straight line detection," *Computer Vision and Image Understanding*, vol. 115, pp. 152–160, February 2011.