KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY, KUMASI

CONSTRUCTION OF A NOVEL CONVOLUTION BASED FRACTIONAL DERIVATIVE MASK FOR IMAGE EDGE ANALYSIS

A DISSERTATION SUBMITTED TO THE DEPARTMENT OF MATHEMATICS, KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE
OF
DOCTOR OF PHILOSOPHY
(APPLIED MATHEMATICS)

COLLEGE OF SCIENCE

BY
APPATI JUSTICE KWAME
AUGUST, 2016

Declaration

I hereby declare that, this thesis is the result of my own original research and that no part of it has been submitted to any institution or organization anywhere for the award of a degree. All inclusion for the work of others has been dully acknowledged.

Justice Kwame, APPATI	.....	.....
PG1101213	Signature	Date

Certified By:

Dr. P. AMOAKO-YIRENKYI	.....	.....
Supervisor	Signature	Date

Prof. I. K. DONTWI	.....	.....
Supervisor	Signature	Date

Prof. S. K. AMPONSAH	.....	.....
Head of Department	Signature	Date

Abstract

This thesis presents a new approach in constructing a more efficient fractional derivative mask for image edge analysis based on the definition and properties of convolution. By the definition of convolution, the generalised Strivastiva-Owa's operator was rewritten with its order restricted to the Riemann-Liouville fractional derivative. Applying linearity, commutative and derivative properties of convolution to the resultant expression, a new mask with higher efficiency, memory effect and computational equivalence to the classical edge detector is developed as per the experimental results obtained. From the experimental results, it is observed that, the new mask has the potency to find edges in details quite significantly as well as hidden edges which is a deficiency of the classical edge detectors. It can also be used on a region growing algorithm during region segmentation acting as an edge function in its termination process. The experiments conducted on the mask were done using some selected well known synthetic and medical images with realistic geometry. Using visual perception and performing both mean square error and peak signal-to-noise ratios analysis, the method demonstrated significant advantages over other known methods.

Contents

List of Tables

List of Figures

Dedication

This thesis is dedicated to my father, Mr. Ebenezer Kwadwo Appati, mother, Mrs. Esther Yaa Mawutor Appati, my siblings, Adelaide Abena Appati, Walter Kwasi Appati, Benedict Kwasi Appati and Emmanuel Atando, my loved one Sandra Yvonne Abena Sikakor Aglomasa, Pastor Emmanuel Ntumi-Darko and to the Department of Mathematics, KNUST.

Acknowledgement

First and foremost, I give thanks to the Almighty God for giving me the strength and knowledge to go through this studies. My heartfelt gratitude goes to my supervisors, Dr. Peter AMOAKO-YIRENKYI and Prof. Isaac Kwame DONTWI for their patience, guidance and keen interest in my work which aided in bringing my research work to a successful completion. I am also grateful to Mr. Kwaku Foukour DARKWAH for his mentorship and endless assistance in times of hardship and rocky stages in my thesis setup. Again, I extend my gratitude to Prof. S. K. AMPONSAH, the current Head of Department of Mathematics and Actuarial Science for his support and advice towards the completion of this project. My special thanks go to the family of Aglomasa and Ntumi-Darko for their prayers and diverse support in times of need. To my father, Mr. Ebenezer K. APPATI, mother, Mrs. Esther Y. M. APPATI, siblings, Adelaide A. APPATI, Walter K. APPATI, Benedict K. APPATI and Emmanuel ATANDO, I say God richly bless you. Finally, I say thank you to the following institutions, Advance Information Technology Institute-Kofi Annan Centre of Excelence (Ghana), CDAC (India), Johanneum Research (Austria), Universitas Thomae Bata Zlinensis (Czech Republic) and National Institute for Mathematical Sciences (Ghana) for their various inputs.

List of Abbreviations

Abbreviations

FDE	Fractional Differential Equation
LIVE	Laboratory for Image & Video Engineering
MRI	Magnetic Resonance Imaging
FIR	Finite Impulse Response
IIR	Infinite Impulse Response
PACS	Picture Archiving Communication System
DICOM	Digital Imaging and Communications in Medicine
PSNR	Peak Signal to Noise Ratio
MSE	Mean Square Error
SSIM	Structural Similarity Index Measure
JPG, JPEG	Joint Photographic Expert Group
ROI	Region Of Interest
SNR	Signal to Noise Ration
POI	Pixel Of Interest

Chapter 1

INTRODUCTION

1.1 Background

Image edge analysis is an entitlement for a set of mathematical methods which aim at identifying points in a digital image at which the image brightness changes sharply (i.e. point of discontinuities). The organisation of these points into a set of curved line segments then becomes the edge. With no doubt, detecting these points forming the edge map is one of the most common and fundamental operations in image processing and analysis. This is so because, edges are consistent with the human perception and serves as the first step in image understanding and interpretation (Guo and Lai, 2013). They provide useful structural information which can be used for region segmentation, feature extraction and object identification. These information, by common practice, are extracted by developing a convolution mask mostly known as the gradient operator. These operators are relatively smaller two dimensional array which is used to modify each pixel value of the original image. This modification is performed according to the value of the neighbourhood around the pixel/point of interest (POI) (Jalab and Ibrahim, 2013). Aside this approach, there are other forms in which these edges can be extracted such as the segmentation and transform based operators. Definition of a particular operator in any of these three categories have their own pros and cons. For instance, the gradient operators modelled with the classical (i.e. Partial Derivative Based) approach are known to be

robust to images with low-level noise while they tend to erroneously recognize fake edges in the presence of excessive noise/artifacts. In the work of Guo and Lai (2013) a transform based edge detector was presented based on box spline wavelet frames. This approach was documented to be simple but robust to noise and compelling in getting the subtle elements of the image, nonetheless, from the computational time perspective, they are slower than the classical gradient operators.

Finally but more recently, other operators which are also gradient operators have emerged. Unlike the classical gradient operator, these class of operators are formulated using the fractional derivative approach. These operators, aside been simple, helps to resolve the disadvantage of the classical edge operators and are robust with higher signal-noise-ratio (Yang et al., 2011; Oustaloup et al., 1991; Gao et al., 2014). Although these operators are robust to noise, there are some fundamental issues with their formulation due to the use of binomial series expansion as a requirement. This series by nature are known to be infinite and one needs to select an appropriate N in other to obtain a gradient mask whose values are approximate values of the series. In effect, some useful data with regards to the edge points extracted are lost at a certain margin of error.

1.2 Motivation

For some few decades now, image edge analysis has been at the heart of imaging science. This has been as a result of the prominent characteristics posed by images making it possible to understand and interpret their content. Many studies have been done in this area but the problem of extracting true edges still remain a challenge to be looked at. Existing methods referred to as the classical approaches performances poorly in the presence of noise. Enhancement has been made to curb the situation

but unfortunately they also tend to be computational expensive. This has created an open window for research.

1.3 Problem Statement

During image edge point extraction, it is expected that only significant pixel values will be preserved and when connected will form the edge map. By convention, derivatives of some functions are required to approximate the supposed edges for extraction by convolution in most cases. Nonetheless, the extraction of these points on noisy images tends to extract noisy pixels as edge points leading to the creation of phantoms. These phantoms end up making analysis on the resultant edge map quite difficult and misleading for a concise conclusion.

In an attempt to resolve some of these issues other operators (Yang et al., 2011; Oustaloup et al., 1991; Garg and Singh, 2012; Gao et al., 2014) mostly based on fractional calculus with improved characteristics over the classical methods like Canny and Prewitt have been proposed.

In particular the fractional based operators have been used in image quality enhancement, image texture enhancement (Pu et al., 2008), image denoising (Jalab and Ibrahim, 2012) and image edge analysis

In Zhang et al. (2010), "Construction of Fractional Differential Masks Based on Riemann-Liouville Definition" was proposed. In their work a variant of the RiemannLiouville definition that unifies to the Gru¨nwald-Letnikov definition was used. Without loss of generality, the interval r0,*x*s was divided into *N* equal shares while the integral part was then rewritten in terms of summation. By using the difference equation, the final expression is simplified to obtain the mask.

The work of Yang et al. (2011) in "The Construction of Fractional Differential Gradient Operator" made use of the fractional differential finite impulse filter transfer function. In rewriting that expression, a binomial series expansion representation was obtained. Expanding the summation term to the number of term equivalent to the mask size, the fractional gradient was formulated.

On the side of Hamid and Rabha (2013), a Riemann-Liouville definition was employed in the work "Texture Enhancement for Medical Images Based on Fractional Differential Masks". From this definition, the integral component was written in terms of summation and a function $f(z)$ was created and differentiated $m \grave{} 1$ times with respect to z. Expanding the resultant expression and setting n to be an integer, a mask of size n was created.

In Gao et al. (2014) on the topic "Edge Detection Based on the Newton Interpolation's Fractional Differentiation" made use of the generalised Gru¨nwald-Letnikov definition. The summation term was expanded to a number of terms equivalent to the size of the mask. Here, it was believe that, these discrete points are not precise enough and needs to be improved using Newton Interpolation equation.

Although the Riemann-Liouville definition theoretically provides an exact value for the purpose of calculus, it is practically difficult when used to evaluate an integral or a derivative. Theoretically, Riemann-Liouville definition is equivalent to the Gru¨nwald-Letnikov definition but one question always arises as to what number of terms should be computed and summed for the Gru¨nwald-Letnikov definition of fractional derivative to be as accurate as that of Riemann-Liouville definition. In an attempt to answer this question Loverro (2004) used up to 171 terms to obtain an error of $1e\acute{} 3\%$. This in a way imply that a mask size of $171 \hat{} 171$ is required for that accuracy to be achieved. Unfortunately, the bigger the mask size the more

computationally expensive it becomes and hence this theoretical equivalence is mostly not achievable in practice.

Interestingly, the masks are extracted without evaluating the entire derivative and therefore when carefully constructed, it can produce a more desired masks results. Unfortunately this idea has not been considered in the quest to find a mask that has memory, computationally efficient to compute and robust to noise.

1.4 Objective

The objective of the thesis are:

1. to review the existing classical edge detectors and fractional edge detector.
2. to develop and formulate a fast and accurate fractional based edge detectors.
3. to conduct an experiment to select appropriate image smoothing techniques for the new mask.
4. to develop a segmentation algorithm based on the mask.
5. to test the performance of the newly formulated fractional edge mask.

1.5 Outline of the Methodology

The following outlines the method employed in achieving the objective of the thesis:

1. Formulation and development of the edge mask using the concept of fractional derivative and convolution
2. Experimental analysis to select an optimal image smoothing functions to be used with the formulated mask

3. Development of an object identification algorithm by segmentation using the new mask and region growing

Performance analysis of the develop mask on an image data using the standard image metric

1.6 Justification of the Study

The design of computer assisted signal and image analysis infrastructure based on the implementation of this proposed mask has many applications in industry.

Particularly, in many medical imaging applications, techniques and processes for creating visual representations of both the interior and exterior features of a living object for clinical analysis and medical intervention can not be over emphasized. Here the focus is to reveal internal structures hidden by the skin and bones, as well as to diagnose and treat disease. With the advent of this proposed mask, segmentation and other feature extractions characterised by edges could be achieved with ease leading to the provision of volumetric quantification.

Again in aerodynamics agencies, aeronautic engineers are required to understand how air moves around objects during aircraft surface manufacturing so as to reduce turbulence. This is mostly achieved by ensuring that the surface of the body of the aircraft is C^2 smooth to avoid generating singularities of airflows. However, the assurance of finding defects in the surface of an aircraft body is to find locations with discontinuous first or second order derivative refer to as hidden edges. Experimentally, this proposed model shows its capability in exposing such edges.

1.7 Organization of Chapters

This thesis is divided into five main chapters. The first chapter is an introduction of the project work. This takes into account the problems in the field of image edge

4.

analysis and a gist of methods appropriate to be adopted in solving the problems at hand including the chapter organisation and some contributions.

The second chapter gives the literature review of the techniques and approaches that describes image analysis and fractional calculus as far as image edge analysis is concerned. The chapter further reviewed some mathematical concepts used through out the study.

In Chapter 3, that is, methodology, explains the theory and concepts which form the basis of the thesis. This deals with how digital filters, image algebra, image transform and some discussion on mathematical morphology in image processing.

Last by not the least, is Chapter 4, were explanation of how the new fractional mask was formulated and used to achieve higher accuracy in image edge analysis was discussed. It also gives a description of the analysis as well as the experimental results obtained from the test database considered in this thesis.

Finally, Chapter 5 concludes the work with some recommendations made on the relevance of the undertaken project.

1.8 Contribution

Classical image edge mask are computationally efficient but performs poorly on noisy data. On the contrary, transform based image edge mask are robust to noise but computationally expensive. In this thesis:

1. a new fractional derivative based convolution mask for image edge analysis has been developed with equivalent computational efficiency as the classical image edge mask but with an inherent property of memory.

2. the new proposed mask has been equipped with smoothing technique inherit the property of robustness to noise.

3. a new segmentation algorithm was built using the new mask as a hybridisation with region growing techniques.

   finally, the proposed mask is able to detect hidden edges which is a deficiency of classical image edge mask

4.

Chapter 2

LITERATURE REVIEW

2.1 Introduction

Fractional calculus, although not new, have shown to be significant in the fields of mathematics, engineering and physics. This development generalises the *n*-fold integer-order integration and differentiation by considering any arbitrary real or complex order. In effect, fractional derivative has made mathematical modelling interesting in describing the mechanical and physical properties of real materials (Hilfer, 2000; Kanbur et al., 2010; Baleanu et al., 2012) due to its special features.

In a few decades, more useful operators of fractional calculus as cited in Jalab and Ibrahim (2012) have evolved from the Riemann-Liouville fractional calculus to handle specific situations. These operators have shown usefulness in image processing and analysis especially in

1. image quality enhancement (Marazzato and Sparavigna, 2009)
2. image texture enhancement (Pu et al., 2008)
3. image denoising (Jalab and Ibrahim, 2012)
   gradient edge detection (Zhang et al., 2010)
5. image restoration (Xuehui et al., 2015)
6. image impainting (Zhang et al., 2012)

Almost every image have a certain amount of information on edge detection useful to the user. The useful information are considered to characterise edges as they define boundaries between an object and its background. Since these edges characterise boundaries of object in an image, they are of importance to image processing. Detecting accurate edges provide the higher probability of analysing basic and silent features associated with objects in an image. Observation by many has indicated that, the common practice used to extract these accurate edges is to develop a gradient convolution mask. Such mask are simple in nature and computationally efficient for a real time implementation. These masks are an array of size $n \hat{} n$, where n is an odd number. This mask modifies the original image to a new image according to the value of the neighbourhoods centred around the Pixel of Interest (POI). The centre of the $n\hat{}n$ mask is what is referred to as the POI. This technique has become increasingly important in the application of modern algorithm in image processing and analysis (Jalab and Ibrahim, 2013).

As a fundamental concept of image processing and analysis, there is a mapping between light frequencies of the image and the gray color values. Changes in the gray values with respect to relative distance of an image determine the type of edges. High frequency transition correspond to large changes in values (relative distances/position) on an image. High frequency transition correspond to noise or edge. In contrast, low frequency transition correspond to small changes in gray color values where background and textures can be found (McAndrew, 2004). Various approaches has been proposed in the construction of edge detection by using classical and fractional derivatives (Oustaloup et al., 1991; Yang et al., 2011; Gao et al., 2014).

4.

In this work, the existing mask (classical edge detectors) and the fractional derivative will be reviewed by considering the use of special functions in developing the convolution mask.

2.2 Development of Fractional Calculus

Fractional calculus has been in existence since 1695 in the regime of pure mathematics. As of that time, only little application could be realised in the field of applied mathematics. Notation governing these operators made most people suspect it as a calculus of fraction but instead it is the integration and differentiation to an arbitrary order. Since it birth till 1822, extensive contributions has evolved as detailed in the next section.

2.2.1 The Beginning of Fractional Calculus

Fractional calculus can be traced back to the Leibniz notation $d^n y\{dx^n$ for the n^{th} ordered derivative of a function $ypxq$, where n is strictly an integer. Leibniz as cited in (Pertz and Gerhardt, 1849) gave response to L'Hospital question about the meaning of the $\frac{1}{2}$ order derivatives in the letter quoted below:

> "You can see here Sir that, one can express a term like $d^{\frac{1}{2}}\overline{xy}$ or $d^{1:2}\overline{xy}$ by an infinite series, even though it seems to be far from the geometry, which usually only considers the differences of positive integer exponents or the negatives with respect to sums, but not yet those, whose exponents are fractional. It is true that it is still to show that it is this series for $d^{1:2}x$; but not only this can be explained in a way. Because the ordinates x are expressed in a geometric series, such that by choosing a constant $d\beta$ it follows that dx " $xd\beta$: a, or (if one chose a as unit) dx " $xd\beta$, meaning

ddx would be $\overline{x^{\cdot\cdot}}d\beta_2$, and d^3x would be $\overline{x^{\cdot\cdot}}d\beta^3$ etc. and d^ex " $\overline{x^{\cdot\cdot}}d\beta^e$. And

thus the differential exponent has been changed by the exponents and by replacing $d\beta$ with $dx : x$, yielding $d^e x \text{ “ } dx : x^{e\ddot{}}x$. Thus it follows that $d^{\frac{1}{2}}$ will be equal to $x \, ? \, dx : x$. It seems like one day very useful consequences will be drawn from this paradox, since there are little paradoxes without usefulness.”

Bernoulli and Leibniz in December 1695 as cited in (Pertz and Gerhardt, 1850a) rekindled the topic of non-integer order derivative as addressed in this letter:

> “From the analogy you have shown that exponentiating and differentiating can easily be transferred to series for $d^m xy$. If for the time being m denotes a fractional or irrational number, can you please explain to me what $d^m xy$ would be, a quantity or something else?”

A detailed explanation to the question raised by Bernoulli was addressed in the letter of Leibniz as cited in (Pertz and Gerhardt, 1850b) in the quote below:

> “What you are looking for in the derivatives, whose exponent is fractional or irrational, have I already considered in a letter to L’Hospital, and even though I have added this, the so structured derivatives can be compared with the ordinary ones. Let for example $d^{1:2}x$ be the proposed derivative. Let further, x be a geometric progression. Let the assumed derivative constant be dh, and we have $xdh : a \text{ “ } dx$, and we get $d^2x \text{ “ } dxdh : a \text{ “ } xdhdh : aa$ and similarly $d^3x \text{ “ } xdh^3 : a^3$, and more general $d^e x \text{ “ } xdh^e : a^e$, as well as $d^{1:2}x \text{ “ } xdh^{1:2} : a^{1:2}$, or $d^{1:2}x \text{ “ } x\,dh : a$ (here 1 : 2 is the same as $\frac{1}{2}$; and $dh : a$ the same as $\frac{dh}{a}$). Thus you can see that such structured derivatives only have meaning by extracting the roots or by exponentiating of ordinary derivatives. I think this is memorable and

you will not be unthankful for it. You have seen that the extra ordinary derivatives can be expressed by composing infinite geometric series and I suspect, that this will hold true for those cases which are not real."

After the death of Leibniz, the problem of non-integer derivatives was not persued until the work of Euler. His work resulted in the birth of Gamma function as a generalisation of factorial of integer numbers (Euler, 1738). Later, Lagrange contributed to the field of fractional calculus by developing the law of exponents for differential operators of integer order (Lagrange, 1849):

$$\frac{d^m}{dx^m}\frac{d^n}{dx^n} = \frac{d^{m+n}}{dx^{m+n}}, \quad m, n \in \mathbb{N} \tag{2.2.1}$$

Lacroix used the generalised factorial given by Gamma function to obtain the fractional derivative of the function $y(x) = x^m$, $m \in \mathbb{N}$ starting with the n^{th} integer order as given below (Lacroix, 1819):

$$\frac{d}{dx^n}y(x) = \frac{d^n}{dx^n}x^m = \frac{m!}{\Gamma(m-n+1)}x^{m-n}, \quad m \geq n \tag{2.2.2}$$

For a special case where $y(x) = x$ and $n = \frac{1}{2}$, Lacroix presented the following:

$$\frac{d^{\frac{1}{2}}}{dx^{\frac{1}{2}}}x = \frac{\Gamma(2)}{\Gamma(\frac{3}{2})}x^{\frac{1}{2}} = \frac{2\sqrt{x}}{\sqrt{\pi}} \tag{2.2.3}$$

This result is similarly obtained by using Riemann-Liouville fractional derivative.

Another more general definition of fractional operations was presented by Fourier as follows (Fourier, 1822):

$$\frac{d^i}{dx^i} f\ x) = \frac{1}{2\pi} \dot{z} \quad \_\_\_ \quad \dot{z} \quad i \quad \hat{} \quad \_\_\dot{} \ `8`8$$

$$i\pi$$

$$\text{p}f\text{p}\alpha\text{q}d\alpha p \cos\ p\text{p}x\ ´\ \alpha\text{q}$$

$$`\qquad dp \qquad (2.2.4)$$

$$2$$

$$´8´8$$

where the number *i* is regarded as positive or negative. This definition of fractional calculus is suitable for sufficiently "well behaved" functions which are not necessarily power or integral functions.

In the work of Liouville (1832a,b,c), two different definitions of fractional derivatives was developed . The first was applied to a function *f*p*x*q which can be presented as a series in the following form:

$$f\text{p}x\text{q} \text{ “ } \ddot{y} \ c_k e^{a_k x} \qquad (2.2.5)$$

$$8$$

$$k\text{“}0$$

Extending the known integer order derivatives $d^n\{dx^n e^{ax} \text{ “ } D^n e^{ax}$ to the fractional case by replacing *n* P N with *α* P C to obtain:

$$D^\alpha f\text{p}x\text{q} \text{ “ } \ddot{y} c_k a_k^\alpha e^{a_k x} \qquad (2.2.6)$$

$$8$$

$$k\text{“}0$$

This definition as in Equation (2.2.6) is restricted to the choices of *α* it can take on to converge. The second definition does not have restriction on the choice of *α* but instead, a restriction on the type of function for which it is applicable. For functions of the type *f*p*x*q " 1{x^a, with an arbitrary parameter *a*, Liouville presented the definition:

$$D_\alpha x'_a \text{ “ } \text{p}´\_\_\_\_\_1\text{q}_\alpha \Gamma\text{p}a \ ` \ \alpha\text{q}x'_{a'\alpha} \qquad (2.2.7)$$

Γpαq

for its fractional derivative of order α.

In the work of Riemann in search of a generalisation of Taylor series, the following definition was deduced:

$$D^{-\alpha}f(x) = \frac{1}{\Gamma(\alpha)}\int_c^x (x-t)^{\alpha-1} f(t)\,dt + \psi(x) \qquad (2.2.8)$$

for a fractional integral of order α of a given function $f(x)$. This definition introduce the function $\psi(x)$ to cater for the ambiguity of the lower limit c of integration which corresponds to the complementary function. Setting $c = 0$ and without a complementary function $\psi(x)$ is the most common definition for fractional integration which in practice is referred to as *Riemann-Liouville fractional integral* (Cayley,

1880; Weber, 1876).

Sonin (1869) used the Cauchy's integral formula as a basis to define fractional calculus as was later extended in the work of Letnikov (1872). This integral formula for integer order derivative was given by:

$$f^{(n)}(x) = \frac{n!}{2\pi i}\int_C \frac{f(t)}{(t-x)^{n+1}}dt \qquad (2.2.9)$$

The generalisation to the fractional case was obtained by replacing the factorial with Gamma function $\alpha! = \Gamma(\alpha+1)$. However, this direct extension to non-integer values α resulted in problems with the integrand as cited in Ross (1977). Finally, it was the work of Laurent (1884) that resulted in today's definition of the *Riemann-Liouville fractional integral and derivative* respectively where $n = \lceil \alpha \rceil$.

$$ {}_cD_x^{-\alpha} f(x) = \frac{1}{\Gamma(\alpha)}\int_c^x (x-t)^{\alpha-1} f(t)\,dt, \qquad (2.2.10)$$

$$ {}_cD_x^{\alpha} f(x_{\mathrm{q}} \quad \frac{d}{dx^n} \quad \frac{1}{\Gamma(1-\alpha)} \quad {}_c \int^x (x-t)^{\alpha} f \quad \text{“} \quad \text{ˆ} \quad \dot{\text{z}} $$

$$ \mathrm{p}t\mathrm{q}dt^{\cdot} \quad (2.2.11) \quad d_n $$

To conclude with this section, Grunwald (1867) and Letnikov (1868) provided another definition for fractional derivative as below:

$$ {}^{GL}D_x^{\alpha} f(x_{\mathrm{q}} \text{“} \lim_{h \to 0} \frac{\quad}{h^{\alpha}} \quad (2.2.12) $$

In Equation (2.2.12), the finite difference $4^{\alpha}{}_h$ is defined as:

$$ \mathrm{p}4_h^{\alpha} f\mathrm{q}\mathrm{p}x\mathrm{q} \text{“} \ddot{\mathrm{y}}\mathrm{p}\acute{}1\mathrm{q}^k \text{ˆ} \quad \cdot f\mathrm{p}x \acute{} kh\mathrm{q}, \alpha \mathrm{ą} 0 \quad (2.2.13) $$

$$ \mathrm{p}4_{\alpha} f\mathrm{q}\mathrm{p}x\mathrm{q} \; k $$

and $\cdot$ is the generalised binomial coefficient. $\alpha \; k$

2.2.2 The Impact of Abel's Integral Equation on Fractional Calculus

Abel (1826, 1881) sought to solve the integral equation arising from the Tautochrone problem studied. This problem consist of the determination of a curve in the px,yq plane such that the time required for a particle to slide down the curve to its lowest point under uniform gravity is independent of its initial position px_0,y_0q on the curve.

Abel's initial solution to the above problem started as follow; from the physical law of nature, the potential energy lost during the descent of a particle is equal to the kinetic energy gained by the particle defined as:

$$ \frac{m}{2} \quad \overline{dt}\text{ˆ} \cdot \text{“} \; mg\mathrm{p}y_0 \acute{} y\mathrm{q} \; (2.2.14) \; 1 \; d\lambda^{\,2} $$

where m is the mass of the particle, λ is the distance of the particle from the starting point along the curve and g is the gravitational acceleration. Separation of the time and space variables yields:

$$? \, 0 \text{ “ } a2gdt \, y \acute{} \overline{y} \tag{2.2.15}$$

and integration from time t “ 0 to time t “ T gives

$$d\lambda$$

$$a \quad \overset{y0}{\dot{z}} \quad 2gT \text{ “p}y_0 \acute{} \, y\text{q}^{\acute{}1\{2}d\lambda \tag{2.2.16}$$ 0

Assuming that the time needed by the particle to reach the lowest point of the curve is a constant, then the left hand of Equation (2.2.16) has to be a constant say k. If we denote the path length λ as a function of height λ “ Fpyq, then it follows that $d\lambda\{dy$ “ F ^{1}pyq. By the change of variables y_0 Ñ x, y Ñ t and denoting $F^{\,1}$ “ f the tautochrone integral equation becomes

$$\overset{x}{\dot{z}} \quad k \text{ “p}x \acute{} \, t\text{q}^{\acute{}1\{2}f\text{p}t\text{q}dt, \tag{2.2.17}$$ 0

where f is the function to be determined. This expression could not be solved by Abel in its normal form until the use of Gamma formula leading to the concept of fractional calculus. By multiplying both sides of the integral equation with 1{Γp1{2q, Abel obtained on the right-hand side a fractional integral of order 1{2:

$$\frac{1}{\Gamma(1/2)}\int_0^x (x-t)^{-1/2} f(t)\,dt = \frac{1}{\Gamma(1/2)}\int_0^x f(x)\,(x-t)^{-1/2}\,dt \qquad (2.2.18)$$

Utilising the left-inverse of the fractional derivative, i.e.

$$k \qquad 1 \qquad x \qquad d^{1/2}$$

$$\frac{d^{1/2}}{dx^{1/2}}\frac{d^{-1/2}}{dx^{-1/2}} f(x) = \frac{d^0}{dx^0} f(x) = f(x) \qquad (2.2.19)$$

the solution of the tautochrone problem is then given by

$$f(x) = \frac{1}{\Gamma(1/2)}\frac{d^{1/2}}{dx^{1/2}} k = \frac{k}{\pi\sqrt{x}} \qquad (2.2.20)$$

where for the last equality, the derivative of order 1{2 of a constant k has been used.

In Samko et al. (1993), it was pointed out that, Abel did not only solve the integral equation (2.2.17) as a special case of the tautochrone problem but instead gave the solution to a more general integral equation.

$$k(x) = \int_0^x \frac{f(t)}{(x-t)^{\alpha}}\,dt, \; x > a, \; 0 < \alpha < 1 \qquad (2.2.21)$$

2.2.3 The Current State of Fractional Calculus in Differential Equation

At this stage of fractional calculus, Weyl (1917) considered the Fourier transform of periodic functions $\phi(x)$ given by:

88

$$\phi(x) = \sum_{k=-\infty}^{\infty} \varphi_k e^{ikx}, \;\; \varphi_k = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{ikx}\phi(x)\,dx \qquad (2.2.22)$$

in order to define fractional integration suitable for these functions, by

$$I_{\pm}^{\alpha}\varphi(x \quad 1 \qquad \infty \qquad \Psi_{\pm}^{\alpha}(x-t)\varphi(t)dt, \qquad \text{(2.2.23)}$$
$$\text{q “} \quad \text{—} \qquad \text{ż}$$
$$2\pi\,{}^{\prime}{}_{8}$$

with some special functions Ψ^{α}-p$x^{\prime}t$q. For 0 ă α ă 1, the fractional integral can be written as:

$$I^{\alpha}\phi\text{p}x\text{q “} \quad \frac{1}{\Gamma\text{p}\alpha\text{q}}\int_{-\infty}^{x}\text{p}x^{\prime}1\text{q}^{\alpha^{\prime}1}\phi\text{p}t\text{q}dt, \quad I^{\alpha}\phi\text{p}x\text{q “} \quad \frac{1}{\Gamma\text{p}\alpha\text{q}}\int_{x}^{\infty}\text{p}x^{\prime}1\text{q}^{\alpha^{\prime}1}\phi\text{p}t\text{q}dt, \qquad \text{(2.2.24)}$$

given that the integrals in Equation (2.2.24) are convergent over an infinite interval. Often, the Riemann-Liouville definition of a fractional integral given in Equation (2.2.10) with lower limit c “ ´8 is referred to as the *Weyl fractional integral*.

Marchaud (1927) presented an integral form of the Gru¨nwald-Letnikov fractional derivative as:

$$_{M}D^{\alpha}f\text{p}x\text{q “ }c\text{ż}_{0}^{\infty}\frac{(\triangle_{t}^{l}f)(x)}{t^{1+\alpha}}dt, \ \alpha > 0 \qquad \text{(2.2.25)}$$

known today as *Marchaud fractional derivative*. The term p$\triangle_{t}^{l}f)(x$q is a finite difference of order l ą α and c is a normalising constant.

In Watanabe (1931), the Leibniz's formula for Riemann-Liouville's fractional derivative was presented and given by:

$$D^{\alpha}\text{p}fg\text{q “ ÿ}\hat{}_{8}{}^{k`\beta}_{\alpha}\dot{}D^{\alpha^{\prime}\beta^{\prime}k}fD^{\beta^{\prime}k}g, \ \beta \text{ P } R \qquad \text{(2.2.26)}$$

k"'8

for analytic functions f and g.

Given a function f with an pn´1q absolute continuous derivative, Caputo (1967) defined a fractional derivative as:

$$D^{\alpha} = \frac{1}{\Gamma(n-\alpha)} \int_0^t (t-s)^{n-\alpha-1} \frac{d}{ds} f(s)\,ds \qquad (2.2.27)$$

This operator is mostly preferred as the initial conditions of the fractional differential equation can be specified in the classical form:

$$y^{k}(0) = b_k,\ k = 0,1,\ldots,n-1 \qquad (2.2.28)$$

which is in contrast with differential equations containing the Riemann-Liouville differential operator.

2.3 Other Related Studies

In this section, a general review of the various categories of edge detectors was presented.

2.3.1 Edge Detectors Based On Partial Derivative

A vital class of existing edge detectors depends on partial derivatives of the input image. Image pixels with most extreme angles or zero Laplacians are characterized as edges (Canny, 1986; Deriche, 1990; Marr and Hildreth, 1980; Koenderink and van Doom, 1992). These gradient based edge detectors normally incorporate three stages:

1. Firstly, noise is diminished if the input data is noisy, more often than not a Gaussian convolution is required for this stage.

2. Secondly, the partial derivatives are evaluated by convolving the image with some sort of kernels. Different kernels have been produced for this reason with various exactnesses along various directions (Zhai et al., 2008; Ziou and Tabbone, 1998). Prewitt and Canny edge detectors, case in point, contrast just in the sort of kernel used to estimated the partial derivatives.

3. Finally, edges are found where the norms of the gradient are greater than a predefined threshold. Standard thresholding technique treat pixels with gradient magnitude more prominent than the threshold as edges. Then again, hysteresis thresholding utilizes two unique thresholds. Any pixel with gradient magnitude greater than the maximum threshold is portrayed as an edge; so are those pixels that are in the neighborhood of this chosen pixel with gradient magnitude higher than the minimum threshold. This hysteresis thresholding procedure prompts associated edges and is usually alluded to as 'linking'.

Nonmaximum suppression is in some cases used to thin edges in a technique, for example, in the Canny's algorithm. Edge estimators of this sort are strong to lowlevel noise, however, they tend to erroneously identify fake edges in the vicinity of extreme noise/artifacts. To reduce this issue, one can further incorporate common neighborhood data (Guo and Huang, 2008).

2.3.2 Edge Detectors Based On Segmentation

Despite the fact that image segmentation is not quite the same as edge detection, they are connected, and one can extract edges from the segmented result. Image segmentation (Mumford and Shah, 1989; Ambrosio and Tortorelli, 1992; Chan and Vese, 2001) format the image space into various subregions such that every locale is homogeneous concerning a few qualities, for example, intensity. The borders of those

subregions structure the edges. To clarify in more scientific term in detail, the two-stage MumfordShah strategy is taken as a case. Given that g is the intensity function of an image data, the strategy centres on separating the image space into two locales Ω_1 and Ω_2. One locale is required to live inside the edge contour Γ and the other locale outside the contour, such that g can be approximated by C^1 functions f_1 , f_2 in Ω_1 , Ω_2, separately. This partition is accomplished by minimizing the functional:

$$E(f_1,f_2,\Gamma) = \int_{\Omega_1} |g - f_1|^2 \, dx + \int_{\Omega_2} |g - f_2|^2 \, dx + \nu \cdot \text{Length}(\Gamma) \tag{2.3.1}$$

with respect to functions f_1 , f_2 and the contour Γ. In a special case where f_1 , f_2 are constant with values c_1, c_2, respectively, the implementation of the MumfordShah model is simplified to the ChanVese model (Chan and Vese, 2001) based on the level set approach (Osher and Sethian, 1988). This represent the edge contour Γ by the zero level set of a Lipschitz function $\varphi : \Omega \to R$ and the regions inside and outside the contour Γ by the regions with positive and negative φ values, respectively. Let $H(\cdot)$ be the Heaviside function defined as $H(z) = 1$ for positive z and 0 elsewhere.

Then the minimizer φ of this expression; with respect to functions f_1 , f_2 and the contour Γ. In a unique situation where f_1 , f_2 are constant with qualities c_1, c_2, respectively, the usage of the MumfordShah model is streamlined to the ChanVese model (Chan and Vese, 2001) in light of the level set methodology (Osher and Sethian, 1988). This represent the edge contour Γ by the zero level set of a Lipschitz function $\varphi : \Omega \to R$ and the locale inside and outside the shape Γ by the regions with positive and negative φ values, respectively. Let $H(\cdot)$ be the Heaviside function

characterized as $H(z) = 1$ for positive z and 0 elsewhere. At that point, the minimizer φ of this expression is;

$$\frac{1}{2}\int_{\Omega}\left[H(\varphi)(g - c_1)^2 + (1 - H(\varphi))(g - c_2)^2\right] + \beta \int_{\Omega} |\nabla H(\varphi)| \qquad (2.3.2)$$

gives the segmentation of the domain and the edge contours where the edges are detected from the 0 level set of φ.

2.3.3 Fractional Based Edge Detectors

In image processing, edge detectors often make use of integer-order derivative operators where the order 1 is referred to as the gradient while the order 2 is the Laplacian. This has turned out to be more imperative in foundational research and designing application. From a hypothetical perspective, fractional differentive extend the order of signal and image processing from classical order to an arbitrary order, which suggests an augmentation of information handling strategies (Gao and Zhou, 2011; Gao et al., 2011a,b; Mathieu et al., 2003; Pu et al., 2010). In this section, a non-integer derivative operator is reviewed to demonstrate how it can improve the criterion of the thin edge detection, detection selectivity in the case of parabolic luminance transitions and the criterion of immunity to noise observed as robustness to noise in general.

Grünwald-Letnikov definition of fractional calculus originates from the classical definition of the integer-order differentiation for a continuous function. This is deduced by generalizing the differential order from an integer to a fraction. Assume that $\forall\ v \in \mathbb{R}$, the signal $F(t) \in [a,t]$ with $a \leq t$ and $a,t \in \mathbb{R}$ is $m \in \mathbb{Z}$ order continuous

differentiable. The fractional derivative of order *v* where *m* is no less than r*v*s can be expressed in the form:

$$ {}_a^G D_t^v F(t_q \text{“} \lim_{h\to 0} \sum_{m=0} h^v \Gamma(m+1)\Gamma(v-m+1) F\ t \quad mh \tag{2.3.3}$$

p ´ q

From Equation (2.3.3), the difference equation is expressed as:

$$d^v F \quad \text{´} v\text{p´}v \text{`} 1\text{q}$$
$$\text{“ } F\text{pt q } \text{`} \text{ p´}v\text{q}F\text{p}t \text{ ´ } 1\text{q } \text{`} \quad F\text{p}t \text{ ´ } 2\text{q } dt^v$$
$$2$$
$$\text{´}v\text{p´}v \text{`} 1\text{qp´}v \text{`} 2\text{q}$$
$$\text{`} \quad F\text{p}t \text{ ´ } 3\text{q } \text{`} \cdots\cdots$$
$$6$$
$$\Gamma\text{p}n \text{ ´ } v\text{q}$$
$$\text{`} \quad F\text{p}t \text{ ´ } n\text{q } \text{`} \cdots\cdots \tag{2.3.4}$$
$$\Gamma\text{p}n \text{ `} 1\text{q}\Gamma\text{p´}v\text{q}$$
$$\text{r}\underline{t\text{´}ha}\text{s p´}1\text{q}m \qquad \Gamma\text{p}v \text{ `} 1\text{q}$$

Similarly, for the signal *F*p*x,y*q, the *v* order derivative could be expressed as:

$$\text{B}^v F \qquad \text{r}^{t-a}_{h}\text{]} (-1)^m \qquad \Gamma(v \text{ `} 1\text{q}$$
$$\text{“} \quad \lim \qquad F\text{p}x \text{ ´ } mh,y\text{q} \tag{2.3.5}$$
$$\text{B}x^v \qquad h\tilde{\text{N}}0 \; m\ddot{\text{y}}\text{“}0 \qquad h^v \qquad \Gamma\text{p}m \text{ `} 1\text{q}\Gamma\text{p}v \text{ ´ } m \text{ `} 1\text{q}$$

$$\frac{\mathrm{B}^{v}F}{\mathrm{B}x^{v}} \approx \lim_{h\to 0} \frac{\sum_{m=0}^{\lfloor \frac{t-a}{h} \rfloor} (-1)^{m} \Gamma(v+1)}{h^{v}\Gamma(m+1)\Gamma(v-m+1)} F(x, y-mh) \quad (2.3.6)$$

$$\frac{\mathrm{B}^{v}F}{\mathrm{B}y^{v}} \approx \lim_{h\to 0} \frac{\sum_{m=0}^{\lfloor \frac{t-a}{h} \rfloor} (-1)^{m} \Gamma(v+1)}{h^{v}\Gamma(m+1)\Gamma(v-m+1)} F(x-mh, y)$$

The differences of fractional partial derivative respectively are expressed as:

$$\begin{aligned}\frac{\mathrm{B}^{v}F}{\mathrm{B}x^{v}} \approx\ & F(x,y) + \frac{(-v)}{2} F(x-1,y) + \frac{(-v)(-v+1)}{2} F(x-2,y) \\ & + \frac{(-v)(-v+1)(-v+2)}{6} F(x-3,y) + \cdots \\ & + \frac{\Gamma(n-v)}{\Gamma(-v)\Gamma(n+1)} F(x-n,y) + \cdots \frac{\Gamma(n-v)}{\Gamma(n-1)\Gamma(-v)} \end{aligned} \quad (2.3.7)$$

$$\begin{aligned}\frac{\mathrm{B}^{v}F}{\mathrm{B}y^{v}} \approx\ & F(x,y) + \frac{(-v)}{2} F(x,y-1) + \frac{(-v)(-v+1)}{2} F(x,y-2) \\ & + \frac{(-v)(-v+1)(-v+2)}{6} F(x,y-3) + \cdots \\ & + \frac{\Gamma(n-v)}{\Gamma(-v)\Gamma(n+1)} F(x,y-n) + \cdots \\ & + \frac{\Gamma(n-v)}{\Gamma(n-1)\Gamma(-v)} \end{aligned} \quad (2.3.8)$$

According to Equations (2.3.7) and (2.3.8), among the n non-zero coefficients, only the coefficient of the first term is the constant 1, the other $n-1$ non-zero coefficients are functions with respect to the fractional order v.

In Gao et al. (2014), these fractional derivative operators are made more precise to improve the fractional algorithm. Let the points $F(t-mh)$, $m = 0,1,2,\cdots,n$ in Equation (2.3.4) be are viewed as nodes. To express any point between $t-mh-h$ and$t-mh+h$, let $\xi = t-mh+\frac{v}{2}h$. Thus, for any three nodes $F(t-mh-h)$ and $F(t-mh)$, $F(t-mh+h)$, using Newton interpolation equation, one has interpolation expression of the signal function $F(t)$ in Equation (2.3.4) as:

$$F(\xi) = F[t-mh-h] + F[t-mh-h, t-mh]$$
$$\times (\xi - (t-mh-h))$$
$$+ F[t-mh-h, t-mh, t-mh+h]$$
$$\times (\xi - (t-mh-h))(\xi - (t-mh)) \qquad (2.3.9)$$

where

$$F[t-mh-h] = F(t-mh-h) \qquad (2.3.10)$$

$$F[t-mh-h, t-mh] = \frac{F(t-mh) - F(t-mh-h)}{h} \qquad (2.3.11)$$

$$F[t-mh-h, t-mh, t-mh+h]$$

$$= \frac{F[t-mh, t-mh+h] - F[t-mh-h, t-mh]}{2h}$$

$$= \frac{\frac{F(t-mh+h)-F(t-mh)}{h} - \frac{F(t-mh)-F(t-mh-h)}{h}}{2h}$$

$$= \frac{F(t-mh+h) - 2F(t-mh) + F(t-mh-h)}{2h^2} \qquad (2.3.12)$$

Putting Equations 2.3.10 to 2.3.12 into Equation 2.3.9, the following result is obtained:

$$F(\xi) = F(t-mh-h) + \frac{F(t-mh) - F(t-mh-h)}{h}$$
$$\times (\xi - (t-mh-h))$$
$$+ \frac{F(t-mh+h) - 2F(t-mh) + F(t-mh-h)}{}$$

$$2h^2$$

$$(\xi - (t - mh - h))(\xi - (t - mh)) \qquad (2.3.13)$$

Since $\xi = t - mh + \frac{v}{2}h$, Equation (2.3.13) becomes:

$$F(\xi) = F(t - mh - h) + \frac{F(t - mh) - F(t - mh - h)}{h}\left((t - mh + \frac{v}{2}h) - (t - mh - h)\right)$$

$$+ \frac{F(t - mh + h) - 2F(t - mh) + F(t - mh - h)}{2h^2}\left((t - mh + \frac{v}{2}h) - (t - mh - h)\right)\left((t - mh + \frac{v}{2}h) - (t - mh)\right)$$

$$= F(t - mh - h) + \left(\frac{v}{2} + 1\right)\left(F(t - mh) - F(t - mh - h)\right)$$

$$+ \frac{v(v+2)}{8}\left(F(t - mh + h) - 2F(t - mh) + F(t - mh - h)\right)$$

$$= \left[\left(\frac{v}{2} + 1\right) - \frac{v(v+2)}{4}\right] F(t - mh)$$

$$+ \left[1 - \left(\frac{v}{2} + 1\right) + \frac{1}{8}v(v+2)\right] F(t - mh - h)$$

$$+ \frac{1}{8}v(v+2) F(t - mh + h)$$

$$= \left(\frac{1}{8}v^2 - \frac{1}{4}v\right) F(t - mh - h) + \left(1 - \frac{v^2}{4}\right) F(t - mh)$$

$$\left(\frac{1}{8}v^2+\frac{1}{4}v\right)F(t-mh+h)\Bigg) \tag{2.3.14}$$

It is observed by comparing $F(t)$ in Equations (2.3.4) and $F(\xi)$ in Equation (2.3.14) that, the signal value $F(\xi)$ is non-node. This non-node signal value $F(\xi)$ is a linear combination of the nodes $F(t-mh-h)$, $F(t-mh)$, and $F(t-mh+h)$ which implies that, $F(\xi)$ contains more information of its neighborhood. Replacing $F(t)$ by $F(\xi)$ and taking Equation (2.3.14) into Equation (2.3.4), the approximation of Equation (2.3.4) is obtained as follows:

$$\frac{d^vF(t)}{dt^v} \approx h^{-v}\sum_{m=0}^{n-1}\frac{(-1)^m\,\Gamma(v+1)}{\Gamma(m+1)\Gamma(v-m+1)}\left(\left(\frac{1}{8}v^2-\frac{1}{4}v\right)F(t-mh-h)+\left(1-\frac{v^2}{4}\right)F(t-mh)+\left(\frac{1}{8}v^2+\frac{1}{4}v\right)F(t-mh+h)\right)$$

$$\approx \left(\frac{v^2}{8}-\frac{v}{4}\right)F(t-1)+\left(1-\frac{v^2}{4}\right)F(t)+\left(\frac{v^2}{8}+\frac{v}{4}\right)F(t+1)$$

$$-\,v\left(\left(\frac{v^2}{8}-\frac{v}{4}\right)F(t-2)+\left(1-\frac{v^2}{4}\right)F(t-1)+\left(\frac{v^2}{8}+\frac{v}{4}\right)F(t)\right)$$

$$+\,\frac{-v(-v+1)}{2}\left(\left(\frac{v^2}{8}-\frac{v}{4}\right)F(t-3)+\left(1-\frac{v^2}{4}\right)F(t-2)+\left(\frac{v^2}{8}+\frac{v}{4}\right)F(t-1)\right)+\ldots$$

$$+\,\frac{\Gamma(v+1)}{\Gamma(m+1)\Gamma(v-m+1)}\left(\left(\frac{v^2}{8}-\frac{v}{4}\right)F(t-m-1)+\left(1-\frac{v^2}{4}\right)F(t-m)\ldots\right.$$

$$\frac{1}{4}$$

192 48 192 16 16 3

` ...... ` „^ 2 ´ ` ` ^1 ´ 2 ` v v $\Gamma(n-v+1)$ v $\Gamma(n-v)$ ____ ________________

8 4 $\Gamma(n)\Gamma(-v)$ 4 $\Gamma(n+1)\Gamma(-v)$

`2 __ ` ` $F(t-n)$ ` ...... (2.3.15) v v $\Gamma(n-v+1)$

8 4 $\Gamma(n+2)\Gamma(-v)$

From Equation (2.3.15), the top $n+2$ terms are chosen to approximate the fractional derivative of $f(t)$. Let:

a_{-1} “ $\frac{v^2}{8}$ ` $\frac{v}{4}$ ... 1

a_0 “ $\frac{v^3}{8}$ ... $\frac{v^2}{2}$ ...

a_1 “ $\frac{v^6}{192}$ ... $\frac{v^5}{48}$ ... $\frac{13v^4}{192}$ ... $\frac{3v^3}{16}$ ... $\frac{9v^2}{16}$ ... $\frac{v}{3}$; $\frac{v^5}{48}$ ... $\frac{5v^4}{48}$ ... $\frac{v^3}{12}$ ... $\frac{3v^2}{4}$ ... $\frac{v}{2}$

a_2 “ $\frac{v^4}{...}$ ... $\frac{5v^3}{...}$ ... $\frac{5v}{...}$

a_3 “ $\left(\frac{v^2}{8}-\frac{v}{4}\right)\frac{\Gamma(n-v-1)}{\Gamma(n)\Gamma(-v)}$ ... $\left(1-\frac{v^2}{4}\right)\frac{\Gamma(n-v)}{\Gamma(n-1)\Gamma(-v)}$

......

a_n “ $\left(\frac{v^2}{8}-\frac{v}{4}\right)\frac{\Gamma(n-v-1)}{\Gamma(n-2)\Gamma(-v)}$

Thus,

$$\frac{d^vF(t)}{dt^v} \approx a_{-1}F(t+1) + a_0F(t) + a_1F(t-1) + \cdots + a_nF(t-n) \quad (2.3.16)$$

Similarly for the signal $F(x,y)$, in Equation (2.3.5) and (2.3.6), the approximate backward differences of fractional partial derivatives respectively on negative $x$coordinate and y-coordinate are expressed in the same way as Equation (2.3.15)

as:

B$_v F v_2$ v v_3 v_2

$\frown_v$ “ ˙Fpx ` 1,$y$q ` ˆ´ ´ ` 1˙Fpx,yq

Bx8 4 8 2

`4 $\frown$ $\frac{}{192} + \frac{}{48} - \frac{13}{192} - \frac{3}{16} + \frac{9}{16} - \frac{}{3}$ $F(x-3,y$ ` 3 ´ ˙Fpx ´ 1,yq ` ˆ´ 5 ´ 4 ` 3 ` 2 ´ ˙Fpx ´ 2,yq v 5v 5v v 5v v 3v v 16 16 4 48 48 12 4

2

`6 5 4 3 2 ˙ q ` ¨¨¨¨¨ v v v v v v

` „ˆ 2 ´ ˙ ` ˆ1 ´ 2 ˙ v v Γpn ´ v ´ 1q v Γpn ´ vq

8 4 ΓpnqΓp´vq 4 Γpn ` 1qΓp´$v$q $v^2$ $v$ Γp$n$ ´ $v$ ` 1q

` ˆ ` ´ . ˙Fpx ´ n,yq ` ¨¨¨¨¨ (2.3.17)

8 4 Γpn ` 2qΓp´vq

B F v v v v_2

$_{vv}$ “ ˆ 2 ` ´ ˙$F$p$x,y$ ` 1q ` ˆ´ 3 ´ ` 1˙Fpx,yq

By 8 4 8 2

`4 ˆ ` 3 ´ ˙Fpx,y ´ 1q ` ˆ´ 5 ´ 4 ` 3 ` 2 ´ ˙Fpx,y ´ 2q v 5v 5v v 5v v 3v v 16 16 4 48 48 12

4 2

`6 ˆ ` 5 ´ 4 ´ 3 ` 2 ´ ˙$F$p$x,y$ ´ 3q ` ¨¨¨¨¨ v v 13v 3v 9v v

192 48 192 16 16 3

` „ˆ 2 ´ ˙ ` ˆ1 ´ 2 ˙ v v Γpn ´ v ´ 1q v Γpn ´ vq

ˆ 8 4 ΓpnqΓp´vq 4 Γpn ` 1qΓp´vq

`2 ` ˙ Fpx,y ´ nq ` ¨¨¨¨¨ (2.3.18) $v$ $v$ Γp$n$ ´ $v$ ` 1q

8 4 Γpn ` 2qΓp´vq

From Equation (2.3.17) and Equation (2.3.18), eight new masks which are respectively along the directions of right downward diagonal, negative x-coordinate, positive x-coordinate, negative y-coordinate, left downward diagonal, positive y-coordinate, left upward diagonal and right upward diagonal are implemented which are noted as the fractional operators written as $NIFD_i$, $i = 0, 1, \cdots, 7$.

In some literature work, it was observed that, the traditional Roberts algorithm can locate relatively accurate edges which are thin but not clear enough as a result of noise. Here, the advantages of the fractional operator is used to improve the shortcomings of Robert's operator. Referring to Equation (2.3.4) for a single variable operator and Equations (2.3.7) and (2.3.8) for a two variable operator, the following can be written to recognize edges which are small and prominent:

$$D^v[g(x,y)] = \left|\frac{\partial^v g(x,y)}{\partial x^v}\right| + \left|\frac{\partial^v g(x,y)}{\partial y^v}\right|$$

$$\frac{\partial^v g(x,y)}{\partial x^v} \approx g(x,y) + (-v)g(x-1,y) + \frac{(-v)(-v+1)}{2!}g(x-2,y) + \dots + \frac{\Gamma(-v+1)}{n!\,\Gamma(-v+n+1)}g(x-n,y)$$

$$= 2 * \left\{[f(x,y+1) - f(x+1,y)]^2 + [f(x+1,y+1) - f(x,y)]^2\right\}^{\frac{1}{2}} + (-v)\left\{\begin{array}{l}[f(x-1,y+1) - f(x,y)]^2 \\ +[f(x,y+1) - f(x-1,y)]^2\end{array}\right\}^{\frac{1}{2}} + \frac{(-v)(-v+1)}{2}\left\{\begin{array}{l}[f(x-2,y+1) - f(x-1,y)]^2 \\ +[f(x-1,y+1) - f(x-2,y)]^2\end{array}\right\}^{\frac{1}{2}} + \dots + \frac{\Gamma(-v+1)}{n!\,\Gamma(-v+n+1)}\left\{\begin{array}{l}[f(x-n,y+1) - f(x-n+1,y)]^2 \\ +[f(x-n+1,y+1) - f(x-n,y)]^2\end{array}\right\}^{\frac{1}{2}} \quad (2.3.19)$$

$$\frac{\partial^v g(x,y)}{\partial y^v} \approx g(x,y) + (-v)g(x,y-1) + \frac{(-v)(-v+1)}{2!}g(x,y-2) + \dots + \frac{\Gamma(-v+1)}{\dots}$$

` ..... ` _____ *g*p*x,y* ´ *n*q

n!Γp´*v* ` *n* ` 1q

" * r*f*p*x,y* ` 1q ´ *f*p*x* ` 1,*y*qs² `r*f*p*x* ` 1,*y* ` 1q ´ *f*p*x,y*qs $^{\frac{1}{2}}$

“²

` $v)\left\{ \begin{array}{l} [f(x,y)-f(x+1,y-1)]^2 \\ +[f\ x\quad 1,y\quad f\ x,y\quad 1\quad ^2 \end{array} \right\}^{\frac{1}{2}}$ p´(2.3.20) p `

q $[f(x,y-1)-f(x+1,y-2)]^2 \Big\}^{\frac{1}{2}}$ ´ p ´ qs

` _____ " ` $+[f(x+1,y-1)-f(x,y-2)]^2 \Big\}$ ..... p´*v*qp´*v* `

1q

2

` _____ " $[f(x,y-n+1)-f(x+1,y-n)]^2 \Big\}^{\frac{1}{2}}$

Γp´*v* ` 1q $+[f(x+1,y-n+1)-f(x,y-n)]^2 \Big\}$

n!Γp´*v* ` *n* ` 1q

where the *g*p*x,y*q is the processed image by fractional Roberts while *f*p*x,y*q is the original image.

2.3.4 Other Edge Detectors

Other edge detection approaches include morphological gradient (Pitas and Venetsanopoulos, 1990; Qi et al., 2007), fractal geometry (Tian et al., 2007; Zhang et al., 2008), Mumford-Shah Green function (Mahmoodi, 2012) and high-order and variableorder total variation (Stefan et al., 2010) based techniques. In a late work of Guo and Lai (2013), a new box spline wavelet frame in 8 directions was developed and applied to image edge investigation. Because of the eight distinct bearings, it can discover edges of different sorts quite well. Notwithstanding the steep edges (i.e. nearby discontinuities in intensity), it can find Dirac edges (i.e. transient changes of intensity) and hidden edges (i.e. neighborhood discontinuity in intensity derivatives).

This technique is straightforward furthermore robust to noise.

Chapter 3

METHODOLOGY

3.1 Introduction

The construction of a fractional based convolution mask is not new in the area of image edge analysis however, research in this area seeks to understand how edge points of object can be extracted fully with insignificant commitment of error. Edges are of importance in many computer application since it serves as the basis in image understanding and interpretation. Nonetheless, these edge points of objects in an image can only be efficiently extracted when the appropriate fundamental mathematical tools are engaged. The following discusses the appropriate tools that will be sufficient enough to efficiently extract the edge points.

3.2 Image Processing

3.2.1 Introduction

Let f be a real-valued function defined on the real line R and square integrable as follows:

$$\int_{-\infty}^{\infty} f^2(t)\,dt < \infty \tag{3.2.1}$$

If $f(t)$ is considered as a function value of a signal at some time t, then the signal f can be analyzed in ways other than the time-value form, i.e $t \to f(t)$. Signals are analyzed

in terms of frequency components and various combinations of time and frequency components. These component parts of the signal (edge map, texture, background and noise) are at times altered to remove unwanted features (Noise/Texture) or compressed to get more efficient storage. However, it is essential for this signal to be reconstructed from its component parts and this constitute the Analysis, Processing and Synthesis in Image Processing.

To analyse a signal is to decompose it into its basic components. If a signal space is considered as a vector space, then this signal can be broken up into series of subspace where each subspace captures a special feature of the signal.

Processing of signal involves the modification of basic components of the signal that are obtained from the analysis. Reconstruction from basic components after it has been altered is what is called synthesis. All these help to understand signals when it comes to comparing the altered signals with respect to the original signal in test for convergence. Here, two image signals are said to have converged if their signal difference is below a predefined value ε. In that case, the two images are considered to be the same.

3.2.2 Image Formation

An image may be viewed as being derived from a continuous image function, $f : R^2 \to R^2 \times R^2$ by taking a finite number of samples where

the function value $f(x)$ represents the intensity and some desired parameters of the physical image at the point $x \in R^2$. Nixon and Aguado (2002) defined digital image as a pixel (unit square area). The value at each pixel is proportional to the brightness of the corresponding pixel and this value is often derived from the output of an Analog/Digital (A/D) converter.

The points in an $M \hat{} N$ array represent the pixel points and the element of these points are element of the matrix. Conventionally, the value of the pixel in the matrix ranges from 0 to $2^p - 1$ where p is an integer. For example, the possible values of $p = 8$ gives the brightness levels between 0 and 255 which is usually displayed as black for 0 and white for 255 respectively with shades of grey in between them.

Color images follow a similar strategy in specifying its pixel intensity. Example, the p$R - G - B$q image space has the following components as red, green and blue. In any colours mode, the pixel's colour can be specified in two ways.

1. First, you can associate an integer value for each pixel that can be used as an index to a table that stores the actual intensity of each colour component. The index is used to recover the actual colour from the table when the pixel is going to be displayed or processed. In this method, the table is known as the image palette and the display is performed by colour mapping.

2. The alternative is to use several image planes to store the colour components of each pixel. This method is known as true colour and it represents an image more accurately, essentially by considering more colour. The most common format uses 8 bits for each of the three $R - G - B$ components. These images are known as 24-bit true colour and they can contain $2^{24} = 16777216$ different colours simultaneously.

3.2.3 Image Arithmetic Techniques

Image arithmetic can be performed on two or more images. It consist of sequence of logical and standard arithmetic operations. The operators in a pixel-by-pixel (pixel value in the same position) fashion are applied to the input images. They are simple and fast in their implementation.

Pixel Addition

The Pixel Addition operator or offset takes two identically sized images as input and produces a third image as output which has the same size as the first two. Each pixel value in the output image is the sum of the values of the corresponding pixel from each of the two input images. This property can however be expanded to allow more than two images to be combined by using various additive laws (Davies, 1990). Given that, P_1 and P_2 are the first and second image respectively at position i,j then, the output pixel values are given by:

$$I(i,j) = P_1(i,j) + P_2(i,j) \quad \text{For a} \qquad (3.2.2)$$

constant value K, the pixel addition can be defined as:

$$I(i,j) = P_1(i,j) + K \qquad (3.2.3)$$

If the pixel values in the input images are vectors of true colour (RGB) rather than scalar values then the output value is obtained by adding separately the components. In situations where the pixel addition is greater than the maximum allowed pixel value, then the pixel value is set to the maximum allowed value which may result in a deceptive image known as the saturation effect.

Pixel Subtraction

The pixel subtraction or difference takes as input two images to produce a third image as output where each pixel value is obtained by subtracting the pixel of the first image from the corresponding pixel values of the second image. Just as it was for the case pixel addition, one can also subtract a constant value K from all pixels of a single image. Marion (1991) and Davies (1990) advised that, the absolute difference between pixel values should be used instead of signed output. The output pixel values are given by:

$$I(i,j) = P_1(i,j) - P_2(i,j) \qquad (3.2.4)$$

$$I(i,j) = P_1(i,j) - K \qquad (3.2.5)$$

If the operator computes absolute difference between the two input images then:

$$I(i,j) = |P_1(i,j) - P_2(i,j)| \quad (3.2.6)$$

In this operation, it is practically impossible for the output pixel values to be outside the range of allowed pixel as with the case of pixel addition, hence the advantage of using absolute differences. However, in the absence of absolute, all negative values are set to zero.

Pixel Multiplication and Scaling

Pixel multiplication more often than not comes in two structures. The first takes information from two input images to create a third image in which the pixel qualities are only those of the first picture multiplied by the comparing pixel values in the second image. The second one is termed as the scaling strategy where every pixel value is multiplied by a predetermined steady *K*. This is done utilizing the formula:

$$I(i,j) = P_1(i,j) \times P_2(i,j) \quad (3.2.7)$$

$$I(i,j) = P_1(i,j) \times K \quad (3.2.8)$$

The constant *K* is regularly a floating point number that might be greater than or less than one relying upon the required impact. In the event that the pixel quality is a vector instead of scalar value then the individual parts (i.e Red, Blue and Green segments) are multiplied independently to create the resultant image. Again if the resultant image qualities are computed to be bigger than the most extreme permitted pixel value, then, they might either be truncated at that most extreme value or wrap-around to proceed upwards from the least permitted number (Boyle and Thomas, 1988). A scaling variable more prominent than one (1) will light up an image and when under one will obscure the image. Scaling for the most part delivers a significantly more natural lighting up/obscuring impact as it preserves the relative complexity (contrast) of the image preferable rather over just adding to the pixel shading (Boyle and Thomas, 1988).

Using pixel-by-pixel multiplication, a binary mask can be utilized to produce another image with a specific end goals. This is accomplished by duplicating 1 (one) to pixels that are to be protected and 0 (zero) otherwise.

Pixel Division

The pixel division operator takes two images as input and produces a third pixel values that are pixel values of the first image divided by the corresponding pixel values of the second image. This can also be implemented by dividing with a specified constant K of the single input image. The division of two images, P_1 and P_2 is given by the formula:

$$I(i,j) = P_1(i,j) \div P_2(i,j) \quad I(i,j) = P_1(i,j) \div K \tag{3.2.9}$$

If integer division is performed, then the resulting image values are rounded down to the next lowest integer for the output image. Unlike pixel subtraction which gives absolute change for each pixel value, division gives the fractional change between the corresponding pixel values.

3.2.4 Point Operations

Single point processing is an image enhancement technique which determines a pixel value of the enhanced image depending only on the value of the corresponding pixel in the image. The process can be described with the mapping function as follows (Marion, 1991):

$$S = M(R) \tag{3.2.10}$$

where R and S are the pixel values in the input and output images respectively. The type of the mapping function M, decides the impact of the operation. It can likewise

be characterized in a specially appointed way utilizing the thresholding or gamma adjustment or by the calculation of the histogram from the input image.

This point operators are also known as Look-Up Table (LUT) Transformations, since the mapping function in the case of a discrete image can be implemented in a look-up table. A sub class of the point operators are:

1. Histogram Equalization
2. Intensity Histogram
3. Local Enhancement

Histogram Equalization Operators

Histogram equalization operators provide a method for modifying the contrast of an image by altering the image such that its intensity has a desired value. Histogram equalization employs a monotonic, non-linear mapping which re-assigns the intensity values of pixels in the image such that the output image contains a uniform distribution of intensities.

This technique is usually introduced using continuous process function rather than discrete process functions (Semmlo, 2004). Let suppose that, the image of interest contains continuous intensity levels on the interval r0,1s and that, the transformation function f which maps an input image Apx,yq onto an output image Bpx,yq is continuous within this interval. Further, let assume that the transfer law which may be written in terms of intensity density levels D_B " fpD_Aq is a single-valued and monotonic increasing so that it is possible to define the inverse law (Boyle and Thomas, 1988):

$$D_A \text{ " } f^{1}\text{p}D_B\text{q} \qquad (3.2.11)$$

All pixels in the input image with densities in the region D_A to D_A ` $dD_A$ will have their pixel values re-assigned such that they assume an output pixel density value in the range from $D_B$ to $D_B$ ` dD_B. The surface areas $h_A(D_A)dD_A$ and $h_B(D_B)dD_B$ will therefore be equal which result in:

$$h_B(D_B) = h_A(D_A) = d(D_A) \tag{3.2.12}$$

where

$$d(D_A) = \frac{df(D_A)}{dD_A}$$

This result can be written in the language of probability theory if the histogram h is regarded as a continuous probability density function P describing the distribution of the assumed random intensity levels (Marion, 1991; Davies, 1990):

$$P_B(D_B) = P_A(D_A) = d(D_A) \tag{3.2.13}$$

In the case of histogram equalization, the output probability densities should all be an equal fraction of the maximum number of intensity levels in the input image D_M where the maximum level considered is 0 (zero). The transfer function or point operator is:

$$d(D_A) = D_M \hat{} P_A(D_A) \tag{3.2.14}$$

therefore,

$$f(D_A) = D_M P_A(u)du = D_M \hat{} F_A(D_A) \tag{3.2.15}$$

where $F_A(D_A)$ is the cumulative probability distribution or the cumulative histogram of the original image. Thus, an image which is transformed using its cumulative histogram produces an output histogram which is flat. A digital implementation of histogram equalization is usually performed by defining a transfer function of the form:

$$fpD_Aq \text{ “ } \max\acute{}0,\text{round}”\left(D_M \times \frac{n_k}{N^2}\right] - 1\right) \qquad (3.2.16)$$

where N is the number of image pixels and n_k is the number of pixels at intensity level k or less.

Intensity Histogram Operators

In the context of image processing, histogram of an image normally refers to a histogram of the pixel values. This histogram is a graph showing the number of pixels in an image at each different intensity value found in that image. For an 8-bit gray-scale image, there are 256 different possible colour intensities and so the histogram will graphically display 256 numbers showing the distribution of colour intensity amongst those gray-scale values.

Histogram can also be taken on colour images. Here, individual histogram of red, green and blue channels can be taken or a 3-D histogram can be produced with three axis representing the channels and brightness at each point. This represent the pixel count, however the exact output from the operation depends heavily on its implementation.

One common use of histogram is to decide what value of threshold to use when converting a gray-scale image to a binary image of 0 (black) and 1 (white). If the image is suitable for thresholding, then the histogram will be bimodal which means the pixel colour will be clustered around two well distinct values. A suitable threshold for separating these two groups will then be found somewhere in between the two peaks in the histogram otherwise it will be unlikely that a good segmentation can be produced by thresholding.

These two histogram processing techniques discussed are global, in the sense that they apply a transformation function whose form is based on the intensity level distribution of an entire image.

Local Enhancement Operators

Although the above methods can enhance the overall contrast of an image thereby making certain details more visible, there are instances where enhancement of details over small areas may be desired (Stollnitz et al., 2006; Oberst, 2007) . The solution in this case is to derive a transformation based on the intensity distribution in the local neighborhood for the patch of the image.

The procedure includes characterizing an area around every pixel of the patch and furthermore, utilizing the histogram qualities of this area to infer a transfer function which maps that pixel unto an output image. This is performed for every pixel in the patch.

3.2.5 Geometric Operations

A geometric operation maps pixel information that is, the intensity value at each pixel location say px_1,y_1q in an input image is mapped into another location say px_2,y_2q in an output image. For basic operators, the first order polynomials take the form: $\begin{matrix} x_2 \\ y_2 \end{matrix} = A \begin{matrix} x_1 \\ y_1 \end{matrix} + B$ (3.2.17)

Image translation can be attained by determining qualities for the matrix, B while rotation, scaling and reflection are characterized by instantiating variables in the matrix, A. When all is said and done, affine transformation which make utilization of both matrices can be utilized to perform a blend of these operations. This transform is regularly utilized while remedying geometric mutilations as a part of an image presented by point of view abnormalities. Geometric operations are likewise used to

enhance the perception of an image. An example is, zooming a fascinating locale of the image.

Geometric Scaling

This operator is used in scaling, zooming, shrinking, interpolation and sub-sampling of images. The scale operator performs a geometric transformation which can be used to shrink or zoom the size of an image. Image reduction, commonly known as sub-sampling is performed by replacing a group of pixel values by one chosen pixel value from within the group or by interpolating between pixel values in a local neighborhood. Image zooming is also achieved by pixel replication or by interpolation.

Geometric Rotation

The rotation operator performs a geometric transform which maps the position $p x_1,y_1 q$ in an input image onto a position $p x_2,y_2 q$ in an output image by rotating it through a user-specified angle θ about an origin O. In most practices, output location $p x_2,y_2 q$ which are outside the boundary of the image are ignored. The operator performs a transformation in the form:

$$x_2 \text{ “ } \cos p\theta q \text{ ˆ } p x_1 \text{ ´ } x_0 q \text{ ´ } \sin p\theta q \text{ ˆ } p y_1 \text{ ´ } y_0 q \text{ ` } x_0$$

$$y_2 \text{ “ } \cos p\theta q \text{ ˆ } p x_1 \text{ ´ } x_0 q \text{ ´ } \sin p\theta q \text{ ˆ } p y_1 \text{ ´ } y_0 q \text{ ` } y_0 \qquad (3.2.18)$$

where $p x_0,y_0 q$ is the coordinate of the centre of rotation of the input image and θ is the angle of rotation with clockwise rotations having positive angles.

Geometric Reflection

The reflection operator geometrically transforms an image such that pixel values located at position $p x_1,y_1 q$ in an input image are reflected about a user-specified image axis to obtain $p x_2,y_2 q$ in the corresponding output image. Some commonly used transformations are the following:

1. Reflection about the vertical axis x_0 of the input image:

$$x_2 = -x_1 + 2 \cdot x_0$$

$$y_2 = y_1 \qquad (3.2.19)$$

2. Reflection about the horizontal axis y_0 of the input image:

$$x_2 = x_1 \qquad y_2 = -y_1 + 2 \cdot y_0 \qquad (3.2.20)$$

3. Reflection about the axis oriented in any arbitrary direction θ and passing through (x_0, y_0):

$$x_2 = x_1 + 2 \cdot A \cdot (-\sin\theta) \qquad y_2 = y_1 + 2 \cdot A \cdot (-\cos\theta) \qquad (3.2.21)$$

where $A = (x_1 - x_0) \cdot \sin\theta - (y_1 - y_0)$. If (x_0, y_0) is not the center of the input image, part of the image will be reflected out of the visible range of the image. In most implementation, such regions are filled out with black pixels (0 values).

Geometric Translation

Under translation, a pixel position located at (x_1, y_1) in the input image is shifted to a new position (x_2, y_2) in the corresponding output image by displacing it through a user specified translation (β_x, β_y). This operator performs transformation of the form:

$$x_2 = x_1 + \beta_x$$

$$x_2 = y_1 + \beta_y \qquad (3.2.22)$$

The operator takes two arguments (β_x, β_y) which specifies the desired horizontal and vertical pixel displacements respectively.

3.2.6 Classification Algorithm

Classification includes a broad range of theoretic decision techniques to the identification of images. Algorithms for classification usually employ three phases of data processing that is: training, validation and testing. In the initial training phase, characteristic properties of typical image features are isolated based on the unique description of each classification category. Thus, the training class and the model are created to identify various objects in the image. The model is validated and then finally the testing. The set of examples used for learning where the target value is known is referred to as the "Training Set". The set of examples used to fine tune the architecture of the classifier and estimate the error is known as the "Validation Set". Finally the "Test Set" is used only to assess the performance of the validated classifier. It is usually not used during the training process so as the error on the test set provides an unbaised estimate of the generalisation error. All classification algorithms are based on the assumption that, the image in question depicts one or more features and that each of these features belongs to one of the several distinct classes (Boyle and Thomas, 1988; Marion, 1991; Davies, 1990). All the several distinct image classes may be specified a priori having knowledge of the attribute of the object (i.e supervised classification) otherwise it is unsupervised classification.

Supervised Classification

In a supervised classification, the training data includes both the input and the desired (targets) result which are given to the model as input during the learning process. Here, the construction of a proper training, validation and test is crucial and these methods are usually fast and accurate. This approach works as follow: assuming one bag is filled with salmons and another bag is filled with sea bass. A model is then built out of the given data taking into account the specific content of the bags. Now, given a single bag containing contents from both bags, use the model

to classify them into their respective bags. Examples of methods that fall under this category are: Neural Networks, Multi Layer Perceptron and Decision Trees.

Unsupervised Classification

In this approach of classification, the model is not provided with the correct result (targets) during the training phase. The idea is to cluster the input data in classes on the basis of their statistical properties only and labeling them. This approach also works as follow: Given a bag with mixed content, build a model out of it. Given another bag with mixed content but with the same features like the first one, classify them into their various group. Examples of such methods are: K-means, Self Organizing Maps, Distance and Normalisation.

Remark

In either case, the motivating criteria for constructing training classes are that, they are unique set of parameters. The convenient way of building a parametric description of this sort is through a feature vector V (i.e V " $v_1, v_2, \cdots, v_n$) where n is the number of attributes which describe each image feature (feature vector) refer to as the training class. Each image feature will form a subspace of the classification space. Each training class consist of a set of objects with similar feature vectors and belong to the same class of subspace. A set of n class is a set of m row vector where each of n-dimensional attributes is a class. This set represent an entity. Different entity may have p entity with the same set of n class description. A collection of all these features form the feature space.

3.2.7 Morphological Operators

Morphological operator is a composite operator of relational operation and flip operation. It uses one original image as input with a structuring element imposed on it

to produce a new image. This structuring element process objects in the input image based on the shape characteristics encoded in it.

The pixel of interest of the structuring element is overlayed on the input image and shifted pixel by pixel over the input image. At each pixel position of the input image, the value of the structuring element in the neighborhood of the Pixel of Interest (POI) are compared to the corresponding value in the input image. By the use of the relational operator, a component wise element is obtained. If certain condition is satisfied, then a flip operation is performed at the pixel position of the input image where the POI is located. The structuring element is usually of size 3 ˆ 3 and the basic morphological operators are dilation and erosion.

Dilation and Erosion

The structuring element used in dilation has entry values of 1's in a 3ˆ3 matrix. In the overlay, the relational operator generates pairs of value from the input image and the structuring element that are either {0,1} and {1,1}. If at least a pair of {1,1} is found, then, the pixel value of the image position where the POI is assigned change to 1 otherwise it remains zero.

Suppose that X is zero/one where xpi,jq is the binary colour of the position pi,jq of the image. Let K be an n ˆ n matrix of ones where n is an odd number. Let the POI of K be m if m " pi,jq (i.e. overlay) then the set of all points pi,jq of the input image such that m " pi,jq with X be non-empty is a dilation of the matrix.

In like manner, erosion is similar to that of dilation. However, this operator assign 0 to the POI if at least a pair of {0,1} is found otherwise it remains one.

Opening and Closing

Opening and closing are both derived from the fundamental operations of erosion and dilation. These operators are normally applied to binary images although there are

grayscale versions. Opening is defined as an erosion followed by a dilation using the same structuring element for both operations. The basic effect of an opening is somewhat like erosion in the sense that, it tends to remove some of the foreground (1) pixels from the edges.

On the other hand, closing is opening performed in reverse. It is defined as a dilation followed by an erosion using the same structuring element for both operations. Closing is also similar to dilation as it tends to enlarge the boundaries of the foreground. The exact operation for both opening and closing is determined by a structuring element. The effect of the operator is to preserve foreground regions that have a similar shape to the structuring element.

Thinning

Thinning is a morphological operation that is used to remove selected foreground pixels from binary images just like erosion/opening operations. It is commonly used to tidy up the output of an edge detection. An edge detection may have noise in the extracted edge points. These neighborhood noise can be removed by thinning. The thinning process uses the hit-and-miss (ham) transform and the erosion operation.

Let *I* be an image and *J* be the structuring operator, then ham(I,J) is the hitand-miss-transform. Suppose thin(I,J) is the operator then:

$$\text{thinp}I,J\text{q} \text{ “ } I \text{ ´ } \text{hamp}I,J\text{q} \qquad (3.2.23)$$

where the subtraction is defined by $I \text{ ´ } J \text{ “ } I\text{X} \text{ „ } J$. The choice of structuring element is determined by the type of edge map obtained.

3.2.8 Digital Filters

A filter is a mask that is used to smoothen noisy image by depressing high frequencies and enhancing low frequencies. Images are often corrupted by noise from several

sources but those that are most frequently encountered in practice are the additive noise (e.g. Gaussian noise) and impulse noise (e.g. salt and pepper noise). Linear filters such as the mean filters are the primary tool for smoothing digital images degraded by additive noise. An image can be filtered either in the frequency or in the spatial domain. Images that are in the spatial domain need to be converted into the frequency domain by an appropriate transform and then filtered. The resultant image is then transformed back into its spatial domain. The filter function is shaped so as to attenuate some frequencies and enhance others. For example, a simple lowpass filter is 1 for frequencies smaller than the cut-off frequency and 0 for all

others.

The corresponding process in the spatial domain is to convolve the input image *f*p*i,j*q with the filter function *h*p*i,j*q. This can be written as:

$$G(i,j) = h(i,j) * f(i,j) \qquad (3.2.24)$$

This mathematical operation is identical to multiplication in the frequency space but the results of the digital implementations vary since we have to approximate the filter function with a discrete and finite kernel.

The discrete convolution can be defined as a 'shift and multiply' operation by moving the POI of the operator over the image pixel positions. The estimations of the comparing pixel values of the image are multiplied and the aggregate of the operation is taken. For a square kernel of size $M \times M$, where M is odd, the output image can be calculated with the following formula:

$$G(i,j) = \sum_{m=-\lfloor \frac{M}{2} \rfloor}^{\lfloor \frac{M}{2} \rfloor} \sum_{n=-\lfloor \frac{M}{2} \rfloor}^{\lfloor \frac{M}{2} \rfloor} h(m,n) f(i-m, j-n) \qquad (3.2.25)$$

Various standard kernels exist for specific applications and examples are the mean and median filters.

Mean Filter

Mean filtering also called averaging filtering is a method for smoothing out noise in an image. It reduce the amount of intensity variation of one pixel and the next.

The idea of mean filtering is to replace each pixel value in an image with the average value of its neighbors including the value of the original image. This has the effect of replacing pixel values which are unrepresentative of their surroundings. The mask (filter) can be created mathematical as follows:

$$\overline{M(i,j)} = \frac{1}{n} \sum_{i} \sum_{j} k(i,j) \qquad (3.2.26)$$

where $k(i,j)$ is the value of the mask at position (i,j) and n is the number of element in the mask. Smaller kernels are often used. Applying a small kernel more than once produces a different effect from a single pass with large kernel.

Median Filter

Median filter also known as the rank filter is used to reduce noise in an image. However, it preserves useful details in the image much better than the mean filter. Instead of replacing the pixel value of the image at the POI with the mean pixel value, it replaces it with the median of the pixel values.

Gaussian Smoothing

The Gaussian Smoothing operator, (Davies, 1990) is a 2-Dimensional convolution operator that is used to remove noise, however, it may end up blurring an image. This kernel has the following definition and it evaluate to 1:

1. The Gaussian distribution in 1-Dimension has the form

$$1 \quad ' \underline{x}^{22}$$

$$G(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-x^2}{2\sigma^2}} \quad (3.2.27)$$

where σ is the standard deviation of the distribution.

2. In 2-Dimensions, a circularly symmetric Gaussian has the form:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (3.2.28)$$

The idea of Gaussian smoothing is to use the 2-Dimensional distribution as a point-spread function and is achieved by convolution. Since the image is stored as a collection of discrete pixels, a discrete approximation to the Gaussian function is required before convolution can be performed. An alternative approach to compute a Gaussian smoothing with a large standard deviation is to convolve the image several times with Gaussian smoothing with smaller standard deviation. Although this is computationally expensive, it can have applicability if the processing is carried out using high performance machines. The filter has the following properties:

1. The mean, median and the mode coincide at the mean position.
2. One variance away from the mean is 68.26%
3. Two variance away from the mean is 95.46% and
4. Three variance away from the mean is 99.72%

3.2.9 The Norm of Two Pixels

The distance transform (norm) is an operator normally applied to binary images only where the foreground value is 1 and the background value is 0. The result of the transform is a gray-scale image. That looks similar to the input image, except that the gray-level intensities of pixel points inside the foreground regions are changed to

show the distance to the closest boundary point which is interpreted as a grayscale value. The figure below shows a distance transform (Euclidean, City Block or Chessboard) for a simple rectangular shape.

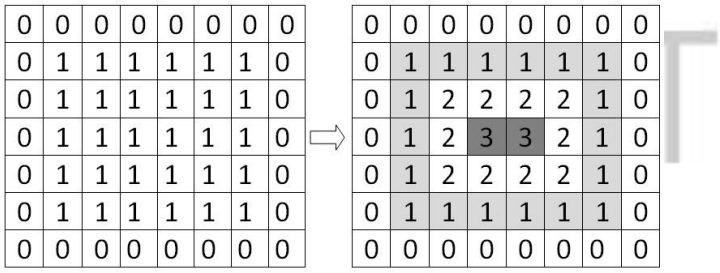


Figure 3.1: The distance transform

The distance transform describe above can be used on a dual image where we compute distance from the background to the foreground. Dual image is an image where the borders are 1 and the interior being 0. There are several different sorts of distance transform, depending upon which distance metric is being used to determine the distance between pixels. The example shown in Figure 3.1 uses the chessboard distance metric but both the Euclidean and the City block metric can be used as

well.

Distance metric is useful in image processing to be able to calculate the distance between two pixels in an image, but this can be complicated in situations where the background is uniformly mixed with the foreground. Sectioned below are the three most important distance transform discussed.

Euclidean Distance

This is the familiar straight line distance. If the two pixels being considered have coordinates px_1,x_2q and py_1,y_2q, then the Euclidean distance is given by:

$$D_{Euclid} \text{“ ap}x_1 \text{´ } x_2\text{q}_2 \text{` p}y_1 \text{´ } y_2\text{q}_2 \qquad (3.2.29)$$

City Block Distance

This is also known as the Manhattan distance. This metric assumes that in going from one pixel to the other it is only possible to travel directly along pixel grid lines. Diagonal moves are not allowed. Therefore, the city block distance is given by:

$$D_{City} = |x_1 - x_2| + |y_1 - y_2| \tag{3.2.30}$$

Chessboard Distance

This metric assumes that you can make moves on the pixel grid as if you were a King making moves in chess, thus diagonal move counts the same as a horizontal move. The metric is given by:

$$D_{Chess} = \max(|x_1 - x_2|, |y_1 - y_2|) \tag{3.2.31}$$

The last two metrics are usually much faster to compute than the Euclidean metric however they are considered to be less accurate.

3.3 Edge Detection by Segmentation

3.3.1 Active Contour Model

The basic idea in active contour models or snakes is to evolve a curve, subject to constraints from a given image u_0, in order to detect objects in that image. For instance, starting with a closed curve that include the object in its interior. The curve moves towards the object through its interior normal and stops on the boundary.

Let Ω be a bounded open subset of $\mathbb{R}^2$ with $\partial\Omega$ its boundary. Let $u_0 : \Omega \to \overline{\mathbb{R}}$ be a given image, and $C(q) : [0,1] \to \mathbb{R}^2$ be a parameterized curve. In the classical snakes or active contour models (Evans and Gariepy, 1969; Bayer and Leveque, 1992; Kass et

al., 1988; Byers and Raftery, 1998), an edge-detector (gradient operator) is used on the image u_0. The evolving curve is stopped when a gradient is detected at the edge of the desired object.

The snake model (Evans and Gariepy, 1969) is:

$$\inf J_1(C) \quad c$$

where

$$J_1(C) = \int_0^1 \alpha |C^1(s)|^2 ds + \beta \int_0^1 |C^2(s)| ds - \lambda \int_0^1 |\nabla u_0(C(s))|^2 ds \qquad (3.3.1)$$

Here, α,β and λ are positive parameters. The first two terms control the smoothness of the contour (the internal energy), while the third term attracts the contour toward the object in the image (the external energy). Equation (3.3.1) is a action integral of the curve C which is a curve of variation. The curve C that minimise the functional $J_1(C)$ is the curve representing the edge of the object in the image. At the end point, the curve should be zero and the derivative should also be zero.

A general edge-detector can be defined by a positive and decreasing function g, depending on the gradient of the image u_0, such that

$$\lim_{(x,y)\to\infty} g = 0$$

An example of g is:

$$g(|\nabla u_0(x,y)|) = \frac{1}{1+|\nabla (G_\sigma(x,y) * u_0(x,y))|^p}, \quad p \geqslant 1 \qquad (3.3.2)$$

where $G_\sigma * u_0$, is the convolution of the image u_0 with the Gaussian $G_\sigma(x,y) = \sigma^{-1}\{2e^{-|x_2-y_2|}\{4\sigma$. The function $g(|\nabla u_0|)$ is positive in homogeneous regions and zero at the edges.

3.3.2 Level Set Model

The level set method of Osher and Sethian (1988) and based on the motion by mean curvature assumption have been used extensively since it allows for cups, corners and topological changes. Moreover, the discretization of the problem is made on a fixed rectangular grid. The curve C is represented implicitly via a Lipschitz function φ, by $C = \{(x,y) | \varphi(x,y) = 0\}$, and the evolution of the curve is given by the zero-level curve at the time t of the function $\varphi(t;x,y) = \varphi_t(x,y)$. Evolving the curve C in normal direction with mean curvature, speed F amounts to solving the differential equation (Osher and Sethian, 1988):

$$\frac{\partial \varphi}{\partial t} = |\nabla \varphi| F, \qquad \varphi(0;x,y) = \varphi_0(x,y) \qquad (3.3.3)$$

where the set $\{(x,y) | \varphi_0(x,y) = 0\}$ defines the initial contour. A particular case is the motion by mean curvature, when $F = \operatorname{div}(\nabla\varphi(x,y)/|\nabla\varphi(x,y)|)$ is the curvature of the level-curve of φ passing through (x,y). The equation becomes $\frac{\partial \varphi}{\partial t} = \nabla\varphi(x,y)$ $\in \mathbb{R}^2$

$$\begin{cases} \frac{\partial \phi}{\partial t} = |\nabla \phi| \operatorname{div}\left(\frac{\nabla \phi}{|\nabla \phi|}\right), & t \in (0,\infty); \\ 0, x, y) = \phi_0(x,y), & \{x,y\} \in \mathbb{R}^2 \end{cases} \qquad (3.3.4)$$

$\varphi(x,y)$

3.3.3 Geometric Active Contour Model

A geometric active contour model based on the mean curvature motion is given by the following evolution equation (Bayer and Leveque, 1992):

$$\frac{\partial \phi}{\partial t} = g(|\nabla I|)\left(\operatorname{div}\left(\frac{\nabla \phi}{|\nabla \phi|}\right) + v\right)|\nabla \phi| \quad , \tag{3.3.5}$$

$$\text{in}(0,\infty) \times \mathbb{R}^2 \quad \phi(0,x,y) = \phi_0(x,y) \in \mathbb{R}^2$$

where:

$g(|\nabla u_0|) = \frac{1}{1+|\nabla u_0|^p}$ is the edge-function with $p = 2$; $v \geq 0$

0 is a constant;

ϕ_0 is initial level set function.

The new level set ϕ_0 moves in the normal direction with speed $g(|\nabla u_0|)(F + v)$ and therefore stops on the desired boundary where g vanishes. The constant v is a correction term chosen so that the quantity $\text{div}\left(\frac{\nabla \phi(x,y)}{|\nabla \phi(x,y)|}\right) + \phi(x,y)$ remains always positive. This constant may be interpreted as a force pushing the curve toward the object, when the curvature becomes null or negative. Also, $v \geq 0$ is a constraint on the area inside the curve, increasing the propagation speed.

In Kass et al. (1988), two variant of the active contour models was proposed. They also depend on the image gradient to stop the curve at the boundary of the object. The first variant is:

$$\frac{\partial \phi}{\partial t} = |\nabla \phi|\left(v + \frac{\nabla}{|\nabla|}\left(\frac{\nabla \phi}{|\nabla \phi|}\right)\right)(|\nabla G_\sigma| + u_0)(M_1 + M_2) \tag{3.3.6}$$

$$\phi(0;x,y) = \phi_0(x,y) \in r(0,\infty) \times \mathbb{R}^2$$

where v is the gradient dependent parameter and M_1 and M_2 are the maximum and minimum values of the magnitude of the image gradient $|\nabla G_\sigma * u_0|$. Again, the speed of the evolving curve becomes zero on the edge points where the set of highest gradient occurs and therefore the curve stops on the desired boundary defined by these strong gradients. The second variant of the model (Bayer and Leveque, 1992) is similar to the geometric model (Evans and Gariepy, 1969), with $p = 1$.

Because all these classical snakes and active contour models rely on the edge function g which depends on the image gradient $|\nabla u_0|$ to stop the curve evolution, these models can detect only objects with edges defined by gradient. If the image u_0 is very noisy, then the isotropic smoothing Gaussian function has to be strong in other to smooth the edges as well.

3.3.4 Chan-Vese Model

In the work of Chan and Vese (2001), a different active contour model without a stopping edge-function was proposed. This model is not based on the gradient of the image u_0 to stop. Instead, the stopping term is modeled based on the MumfordShah segmentation techniques so as to detect contours both with or without gradient (Mumford and Shah, 1989). The Mumford-Shah functional as quoted is:

$$F(\mu, C) = \mu \cdot \text{length}(C) + \lambda \int_{\Omega} |u_0 - u|^2 dx + \int_{\Omega \setminus C} |\nabla u|^2 dx \qquad (3.3.7)$$

In the Chan-Vese technique, let ω = region of object, Ω = region of image/image domain where $\omega \subset \Omega$. Let C be $\partial\omega$, $\bar{\omega}$ is the closure of ω. The region beyond the boundary C denotes $\Omega \setminus \bar{\omega}$. Assume that the image u_0 is formed by two distinct regions

of approximately piecewise-constant intensities u_0^i and u_0^o respectively. Assume further that the object to be detected is represented by the region with the value u_0^i. Let the boundary of u_0^i be C_0. Define the image

$$u_o = \begin{cases} u_0^i & \text{inside } C \\ u_{o0} & \text{outside } C \end{cases} \quad \omega \in \Omega \quad (3.3.8)$$

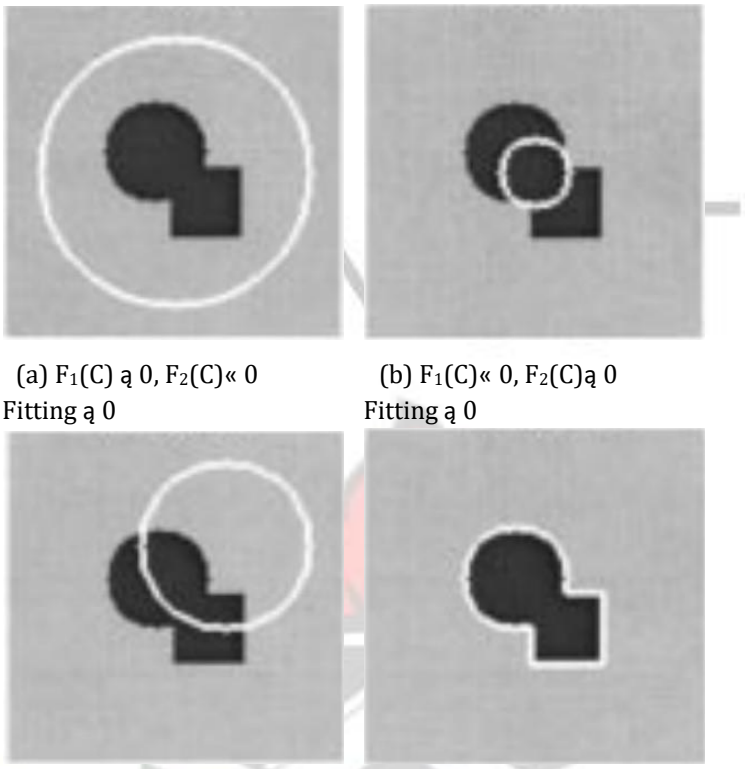
Let consider the following action integral/functional of C ("fitting" term):

$$F_1(C) + F_2(C) = \int_{\text{inside}(C)} |u_0(x,y) - c_1|^2 dxdy + \int_{\text{outside}(C)} |u_0(x,y) - c_2|^2 dxdy \quad (3.3.9)$$

where the parameters c_1, c_2 depends on C and they are the averages of u_0 inside C and outside C respectively. In this case, the boundary C_0 of the object is the minimiser of the fitting term.

$$\inf_C \{F_1(C) + F_2(C)\} \approx F_1(C_0) + F_2(C_0) \quad (3.3.10)$$

If the curve C is outside the object, then $F_1(C) > 0$ and $F_2(C) \approx 0$. If the curve C is within the closure of the object region, then $F_1(C) \approx 0$ but $F_2(C) > 0$. If the curve C lies partially on the closure and outside the region of the object, then $F_1(C) > 0$ and $F_2(C) > 0$. Finally, the fitting energy is minimised if $C = C_0$. These are illustrated in Figure 3.2.



(a) $F_1(C) \geq 0, F_2(C) \leq 0$ Fitting ≥ 0 (b) $F_1(C) \leq 0, F_2(C) \geq 0$ Fitting ≥ 0

(c) $F_1(C) \geq 0, F_2(C) \geq 0$ Fitting ≥ 0 (d) $F_1(C) \leq 0, F_2(C) \leq 0$ Fitting "0

Figure 3.2: Illustration of the fitting term minimisation all possible cases

From the fitting term, the energy functional $F(c_1,c_2,C)$ is defined by adding some regularising term like the length of the curve C and/or the area of the region inside C.

$$F(c_1,c_2,C) = \mu \cdot Length(C) + \nu \cdot Area(inside(C)) + \lambda_1 \int_{inside(C)} |u_0(x,y) - c_1|^2 dxdy + \lambda_2 \int_{outside(C)} |u_0(x,y) - c_2|^2 dxdy \quad (3.3.11)$$

where $\mu,\nu \geq 0, \lambda_1,\lambda_2 \geq 0$ are fixed parameters. However in most numerical calculations the following constant are set to $\lambda_1 = \lambda_2 = 1$, $\nu = 0$ and μ is arbitrary.

Level Set Formulation of the Chan-Vese Model

In the level set method (Osher and Sethian, 1988), C Ă Ω is represented by the zero level set of a Lipschitz function φ : Ω Ñ R, such that

$$\begin{cases} C \text{ “ B}\omega \text{ “ tp}x,y\text{q P } \Omega : \varphi\text{p}x,y\text{q “ 0u, inside}p C\text{q “ } \omega \text{ “ tp}x,y\text{q P } \Omega : \varphi\text{p}x,y\text{q ą 0u,} \\ \text{outside}p C\text{q “ } \Omega\text{z}\omega \text{ “ tp}x,y\text{q P } \Omega : \varphi\text{p}x,y\text{q ă 0u,} \end{cases} \quad (3.3.12)$$

For the level set formulation of our variational active contour model, the variable C is replaced by the variable φ. Using the Heaviside function H and the one-dimensional Dirac measure δ_0 as defined respectively by:

$$H\text{pz q} \text{ “ } \begin{cases} 1, & \text{if } z \text{ ě } 0 \\ 0, & \text{if } z \text{ ă } 0 \end{cases} \qquad \delta_0\text{pz q} \text{ “ } \frac{d}{dz} H\text{pz q} \quad (3.3.13)$$

the terms in the energy functional F are expressed in the following way:

$$Length\text{t}\varphi \text{ “ 0u} \text{ “ } \int_\Omega |5\, H\text{p}\varphi\text{p}x,y\text{qq}|dxdy$$

$$\text{“ } \int_\Omega \delta_0(\phi(x,y))|\bigtriangledown \phi(x,y)|dxdy \quad (3.3.14)$$

$$Area\text{t}\varphi \text{ ě 0u} \text{ “ } \int_\Omega H(\phi(x,y))dxdy \quad (3.3.15)$$

and

$$\dot{z} \quad \dot{z} \quad |u_0(x,y) - c_1|^2 dxdy \quad - \quad \dot{z} \; |u_0(x,y) - c_1|^2 H(\varphi(x,y)) dxdy \qquad (3.3.16)$$

$$\varphi \geq 0 \quad \Omega$$

$$|u_0(x,y) - c_2|^2 dxdy \quad - \quad \dot{z} \; |u_0(x,y) - c_2|^2 (1 - H(\varphi(x,y))) dxdy \qquad (3.3.17)$$

$$\varphi < 0 \quad \Omega$$

Then the energy functional $F(c_1, c_2, \varphi)$ can be written as

$$F(c_1, c_2, \varphi) = \mu \dot{z}_{\Omega} \delta(\varphi(x,y)) |\nabla \varphi(x,y)| dxdy$$

$$+ \nu \dot{z}_{\Omega} H(\varphi(x,y)) dxdy$$

$$+ \lambda_1 \dot{z}_{\Omega} |u_0(x,y) - c_1|^2 H(\varphi(x,y)) dxdy$$

$$+ \lambda_2 \dot{z}_{\Omega} |u_0(x,y) - c_2|^2 (1 - H(\varphi(x,y))) dxdy \qquad (3.3.18)$$

Keeping φ fixed and minimising the energy functional $F(c_1, c_2, \varphi)$ the constant function of φ can be expressed by:

$$c_1(\varphi) = \frac{\int_\Omega u_0(x,y)H(\phi(x,y))dxdy}{\int_\Omega H(\phi(x,y))dxdy} \quad (3.3.19)$$

if the neighborhood of x,y is in the interior of $\varphi = C$ and

$$c_2(\varphi) = \frac{\int_\Omega u_0(x,y)(1 - H(\varphi(x,y)))dxdy}{\int_\Omega (1 - H(\varphi(x,y)))dxdy} \quad (3.3.20)$$

if the neighborhood of x,y is in the exterior of $\varphi = C$. For the corresponding "degenerate" cases, there are no constrains on the values of c_1 and c_2.

Remark: By Equation (3.3.18), the energy functional can be written in terms of $H(\varphi)$. Let $\chi_\omega = H(\varphi)$, then the energy functional can be written in the new form F as:

$$\begin{aligned} F(\chi_\omega) = & \mu \int_\Omega |\nabla \chi_\omega(x,y)| dxdy + \nu \int_\Omega \chi_\omega(x,y) dxdy \\ & + \lambda_1 \int_\Omega (u_0(x,y) - c_1)^2 \chi_\omega(x,y) dxdy \\ & + \lambda_2 \int_\Omega (u_0(x,y) - c_2)^2 (1 - \chi_\omega(x,y)) dxdy \end{aligned} \quad (3.3.21)$$

Therefore, the new minimisation problem can be considered as:

$$\inf_{\chi_\omega} \mathcal{F}(\chi_\omega), \quad \chi_\omega(x,y) \in \{0,1\} \text{ a.e.} \quad (3.3.22)$$

among the characteristic functions of the level sets φ with finite perimeters in Ω. Here, L ´ *a.e.* means almost everywhere with respect to the Lebesgue measure.

The energy functional $F(c_1,c_2,\varphi)$ in Equation (3.3.18) is minimised to obtain the associated Euler Lagrange for the unknown function φ. In order to compute the associated Euler-Lagrange equation for the unknown function φ, a slightly regularised versions of the functions $H(\varphi(x,y,\varepsilon)) = H_\varepsilon$ and $\delta(\varphi(x,y,\varepsilon)) = \delta_\varepsilon$ were considered.

As $\varepsilon \to 0$, δ_ε and H_ε goes to H and δ respectively. Let H_ε be any $C^2(\overline{\Omega})$ regularisation of H and $\delta_\varepsilon = H'_\varepsilon$. Let denote by F_ε the associated regularised functional defined by:

$$\begin{aligned} F_\varepsilon(c_1,c_2,\varphi) = & \int_\Omega \delta_\varepsilon(\varphi(x,y))\,|\nabla\varphi(x,y)|\,dxdy \\ & + \nu \int_\Omega H_\varepsilon(\varphi(x,y))\,dxdy \\ & + \lambda_1 \int_\Omega |u_0(x,y) - c_1|^2 H_\varepsilon(\varphi(x,y))\,dxdy \\ & + \lambda_2 \int_\Omega |u_0(x,y) - c_2|^2 (1 - H_\varepsilon(\varphi(x,y)))\,dxdy \end{aligned} \tag{3.3.23}$$

Keeping c_1 and c_2 fixed and minimising F_ε with respect to φ, the associated Euler-Lagrange equation for φ is deduced. Parameterising the descent direction by an artificial time $t \geq 0$, the Euler Lagrange equation in $\varphi(t;x,y)$ (with $\varphi(0;x,y) = \varphi_0(x,y)$ defining the initial contour) is:

$$\begin{aligned} \frac{\partial\varphi}{\partial t} = & \delta_\varepsilon(\varphi)\left[\mu\,\mathrm{div}\left(\frac{\nabla\varphi}{|\nabla\varphi|}\right) - \nu - \lambda_1(u_0 - c_1)^2 + \lambda_2(u_0 - c_2)^2\right] \\ & \varphi(0;x,y) = \varphi_0(x,y) \end{aligned}$$

$$\frac{\delta_{\varepsilon}p\varphi q}{B\varphi \mid 5\,\bar{\varphi}\mid} \quad " \quad 0 \text{ on } B\Omega$$
$$Bn$$

$$pt;x,yq \quad P \quad p0,8q \hat{} \Omega$$

$$px,yq \quad P \quad \Omega \qquad (3.3.24)$$

3.3.5 Region Growing

Consider an image object to be a two-dimensional curved surface in 3-D space. Consequently, this can be looked at as a landscape, much in the similar way as a digital elevation of pixel value. Suppose the landscape is a homogeneous region of grayscale, binary or true colour value. Now, a pixel point of the landscape is selected and called a seed/initial pixel point of the region. Moving up pixel by pixel all pixels that are part of the homogeneous landscape are added until an edge/mountain is reached then the algorithm stops. This process of adding new pixels to the current pixel is called the prediction process. If the current pixel is on the boundary and the next predicted pixel falls outside the region then the follow conditions may apply. That is, the predicted pixel may either:

1. belong to a new region
2. contains impulse noise
3. is a mixed pixel, i.e. located at a boundary
4. belongs to a previous region

If the predicted pixel belongs to the current region that is not connected with the current pixel, then a new region is created. Instances where further prediction pixels encounter the grouped pixels of the homogeneous region, phantom regions are created. This implies the prediction procedure requires further procedure to remove these noisy clusters, elongated mixed pixels and phantom regions.

Phase 1: Prediction

The prediction procedure take into account the mean and covariance of the prediction and current pixel position. Let each examined homogeneous region R_k consisting of pixel $g_k(i,j)$. Then:

$$g_k(i,j) = \bar{g}_k(i,j) + n_k(i,j) \quad (3.3.25)$$

where $\bar{g}_k(i,j)$ is the weighted average prediction pixel in the neighborhood of the current pixel and $n_k(i,j)$ is an additive noise field. Let $n_k(i,j)$ be a signal-independent, identically-distributed, region-independent, Gaussian white noise field satisfying $n(i,j) = n_k(i,j) \sim N(0, \sigma_n^2 I)$. Then $g_k(i,j)$ is homogeneous or wise-sense stationary random field with mean value:

$$E[g_k(i,j)] = \mu_k \quad (3.3.26)$$

and the covariance function:

$$C_k(i,j,i^1,j^1) = C_k(i - i^1, j - j^1) = C_k(p,q) \quad (3.3.27)$$

being spatially invariant. Suppose that each region has the same covariance function then: $C_k(p,q) = C(p,q)$ for all k. A commonly used covariance function is the circularly symmetric or isotropic exponential function given by:

$$C(p,q) = \sigma^2 \exp\left(-\frac{\sqrt{p^2 + q^2}}{2\omega}\right) \quad (3.3.28)$$

where σ^2 represents the variance of the random field. Let $\sigma^2 = \sigma_n^2$ with σ_n^2 the variance of the additive noise field $n(i,j)$. The prediction function to obtain $\bar{g}_k(i,j)$ such that the random field has covariance function according to Equation (3.3.28) is the weighting function/predictor:

$$r(p,q) = \exp\left(-\frac{(p^2 + q^2)}{2\omega^2}\right) \qquad (3.3.29)$$

Figure 3.3 illustrate the sampling of Equation (3.3.29) for use on a regularised grid. From the graph, the function is truncated after $i - 3$ since the weights are becoming insignificantly small, g_i is estimated from g_{i-1}, g_{i-2} and g_{i-3} according to the sampled weights. The coefficients of the weight function should sum up to 1 that is $\sum_{(p,q)\in R_k} r(p,q) = 1$. Since the aim is to test whether pixel (i,j) is part of region k, the hypothesis

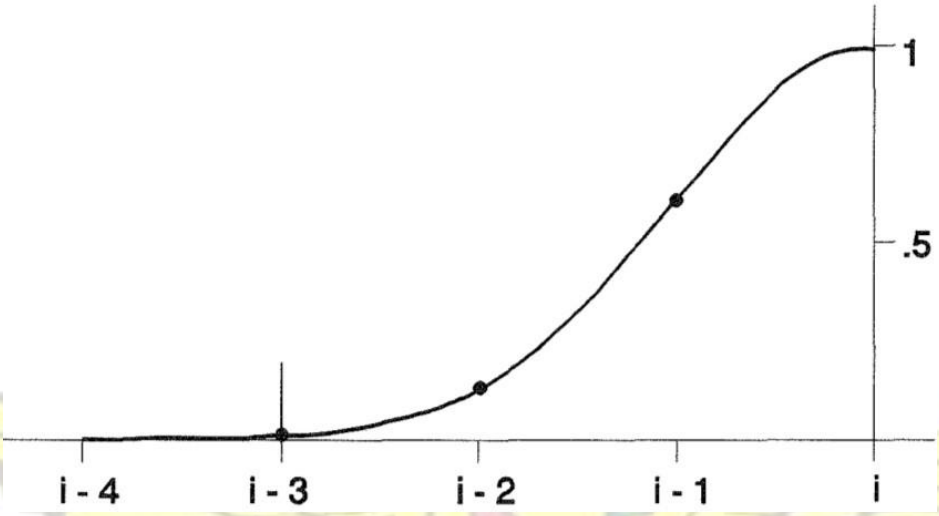


Figure 3.3: Sampling and truncation of the predictor $r(p,q) = \exp\left(-\frac{(p^2 + q^2)}{2\omega^2}\right)$ ing insignificantly small, g_i is estimated from g_{i-1}, g_{i-2} and g_{i-3} according to the sampled weights. The coefficients of the weight function should sum up to 1 that is $\sum_{(p,q)\in R_k} r(p,q) = 1$. Since the aim is to test whether pixel (i,j) is part of region k, the hypothesis

$$\Delta g_k(i,j) = |\bar{g}_k(i,j) - g_k(i,j)| = 0 \qquad (3.3.30)$$

that (i,j) is not part of region k is checked against the hypothesis

$$\Delta g_k(i,j) = |\bar{g}_k(i,j) - g_k(i,j)| \neq 0 \qquad (3.3.31)$$

Using the prediction Equation (3.3.29) under the restriction $\sum_{(p,q)\in R_k} r(p,q) = 1$, the prediction $\bar{g}_k(i,j)$ of $g_k(i,j)$, becomes:

$$\bar{g} = \frac{\sum_{(p,q)\in R_k} (p,q)\, g_k(i-p, j-q)\, r}{\sum_{k}(i,j) = \sum_{(p,q)\in R_k} r(p,q)} \qquad (3.3.32)$$

with prediction variance:

$$\sigma_n^2(i,j) = \frac{\sum_{p,q \in R_k} (p,q)}{r(p,q)^2} \qquad (3.3.33)$$

and the variance of the estimate $g_k(i,j)$ becomes:

$$\hat{\sigma}_k^2(i,j) = \sigma_n^2(i,j) \frac{\sum_{p,q \in R_k} r^2(p,q)}{\left(1 - \sum_{p,q \in R} r(p,q)\right)^2} \qquad (3.3.34)$$

since $r(0,0) = 1$ according to Equation (3.3.29).

The standard z-score is applied to test whether the hypothesis that pixel (i,j) is part of region R_k which leads to the null-hypothesis $g_k(i,j) = 0$

$$z = \frac{g_k(i,j)}{\hat{\sigma}_k(i,j)} \qquad (3.3.35)$$

where $g_k(i,j) = |\bar{g}_k(i,j) - g_k(i,j)|$ and $\hat{\sigma}_k^2(i,j)$ as defined in Equation (3.3.34). So, the final decision rule becomes: IF

$$\min_{k \in \mathcal{U}} (\triangle g_k(i,j)) \leq z_\alpha \sigma_n \qquad (3.3.36)$$

THEN assign pixel (i,j) to region R_k for which $g_k(i,j)$ is minimum

ELSE start a new region.

$\mathcal{U}$ is the set of adjacent regions of pixel (i,j), α is the probability that pixel (i,j) is wrongly assigned to region R_k and

$$\frac{}{\left(1 - \sum_{p,q \in R_k} r(p,q)\right)^2}$$

$$\ddot{y} \quad \text{“} \quad 1 \grave{} \quad {}_{p,q \text{P} R} r\text{p}p,q\text{qq}_2 \quad (3.3.37)\ \text{p}$$

rk

Phase II: Merging

The result of Phase I is that, homogeneous regions are created. Although Phase I tends to trace edges at the correct location, phantom edges are introduced, due to dependency on the scan direction of the predictor. Furthermore, noisy image and mixed pixels boundary regions in Phase I are traced as homogeneous regions. The aim of Phase II, is to remove:

1. phantom regions
2. small/noisy regions and
3. mixed pixels boundary regions

To remove a region is to merge the region with another region.

Phase IIa: Removal of Phantom Regions

Statistical formulation as to whether two adjacent regions R_k and R_l are homogeneous requires information about the means and variances of R_k and R_l. For each of the adjacent region from Phase I, the variance of the noise is estimated and the z-score replaced by the t-score.

To determine the t-score, let R_k and R_l be the two adjacent and homogeneous regions. Let pμ_k,σ_k^2q and pμ_l,σ_l^2q be the mean grey values and variances of R_k and R_l, respectively:

$$\mu_{\text{p}\ddot{}\text{q}} \text{“} \quad \frac{1}{n_{\text{p}\ddot{}\text{q}\ i\text{P}R}} \ddot{y}_{\text{p}\ddot{}\text{q}} g_i ; \qquad \sigma_{\text{p}\ddot{}\text{q}2} \text{“} \quad \frac{1}{n_{\text{p}\ddot{}\text{q}\ i\text{P}R_{\text{p}\ddot{}\text{q}}}} \ddot{y}\ \text{p}\mu_{\text{p}\ddot{}\text{q}} \acute{} g_i\text{q}_2 \qquad (3.3.38)$$

where p¨q " *k,l*. Then the *t*-score becomes:

$$t = \frac{\mu_{kl} - \mu_{kl}(hyp)}{\hat{\sigma}_{kl}} \quad (3.3.39)$$

where:

μ_{kl} " $|\mu_k - \mu_l|$ μ_{kl}p*hyp*q "

0

$\hat{\sigma}^2_{kl} = \hat{\sigma}^2_p/n_k + \hat{\sigma}^2_p/n_l = \hat{\sigma}^2_p(n_k + n_l)/n_k n_l$ where $\hat{\sigma}^2_p$, the pooled variance is obtained by a weighted averaging of the variance estimates of the two regions R_k and R_l with the weights based on their respective degrees of freedom $n_k - 1$ and p$n_l - 1$q:

$$\hat{\sigma}^2_p = \frac{(n_k - 1)\hat{\sigma}^2_{k2} + (n_l - 1)\hat{\sigma}^2_{l2}}{(n_k - 1) + (n_l - 1)} \quad (3.3.40)$$

By assuming that n_k and n_l are large: $n_k \approx n_k - 1$ and $n_l \approx n_l - 1$ then

$$\hat{\sigma}^2_{kl} = \frac{\hat{\sigma}^2_k}{n_l} + \frac{\hat{\sigma}^2_l}{n_k} \quad (3.3.41)$$

The final decision rule becomes: IF

$$|\mu_k - \mu_l| > t_{\alpha,\nu} \sqrt{\frac{\hat{\sigma}^2_k}{n_l} + \frac{\hat{\sigma}^2_l}{n_k}} \quad (3.3.42)$$

THEN merge region R_k and R_l.

where α is the probability that two similar regions are wrongly not merged and

? ?

$\nu = n_k + n_l - 2$ is the degrees of freedom of the *t*-score. To avoid a situation where, similar regions will not be merged, α should be rather small.

Phase IIb: Removal of Small/Noisy Regions

The aim is simply to remove regions due to small noisy clusters. The *t*-score is used to join two similar region according to Equation (3.3.42).

Phase IIc: Removal of Mixed Regions

The aim of Phase IIc, is to remove elongated regions which are due to mixed pixel values at the region borders. Removal of mixed regions solely based on defined width of the border is insufficient to perform this task. Instead the use of an appropriate measure to describe the significance of a silver polygon is the quotient of the area size, A, of the region and the standard deviation of the area (σ_A). The quotient is the z-score. According to a one-sided z-test at 97.5% confidence level, the z-test statistic for merging two regions is: $A \geq 1.96\sigma_A$.

To determine σ_A^2 , it was further shown that, if all co-ordinates are assumed to be uncorrelated then:

$$\sigma_{A2} = 4\sigma_c^2 \sum_{i=1}^{n} \left[(y_{i+1} - y_i)^2 + (x_{i+1} - x_i)^2 \right] \qquad (3.3.43)$$

where $(x_i, y_i), i = 1, \ldots, n$ are the coordinates of the border pixels of the region and σ_c^2 the variance of the coordinates of the border pixels, which should be known a priori.

3.4 Thresholding

Thresholding is used to separate out the regions of the image corresponding to the objects of interest from the regions of the image that correspond to the background. According to Marion (1991); Nixon and Aguado (2002), threshloding make

segmentation easier. Thresholding enables one to see what areas of an image consist of pixels whose values lie within a specified range of intensities. In general, the end result of threshloding is segmentation which is performed either on a noisy image to remove the noise or on a noise free image to segment an object.

The input to a thresholding operation is typically a gray-scale or colour image. In its implementation, the output is a binary representation of the object in the image. Black has 0 pixel values which correspond to the background while white has 1 pixel values and that correspond to the foreground. This image separation is determined by a single parameter known as the intensity threshold. Each pixel in the image is compared with this threshold. If the pixel value is higher than the threshold, the pixel value is set to white (1) in the output or otherwise it is set to black (0). Multiple threshold values can be specified for different region of the image so that an intensity value can be set to white whiles everything else is set to black for each region. However, each region will give a binary image pattern. In a colour or multi-spectral image, it is possible to define a function using the RGB channel as arguments and set threshold for the function value in order to have the binary image.

Not all images can be clearly segmented into foreground and background using simple thresholding. Using simple threshloding is determine by using intensity histogram of the image. If it is possible to separate out the foreground of an image on the basis of pixel intensity, then the pixels within the foreground must be distinctly different from the intensity of pixels within the background. In this case, a distinct peak value is expected in the histogram corresponding to the foreground such that the threshold can be chosen to isolate this peak accordingly. If such a peak does not exist, then it is unlikely that simple thresholding will produce a good segmentation. In this case adaptive thresholding which takes into account the properties of the entire image may be a better option to be considered. There are two essential

thresholding techniques: hard thresholding and soft thresholding. Hard thresholding furthermore, soft thresholding are denoted as the accompanying:

1. By definition, the hard threshold operation is given by:

$$D(U,\lambda) = \begin{cases} U & \text{If } |U| \geq \lambda \\ 0 & \text{otherwise} \end{cases} \quad (3.4.1)$$

2. By definition the soft threshold operation is given by:

$$D(U,\lambda) = \text{sign}(U) \max(0, |U| - \lambda) \quad (3.4.2)$$

The soft threshold value guideline is generally picked over hard thresholding. The shrinkage rules are used to obtain a threshold value as described below.

3.4.1 Universal Shrink (US)

This threshloding technique is defined as:

$$\lambda = \sigma\sqrt{2 \log(N)} \quad (3.4.3)$$

where N being the signal length of pixel and σ the noise variance. This method gives a superior assessment for the soft threshold if the pixel distribution is extensive. On the contrary, it tends to over smoothen the signal, accordingly losing a few points of interest of the unique signal, which bring about an increased estimation error.

3.4.2 Minimax Shrink (MS)

In this approach, the threshold value is computed utilizing minmax rule. The minimax estimator is the one that realises the minimum of the maximum Mean Square Error gotten for the image pixel value. The minimax threshold is calculated by the mathematical statement beneath where M is the signal length.

$$\lambda \text{“ } 0.394 \text{ ` } 0.264\log p M q \tag{3.4.4}$$

This has the upside of giving great prescient performance.

3.4.3 Sure Shrink (SS)

Beyond any doubt Sure Shrink is a thresholding strategy which utilize the sub band adaptive threshold. A different threshold is calculated for every point of interest in the sub band based upon SURE (Steins Unbiased Risk Estimator). It is a mix of the universal threshold and the SURE estimator. Mathematically, this is characterized as:

$$\lambda_i \text{“ } \min p t, \sigma a 2 \text{ ¨ } \overline{\log p M q q} \tag{3.4.5}$$

where M is number of discrete transform coefficients in the i^{th} subband, t represent the quality that minimizes Steins Unbiased Risk Estimator. σ is the noise variance. The SURE shrink has yields great image denoising performance and approaches the genuine Mean Square Error of the ideal soft threshold estimator. It is utilized in threshloding images in the frequency domain.

3.4.4 Bayes Shrink (BS)

This shrinkage type is an adaptive information driven threshloding technique for image denoising by means of soft thresholding in the frequency domain. The threshold is driven in a Bayesian structure and it is assumed to generalise the Gaussian distribution for the transform coefficients in every point of interest of the sub band. In the end, it tries to discover the threshold value which minimizes the Bayesian Risk. Mathematically, it is computed as:

$$\sigma_y^2 = \sigma_x^2 + \sigma^2 \tag{3.4.6}$$

σ_y^2 is the variance of noisy image

σ_x^2 is the variance of original image

σ^2 is the noise variance

$$\sigma_y^2 = \frac{1}{M}\sum_{m=1}^{M} B_m^2 \tag{3.4.7}$$

where B_m are the coefficients obtained from the frequency domain at each scale, M is the aggregate number of subband coefficients.

$$\sigma_x^2 \text{“ bmaxp}\sigma_y^2 \text{ ´ } \sigma^2\text{,0q} \tag{3.4.8}$$

where σ “ $\frac{\text{median}_{\text{p}d_{ij}\text{q}}}{0.6745}$

$$\lambda = \frac{\sigma^2}{\sigma_x} \tag{3.4.9}$$

The reconstruction using Bayes Shrink is smoother and more visually appealing than one obtained using Sure Shrink.

3.4.5 Normal Shrink (NS)

The Normal Shrink strategy is computationally more effective and versatile in light of the fact that the parameters required for evaluating the threshold relies on upon the subband information. This quantity is mathematically defined as:

$$\lambda = \beta\frac{\sigma^2}{\sigma_y} \tag{3.4.10}$$

where β is a scale parameter defined as:

d

L

$$\beta = \log \frac{L_k}{J} \quad (3.4.11)$$

L_k is the length of the subband at the k^{th} decomposition and J is the aggregate number of decompositions. The efficiency of the Normal Shrink is similar to that of Bayes Shrink, however, the normal shrink conserve edges more appropriately compared to that of the Bayes shrink. This shrinkage technique additionally apply to frequency domain only.

3.5 Fractional Calculus

The main objects of classical calculus are derivatives and integrals of functions. These two operations are inverse to each other. Given a function $f(t)$ and keeping its derivatives on the left-hand side and its integrals on the right-hand side, the following infinite sequence is obtained:

$$\cdots \quad \frac{d^2 f(t)}{dt^2}, \frac{df(t)}{dt}, f(t), \int_a^t f(\tau)d\tau, \int_a^t \int_a^{\tau_1} f(\tau)d\tau d\tau \quad \cdots \quad (3.5.1)$$

Fractional calculus tries to interpolate this sequence so as to unify the classical derivative and integral operations and generalise them to an arbitrary order. There are many ways this operator can be defined mathematically and they are mostly named according to their authors. For example, the Grünwald-Letnikov definition of differintegral looks at the infinitesimal division and limit of the classical definition of derivative and integrals. This definition has a disadvantage of technical difficulty in its computation and the restriction on the choice of functions. However, there are other variant like Riemann-Liouville definition. As far as this work is concern, attention will be focused on the Riemann-Liouville, Caputo and the Miller-Ross

definitions with some preliminary concepts (special function and convolution) as they are the most widely used ones in applications.

3.5.1 Special Functions

The generalisation of classical calculus to fractional calculus is mostly accomplished with the knowledge of some special functions which are used in deriving important results of fractional calculus. For this reason, an overview of the definition and properties of these selected special functions used in fractional calculus is given as follows.

Euler's Gamma function

Euler Gamma function is an important function in the study of calculus. It can be encountered in nearly all part in the discipline of Fractional Calculus and furthermore many special functions can be expressed in terms of this function. Euler Gamma function can be defined in a number of ways, however, the below definition will be adopted.

Definition 3.5.1. For $z \in \mathbb{C}\setminus\{0,-1,-2,-3,\ldots\}$, Euler's Gamma function $\Gamma(z)$ is defined as:

$$\Gamma(z) = \int_0^{\infty} t^{z-1} e^{-t}\, dt, \quad \text{if} \quad \mathrm{Re}(z) > 0 \tag{3.5.2}$$

This function is defined in the whole complex plane except zero and negative integers. At this exceptional value, the Gamma function experience poles as shown in Figure 3.4.

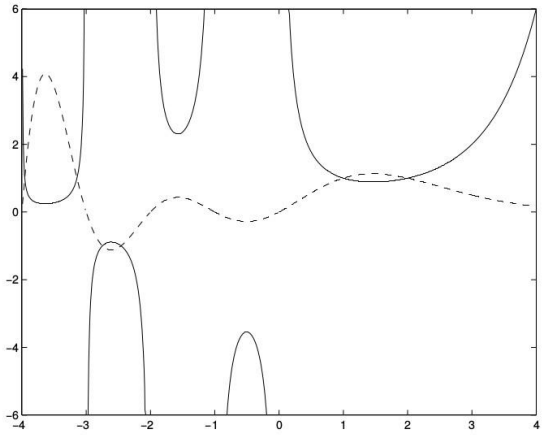


Figure 3.4: Euler's Gamma function Γpzq (solid) and its reciprocal (dashed)

Theorem 3.5.1. *The Euler's Gamma function satisfy the following properties:*

1. *For Re*pzq ą 0 *then:*

$$\Gamma(z) = \int_0^1 \left(\ln \frac{1}{t}\right)^{z-1} dt \qquad (3.5.3)$$

2. *For z* P Czt0,´1,´2,´3,...u

$$\Gamma(z+1) = z\Gamma(z) \qquad (3.5.4)$$

3. *For n* P N

$$\Gamma(n) = (n-1)! \qquad (3.5.5)$$

$$0! = 1$$

4. *For z* P Czt0,1,2,3,...u

$$\Gamma(1-z) = -z\Gamma(-z) \qquad (3.5.6)$$

*5. (Limit representation) For Re*pzq ą 0 *the following limit holds:*

$$\Gamma(z) = \lim_{n\to\infty} \frac{n!n^z}{z(z+1)\cdots(z+n)} \quad (3.5.7)$$

The Limit representation is equivalent to Euler's infinite product, given by

$$\frac{1}{z\Gamma(z)} = e^{\gamma z}\prod_{n=1}^{\infty}\left(1+\frac{z}{n}\right)e^{-z/n} \quad (3.5.8)$$

6. *(Weierstrass definition) Let z* P Czt0,´1,´2,´3,...u*. Then Euler's Gamma function can be defined by*

$$\frac{1}{\Gamma(z)} = ze^{\gamma z}\prod_{n=1}^{\infty}\left(1+\frac{z}{n}\right)e^{-z/n} \quad (3.5.9)$$

where γ is the Euler constant given by:

$$\gamma = \lim_{n\to\infty}\left(\sum_{k=1}^{n}\frac{1}{k} - \ln(n)\right) \approx 0.5772156649 \quad (3.5.10)$$

7. *Euler's Gamma function is analytic for all z* P Czt0,´1,´2,´3,...u

8. *Euler's Gamma function is never zero.*

9. *(Reflection Theorem) For all non-integer z* P C

$$\pi \qquad \pi$$

$$\Gamma(z)\Gamma(1-z) = \frac{\pi}{\sin(\pi z)}, \quad \textit{and} \quad \Gamma(z)\Gamma(-z) = -\frac{\pi}{z\sin(\pi z)} \tag{3.5.11}$$

10. For half-integer arguments, $\Gamma(n+\frac{1}{2})$, $n \in \mathbb{N}$ *has the special form*

$$\Gamma\left(n+\frac{1}{2}\right) = \frac{(2n-1)!!}{2^n}\sqrt{\pi} \tag{3.5.12}$$

where $n!!$ *is the double factorial defined as:*

$$n!! = \begin{cases} n(n-2)(n-4)\cdots 5\cdot 3\cdot 1 & n > 0 \text{ odd} \\ n(n-2)(n-4)\cdots 6\cdot 4\cdot 2 & n > 0 \text{ even} \\ 1 & n = 0,-1 \end{cases} \tag{3.5.13}$$

Directly connected to Euler's Gamma function is the definition of the generalised binomial coefficients:

Definition 3.5.2. The generalised binomial coefficients $\binom{\alpha}{k}$ for $\alpha \in \mathbb{R}$ and $k \in \mathbb{N}$ are defined by

$$\binom{\alpha}{k} = \frac{\alpha(\alpha-1)(\alpha-2)\cdots(\alpha-k+1)}{k!} \tag{3.5.14}$$

With the help of Euler's Gamma function, a number of additional special functions useful in the generalisation of the solution of classical calculus to its fractional counterpart can be defined.

Beta function

A special function which is connected to the Euler's Gamma function in a direct way, is given by the Beta function defined as follows:

Definition 3.5.3. The Beta function *B*pz,*w*q in two variables *z,w* P C is defined by:

$$B(z,w) = \frac{\Gamma(z)\Gamma(w)}{\Gamma(z+w)} \tag{3.5.15}$$

Theorem 3.5.2. *The Beta function possesses the following properties:*

1. *For Re*pzq, *Re*pwq ą 0, *the Equation (3.5.15) is equivalent to:*

$$B(z,w) = \int_0^1 t^{z-1}(1-t)^{w-1}dt = \int_0^\infty \frac{t^{z-1}}{(1+t)^{z+w}}dt$$

$$B(z,w) = 2\int_0^{\pi/2} \sin^{2z-1}(t)\cos^{2w-1}(t)dt \tag{3.5.16}$$

2. *B*pz ` 1,*w* ` 1q *is the solution of the Beta Integral:*

$$\int_0^1 t^z(1-t)^w dt = B(z+1, w+1) = \frac{\Gamma(z+1)\Gamma(w+1)}{\Gamma(z+w+2)} \tag{3.5.17}$$

3. *The following identities hold:*

   (a) *B*pz,*w*q " *B*p*w*,*z*q,

   (b) *B*pz,*w*q " *B*pz ` 1,*w*q ` *B*pz,*w* ` 1q,

   (c) *B*pz,*w* ` 1q "$\frac{w}{z}B(z+1,w) = \frac{w}{z+w}B(z,w)$.

From a more analytic point of view, the Mittag-Leffler function, which also is strongly connected to Euler's Gamma function will be defined next.

Mittag-Leffler function

Definition 3.5.4. For *z* P C, the single parametric Mittag-Leffler function $E_\alpha$pzq is defined by:

$$E_\alpha(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(\alpha k+1)}, \quad \alpha > 0 \tag{3.5.18}$$

and the generalised Mittag-Leffler Function $E_{\alpha,\beta}$pzq is given by:

$$E_{\alpha,\beta}(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(\alpha k+\beta)}, \quad \alpha,\beta > 0 \tag{3.5.19}$$

In the following theorem, the properties of the Mittag-Leffler function are stated.

Theorem 3.5.3. *The Mittag-Leffler function possesses the following properties:*

1. *For* |*z*| ă 1*, the generalised Mittag-Leffler function satisfies:*

$$\dot{z}_0^{\infty} e^{-t} t^{\beta-1} E_{\alpha,\beta}\ t^{\alpha} z\ dt \quad \text{p} \quad \text{q} \quad \text{“} \frac{1}{z-1} \tag{3.5.20}$$

2. *For* |*z*| ă 1*, the Laplace transform of the Mittag-Leffler function* E_{α}pz$^{\alpha}$q *is given by:*

$$\dot{z}_0^{\infty} e^{-zt} E\ \ z^{\alpha}\ dt \quad {}_{\alpha}\text{p} \quad \text{q} \quad \text{“} \frac{1}{z^{\prime} z^{1^{\prime}\alpha}} \tag{3.5.21}$$

3. *The Mittag-Leffler function in Equation (3.5.18) converges for every z* P C*.*

4. *For special values of α the Mittag-Leffler function is given by:*

   (a) $E_0 = \frac{1}{z-1}$ *(b)* E_1pzq “ e^z

   (c) E_2pz^{2}q “ coshpzq *(d)* E_2p$^{\prime}z^2$q “ cospzq

From property 4(b) and (d), the Mittag-Leffler function of α “ 0.25 to α “ 2 as shown in Figure 3.5 is plotted. The Mittag-Leffler function is related to the

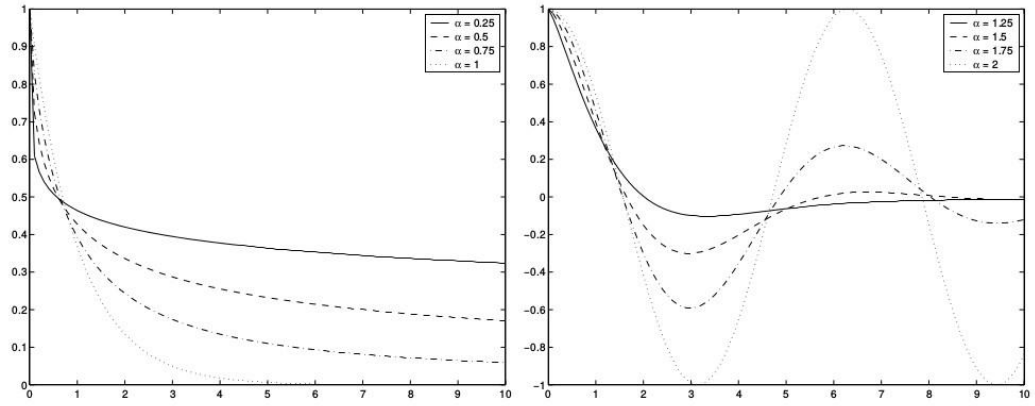


Figure 3.5: Mittag-Leffler Function E_{α}p$^{\prime}z^{\alpha}$q for different values of α.

generalised hyper-geometric function $F_{n,r}^{\beta}$ pzq defined as:

$$F_{n,r}^{\beta}(z) := \sum_{k=0}^{\infty} \frac{\beta^k}{(nk+r)!} z^{nk+r} \qquad (3.5.22)$$

and to the confluent hyper-geometric function $p_1F_1pa;b;zqq$, also known as Kummer's function of the first kind and defined as:

$$_1F_1pa;b;zq := \sum_{k=0}^{\infty} \frac{\Gamma pa+kq}{\Gamma paq} \frac{\Gamma pbq}{\Gamma pb+kq} \frac{z^k}{k!} \qquad (3.5.23)$$

3.5.2 Convolution

Convolution is the integral or summation of two component functions. It measures the amount of overlap as one function is shifted over the other and is indicated by the operator "˚". In one dimension, if *f* and *g* are functions of *t*, then the convolution of *f* and *g* over an infinite interval is the integral given by:

$$f \circ g = \int_{-\infty}^{\infty} f(\tau) g(t-\tau) d\tau \qquad (3.5.24)$$

If the convolution is performed over a finite range r0,*t*s, then the convolution is:

$$rf \circ gsptq = \int_0^t f(\tau) g(t-\tau) d\tau \qquad (3.5.25)$$

Moreover, if *frn*s and *grn*s are functions with respect to a single discrete point *n*, then convolution takes the following form:

$$Arn_a s \circ Brn_b s = Crn_c s = \sum_{\tau=0}^{n_a-1} Ar\tau sBrn_c - \tau s \qquad (3.5.26)$$

where 0 ď n_c ă n_a ` n_b ´ 1.

The one-dimensional versions are given here as they are the most simple representation of the convolution operation. However, the 2-D extension of 1-D discrete convolution is given in Equation by:

$$Ari_a,j_as \circ Bri_b,j_bs \text{“} Cri_c,j_cs \text{“} \ddot{y}_{a1} \ddot{y}_{a2} Ar\tau_1,\tau_2 sBri \acute{} \tau_1,j \acute{} \tau_2 s \quad (3.5.27)$$

$$\tau \text{“} 0 \; \tau \text{“} 0$$

Properties of Convolution

1. *Commutative Property:* The commutative property for convolution is expressed in mathematical form as:

$$arn_is \circ brn_js \text{“} brn_js \circ arn_is \quad (3.5.28)$$

2. *Associative Property:* The associative property provides that the order of the convolutions doesn't matter and is expressed as:

$$parn_is \circ brn_jsq \circ crn_ks \text{“} arn_is \circ pbrn_jsq \circ crn_ksq \quad (3.5.29)$$

3. *Distributive Property:* In equation form, the distributive property is written as:

$$arn_is \circ brn_js \grave{} arn_is \circ crn_ks \text{“} arn_is \circ pbrn_js \grave{} crn_ksq \quad (3.5.30)$$

4. *Differentiation Property:*

$$\frac{B}{Bx}pa \circ bq \text{“} \frac{Ba}{Bx} \circ b \quad (3.5.31)$$

That is:

$$\dot{z}_1^1 \frac{BB}{Bt_0} fp\tau qgpt \acute{} \tau qdt \text{“} \dot{z}_1^1 \frac{BB}{Bt_0} gp\tau qfpt \acute{} \tau qdt \quad (3.5.32)$$

3.5.3 The Riemann-Liouville Differintegral

Based on the Cauchy iterative integral formula in Equation (3.5.33) which provide a good basis of generalisation, the Riemann-Liouville differintegral is defined.

ż $I_a^n f(t$ ..... $\int_a^t \int_a^{\tau_1} \cdots \int_a^{\tau_{n-1}} f(\tau)\, d\tau\, d\tau_1 \cdots d\tau_{n-1}$ q "ż ..... ż p q

n a a

" $\frac{1}{(n-1)!} \int_a^t (t-\tau)^{n-1} f(\tau)\, d\tau$ (3.5.33) pn ´ 1q!

Proof. By mathematical induction, Equation (3.5.33) can be proven. The case of $n = 1$ is obviously fulfilled, so the case $n = 2$ is demonstrated in this way. Substituting $n = 2$ and computing Equation (3.5.33) gives:

1 $u = t - \tau$ $u = -1$

$\int_a^t (t-\tau) f(\tau)\, d\tau$ $v' = f(\tau)$ $v = \int_a^\tau f(r)\, dr$ $(t-\tau) f(\tau) d\tau$ "

1 ˇˇˇˇ şż ż ˇˇˇˇ

$1! \int_a^t \left(\int_a^\tau f(r)\, dr \right) d\tau$

"

$\int_a^t (t-\tau) f(\tau)\, d\tau + \int_a^t \int_a^\tau f(r)\, dr\, d\tau$

a $\tau = a$ a a $\tau = t$ t

" I_a^2 (3.5.34)

Now, suppose the equation holds for n and integrating again for the second time, the following expression is obtained:

$$\dot{z}_a I_a^n f(r)dr \text{ “ } \dot{z}_a^t \frac{}{} \dot{z}_a^r (r-\tau)^{n-1} f(\tau)d\tau dr$$

$$= \frac{1}{(n-1)!}\int_a^t \left[\int_a^t (r-\tau)^{n-1} f(\tau)\,d\tau\right] dr \text{ “ } \frac{1}{(n-1)!}\int_a^t \left[\int_\tau^t (r-\tau)^{n-1} f(\tau)\,dr\right] d\tau$$

$$= \frac{1}{(n-1)!}\int_a^t f(\tau)\left[\frac{(r-\tau)^n}{n}\right]_\tau^t d\tau$$

$$= \frac{1}{(n-1)!}\int_a^t (t-\tau)^n f(\tau)\,d\tau$$

$$\text{“ } \frac{1}{(n-1)!}\int_a^t (t-\tau)^n f(\tau)d\tau = I_a^{n+1} f(t) \qquad (3.5.35)$$

This completes the proof of the Cauchy formula. □

Substituting a positive real number α for the interest n and using the Gamma function instead of the factorial, the Cauchy formula is generalised as:

$$I_a^\alpha f(t) \text{ “ } \frac{1}{\Gamma(\alpha)}\int_a^t (t-\tau)^{\alpha-1} f(\tau)d\tau \qquad (3.5.36)$$

This integral is only valid for arbitrary order $\alpha > 0$ and does not hold for $\alpha = 0$ which formally corresponds to the identity operator. However, by assuming the limit for $\alpha \to 0$, the above definition can be extended by setting:

$$I_a^0 f(t) = f(t) \qquad (3.5.37)$$

Unlike the fractional integral, there are n^{th} derivative analogous to Equation (3.5.33). This is derived through the fractional integral by perturbing the integer order and applying an appropriate number of classical derivative. The result of this idea is the following for α ą 0:

$$D_a^{\alpha} f(t_{\mathsf{q}} \quad {}_{dt^n}[I_a^{n-\alpha} f(t \quad \text{“} \_\_d_n \text{qs “} \_\_\_\_\_ \_\_1\ dtd_{nn} \dot{z}_t \text{p}t' \tau \text{q}_{n'\alpha'1} f\text{p}\tau\text{q}d\tau$$

(3.5.38) Γp$n'\alpha$q $_a$

where n “ rαs`1. If $\alpha$ “ $k$ and $k$ P N$_0$ then $n$ “ $k$`1 and the following is obtained:

$$D_a^k f(t \quad \text{q “} \_\_\_\_ \_\_\_\__{k`1}\, \dot{z}_a \overset{f\ \tau\ d\tau}{\_\_\_\_\_} \text{p q “} \_\_\_\_\_ dt^k \text{ (3.5.39) } 1\ d_{k`1\ t}$$

$d_k f\text{p}t\text{q}$

Γp1q dt

Putting $D_a^{-\alpha} = I_a^{\alpha}$ with $f^{\text{p0q}}\text{p}t\text{q}$ “ $f\text{p}t\text{q}$, both the fractional integral and derivative can be written in on expression to formulate the definition of the Riemann-Liouville differintegral.

Definition 3.5.5 (The Riemann-Liouville differintegral)**.** Let a, T, α be real constants where a ă T, n “ maxp0,rαs ` 1q and $f\text{p}t\text{q}$ an integrable function on xa,Tq. For n ą 0, additionally assume that $f\text{p}t\text{q}$ is n-times differentiable on xa,Tq except on a set of measure zero. Then the Riemann-Liouville differintegral is defined for t P xa,Tq as:

$$^{RL}D_a^{\alpha} f(t_{\mathsf{q}} \text{ “} \_\_\_\_\_ \_\_\_1\ d_{nn} \dot{z}_t \text{p}t' \tau \text{q}^{n'\alpha'1} f\text{p}\tau\text{q}d\tau \text{ (3.5.40) } \Gamma\text{p}n' \alpha\text{q}\ dt$$

$_a$

where $^{RL}D_a^{\alpha}$ is the Riemann-Liouville differintegral operator.

3.5.4 The Caputo Differintegral

The fractional integral of Caputo is given by the same expression like before, hence for $\alpha \geq 0$, the following is obtained:

$$ {}^{C}D_a^{-\alpha} f(t) = {}^{RL}D_a^{-\alpha} f(t) \tag{3.5.41} $$

The difference occurs for fractional derivative. A non-integer-order derivative is again defined by the help of the fractional integral, but here, the function $f(t)$ is first differentiated and then later integrated using the fractional integral to the required order. This idea leads to the definition of the Caputo differintegral.

Definition 3.5.6 (The Caputo differintegral)**.** Let a, T, α be real constants with $a < T$ and $n_c = \max(0, \lceil \alpha \rceil)$ and $f(t)$ a function which is integrable on $[a, T]$ in case $n_c = 0$ and n_c-times differentiable on $[a, T]$ except on a set of measure zero where $n_c \geq 0$. Then the Caputo differintegral is defined for $t \in [a, T]$ as:

$$ {}_{c}D_a^{\alpha} f(t) = I_a^{n_c - \alpha} \frac{d^{n_c}}{dt^{n_c}} f(t) = \frac{1}{\Gamma(n_c - \alpha)} \int_a^t (t - \tau)^{n_c - \alpha - 1} f^{(n_c)}(\tau)\, d\tau \tag{3.5.42} $$

For $\alpha \geq 0$, $\alpha \notin \mathbb{N}_0$, Equation (3.5.42) is often written in the form:

$$ {}^{C}D_a^{\alpha} f(t) = \frac{1}{\Gamma(n_c - \alpha)} \int_a^t (t - \tau)^{n_c - \alpha - 1} f^{(n_c)}(\tau)\, d\tau \tag{3.5.43} $$

The difference between the values of n and n_c is for integers as can be seen in Figures (3.6) and (3.7) however, both cases coincide with classical derivatives at those points.

Clearly, the Caputo derivative can also be written by the help of the fractional integral of Riemann-Liouville as: _____ (3.5.44)

$$ {}^{C}D_a^{\alpha} f(t) = D_a^{-(n_c-\alpha)} \quad \frac{d^{n_c} f(}{dt^{n_c}} $$

tq

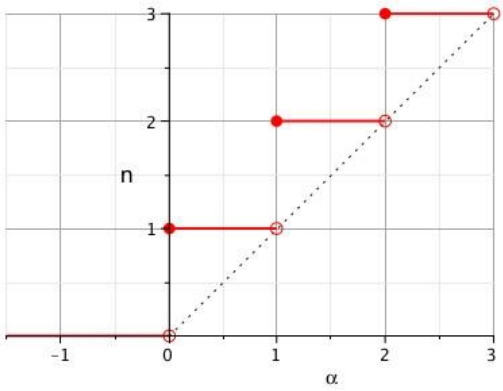


Figure 3.6: Function *n* " r*α*s ` 1 used for the Riemann-Liouville derivative.

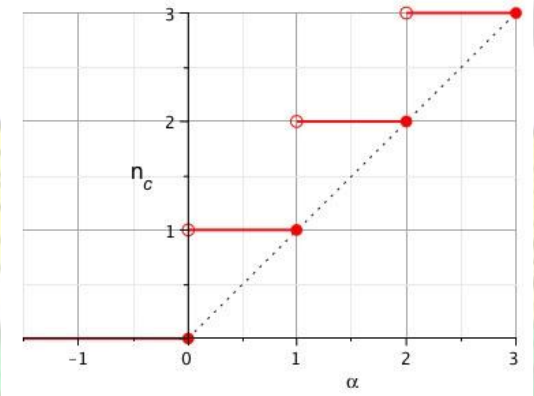


Figure 3.7: Function *n* " ´r´*α*s used for the Caputo derivative.

In practice, the Caputo definition is mostly preferred since the Riemann-Liouville approach requires the initial conditions for differential equations in terms of noninteger derivatives which hardly have physical interpretation whereas the Caputo approach uses the integer-order initial conditions.

3.5.5 Sequential Fractional Derivatives

The sequential fractional derivatives is mostly known as the Miller-Ross fractional derivative. The main idea of this type of derivative arises from the method of how the

derivatives of higher orders are defined. In order to obtain the n^{th} derivative, the Riemann-Liouville derivative is n times given as:

$$ {}^{S}D_a^{\sigma} f(t_{\mathsf{q}} \quad \text{„} \; {}^{RL}D_a^{-\alpha_0} \; {}^{RL}D_a^{\alpha_n} \; {}^{RL}D_a^{\alpha_{n-1}} \ldots\ldots {}^{RL}D_a^{\alpha_1} f(t_{\mathsf{q}} \qquad (3.5.45) $$

where

$$ \sigma \text{“ ÿ } \alpha_j \text{´ } \alpha_0, \qquad \text{and } \alpha_j \text{ P p0,1y, } \alpha_0 \text{ P x0,1q} \qquad (3.5.46) $$

Apparently it is possible to look at the Riemann-Liouville and Caputo fractional derivatives in a sequential language. The natural sequence for the Riemann-Liouville derivative is:

$$ {}^{RL}D_a^{\alpha} f(t_{\mathsf{q}} \text{“} \; \underbrace{d \; d \; \ldots \; d}_{dt_{\text{oo mo oo}} dt_{\mathsf{n}}} \; {}^{RL}D_{\alpha}^{-(n-\alpha)} f(t \; \_\_ \; \_\_ \text{ q (3.5.47) lo } \overset{dt}{}_{n} $$

which disagrees with the conditions on the orders in sequence as the last term is an integral. In practice, this operator is written in its equivalent way as:

$$ {}^{RL}D_a^{\alpha} f(t \qquad \underbrace{\overline{dt}\,\overline{dt} \quad \overline{dt}}_{n-1} \; {}^{RL}D_{\alpha}^{\alpha-n+1} f(t \quad d\,d \quad d \qquad \text{q “ } \ldots\ldots \text{q} \qquad (3.5.48) $$

The sequence for the Caputo fractional derivative also uses integer-order derivatives but at the other side. Here the integral is placed first in the sequence and hence there is no problem with this definition:

$$ {}^{C}D_a^{\alpha} f(t) = D_a^{-(n-\alpha)} \quad \ldots \quad f(t \; \underbrace{dt\,dt \quad dt}_{n=n_c} \qquad \text{q} \qquad \frac{d\,d}{\_\_} \quad \frac{d}{\_\_} \qquad (3.5.49) $$

In practice the sequential derivative appears naturally when one expression containing a derivative is substituted into another.

3.5.6 Basic Properties of Differintegrals

This section discusses the linearity and composition of fractional derivative and internal as well as the equivalence and continuity with respect to the order of the derivative.

Linearity

Let λ, μ, α and a be real constants and $f$pt$\mathsf{q}$, $g$pt$\mathsf{q}$ be arbitrary functions for which the needed operations are defined. Then the linearity can be expressed by:

$$D_a^\alpha(\lambda f(t) + \mu g(t)) = \lambda D_a^\alpha f(t) + \mu D_a^\alpha g(t\mathsf{q} \qquad (3.5.50)$$

where D indicates one of the differintegrals introduced above. For a uniformly convergent series of the function, the following can also holds:

$$D_a^\alpha \sum_{k=0}^{\infty} f_k(t) = \sum_{k=0}^{\infty} D_a^\alpha f_k(t\mathsf{q} \qquad (3.5.51)$$

This is proven using the Riemann-Liouville derivative as a test example given that α ą 0 and n " rαs ` 1.

$$\mathsf{q} \quad " \frac{1}{\Gamma(n - \alpha)} \frac{d^n}{dt^n} \int_a^t (t - \tau)^{n-\alpha-1} f(\tau) d\tau$$

$$\ddot{Y}\, {}_{RL}D_a^\alpha \sum_{k=0}^{\infty} f_k(t\mathsf{q} \quad \Gamma\mathsf{p} \quad \alpha\mathsf{q}\, dt$$

$$" \frac{1}{\Gamma(n - \alpha)} \frac{d^n}{dt^n} \int_a^t (t - \tau)^{n-\alpha-1} f(\tau) d\tau$$

$$\Gamma\mathsf{p}n - \alpha\mathsf{q}\, dt$$

$$"\Gamma \frac{1}{\Gamma(n - \alpha)} \frac{d^n}{dt^n} \int_a^t (t - \tau)^{n-\alpha-1} f(\tau) d\tau\, dt$$

“ 1 ÿ8 dtdnn ż tpt ´ τqn´α´1fkpτqdτ

Γpn ´ αq k“0 a

8

“ ÿ RL Dα a fk(tq (3.5.52)

k“0

Equivalence of the Approaches

The operation of fraction differentiation is not commutative however, the equivalence of the Riemann-Liouville and the Caputo approach is a fundamental question which need a separate treatment as shown in this section. Let *f*pτq be p*n*´1q-times continuously differentiable and *f*pnqpτq be integrable. Now suppose *α* ą 0 and *n* “ r*α*s ` 1 but *α* ‰ *n* ´ 1 where *n* “ *nc* then the following expressions hold:

RL Dα a f(t 1 dn t n u “ fpτq u1 “ f1pτq

q “Γ pn ´ αq dtn ż pt ´ τq ´α´1fpτqdτ “ ˇˇˇˇ v1 “ pt ´ τqn´α´1 v “ ´p

t´nτ´qnα´α ˇˇ ˇ

a

1 dn pt ´ aqn´αfpaq t pt ´ τqn´α

“ Γpn ´ αq dtn „ n ´ α ` ża n ´ αf1pτqdτ

« “ nkÿ“0 Γpn ` k ´ α ` 1q ` Γp2n ´ αq ża pt ´ τq ´α´1fpnqpτqdτff dnn´1 pt ´ aqn`k´αfpkqpaq 1 t 2n

dt

n´1 k´αfpkqpaq pt ´ aq 1 t ÿ ż

n´α´1fpnqpτqdτ

$$\ldots = \sum_{k=0}^{n-1} \frac{(t-a)^{k-\alpha} f^{(k)}(a)}{\Gamma(k-\alpha+1)} + {}^{C}D_a^{\alpha} f(t) \qquad (3.5.53)$$

This expression indicates that the operators in general are unequal but if α is an integer then both approaches coincide with the classical derivative.

Composition

The composition of integrals are first treated here as they are defined in the same way in both approaches. Given $a \in \mathbb{R}$, $\alpha, \beta > 0$ and $f(t)$ an integrable function, the following expression can be written:

$$D_a^{-\alpha}(D_a^{-\beta} f(t)) = \frac{1}{\Gamma(\alpha)} \int_a^t (t-\xi)^{\alpha-1} \left[\frac{1}{\Gamma(\beta)} \int_a^{\xi} (\xi-\tau)^{\beta-1} f(\tau) d\tau \right] d\xi$$

$$= \frac{1}{\Gamma(\alpha)\Gamma(\beta)} \int_a^t \int_a^{\xi} (t-\xi)^{\alpha-1} (\xi-\tau)^{\beta-1} f(\tau) d\tau d\xi$$

$$= \frac{1}{\Gamma(\alpha)\Gamma(\beta)} \int_a^t f(\tau) \int_{\tau}^{t} (t-\xi)^{\alpha-1} (\xi-\tau)^{\beta-1} d\xi d\tau$$

$$= \frac{1}{\Gamma(\alpha)\Gamma(\beta)} \int_a^t f(\tau)(t-\tau)^{\alpha+\beta-1} B(\alpha,\beta) d\tau$$

$$= \frac{1}{\Gamma(\alpha+\beta)} \int_a^t (t-\tau)^{\alpha+\beta-1} f(\tau)d\tau = D_a^{-(\alpha+\beta)} f(t) \quad (3.5.54)$$

This shows that the fractional integrals are commutative that is:

$$D_a^{-\alpha}(D_a^{-\beta} f(t)) = D_a^{-(\alpha+\beta)} f(t) \ , \quad a \in \mathbb{R}, \ \ \alpha, \beta > 0 \qquad (3.5.55)$$

Riemann-Liouville Derivatives

The other composition of Riemann-Liouville differintegral is as provided below:

1. The first variant is by definition given as:

$$\frac{d^m}{dt^m} D_a^{\alpha} f(t) = D_a^{\alpha+m} f(t \quad \text{for } \alpha \in \mathbb{R}, m \in \mathbb{N} \qquad (3.5.56)$$

2. This variant says, the Riemann-Liouville derivative operator is a left inverse to the Riemann-Liouville integral operator of the same order.

$$D_a^{\alpha}(D_a^{-\alpha} f(t)) = f(t) \ , \quad \alpha > 0 \qquad (3.5.57)$$

The second variant is proven using the previous expression in Equation (3.5.38), (3.5.55) and (3.5.56) where $n = \lceil \alpha \rceil + 1$ as follows:

$$D_a^{\alpha}(D_a^{-\alpha} f(t) = \frac{d^n}{dt^n}[D_a^{-(n-\alpha)}(D_a^{-\alpha} f(t) = \frac{d^n}{dt^n}(D_a^{-n} f(t)) = f(t \quad (3.5.58)$$

Hence the inverse formula does not hold because of the sum of the new terms that occurred as shown below:

$$D_a^{-\alpha}(D_a^{\alpha} f(t) = f(t) + \sum_{k=1}^{n} D_a^{\alpha-k} f(t) \frac{(t-a)^{\alpha-k}}{\Gamma(\alpha-k+1)} \qquad (3.5.59)$$

$$\Gamma(\alpha-k+1) \quad k=1 \quad t=a$$

The proof is as detailed below:

$$D_a^{-\alpha}(D_a^{\alpha}f(t)) = \frac{1}{\Gamma(\alpha)}\int_a^t (t-\tau)^{\alpha-1}D_a^{\alpha}f(\tau)d\tau$$

$$= \frac{1}{\Gamma(\alpha+1)}\frac{d}{dt}\int_a^t (t-\tau)^{\alpha}\frac{d^n}{d\tau^n}[D_a^{-(n-\alpha)}f(\tau)]d\tau$$

$$= \frac{d}{dt}\left\{D_a^{-(\alpha+1-n)}(D_a^{-(n-\alpha)}f(t)) - \sum_{k=1}^{n}\frac{d^{n-k}}{dt^{n-k}}D_a^{\alpha-n}f(t)|_{t=a}\frac{(t-a)^{\alpha-k+1}}{\Gamma(\alpha-k+2)}\right\}$$

$$= f(t) - \sum_{k=1}^{n} D_a^{\alpha-k}f(t)|_{t=a}\frac{(t-a)^{\alpha-k}}{\Gamma(\alpha-k+1)} \qquad (3.5.60)$$

With the first one which is the fractional derivative of the fractional integral, two situations may occur:

1. $\beta \geq \alpha \geq 0$: here the relations in Equation (3.5.55) and (3.5.57) is required.

$$D_a^{\alpha}(D_a^{-\beta}f(t)) = D_a^{\alpha}[D_a^{-\alpha}(D^{-(\beta-\alpha)}f(t))] = D_a^{\alpha-\beta}f(t) \qquad (3.5.61)$$

2. $\alpha > \beta \geq 0$: here an introduction of $n = [\alpha] + 1$ is required with relations in Equation (3.5.38), (3.5.55) and (3.5.56):

$${}^{RL}D_a^{\alpha}({}^{RL}D_a^{-\beta}f(t)) = \frac{d^n}{dt^n}[{}^{RL}D_a^{-(n-\alpha)}({}^{RL}D_a^{-\beta}f(t))] = \frac{d^n}{dt^n}({}^{RL}D_a^{\alpha-\beta-n}f(t)) = {}^{RL}D_a^{\alpha-\beta}f(t) \qquad (3.5.62)$$

Both cases lead to the same consequence as summarised below:

$$ {}^{RL}D_a^{\alpha}({}^{RL}D_a^{-\beta}f(t_{qq} \text{“} \quad {}^{RL}D_a^{\alpha-\beta}f(t) , \quad \alpha,\beta \geqslant 0 \qquad (3.5.63)$$

Next is with the fractional integration of a fractional derivative. This again requires splitting the computation into the two cases, *β* ě *α* ě 0 and *α* ą *β* ě 0 but the process is in both cases exactly the same. Let *m* “ r*β*s ` 1 and suppose that *f*p*t*q has continuous derivatives of sufficient order then:

$$D_a^{-\alpha}(D_a^{\beta}f(t_{qq} \quad \text{“} D_a^{\beta-\alpha}[D_a^{-\beta}(D_a^{\beta}f(t_{qqs}$$

$$\text{“} D_a^{-\alpha}{}_{`\beta}f\text{p}t\text{q} \text{´} \ddot{y}_m D_{a\beta\acute{}k}f\text{p}t\text{q}^{\check{}\check{}\check{}\check{}\check{}} \_\_\_\_\_ \text{p}t \text{´} a\text{q}_{\alpha\acute{}k} \qquad (3.5.64)$$

$$\Gamma\text{p}1 \text{`} \alpha \text{´} k\text{q}$$

$$k\text{“}1 \qquad t\text{“}a$$

The last possibility is the fractional derivative of a fractional derivative. Let *α,β* ą 0 and *n* “ r*α*s ` 1, *m* “ r*β*s ` 1. Then the below expression can be derived as:

$${}^{RL}D_a^{\alpha}({}^{RL}D_a^{\beta}f(t$$

$$qq \qquad \text{“} \quad \_\_\_ d_{nn}$$

$$\text{r}{}^{RL}D^{\acute{}}\text{p}n\acute{}\alpha\text{q}\text{p}{}^{RL}D_{a\beta}f\text{p}t\text{q}$$

$$\text{qs}_{\,a}$$

$$dt$$

$$\text{“} \quad {}_{RL}D_{a\alpha`\beta}f\text{p}t\text{q} \text{´} \ddot{y}_m \quad {}_{RL}D_{a\beta\acute{}k}f\text{p}t\text{q}^{\check{}\check{}\check{}\check{}\check{}} \quad \text{p}t \text{´} a\text{q}_{\alpha\acute{}k} \qquad (3.5.65)$$

$$\Gamma\text{p}1 \text{´} \alpha \text{´} k\text{q}$$

$$k\text{“}1 \qquad t\text{“}a$$

If *f*p*t*q is p*m* ´ 1q-times continuously differentiable and $f^{\text{p}m\text{q}}$ is integrable, then the conditions $D_a^{\beta-k}f(t)|_{t=a} = 0$ for all *k* “ 1,⋯⋯ ,*m* are equivalent to the conditions

$f^{(k)}(a) = 0$ for all $k = 0, \cdots, m-1$. It is then possible to write the Riemann-Liouville derivative after some integration by parts in the form ($\alpha > 0$, $n = [\alpha] + 1$):

$$D_a^{\alpha} f(t) = \sum_{k=0}^{n-1} \frac{f^{(k)}(a)(t-a)^{k-\alpha}}{\Gamma(1+k-\alpha)} + \frac{1}{\Gamma(n-\alpha)} \int_a^t (t-\tau)^{n-\alpha-1} f^{(n)}(\tau) d\tau \qquad (3.5.66)$$

Caputo Derivative

The results of the compositions of Riemann-Liouville differintegrals can be applied here. Considering all relations used in Riemann-Liouville the following is derived for $\alpha, \beta > 0$ as:

$${}_C D_a^{\alpha} {}_C D_a^{\beta} f(t) = {}_C D_a^{\alpha+\beta} f(t) \qquad (3.5.67)$$

3.6 Development in Gradient Edge Mask

3.6.1 Review of Classical Edge Detectors

Edge detection refers to the process of identifying and locating sharp discontinuities in an image (Gonzalez and Woods, 2000). These discontinuities usually are abrupt changes in pixel intensity. Such discontinuities are known to characterise object boundaries in an image scene (Sangwine and Horne, 1998). Using standard approach, object boundaries in an image can be extracted by performing convolution on the image using a 2D filter. The filter (mask) are matrix operator constructed to be sensitive to large gradients in the image while returning values of zero in isotropic regions (Umbaugh, 1998). The choice of edge detector depends on many features as outlined below:

1. *Edge orientation:* The geometry of the operator determines a characteristic direction in which it is most sensitive to edges. The operators can be optimised to detect horizontal, vertical or diagonal edges.

2. *Noise environment:* Edge detection is difficult in noisy images, since both the noise and the edges contain high frequency content. Attempts to reduce the noise result in blurred and distorted edges. Operators used on noisy images are typically larger in scope, so they can average enough data to discount localised noisy pixels. This results in a less accurate localisation of the detected edges.

3. *Edge structure:* Not all edges involve a steep change in intensity. Effects such as refraction or poor focus of photographic equipment can result in objects with boundaries defined by a gradual change in intensity. In that case, an operator needs to be chosen to be responsive to such a gradual change.

In general, derivative edge detectors can be classified into two main categories:

1. *Gradient Operators:* It detects the edges by looking for the maximum and minimum in the first derivative of the image pixel value.

2. *Laplacian Operators:* The Laplacian method searches for zero crossings in the second derivative of the image pixel value to find the object boundaries.

By the literature, gradient based operators are mostly preferred since they give better scaling on most images that have varied distortions (Ziou and Tabbone, 1997). Sobel, Prewitt and Canny edge detectors are examples of such operators and are considered below:

3.6.2 Sobel Operator

In the standard Sobel operator for a 3 ˆ 3 mask operator, the pixel of interest on the Cartesian grid with the derivative gradient “e” and its eight neighbors having derivative estimate (density) is as shown in Figure 3.8. The component derivative

a	b	c
d	e	f
g	h	i

Figure 3.8: A general 3 ˆ 3 Gradient Stencil

estimate vector *G* is defined as:

$$G = \frac{\text{density difference}}{\text{distance to neighbour}} \quad (3.6.1)$$

This vector is determined such that the direction of *G* will be given by the unit vector to approximate neighbour pairs p*a,i*q,p*b,h*q,p*c,g*q,p*f,d*q called antipodal pairs. Each unit vector is defined in respect to the POI. If the position of the POI is p0,0q, then the position of *f,c,b* and *a* are r1,0s, r1,1s, r0,1s and r´1,´1s respectively with unit vectors r1,0s, r1,1s{*R*, r0,1s and r´1,´1s{*R* respectively where *R* is the magnitude.

-1	0	1
-2	0	2
-1	0	1

Figure 3.9: Sobel 3 ˆ 3 *x*-directional mask

The vector *G* is the sum of the derivative gradient and is given by:

c ´ g r1,1s *a ´ i* r´1,1s

„

$$G = \frac{}{R} \cdot \frac{}{R} \; ` \; \frac{}{R} \cdot \frac{}{R} \; ` \; pb \acute{} h\text{q} \cdot \text{r0,1s} \; ` \; pf \acute{} d\text{q} \cdot \text{r1,0s} \quad (3.6.2)$$

?

where $R =$ 2 which represent the distance to neighbours.

$$G = \frac{}{2} \; ` \; f \acute{} d \, , \; \frac{}{2} \; ` \; b \acute{} h \; (3.6.3) \; pc \acute{} g \acute{} a \; ` \; i\text{q} \; pc \acute{} g \; ` \; a \acute{} i\text{q}$$

Converting G into a line expression and by multiplication by 2, the following is obtained:

$$G^1 \text{“ } 2 \text{ ¨ } G \text{“ rp} c \text{ ´ } g \text{ ´ } a \text{ ` } i\text{q ` } 2 \text{ ¨ p} f \text{´ } d\text{q,p} c \text{ ´ } g \text{ ` } a \text{ ´ } i\text{q ` } 2 \text{ ¨ p} b \text{ ´ } h\text{qs} \quad (3.6.4)$$

The G^1 is reassigned as G from Equation (3.6.4):

$$G_x \text{ “ p} c \text{ ´ } g \text{ ´ } a \text{ ` } i\text{q ` } 2 \text{ ¨ p} f \text{ ´ } g\text{q} \quad (3.6.5)$$

$$G_y \text{ “ p} c \text{ ´ } g \text{ ` } a \text{ ´ } i\text{q ` } 2 \text{ ¨ p} b \text{ ´ } h\text{q} \quad (3.6.6)$$

The Sobel x-directional mask is obtain from the coefficient of the variable where each coefficient is placed at the position of the respective variables it multiplies. That is, a “ ´1, b “ 0, c “ 1, d “ ´2, e “ 0, f “ 2, g “ ´1, h “ 0, i “ 1 hence the x mask is obtained as:

Similar matrix is developed for the y-directional mask. Here, a “ 1, b “ 2, c “ 1, d “ 0, e “ 0, f “ 0, g “ ´1, h “ ´2, i “ ´1 to obtain:

Again, given the 5ˆ5 matrix of image gradient estimates shown in Figure 3.11 below, the position “m” becomes the POI.

1	2	1
0	0	0
-1	-2	-1

Figure 3.10: Sobel 3 ˆ 3 y-directional mask

a	b	c	d	e
f	g	h	i	j
k	l	m	n	o
p	r	s	t	u
v	w	x	y	z

Figure 3.11: A general 5 ˆ 5 gradient stencil

If the position of POI is p0,0q, then the positions of *i,g,h,n,d,b,e,a,j,f,c,o* are r1,1s, r´1,1s, r0,1s, r1,0s, r1,2s, r´1,2s, r2,2s, r´2,2s, r2,1s, r´2,1s, r0,2s, r2,0s with unit vectors r1,1s{R_1, r´1,1s{R_1, r0,1s, r1,0s, r1,2s{R_2, r´1,2s{R_2, r2,2s{R_3, r´2,2s{R_3, r2,1s{R_2, r´2,1s{R_2, r0,2s{R_4, r2,0s{R_4 respectively where R is the magnitude. The vector G is the sum of the derivative gradient and is given by:

$$G \quad \frac{\text{p}i\acute{}r\text{q}}{R_1}\cdot\frac{\text{r1,1s}}{R_1}\grave{}\frac{\text{p}g\acute{}t\text{q}}{R_2}\cdot\frac{\text{r}\acute{}\text{1,1s}}{R_2}\grave{}\text{p}h\acute{}s\text{q}\cdot\text{r0,1s}\grave{}\text{p}n\acute{}l\text{q}\cdot\text{p1,0q}\grave{}R_1\quad R_1$$

$$\frac{\text{p}d\acute{}w\text{q}}{R_2}\cdot\frac{\text{r1,2s}}{R_2}\grave{}\frac{\text{p}b\acute{}y\text{q}}{R_2}\cdot\frac{\text{r}\acute{}\text{1,2s}}{R_2}\grave{}\frac{\text{p}e\acute{}v\text{q}}{R_3}\cdot\frac{\text{r2,2s}}{R_3}\grave{}\frac{\text{p}a\acute{}z\text{q}}{R_3}\cdot\frac{\text{r}\acute{}\text{2,2s}}{R_3}\grave{}$$

$$\frac{\text{p}j\acute{}p\text{q}}{R_2}\cdot\frac{\text{r2,1s}}{R_2}\grave{}\cdot\grave{}\cdot\grave{}\cdot\frac{\text{p}f\acute{}u\text{q}}{R_4}\cdot\frac{\text{r}\acute{}\text{2,1s}}{R_4}\grave{}\frac{\text{p}c\acute{}x\text{q}}{R_4}\cdot\frac{\text{r0,2s}}{R_4}\quad\frac{\text{p}o\acute{}k\text{q}}{R_4}\quad\frac{\text{r2,0s}}{R_4}\qquad(3.6.7)$$

where $R_1 = 2$, $R_2 = 5$, $R_3 = 8$, $R_4 = 2$

$$G = \frac{\text{p}i\acute{}r\acute{}g\grave{}t\text{q}}{2}\grave{}\text{p}n\acute{}l\text{q}\grave{}\frac{\text{p}d\acute{}w\acute{}b\grave{}y\text{q}}{5}\grave{}\frac{2\text{p}e\acute{}v\acute{}a\grave{}z\text{q}}{8}\grave{}\frac{2\text{p}j\acute{}p\acute{}f\grave{}u\text{q}}{5}\grave{},\frac{2\text{p}o\acute{}k\text{q}}{4}$$

$$\frac{\text{p}i\acute{}r\grave{}g\acute{}t\text{q}}{2}\grave{}\text{p}h\acute{}s\text{q}\grave{}\frac{2\text{p}d\acute{}w\grave{}b\acute{}y\text{q}}{5}\grave{}\frac{2\text{p}e\acute{}v\grave{}a\acute{}z\text{q}}{8}\grave{}\frac{\text{p}j\acute{}p\grave{}f\acute{}u\text{q}}{5}\grave{}\frac{2\text{p}c\acute{}x\text{q}}{4}\qquad(3.6.8)$$

Converting G into a line expression by multiplication of 20, the following is obtain:

$$G^1 = 20\cdot G = \text{r10}\cdot\text{p}i\acute{}r\acute{}g\grave{}t\grave{}o\acute{}k\text{q}\grave{}20\cdot\text{p}n\acute{}l\text{q}\grave{}4\cdot\text{p}d\acute{}w\acute{}b\grave{}y\text{q}\grave{}$$

$$5\cdot\text{p}e\acute{}v\acute{}a\grave{}z\text{q}\grave{}8\cdot\text{p}j\acute{}p\acute{}f\grave{}u\text{q}\quad,\quad 10\cdot\text{p}i\acute{}r\grave{}g\acute{}t\grave{}c\acute{}x\text{q}\grave{}20\cdot\text{p}h\acute{}s\text{q}\grave{}$$

$$8\cdot\text{p}d\acute{}w\grave{}b\acute{}y\text{q}\grave{}5\cdot\text{p}e\acute{}v\grave{}a\acute{}z\text{q}\grave{}4\cdot\text{p}j\acute{}p\grave{}f\acute{}u\text{qs}\qquad(3.6.9)$$

The G^1 is reassigned to G from Equation (3.6.9):

$$G_x \text{ “ } 10 \text{ ¨ p}i \text{ ´ } r \text{ ´ } g \text{ ` } t \text{ ` } o \text{ ´ } k\text{q ` } 20 \text{ ¨ p}n \text{ ´ } l\text{q ` } 4 \text{ ¨ p}d \text{ ´ } w \text{ ´ } b \text{ ` } y\text{q ` } 5 \text{ ¨ p}e \text{ ´ } v \text{ ´ } a \text{ ` } z\text{q ` } 8 \text{ ¨ p}j \text{ ´ } p \text{ ´ } f\text{ ` } u\text{q} \quad (3.6.10)$$

$$G_y \text{ “ } 10 \text{ ¨ p}i \text{ ´ } r \text{ ` } g \text{ ´ } t \text{ ` } c \text{ ´ } x\text{q ` } 20 \text{ ¨ p}h \text{ ´ } s\text{q ` } 8 \text{ ¨ p}d \text{ ´ } w \text{ ` } b \text{ ´ } y\text{q ` } 5 \text{ ¨ p}e \text{ ´ } v \text{ ` } a \text{ ´ } z\text{q ` } 4 \text{ ¨ p}j \text{ ´ } p \text{ ` } f \text{ ´ } u\text{q} \quad (3.6.11)$$

The Sobel *x*-directional mask is obtain from the coefficients of the variables where each coefficient is placed at the position of the respective variable it multiplies. That is, *a* “ 5, *b* “ 4, *c* “ 0, *d* “ ´4, *e* “ ´5, *f* “ 8, *g* “ 10, *h* “ 0, *i* “ ´10, *j* “ ´8, *k* “ 10, *l* “ 20, *m* “ 0, *n* “ ´20, *o* “ ´10, *p* “ 8, *r* “ 10, *s* “ 0, *t* “ ´10, *u* “ ´8, *v* “ 5, *w* “ 4, *x* “ 0, *y* “ ´4, *z* “ ´5, hence the following is obtained.

Similarly, the matrix is develop for the *y*-directional mask. Here, *a* “ 5, *b* “ 8, *c* “ 10, *d* “ 8, *e* “ 5, *f* “ 4, *g* “ 10, *h* “ 20, *i* “ 10, *j* “ 4, *k* “ 0, *l* “ 0, *m* “ 0,

5	4	0	-4	-5
8	10	0	-10	-8
10	20	0	-20	-10
8	10	0	-10	-8
5	4	0	-4	-5

Figure 3.12: Sobel 5 ˆ 5 *x*-directional mask

n “ 0, *o* “ 0, *p* “ ´4, *r* “ ´10, *s* “ ´20, *t* “ ´10, *u* “ ´4, *v* “ ´5, *w* “ ´8, *x* “ ´10, *y* “ ´8, *z* “ ´5 to obtained.

5	8	10	8	5
4	10	20	10	4
0	0	0	0	0
-4	-10	-20	-10	-4

-5	-8	-10	-8	-5

Figure 3.13: Sobel 5 ˆ 5 *y*-directional mask

3.6.3 Prewitt Operator

Prewitt gradient kernel is based on the idea of contrast difference instead of central gradient estimate as in Sobel and general basis vectors instead of the unit vectors.

The component of *G* is the density difference given as(Phillips, 1994):

$$G = pc + gq - r1,1s - pa + iq - r1,1s - pf + dq - r1,0s - pb + hq - r0,1s$$

$$= pc + g + a - i - f + dq, pc + g - a + i - b + hq \qquad (3.6.12)$$

The horizontal and vertical mask (G_x and G_y) which are obtained from Equation (3.6.12) are as shown in Figure 3.15 and 3.14 respectively:

1	1	1
0	0	0
-1	-1	-1

Figure 3.14: Prewitt 3 ˆ 3 *y*-directional mask

-1	0	1
-1	0	1
-1	0	1

Figure 3.15: Prewitt 3 ˆ 3 *x*-directional mask

In a similar fashion for a 5 ˆ 5 matrix, *G* is obtained as:

$$G \quad \text{“p}i\,'\,r\,'\,g\,`\,t\,`\,n\,'\,l\,`\,d\,'\,w\,'\,b\,`\,y\text{q}\,`\,2\,\text{¨ p}e\,'\,v\,'\,a$$

$$`\,z\,`\,j\,'\,p\,'\,f\,`\,u\,`\,o\,'\,k\text{q , p}i\,'\,r\,`\,g\,'\,t\,`\,h\,'\,s\,`\,j\,'\,p\,`$$

$$f\,'\,u\text{q}\,`$$

$$2\,\text{¨ p}d\,'\,w\,`\,b\,'\,y\,`\,e\,'\,v\,`\,a\,'\,z\,`\,c\,'\,x\text{q} \qquad (3.6.13)$$

The horizontal and vertical masks are obtained from the coefficient and are displayed in Figure 3.16 and (3.17).

-2	-1	0	1	2
-2	-1	0	1	2
-2	-1	0	1	2
-2	-1	0	1	2
-2	-1	0	1	2

Figure 3.16: Prewitt 5 ˆ 5 *x*-directional mask

2	2	2	2	2
1	1	1	1	1
0	0	0	0	0
-1	-1	-1	-1	-1
-2	-2	-2	-2	-2

Figure 3.17: Prewitt 5 ˆ 5 *y*-directional mask

3.6.4 Canny Algorithm

1. **Canny Optimum Criteria:** Canny (1986) stated that, a well performing edge detection should have the following three objectives:

(a) Low probability of "mismarking" non-edge points and low probability of "nonmarking" the real edge points.

(b) The pixel marked as edge points should be as close to the centre of the real edges as possible. The real edges here are referred to the ground truth of the edges of object in an image.

(c) The result of applying the operator should return unique pixel edge point.

To achieve these objectives, Canny proposed that the edge detection operator should satisfy these three optimal criteria: Signal to Noise Ratio (SNR) criterion; Location Accuracy criterion and Single-Edge Response criterion as detailed below:

(a) **SNR Criterion:** This criterion ensures that the real edge is not missed while the non-edge point is not detected as the edge. Mathematical it is defined as:

$$SNR = \frac{\int_{-w}^{w} E'(x) f(x)\,dx}{\sigma \int_{-w}^{w} f(x)\,dx} \qquad (3.6.14)$$

where $f(x)$ is the filter impulse response (image gradient), $E'(x)$ is the edge threshold function, x is the pixel position with respect to the POI, σ is the root mean square of Gaussian noise. The greater the value of SNR, the higher the quality of the edge detector.

(b) **Location Accuracy Criterion:** Distance between the actual and located position of the edge should be minimal. It means the location of the detected edge should be as close to the real edge as possible. The mathematical expression is:

$$Loc = \frac{1}{\sigma \int_{-w}^{w} f(x) f'(x)\,dx} \qquad (3.6.15)$$

The greater the value of *Loc*, the higher the quality of the detector.

(c) **One Response Criterion:** It minimises multiple edge location to a single edge location by maximally suppressing spurious edges as given by.

$$x_{max} = \frac{\int_{-\infty}^{\infty} f'(x)^2\,dx}{\int_{-\infty}^{\infty} f''(x)^2\,dx} \cdot \frac{f(x)^2}{\int_{-\infty}^{\infty} f(x)^2\,dx} \cdot 2\pi \qquad (3.6.16)$$

x_{max} is the unique edge and meeting this criterion ensures that there is a unique point.

2. **Canny Edge Detection:** Canny algorithm is the optimum application of the product of SNR and the location characteristic. It works by:

(a) first smoothing the image using a Gaussian filter

(b) calculating the magnitude and direction of the resultant gray level gradient (c) take the non-maxima suppression on the gradient magnitude

(d) finally detecting and connecting the edge from the candidate points using the high and low threshold

Figure 3.18 gives the flowchart of the Canny's algorithm.

(a) **Gaussian Smoothing Filter:** Image smoothing by Gaussian filter (Yuan, 2000; Sanjeev et al., 2002; Barash and Comaniciu, 2004) means removing

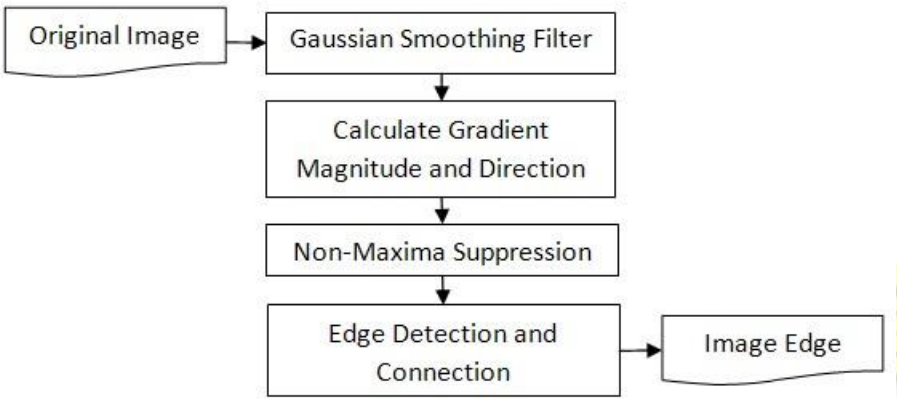


Figure 3.18: Canny edge detection algorithm

image noise. The filter as a function is defined as:

$$G(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \quad (3.6.17)$$

and the gradient vector given as:

$$\begin{bmatrix} \frac{\partial G}{\partial x} \\ \frac{\partial G}{\partial y} \end{bmatrix}$$

5 G “ (3.6.18)

where ^

^ BGx^2 y^2

— “ kxexp^´ —$_2$ ¨exp^´ —$_2$ · “ h_1pxqh_2pyq (3.6.19)

Bx2σ 2σ

BGy_2 x_2

— “ ky exp^´ —$_2$ ¨exp^´ —$_2$ · “ h_1pyqh_2pxq (3.6.20)
By2σ 2σ

The parameter σ can be set according to user requirement and this determine smoothness of the filtered image. The larger the σ is, the wider the frequency band of the filter and can be adjusted according to the different images concern.

(b) **Gradient Magnitude and Direction Calculation:** From Equation (3.6.18), the gradient magnitude and direction of individual point is calculated. Discretising Equation (3.6.19) and Equation (3.6.20), the following expressions are obtained respectively:

BG
P_xpi,jq “ — “ rGpi,j ` 1q ´ $G$p$i,j$q `
Bx

Gpi ` 1,$j$ ` 1q ´ Gpi ` 1,$j$qs{2 (3.6.21)

BG
P_ypi,jq “ — “ rGpi,jq ´ Gpi ` 1,$j$q `
By

$$G(i,j-1) - G(i-1,j-1)\} \quad (3.6.22)$$

The image gradient magnitude $M(i,j)$ and the edge direction $\theta(i,j)$ at the point (i,j) are derived as:

$$M(i,j) = \sqrt{P_x^2(i,j) + P_y^2(i,j)} \quad (3.6.23)$$

$$\theta(i,j) = \arctan\left(\frac{P_y(i,j)}{P_x(i,j)}\right) \quad (3.6.24)$$

(c) **The Non-maxima Suppression on the Gradient Magnitude:** Given $M(i,j)$, the pixel direction $\theta(i,j)$ is determined to be $(0,180)$, $(45,235)$, $(90,270)$ and $(135,315)$. Create a matrix of the same dimension as $M(i,j)$ with all entries being zero. An arbitrary n neighborhood null matrix is defined with the centre being the chosen POI. The POI is mapped to the gradient magnitude $M(i,j)$ of the image and shifted across all elements of the image gradient magnitude matrix. At each POI, the element in the direction determine by $\theta(i,j)$ is compared and covered by the null matrix against the element of the POI. If the element at the POI is of higher value than all the elements in the given direction then the value at the corresponding position in the complementary value is set to 1 otherwise, is set to 0. This process is repeated as the POI of the null matrix move across the element of the gradient magnitude matrix.

The complementary matrix of binary zero and one indicate edge lines at the one position, however, these edge lines are not optimal. Isolated edge points or outlier may be included as edge boundaries

(d) **The Dual Threshold and Edge Connection:** The dual threshold algorithm is used to obtain an optimal edge boundary at which point the true edges are connected. To connect the true edges, the dual threshold is perform and this take the following steps to accomplish:

i. Set manually the high threshold value T_h and low threshold value T_l ii. Scan the image and choose any pixel p*i,j*q in the candidate edge image with pixel value 1.

iii. Look up the value *M*p*i,j*q at the pixel position p*i,j*q of the gradient magnitude matrix *M*.

1. the pixel position p*i,j*q will be marked as an edge point if the gradient magnitude *M*p*i,j*q ě T_h
2. the pixel position p*i,j*q will be marked as non-edge point if the gradient magnitude *M*p*i,j*q ă T_l
3. the pixel position p*i,j*q is marked for further processing to test for the connectivity of the edge if *M*p*i,j*q is between T_h and T_l. If there are two adjacent pixel which are edge point then the point is marked as edge point otherwise it is marked as non edge point.

iv. Finally, connect through the marked edge points to get the final edge map.

3.7 Extending the Classical to Fractional Edge Detectors

The concept to the formulation of the classical edge detectors can be extended to the world of fractional edge detectors. Here, the Tiansi mask which is worth looking at will be discussed along with the formulation of the proposed method.

3.7.1 Derivation of Tiansi Fractional Differential Gradient Mask

Extending the concept of classical edge detector to fractional edge detector, a mask based on the fractional differential finite impulse (FIR) proposed by Tseng (2006) was developed. Given below is the FIR where v is the differential order, D is the differential operator, T is the systemic sampling period and z is the displacement (random) variable:

$$D^{v}\{f(z)\} \approx \frac{1}{T^{v}} f(z) - v f(z-1) \quad (3.7.1)$$

Referencing the binomial series expansion the following expressions are obtained:

$$D^{v}\{f(z)\} \approx T^{-v}\left[1 - vz^{-1} + \frac{v(v-1)}{2!}z^{-2} - \frac{v(v-1)(v-2)}{3!}z^{-3} + \cdots\right] f(z)$$

$$\approx \frac{1}{T^{v}} \sum_{i=0}^{\infty} (-1)^{i} \frac{\Gamma(v+1)}{\Gamma(i+1)\,\Gamma(v-i+1)} z^{-i} f(z) \quad (3.7.2)$$

$$T(f(z) - f(z-1))$$

where Γp¨q is the Gamma function. For a given function *f*p*x*q, the signal differential equation from Equation (3.7.2) is obtained as:

$$\frac{d^{v} f(t)}{dt^{v}} \approx f(t) * (v)f(t-1) * \frac{v(v-1)}{2} f(t-2) * \cdots * \sum_{n} \frac{\Gamma(v+1)}{(1)\,n!\,\Gamma(v-n+1)} f(t-n) \quad (3.7.3)$$

The coefficient of Equation (3.7.3) are used to build the Tiansi mask according to the determined directional gradient to be applied to the digital image.

$$a_1 = -v,\ a_2 = \frac{v(v-1)}{2},\ a_3 = -\frac{v(v-1)(v-2)}{6},$$
$$a_4 = \frac{v(v-1)(v-2)(v-3)}{24},\ \cdots,\ a_n = (-1)^{n} \frac{\Gamma(v+1)}{n!\,\Gamma(v-n+1)} \quad (3.7.4)$$

Given $a_1, a_2, \cdots a_n$, a $(2n-1) \times (2n-1)$ mask is obtained with POI as center and $a_1, a_2, \cdots a_n$ defining positive sense of the direction and $-a_1, -a_2, \cdots, -a_n$ defining the negative sense of the direction. If $I(x,y)$ is the matrix element of the mask (x,y), then the following directional mask is obtained as follows:

1. *135° direction*

$$D_{RU \oslash LD} = D^{v}_{RU} - D^{v}_{LD}$$
$$= a_1 I(x-1, y-1) - a_1 I(x-1, y-1) - \cdots - a_n I(x-n, y-n) - a_n I(x-n, y-n) \quad (3.7.5)$$

$\frac{(v^2-v)}{2}$	0	0	0	0
0	$-v$	0	0	0
0	0	0	0	0

0	0	0	*v*	0
0	0	0	0	$\frac{pv'v^2q}{2}$

Figure 3.19: Tiansi 135° directional mask

2. *90°direction*

$D_{YU\emptyset YD}$ “ $D^v_{YU} - D^v_{YD}$

“ $a_1Ipx,y' 1q' a_1Ipx,y` 1q` \cdots\cdots` a_nIpx,y' nq' a_nIpx,y` nq$ (3.7.6)

0	0	$\frac{pv^2'vq}{2}$	0	0
0	0	′*v*	0	0
0	0	0	0	0
0	0	*v*	0	0
0	0	$\frac{pv'v^2q}{2}$	0	0

Figure 3.20: Tiansi 90° directional mask

3. *45° direction*

$D_{LU\emptyset RD}$ “ $D^v_{LU} - D^v_{RD}$

“ $a_1Ipx' 1,y' 1q' a_1Ipx` 1,y` 1q` \cdots\cdots` a_nIpx' n,y' nq' a_nIpx` n,y` nq$ (3.7.7)

0	0	0	0	$\frac{pv^2'vq}{2}$
0	0	0	′*v*	0
0	0	0	0	0
0	*v*	0	0	0

$\frac{p\sqrt{2}q}{2}$	0	0	0	0

Figure 3.21: Tiansi 45° directional mask

4. *180°direction*

$$D_{XL\emptyset XR} \quad \text{“ } D^{v}_{XL} - D^{v}_{XR}$$

$$\text{“ } a_1 I p x \acute{} 1, y q \acute{} a_1 I p x \grave{} 1, y q \grave{} \cdots\cdots \grave{} a_n I p x \acute{} n, y q \acute{} a_n I p x \grave{} n, y q \quad (3.7.8)$$

0	0	0	0	0
0	0	0	0	0
$\frac{p\sqrt{2}\acute{}\sqrt{}q}{2}$	$\acute{}\sqrt{}$	0	$\sqrt{}$	$\frac{p\sqrt{}\sqrt{2}q}{2}$
0	0	0	0	0
0	0	0	0	0

Figure 3.22: Tiansi 180° directional mask

Once you have this element, then all other element of the mask is set to zero including the element of the POI. The larger the mask the greater the error in determining the true edge while 3 ˆ 3 adds unwanted details to the edges. Hence a 5 ˆ 5 is mostly preferred.

Using the Sobel operator as obtained in Figure 3.12, Figure 3.13 and Equation (3.6.10), replace the coefficient by the direction coefficient to get two tables G_x and G_y from G. Pick out the G_x which is the x component of G and apply the general weight to the x component vector in the following way $1,2,\cdots\cdots,pn \acute{} 2q,\cdots\cdots,2,1$ where n is odd. For example, n “ 5 will yield the weights p1,2,3,2,1q. Multiply the weight along the row to get the Tiansi x directional mask.

Similarly, pick the G_y component and multiply the weights along the column to get the Tiansi *y* directional mask.

$\frac{p\nu^{2'}\nu q}{2}$	$'\nu$	0	ν	$\frac{p\nu'\nu^2 q}{2}$
$p\nu^{2'}\nu q$	$'2\nu$	0	2ν	$p\nu'\nu^2 q$
$\frac{3p\nu^{2'}\nu q}{2}$	$'3\nu$	0	3ν	$\frac{3p\nu'\nu^2 q}{2}$
$p\nu^{2'}\nu q$	$'2\nu$	0	2ν	$(\nu'\nu^2)$
$\frac{p\nu^{2'}\nu q}{2}$	$'\nu$	0	ν	$\frac{p\nu'\nu^2 q}{2}$

Figure 3.23: Tiansi *x*-directional fractional mask

$\frac{p\nu^{2'}\nu q}{2}$	$p\nu^{2'}\nu q$	$\frac{3p\nu^{2'}\nu q}{2}$	$p\nu^{2'}\nu q$	$\frac{p\nu^{2'}\nu q}{2}$
$'\nu$	$'2\nu$	$'3\nu$	$'2\nu$	$'\nu$
0	0	0	0	0
ν	2ν	3ν	2ν	ν
$\frac{p\nu'\nu^2 q}{2}$	$p\nu'\nu^2 q$	$\frac{3p\nu'\nu^2 q}{2}$	$p\nu'\nu^2 q$	$\frac{p\nu'\nu^2 q}{2}$

Figure 3.24: Tiansi *y*-directional fractional mask

Chapter 4

PROPOSED MASK FORMULATION AND IMAGE ANALYSIS

4.1 Construction of the Proposed Mask

Consider the formulation of the generalised Srivastava-Owa fractional integral and derivative of the form (Ibrahim, 2011):

$$I_{z}^{\alpha,\mu} f(z) = \frac{\Gamma(\mu)}{\Gamma(\alpha)} z^{-\mu} \int_0^z (z-\xi)^{\alpha-1} \xi^{\mu} f(\xi) d\xi \qquad (4.1.1)$$

$$D_{z}^{\alpha,\mu} f(z) = \frac{\Gamma(\mu)}{\Gamma(\alpha)} z^{-\mu} \int_0^z (z-\xi)^{\alpha-1} \xi^{\mu} f(\xi) d\xi, \; 0 \le \alpha < 1 \qquad (4.1.2)$$

where $n \in \mathbb{N}$, α and $\mu \neq -1$ are real numbers and the function $f(z)$ is analytic in a simply connected region of the complex z-plane $\mathbb{C}$ containing the origin. The multiplicity of $(z-\xi)^{\alpha}$ is removed by requiring that $\log(z-\xi)$ be real when $(z-\xi) > 0$. Given that $\mu = 0$, both expressions result into the standard Riemann-Liouville fractional integral and derivative respectively.

Taking the Riemann-Liouville fractional integral in Equation (4.1.1) at $\mu = 0$ and considering an order of $(1-\alpha)$ on it, the following is obtained as:

$$I_{z}^{1-\alpha} f(z) = \frac{1}{\Gamma(1-\alpha)} \int_0^z f(\xi)(z-\xi)^{-\alpha} d\xi; \; \alpha \neq 0 \qquad (4.1.3)$$

Γp1 ´ αq 0

Representing Equation (4.1.2) at μ " 0 in the form shown in Equation (4.1.4), it is observed that, it is a composite function of Equation (4.1.3). Hence, Equation (4.1.4) can be written as Equation (4.1.5).

$$D_z^{\alpha} f(z\text{q} \qquad \text{" } dz\,\Gamma\text{p1 ´ } \alpha\text{q ż}_0\,\text{pz ´ } \xi\text{q}_{\alpha}\,d\xi;\ 0 \text{ ď } \alpha \text{ ă } 1 \quad (4.1.4)\ d \qquad 1$$

$$\text{" } \frac{d}{dz} I_z^{1-\alpha} f(z\text{q} \qquad (4.1.5)$$

Definition 4.1.1. Let *f*pzq and *g*pzq be two functions, then the convolution of *f* and *g*, denoted by $f \circ g$, is the function on $z \text{ ě } 0$ given by:

$$\text{ż}_{\xi=0}^{z} f\text{pzq} \circ g\text{pzq} = f\text{pξq}g\text{pz ´ ξq}d\xi \qquad (4.1.6)$$

Definition 4.1.2. Let α be a constant and let *f* and *g* be two functions, then:

$$\text{ż}_{\xi=0}^{z} r\alpha f\text{pξqs}g\text{pz ´ ξq}d\xi = \text{ż}_{\xi=0}^{z} f\text{pξqr}\alpha g\text{pz ´ ξqs}d\xi = \alpha\text{ż}_{\xi=0}^{z} f\text{pξq}g\text{pz ´ ξq}d\xi \qquad (4.1.7)$$

which can be written as:

$$r\alpha f\text{s} \circ g = f \circ r\alpha g\text{s} = \alpha r f \circ g\text{s}.$$

If *D* is a differential operator and the function *f* and *g* are analytic, then by the derivative property of convolution, the following is obtained:

$$D\text{p}\alpha r f \circ g\text{sqpzq} = \alpha D f\text{pzq} \circ g\text{pzq}. \qquad (4.1.8)$$

By invoking Definition (4.1.1), Equation (4.1.3) can be written as shown below:

$$I_z^{1-\alpha} f(z) = \frac{1}{\Gamma(\alpha)} \int_0^z (z-\xi)^{\alpha-1} f(\xi)\, d\xi; \ \alpha > 0$$

$$= f(z) * g(z) = g(z) * f(z) = \frac{z^{-\alpha}}{\Gamma(1-\alpha)} * f(z). \quad (4.1.9)$$

Substituting Equation (4.1.9) into Equation (4.1.5) and applying Theorem (4.1.2), the following expression is derived:

$$D_z^{\alpha} f(z) = \frac{d}{dz} \left[\frac{1}{\Gamma(1-\alpha)} \int_0^z (z-\xi)^{-\alpha} f(\xi)\, d\xi \right]$$

$$= \frac{d}{dz} [f(z) * g(z)] \quad (4.1.10)$$

$$= f'(z) * g(z) + f(0) g(z).$$

Definition 4.1.3. For any doublet $(n, p) \in \mathbb{N} \times \mathbb{Q} \setminus (-1, 1)$ and for $z \in \mathbb{R}$, a derivative of order n of the function f such that:

$$f(z) = \begin{cases} a & \text{if } z = 0 \text{ and } p = 0 \\ az^{-p} = 0 & \text{if } z \neq 0 \text{ and } p \neq 0 \\ ae^{-p \ln(z)} & \end{cases} \quad (4.1.11)$$

is defined by:

$$\frac{d_n(az^{-p})}{dz^n} = \frac{d^n z^{-p}}{dz^n}$$

$$= \frac{(-1)^n (p+n-1) a\Gamma(p+1)}{\Gamma(p+n)} z^{-(p+n)} \text{ for } p \neq 0. \quad (4.1.12)$$

In order to make the generation of the mask more clearer, let $g(z) = \frac{d}{dz}\left(\frac{z^{-\alpha}}{\Gamma(1-\alpha)}\right)$ from Equation (4.1.10) be written it in the form shown below:

$$g(z) = \frac{1}{\Gamma(1-\alpha)}\frac{d}{dz}z^{-\alpha}, \quad 0 \leqslant \alpha < 1 \tag{4.1.13}$$

By direct comparison of Equation (4.1.13) with the expression in Definition (4.1.3), $a = \frac{1}{\Gamma(1-\alpha)}$, $n = 1$ and $p = \alpha$, hence, the evaluation gives:

$$g(z) = \frac{\Gamma(\alpha + 1)z^{\alpha - 1}}{\Gamma(\alpha)\Gamma(1-\alpha)}, \quad 0 < \alpha < 1 \tag{4.1.14}$$

Now, by literally applying this approach to the generalised differential operator in Equation (4.1.2), the following is obtained:

$$D_z^{\alpha,\mu} f(z) \cong \frac{(\mu+1)^\alpha}{\Gamma(1-\alpha)}\frac{d}{dz}\left(z^{-\alpha(\mu+1)} * p(z)\right)$$

$$* \frac{z^{\mu-1}}{\Gamma(1-\alpha)} \tag{4.1.15}$$

where $p(z) = z^\mu f(z)$. At this point, it is necessary to strip off the term z^μ in $p(z)$ to get:

$$D_z^{\alpha,\mu} f(z) \cong \frac{(\mu+1)^\alpha}{\Gamma(1-\alpha)}\frac{d}{dz}\left(z^{-\alpha(\mu+1)} * z^\mu f(z)\right)$$

$$* \frac{z^{\mu-1}}{\Gamma(1-\alpha)} \tag{4.1.16}$$

Since α and μ are both constant parameters, it suffices that, Equation (4.1.12) is still applicable to Equation (4.1.16) which results in:

$$g_{\mu}(z) = \frac{\mu \Gamma(\alpha)}{\Gamma(\alpha(\mu-1)+1)\Gamma(\mu-1)} z^{\alpha(\mu-1)} \quad (4.1.17)$$

Definition 4.1.4. Given that $f(z)$ is analytic with $z \in R$ and $\alpha \in Q$, then the higher ordered generalised fractional derivative operator is defined as:

$$D_z^{\alpha,\mu} f(z) = \frac{\Gamma(1+\mu)}{\Gamma(m-\alpha)\Gamma(\mu+1)} \int_0^z \frac{\xi^{\mu} f(\xi)}{(z^{\mu+1}-\xi^{\mu+1})^{\alpha-m+1}} d\xi, \; m-1 < \alpha \leq m \quad (4.1.18)$$

To develop the generalised gradient edge detection mask out of Equation (4.1.18) by following the previous steps, let

$$p_m^{\alpha,\mu} = \frac{(\mu+1)^{\alpha}}{\Gamma(1+\alpha-m)} \quad (4.1.19)$$

By concentrating on the integral part of Equation (4.1.18) and observing the laid down norms, Equation (4.1.20) becomes Equation (4.1.21).

$$\int_0^z \frac{\xi^{\mu} f(\xi)}{(z^{\mu+1}-\xi^{\mu+1})^{\alpha-m+1}} d\xi = \int_0^z \xi^{\mu-m} f(\xi)[(z^{\mu+1}-\xi^{\mu+1})^{-(\alpha-m)}] d\xi$$

$$\hat{} = z^{\mu-m}(z^{\mu+1})^{-\alpha+m} \circ f(z). \quad (4.1.20)$$

$$D_z^{\alpha,\mu} f(z) = p_m^{\alpha,\mu} \cdot z^{\mu-m}(z^{\mu+1})^{-\alpha+m} \circ f(z) s \, d^{m-1} dz$$

$$= p_m^{\alpha,\mu}\left(\frac{d}{dz}\right)^{m+1} Z^{-(\mu+1)(\alpha-m)+m-\mu} * f(z)$$

$$= p_{\alpha,\mu m}\left(\frac{d}{dz}\right)^{m+1} Z^{-(\alpha(\mu+1)-\mu(m+1))} * f(z) \qquad (4.1.21)$$

Picking out the mask component in Equation (4.1.21) and subjecting it to Definition (4.1.3), the final result arrived at is as shown in Equation (4.1.22).

$$g_\mu^m(z) = p_{\alpha,\mu m}\left(\frac{d}{dz}\right)^{m+1} Z^{-(\alpha(\mu+1)-\mu(m+1))}$$

$$= p_m^{\alpha,\mu}(-1)^{m+1}\frac{\Gamma(\alpha(\mu+1)+(m+1)(1-\mu))}{\Gamma(\alpha(\mu+1)-\mu(m+1))} Z^{-(\alpha(\mu+1)+(m+1)(1-\mu))} \qquad (4.1.22)$$

$$= \frac{1}{\Gamma(1-\alpha+m)}\left(\frac{\mu+1}{1}\right)^{\alpha}(-1)^{m+1}\frac{\Gamma(\alpha(\mu+1)+(m+1)(1-\mu))}{\Gamma(\alpha(\mu+1)-\mu(m+1))} Z^{-(\alpha(\mu+1)+(m+1)(1-\mu))}$$

4.1.1 Edge Mask Representation

Since the Riemann-Liouville fractional derivative mask is the desired operator of interest for the image edge analysis, Equation (4.1.13) is considered. However, this equation cannot be applied directly to an image in its 1-Dimensional form and needs to be transformed into 2-Dimensions. This is achieved by setting:

$$z \tilde{N} a\overline{x^2 ` y^2}. \quad (4.1.23)$$

Substituting into Equation (4.1.13) this expression, Equation (4.1.24) and Equation (4.1.25) which are the gradient operators in the direction of *x* and *y* respectively are obtained in its two dimensional (2D) form.

$$\Theta_x px,yq \quad “ \quad \frac{1}{\Gamma(1-\alpha)}\frac{d}{dx}\rho x^2+y^2)^{-\alpha/2}, \quad 0 \leqslant \alpha < 1$$

$$\frac{\qquad\qquad}{2 ´\alpha\{2´1.} \quad “ \quad ´ \alpha ¨ x \quad p 2 \quad (4.1.24)$$

$$x ` y q$$

$$\Gamma p1 ´ \alpha q$$

$$\Theta_y px,yq \quad “ \quad \frac{1}{\Gamma(1-\alpha)}\frac{d}{dy}\rho x^2+y^2)^{-\alpha/2}, \quad 0 \leqslant \alpha < 1$$

$$“ \quad ´\frac{\qquad\qquad}{\Gamma p1 ´ \alpha q}\alpha ¨ y p x2 ` y2q´\alpha\{2´1. \quad (4.1.25)$$

Since Equation (4.1.24) and Equation (4.1.25) will be applied to a digital image, it has to be rewritten in a more presentable way as shown in the equations below:

$$\Theta_x px_i,y_i q \quad “ \quad \frac{\qquad\qquad}{y_{i2}q´\alpha\{2´1 \quad \Gamma p1 ´ \alpha q} \alpha ¨ x_i \quad x2_i ` \quad (4.1.26)$$

$$\Theta_y px_i,y_i q \quad “ \quad \frac{\qquad\qquad}{´ \quad \Gamma p1 ´ \alpha q} \alpha \cdot\cdot y_i \quad 2 \quad y_i 2q´\alpha\{2´1 \quad px_i ` \quad (4.1.27)$$

where ´*m* ď *i* ď *m* and ´*n* ď *j* ď *n* with p2*m*`1qˆp2*n*`1q being the mask (grid) size. In order to construct the actual mask, a meshgrid is first generated. For the purpose of demonstration in this work, a 5 ˆ 5 grid is considered as shown in Table

4.1 and 4.2 for the *x* and *y* components respectively.

Table 4.1: Grid of *x* component

-2	-1	0	1	2
-2	-1	0	1	2
-2	-1	0	1	2
-2	-1	0	1	2
-2	-1	0	1	2

Table 4.2: Grid of *y* component

-2	-2	-2	-2	-2
-1	-1	-1	-1	-1
0	0	0	0	0
1	1	1	1	1
2	2	2	2	2

Evaluating Equation (4.1.26) and (4.1.27) using the contents of Table 4.1 and 4.2, the proposed fractional mask in the direction of *x* and *y* is obtained as shown in Figure 4.3 and 4.4 respectively.

Table 4.3: Proposed *x*-directional fractional mask

$\frac{?}{2\alpha\ 8^{\alpha}}$ 8Γp1′αq	$\frac{? \alpha}{5^{\alpha}}$ 5Γp1′αq	0	$\frac{?}{'\alpha\ 5^{\alpha}}$ 5Γp1′αq	$\frac{?}{'2\alpha\ 8^{\alpha}}$ 8Γp1′αq
$\frac{?}{2\alpha\ 5^{\alpha}}$ 5Γp1′αq	$\frac{? \alpha}{2^{\alpha}}$ 2Γp1′αq	0	$\frac{?}{'\alpha\ 2^{\alpha}}$ 2Γp1′αq	$\frac{?}{'2\alpha\ 5^{\alpha}}$ 5Γp1′αq
$\frac{?}{2\alpha\ 4^{\alpha}}$ 4Γp1′αq	$\frac{\alpha}{}$ Γp1′αq	0	$\frac{'\alpha}{}$ Γp1′αq	$\frac{?}{'2\alpha\ 4^{\alpha}}$ 4Γp1′αq
$\frac{?}{2\alpha\ 5^{\alpha}}$ 5Γp1′αq	$\frac{? \alpha}{2^{\alpha}}$ 2Γp1′αq	0	$\frac{?}{'\alpha\ 2^{\alpha}}$ 2Γp1′αq	$\frac{?}{'2\alpha\ 5^{\alpha}}$ 5Γp1′αq
$\frac{?}{2\alpha\ 8^{\alpha}}$ 8Γp1′αq	$\frac{? \alpha}{5^{\alpha}}$ 5Γp1′αq	0	$\frac{?}{'\alpha\ 5^{\alpha}}$ 5Γp1′αq	$\frac{?}{'2\alpha\ 8^{\alpha}}$ 8Γp1′αq

Table 4.4: Proposed *y*-directional fractional mask

$\frac{?}{2\alpha\ 8^{\alpha}}$ 8Γp1′αq	$\frac{?}{2\alpha\ 5^{\alpha}}$ 5Γp1′αq	$\frac{?}{2\alpha\ 4^{\alpha}}$ 4Γp1′αq	$\frac{?}{2\alpha\ 5^{\alpha}}$ 5Γp1′αq	$\frac{?}{2\alpha\ 8^{\alpha}}$ 8Γp1′αq
$\frac{? \alpha}{5^{\alpha}}$ 5Γp1′αq	$\frac{? \alpha}{2^{\alpha}}$ 2Γp1′αq	$\frac{\alpha}{}$ Γp1′αq	$\frac{? \alpha}{2^{\alpha}}$ 2Γp1′αq	$\frac{? \alpha}{5^{\alpha}}$ 5Γp1′αq

0	0	0	0	0
?— ´α 5α 5Γp1´αq	? ´α 2α 2Γp1´αq	´α Γp1´αq	? ´α 2α 2Γp1´αq	? ´α 5α 5Γp1´αq
? ´2α 8α 8Γp1´αq	? ´2α 5α 5Γp1´αq	? ´2α 4α 4Γp1´αq	? ´2α 5α 5Γp1´αq	?— ´2α 8α 8Γp1´αq

4.1.2 Implementation of the Proposed Mask

Step 1. Load the image data (I)

Step 2. Convert to grayscale if not in grayscale mode.

Step 3. Discretize the image domain into *m* ˆ *n* partition where *m* denotes number of rows and *n*, number of columns

Step 4. Generate the fractional convolutions mask G_x and G_y in the direction of *x* and *y* respectively.

Step 5. Perform convolution on the image I using the mask G_x and G_y to obtain $\mathcal{D}_x^\alpha \mathcal{I}(x, y)$ and $\mathcal{D}_y^\alpha \mathcal{I}(x, y)$ respectively.

Step 6. Calculate the magnitude $G_M(x,y)$ and orientation $\theta(x,y)$ of the output from *Step 5* as expressed mathematically below:

$$G_M(x,y) = \sqrt{(\mathcal{D}_x^\alpha \mathcal{I}(x,y))^2 + (\mathcal{D}_y^\alpha \mathcal{I}(x,y))^2}$$

$$\theta(x,y) = \tan^{-1}\left(\frac{\mathcal{D}_y^\alpha \mathcal{I}(x,y)}{\mathcal{D}_x^\alpha \mathcal{I}(x,y)}\right).$$

Step 7. Perform Non-Maximum Suppression

Step 8. Perform the Hysteresis Thresholding

Step 9. Finally, perform performance evaluation of the proposed method using the standard measures.

4.2 Performance Metric for Proposed Mask on Images

4.2.1 Introduction

Image edge quality is a characteristic of an image that measures the performance of the mask in detecting the edges of object in an image. The stronger the performance the lesser the degradation caused by the mask. Mean Square Error (MSE), Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM) are used to measure the efficiency of the proposed mask in this work. All this measures are "full-reference" metric that is, they are relative measures.

Let A be a matrix representing the ground truth edge image, B be a matrix representing the denoised edge image, i - pixel row index, j - pixel column index, M and N being the image size (dimension), *Peak* is the maximum allowable pixel value of an image which is 256 for an 8 bit image and 1 for a binary image, μ_A is the estimated mean of image A, μ_B is the estimated mean of image B, σ_A is the standard deviation of image A, σ_B is the standard deviation of image B. C_1,C_2 are constant value given by C_1 " pK_1Lq^2 and C_2 " pK_2Lq^2 respectively where L is the dynamic range of the image just like *Peak* with K_1,K_2 ăă 1.

4.2.2 Data Presentation

Data normally used for testing the performance of a mask is usually presented in two forms depending on the data source or repository. This concept is as shown in Figure 4.1 where Layout 1 and 2 refer to the forms in which data is acquired. Using Layout 1, image is first captured and quantised into the digital image. This image is further

processed to obtain both the Ground Truth edge map pAq and the Noisy Image pB^1q and made available for performance analysis. In Layout 2, only the digital image is made available to users. To obtain the ground truth edge map A, a user specified mask is applied to the digital image while external noise is manually introduced to the digital image to obtain the noisy image B^1. At this point, all required images are available either from Layout 1 or 2. The noisy image is then supplied as input to the edge mask to extract the denoised image edge map B. This denoised image B is finally compare to the ground truth image A to access the performance of the mask using the standard measures treated in the sections below.

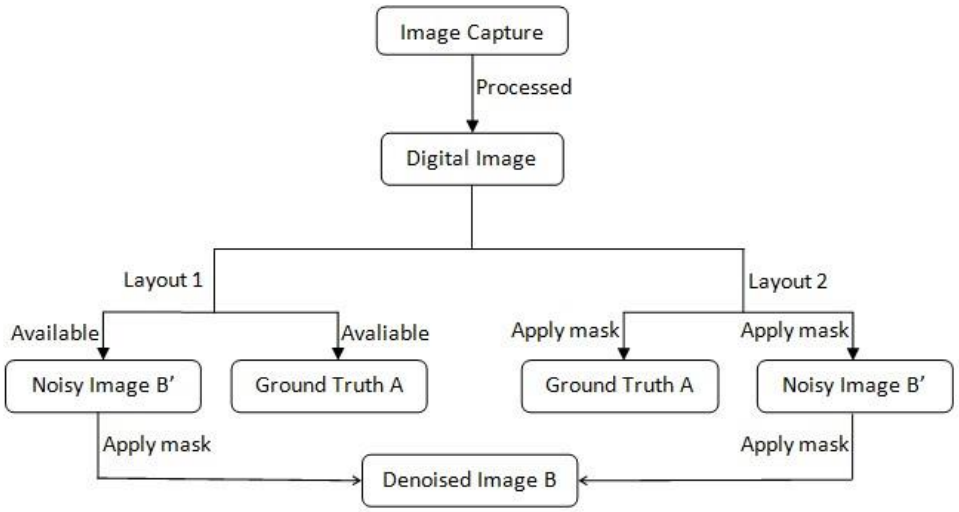


Figure 4.1: Data presentation of ground truth and denoised image

4.2.3 Mean Squared Error

It measure the square deviation of the denoised image from the ground truth image and indicate how close the denoised edge is to the ground truth edge image. It is given by:

$$MSE = \frac{1}{M \cdot N} \sum_{i=1}^{M} \sum_{j=1}^{N} \text{p}A\text{p}i,j\text{q} \acute{} B\text{p}i,j\text{qq}^2. \qquad (4.2.1)$$

The smaller the value the more accurate the detection of edges using the mask.

4.2.4 Peak Signal to Noise Ratio

PSNR is measured in decibles(DB). It measures how much edge is preserved. The factor for comparison is the log*MSE* which measures how far it is from the log*Peak*^{2}.

An image with 2 bits per pixel contains either 0 or 1 and peak is defined as 1. The PSNR is given by:

$$PSNR = 10 \log_{10} \frac{Peak^2}{MSE} \quad (4.2.2)$$

The greater the PSNR value, the better the detection.

4.2.5 Structural Similarity Index Measure

It is a measure of how the ground truth and the denoised edge image are closely related. It measure the exactness of coincidence of pixel position of the denoised edge image and the ground truth edge image position. This is given by:

$$\sigma_{AB} = \frac{1}{N} \sum_{i=1}^{N} (A_i - \mu_A)(B_i - \mu_B) \quad (4.2.3)$$

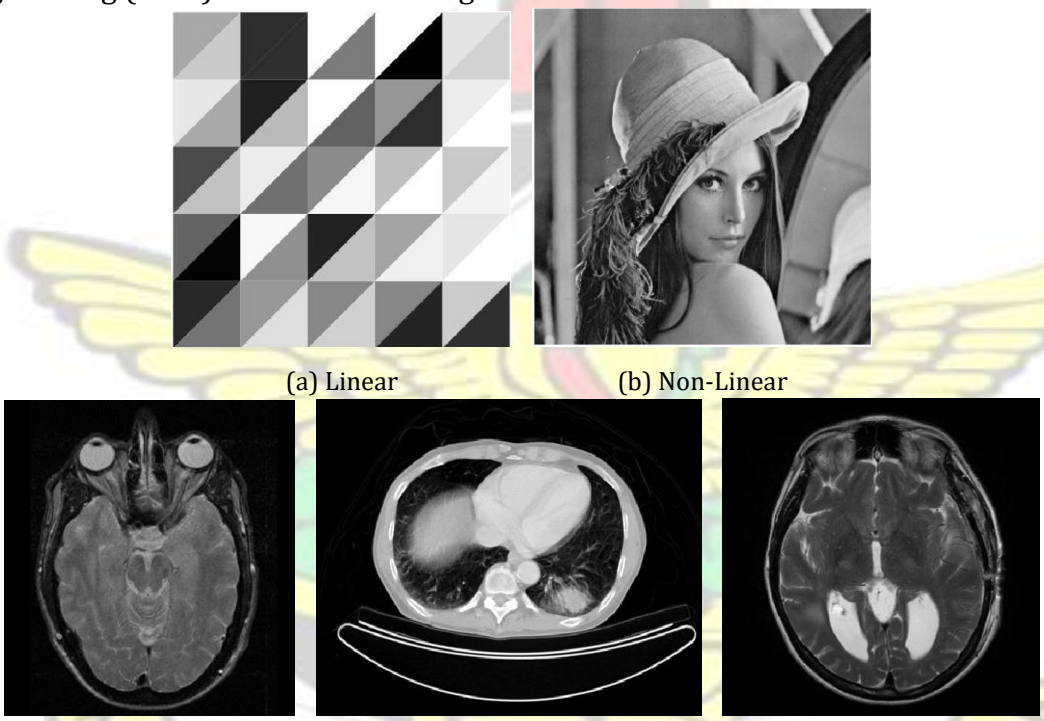
$$SSIM(A,B) = \frac{(2\mu_A\mu_B + C_1)(2\sigma_{AB} + C_2)}{(\mu_A^2 + \mu_B^2 + C_1)(\sigma_A^2 + \sigma_B^2 + C_2)} \quad (4.2.4)$$

Similarly, the higher its value, the better the performance of the mask.

4.3 Image Source

In order to test the efficiency of the proposed convolution mask, data is required. For this purpose, five standard test images are collected from different repositories as shown in Figure 4.2. This is to check the scalability of the algorithm to different imaging issues with the existing algorithm which serve as a benchmark for performance measure. Figure 4.2a and 4.2b are collected from Live for Image & Video

Engineering (LIVE) database while Figure 4.2c - 4.2e are collected from the OsiriX



(a) Linear (b) Non-Linear

(c) Medical Image 1 (d) Medical Image 2 (e) Medical Image 3

Figure 4.2: Original image

Imaging Software, an advanced open-source PACS workstation DICOM Viewer. It is known that during image acquisition, the resulting image will be affected with various artefact regarded as noise. However, this noise is not expected to be visible to all algorithms during edge detection since this influences the final output of the

edge map. Figure 4.3 is an example of some images distorted with gaussian white noise of standard deviation 15.

(a) Linear (b) Non-Linear (c) Medical

Figure 4.3: Some noisy images with standard deviation = 15

4.4 Experimental Result 1:- Performance Test

In this section, the general performance of the proposed mask using different fractional orders of the differential operator is demonstrated. Here, the Peak Signal to Noise Ratio (PSNR) and the Mean Square Error (MSE) of the edge information are extracted during the implementation of the fractional mask and Canny. These algorithms are applied to the images tagged as Linear, Non-Linear and Medical Image 1. It is interesting to note that, these algorithms behave differently with varying mask size. In this work, a mask of size 3 ˆ 3, 5 ˆ 5, 7 ˆ 7 and 9 ˆ 9 are tested on each image for a fair comparison. It is expected that, the higher the value of the PSNR, the better the edge information extracted. In contrast, the lower the MSE value, the closer the edge information extracted is, to the true and actual edge map.

4.4.1 Observation on a 3 ˆ 3 mask

From Table 4.5, it was observed that, the proposed mask on a whole, performs better compared to the Canny algorithm at all image types. The fractional order of α “ 0.5

demonstrated a higher efficiency on the image tagged as Linear, α " 0.2 was best on the Non-Linear while α " 0.1 was chosen as effective on the Medical Image 1.

Although the other α values were not considered due to low performance, it could be noted that, they were all above the Canny performance value. This indicates that, the fractional mask as far as 3 ˆ 3 mask possess better efficiency.

Table 4.5: Proposed method with Canny using a 3 ˆ 3 mask size

masksize	Method	Linear		Non-Linear		Medical	
		MSE	PSNR	MSE	PSNR	MSE	PSNR
3 ˆ	CANNY	0.2571	5.8995	0.3012	5.2111	0.3178	4.9782
	PRO α " 0.1	0.2476	6.0633	0.2945	5.3086	**0.2888**	**5.3935**
	PRO α " 0.2	0.2479	6.0564	**0.2943**	**5.3118**	0.2892	5.3877
	PRO α " 0.3	0.2471	6.0718	0.2945	5.3087	0.2907	5.3650
	PRO α " 0.4	0.2476	6.0632	0.2947	5.3059	0.2911	5.3597
	PRO α " 0.5	**0.2465**	**6.0817**	0.2953	5.2973	0.2916	5.3516
	PRO α " 0.6	0.2476	6.0623	0.2957	5.2915	0.2922	5.3432
	PRO α " 0.7	0.2480	6.0558	0.2960	5.2864	0.2940	5.3165
	PRO α " 0.8	0.2475	6.0637	0.2960	5.2874	0.2942	5.3132
	PRO α " 0.9	0.2478	6.0588	0.2962	5.2845	0.2957	5.2919

4.4.2 Observation on a 7 ˆ 7 mask

With the 7ˆ7 mask, the proposed method again, at all fractional orders of α performs far better than the Canny approach. However, α " 0.1 stand out as the most efficient at various image levels. By literature, fractional calculus have gain higher interest in applied mathematics, but the choice of the fractional order is usually a problem to decide on. At this point, one could conclude that, α " 0.1 is a stable choice for the proposed method using the mask of size 7 ˆ 7 although such conclusion is not explicit until further investigations are made which will be seen in later sections.

It is also observed that, this fractional mask has a linear relationship with the α values as shown in Table (4.6). The higher the fractional order, the higher the MSE

value and the lower the PSNR value at all instances of the images used. This again inspire us to confirm that, this mask is stable to some degree of tolerance as it poses less stress on users in search of an appropriate fractional order. With the assumption that these properties inherited by this mask is homogeneous for all images, then the implementation of this mask in a real time as far as fractional world is concerned will reduce the required parameters by 2 which will improve the runtime of this algorithm on a general bases.

Table 4.6: Proposed method with Canny using a 7 ˆ 7 mask size

7 ˆ masksize	Method	Linear		Non-Linear		Medical	
		MSE	PSNR	MSE	PSNR	MSE	PSNR
	CANNY	0.1296	8.8727	0.1025	9.8911	0.1462	8.3508
	PRO α " 0.1	**0.0308**	**15.1136**	**0.0509**	**12.9351**	**0.0761**	**11.1853**
	PRO α " 0.2	0.0324	14.8995	0.0512	12.9059	0.0826	10.8284
	PRO α " 0.3	0.0349	14.5731	0.0525	12.8024	0.0857	10.6699
	PRO α " 0.4	0.0386	14.1369	0.0540	12.6788	0.0878	10.5664
	PRO α " 0.5	0.0412	13.8542	0.0559	12.5255	0.0906	10.4278
	PRO α " 0.6	0.0441	13.5601	0.0581	12.3555	0.0929	10.3218
	PRO α " 0.7	0.0459	13.3825	0.0604	12.1876	0.0966	10.1494
	PRO α " 0.8	0.0529	12.7643	0.0635	11.9705	0.0999	10.0030
	PRO α " 0.9	0.0571	12.4334	0.0647	11.8881	0.1039	9.8346

4.4.3 Observation on a 9 ˆ 9 mask

One will expect that, the 9 ˆ 9 mask will have the same properties as the 7 ˆ 7 mask. Unfortunately, this is not so as observed from Table 4.7, although nearly the same. For this mask, two α values was recorded as efficient depending on the category of image. The linear image had an appropriate edge estimation at α " 0.2 while the Non-Linear and the Medical was at α " 0.1. A similar conclusion as was with the 7 ˆ 7 mask can be drawn for this mask as well but only for the Medical and Non-Linear image. This mask also advocates the choice of α " 0.1 as efficient since it scales better for the

medical and non-linear images and exhibiting some of the interesting properties of the 7 ˆ 7 mask.

4.4.4 Observation on a 5 ˆ 5 mask

At this section, three algorithms are being considered. That is: Canny, Tiansi and the proposed mask. Tiansi and the proposed mask are fractional based, while Canny Table 4.7: Proposed method with Canny using a 9 ˆ 9 mask size

9 ˆ 9 masksize	Method	Linear		Non-Linear		Medical	
		MSE	PSNR	MSE	PSNR	MSE	PSNR
	CANNY	0.0810	10.9149	0.0762	11.1777	0.1148	9.4000
	PRO α " 0.1	0.0142	18.4743	**0.0347**	**14.5988**	**0.0532**	**12.7443**
	PRO α " 0.2	**0.0140**	**18.5449**	0.0360	14.4375	0.0543	12.6515
	PRO α " 0.3	0.0143	18.4441	0.0368	14.3451	0.0558	12.5306
	PRO α " 0.4	0.0146	18.3638	0.0381	14.1945	0.0568	12.4597
	PRO α " 0.5	0.0146	18.3684	0.0400	13.9797	0.0588	12.3071
	PRO α " 0.6	0.0149	18.2827	0.0417	13.7940	0.0605	12.1796
	PRO α " 0.7	0.0193	17.1344	0.0436	13.6057	0.0702	11.5370
	PRO α " 0.8	0.0210	16.7856	0.0448	13.4854	0.0724	11.4034
	PRO α " 0.9	0.0227	16.4349	0.0462	13.3500	0.0754	11.2258

is classical based (ordinary derivative). Tiansi algorithm in this work is used to validate whether our proposed method is indeed efficient or not and also to ascertain to the fact that, the use of fractional calculus in modern signal and image processing is in general efficient and needs to be encouraged for implementation. From Table 4.8, three different α values are presented depending on the image choice. It is clear that, α " 0.1 is a good choice for the linear, α " 0.8 for a non-linear and α " 0.6 for a medical using the proposed method. In contrast, α " 0.1 is a best choice for the linear and non-linear and α " 0.4 for the medical using Tiansi approach.

However, values from the proposed method as printed in bold characters in Table 4.8 exhibit the highest values among the three methods showing how efficient the

proposed mask is. Although the proposed method ranks first among the existing methods, it is worth noting that, the Tiansi algorithm performs better at all values of α compared to the benchmark set by Canny's algorithm. This in totality demonstrate the scalability of the fractional order derivative based mask as opposed to masks developed with classical order derivative in signal and image processing.

Finally, it is observed that, some α values of the Tiansi method is better than the proposed method and vice-versa by direct mapping and this serves as a caution not to jump into rapt conclusions that the proposed method out perform the existing fractional approach. Observing the edge information in Figure 4.4 closely, one could see that the output of the Canny has quite a number of crispy non-edge across the three images which are quite distinct compared to those of Tiansi and the proposed method. But it seems as though the Proposed and Tiansi are both performing almost nearly the same in terms of performance. Further investigation are therefore encouraged as will be seen soon in the later section for a clear and visible discussion in order to draw an explicit conclusion if there is any.

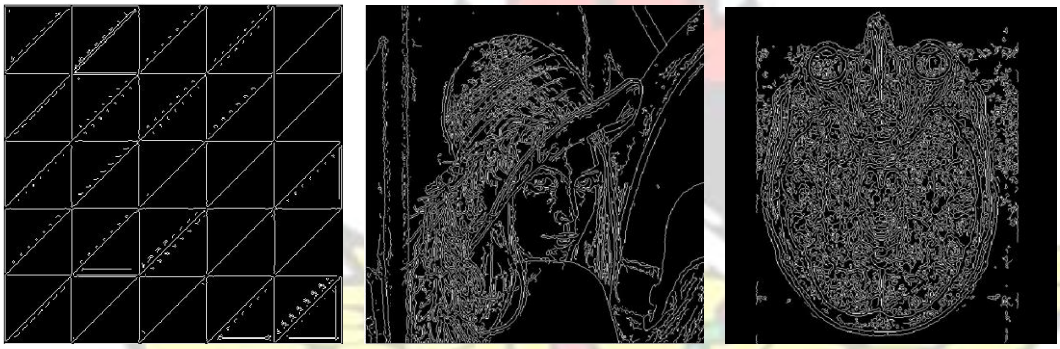
Table 4.8: Proposed method and Tiansi with Canny using a 5 ˆ 5 mask size

5 ˆ masksize	Method	Linear		Non-Linear		Medical	
		MSE	PSNR	MSE	PSNR	MSE	PSNR
	CANNY	0.2266	6.4483	0.2670	5.7342	0.2393	6.2102
	PRO α " 0.1	**0.1580**	**8.0138**	0.1607	7.9402	0.1538	8.1298
	PRO α " 0.2	0.1594	7.9754	0.1659	7.8009	0.1594	7.9746
	PRO α " 0.3	0.1651	7.8232	0.1730	7.6206	0.1669	7.7753
	PRO α " 0.4	0.1718	7.6488	0.1784	7.4856	0.1732	7.6155
	PRO α " 0.5	0.1759	7.5465	0.1858	7.3088	0.1786	7.4805
	PRO α " 0.6	0.1806	7.4329	0.1945	7.1100	**0.1419**	**8.4810**
	PRO α " 0.7	0.1849	7.3301	0.2003	6.9842	0.1462	8.3519
	PRO α " 0.8	0.1889	7.2382	**0.1591**	**7.9826**	0.1502	8.2345
	PRO α " 0.9	0.1930	7.1440	0.1655	7.8130	0.1573	8.0340
	TIA α " 0.1	0.1702	7.6899	0.1765	7.5337	0.1686	7.7307
	TIA α " 0.2	0.1764	7.5344	0.1825	7.3883	0.1762	7.5388

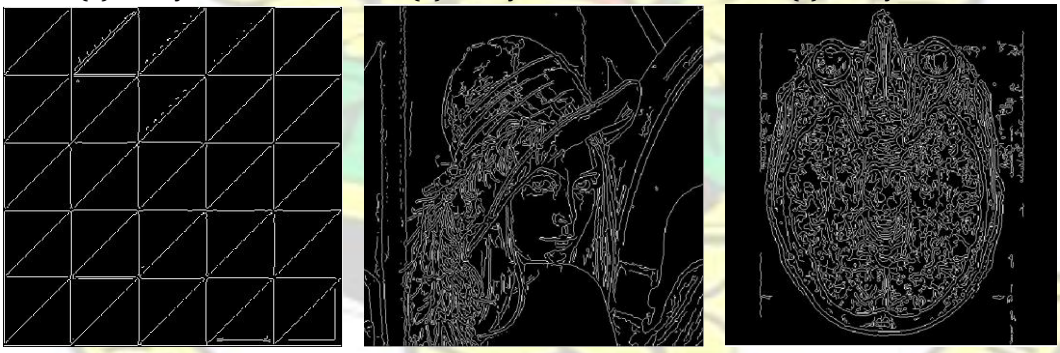
	TIA α " 0.3	0.1835	7.3626	0.1901	7.2093	0.1842	7.3468
	TIA α " 0.4	0.1896	7.2214	0.1981	7.0312	0.1484	8.2843
	TIA α " 0.5	0.1957	7.0833	0.2081	6.8166	0.1542	8.1201
	TIA α " 0.6	0.2035	6.9149	0.2171	6.6326	0.1640	7.8520
	TIA α " 0.7	0.2140	6.6951	0.1865	7.2938	0.1785	7.4829
	TIA α " 0.8	0.1681	7.7450	0.2008	6.9718	0.1922	7.1620
	TIA α " 0.9	0.1830	7.3749	0.2159	6.6569	0.2077	6.8246

4.4.5 General Observation

From all the observations made on the 3 ˆ 3, 5 ˆ 5, 7 ˆ 7 and 9 ˆ 9 mask sizes as witnessed earlier, the following are noticed. Increasing the mask size irrespective of



(a) Canny Linear (b) Canny Non-Linear (c) Canny Medical



(d) Tiansi Linear (e) Tiansi Non-Linear (f) Tiansi Medical

(g) Proposed Linear (h) Proposed Non-Linear (i) Proposed Medical Figure 4.4: Best output using mask size 5 ˆ 5

the image type or the derivative operator increases the performance metric value. This perhaps may be due to the parameter σ in the Gaussian filter which has a great influence on the size filter. This filter helps in performing image pre-processing to smoothen the image before edge detection is performed. Increasing the size of the filter, increases the chance of locating the edges more clearly for further processing.

4.5 Experimental Result 2:- Noise Immunity

In practice, during image acquisition, one will often expect that the resulting image will be contaminated with some uncontrollable features. These features, depending on what one wants can be made negligible with no further processing required. Among some of the features encountered are as follows:

1. gaussian: Gaussian white noise with constant mean and variance
2. salt & pepper: On and off pixels
3. localvar: Zero mean Gaussian white noise with an intensity dependent variance
4. speckle: Multiplicative noise
5. poisson: Shot noise
6. motion blur: blurry pixels
7. erosion: Morphological erosion
8. dilation: Morphological dilation
9. jpg compression blocking effect: Compression artefacts

However, in this work, only four out of the above mentioned will be treated, since they are the most occurring contamination that an image is likely to be subject to easily by literature. Ability to handle these features make an algorithm invariant and robust, hence making its implementation useful. In the next subsections, a detailed analysis of the behaviour of Canny and the proposed method with various mask size at varying noise type will be discussed. Here, the performance metric used is the structural similarly index measure. This metric has the capability to measure how much an image is distorted after being made to go through the various noise types. The higher the value, the higher the similarity of the recovered image is to the expected image.

4.5.1 Noise immunity with 3 ˆ 3 mask

From Table 4.9 and 4.10, it is clear that for each image with a particular noise type, an increase in the noise level leads to a decay in the performance value. In applying a 3ˆ3 mask to the linear image, the proposed method experiences a sharp decay from noise level 30 to 45. This makes the Canny operator robust to motion blur noise as far as linear image is concern. Nonetheless, this is not true for the non-linear and medical image under motion blur. Again, all images under Gaussian white noise using the proposed method has a higher value than Canny but interestingly the same also holds for all images under Salt & Pepper noise using the Canny's method. Taking a final look with images under the speckle noise, the Canny method exhibit an alternating performance with the proposed method. However, the proposed method takes the lead with the non-linear and medical image. It will be hard to conclude with this mixed feeling that the proposed method out performs the Canny as they both have their own flaws. At this point and taking reference to a 3 ˆ 3 mask, adaptive implementation is suggested in other to increase performance.

4.5.2 Noise immunity with 7 ˆ 7 mask

Observation of the noise immunity in Table 4.11 and 4.12 reveals that; the proposed method at all times performs better than the Canny technique except at the medical image subject under the Salt & Pepper noise. From this, it is prudent to suggest the use Canny approach to resolve such issue in instances where image tagged medical is contaminated with salt & pepper noise. However, it will be of interest to know that, the proposed method of mask size 3 ˆ 3 performs best at all level of the mask size. Hence, to remain in the domain of fractional calculus, it is recommended that Table 4.9: Noise immunity with 3 ˆ 3 Canny mask

Motion	Noise SD	20	25	30	35	40	45
	Linear	0.7536	0.7094	0.6124	0.6013	0.5848	0.5910
	Non-Linear	0.5785	0.4246	0.3709	0.3618	0.3529	0.3386
	Medical	0.6707	0.4566	0.3815	0.3785	0.3774	0.3681
Gauss	Linear	0.1081	0.0892	0.0777	0.0655	0.0563	0.0555
	Non-Linear	0.0861	0.0650	0.0535	0.0431	0.0375	0.0331
	Medical	0.0426	0.0361	0.0235	0.0208	0.0155	0.0129
S&P	Linear	0.7103	0.5708	0.4780	0.4003	0.3392	0.2523
	Non-Linear	0.6731	0.5846	0.4842	0.4230	0.3655	0.2897
	Medical	0.7241	0.6630	0.5875	0.4919	0.3687	0.3162
Speckle	Linear	0.2618	0.2416	0.2266	0.2132	0.2021	0.1947
	Non-Linear	0.2303	0.1966	0.1641	0.1513	0.1338	0.1147
	Medical	0.5528	0.5280	0.4972	0.4614	0.4571	0.4373

Table 4.10: Noise immunity with 3 ˆ 3 FDE at α “ 0.5

Motion	Noise SD	20	25	30	35	40	45
	Linear	0.7788	0.7137	0.6056	0.5839	0.5704	0.5569
	Non-Linear	0.6364	0.4761	0.4056	0.3939	0.3847	0.3647
	Medical	0.7040	0.5028	0.4251	0.4174	0.4127	0.4025
	Linear	0.1249	0.1007	0.0893	0.0785	0.0686	0.0604

Gauss	Non-Linear	0.1033	0.0793	0.0665	0.0559	0.0480	0.0430
	Medical	0.0593	0.0451	0.0336	0.0274	0.0199	0.0178
S&P	Linear	0.6006	0.5064	0.4258	0.3549	0.3100	0.2409
	Non-Linear	0.6909	0.5616	0.4560	0.3974	0.3267	0.2580
	Medical	0.7545	0.6764	0.5963	0.4477	0.3870	0.2940
Speckle	Linear	0.2526	0.2276	0.2298	0.2117	0.2060	0.1967
	Non-Linear	0.2887	0.2305	0.2005	0.1915	0.1629	0.1436
	Medical	0.6337	0.5858	0.5691	0.5267	0.5125	0.4960

a swap of fractional mask should be adopted instead of hopping into another mask structure.

4.5.3 Noise immunity with 9 ˆ 9 mask

The structural similarity index measure as shown in Table 4.13 and 4.14 using mask size 9ˆ9 indicates that the proposed method once again performs better for all images at all noise type with varying noise level. However, this observation again is not valid Table 4.11: Noise immunity with 7 ˆ 7 Canny mask

Motion	Noise SD	20	25	30	35	40	45
	Linear	0.9770	0.9352	0.9112	0.8528	0.7886	0.7685
	Non-Linear	0.8399	0.7029	0.5816	0.5188	0.4836	0.4618
	Medical	0.8096	0.6450	0.5414	0.5034	0.4873	0.4756
Gauss	Linear	0.3329	0.2984	0.1984	0.2002	0.1816	0.1355
	Non-Linear	0.2168	0.1610	0.1409	0.1130	0.0946	0.0866
	Medical	0.3416	0.2301	0.1167	0.1014	0.0668	0.0595
S&P	Linear	0.7577	0.6590	0.5930	0.5193	0.4242	0.3738
	Non-Linear	0.7300	0.6223	0.5186	0.4388	0.3857	0.3127
	Medical	0.6902	0.5847	0.5017	0.3985	0.3274	0.2593
Speckle	Linear	0.5638	0.4109	0.4196	0.3537	0.3592	0.3213
	Non-Linear	0.6205	0.4995	0.4910	0.4148	0.3456	0.3378

	Medical	0.7696	0.7213	0.7005	0.6748	0.6673	0.6574

Table 4.12: Noise immunity with 7 ˆ 7 FDE at α " 0.5

Motion	Noise SD	20	25	30	35	40	45
	Linear	0.9820	0.9540	0.9259	0.8792	0.8204	0.7992
	Non-Linear	0.8665	0.7562	0.6426	0.5662	0.5219	0.4961
	Medical	0.8466	0.7085	0.5900	0.5357	0.5082	0.4939
Gauss	Linear	0.6136	0.3837	0.3749	0.2832	0.2750	0.2799
	Non-Linear	0.5000	0.3616	0.2218	0.1723	0.1367	0.1275
	Medical	0.5370	0.3380	0.3544	0.1866	0.1445	0.1009
S&P	Linear	0.8089	0.7549	0.6689	0.6732	0.5958	0.4860
	Non-Linear	0.8150	0.7181	0.6169	0.5534	0.4522	0.4559
	Medical	0.6529	0.5854	0.4799	0.3947	0.3259	0.2649
Speckle	Linear	0.7800	0.6257	0.6336	0.4921	0.4228	0.4460
	Non-Linear	0.7456	0.7243	0.6520	0.5742	0.5087	0.4310
	Medical	0.7720	0.7466	0.7289	0.7109	0.6961	0.6829

for the image tagged medical under the salt & pepper noise but instead an alternating performance is noticed. Due to this alternating characteristic of Canny and the proposed method on the medical image under such noise, one cannot to conclude that the proposed method is efficient compared to Canny and vice-versa. Nevertheless, by swapping to the 3 ˆ 3 mask during implementation, one can satisfactorily say that, the proposed method in general performs better than the Canny's method. In situations where a 3 ˆ 3 mask is exclusively required, then there is a 50% chance of both methods working better depending on the noise type and the image category. Finally, all the three methods are compared in the next section using a 5 ˆ 5 mask size for a more conclusive remark.

Table 4.13: Noise immunity with 9 ˆ 9 Canny mask

	Noise SD	20	25	30	35	40	45
	Linear	0.9714	0.9355	0.9083	0.8639	0.7971	0.7770

Motion	Non-Linear	0.8716	0.7344	0.6140	0.5439	0.5036	0.4792
	Medical	0.8199	0.6667	0.5618	0.5152	0.4919	0.4815
Gauss	Linear	0.4364	0.2718	0.2716	0.2771	0.2089	0.1944
	Non-Linear	0.3338	0.2347	0.1542	0.1469	0.1256	0.1021
	Medical	0.4525	0.2374	0.1634	0.1323	0.1140	0.0881
S&P	Linear	0.7592	0.6748	0.6125	0.5529	0.4473	0.4384
	Non-Linear	0.7285	0.6679	0.5631	0.4614	0.4139	0.3344
	Medical	0.6997	0.5756	0.4879	0.4012	0.3158	0.2577
Speckle	Linear	0.6689	0.4865	0.5035	0.4113	0.4316	0.3536
	Non-Linear	0.6958	0.5763	0.5672	0.4927	0.3948	0.3511
	Medical	0.7557	0.7328	0.7107	0.6896	0.6820	0.6588

Table 4.14: Noise immunity with 9 ˆ 9 FDE at α " 0.5

Motion	Noise SD	20	25	30	35	40	45
	Linear	0.9818	0.9493	0.9270	0.8973	0.8447	0.8151
	Non-Linear	0.9098	0.8058	0.7038	0.6222	0.5671	0.5380
	Medical	0.8805	0.7732	0.6670	0.6031	0.5667	0.5427
Gauss	Linear	0.8422	0.6467	0.6009	0.4269	0.4777	0.3587
	Non-Linear	0.6410	0.5999	0.4123	0.3266	0.2665	0.1951
	Medical	0.6285	0.5280	0.4705	0.3381	0.2510	0.2579
S&P	Linear	0.9254	0.8855	0.8496	0.7843	0.7707	0.7043
	Non-Linear	0.8621	0.8072	0.7358	0.6759	0.5891	0.6016
	Medical	0.6925	0.5735	0.4810	0.3943	0.4067	0.3767
Speckle	Linear	0.9186	0.8753	0.8085	0.6825	0.5964	0.6200
	Non-Linear	0.8521	0.8050	0.7481	0.6977	0.6382	0.5735
	Medical	0.8218	0.7981	0.7731	0.7530	0.7426	0.7194

4.5.4 Noise immunity with 5 ˆ 5 mask

In this section, all the three methods were compared in graphical form for clearer and distinctive observation. These methods were made to undergo all the four noise types at six different noise levels for each test image. Each block of graph contains three subplots

with the first one being for Linear, followed by Non-Linear and lastly the Medical. Each subplot also has three plots with the continuous blue line being for Canny, followed by the red short dashed discontinuous line been for the Proposed and finally the black continuous line with black asterisk markers been for Tiansi. The first block that is, Figure 4.5 is a plot of SSIM with motion blur noise, Figure 4.6 is a plot of SSIM with gaussian white noise, Figure 4.7 is a plot of SSIM with salt & pepper noise while the final block, Figure 4.8 is a plot of SSIM with speckle noise. Observing all the figures except Figure 4.7, it is evident once more that, the Canny method under no circumstance performs close to methods formulated in the fractional calculus approach. However, there is a kind of tough competition between the proposed method and the Tiansi method by looking at the medical and the linear image of Figure 4.5 and Figure 4.6. Once more, the Canny method works well for images tagged as medical with salt & pepper noise effect although the margin is not that wide. By mere visual observation of all the graphs, one can confidently say without doubt that, the fractional mask performs most better than the classical approach but on the general score, the proposed method tops the race in the context of performance and noise immunity. In the section that follows, the medical image segmentation is consider as will be seen shortly.

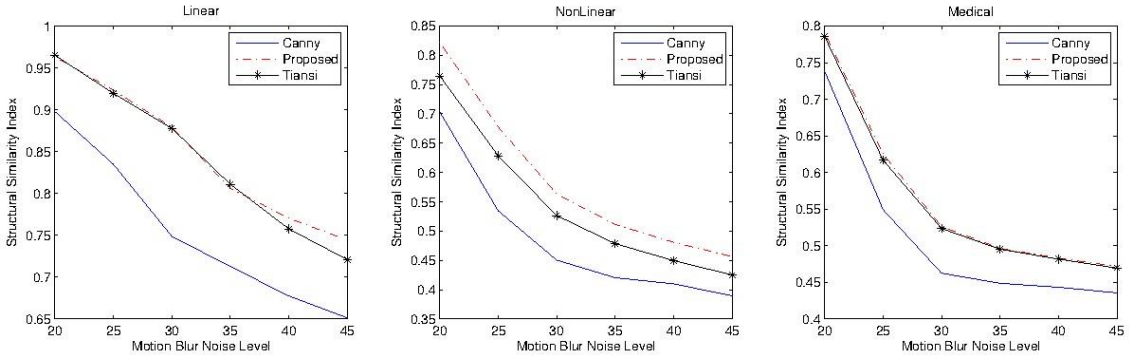


Figure 4.5: Effect of operators on motion blur noise

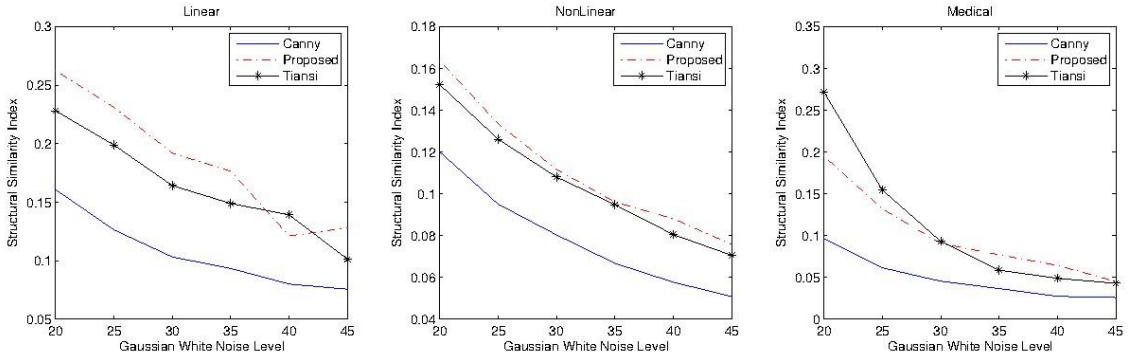


Figure 4.6: Effect of operators on gaussian noise

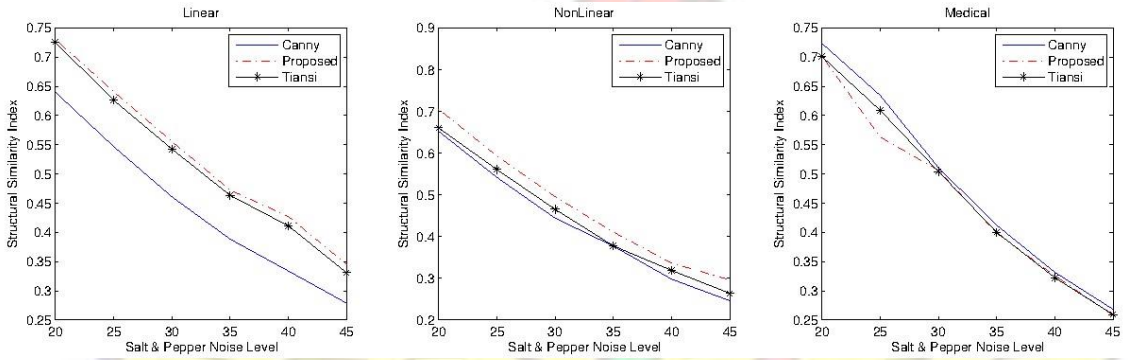


Figure 4.7: Effect of operators on salt and pepper noise

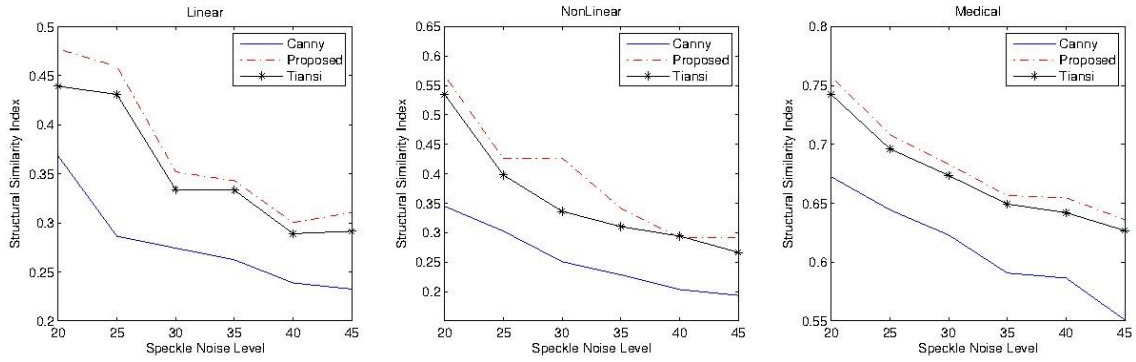


Figure 4.8: Effect of operators on speckle noise

4.6 Experimental Result 3:- Segmentation

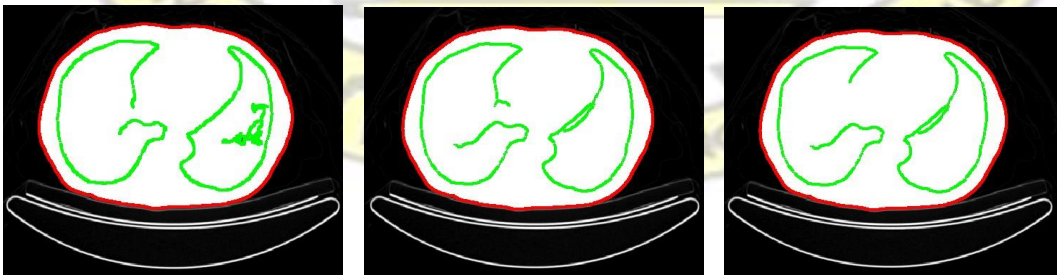
One of the most important and ubiquitous tasks in medical image analysis is segmentation. This is the process of partitioning an image into a set of distinct regions which are different in some important qualitative or quantitative way. This

is a critical intermediate step in all high level object recognition tasks. Surgical operation in the world of today is highly driven by computer assisted imagery. Since life of people are totally subjective to the final output of the graphic generated by the medical instrument, it becomes crucial that the existing techniques scale to a considerable amount of tolerance for effective communication to the end users. To test the tolerance of the proposed method in this context, three standard medical test images are selected with their results compared to that of Canny. Single Seed Region Growing algorithm is employed at this stage for the segmentation, based on the output of the edge maps generated by Canny and the proposed method. A mask size of 9 is used

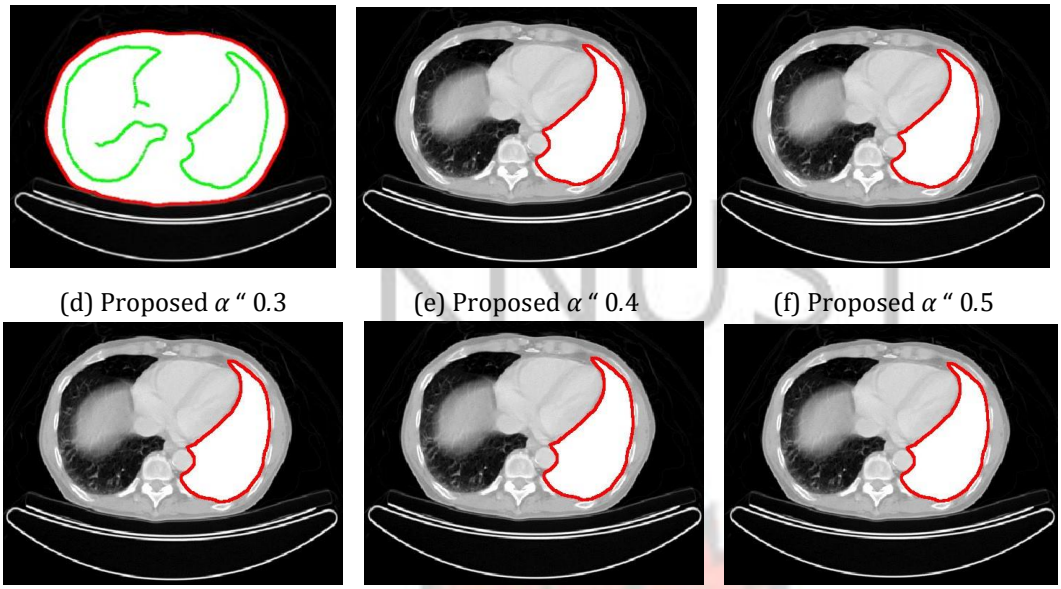
for this purpose.

4.6.1 Segmentation on Medical Image 1

From Figure 4.9, one will notice two colours, green and red. The green is meant for interior boundaries while the red is meant for external boundaries. The focus of this image is for the segmentation to extract only one region of interest and that is the shape to the right. Hence it is expected that only one boundary be seen in the image. An attempt with the Canny method failed by extraction three main boundaries with some isolated objects (noise) as boundaries. The same was the case for the proposed method at α " 0.1,0.2 and 0.3, however, from α " 0.4 to 0.9, the desired region of interest was segmented successfully.



(a) Canny (b) Proposed α " 0.1 (c) Proposed α " 0.2

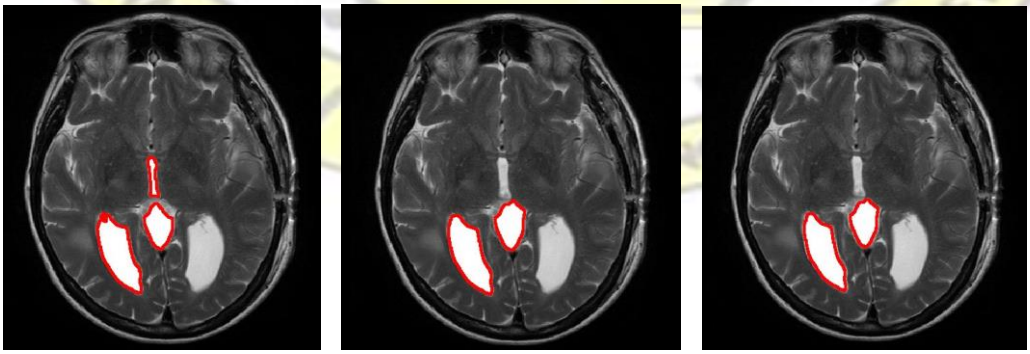


(d) Proposed α " 0.3 (e) Proposed α " 0.4 (f) Proposed α " 0.5

(g) Proposed α " 0.6 (h) Proposed α " 0.7 (i) Proposed α " 0.8 Figure 4.9: Segmentation using medical image 1

4.6.2 Segmentation on Medical Image 2

With regard to the second image, the methods are expected to extract four regions of interest. Using Canny's method, three out of four regions is extracted. On testing our proposed method on the same image, α " 0.1 and 0.2 extracted two regions to the left, α " 0.3,0.4 and 0.5 also extracted two regions to the right, α " 0.6,0.7 and α " 0.8 increased the number of regions extracted to three. In the implementation of α " 0.9, the mask is able to extract all these four required regions of interest which fulfils the assigned task.



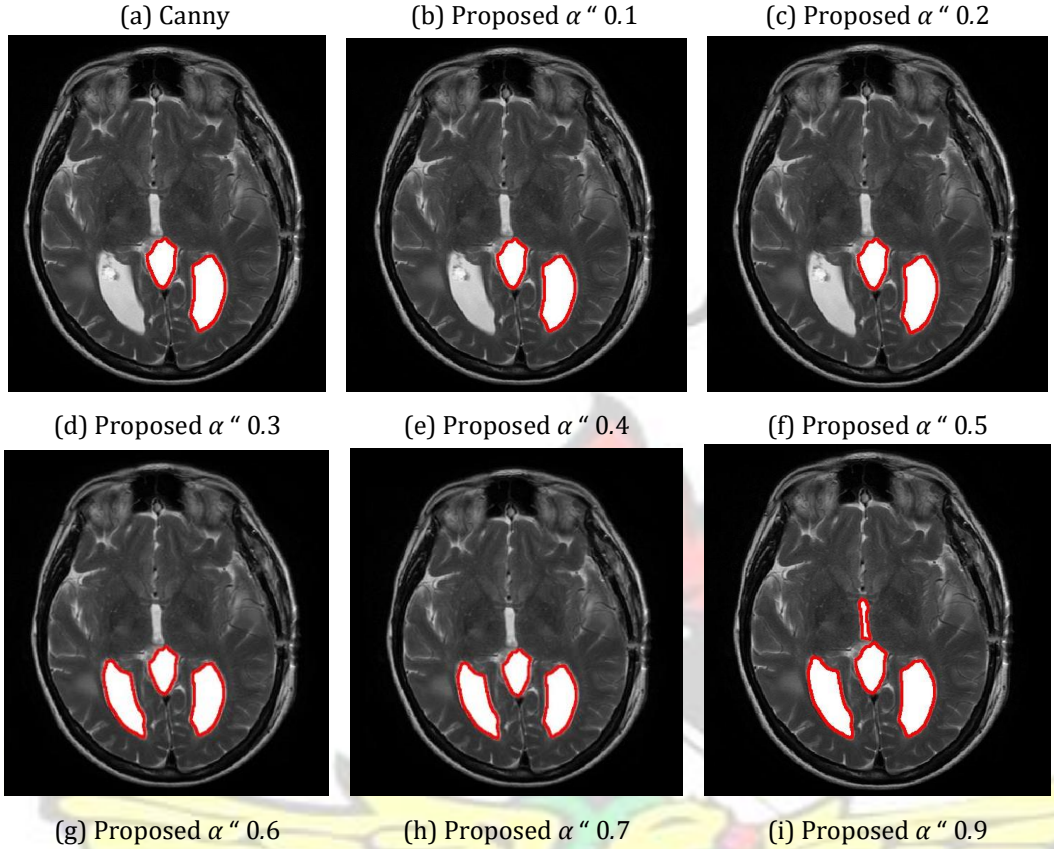
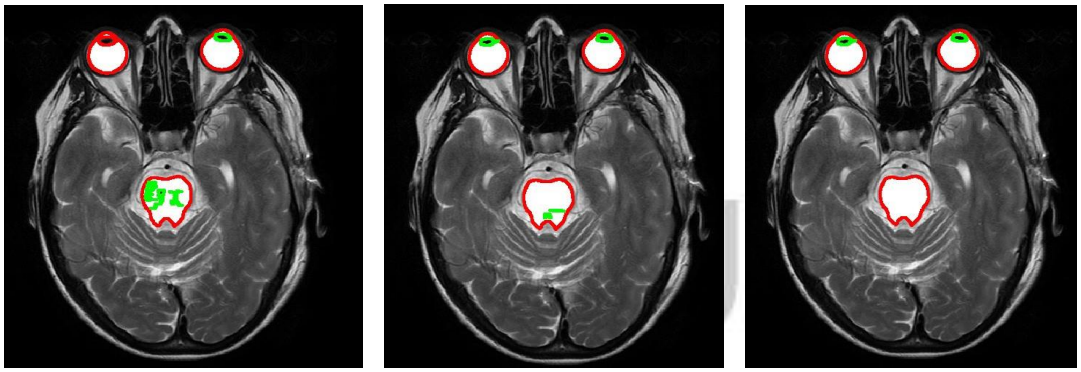


Figure 4.10: Segmentation using medical image 2

4.6.3 Segmentation on Medical Image 3

In the last medical image, the intended purpose was to test if the algorithm can scale effectively to locate both external and interior boundaries. As could be seen in the output image of Canny, it was able to extract all the required external boundaries but got confused as to what an interior boundary was. The same was observed with the proposed method using α " 0.1 although the confusion is quite minimal. In the second attempt with α " 0.2, all the interior as well as the external boundaries is captured. One will notice that, only three images is displayed under this section. This is solely because, at α " 0.3 to 0.9, the same output was witnessed and there is no need presenting them since α " 0.2 could guarantee a perfect segmentation.



(a) Canny (b) Proposed α " 0.1 (c) Proposed α " 0.2

Figure 4.11: Segmentation using medical image 3

4.7 Experimental Result 4:- Behavioural Analysis of Selected Image Smoothing Techniques

In this section, an experimental analysis of two most widely used smoothing functions (i.e. Gaussian and median filter) and spline function are discussed. A demonstration of their effect on the proposed fractional derivative mask is presented in the following subsections. In subsection 4.7.1, a discussion on how the experiment is carried out is presented. Next is subsection 4.7.2, where the selection of an optimal filter length required to attain a high structural similarity index measure is discussed. In subsection 4.7.3, an explanation to how the parameter σ of the Gaussian function can be optimally selected keeping in mind either the homogeneity or the adaptive assumption is detailed here. Finally in subsection 4.7.4, a performance analysis of the three selected smoothing functions are compared with justifications.

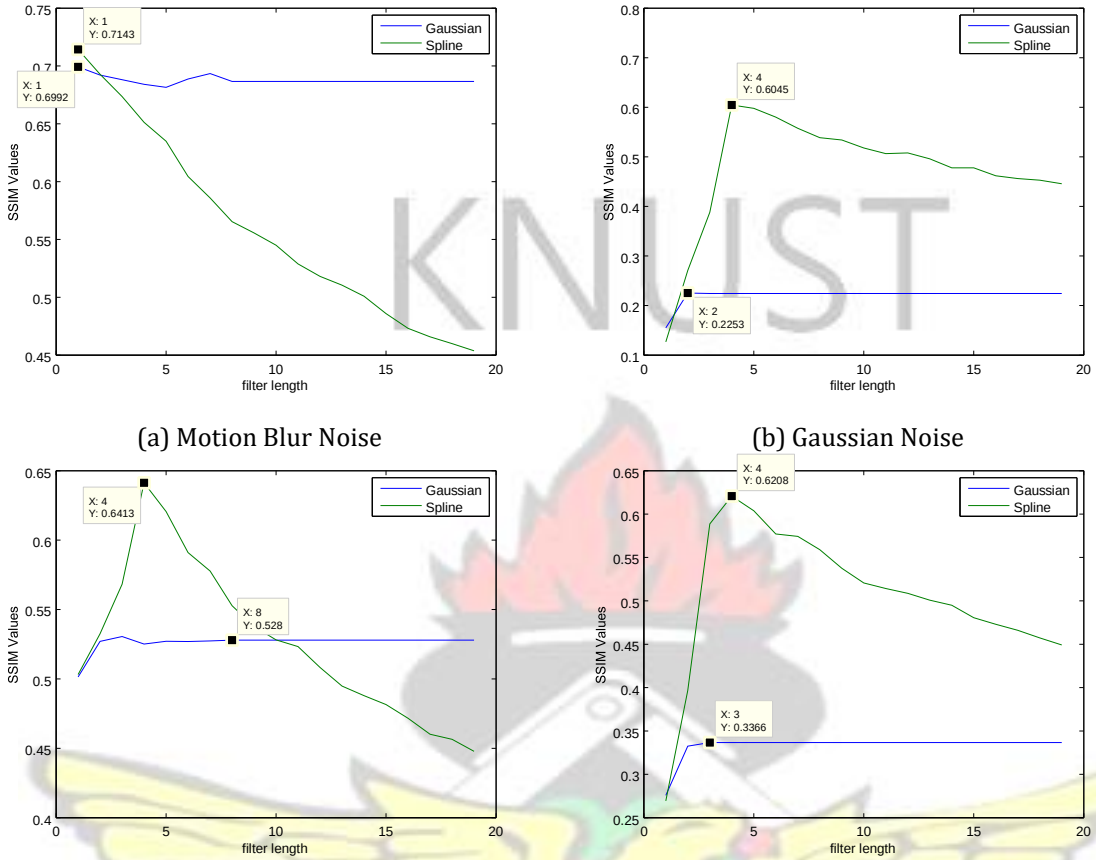
4.7.1 Experimental structure

In order to carry out a more accurate experiment, the following measures and structures was considered. A performance analysis of the behaviour of some selected

smoothing functions on the proposed fractional gradient operator under various noise types is expected. Here, three smoothing functions are considered, that is: Gaussian, Median and Spline function. The noise types also considered are: motion blur, Gaussian, salt & pepper and speckle. One could observe that, aside the median smoothing function, the other remaining two smoothing functions are parametric and an appropriate value is required for each parameter for a proper analysis to be done. The Gaussian function requires a discrete and finite length to form the filter size as well as a sigma parameter describing how peaky the filter should look like. On the other hand, only the discrete and finite length is required by the spline function. The selection of an appropriate filter size is done based on the Structural Similarity index Measure (SSIM).

4.7.2 Optimal selection of filter size for spline and Gaussian function

In the selection of an optimal filter size for the spline and Gaussian function, the σ parameter is assumed to be constant as well as the order of the proposed fractional gradient operator. For the purpose of demonstration, the order α is set to 0.9 while the σ parameter of the Gaussian filter is set to 1. From Figure 4.12, there are four graphs with each graph having two plots. One of the plot is a plot of SSIM against the filter size taking into consideration the Gaussian smoothing function while the other one is for spline smoothing function.



(a) Motion Blur Noise

(b) Gaussian Noise

(c) Salt & Pepper Noise

(d) Speckle Noise

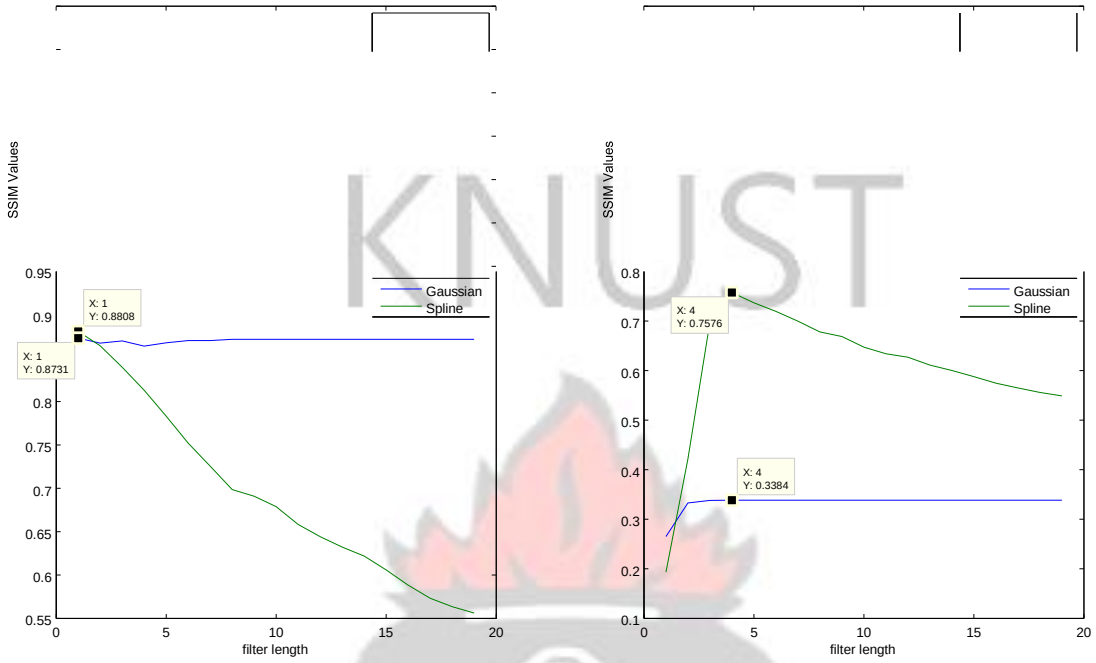
Figure 4.12: Efficiency of smoothing functions on Gaussian noise 3 ˆ 3

In Figure 4.12a, is a plot showing the effect of Gaussian and spline smoothing function under the motion blur noise with varying filter size while keeping all other parameters constant. The same can be said about Figure 4.12b-d, but this time around for Gaussian, Salt & Pepper and Speckle noise respectively. As could be seen from Figure 4.12a-d, the spline smoothing function has a decaying life cycle with varying filter size just after the optimum SSIM is obtained. Nonetheless, the observation is not true for the Gaussian smoothing function, instead, it experience a point of stability as the filter size grow to infinity. Taking a close look at the following sub graphs in Figure 4.12, one notice that, apart from sub Figure 4.12a which suggest a filter size of 3, the remaining figures provide a choice of 9 as filter size using the

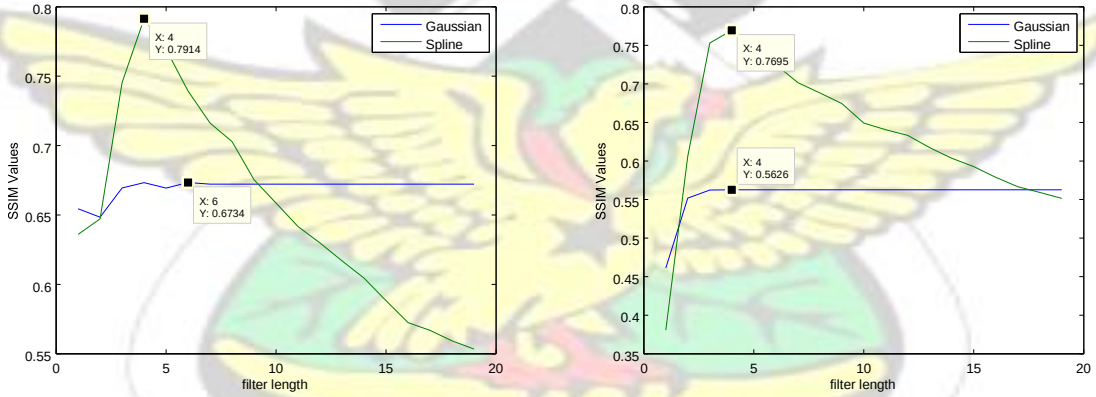
spline smoothing function. As far as the Gaussian smoothing function is concern, it was observed that the choice of the filter size was not consistent and keeps varying with varying noise type. However, this smoothing function also has an interesting property exhibited in Figure 4.12b-d aside Figure 4.12a. As well noted, one could observe that, an optimum filter size is obtained with the Gaussian smoothing function immediately the first stable point is achieved. The following are the optimal filter size recorded for the Gaussian smoothing function: 3, 5, 17 and 7 respectively as indicated in Figure 4.12.

From Figure 4.13, is a representation of these smoothing functions with the proposed fractional gradient operator of size 5 ˆ 5. Similar remarks and observations can be made with the spline smoothing function where again a filter size of 9 is appropriate to obtain a high similarity measure for Gaussian, salt & pepper and speckle noise type and 3 for motion blur noise. At the moment it can be concluded in a way that, a filter of size 9 is efficient when using spline as a smoothing function. With the Gaussian function, the following values 3, 9, 13 and 9 are respectively obtained as the filter size depending on the noise type.

Now to Figure 4.14, where the same experiment was conducted but with 7 ˆ 7 as the operator's (mask) size. In this observation, a slight change is noted, in that, 7 merged out to be the optimal filter size for Gaussian, salt & pepper and speckle noise while motion blur still remain as 3. With this mask type, it is also noticed that, the optimal filter size is not consistent and cannot be tracked for the Gaussian



(a) Motion Blur Noise (b) Gaussian Noise



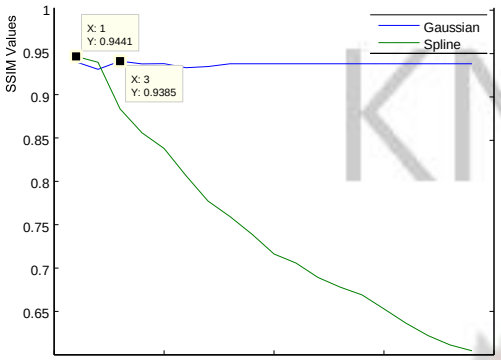
(c) Salt & Pepper Noise (d) Speckle Noise

Figure 4.13: Efficiency of smoothing functions on motion blur noise 5 ˆ 5

smoothing function unlike with the other mask type. From Figure 4.13, 7, 3, 5 and 7 are obtained as optimal length for motion blur, Gaussian, salt & pepper and speckle noise type respectively with no special property identified.

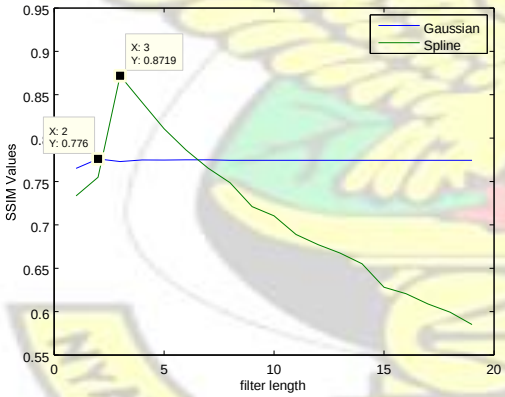
Finally is Figure 4.15 with operator of mask size 9 ˆ 9. Similar trends and characteristics as observed in the Figure 4.12 and Figure 4.13 was noted here. The

only



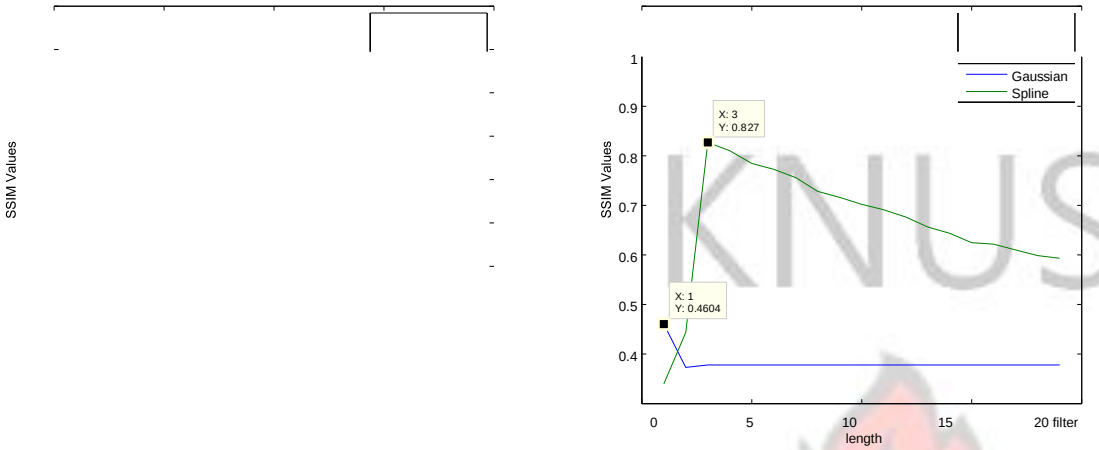
difference is the selection of the optimal filter size which is 7 for the spline smoothing function. Turning to the Gaussian smoothing function, is the following being the optimal filter size for motion blur, Gaussian, salt & pepper respectively that is 3, 7, 9 and 5. At this point, a single optimal value cannot be selected looking

(a) Motion Blur Noise

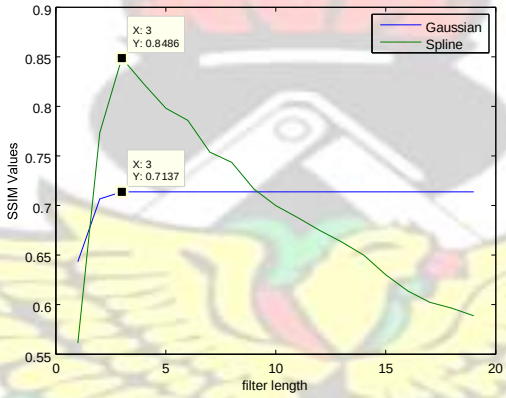


(c) Salt & Pepper Noise

20



(b) Gaussian Noise

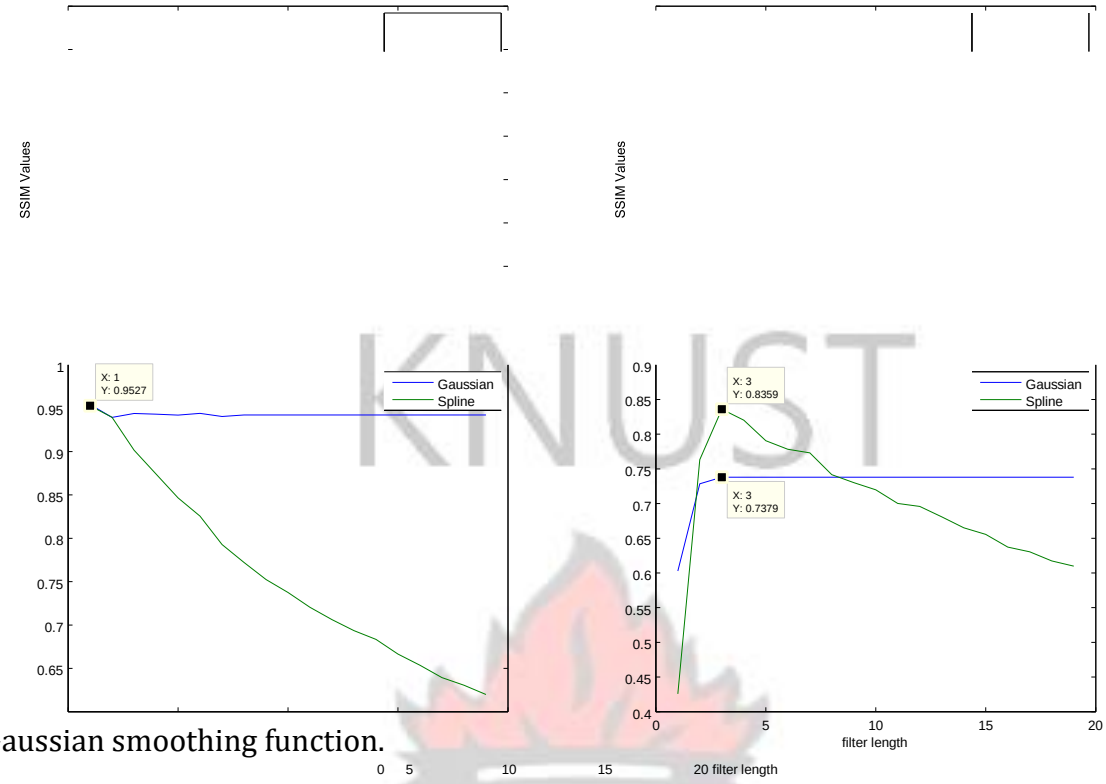


(d) Speckle Noise

Figure 4.14: Efficiency of smoothing functions on salt & pepper noise 7 ˆ 7

at the dynamic nature of the smoothing functions at various noise type although the spline function gives some defined points for which optimal performance is always achieved. In effect, the homogeneity assumption cannot be used here and hence the need to resort to an adaptive approach.

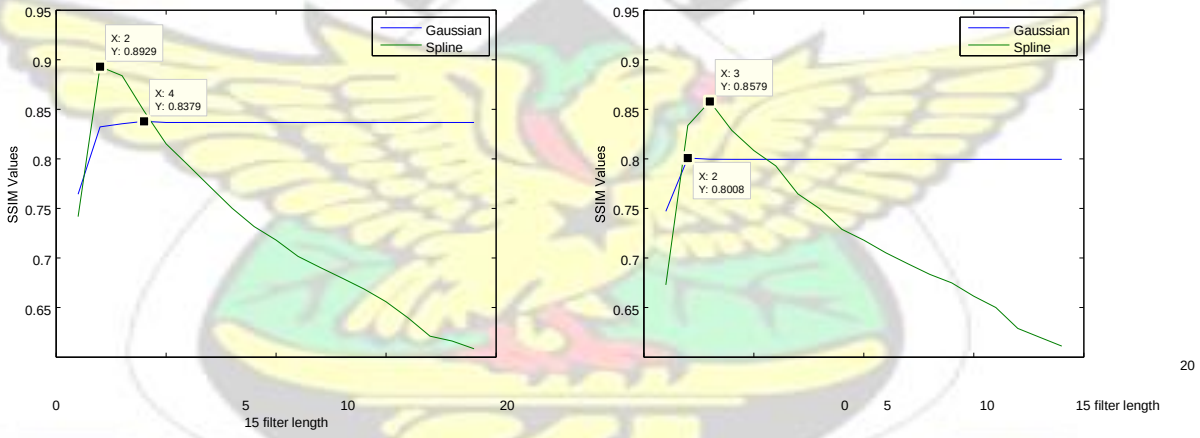
Table 4.15 gives a summary of the supposed optimal value for the filter size for both the spline and the Gaussian smoothing function at varying noise type. From the table, it is more clear that, a filter of size 3 is more stable irrespective of the smoothing function with motion blur noise expect with the 7 ˆ 7 mask size of the



Gaussian smoothing function.

(a) Motion Blur Noise (b) Gaussian Noise

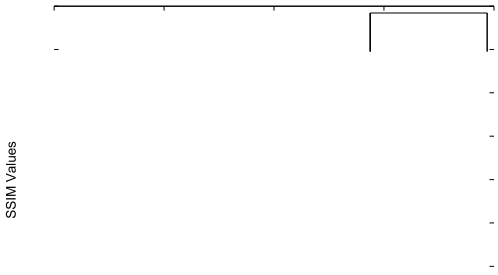
Figure 4.15: Efficiency of smoothing functions on speckle noise 9 ˆ 9

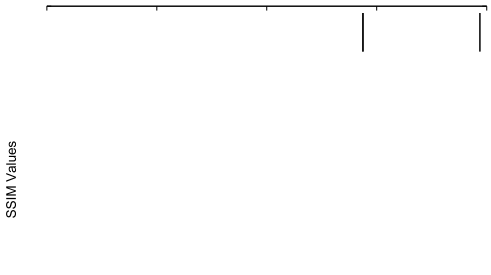


(c) Salt & Pepper Noise (d) Speckle Noise

Table 4.15: Optimal filter size selection

	Mask Size	Motion Blur	Gaussian	Salt & Pepper	Speckle
Spline	3 ˆ 3	3	9	9	9
	5 ˆ 5	3	9	9	9





	7 ˆ 7	3	7	7	7
	9 ˆ 9	3	7	5	7
Gaussian	3 ˆ 3	3	5	17	7
	5 ˆ 5	3	9	13	9
	7 ˆ 7	7	3	5	7
	9 ˆ 9	3	7	9	5

4.7.3 Optimal selection of the parameter σ for the Gaussian function

As stated in the previous section, one observe that, the value for the Gaussian smoothing function as provided in Table 4.15 is only optimal with the assumption that the second parameter (σ) is constant. In this subsection, a detail exploration is made as to whether these measures obtained are indeed optimal or rather sub optimal. In Table 4.16, the following observations were made in varying the σ parameter simultaneously with the filter size at various noise types and mask size.

Table 4.16: Optimal σ and filter size

Mask Size	Motion Blur		Gaussian		Salt & Pepper		Speckle	
	sigma	filter	sigma	filter	sigma	filter	sigma	filter
3 ^ 3	1	3*	6	9	4	9	4	15
5 ^ 5	9	3	4	9**	3	11	5	11
7 ^ 7	1	7*	4	11	3	19	3	11
9 ^ 9	1	3*	4	9	3	13	2	7

Only three values were seen to be optimal from Table 4.15 in relation to Table 4.16 as marked with single asterisks. The value marked with double asterisks was noted to be optimal and coincide with that of Table 4.15, however, a sigma value of 4 was more efficient compared to the sigma value of 1. All other values recorded in Table 4.16 are the most optimal values needed for performance analysis in the following subsection.

4.7.4 Performance analysis of the smoothing functions

Finally, in obtaining the required optimal values for the spline and Gaussian smoothing function, performance test is then carried out on the three selected smoothing function to ascertain which one function works better and under which condition it should be considered. From Table 4.17, it is clear that, the choice of a

single smoothing function to generalise images of varying distortions is not enough. The noise type has a lager effect on the selection of the smoothing function as opposed to the effect of the mask size. From the table, based on the optimal parameter selection,

Table 4.17: Performance measure using SSIM

	Mask Size	Motion Blur	Gaussian	Salt & Pepper	Speckle
3 × 3	Median	0.6654	0.1335	**0.6701**	0.2526
	Spline	**0.7143**	**0.5995**	0.6374	0.6127
	Gaussian	0.6992	0.5988	0.6464	**0.6208**
5 × 5	Median	0.8511	0.2032	**0.8654**	0.3852
	Spline	**0.8808**	0.7573	0.8008	0.7722
	Gaussian	0.8763	**0.7681**	0.8012	**0.7790**
7 × 7	Median	0.9099	0.2962	**0.9392**	0.5165
	Spline	**0.9441**	0.8066	0.8720	0.8441
	Gaussian	0.9385	**0.8225**	0.8773	**0.8560**
9 × 9	Median	0.9317	0.3949	**0.9416**	0.6056
	Spline	0.9527	0.8472	0.8880	0.8635
	Gaussian	**0.9543**	**0.8485**	0.9002	**0.8742**

the Gaussian smoothing function will always perform best on an image irrespective of the mask size provided the image is distorted with speckle noise. On the other hand, the Median smoothing function will also perform best in all the types when the image has salt & pepper distortion effect. With Motion Blur and Gaussian distortion type, a flipping effect was noted in that, the Spline smoothing will remain efficient for Motion Blur noise except for a mask size of 9. On the contrary, the Gaussian smoothing will also remain efficient except for a mask size of 3.

4.8 Experimental Result 5:- Detection of Hidden Edges

The ability of the newly proposed fractional convolution mask to detect hidden edges i.e. locations with some kind of discontinuous derivatives of image intensity was

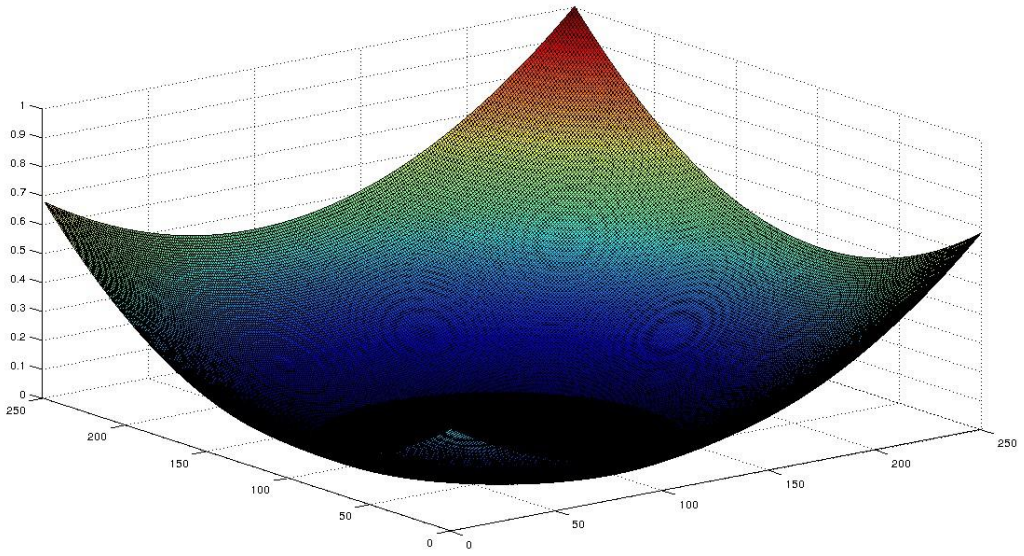
tested for. Detecting such edges has significant importance in applications for instance, aircraft surface manufacturing. In order to reduce turbulence, the surface of the body of an aircraft needs to be C^2 smooth to avoid generating singularities of the airflows. The common practice guaranteed to find defects in the surface of an aircraft body is to find locations with discontinuous first- or second- order derivatives reported as hidden edges. To simulate this concept, a surface with discontinuous firstorder derivatives is synthetically created. An example of such surface was developed using the equations below over $p x,y q \in r0,2.55s^2$.

$$z_1 p x,y q = \begin{cases} p x - 1 q^2 + p y - 1 q^2 - 0.5 & \text{if } p x - 1 q^2 + p y - 1 q^2 \geq 0.5 \\ 0 & \text{otherwise} \end{cases} \quad \text{“(4.8.1)}$$

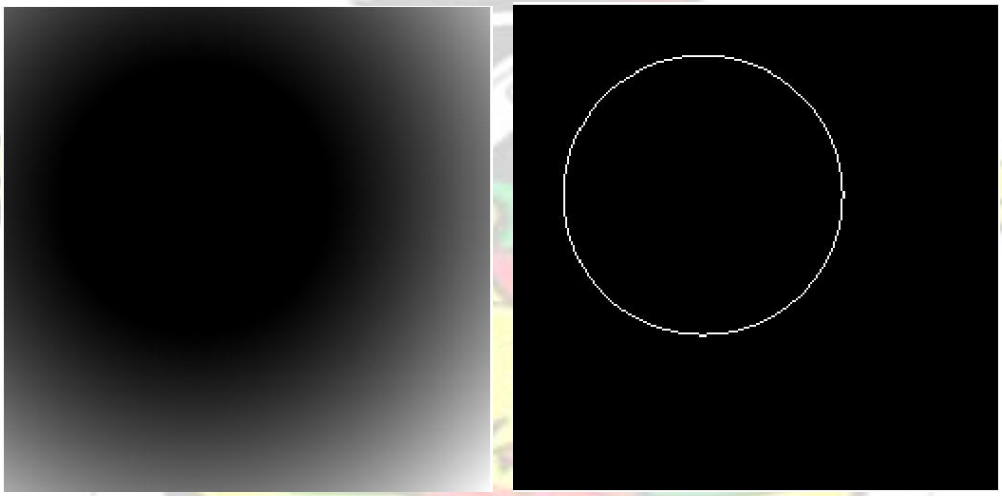
The three-dimensional (3D) graph and the 2D intensity image of the function $z_1 p x,y q$ are as shown in the left and the right panels of Figure 4.16 respectively. From the 2D intensity image, hardly will one see any edge. However, from the 3D surface plot, the places where the first-order derivatives are discontinuous can easily be seen . Taking another example for demonstration purpose, let

$$z_2 p x,y q = \begin{cases} p x + y - 1.2 q^2 - 0.5 & \text{if } p x + y - 1.2 q^2 \geq 0.5 \\ 0 & \text{otherwise} \end{cases} \quad \text{“(4.8.2)}$$

over $p x,y q \in r0,2.55s^2$. The surface plot, the intensity image of the function $z_2 p x,y q$ and the hidden edges are as shown in Figure 4.17.

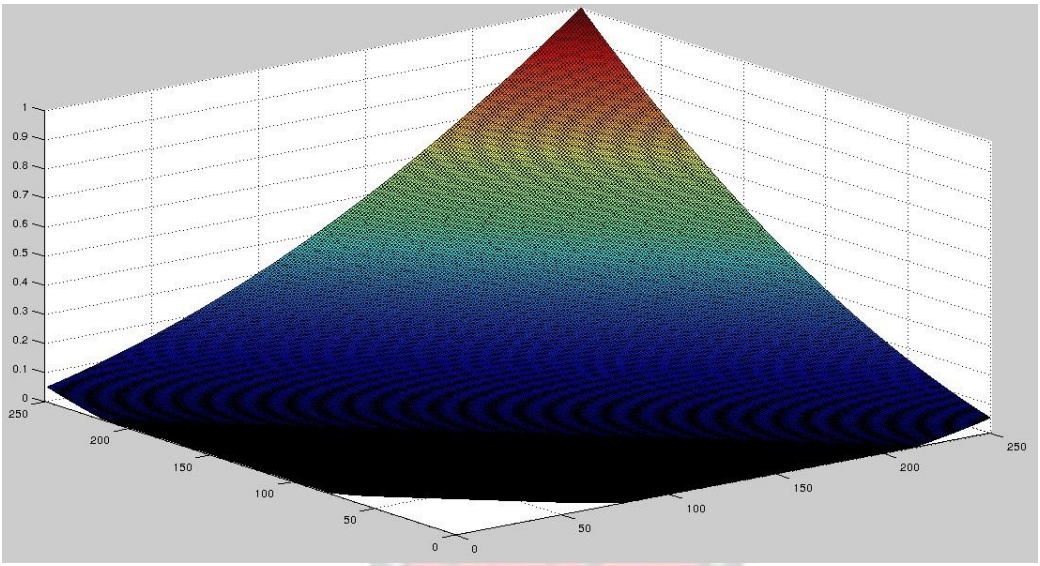


(a) Graph of First Equation

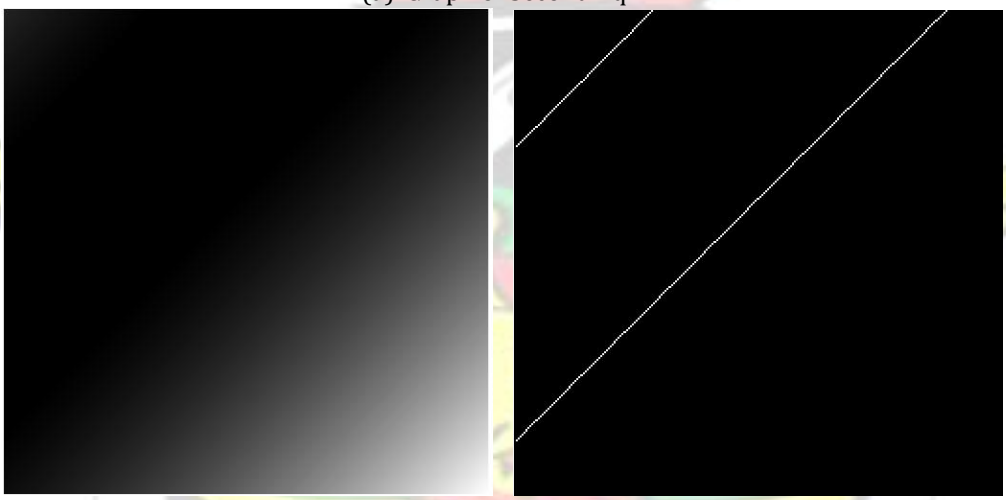


(b) Image (c) Edge Map

Figure 4.16: Result of Equation (4.8.1)



(a) Graph of Second Eqn



(b) Image (c) Edge Map

Figure 4.17: Result of Equation (4.8.2)

Chapter 5

CONCLUSION and RECOMMENDATIONS

5.1 Conclusion

In this work, a new fractional based convolution edge mask centred on StrivastavaOwas's derivative was developed for image analysis and synthesis which has equivalent computational efficiency as that of the classical based Canny approach. Unlike other constructions, this work extracted the mask, maintaining enough memory without the need for complicated optimization criteria. Both quantitative and qualitative comparative analysis with existing edge detectors was performed which demonstrated the effectiveness and efficiency of the proposed construction in detecting several edge types including step, Dirac edges and hidden edges found in the images used to perform the experiments. In addition, it was shown that the resulting mask is robust to noise. Finally, an object identification by segmentation using the resulting mask was performed which generated mostly significant improvement over the methods
studied.

As per the performance analysis and optimal selection of an image smoothing function on the newly created fractional convolution mask, a set of image smoothing functions as well as their corresponding noise type was proposed. From the experimental result it was noticed that the use of a single function as mostly practised by assumptions is not a good idea since it does not bring out the true nature of the image under study and may lead to loss of relevant information.

5.2 Recommendation for Further Studies

5.2.1 Recommendation

Based on the conclusion it is recommended that the proposed mask should be used for edge analysis in the field of signal and image processing. Users are entreated to set the parameter of the mask size to the least possible grid size at instances where more edge information is required and increase them where only prominent edge maps are needed especially in region segmentation.

5.2.2 Further Studies

1. A theoretical analysis to investigate the quantum of memory lost in the Tiansi mask compared to that of the proposed mask should be considered.
2. The concept of dynamical system should also be considered in quest to identify and classify fractal edges found by the proposed mask as well as the order of the found fractals.
3. The proposed mask should be tried on images from very high resolution image acquisition devices such as satellite images.
4. The experimental analysis should be done in two phases. That is: pre-smoothing and post-smoothing to test the performance of the proposed mask.
5. Finally, the order of the proposed fractional mask should be increased into the Laplacian range to check for its robustness to noise factors.

Appendix A

MATLAB Code for Performance Test

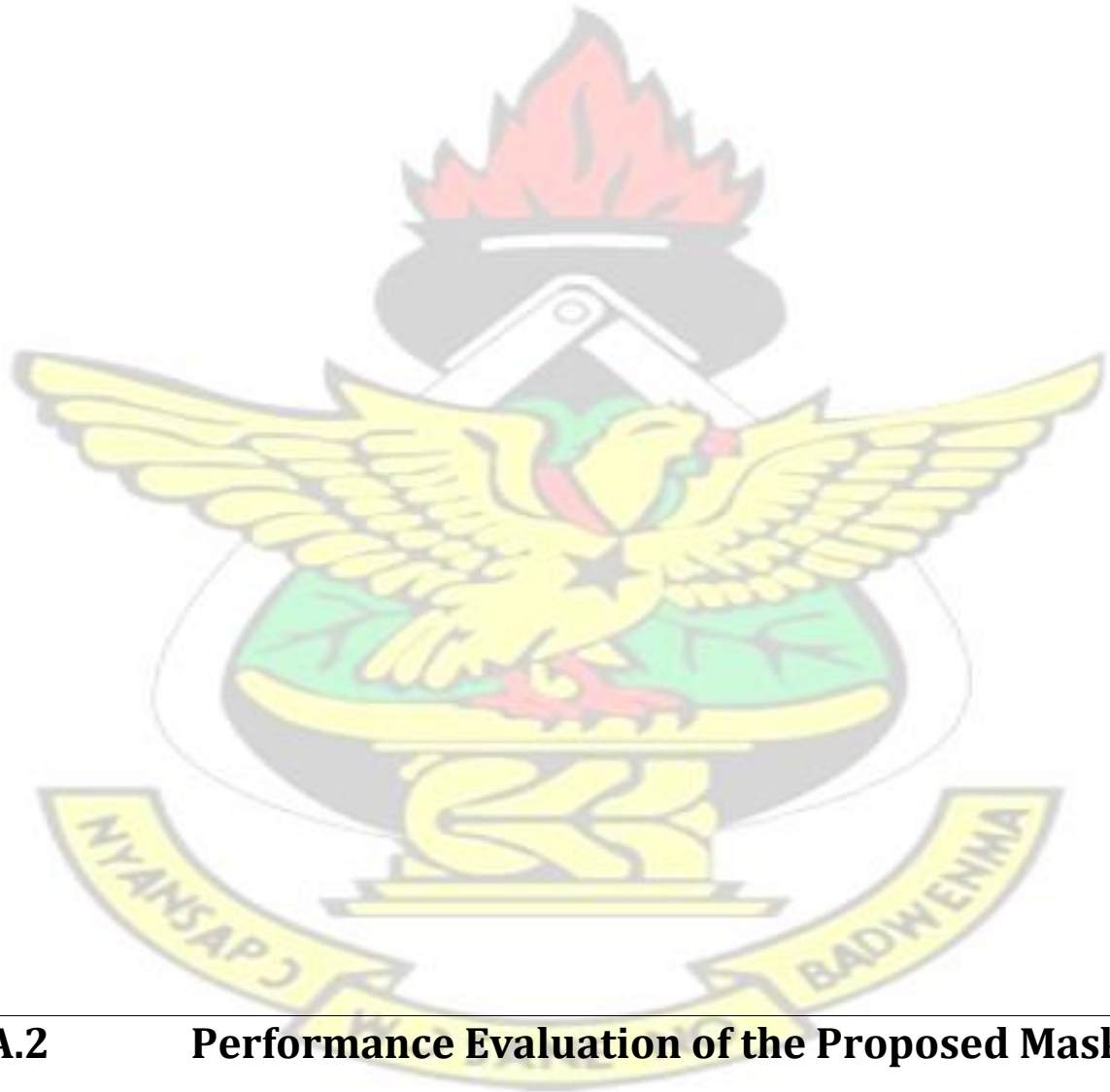
A.1 Fractional Implementation of Canny Mask

```
function e = fractionalEdge2DMain(origImg,alphav,width,sigma)
% implementation of the proposed fractional mask base on the ...
concept of
%canny algorithm.
%INPUTS: origImg = image
%       alphav = fractional order
%       width = size of fractional mask
%       sigma = size of guassian filter
%OUTPUT: e = binary image containing the edge map
%code modified by Justice Kwame Appati: justicekwameappati@gmail.com %date of
modification: 18th Dec. 2014

 if isa(origImg,'single') && isa(origImg,'double')
 origImg = im2single(origImg);
 end
 [rol,cow] = size(origImg);
 e = false(rol,cow);

PercentOfPixelsNotEdges = .7;
threshRatio = .4;

tt = -width:width;
siga = sigma^2;
gau = exp(-(tt.*tt)/(2*siga))/(2*pi*siga);

[x,y]=meshgrid(tt,tt);
```

```
dgau2D=(-alphav/(gamma(1-alphav)))*x.*((x.^2 + ...
    y.^2).^(-alphav/2-1));
dgau2D(width+1,width+1) = 0;

dgau2D = dgau2D/max(dgau2D(:));
imgSmooth=imfilter(origImg,gau,'conv','replicate'); % run ... across rows
imgSmooth=imfilter(imgSmooth,gau','conv','replicate'); % and ...
    along columns

%apply directional derivatives
imgX = imfilter(imgSmooth, dgau2D, 'conv','replicate');
imgY = imfilter(imgSmooth, dgau2D', 'conv','replicate');

    imgMag = sqrt((imgX.*imgX) + (imgY.*imgY));
    magmax = max(imgMag(:));
    if magmax>0
    imgMag = imgMag / magmax;       % normalize
    end

% Select the thresholds
counts=imhist(imgMag, 64);
threshHigh = find(cumsum(counts) > ...
    PercentOfPixelsNotEdges*rol*cow,1,'first') / 64;
threshLow = threshRatio*threshHigh;
thresh = [threshLow threshHigh];


    strongInd = [];
    for dir = 1:4
    localMaxInd = cannyFindLocalMaxima(dir,imgX,imgY,imgMag);
    weakInd = localMaxInd(imgMag(localMaxInd) > threshLow);
    e(weakInd)=1;
    strongInd = [strongInd; weakInd(imgMag(weakInd) > ...
        threshHigh)]; %#ok<AGROW>
    end

    if isempty(strongInd)
    strongR = rem(strongInd-1, rol)+1;
    strongC = floor((strongInd-1)/rol)+1;
    e = bwselect(e, strongC, strongR, 8);
    e = bwmorph(e, 'thin', 1);
    end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%
%    Local Function : cannyFindLocalMaxima function localMaxInd = ...
     cannyFindLocalMaxima(direction,iix,iiy,imgMag)

[rol,cow] = size(imgMag);



    switch direction
    case 1
    indx = find((iiyd0 & iix>-iiy) | (iiyĕ0 & iix<-iiy));
    case 2
    indx = find((iix>0 & -iiyĕiix) | (iix<0 & -iiydiix));
    case 3
    indx = find((iixd0 & iix>iiy) | (iixĕ0 & iix<iiy));
    case 4
    indx = find((iiy<0 & iixdiiy) | (iiy>0 & iixĕiiy)); end

      % Exclude the exterior pixels
      if isempty(indx)
      v = mod(indx,rol);
      extIdx = (v==1 | v==0 | indxdrol | (indx>(cow-1)*rol));
      indx(extIdx) = [];
      end

ixv = iix(indx);
iyv = iiy(indx);
magGrad = imgMag(indx);

    % Do the linear interpolations for the interior pixels
    switch direction
    case 1
    d = abs(iyv./ixv);
    magGrad1 = imgMag(indx+rol).*(1-d) + imgMag(indx+rol-1).*d;
    magGrad2 = imgMag(indx-rol).*(1-d) + imgMag(indx-rol+1).*d;
    case 2
    d = abs(ixv./iyv);
    magGrad1 = imgMag(indx-1).*(1-d) + imgMag(indx+rol-1).*d;
    magGrad2 = imgMag(indx+1).*(1-d) + imgMag(indx-rol+1).*d;
    case 3
    d = abs(ixv./iyv);
    magGrad1 = imgMag(indx-1).*(1-d) + imgMag(indx-rol-1).*d;
    magGrad2 = imgMag(indx+1).*(1-d) + imgMag(indx+rol+1).*d;
```

```
109         case 4
110         d = abs(iyv./ixv);
111         magGrad1 = imgMag(indx-rol).*(1-d) + imgMag(indx-rol-1).*d; 112         magGrad2 =
imgMag(indx+rol).*(1-d) + imgMag(indx+rol+1).*d;
113     end
114     localMaxInd = indx(magGrad≥magGrad1 & magGrad≥magGrad2);
```

A.2 Performance Evaluation of the Proposed Mask

```
function out = performance() clear, clc,
close all hidden
  kernelSize = 2; %this determine the size of the smoothing filter ...
        for FDE
  sigma = 0.6; %this determine the size of the Canny mask
  orders = 0.1:0.1:0.9; % order of FD 0 < alpha < 1

        %% Reading the Image and performing 2D operations
        % I = im2double(imread('nonlinear.png'));
        I = im2double(imread('linear.png'));
        % I = im2double(dicomread('medical.dcm'));
        if length(size(I))>2
        I = rgb2gray(I);
        end
        if or(min(I(:))‰0,max(I(:))‰1)
        I = (I-min(I(:)))/(max(I(:))-min(I(:)));
        end
        SD=15;
        N=imnoise(I,'gaussian',0,(SDˆ2/255ˆ2));
        %% comparative stage
        bwC = edge(I,'cannyold',[],sigma);
        bwCN = edge(N,'cannyold',[],sigma);

       for ip = 1:length(orders)
       alpha = orders(ip);
       %% the fractional operations here
       bw = fractionalEdge2DMain(I,alpha,kernelSize,sigma);
       bwN = fractionalEdge2DMain(N,alpha,kernelSize,sigma);
       bwT = fractionalTiansi(I,alpha,kernelSize,sigma);
       bwTN = fractionalTiansi(N,alpha,kernelSize,sigma);
       if ip==1
       datg =      {double(bwC),double(bw),double(bwT)};
       datgN =     {double(bwCN),double(bwN),double(bwTN)};
       else
       datg =      {double(bw),double(bwT)};
       datgN = {double(bwN),double(bwTN)};
       end
       signalData = addNoisePerform(dat_g,dat_gN)
       datW(ip,:) =signalData(end,:);
       end
       datWC(1,:) = signalData(1,:);
       signalData.dataWC = datWC;
       signalData.dataW = datW;
       signalData.dataWT = datWT;
```

```
        dd = strcat('DataCompareS',num2str(kernelSize),'.mat');
        save(dd,'signalData') out = signalData;

%% ========== OTHER REQUIRED FUNCTIONS=============
function noiseLevel = addNoisePerform(img,imgN)
mn = length(img);
```


```
PSNRv = zeros(mn,1);
MSEv = zeros(mn,1);

for i = 1:mn
[MSEv(i), RMSEv(i)] = RMSE2(img{i},imgN{i});
[PSNRv(i),SNRv(i)] = psnr(img{i},imgN{i});
[SSIMv(i), SSIMmap] = ssim(imgN{i},img{i});
end
noiseLevel = [MSEv,PSNRv];
```


Appendix B MATLAB

Code for Noise Immunity


```
function noiseImmunity()
%Matlab code written By Justice Kwame Appati
%email: justicekwameappati@gmail.com
%Date: 6th March, 2015
%This code compute the SSIM of image based on the noise level, %noise type and the fractional order parameter set.

%parameter definitions
noiseLevel = 20:5:45;
noiseType = {'motion blur','gaussian','salt & pepper','speckle'};
fdeSet = {[0.4,1],[0.9,3],[1,4]}; %mask size
fdeOrder = 0.5;%0.1:0.1:0.9;
imageSet = {'linear.png','nonlinear.png','medical.dcm'};

%result output formatting and display
%table output configuration
for i = 1:length(fdeSet)
if i==1%for 3 x 3
tabCanny33 = ...
zeros(length(imageSet)*length(noiseType),length(noiseLevel));
tabFDE0 5 3 3 = tabCanny3_3;
elseif i==2%for 7 x 7 tabCanny77 = ...
zeros(length(imageSet)*length(noiseType),length(noiseLevel));
tabFDE0 5 7 7 = tabCanny7_7;
elseif i==3%for 9 x 9 tabCanny99 = ...
zeros(length(imageSet)*length(noiseType),length(noiseLevel));
tabFDE0 5 9 9 = tabCanny9_9;
end
end

%start of main section
idx = 1;
```

```
32                          for i=1:length(fdeSet)
33                          %configure mask size here
34                          sigma = fdeSet{i}(1);
35                          kernelSize = fdeSet{i}(2);
36                          for j=1:length(noiseType)
37                          %noise type configuration
38                          Ntype = noiseType{j};
39                          for k=1:length(imageSet)
40                          %image selection and configuration
41                          if k==length(imageSet)
42                          I = im2double(dicomread(imageSet{k}));
43                          else
44                          I = im2double(imread(imageSet{k}));
45                          end
46                          %image formating into grayscale and normalisation
47                          if length(size(I))>2
48                          I = rgb2gray(I);
49                          end
50                          if or(min(I(:))‰0,max(I(:))‰1)
51                          I = (I-min(I(:)))/(max(I(:))-min(I(:)));
52                          end
53                          %add noise level (SD values) after image is ok for ...
                        experimental work
54                          for nL = 1:length(noiseLevel)
55                          nSD = (noiseLevel(nL)^2/255^2);
56                          [ , N] = noisyImageGeneration(I, 0, nSD, 0.01, ...
                        2, 2, Ntype);
57                          %apply the edge detectors
58                          bwC = edge(I,'cannyold',[],sigma);
59                          bwCN = edge(N,'cannyold',[],sigma);
60                          for ip = 1:length(fdeOrder)
61                          alpha = fdeOrder(ip);
62                          bw = ...
                                                fractionalEdge2DMain(I,alpha,kernelSize,sigma);
63                          bwN = ...
                                                fractionalEdge2DMain(N,alpha,kernelSize,sigma);
64                          %    bwT = ...
                        fractionalTiansi(I,alpha,kernelSize,sigma);%,'');
65                          %    bwTN = ...
                        fractionalTiansi(N,alpha,kernelSize,sigma);%,'');
66                          %calculate for the SSIM
67                          if ip==1
68                          datg = ...
                                                {double(bwC),double(bw)};%,double(bwT)};
69                          datgN = ...
```

```
                                                {double(bwCN),double(bwN)};%,double(bwTN)};
70                                          else
71                                          datg = {double(bw)};%,double(bwT)};
72                                          dat_gN = {double(bwN)};%,double(bwTN)};
73                                          end
74                                          signalData = addNoisePerform(dat_g,dat_gN); 75 %feed the
                                            tables with SSIM values from ...
                                              signal Data
76                                          %    idx = idx + 1;
77                                          if i==1%for 3 x 3
78                                          tabCanny33(idx,nL) = signalData(1);
79                                          tabFDE0 5 3 3(idx,nL) = signalData(2);
80                                          elseif i==2%for 7 x 7
81                                          tabCanny77(idx,nL) = signalData(1);
82                                          tabFDE0 5 7 7(idx,nL) = signalData(2);
83                                          elseif i==3%for 9 x 9
84                                          tabCanny99(idx,nL) = signalData(1);
85                                          tabFDE0 5 9 9(idx,nL) = signalData(2);
86                                          end
87                                          end
88                                          end
89                                          idx = idx + 1;
90                                          end
91                                          end
92                                          idx = 1;
93                                          end
94                                          disp('+++Display of Canny 3 x 3 mask')
95                                          disp(tabCanny33)
96                                          disp('+++Display of Proposed 3 x 3 mask')
97                                          disp(tabFDE0 5 3 3)
98
99  disp('+++Display of Canny 3 x 3 mask')
100 disp(tabCanny77)
101 disp('+++Display of Proposed 3 x 3 mask')
102 disp(tabFDE0 5 7 7)
103
104 disp('+++Display of Canny 3 x 3 mask')
105 disp(tabCanny99)
106 disp('+++Display of Proposed 3 x 3 mask')
107 disp(tabFDE0 5 9 9)
108 disp('The first 3 rows = Motion Blur Noise, next 3 rows = ... Gaussian White Noise,next 3 rows = Salt
    and Pepper Noise and ... the last 3 rows = Speckle Noise')
109
110
111 %SSIM and MAP evaluation function
```

```
function noiseLevel = addNoisePerform(img,imgN)
mn = length(img);
SSIMv = zeros(mn,1);

for i = 1:mn
[SSIMv(i), ] = ssim(imgN{i},img{i});

end
noiseLevel = SSIMv;
```


Appendix C

MATLAB Code for Region Growing Segmentation


```
function regionGrowingSegmentation()
% This function is used for the segmentation of image based on ... the concept of
% single seed region growing. The red line represent the outer ...
    boundary
% whiles the green line is for the inner boundary if there is any.
% Usage: Edit line 8 with your own image and run the ...
    application. Click on
% the first image that appears as your region of interest. Hit ... Enter and
% wait for the segmentated result.
% Date: 8th March, 2015, justicekwameappati@gmail.com

    I = im2double(imread('medtest.png'));
    % I = im2double(dicomread('medical019.dcm'));
    binaryImage = edge(I,'canny');
    % binaryImage = fractionalEdge2DMain(I,0.5,4,1);
    pt = 5;
    binaryImageO = bwareaopen(binaryImage, pt);
    epsilon = 0.05e-10;
    mseer = mse(im2double(binaryImage),im2double(binaryImageO));
    atemptsMax = 5; atemptsMin = 0;

     while or(mseer ‰ epsilon,atemptsMin‰atemptsMax)
     temp = binaryImageO;
     epsilon = mseer;
```

```
24          pt = pt + 5;
25          binaryImageO = bwareaopen(temp, pt);
26          mseer = mse(im2double(temp),im2double(binaryImageO));
27          if mseer==epsilon
28          atemptsMin=atemptsMin+1;
29          else
30          atemptsMin=0;
31          end
32          end
33
34          J = regiongrowing(binaryImageO);
35          [B, ,N] = bwboundaries(J);
36          imshow(J+I); hold on;
37          for k=1:length(B),
38          boundary = B{k};
39          if(k > N)
40          plot(boundary(:,2), boundary(:,1), 'g','LineWidth',3);
41          else
42          plot(boundary(:,2), boundary(:,1), 'r','LineWidth',3);
43          end
44          end
```

REFERENCES

Abel, N. H. (1826), "Auflosung einer mechanischen Aufgabe," *J. fur reine und angew. Math.*, 1, 153–157.

Abel, N. H. (1881), "Solution de quelques problemes a laide dintegrales d efinies," *Oeuvres Completes*, 1, 16–18.

Ambrosio, L. and Tortorelli, V. M. (1992), "On the approximation of free discontinuity problems," *Boll. Un. Mat. Ital. B*, 7, 105–123.

Baleanu, D., Diethelm, K., Scalas, E., and Trujillo, J. J. (2012), "Fractional Calculus Models and Numerical Methods," *Series onComplexity, Nonlinearity and Chaos, World Scientific, Hacken- sack, NJ, USA*, 3.

Barash, D. and Comaniciu, D. (2004), "A Common Framework for Nonlinear Diffusion, Adaptive Smoothing, Bilateral Filtering and Mean Shift," *. Image and Vision Computing*, 22, 7381.

Bayer, R. and Leveque, R. (1992), "Analysis of one-dimensional model for the immersim boundary method," *SIAM J. Num. Anal.*, 29, 332–364.

Boyle, R. and Thomas, R. (1988), "Computer Vision: A first course," *Blackwell Scientifi Publication*, pp. 32 – 34.

Byers, S. and Raftery, A. (1998), "Nearest-Neighbor Clutter Removal for Estimating Features in Spatial Point Processes," *Journal of American Statistical Association*, 93, 577–584.

Canny, J. (1986), "Computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell, PAMI-8:*, pp. 679 – 698.

Caputo, M. (1967), "Linear models of dissipation whose Q is almost frequency independent II," *Geophys. J. Royal astr. Soc.*, 13, 529539.

Cayley, A. (1880), "Note on Riemanns paper Versuch einer allgemeinen Auflosung der Integration und Differentiation," *Math. Ann.*, 16, 81–82.

Chan, T. F. and Vese, L. A. (2001), "Active Contours Without Edges," *IEEE TRANSACTIONS ON IMAGE PROCESSING*, 10.

Davies, E. (1990), *Machine Vision: Theory, algorithms and practicalities.*

Deriche, R. (1990), "Fast algorithms for low-level vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, 12, 78–87.

Euler, L. (1738), "De progressionibus transcendendibus seu quarum termini generales alge- braicae dari nequent," *Comm. Acad. Sci. Petropolitanae*, 5, 36–57, Translated to english by S. G. Langton, University of San Diego, www.sandiego.edu/langton.

Evans, L. C. and Gariepy, R. (1969), *Measure Theory and Fine Properties of Functions*, CRC Press.

Fourier, J. B. J. (1822), *The orie analytique de la chaleur, pages 561562.*, Didot, Paris.

Gao, C. and Zhou, J. (2011), "Image Enhancement Based on Quaternion Fractional Directional Differentiation," *Acta Automatica Sinica*, 37, 150–159.

Gao, C., Zhou, J., Hu, J., and Lang, F. (2011a), "Edge Detection of Color Image Based on Quaternion Fractional Differential," *IET Image Processing*, 5, 261–272.

Gao, C., Zhou, J., Zheng, X., and Lang, F. (2011b), "Image Enhancement Based on Improved Fractional Differentiation," *Journal of Computational Information Systems*, 7, 257–264.

Gao, C., Zhou, J., and Zhang, W. (2014), "Edge Detection Based on the Newton Interpolations Fractional Differentiation," *The International Arab Journal of Information Technology*, 11, 223–228.

Garg, V. and Singh, K. (2012), "An improved Grunwald-Letnikov fractional differential mask for image texture enhancement," *International Journal*, 3, 130135.

Gonzalez, R. C. and Woods, R. E. (2000), *Digital Image Processing*, Pearson Education Asia Pie Ltd.

Grunwald, A. K. (1867), "Ueber begrenzte Derivationen und deren Anwendung," *Z. angew. Math. und Phys.*, 12, 441–480.

Guo, W. and Huang, F. (2008), "A local mutual information guided denoising technique and its application to self-calibrated partially parallel imaging," *in MICCAI, Part II, Lecture Notes in Comput. Sci. 5242, Springer, Berlin*, pp. 939–947.

Guo, W. and Lai, M.-J. (2013), "Box Spline Wavelet Frames for Image Edge Analysis," *SIAM J. Imaging Science*, 6, 1553–1578.

Hamid, A. J. and Rabha, W. I. (2013), "Texture Enhancement for Medical Images Based on Fractional Differential Masks," *Discrete Dynamics in Nature and Society*, pp. 1–10.

Hilfer, R. (2000), *Applications of Fractional Calculus in Physics,*, World Scientific, Singapore.

Ibrahim, R. W. (2011), "On generalized srivastava-owa fractional operators in the unit disk," *Advances in Difference Equations*, pp. 1–10.

Jalab, H. A. and Ibrahim, R. W. (2012), "Texture feature extraction based on fractional mask convolution with cesaro means for content-based image retrieval," *12th Pacific Rim International Conference on Trends in Artificial Intelligence (PRICAI 12)*, p. 170179.

Jalab, H. A. and Ibrahim, R. W. (2013), "Texture enhancement based on the savitzky-golay fractional, differential operator," *Mathematical Problems in Engineering*.

Kanbur, M., Narin, I., Ozdemir, E., Dinc, E., and Baleanu, D. (2010), "Fractional wavelet transform for the quantitative spectral analysis of two-component system," *New Trends in Nanotechnology and Fractional Calculus Applications*, pp. pp. 321331, Springer, New York, NY, USA,.

Kass, M., Witkin, A., and Terzopoulos, D. (1988), "Snakes: Active contour models," *Internatioal Journal of Computer Vission*, pp. 321–331.

Koenderink, J. and van Doom, A. J. (1992), "Generic neighborhood operators," *IEEE Trans. Pattern Anal. Mach. Intell.*, 14, 597–605.

Lacroix, S. F. (1819), *Trait e du calcul diff erentiel et du calcul inte gral*, pp. 409–410, Imprimeur-Libraire pour les sciences, Paris.

Lagrange, J. L. (1849), *Sur une: Nouvelle esp'ece de calcul relatif. A la diff erentiation et a linte gration des quantit es variables*, Gauthier-Villars, Paris.

Laurent, H. (1884), "Sur le calcul des de rive es a' indicies quelconques," *Nouv. Annales de Mathe matiques*, 3, 240–252.

Letnikov, A. V. (1868), "Theory of differentiation with an arbitrary index (Russian)," *Moscow Matem. Sbornik*, 3, 1–66.

Letnikov, A. V. (1872), "An explanation of the concepts of the theory of differentiation of arbitrary index (Russian)," *Moscow Matem. Sbornik*, 6, 413–445.

Liouville, J. (1832a), "Me moire sur le calcul des diffe rentielles a' indices quelconques." *J. lEcole Roy. Polyte chn.*, 13, 71–162.

Liouville, J. (1832b), "Me moire sur lintegration de le quation: (mx 2 + nx + p)d 2 x/dx 2 + (qx +pr)dy/dx + sy = 0 a' laide des diffe rentielles indices quelconques," *J. lEcole Roy. Polyte chn.*, 13, 163–186.

Liouville, J. (1832c), "Me moire sur quelques questions de ge ome trie et de me canique, et sur un nouveau genre de calcul pour re soudre ces questions," *J. lEcole Roy. Polyte chn*, 13, 1–69.

Loverro, A. (May 2004), "Fractional Calculus: History, Definitions and Application for the Engineer," *Journal on the Theory of Ordered sets and its Applications*, pp. 1–28.

Mahmoodi, S. (2012), "Edge detection filter based on MumfordShah Green function," *SIAM J. Imaging Sci.*, 5, 343–365.

Marazzato, R. and Sparavigna, A. C. (2009), "Astronomical image processing based on fractional calculus: the AstroFracTool," *Instrumentation and Methods for Astrophysics, http://arxiv.org/ abs/0910.4637v2.*

Marchaud, A. (1927), "Sur les de rive es et sur les diffe rences des fonctions des variables re elles," *J. math. pures et appl.*, 6, 371–382.

Marion, A. (1991), *An introduction to image processing*, Chapman and Hall.

Marr, D. and Hildreth, E. C. (1980), "Theory of edge detection," *Proc. R. Soc., B 207:*, pp. 187 – 217.

Mathieu, B., Melchior, P., Oustaloup, A., and C., C. (2003), "Fractional Differentiation for Edge Detection," *Signal Processing*, 83, 2421–2432.

McAndrew, A. (2004), "An Introduction to Digital Image Processing with Matlab Notes for SCM2511 Image Processing," *School of Computer Science and Mathematics; Victoria University of Technology*.

Mumford, D. and Shah, J. (1989), "Optimal approximation by piecewise smooth functions and associated variational problems," *Comm. Pure Appl. Math.*, 42, 557–685.

Nixon, M. S. and Aguado, S. A. (2002), "Feature extraction and image processing," *Butterworth and Heinemann*, pp. 31 – 65.

Oberst, U. (2007), "The fast fourier transfrom," *Control and Optimization, SIAM*, 46(2), 496 – 541.

Osher, S. and Sethian, J. A. (1988), "Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations," *J. Comput. Phys.*, 79, 12–49.

Oustaloup, A., Mathieu, B., and Melchior, P. (September 3 6, 1991), "Edge detection using non integer derivation," in *Presented at the IEEE European Conference on Circuit Theory and Design (ECCTD91), Copenhagen, Denmark*.

Pertz, G. H. and Gerhardt, C. J. (1849), *Dritte Folge Mathematik (Erster Band)*, chap. Briefwechsel zwischen Leibniz, Hugens van Zulichem und dem Marquis de lHospital, pp. 301–302, A. Asher & Comp.

Pertz, G. H. and Gerhardt, C. J. (1850a), *Dritte Folge Mathematik (Erster Band).*, chap. Briefwechsel zwischen Leibniz, Jacob Bernoulli, Joh. Bernoulli und Nic. Bernoulli., p. 225, A. Asher & Comp.

Pertz, G. H. and Gerhardt, C. J. (1850b), *Dritte Folge Mathematik (Erster Band).*, chap. Briefwechsel zwischen Leibniz, Jacob Bernoulli, Joh. Bernoulli und Nic. Bernoulli., p. 228, A. Asher & Comp.

Phillips, D. (1994), *Image Processing, Analyzing and Enhancing Digital Images*, Prentice Hall.

Pitas, I. and Venetsanopoulos, A. N. (1990), *Nonlinear Digital Filters: Principles and Applications*, Kluwer Academic Publishers, Dordrecht, The Netherlands.

Pu, Y., Wang, W., Zhou, J., Wang, Y., and Jia, H. (2008), "Fractional differential approach to detecting textural features of digital image and its fractional differential filter implementation," *Science in China F*, 51, 13191339.

Pu, Y., Zhou, J., and Yuan, X. (2010), "Fractional Differential Mask: A Fractional Differential Based Approach for Multi-Scale Texture Enhancement," *IEEE Transactions on Image Processing*, 19, 491–511.

Qi, D., Guo, F., and Yu, L. (2007), "Medical image edge detection based on omnidirectional multi-scale structure element of mathematical morphology," *in Proceedings of the IEEE International Conference on Automation and Logistics, IEEE, Washington, DC*, pp. 2281–2286.

Ross, B. (1977), "The development of fractional calculus 16951900," *Historia Math.*, 4, 75–89.

Samko, S., Kilbas, A., and Marichev, O. (1993), *Fractional Integrals and derivatives: Theory and Applications*, Gordon and Breach Science Publishers, Yverdon.

Sangwine, S. J. and Horne, R. E. N. (1998), *The Colour Image Processing Handbook*, Chapman and Hall.

Sanjeev, A., Simon, M., and Neil, G. (2002), "A tutorial on particle filters for online nonliner/non-gaussian Bayesian tracking," *IEEE Trans. On Signal Processing*, 50, 174–188.

Semmlo, J. L. (2004), "Biosignal and biomedical image processing," *Marcel Dekker Inc*.

Sonin, N. Y. (1869), "On differentiation with arbitrary index," *Moscow Matem. Sbornik*, 6, 138.

Stefan, W., Renaut, R. A., and Gelb, A. (2010), "Improved total variation-type regularization using higher-order edge detectors," *SIAM J. Imaging Sci.*, 3, 232– 251.

Stollnitz, E., DeRose, T., and Salesin, D. (2006), "Wavelets for computer graphics," *UK: Cambridge University Press*.

Tian, B., Yuan, H., and Yue, X. (2007), "Feature extraction algorithm for space targets based on fractal theory," in *Proceedings of the Second International Conference on Space Information Technology, SPIE Proc. 6795, SPIE, Bellingham, WA, , 679518.*

Tseng, C. C. (2006), "Design of FIR and IIR fractional order Simpson digital integrators," *Journal of Signal Processing*, 87, 1045–1057.

Umbaugh, S. E. (1998), *Computer Vision and Image Processing, A Practical Approach using CVIP tools*, Prentice Hall PTR.

Watanabe, Y. (1931), "Notes on the generalized derivative of Riemann-Liouville and its application to Leibnitzs formula, I and II." *To hoku Math. J.*, 34.

Weber, H. (1876), *Versuch einer Auffassung der Integration und Differentiation*, pp. 331–344, Teubner, Leibzig.

Weyl, H. (1917), "Bemerkungen zum Begriff des Differentialquotienten gebrochener Ordnung," *Vierteljshr. Naturforsch. Gesellsch. Zu rich*, 62, 296–302.

Xuehui, Y., Shangbo, Z., and Muhammad Abubakar, S. (2015), "Fractional nonlinear anisotropic diffusion with p-Laplace variation method for image restoration," *Multimedia Tools and Applications*, pp. 1–22.

Yang, Z., Lang, F., Yu, X., and Zhang, Y. (2011), "The construction of fractional differential gradient operator," *Journal of Computational Information Systems*, 7, 43284342.

Yuan, Y. (2000), "A fast algorithm for determinging the gaussian filtered mean line in surface metrology," *PrecisionEng.*, 24, 62–69.

Zhai, L., Dong, S., and Ma, H. (2008), "Recent methods and applications on image edge detection," *in Proceedings of the International Workshop on Geoscience and Remote Sensing, Shanghai, China*, pp. 332–335.

Zhang, L., Butler, A., and Sun, C. (2008), "Fractal dimension assessment of brain white matter structural complexity post stroke in relation to upper-extremity motor function," *Brain Research*, 1228, 229–240.

Zhang, Y., Pu, Y., and Zhou, J. (2010), "Construction of fractional differential masks based on Riemann-Liouville definition," *Journal of Computational Information Systems*, 6, 31913199.

Zhang, Y., Pu, Y. F., Hu, J. R., and Zhou, J. L. (2012), "A Class of Fractional-Order Variational Image Inpainting Models," *Appl. Math. Inf. Sci*, 6, 299–306.

Ziou, D. and Tabbone, S. (1997), "Edge Detection Techniques - An Overview," Tech. Rep. 195.

Ziou, D. and Tabbone, S. (1998), "Edge detection techniquesan overview, Internat," *J. Pattern Recognition Image Anal.*, 8, 537–559.