

# University of Cincinnati

Date: 3/11/2020

I. Inam Naser, hereby submit this original work as part of the requirements for the degree of Doctor of Philosophy in Computer Science & Engineering.

It is entitled:

**Rotated Polar Coordinate system, its Solid Vector Mathematical Operations, and 3-D Unsharp Masking and Gradient-Based Laplacian Spatial Filters of a Field of Vectors for Geometrical Edges Detection**

Student's name: Inam Naser

This work and its defense approved by:

Committee chair: Anca Ralescu, Ph.D.

Committee member: Bahaa I.K. AnsafPhD

Committee member: Kenneth Berman, Ph.D.

Committee member: Carla Purdy, Ph.D.

Committee member: Dan Ralescu, Ph.D.



36603

Rotated Polar Coordinate system, its Solid Vector Mathematical  
Operations, and 3-D Unsharp Masking and Gradient-Based  
Laplacian Spatial Filters of a Field of Vectors for Geometrical  
Edges Detection

A dissertation submitted to the  
Graduate School of the University of Cincinnati  
in partial fulfillment of the requirements for the degree of  
Doctor of Philosophy  
in the Department of Electrical Engineering and Computer Science  
of the College of Engineering & Applied Science

by

Inam Naser

M.S. University of Technology of Iraq, June 2008

B.S. University of Technology of Iraq, October 2001

Committee:

Professor Anca Ralescu, Advisor and Chair

Professor Carla Purdy

Professor Kenneth Berman

Professor Dan Ralescu

Professor Bahaa I. K. Ansaf

Spring 2020

# Abstract

Mathematical operations that are applicable to the three and hyper-dimensional space remain a key factor that act as a foundation for a wide range of applications of 3-d image processing and computer vision.

The recently proposed Solid Vector Subtraction opened the way to propose the high pass spatial filters of a field of vectors geometrical edge detectors.

Geometrical edge magnitude and direction detection has a wide spectrum of applications including object recognition and detection, autonomous navigation systems, and three-dimensional localization and mapping.

This dissertation proposed three research topics: (1) the first is the Solid Vector Addition, Multiplication, Division, Dot Product and Cross Product operations and their coordinate system definitions; (2) the second is the Gradient-based Laplacian of a field of vectors; and (3) the third is the Unsharp Masking of a field of vectors.

The problem statement of the first research proposal is to define the Solid Vector: (1) coordinate system (like whether it is Spherical or Polar or something else); (2) hyper dimensional space; (3) whole complementary set of addition, multiplication, division, dot product and cross Product operations.

The problem statement of the second research is that the definition of the Gradient-Based Laplacian of a field of vectors (used for detecting 3-D surface geometrical edges) depends on the definition of the Gradient of a field of vectors which was only just recently proposed. Also, the Gradient-Based Laplacian must comply with the typical rules for the Laplacian operator.

---

The problem statement of the third research is that the Cartesian Components-Wise mathematical subtraction operation used in the Unsharp Masking of a field of scalars is not useful to develop the Unsharp Masking of a field of vectors for geometrical edge magnitude and direction detection in point cloud surfaces.

The contributions of the first research are proposing novel definitions of: (1) the Rotated Polar coordinate system; (2) the 2-d, 3-d, hyper dimensional Rotated Polar coordinate system space; (3) the Solid Vector Addition; (4) the Solid Vector Multiplication; (5) the Solid Vector Division; (6) the Solid Vector Dot Product; and (7) the Solid Vector Cross Product.

The contributions of the second research are: (8) proposing a novel definition of the Gradient-Based Laplacian of the field of vectors; (9) doing its behavioral analyses when the absolute difference and when the signed difference are used; (10) doing its performance analyses on TUM data set, and its comparison study with the state of the art edge detectors on NYUD data set which shows that it is efficient.

The contributions of the third research are: (11) proposing a novel definition for the Unsharp Masking of a field of vectors; (12) doing its behavioral analyses; and (13) doing its performance analyses on TUM data set, and its comparison study with the state of the art edge detectors on NYUD data set.

During the work of this dissertation, five research papers have been written; four published and one submitted to international conferences and IEEE Explore.



# Dedication

I dedicate this achievement to my dad's memory, my mom, and with my deepest gratitude to my lovely family, my husband, Jalal Al-Anssari and my sweet kids: Ibrahim, Lana, and Tamara. Thanks Jalal. We had a long journey, despite it was difficult, but it was also filling with nice moments and I am sure after a couple of years when we remember this journey, we will smile. Thanks Ibrahim, Lana, and Tamara, your mom thanks you and at the same time apologizes because she spent the most important years of your life being busy with her study.

# Acknowledgments

I would like to thank the Iraqi Ministry of Higher Education for supporting the scholarship program that gave me the opportunity to pursue my Ph.D. in computer engineering from USA.

Special thanks to my adviser, Professor Anca Ralescu, for her continuous support throughout doing my research. She was always showing her interest and support to me and my family and trying to help us to pass our difficult times. Thanks to the members of the Dissertation Committee: Professor Carla Purdy, Professor Kenneth Berman, Professor Dan Ralescu, and Professor Bahaa I. K. Ansaf.

I would like to thank the University of Cincinnati and the College of Engineering and Applied Science (CEAS), especially department of Electrical Engineering and Computer Science (EECS) for all the introduced facilities and the faculty who has embraced me throughout my doctoral study mission. In addition, I would like to thank all the people who contributed and helped to the success of my academic career, from teaching to supervising in all the study stages.

Im thankful to my dad, mom, my uncle Abdulsatar, and all my siblings who were keeping pushing me to go forward throughout my long journey chasing my dream to finish my Ph.D.

I would like also to thank my uncle Hamed, and my cousins Ekhlas Falih and Mohammad Noor for financially guaranteed me during my scholarship program from the Iraqi Ministry of Higher Education.

# Table of Contents

	<b>Page</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Introduction . . . . .	3
1.2 The Objectives . . . . .	6
1.3 The Problem Statement . . . . .	7
1.4 The Methodologies . . . . .	7
1.5 The Contributions . . . . .	8
1.6 Dissertation Outline . . . . .	11
<b>2 Background Theories</b>	<b>12</b>
2.1 3D Camera . . . . .	14
2.2 Point Cloud . . . . .	15
2.3 Point Cloud Normal Vectors . . . . .	16
2.4 Neighborhood Search Method . . . . .	17
2.5 Edge Detection . . . . .	19
2.6 Unsharp Masking . . . . .	20
2.7 Gradient-Based Second Order Derivative . . . . .	21
2.8 Convolution . . . . .	23
2.9 Edges Types . . . . .	24
<b>3 The Solid Vector Operations</b>	<b>26</b>
3.1 Introduction . . . . .	28



3.2	The Literature Review . . . . .	32
3.2.1	The Cartesian Coordinate System and Its Component-Wise Cartesian Mathematical Vector Operations . . . . .	32
3.2.2	The Two-Dimensional Polar Coordinate System and Its Two-Dimensional Component-Wise Polar Mathematical Vector Operations . . . . .	35
3.2.3	The Three-Dimensional Spherical Coordinate System and Its Three-Dimensional Spherical Mathematical Vector Operations . . . . .	36
3.2.4	The Mathematical Solid Vector Subtraction Operation . . . . .	38
3.2.5	The Hyper-Plane . . . . .	38
3.3	The Rotated Polar Coordinate System . . . . .	39
3.3.1	The Rotated Polar Coordinate System Center . . . . .	40
3.3.2	The Rotated Polar Coordinate System Axes . . . . .	40
3.3.3	The Rotated Polar Coordinate System Plane/ Hyper-Plane . . . . .	42
3.3.4	The Rotated Polar Coordinate System Unit Vectors . . . . .	42
3.4	The Rotated Polar Mathematical Operations . . . . .	43
3.4.1	Solid Vector Addition Operation . . . . .	44
3.4.2	Solid Vector Multiplication Operation . . . . .	46
3.4.3	Solid Vector Division Operation . . . . .	48
3.4.4	Solid Vector Directional Dot Product Operation . . . . .	49
3.4.5	Solid Vector Directional Cross Product Operation . . . . .	50
3.5	The Conclusions . . . . .	51
<b>4</b>	<b>The Gradient-Based Laplacian of a Field of Vectors</b>	<b>53</b>
4.1	Introduction . . . . .	55
4.1.1	Literature Review . . . . .	57
4.2	Gradient-Based Laplacian Spatial Filter of a Field of Vectors . . . . .	58
4.2.1	Definition of the Gradient-Based Laplacian for two-dimensional field of vectors . . . . .	59

4.2.2	Definition of the Gradient-Based Laplacian for three-dimensional field of vectors . . . . .	61
4.3	Design of The Gradient-Based Laplacian Algorithm . . . . .	62
4.3.1	The Classification . . . . .	64
4.3.2	The Pseudo Code and The Complexity . . . . .	64
4.4	Behavioral Analyses . . . . .	64
4.5	Performance Analyses, Comparison Study, and Experimental Results . . . . .	68
4.5.1	Performance Analyses on The TUM data set . . . . .	70
4.5.2	Comparative Study on the NYUD data set . . . . .	71
4.6	Conclusions and Future Work . . . . .	74
<b>5</b>	<b>The Unsharp Masking of a Field of Vectors</b>	<b>75</b>
5.1	Introduction . . . . .	77
5.2	Literature Review . . . . .	79
5.2.1	Traditional Edge Detection Methods . . . . .	79
5.2.2	Unsharp Masking of a field of scalars . . . . .	79
5.2.3	The Gradient and Gradient-Based Laplacian of a field of vectors . . . . .	80
5.3	The Smoothing Spatial Filter of a Field of Vectors . . . . .	81
5.4	The Unsharp Masking of a Field of Vectors . . . . .	84
5.4.1	Definition of The Unsharp Masking . . . . .	84
5.4.2	Classification . . . . .	87
5.4.3	Pseudo Code and Complexity . . . . .	88
5.5	Behavioral analyses . . . . .	88
5.6	Performance Analyses, Comparison Study, and Experimental Results . . . . .	90
5.6.1	Performance Analyses on The TUM data set . . . . .	91
5.6.2	Comparative Study on the NYUD data set . . . . .	92
5.7	Conclusions and Future Work . . . . .	93

<b>6 The Conclusions and Future Work</b>	<b>97</b>
6.1 The Conclusions . . . . .	99
6.2 The future work . . . . .	101

# Table of Figures

	<b>Page</b>
Figure 2.1– This figure shows the time line of the Kinect projects [5]. . . . .	15
Figure 2.2– This figure shows the hardware specifications of the first- and second-generation Kinect models [5]. . . . .	15
Figure 2.3– Unsharp masking for edge detection [6]. . . . .	21
Figure 2.4– Shows variations of Laplacian filters: (a) top left, filter mask used to implement equation 2.4. (b) top right, mask used to implement the diagonal terms of the Laplacian. (c) and (d) are two other variations of the Laplacian mask [7]. . . . .	23
Figure 2.5– Shows one-dimensional edge profiles [8]. . . . .	25
Figure 3.1– Cited from [3], this figure shows the Solid Vector Subtraction operation. . . . .	39
Figure 3.2– This figure shows the Rotated Polar coordinate system. . . . .	40
Figure 3.3– shows the Solid Vector addition operation. . . . .	46
Figure 3.4– Solid Vector Directional Multiplication of a vector by an angular scalar. . . . .	48
Figure 3.5– Directional dot product. . . . .	50
Figure 4.1– Cross section of geometrical shape of the surface of the surfaced point cloud. (A) Angles Direction Measurements. (B1), No Change. (C1), Step. (D1), Ramp. (E1), Onset and End of Ramp. (F1), series of Step edges of Spiral of constant acceleration [3]. . . . .	58
Figure 4.2– Shows examples B, C, D, E, and F. . . . .	69

Figure 4.3– Edge detection results of the Gradient-Based Laplacian magnitude operators on the TUM data set. Black, red, green, light blue, and dark blue pixels represent the true positives, false positives caused by algorithm, false negatives caused by algorithm, false positive caused by noise, and false negative caused by noise, respectively. . . . . 71

Figure 4.4– This figure shows the edge detection results of the Gradient Based Laplacian magnitude operators on the NYU Depth data set. Black, red, green, light blue, and dark blue pixels represent the true positives, false positives caused by algorithm, false negatives caused by algorithm, false positive caused by noise, and false negative caused by noise, respectively. . . . . 73

Figure 5.1– Cross section of geometrical shape of the surface of the surfaced point cloud. (A) Angles Direction Measurements. (B1), No Change. (C1), Step. (D1), Ramp. (E1), Onset and End of Ramp. (F1), Step. This figure is cited from [3]. . . . . 81

Figure 5.2– This figure shows the **Solid Vector** subtraction operation [3]. . . . . 82

Figure 5.3– Illustrates the behavioral analyses of the Unsharp Masking applied on the areas of examples B,C,D,E and F. . . . . 94

Figure 5.4– Illustrates the Unsharp Masking edge directions of the Plane area (top-left), Step edge (top-right), and Ramp area (bottom-left). . . . . 95

Figure 5.5– This figure shows the edge detection results of the proposed Unsharp magnitude and direction operators on the TUM data set. Black, red, green, light blue, and dark blue pixels represent the true positives, false positives caused by algorithm, false negatives caused by algorithm, false positive caused by noise, and false negative caused by of noise, respectively. . . . . 95

Figure 5.6– This figure shows the edge detection results of the proposed Unsharp Mask-  
ing magnitude and direction operators on the NYU Depth data set. Black, red,  
green, light blue, and dark blue pixels represent the true positives, false positives  
caused by algorithm, false negatives caused by algorithm, false positive caused  
by noise, and false negative caused by of noise, respectively. . . . . 96

# Table of Tables

	<b>Page</b>
Table 4.1 Signal Area Under Mask . . . . .	63
Table 4.2 Dynamic Mask . . . . .	63
Table 4.3 Best performance of the proposed Gradient-Based Laplacian edge detector used to detect edges on the TUM data set. . . . .	71
Table 4.4 Best performance indicators obtained for each of the 13 detectors on the NYU Depth dataset. The first 12 performance indicators are obtained from Lejeune <i>et al.</i> [9]. . . . .	74
Table 5.1 Signal Area Under Mask . . . . .	84
Table 5.2 Static Mask . . . . .	84
Table 5.3 Best performance of the proposed Unsahrp Masking edge detector used to detect edges on the TUM data set . . . . .	92
Table 5.4 Best performance indicators obtained for each of the 13 detectors on the NYU Depth dataset. The first 12 performance indicators are obtained from Lejeune <i>et al.</i> [9]. . . . .	93

# Table of Algorithms

	<b>Page</b>
Algorithm 1    The Gradient-Based Laplacian Algorithm . . . . .	65
Algorithm 2    Unsharp Masking algorithm . . . . .	89



# **Chapter 1**

## **Introduction**

**Contents**

---

1.1	Introduction	3
1.2	The Objectives	6
1.3	The Problem Statement	7
1.4	The Methodologies	7
1.5	The Contributions	8
1.6	Dissertation Outline	11

---

## 1.1 Introduction

Mathematical operations that are applicable to the three and hyper-dimensional space remain a key factor that act as a foundation for a wide range of applications of 3-d image processing and computer vision—such as: (1) developing 3-d and hyper dimensional artificial intelligence and machine learning descriptors and algorithms, (2) detecting imaging similarities and dissimilarities, (3) controlling of 3-D objects rotations in the 3-D space—such as drones, or CAD virtual reality objects, (4) sculpturing 3-D objects using CAD applications (such as personalizing biomedical equipment like artificial bones using Scada machines), (5) developing indoor or outdoor navigation using vision-based GPS system, (6) developing vision-based obstacles avoiding applications (such as for UAVs, and elderly falling alerting system).

The Solid Vector Subtraction operation was recently proposed by Al-Anssar J. Naser I. and Ralescu A. [3].

The term Solid Vector is defined as that the vector is handled as one solid quantity by only two components that are: 1) the first is an only one magnitude (length) layer ( $r$ ), which keeps the vector as a one solid quantity; and 2) the second is an only one directional (angular) component layer ( $\theta$ ), which also keeps the vector as a one solid quantity and undivided into many angular directional components.

Being 2d, 3d and hyper dimensional space, and being Solid Vector make such a coordinate system and its mathematical operations quite useful for the above mentioned applications.

In literature, the typical mathematical coordinate systems and their mathematical operations are the component-wise Cartesian, Spherical and Polar coordinate systems.

The component-wise Cartesian coordinate system and its mathematical operations (that consist of a whole well defined complementary set of Subtraction, Addition, Multiplication, Division, Dot Product and Cross product operations) are applicable to the 2d, 3-d and hyperspace, but their problem is that they are not compliant to the Solid Vector definition, because they handle the vector as a separate layers of  $x, y, z, \dots$  axes which makes building machine learning and data mining algorithms processing and time consuming because in high dimensional data, they have to deal

with too many layers at the same time. Therefore, in the current research, the Cartesian is out of the interest.

The Spherical coordinate system and its mathematical operations are applicable to the 3-d and hyperspace, but their problem is that they are not compliant to the Solid Vector definition, because they handle the vector as: (1) two directional angular layers of  $\theta$  and  $\phi$  Spherical vector components, not only one as the Solid Vector definition stated; and (2) its vector magnitude  $r$  component. Therefore, in the current research, the Spherical is out of the interest.

The Polar coordinate system, here after refereed as the Fixed Polar coordinate system, and its mathematical operations are compliant to the above definition of the Solid Vector, because their vector has only two components: (1) one angular component  $\theta$  and (2) one length component  $r$ ; but their problem is that they are not extendable to the 3d or hyper-dimensional space, because: (1) they have a static coordinate system directional Zero vector that is *PoleY*, that is the lowest starting point of its  $\theta$  angular component, and (2) they are fixed to the two-dimensional still plane of their  $x, y$  axes. Therefore, in the current research, the Polar is out of the interest.

On the other side, a novel mathematical operation that is called the Solid Vector Subtraction operation was recently proposed in [3] by Al-Anssari J. *et al.*. This Solid Vector Subtraction operation is applicable to the 2d, 3-d and hyper dimensional space and is compliant to the above Solid Vector definition, because it handles the vector as a one solid quantity unit similarly to the Fixed Polar coordinate system, because it has: only one directional angular component  $\theta$  and only one magnitude  $r$  vector component.

On the other hand, geometrical surface edge magnitude and direction detection is still important for the applications 3-d object recognition and detection, 3-d Simultaneous Localization and Mapping, and many others. The traditional geometrical edge magnitude detectors (such as traditional Gradient, Laplacian, Gradient-based Laplacian and Unsharp Masking of a field of scalars operators) have used the *Component-Wise* vector operations; furthermore they could not be extended to detect geometrical edge directions.

According to [7] "Unfortunately, the gradient discussed in section 3.6.4 is not defined for vector

quantities... If accuracy is an issue, however, then obviously we need a new definition of the gradient applicable to vector quantities....As we just mentioned, the gradient we studied in Section 3.6.4 is applicable to a scalar function  $f(x, y)$ ; it is not applicable to vector functions”, where section 3.6.4 is the gradient for the field of scalars.

Inspired by the above quotes, J. Al-Anssari, I. Naser and Ralescu A. proposed in [3] their novel mathematical **Solid Vector** Subtraction operation. Based on this Solid Vector Subtraction operation, novel algorithms for the Gradient magnitude of a field of vectors and the Laplacian magnitude and direction of a field of vectors were proposed and implemented in [3] and [4], which were analogous to the Gradient of a field of scalars and the Laplacian of a field of scalars.

The recent advances in RGB-D cameras allow the capture of three-dimensional point clouds, images, of indoor environments. The raw original surfaced point cloud is organized in geometrical surfaces of one pixel width that reflects the environment that is captured by the depth sensor. Each of the point cloud pixels holds the normal vector that is perpendicular to its surface. Segmenting geometrical surfaces of the point cloud of the indoor environment into primitive shapes remains a big problem.

In the current dissertation, three researches proposals are proposed in three chapters as follows: (1) first is the Solid Vector operations research that extend the Solid Vector Subtraction operation and in which its complementary Solid Vector Addition, Multiplication, Division, Dot Product and Cross Product mathematical operations and their coordinate system definitions are proposed; (2) second, based on this Solid Vector Subtraction operation, novel 3d Gradient-based Laplacian of a field of vectors geometrical edge detector high pass spatial filter is proposed which is analogous to the Gradient-Based Laplacian of a field of scalars; and (3) third, based on this Solid Vector Subtraction operation, novel 3d Unsharp Masking geometrical edge detector high pass spatial filter is proposed, which is analogous to the Unsharp Masking of a field of scalars.

The surfaced Gradient point cloud also consists of pixels that are organized in the geometrical surfaces of one pixel width. Each of its pixels holds the value of the Gradient magnitude of the surfaced original point cloud. The Gradient point cloud is successfully classified into two types of

segments of pixels: (1) Plane edges, (2) Step and Ramp edges. In literature of the field of scalars of the gray-scale images, the 2nd derivative, the Laplacian and the Unsharp Masking can further segment the gray-scale intensity images into two segments: (1) Plane and Ramp intensity areas, and (2) Step intensity edges; so the proposed Gradient-based Laplacian and Unsharp Masking do as well.

## 1.2 The Objectives

As indicated above, there are three research proposals in this dissertation. The following are the objectives of each one of these research proposals:

1. The **objective** of the first research is to extend this Solid Vector Subtraction operation by proposing definitions of: (1) its coordinate system; (2) its hyper dimensional space; and (3) its other whole complementary set of Solid Vector Addition, Multiplication, Division, Dot Product, and Cross Product operations. This coordinate system, hereafter refereed as the Rotated Polar coordinate system and these mathematical operations, hereafter refereed as Solid Vector mathematical operation handle the vector as a one solid quantity unit, in addition to its property of being applicable to the 2d, 3d and hyper dimensional space.
2. The **objective** of the second research is to design and implement the second-order derivative, the Gradient-Based Laplacian, of a field of vectors in an analogous manner to the definition of the Gradient-Based Laplacian of a field of scalars, which typically classify two types of segments of the geometrical surfaces of the point cloud: (1) the Plane and Ramp areas; (2) and the Step edge areas.
3. The **objective** of the third research is to propose a novel Unsharp Masking of a field of Vectors that detects geometrical edge magnitude and direction of the 3-d point clouds, that is analogous to the traditional Unsharp masking of a field of scalars that is used for the intensity edge detection of the gray-scale images.

## 1.3 The Problem Statement

The problem statement of each one of the three research proposals are as follows:

1. The **problem statement** of the first research proposal is how to answer the following questions that were surfaced from the recent proposal of the Solid Vector Subtraction operation: 1) what is the definition of its coordinate system (like whether it is Spherical or Polar or something else), 2) what is the definition of its hyper dimensional space and 3) what are the definitions of its other whole complementary set of Solid Vector 1) Addition, 2) Multiplication, 3) Division, 4) Dot Product and 5) Cross Product operations.
2. The **problem statement** of the second research is that the recently proposed Gradient of a field of vectors high pass spatial filter is needed to define the Gradient-Based Laplacian, the 2nd derivative of a field of vectors edges detector that is analogous to the Gradient-Based Laplacian of the field of scalars; and that produces two types of segments: (1) Step edges areas segment; (2) Plane and Ramp areas segment of the point cloud 3-D surfaces of the environment captured by the depth sensor.
3. The **problem statement** of the third research is that the Cartesian Components-Wise mathematical subtraction operation used in the Unsharp Masking of a field of scalars, is not useful to develop the Unsharp Masking of a field of vectors for geometrical edge magnitude and direction detection in point cloud surfaces.

## 1.4 The Methodologies

The following are the **methodologies** to address the problem statements of each of the three researches of current dissertation:

1. The **methodologies** to address the problem statement of the first research are:

- 1) a) One of the operands vectors of the Solid Vector Subtraction operation is made as the first axis of the coordinate system that is the Directional Zero vector that replaces the *PoleY* of the Fixed Polar, and that is the lowest starting point of the angular component  $\theta$ ; and b) the Directional Sign Axis of the Solid Vector Subtraction operation is made as the second axis of the coordinate system, that replaces the  $x$  axis of the Fixed Polar; in order to get a rotated version of the Fixed Polar that here after, in the current research, referred to it as the Rotated Polar coordinate system that is implemented in the 3-D dimensional space and that is not fixed to the still plane of the  $x, y$  axes.
  - 2) A pair of vectors are specified as components of a Hyper Rotated Polar when the Solid Vector Subtraction operations between them and their Directional Zero vector produce the same Directional Sign Axis (including both of its opposite directions the positive and negative), in order to define its extension to the hyper dimensional space.
  - 3) The triangulation is used in order to propose the Solid Vector 1) Addition, 2) Multiplication, 3) Division, 4) Dot Product and 5) Cross Product operations.
2. The **methodology** of the second research is to use the recently proposed Gradient of a field of vectors high pass spatial filter as a foundation and preliminary step to propose the Gradient-based Laplacian of a field of vectors geometrical edge detector.
  3. The **methodology** of the third research is to use the mathematical **Solid Vector** Subtraction operation to propose the Unsharp Masking of a field of vectors, instead of the Cartesian Component-Wise subtraction operation that is used in the Unsharp Masking of the field of scalars.

## 1.5 The Contributions

There are thirteen contributions items in this dissertation that are distributed into three researches as follows:



The **contributions** of the first research are proposing:

1. A novel definition of a Rotated Polar coordinate system based on two novel axes: 1) first axis is the dynamic Operation-Based Zero Vector; and 2) second axis is the Directional Sign Axis of the Solid Vector Subtraction operation;
2. A novel definition of the 2-d, 3-d, hyper dimensional Rotated Polar coordinate system space;
3. A novel definition of the Solid Vector Addition;
4. A novel definition of the Solid Vector Multiplication;
5. A novel definition of the Solid Vector Division;
6. A novel definition of the Solid Vector Dot Product;
7. A novel definition of the Solid Vector Cross Product.

The **contributions** of the second research are:

8. Proposing a novel algorithm of the Gradient-Based Laplacian of a field of vectors (used for detecting 3-D surface geometrical edges) that is based on the novel algorithm of the Gradient of a field of vectors, that was proposed in [3] research. In the Gradient-Based Laplacian algorithm, the first predefined operation of the convolution is defined as the absolute scalar difference of the Gradient values of the neighbor pixels from the Gradient value of the center pixel, in contrast to the existing work [10] of gray-scale edge detection, in which this is defined as the signed Gradient difference operation.
9. Doing behavioral analysis for the Gradient-based Laplacian on Step edge, Plane, and Ramp areas for two cases; when the absolute difference and when the signed difference are used in order to prove that the former is the right one and the latter is the wrong.
10. Doing performance analysis for the Gradient-based Laplacian on TUM data set, and comparison study with the state of the art edge detectors on NYUD data set.

The **contributions** of the third research are:

11. Proposing a novel algorithm for the Unsharp Masking of a field of vectors that is based on the Solid Vector subtraction operation and that is analogous to the Unsharp Masking of a field of scalars.
12. Doing behavioral analysis for the Unsharp Masking on the Step edges; Plane areas; and onset, end and along Ramp areas.
13. Doing performance analysis for the Unsharp Masking on TUM data set, and comparison study with the state of the art edge detectors on NYUD data set.

During the work of this dissertation, four research papers have been published in international conferences and IEEE Explore. Those research papers are listed below:

1. I. Naser, J. Al-Anssari and A. Ralescu, "Three-Dimensional Gradient-Based Laplacian Spatial Filter of a Field of Vectors for Geometrical Edges Magnitude Detection in Point Cloud Surfaces," 2019 Joint 8th International Conference on Informatics, Electronics & Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision & Pattern Recognition (icIVPR), Spokane, WA, USA, 2019, pp. 354-361 [1].
2. I. Naser, J. Al-Anssari and A. Ralescu, "Three-Dimensional Unsharp Masking Spatial Filter of a Field of Vectors for Geometrical Edges Magnitude and Direction Detection in Point Cloud Surfaces," 2019 IEEE International Conference on Humanized Computing and Communication (HCC), Laguna Hills, CA, USA, 2019, pp. 68-76 [2].
3. J. Al-Anssari, I. Naser and A. Ralescu, "Three-Dimensional Gradient Spatial Filter of a Field of Vectors for Geometrical Edges Magnitude Detection in Point Cloud Surfaces," 2019 Joint 8th International Conference on Informatics, Electronics & Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision & Pattern Recognition (icIVPR), Spokane, WA, USA, 2019, pp. 362-370 [3].

4. J. Al-Anssari, I. Naser and A. Ralescu, "Three-Dimensional Laplacian Spatial Filter of a Field of Vectors for Geometrical Edges Magnitude and Direction Detection in Point Cloud Surfaces," 2019 IEEE International Conference on Humanized Computing and Communication (HCC), Laguna Hills, CA, USA, 2019, pp. 83-93 [4].

## 1.6 Dissertation Outline

From this point on, this dissertation is organized as follows:

Chapter 2 background theories; in which: section 2.1, 3d camera; section 2.2, the point cloud; section 2.3, the point cloud normal vectors; section 2.4, the neighborhood search method; section 2.5, edge detection; section 2.6, Unsharp Masking; section 2.7, Gradient-based second order derivative; section 2.8, the convolution; section 2.9, the edges types.

Chapter 3, the solid vector operations; in which: section 3.1, introduction; section 3.2, the literature review; section 3.3, the rotated polar coordinate system; section 3.4, the rotated polar mathematical operations; section 3.5, the conclusions.

Chapter 4, the Gradient-Based Laplacian of a field of vectors; in which: section 4.1, introduction; section 4.2, the novel Laplacian algorithm; section 4.3, the design of the Laplacian algorithm; section 4.4, the behavioral analyses; section 4.5, performance analyses, comparison Study, and experimental Results; section 4.6, conclusions and directions of future work;

Chapter 5.4, the Unsharp Masking of a field of vectors; in which: section 5.1, introduction; section 5.2 literature review; section 5.3 Smoothing spatial filter of a field of vectors; section 5.4 Unsharp Masking of a field of vectors; section 5.5 behavioral analyses; section 5.6 performance analyses, comparison study, and experimental results; section 5.7 conclusions and future work;

Chapter 6, the conclusions and future work; in which: section 6.1, conclusions; section 6.2, future work;

And bibliography.

## **Chapter 2**

### **Background Theories**

## Contents

---

2.1	3D Camera . . . . .	14
2.2	Point Cloud . . . . .	15
2.3	Point Cloud Normal Vectors . . . . .	16
2.4	Neighborhood Search Method . . . . .	17
2.5	Edge Detection . . . . .	19
2.6	Unsharp Masking . . . . .	20
2.7	Gradient-Based Second Order Derivative . . . . .	21
2.8	Convolution . . . . .	23
2.9	Edges Types . . . . .	24

---

## 2.1 3D Camera

Three dimensional cameras (e.g. Kinect cameras) are the cameras that measure the depth between each pixel of the sensor plane and the corresponding point of an object in the scene.

Because 3D information is important for many computer vision tasks, several methods for capturing 3D images (information) have been developed, for example stereo-matching method that is presented in [11], and other method that is presented in [12] that measures the range through the use of active sensors which allow achieving a high level of precision.

Recent developed range cameras, (e.g. time-of-flight (ToF) cameras [13] and structured-light (SL) cameras (the Microsoft Kinect 1 [14]), are capable to produce range images, at high spatial resolution and real-time frame rate, which contain, at every pixel, the range of the corresponding element in the scene.

The time-of-flight (ToF) camera is the second generation of Kinect. It measures the time that the light spends in flight to estimate distance (depth). While the structured light camera is the first version of Kinect, which was introduced in origin as a motion sensor for Microsoft Xbox 360 video game console [15]. A structured light camera projects a light pattern on the object and analyzes the distortion of the pattern to get the depth [5], [16]. The time line of Kinect projects can be seen in figure 2.1, which is taken from [5].



The Kinect for Windows camera has two separate sensors; color sensor which returns color image data, and depth sensor which returns depth data. The authors in [5] show comparison of the hardware characteristics of the two versions, Kinect v1 and Kinect v2, as figure 2.2 shows below.

In the current research images from TUM and NYUD datasets are used, both of these dataset images are captured using Kinect for windows v1. These images include color images and depth (range) images. The color images and depth images are used together to create the point cloud data; which is actually the input to the current research algorithms.



Figure 2.1: This figure shows the time line of the Kinect projects [5].

*Comparing the Different Kinect Generations*

	1 <sup>st</sup> Generation Kinect	2 <sup>nd</sup> Generation Kinect
Color resolution/rate	1280x960 @ 12 Hz <i>or</i> 640x480 @ 30 Hz	1920x1080 @ 30 Hz
Infrared resolution/rate	640x480 @ 30 Hz	512x424 @ 30 Hz
Depth resolution/rate	320x240 @ 30 Hz	512x424 @ 30 Hz
Depth range*	0.4 m – 3.0 m <i>or</i> 0.8 m – 4.0 m	0.5 m – 4.5 m
Depth sensing technology	Structured light	Time-of-flight
Field of view (horizontal)	58°	71°
Mic array	4 elements	4 elements
Tilt motor	±27°	none

Figure 2.2: This figure shows the hardware specifications of the first- and second-generation Kinect models [5].

## 2.2 Point Cloud

A point cloud is a set of data points defined by a given coordinate system. These points represent the surface of an object in real world environment. The point cloud is captured by 3D scanners;

one example of these scanners is Microsoft Kinect depth camera.

Point clouds from depth cameras, such as Microsoft Kinect, are classified into organized point clouds, which identify an organized image (or matrix) like structure, where the data is represented by rows and columns. In contrast, unorganized point clouds have no structure and the data is not being divided into rows and columns [5], [17].

Organized point clouds have more advantages, where the data facilitates many types of vision algorithms such as registration [5]. In addition, the operations of nearest neighbor are more efficient when the relationship between the adjacent pixels is known [17].

In the current research, the 3d point cloud data type is an input argument to the proposed algorithms. This point cloud data type is converted from both the RGB color image and its depth image data types by using a Point Cloud library built-in function. This 3D point cloud data type has a wide field range of computer vision and image processing applications. Also, this point cloud 3D data type is perfect to achieve the objective of this work to detect the geometrical edges of indoor environment.

The point cloud has been used for many purposes; for example, reconstructing the surface of objects [18] and [19], representing and estimating the motion of 3D objects [20], creating 3D meshes for medical imaging [21], and augmented reality system [22].

## 2.3 Point Cloud Normal Vectors

The development of affordable depth sensors, such as Microsoft Kinect cameras, has been a perfect source of getting point clouds as inputs to develop many algorithms in computer vision applications, like geometrical edge detection. These cameras have the ability to capture high resolution color and depth images at the same time, and these two images can be used to produce a point cloud as data, as what has been done in this research.

One of the most distinctive information that can be gained from point clouds is normal vectors, and it is suitable to be used in many applications, such as object detection approaches [23], sur-



face reconstruction algorithm [24], point cloud registration [25], and extracting datum feature and analyzing deformation [26].

The point cloud that is acquired from depth (range) sensors presents a noisy sampling of the real world surfaces, and the specific information of these surfaces, like orientation and curvature, is lost in the sampling procedure. To retrieve this information, a set of vectors, which are orthogonal to the tangential plane of every surface point they contact, is constructed to estimate the normal vectors [27].

The normal estimation methods have been divided into two different groups; averaging methods and optimization-based methods [23] and [27]. And there have been used only two types of graph for normal vector estimation: k nearest neighbor (kNN) and Delaunay tessellation (DT) [27].

Different algorithms for normal vectors estimation have been proposed by different researchers. The authors in [23] present two algorithms to perform efficient estimation using integral images by employing an adaptive window size for analyzing local surfaces. While a novel method for estimating the normal vector based on bi-linear interpolation was presented in [28]. Also, a new normal vector estimation method for point clouds is presented in [24] based on the matching results of the local Delaunay triangle mesh formed at each point. And, a method has been developed by [29] of normal vector estimation for point cloud data depending on fitted directional tangent vectors at the data point.

In this research, the method that is used to estimate the normal vectors of the point clouds is the Integral Images, and exactly the *Average3dGradient* method. For specified information about Integral Images, one can see the details in [23] and [30].

## 2.4 Neighborhood Search Method

In the current research, the algorithms that are proposed deal with every point and its neighbors, and to find those neighbors; a neighborhood search method is needed to be used and the used method here is k-d tree method.

A k-d tree (k-dimensional tree), is a data structure used to organize points in a k-dimensional space; in this research, the point clouds are 3D, so the K-D tree is 3D. It is a multidimensional binary search tree developed by [31] as a data structure for information storage to be recovered by searches associated with that information.

In K-D tree, a non-leaf node divides the space into two parts, and that parts are called half-spaces. The left subtree of that node represents the points to the left of that space, and the right subtree represents points to the right of the space.

A neighborhood search method is an imported process in many applications of computer science and there are many neighborhood search methods available; the standard and common one is the KNN (k-nearest neighbor), which finds the k closest points in the data to an inquiry point or set of inquiry points. While radius search method finds all points in the data within a specific distance from an inquiry point or set of inquiry points.

Also, many scholars have used the variable neighborhood search method (VNS), which is proposed in [32] and is a metaheuristic method used to solve a set of combinatorial and global optimization problems and manages a Local Search technique. For example, Khelifa M. and Boughaci D. in [33] proposed a method for the traveling tournaments problem in sport scheduling (TTP) based on VNS; while orienteering problem with hotel selection (OPHS) has been presented by Divsalar A. *et al.* in [34] using a skewed variable neighborhood search.

Also, some researches proposed neighborhood search methods for special purposes. Such as the augmented large neighborhood search (ALNS) method that was proposed by Kim B. *et al.* in [35] for the team orienteering problem in which a team of vehicles tries to gather rewards at a given number of stops within a specific time frame. And, a hierarchical neighborhood search method was presented by Svanberg K. and Werme M. in [36] for solving topology optimization problems defined on discretized linearly elastic continuum structures.

## 2.5 Edge Detection

Edge detection is an approach used often for images segmentation basing on the sudden local change in intensity; edges (or edge segments) are the connected edge pixels sets, when the edge pixels are the pixels where the sudden change of intensity of an image function is happened, and an example of edge segment is a line, when the background intensity of both sides of the line is different than its intensity. Edge detectors are approaches for local image processing designed for edge pixels detecting [7].

The information of edge detection is beneficial for many applications in image processing and computer vision, such as 3D reconstruction, motion, recognition, image enhancement and restoration, image registration, image compression, and so on [37], also the edge detection is basic tool in feature detection and feature extraction areas [38]. This information is also applicable to point cloud segmentation, data compression, and 3D modeling.

The basic edge detection, by detecting the change of the intensity, is done by using first-order or second-order derivatives. The first-order derivative is the Gradient, which is a tool to find edge strength and direction at location  $(x, y)$  of an image, and is defined by a vector. This vector points in the direction of the greatest rate of change of the image function at location  $(x, y)$ , so it has important geometrical property. The magnitude (length) of that vector represented by the value of the change rate in the direction of the Gradient vector, while its direction represented by the angle that measured with respect to the  $x$ -axis. The second-order derivative is the Laplacian, which is a method represented by defining a discrete formula of the second-order derivative and building a filter mask based on that formula [7].

In this research, the edge detection is done in point cloud surfaces and the algorithms that are proposed deal with the normal vectors (field of vectors), in contrast to the previous works mentioned above. The first algorithm is the Gradient-based Laplacian of a field of vectors for detecting 3-D surface geometrical edges magnitude that is based on the Gradient of a field of vectors values that was proposed by [3] that is based of the Solid Vector subtraction operation that was proposed by [3]. While the second algorithm uses the novel Solid Vector subtraction operation

that was proposed by [3] to propose the novel Unsharp Masking of a field of vectors for detecting geometrical edge magnitude and direction of the 3-d point clouds.

Edge detection in point cloud has been studied extensively and many methods were developed. For example, one algorithm presented by [39] in which the two order polynomial surface on the scattered point cloud in the local coordinate system was approximated, and the estimated differential curvature value of the point clouds was utilized for picking the possible edge points with local maximum curvature. Another quadric surface approximation method was used by [40] for estimating curvature and normal of local region and detecting the edge points based on curvatures.

## 2.6 Unsharp Masking

Unsharp Masking is a method used for image sharpening in computer vision. It has been used in many researches as a classical tool to enhance the edge sharpness, and has been used in printing and publishing areas.

This technique is presented briefly in [7], and includes three steps: blurring the original image, subtracting the blurred image from the original one (getting a mask), and adding that mask to the original image [7] and [41]. These steps can be represented also by adding a scaled version of an image (high pass filtered) to the image itself as shown in figure 2.3 from [6].

In the current research, 3D Unsharp masking algorithm of a field of vectors was proposed to detect edges magnitude and direction in point cloud, imitating the same three steps in the classic Unsharp masking: 1) Rounding the vectors of the original point cloud using a smoothing spatial filter. 2) Subtracting the original point cloud vectors from the smoothed version using the solid vector subtracting operation [3] resulting the directional difference which consists of the edges magnitude and direction. 3) Thresholding the edges magnitude according to a predefined certain scalar threshold value. 4) Assigning the edges magnitude scalar values as colors to the same pixels positions of a newly created point cloud. 5) Assigning the edges direction vectors quantities as vectors to the same pixels positions of a newly created point cloud.

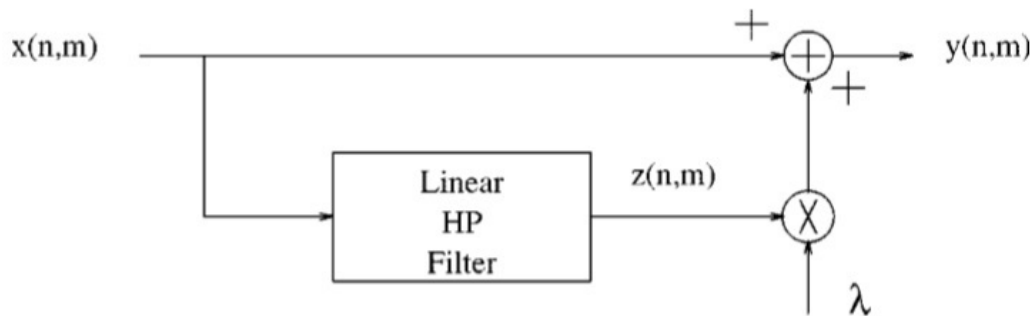


Figure 2.3: Unsharp masking for edge detection [6].

There are varieties of algorithms that have been developed by scholars for Unsharp Masking. For example, a generalized Unsharp Masking algorithm was proposed by [42] which addressed three issues; simultaneous enhancing contrast and sharpness, reducing the halo effect, and solving the out-of-range problem. Also, the overshoot artifacts that occurred around side-planar edges were detected and measured by [43], when they were selected as features in sharpening identification. In addition, an adaptive Unsharp Masking method is proposed by [44] exploiting the estimated local blurriness as the information for pixel-wise enhancement, and this method was called blurriness guided UM, or BUM. And the authors in [6] introduced an adaptive Unsharp Masking for image enhancement employing two directional filters; where the coefficients of that filters are updated using a GaussNewton adaptation strategy, to enhance the contrast in high detail areas, while in the smooth areas, there is a little or no image sharpening occurs.

## 2.7 Gradient-Based Second Order Derivative

Second-order derivative (the Laplacian) is an approach used for edge detection and image segmentation. The classic approach, 2D second-order derivative represented by defining a discrete formulation of the second-order derivative and then creating a filter mask based on that formula-

tion. The Laplacian is isotropic filter, which is rotation invariant [45], and it is a linear operator. The Laplacian for a function (image)  $f(x,y)$  of two variables can be defined by equations 2.1, 2.2, 2.3 and 2.4 [7].

$$\nabla^2(f) = \frac{\delta^2 f}{\delta x^2} + \frac{\delta^2 f}{\delta y^2}, \quad (2.1)$$

For expressing equation 2.1 in discrete form, equations 2.2 and 2.3 represent that in x-direction and y-direction respectively.

$$\frac{\delta^2 f}{\delta x^2} = f(x + 1, y) + f(x - 1, y) - 2f(x, y), \quad (2.2)$$

$$\frac{\delta^2 f}{\delta y^2} = f(x, y + 1) + f(x, y - 1) - 2f(x, y), \quad (2.3)$$

And from the preceding equations, the discrete Laplacian can be shown in equation 2.4 which is implemented using the filter masks in figure 2.4 [7].

$$\nabla^2(f) = [f(x + 1, y) + f(x - 1, y) + f(x, y + 1) + f(x, y - 1)] - 4f(x, y), \quad (2.4)$$

The proposed Laplacian in the current dissertation is the second-order derivative of 3D field of vectors of the geometrical surface of the point cloud. This approach consists of describing a mathematical expression of the second-order derivative and then constructing a surfaced dynamic filter mask dependents to that mathematical expression. The dynamic mask should be isotropic, therefore; the Laplacian is an isotropic derivative operator. Chapter four explains the proposed Laplacian design and equations in details.

0	1	0	1	1	1
1	-4	1	1	-8	1
0	1	0	1	1	1
0	-1	0	-1	-1	-1
-1	4	-1	-1	8	-1
0	-1	0	-1	-1	-1

Figure 2.4: Shows variations of Laplacian filters: (a) top left, filter mask used to implement equation 2.4. (b) top right, mask used to implement the diagonal terms of the Laplacian. (c) and (d) are two other variations of the Laplacian mask [7].

## 2.8 Convolution

Convolution is a fundamental mathematical operation to many popular image processing operators performed by sliding a mask (kernel) over an image. In this operation, the mask is first rotated 180 degree, but when the filter mask is symmetric, the convolution produces the same result without rotating. Convolution produces a multiplication way of two arrays of numbers of different sizes, but of the same dimensionality, to result a third one of the same dimensionality. When the mask

visits a new pixel (from the input signal), a neighborhood search method is used to find the pixels of the signal area under that mask which are positioned around that new signal (image) pixel, then two predened operations, the sum of the products, are computed between the corresponding pixels of the mask and the signal area under that mask [7].

In the current research, the mask for the Unsharp masking algorithm is static, where all of its values are equal to 1, while the mask of the Gradient-based Laplacian algorithm is dynamic, where all of its values are set equal to the Gradient value of the center pixel of the signal area under the mask.

## 2.9 Edges Types

Edges in general consist of three types: horizontal edges, vertical edges, and diagonal edges [46]. An edge in an image is represented by a local change in the image intensity related to a discontinuity in the image intensity. This discontinuity is defined as edge in different types, such as in [8], two types of edges are defined; step discontinuities, in which the image intensity suddenly changes from one value on one side of the discontinuity to a various value on the opposite side, and line discontinuities in which the image intensity suddenly changes value but then goes back to the starting value within some short distance. Where the intensity changes are not immediate but occur over a finite distance, step edges become ramp edges and line edges become roof edges. Figure 2.5 from [8] illustrates these edge profiles. Also three types of edges have been mentioned in [7]; step edge which include a transformation between two intensity levels happens over the distance of one pixel, ramp edge which represented by a set of connected points with no thin (1 pixel thick) path, and roof edge is represented as models of lines through a region when the width of a roof edge is specified by the thickness and sharpness of the line.

The edges types that are depended in this research are the same edges that defined in [3], which are: 1) Plane areas: areas of constant direction of their vectors; 2) Ramp areas: areas of constant rate of change in the direction of their vectors; 3) Step edges: areas of varying rate of change in



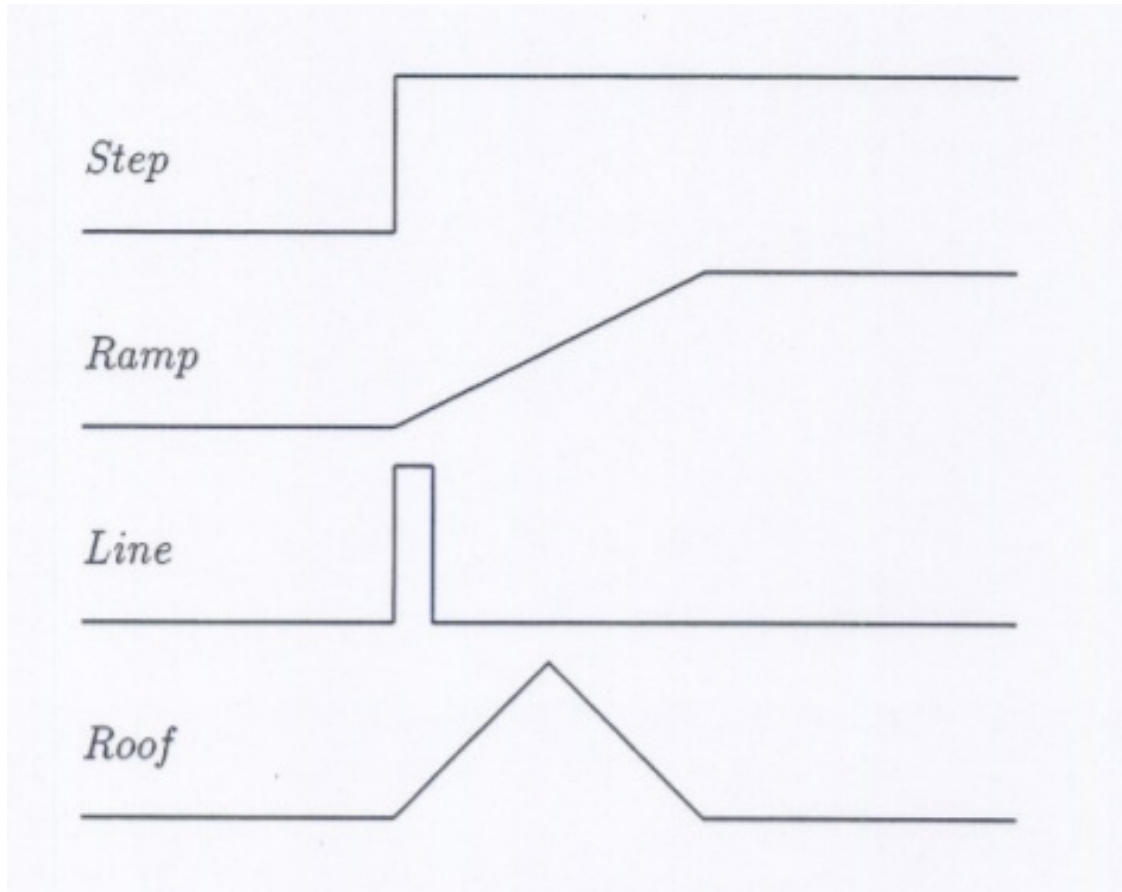


Figure 2.5: Shows one-dimensional edge profiles [8].

the direction of their vectors.

These three edges are classified by the Gradient-based Laplacian and the Unsharp Masking of a field of vectors that are proposed in the current dissertation into two segments, the first one represented by the plane and ramp areas and the second segment represented by the step edges.

## **Chapter 3**

# **The Solid Vector Operations**

## Contents

---

3.1	Introduction . . . . .	28
3.2	The Literature Review . . . . .	32
3.2.1	The Cartesian Coordinate System and Its Component-Wise Cartesian Mathematical Vector Operations . . . . .	32
3.2.2	The Two-Dimensional Polar Coordinate System and Its Two-Dimensional Component-Wise Polar Mathematical Vector Operations . . . . .	35
3.2.3	The Three-Dimensional Spherical Coordinate System and Its Three- Dimensional Spherical Mathematical Vector Operations . . . . .	36
3.2.4	The Mathematical Solid Vector Subtraction Operation . . . . .	38
3.2.5	The Hyper-Plane . . . . .	38
3.3	The Rotated Polar Coordinate System . . . . .	39
3.3.1	The Rotated Polar Coordinate System Center . . . . .	40
3.3.2	The Rotated Polar Coordinate System Axes . . . . .	40
3.3.3	The Rotated Polar Coordinate System Plane/ Hyper-Plane . . . . .	42
3.3.4	The Rotated Polar Coordinate System Unit Vectors . . . . .	42
3.4	The Rotated Polar Mathematical Operations . . . . .	43
3.4.1	Solid Vector Addition Operation . . . . .	44
3.4.2	Solid Vector Multiplication Operation . . . . .	46
3.4.3	Solid Vector Division Operation . . . . .	48
3.4.4	Solid Vector Directional Dot Product Operation . . . . .	49
3.4.5	Solid Vector Directional Cross Product Operation . . . . .	50
3.5	The Conclusions . . . . .	51

---

**Abstract**

Mathematical operations that are applicable to the three and hyper-dimensional space remain a key factor that act as a foundation for a wide range of applications of 3-D image processing and computer vision. The Solid Vector Subtraction operation was recently proposed. The objectives of the current research is to extend this Solid Vector Subtraction operation by proposing definitions of: (1) its coordinate system; (2) its hyper dimensional space; and (3) its other whole complementary set of Solid Vector Addition, Multiplication, Division, Dot Product, and Cross Product operations. This coordinate system, hereafter refereed as the Rotated Polar coordinate system and these mathematical operations, hereafter refereed as Solid Vector mathematical operations handle the vector as a one solid quantity unit, in addition to its property of being applicable to the 2d, 3d and hyper dimensional space. The contributions of the current research are proposing, (1) novel definition of a Rotated Polar coordinate system based on two novel axis: first axis that is the dynamic Operation-Based Zero Vector, and second axis that is the Directional Sign Axis of the Solid Vector Subtraction operation; (2) novel definition of the 2-D, 3-D, hyper dimensional Rotated Polar coordinate system space; (3) novel definition of the Solid Vector Addition; (4) novel definition of the Solid Vector Multiplication; (5) novel definition of the Solid Vector Division; (6) novel definition of the Solid Vector Dot Product; (7) novel definition of the Solid Vector Cross Product.

**3.1 Introduction**

Mathematical operations that are applicable to the three and hyper-dimensional space remain a key factor that act as a foundation for a wide range of applications of 3-D image processing and computer vision—such as: (1) developing 3-D and hyper dimensional artificial intelligence and machine learning descriptors and algorithms, (2) detecting imaging similarities and dissimilarities, (3) controlling of 3-D objects rotations in the 3-D space—such as drones, or CAD virtual reality objects, (4) sculpturing 3-D objects using CAD applications (such as personalizing biomedical equipment like artificial bones using Scada machines), (5) developing indoor or outdoor navigation

using vision-based GPS system, (6) developing vision-based obstacles avoiding applications (such as for UAVs, and elderly falling alerting system).

The Solid Vector Subtraction operation was recently proposed by Al-Anssar J. Naser I. and Ralescu A. [3].

The **objective** of the current research is to extend this Solid Vector Subtraction operation by proposing definitions of: (1) its coordinate system; (2) its hyper dimensional space; and (3) its other whole complementary set of Solid Vector Addition, Multiplication, Division, Dot Product, and Cross Product operations. This coordinate system, hereafter refereed as the Rotated Polar coordinate system and these mathematical operations, hereafter refereed as Solid Vector mathematical operations handle the vector as a one solid quantity unit, in addition to its property of being applicable to the 2d, 3d and hyper dimensional space.

The term Solid Vector is defined as that the vector is handled as one solid quantity by only two components that are: 1) the first is only one magnitude (length) layer ( $r$ ), which keeps the vector as a one solid quantity; and 2) the second is only one directional (angular) component layer ( $\theta$ ), which also keeps the vector as a one solid quantity and undivided into many angular directional components.

Being 2d, 3d and hyper dimensional space, and being Solid Vector make such a coordinate system and its mathematical operations quite useful for the above mentioned applications.

In literature, the typical mathematical coordinate systems and their mathematical operations are the component-wise Cartesian, Spherical and Polar coordinate systems.

The component-wise Cartesian coordinate system and its mathematical operations (that consist of a whole well defined complementary set of Subtraction, Addition, Multiplication, Division, Dot Product and Cross product operations) are applicable to the 2D, 3-D and hyperspace, but their problem is that they are not compliant to the Solid Vector definition, because they handle the vector as a separate layers of  $x, y, z, \dots$  axes which makes building machine learning and data mining algorithms processing and time consuming because in high dimensional data, they have to deal with too many layers at the same time. Therefore, in the current research, the Cartesian is out

of the interest.

The Spherical coordinate system and its mathematical operations are applicable to the 3-D and hyperspace, but their problem is that they are not compliant to the Solid Vector definition, because they handle the vector as: (1) two directional angular layers of  $\theta$  and  $\phi$  Spherical vector components, not only one as the Solid Vector definition stated; and (2) its vector magnitude  $r$  component. Therefore, in the current research, the Spherical is out of the interest.

The Polar coordinate system, hereafter refereed as the Fixed Polar coordinate system, and its mathematical operations are compliant to the above definition of the Solid Vector, because their vector has only two components: (1) one angular component  $\theta$  and (2) one length component  $r$ ; but their problem is that they are not extendable to the 3d or hyper-dimensional space, because: (1) they have a static coordinate system directional Zero vector that is *PoleY*, that is the lowest starting point of its  $\theta$  angular component, and (2) they are fixed to the two-dimensional still plane of their  $x, y$  axes. Therefore, in the current research, the Polar is out of the interest.

On the other side, a novel mathematical operation that is called the Solid Vector Subtraction operation was recently proposed in [3] by Al-Anssari J. *et al.*. This Solid Vector Subtraction operation is applicable to the 2D, 3-D and hyper dimensional space and is compliant to the above Solid Vector definition, because it handles the vector as a one solid quantity unit similarly to the Fixed Polar coordinate system, because it has: only one directional angular component  $\theta$  and only one magnitude  $r$  vector component.

Al-Anssari J. *et al.* in [3] and Naser I. *et al.* in [1] used it as a foundation for their novel 3-D Gradient of a field of vectors geometrical edge detector and their novel 3-D Gradient-Based Laplacian geometrical edge detector high pass spatial filters respectively.

This usefulness of the Solid Vector Subtraction operation made it of a special interest and surfaced the **problem statement** of the current research of how to answer the following questions: 1) what is the definition of its coordinate system (like whether it is Spherical or Polar or something else), 2) what is the definition of its hyper dimensional space and 3) what are the definitions of its other whole complementary set of Solid Vector 1) Addition, 2) Multiplication, 3) Divi-

sion, 4) Dot Product and 5) Cross Product operations.

The **methodologies** to address the problem statement of the current research are: 1) a) making one of the operands vectors of the Solid Vector Subtraction operation as the first axis of the coordinate system that is the Directional Zero vector that replaces the *PoleY* of the Fixed Polar, and that is the lowest starting point of the angular component  $\theta$ ; and b) making the Directional Sign Axis of the Solid Vector Subtraction operation as the second axis of the coordinate system, that replaces the  $x$  axis of the Fixed Polar; in order to get a rotated version of the Fixed Polar that here after, in the current research, refereed to it as the Rotated Polar coordinate system that is implemented in the 3-D dimensional space and that is not fixed to the still plane of the  $x, y$  axes., 2) specifying a pair of vectors as components of a Hyper Rotated Polar when the Solid Vector Subtraction operations between them and their Directional Zero vector produce the same Directional Sign Axis (including both of its opposite directions the positive and negative), in order to define its extension to the hyper dimensional space. and 3) using the triangulation in order to propose the Solid Vector 1) Addition, 2) Multiplication, 3) Division, 4) Dot Product and 5) Cross Product operations.

The **contributions** of the current research are proposing:

- 1) novel definition of a Rotated Polar coordinate system based on two novel axis: 1) first axis that is the dynamic Operation-Based Zero Vector. and 2) second axis that is the Directional Sign Axis of the Solid Vector Subtraction operation.
- 2) Novel definition of the 2-D, 3-D, hyper dimensional Rotated Polar coordinate system space.
- 3) Novel definition of the Solid Vector Addition.
- 4) Novel definition of the Solid Vector Multiplication.
- 5) Novel definition of the Solid Vector Division.
- 6) Novel definition of the Solid Vector Dot Product.
- 7) Novel definition of the Solid Vector Cross Product.

From this point on, this chapter is organized as follows: section 3.2, the literature review; section 3.3, the rotated polar coordinate system; section 3.4, the rotated polar mathematical operations; section 3.5, the conclusions.

## 3.2 The Literature Review

The well known in literature three coordinate systems and their mathematical operations include: (1) Cartesian coordinate system and its mathematical vector operations; (2) two-dimensional Polar coordinate system and its mathematical vector operations; (3) Spherical coordinate system and its mathematical vector operations; (4) Cartesian Magnitude Dot Product operation; (5) Cartesian Magnitude Cross Product operation; and (6) Solid Vector Subtraction operation. Some other types of coordinate systems can be found in [47]. The Hyper-Plane concept in literature is also presented at the end of this section.

### 3.2.1 The Cartesian Coordinate System and Its Component-Wise Cartesian Mathematical Vector Operations

#### The Cartesian Coordinate System

The Cartesian coordinate system is a three-dimensional system. Its unit vectors point to the direction of the increasing coordinates. A point  $P$  can be represented as a three numbers  $P(x, y, z)$ , where  $x, y, z$  are the coordinates of  $P$ . These coordinates range from  $-\infty$  to  $+\infty$ . The Cartesian vector can be represented by three components  $V(V_1, V_2, V_3)$  and can be decomposed into a sum of its components [48]. Some other papers about the Cartesian coordinate system can also be found in [49–53].



**The Component-Wise Cartesian Mathematical Vector Operations**

The features of the Component-Wise Cartesian Mathematical Vector Operations are specified as:

1. they can be of any number of dimensions,
2. they are implemented in the Cartesian coordinate system,
3. they are component-wise operations because they handle the operands and resulting vectors as a separated layers of Cartesian axes (e.g.  $x, y, z \dots$ ),
4. their Cartesian component-wise subtraction vector operation produces the resulting vector that connects the end points of the two vectors operands,
5. their Cartesian component-wise addition vector operation produces the resulting vector that connects the tail point of the first vector operand to the end point of the second vector operand as equation 3.1 shows,
6. their Cartesian component-wise multiplication of vector times scalar operation produces the resulting vector that is the linear extension to the length of the vector operand as equation 3.2 shows, and
7. their Cartesian component-wise division of vector over scalar operation produces the resulting vector that is the linear shrink to the length of the vector operand.

As Larson R. *et al.* presented in their book [54], let  $u = (u_1, u_2, u_3, \dots, u_n)$  and  $v = (v_1, v_2, v_3, \dots, v_n)$  be vectors in  $R^n$  and let  $c$  be a real number. Then the sum of  $u$  and  $v$  is defined as the vector

$$u + v = (u_1 + v_1, u_2 + v_2, u_3 + v_3, \dots, u_n + v_n), \quad (3.1)$$

So on for the Subtraction.

While the scalar multiple of  $u$  by  $c$  is defined as the vector

$$cu = (cu_1, cu_2, cu_3, \dots, cu_n), \quad (3.2)$$

So on for the scalar division.

For more information, reference to the Cartesian coordinate system and its operations can also be found in [55].

### The Cartesian Magnitude Dot Product

The Dot product (scalar product, or inner product), here after refereed as Magnitude Dot Product, operation is a linearity measure of the two vectors.

The Magnitude Dot Product between  $V_1$  and  $V_2$  is defined by equation 3.3.

$$V_1 \cdot V_2 = \langle V_1, V_2 \rangle = \|V_1\| \|V_2\| \cos(\theta) \quad (3.3)$$

Its result is a scalar quantity. When it is applied on two perpendicular vectors, their result is zero. The order of its vector operands in its multiplication does not matter [48].

### The Cartesian Magnitude Cross Product

The Cross product (or Vector product), here after refereed as the Magnitude Cross Product, produces the resulting vector that is perpendicular on the vector operands and its length is the positive area of the parallelogram having the two vector operands as sides.

The Magnitude Cross product operation between  $V_1$  and  $V_2$  results in a vector  $VC$  which has a magnitude and direction, and is defined by equation 3.4.

$$VC = (V_1 \times V_2)_{\text{Magnitude Cross product}} \quad (3.4)$$

The magnitude of the produced vector  $VC$  is defined by equation 3.5.

$$VC_{\text{Magnitude Cross Product}} = \|V_1\| \|V_2\| \sin(\theta) \quad (3.5)$$

The direction of the produced vector  $VC$  is computed using the right hand rule. When the Magnitude Cross Product is applied on two parallel vectors, the result is zero vector, and the order of the vector operands in the multiplication does matter [48].

### 3.2.2 The Two-Dimensional Polar Coordinate System and Its Two-Dimensional Component-Wise Polar Mathematical Vector Operations

#### The Two-Dimensional Fixed Polar Coordinate System

It is a two-dimensional system. Its unit vectors denoted as  $a_\theta$  and  $a_r$  are not drawn at the origin but at a convenient point in 2d plane. They point to the direction of the increasing coordinates variables and are orthogonal to each other. A point  $P$  can be represented as a two numbers  $P(\theta, r)$ . Where  $\theta, r$  are the coordinates of  $P$ . The  $\theta$  coordinate range from 0 to  $2\pi$ . The  $r$  coordinate ranges from zero to  $+\infty$ . The Polar vector can be represented by two components  $A(A_\theta, A_r)$ , where  $A_\theta$  and  $A_r$  are called the components of  $A$ . A Polar vector can be decomposed into a sum of its components  $A = A_\theta + A_r$  [56]. Another research that tackled applications for the Polar coordinate system can be found in [57].

#### The Two-Dimensional Fixed Polar Mathematical Vector Operations

The features of the two-dimensional Fixed Polar mathematical vector operations are specified as:

1. They are implemented in the Polar coordinate system
2. They handle the vector as only two separated layers: (1) the first consists of only one angular layer (e.g.  $\theta$ ) that starts from Pole Y; and (2) the second consists of only one magnitude layer (e.g.  $r$ )
3. Their Polar Addition and Subtraction operations are defined as that: let  $u = (u_\theta, u_r)$  and  $v = (v_\theta, v_r)$  be vectors in  $R^2$  and let  $c$  be a real number. Then the sum of  $u$  and  $v$  is defined as the vector

$$u + v = (u_\theta + v_\theta, u_r + v_r) \quad (3.6)$$

So on for the Subtraction.

While the Polar scalar multiplication and division of  $u$  by  $(c_\theta, c_r)$  is defined as the vector

$$(c_\theta, c_r)u = (c_\theta u_\theta, c_r u_r) \quad (3.7)$$

So on for the scalar division.

4. Their Dot product operation is defined as that: if  $A$  and  $B$  are Polar vectors represented in terms of components  $A(A_\theta, A_r)$  and  $B(B_\theta, B_r)$ , their Scalar product is:

$$A.B = A_\theta B_\theta + A_r B_r \quad (3.8)$$

5. The Polar vector magnitude is

$$|A| = \sqrt{A_\theta^2 + A_r^2} \quad (3.9)$$

### 3.2.3 The Three-Dimensional Spherical Coordinate System and Its Three-Dimensional Spherical Mathematical Vector Operations

#### The Three-Dimensional Spherical Coordinate System

It is a three-dimensional system. Its unit vectors denoted as  $a_\theta$ ,  $a_\phi$  and  $a_r$  are not drawn at the origin but at a convenient point in 3d space. They point to the direction of the increasing coordinates variables and are orthogonal to each other. A point  $P$  can be represented as a three numbers  $P(\theta, \phi, r)$ , where  $\theta, \phi, r$  are the coordinates of  $P$ . The  $\theta$  coordinate range from 0 to  $\pi$ . The  $\phi$  coordinate range from 0 to  $2\pi$ . The  $r$  coordinate ranges from *zero* to  $+\infty$ . The Spherical vector can be represented by three components  $A(A_\theta, A_\phi, A_r)$ , where  $A_\theta$ ,  $A_\phi$  and  $A_r$  are called the component of  $A$ . A Spherical vector can be decomposed into a sum of its components  $A = A_\theta + A_\phi + A_r$  [48]. In their research [58], Chao H. *et al.* established a linear model with spherical coordinates for relative motion in an elliptical reference orbit. Wang J. *et al.* in [59] proposed an effective method

to detect the recompression in the color images by using the conversion error, rounding error, and truncation error on the pixel in the spherical coordinate system.

### The Three-Dimensional Spherical Mathematical Vector Operations

The features of the 3d Spherical mathematical vector operations are:

1. They are implemented in the Spherical coordinate system.
2. They handle the vector as three separated layers: (1) the first consists of one angular layer (e.g.  $\theta$ ) that starts from Pole Y; (2) the second consists of one angular layer (e.g.  $\phi$ ) that starts from Pole X; and (3) the third consists of one magnitude layer (e.g.  $r$ ).
3. Their Spherical Addition and Subtraction operations are defined such that: let  $u = (u_\theta, u_\phi, u_r)$  and  $v = (v_\theta, v_\phi, v_r)$  be vectors in  $R^3$  and let  $(c_\theta, c_\phi, c_r)$  be a real multipliers numbers. Then the sum of  $u$  and  $v$  is defined as the vector

$$u + v = (u_\theta + v_\theta, u_\phi + v_\phi, u_r + v_r) \quad (3.10)$$

So on for the Subtraction.

While the Spherical scalar multiplication and division of  $u$  by  $(c_\theta, c_\phi, c_r)$  is defined as the vector

$$(c_\theta, c_r)u = (c_\theta u_\theta, c_\phi u_\phi, c_r u_r) \quad (3.11)$$

So on for the scalar division.

4. Their Dot product operation is defined in [48] as that if  $A$  and  $B$  are Spherical vectors represented in terms of components  $A(A_\theta, A_\phi, A_r)$  and  $B(B_\theta, B_\phi, B_r)$ , their Scalar product is:

$$A \cdot B = A_\theta B_\theta + A_\phi B_\phi + A_r B_r \quad (3.12)$$

5. Their Cross product operation is defined in [48] as that if  $A$  and  $B$  are Spherical vectors represented in terms of components  $A(A_\theta, A_\phi, A_r)$  and  $B(B_\theta, A_\phi, B_r)$ , their Cross product can be obtained by evaluating the following determinant:

$$A \times B = \begin{vmatrix} a_\theta & a_\phi & a_r \\ A_\theta & A_\phi & A_r \\ B_\theta & B_\phi & B_r \end{vmatrix} \quad (3.13)$$

6. The Spherical vector magnitude is

$$|A| = \sqrt{A_\theta^2 + A_\phi^2 + A_r^2} \quad (3.14)$$

### 3.2.4 The Mathematical Solid Vector Subtraction Operation

In the Solid Vector Subtraction operation, that was proposed by Al-Anssari J., Naser I. and Ralescu A in [3], and is illustrated in figure 3.1, the given inputs are the two vector operands  $V(P_u)$  and  $V(P_i)$ , where the first vector operand  $V(P_u)$  is made as the Directional Pole Zero Axis, and the produced outputs are: (1) the directional Solid Vector difference vector  $VT$  that consists of: (a) the angular displacement  $\theta$  that is in the direction of (b) the Directional Sign Axis  $VS$ , where the Directional Sign Axis is the vector that is produced from the Cartesian vector subtraction of the second vector operand  $V(P_i)$  from the projection of  $V(P_i)$  on the Directional Zero Axis  $V(P_u)$ ; and (2) the length difference  $M$ .

### 3.2.5 The Hyper-Plane

A hyper-plane, in geometry, is a subspace that its dimension is one less than that of its ambient space. For a 3d space, its hyper-planes are the 2d planes. For a 2d space, its hyper-planes are the

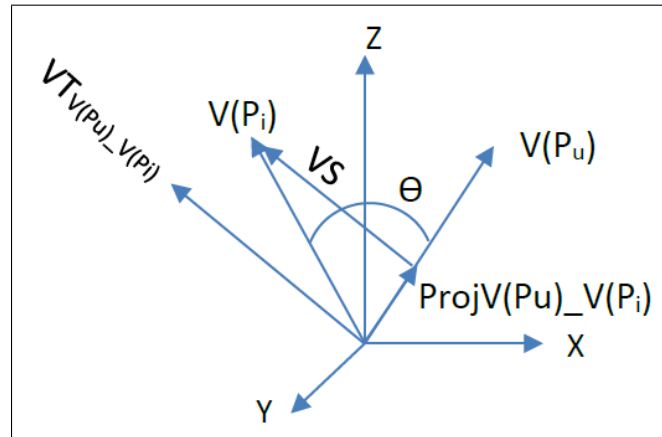


Figure 3.1: Cited from [3], this figure shows the Solid Vector Subtraction operation.

1d lines. Hyper-planes are useful for applications related to machine learning especially to create support vector machines for such tasks as computer vision and natural language processing [60,61].

### 3.3 The Rotated Polar Coordinate System

In this section, the Rotated Polar coordinate system is proposed which is a rotated version of the Fixed Polar coordinate system from fixed two-dimensional  $x, y$  plane to a freely rotating plane/hyper-plane in the Cartesian space around the origin point Null vector, where all of its vectors are confined in this same Plane/ Hyper-Plane.

The Rotated Polar coordinate system that is shown in figure 3.2, is very convenient whenever problems that having Spherical symmetry are being dealt with (e.g. edge detection in point clouds surfaces that is based on the Solid Vector Subtraction operation). More specifically, it is used to implement this Solid Vector Subtraction and its complementary set of the Solid Vector mathematical operations presented in section 3.4; while its corresponding Cartesian coordinate system is used to localize its vectors in the space. In the following subsections, its center, polar axes, its plane/ hyper-plane and its unit vectors are defined.

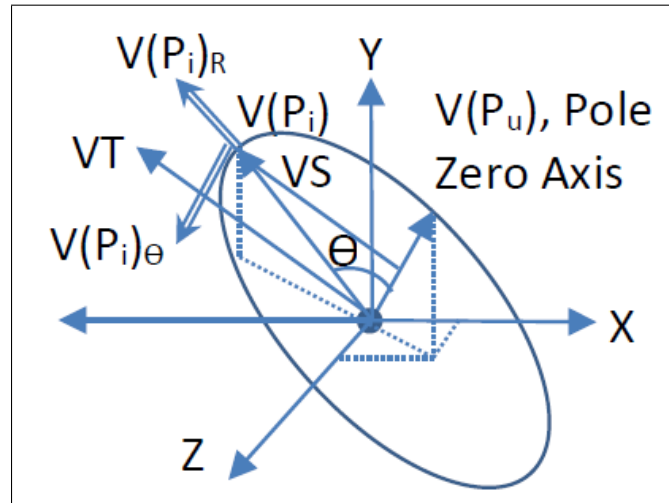


Figure 3.2: This figure shows the Rotated Polar coordinate system.

### 3.3.1 The Rotated Polar Coordinate System Center

The center of the Rotated Polar coordinate system falls on the *NullVector* origin point of its Cartesian coordinate system.

### 3.3.2 The Rotated Polar Coordinate System Axes

The Rotated Polar coordinate system has only two axes that are: (1) The Directional Pole Zero Axis, and (2) the Directional Sign Axis, as described below.

#### The Directional Pole Zero Axis

It is the first axis of the Rotated Polar coordinate system denoted by  $V_{zero}$  and has the following properties:

1. It is operation-based that is specific for its Solid Vector mathematical operation, dynamically change with it, and can be different from the others.
2. It does not have a fixed orientation that fall on the Cartesian axes, but it freely rotates around the origin point Null Vector and can be pointing anywhere in the 3-D/ hyper-dimensional space.



3. Has a direction that is set up to describe the “no quantity”, “lowest starting”, “empty” direction with respect to the other vectors operands, that are involved in the Solid Vector mathematical operation, and can be one of them (e.g. in the Solid Vector Subtraction, the subtrahend is the Directional Pole Zero Axis; and as will be proposed later in the current research, the addend is the Directional Pole Zero Axes of the Solid Vector Addition).

The above properties of the first axis of the Rotated Polar enable the Solid Vector mathematical operations to comply with the Solid Vector term definition (3.4.1).

They are on the contrary to the following properties of the first axis of Spherical and Fixed Polar coordinate systems (Pole Y) that make the Spherical and Fixed Polar not compliant to the Solid Vector definition:

1. It is coordinate system-based that is static (universal), the same for all its Spherical and Fixed Polar mathematical operations.
2. It has a fixed orientation that falls on the Cartesian  $y$  axes, Pole Y.
3. Its Cartesian  $y$  axis direction describes the “no quantity”, “lowest starting”, “empty” direction with respect to the other vectors operands, that are involved in the Spherical and Fixed Polar mathematical operations.

### **The Directional Sign Axis**

Denoted by  $VS$ , it is the second axis of the Rotated Polar coordinate system which is produced by the Directional Solid Vector Subtraction operation between the Directional Pole Zero Axis and the other vector operand that is involved in the mathematical Solid Vector Subtraction operation. It is perpendicular to the Directional Pole Zero Axis and describes the directional sign of the orientation divergence between their Zero Vector and operand vectors.

### 3.3.3 The Rotated Polar Coordinate System Plane/ Hyper-Plane

The Rotated Polar plane/ hyper-plane is defined by the two axes; the Directional Pole Zero Axis and the Directional Sign Axis of its Rotated Polar coordinate system, and according with their rotation, it freely rotates around its corresponding Cartesian coordinate system origin point Null Vector. It also contains all of the other Rotated Polar coordinate system vectors.

The definition 3.3.1 of the theory of vector spaces of the Rotated Polar Coordinate System 2D, 3D Plane and Hyper-Plane of a set of vectors is proposed here, that is:

**Definition 3.3.1.** A set of vectors is said to be planar/ hyper-planar if a plane/ hyper-plane in the space contains the origin point Null Vector, the Directional Pole Zero vector, and all the other vectors in this set with at least one of them non parallel with the Directional Pole Zero Vector; if no plane/ hyper-plane in the space can be written in this way, then the vectors are said to be non-planar/ hyper-planar.

Also, its mathematical definition 3.3.2 is proposed, that is:

**Definition 3.3.2.** A set of vectors is said to be planar/ hyper-planar if the Solid Vector subtraction operations between each of its vectors and their shared same Directional Pole Zero Vector produces the same Directional Sign Axis (including both of its positive and negative signs). If any of the Solid Vector subtraction operations between any one of its vectors and their shared same Directional Pole Zero Vector produces a different Directional Sign Axis (including both of its positive and negative signs), the set of vectors is said to be non-planar/ hyper-planar.

### 3.3.4 The Rotated Polar Coordinate System Unit Vectors

In the Rotated Polar coordinate system, where  $V(P_u)$  is the Directional Pole Zero Axis, and  $V(P_i)$  is a vector in this coordinate system,  $VS$  is the Directional Sign Axis that is produced from the Solid Vector Subtraction operation between  $V(P_u)$  and  $V(P_i)$ . Unit vectors of  $V(P_i)$  in this system, denoted  $V(P_i)_R$ ,  $V(P_i)_\theta$  are usually not drawn at the origin point Null Vector, but at a convenient point in space, that is the end point of  $V(P_i)$ . They are orthogonal to each other.  $V(P_i)_\theta$

points in the direction of the increasing Absolute Solid Vector Directional Difference between  $V(P_i)$  and the Directional Pole Zero axis  $V(P_u)$  on the plane/ hyper-plane that contains  $V(P_u)$  and  $VS$  and on the side of the Directional Sign Axis  $VS$ .  $V(P_i)_R$  points in the direction of the increasing vector length.

$$V(P_i) : (R, \theta@VS) \quad (3.15)$$

Where  $R, \theta@VS$  are called  $V(P_i)$  Rotated Polar vector components;  $\theta$  is the angular displacement, @ means “belongs to”, and  $VS$  is the Directional Sign Axis that  $\theta$  belongs to.

The ranges of these components variables are

$$0 \leq V(P_i)_R < \infty \quad (3.16)$$

$$0 \leq V(P_i)_\theta \leq 1 \quad (3.17)$$

Where the  $\theta$  Zero number starts from Directional Pole Zero vector  $V(P_u)$ , measured by the radian angle unit.

### 3.4 The Rotated Polar Mathematical Operations

In this section, the novel Solid Vector mathematical Addition, Multiplication, Division, Dot Product and Cross Product operations are proposed that are a whole complementary set to the Solid Vector Subtraction operation that Al-Anssari J. proposed in [3]. They are called Solid Vector because they handle the vector as a one solid quantity unit where they handle the only two Rotated Polar vector components  $\theta@VS$  and  $R$ , where the Zero reference for their  $\theta@VS$  directional polar component is the Directional Pole Zero Axis  $V_{Zero}$ , that is dynamic specific for each operation and change with it.

The Solid Vector term is defined as:

**Definition 3.4.1.** The vector that has only two polar components that are: (1) only one directional  $\theta@VS$  component; and (2) only one length  $r$  component.

Note that, the symbol  $VT$  refer to  $\theta@VS$ , as equations 3.18 shows, therefore, sometimes the  $\theta@VS$  symbol is used instead of  $VT$  or vice versa. So on for  $M$  and  $R$ , as equation 3.19 shows.

$$VT = \theta@VS \quad (3.18)$$

$$M = R \quad (3.19)$$

Their corresponding Cartesian vector components are used to localize their vectors in the space. The given inputs and produced output of each of the Solid Vector operation are listed as follows:

### 3.4.1 Solid Vector Addition Operation

In the Solid Vector Addition operation, that is illustrated in figure 3.3, the given inputs are the vector operand  $V(P_u)$  that is, in this operation, considered the Rotated Pole Zero Axis, and the Rotated Polar components  $VT = \theta@VS$  and  $M = R$ ; the produced output is the Cartesian result vector  $V(P_i)$ .

The mathematical Solid Vector addition operation of the vector operand  $V(P_u)$  with  $VT$  and  $M$  consists of two sub-operations: (1) the Directional addition operation; (2) and the Magnitude addition operation, as equation 3.20 shows.

$$V(P_i) = (V(P_u) + \{VT_{V(P_i).V(P_u)}, M_{V(P_i).V(P_u)}\})_{SV} \quad (3.20)$$

1. First, to compute the Directional Solid Vector Addition operation, as the following steps show,  $V(P_u)$  is added to  $VT_{V(P_i).V(P_u)}$ :-

- (a) It is supposed that the length of the prime output vector,  $V'(P_i)$  is equal to the length of  $V(P_u)$  as shown in equation 3.21.

$$\|V'(P_i)\| = \|V(P_u)\|, \quad (3.21)$$

(b) Therefore, the vector  $V'(P_i)$  is computed by the Cartesian vector addition operation of the two vectors: (1)  $Proj_{V(P_u)}V'(P_i)$ , the projection of  $V'(P_i)$  on  $V(P_u)$ ; (2) and  $Proj_{VT_{V(P_i)}V(P_u)}V'(P_i)$ , the projection of  $V'(P_i)$  on  $VT_{V(P_i)}V(P_u)$ , as equation 3.22 shows.

$$\begin{aligned} V'(P_i) \\ = Proj_{V(P_u)}V'(P_i) + Proj_{VT_{V(P_i)}V(P_u)}V'(P_i) \end{aligned} \quad (3.22)$$

Where in equation 3.22, the vector  $Proj_{V(P_u)}V'(P_i)$  is computed by using equation 3.23, and the vector  $Proj_{VT_{V(P_i)}V(P_u)}V'(P_i)$  is computed by using equation 3.24.

$$Proj_{V(P_u)}V'(P_i) = \frac{\|V'(P_i)\| \cos(\theta)}{\|V(P_u)\|} V(P_u), \quad (3.23)$$

$$\begin{aligned} Proj_{VT_{V(P_i)}V(P_u)}V'(P_i) \\ = \frac{\|V'(P_i)\| \sin(\theta)}{\|VT_{V(P_i)}V(P_u)\|} VT_{V(P_i)}V(P_u); \end{aligned} \quad (3.24)$$

Where in equation 3.23 and equation 3.24, the radian angle  $\theta$  between  $V(P_u)$  and  $V'(P_i)$  is equal to the length of the vector  $VT_{V(P_i)}V(P_u)$ , as equation 3.25 shows; and the name of the vector  $VT$  is a shortcut name for the same vector  $VT_{V(P_i)}V(P_u)$ .

$$\theta = \|VT_{V(P_i)}V(P_u)\|, \quad (3.25)$$

2. Second, the magnitude difference,  $M_{V(P_i)}V(P_u)$ , is added to  $V'(P_i)$  to get the result outcome of the *Solid Vector* addition operation,  $V(P_i)$ , as shown by equation 3.26.

$$V(P_i) = \frac{\|V'(P_i)\| + M_{V(P_i)}V(P_u)}{\|V'(P_i)\|} V'(P_i) \quad (3.26)$$

The produced result vector  $V(P_i)$  is represented in the Cartesian coordinate system.

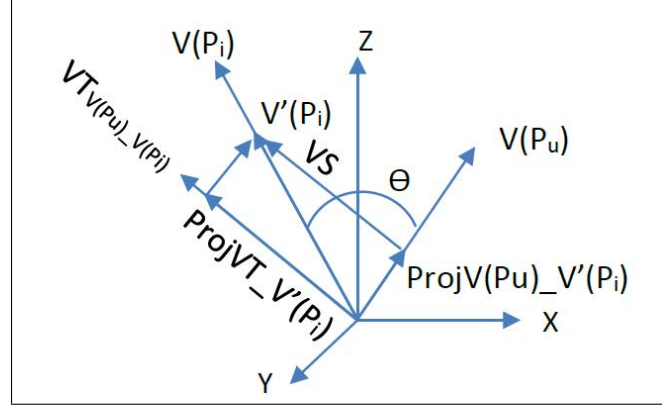


Figure 3.3: shows the Solid Vector addition operation.

### 3.4.2 Solid Vector Multiplication Operation

In the Solid Vector Multiplication operation, the given inputs are: the Directional Pole Zero Axis  $V_{zero}$ , the vector operand  $V(P_2)$ , and the Rotated Polar angular  $\theta$  multiplier and magnitude  $M$  multiplier. The Directional Sign Axis is computed by the Directional Solid Vector Subtraction operation between the vector operand  $V(P_2)$  and the Directional Pole Zero Axis  $V_{zero}$ . And the produced outputs are: the angular component  $\theta@VS$  and the magnitude component  $M$  combined in the produced Cartesian vector  $V(P_1)$  result.

There are two types of the multiplication process between a vector and a scalar: (1) multiplication of the vector  $V(P_2)$  direction by an angular scalar multiplier ( $\theta_{multiplier}$ ); (2) multiplication of the vector  $V(P_2)$  length by a magnitude scalar multiplier ( $M_{multiplier}$ ), as equation 3.27 shows.

$$V(P_1) = \left( V(P_2) * \left\{ \begin{array}{l} \theta_{multiplier} \\ M_{multiplier} \end{array} \right\} \right)_{SV \text{ with respect to } V_{zero}} \quad (3.27)$$

To perform the multiplication process of equation 3.27, first  $V(P_2)$  is multiplied by the angular multiplier ( $\theta_{multiplier}$ ), then the produced vector is multiplied by the magnitude multiplier ( $M_{multiplier}$ ) as the following two sub-processes.

### Multiplication of The Vector Direction by an Angular Scalar Multiplier

In order to multiply the vector direction by the angular scalar,  $\theta_{multiplier}$ , it is needed to get the Directional Pole Zero Axis,  $V_{zero}$ , of this multiplication process, then to perform the following steps sequentially:

1. Getting the input Directional Pole Zero vector  $V_{zero}$ .
2. Solid Vector Subtracting  $V(P_2)$  from  $V_{zero}$  in order to compute Absolute Directional Solid Vector difference ( $\theta_{V(P_2).V_{zero}}$ ) and the Directional Sign Axis ( $VS$ ) according to the Solid Vector Subtraction operation presented in [3].
3. Multiplying the Absolute Directional Solid Vector difference ( $\theta_{V(P_2).V_{zero}}$ ) by the radian angular multiplier ( $\theta_{multiplier}$ ) to produce the prime output vector ( $V'(P_1)$ ) Rotated Polar angular component ( $\theta_{V'(P_1).V_{zero}}$ ).

$$\theta_{V'(P_1).V_{zero}} = \theta_{V(P_2).V_{zero}} * \theta_{multiplier} \quad (3.28)$$

4. The length of the prime output vector ( $V'(P_1)$ ) is supposed to be equal to the length of the input vector ( $V(P_2)$ ) as equation 3.29 shows.

$$\|V'(P_1)\| = \|V(P_2)\| \quad (3.29)$$

5. Computing the projection of the prime output vector ( $V'(P_1)$ ) on both the Directional Pole Zero vector ( $V_{zero}$ ) and the Directional Sign Axis ( $VS$ ).

$$Proj_{V_{zero}} V'(P_1) = \frac{\|V'(P_1)\| \cos(\theta_{V'(P_1).V_{zero}})}{\|V_{zero}\|} V_{zero} \quad (3.30)$$

$$Proj_{VS} V'(P_1) = \frac{\|V'(P_1)\| \sin(\theta_{V'(P_1).V_{zero}})}{\|VS\|} VS \quad (3.31)$$

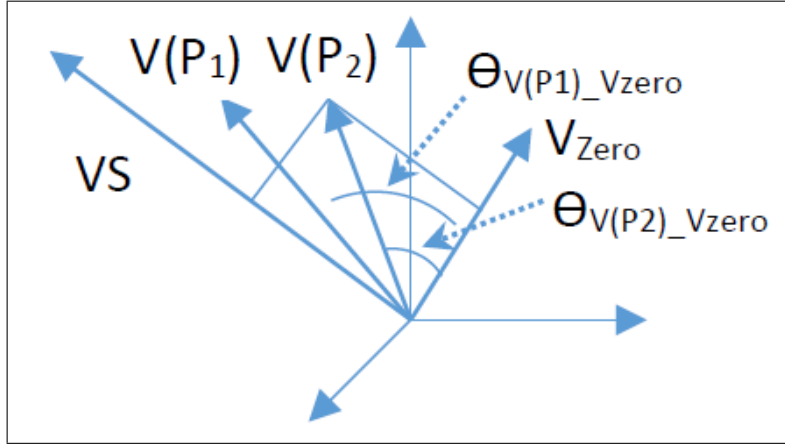


Figure 3.4: Solid Vector Directional Multiplication of a vector by an angular scalar.

6. Computing the prime output vector ( $V'(P_1)$ ) by adding the above produced two vectors ( $Proj_{V_{zero}}V'(P_1)$ ) and ( $Proj_{VS}V'(P_1)$ ) using the Cartesian vectors addition operation as shown in equation (3.32).

$$V'(P_1) = Proj_{V_{zero}}V'(P_1) + Proj_{VS}V'(P_1) \quad (3.32)$$

### Multiplication of The Vector Length by a Magnitude Scalar Multiplier

To multiply the vector length by a magnitude multiplier, the length of the vector is just multiplied by the magnitude multiplier and the direction of the vector is kept the same.

$$V(P_1) = M_{multiplier} * V'(P_1); \quad (3.33)$$

Where in equation 3.33 the multiplication here is a Cartesian scalar by a Vector multiplication.

### 3.4.3 Solid Vector Division Operation

In the Solid Vector Division operation, the given inputs are: the Directional Pole Zero Axis  $V_{zero}$ , the vector operand  $V(P_2)$ , and the Rotated Polar angular  $\theta$  divisor and magnitude  $M$  divisor. The Directional Sign Axis  $VS$  is computed by the Directional Solid Vector Subtraction operation



between the vector operand  $V(P_2)$  and the Directional Pole Zero Axis  $V_{zero}$ . And the produced results are: the angular component  $\theta@VS$  and the length component  $M$  combined in a Cartesian vector  $V(P_1)$  result.

There are two types of the division process between a vector and a scalar: (1) division of the vector direction by the Rotated Polar angular divisor ( $\theta_{divisor}$ ); (2) division of the vector length by a magnitude divisor ( $M_{divisor}$ ), as equation 3.34 shows.

$$V(P_1) = \left( V(P_2) * \left\{ \begin{array}{c} \frac{1}{\theta_{Divisor}} \\ \frac{1}{M_{Divisor}} \end{array} \right\} \right)_{SV \text{ with respect to } V_{zero}} \quad (3.34)$$

To perform the division process of equation 3.34, first  $V(P_2)$  is multiplied by the angular divisor ( $\frac{1}{\theta_{divisor}}$ ), then the produced vector is multiplied by the magnitude divisor ( $\frac{1}{M_{divisor}}$ ) in a similar way to the Solid Vector Multiplication process described above.

### 3.4.4 Solid Vector Directional Dot Product Operation

In the Solid Vector Directional Dot Product operation, the given inputs are the vector operands  $V(P_1)$  and  $V(P_2)$ , and the Directional Pole Zero Axis ( $V_{zero}$ ); the produced result is a scalar quantity; This operation is performed by the following steps:

1. The vector  $V(P_1)$  is Solid Vector subtracted from  $V_{zero}$  in order to get  $VT_{V(P_1)-V_{zero}}$  as shown in equation 3.35.

$$VT_{V(P_1)-V_{zero}} = (V(P_1) - V_{zero})_{SV} \quad (3.35)$$

2. The vector  $V(P_2)$  is Solid Vector subtracted from  $V_{zero}$  in order to get  $VT_{V(P_2)-V_{zero}}$  as shown in equation 3.36.

$$VT_{V(P_2)-V_{zero}} = (V(P_2) - V_{zero})_{SV} \quad (3.36)$$

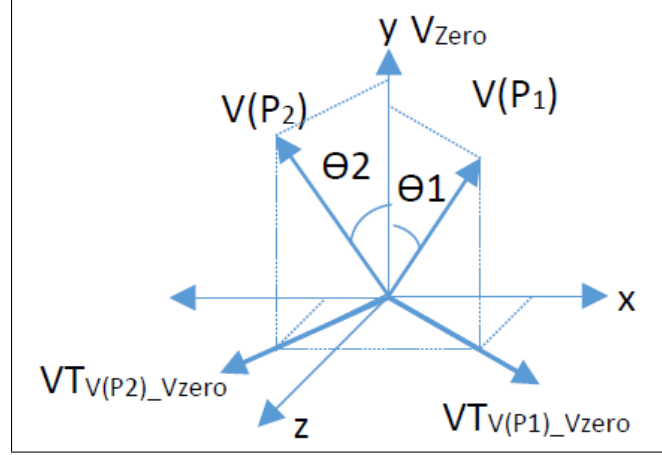


Figure 3.5: Directional dot product.

3. The Solid Vector Directional Dot Product operation between  $V(P_1)$  and  $V(P_2)$  is equal to the Cartesian Magnitude Dot Product operation, that is presented in the section 3.2.1, between the two vectors computed above ( $VT_{V(P_1)-V_{zero}}$ ) and ( $VT_{V(P_2)-V_{zero}}$ ) as shown in equation 3.37.

$$\begin{aligned}
 & (V(P_1) \cdot V(P_2))_{SV \text{ Directional Dot Product}} \\
 &= (VT_{V(P_1)-V_{zero}} \cdot VT_{V(P_2)-V_{zero}})_{Cartesian \text{ Dot Product}} \\
 &= \|VT_{V(P_1)-V_{zero}}\| \|VT_{V(P_2)-V_{zero}}\| \\
 & \quad \cos(\theta_{VT_{V(P_1)-V_{zero}}-VT_{V(P_2)-V_{zero}}})
 \end{aligned} \tag{3.37}$$

### 3.4.5 Solid Vector Directional Cross Product Operation

In the Solid Vector Directional Cross Product operation, the given inputs are the vector operands  $V(P_1)$  and  $V(P_2)$ , and the Directional Pole Zero Axis ( $V_{zero}$ ); the produced output is the Cartesian result vector  $V(P_3)$ ; This operation is performed by the following steps:

1. The vector  $V(P_1)$  is Solid Vector subtracted from  $V_{zero}$  in order to get  $VT_{V(P_1)-V_{zero}}$  as shown in equation 3.38.

$$VT_{V(P_1)-V_{zero}} = (V(P_1) - V_{zero})_{SV} \quad (3.38)$$

2. The vector  $V(P_2)$  is Solid Vector subtracted from  $V_{zero}$  in order to get  $VT_{V(P_2)-V_{zero}}$  as shown in equation 3.39.

$$VT_{V(P_2)-V_{zero}} = (V(P_2) - V_{zero})_{SV} \quad (3.39)$$

3. The Solid Vector Directional Cross Product operation between  $V(P_1)$  and  $V(P_2)$  produces a vector  $V(P_3)$  which is equal to the Cartesian Magnitude Cross Product operation between the two vectors computed above ( $VT_{V(P_1)-V_{zero}}$ ) and ( $VT_{V(P_2)-V_{zero}}$ ) as shown in equation 3.40.

$$\begin{aligned} V(P_3) &= (V(P_1) \times V(P_2))_{SV \text{ Directional Cross Product}} \\ &= (VT_{V(P_1)-V_{zero}} \times \\ &VT_{V(P_2)-V_{zero}})_{Cartesian \text{ Cross Product}} \end{aligned} \quad (3.40)$$

Thus, the produced vector  $V(P_3)$  magnitude is computed using equation 3.41.

$$\begin{aligned} V(P_3)_{Magnitude} &= \|VT_{V(P_1)-V_{zero}}\| \|VT_{V(P_2)-V_{zero}}\| \\ &\sin(\theta_{VT_{V(P_1)-V_{zero}}-VT_{V(P_2)-V_{zero}}}) \end{aligned} \quad (3.41)$$

The direction of the produced vector  $V(P_3)$  is computed using the right hand rule that is implemented on the two vectors ( $VT_{V(P_1)-V_{zero}}$ ) and ( $VT_{V(P_2)-V_{zero}}$ ).

### 3.5 The Conclusions

The recently proposed Solid Vector Subtraction operation has surfaced up the questions of the current research problem statement. Those questions are about the definition of its coordinate

system, hyper-space, and its other whole complementary set of Solid Vector operations. In order to answer these questions, the following were proposed: (1) a novel definition for Rotated Polar coordinate system based on two novel axes: (a) the dynamic Operation-Based Zero Vector, and (b) the Directional Sign Axis of the Solid Vector Subtraction operation; (2) a novel definition for the 2d, 3d, hyper dimensional Rotated Polar coordinate system; (3) a novel definition of the Solid Vector Addition; (4) a novel definition of the Solid Vector Multiplication; (5) a novel definition of the Solid Vector Division; (6) a novel definition of the Solid Vector Dot Product; and (7) a novel definition of the Solid Vector Cross Product.

These Solid Vector mathematical operations are applicable to 2d, 3d and hyper-dimensional space; and they handle the vector as only two layers: one directional angular component, and one magnitude component; which make them Solid Vector compliant and act as a foundation for the future work that will be building 2d, 3d and hyper-dimensional artificial intelligence and machine learning applications—such as 3d object recognition and detection, 3d SLAM, 3d rotation control of virtual and physical objects rotation and 3d virtual sculpturing.

## **Chapter 4**

# **The Gradient-Based Laplacian of a Field of Vectors**

## Contents

---

4.1	Introduction . . . . .	55
4.1.1	Literature Review . . . . .	57
4.2	Gradient-Based Laplacian Spatial Filter of a Field of Vectors . . . . .	58
4.2.1	Definition of the Gradient-Based Laplacian for two-dimensional field of vectors . . . . .	59
4.2.2	Definition of the Gradient-Based Laplacian for three-dimensional field of vectors . . . . .	61
4.3	Design of The Gradient-Based Laplacian Algorithm . . . . .	62
4.3.1	The Classification . . . . .	64
4.3.2	The Pseudo Code and The Complexity . . . . .	64
4.4	Behavioral Analyses . . . . .	64
4.5	Performance Analyses, Comparison Study, and Experimental Results . . . . .	68
4.5.1	Performance Analyses on The TUM data set . . . . .	70
4.5.2	Comparative Study on the NYUD data set . . . . .	71
4.6	Conclusions and Future Work . . . . .	74

---

### Abstract

Also known as the second derivative, the Gradient-Based Laplacian, that is used for geometrical edge magnitude detection in point clouds, is still an important problem that can form a basis for a wide range of applications such as object recognition and detection. Typically, the Laplacian of a field of scalars is used as gray-scale edge detector and classifier. The problem statement is defining the Gradient-Based Laplacian of a field of vectors (used for detecting 3-D surface geometrical edges) that is analogous to the Gradient-Based Laplacian of a field of scalars (used for detecting intensity edges) in the sense that it complies with the typical rules for the Laplacian operator. The contributions of the current research are: (1) proposing a novel definition of the Gradient-Based Laplacian of the field of vectors (used for detecting 3-D surface geometrical edges) using the absolute difference of the Gradient values; (2) doing behavioral analysis on the Step edge, Plane, and Ramp areas for two cases; when the absolute difference and when the signed difference are used; (3) doing performance analysis on TUM data set, and comparison study with the state of the art edge detectors on NYUD data set which shows that the proposed Gradient-Based Laplacian is efficient.

## 4.1 Introduction

The recent advances in RGB-D cameras allow the capture of three-dimensional point clouds, images, of indoor environments. The raw original surfaced point cloud is organized in geometrical surfaces of one pixel width that reflects the environment that is captured by the depth sensor. Each of the point cloud pixels holds the normal vector that is perpendicular to its surface. Segmenting geometrical surfaces of the point cloud of the indoor environment into primitive shapes remains a big problem. A novel algorithm for the Gradient magnitude of a field of vectors was proposed and implemented in [3], which is analogous to the Gradient of a field of scalars. The surfaced Gradient point cloud also consists of pixels that are organized in geometrical surfaces of one pixel width. Each of its pixels holds the value of the Gradient magnitude of the surfaced original point cloud. The Gradient point cloud is successfully classified into two types of segments of pixels:

(1) Plane edges, (2) Step and Ramp edges. In literature of the field of scalars of the gray-scale images, the 2nd derivative, the Laplacian can further segments the gray-scale intensity images into two segments: (1) Plane and Ramp intensity areas, and (2) Step intensity edges.

The **objective** is to design and implement the second-order derivative, the Gradient-Based Laplacian of a field of vectors in an analogous manner to the definition of the Gradient-Based Laplacian of a field of scalars, which typically classifies two types of segments of the geometrical surfaces of the point cloud: (1) the Plane and Ramp areas; (2) and the Step edge areas.

Therefore, the **problem statement** is defining the Gradient-Based Laplacian, the 2nd derivative of a field of vectors edges detector that is analogous to the Gradient-Based Laplacian of the field of scalars; and that produces two types of segments: (1) Step edges areas segment; (2) Plane and Ramp areas segment of the point cloud 3-D surfaces of the environment captured by the depth sensor.

The **contributions** of this research are: (1) Proposing a novel algorithm of the Gradient-Based Laplacian of a field of vectors (used for detecting 3-D surface geometrical edges) that is based on the novel algorithm of the Gradient of a field of vectors, that was proposed in [3] research. In the Gradient-Based Laplacian algorithm, the first predefined operation of the convolution is defined as the absolute scalar difference of the Gradient values of the neighbor pixels from the Gradient value of the center pixel, in contrast to the existing work [10] of gray-scale edge detection, in which this is defined as the signed Gradient difference operation. (2) Doing behavioral analysis on Step edge, Plane, and Ramp areas for two cases; when the absolute difference and when the signed difference are used in order to prove that the former is the right one and the latter is the wrong. And (3) doing performance analysis on TUM data set, and comparison study with the state of the art edge detectors on NYUD data set.

From this point on, this chapter is organized as follows: the remainder of this section is a review of existing researches on computing the Laplacian. The novel Laplacian algorithm is presented in section 4.2, the design of the Laplacian algorithm is presented in section 4.3. The behavioral analyses is presented in section 4.4 and performance analyses, comparison Study, and experimental



Results are discussed in section 4.5. Conclusions and directions of future work are presented in section 4.6.

### 4.1.1 Literature Review

The existing works on the definition of the Laplacian compute it directly from the original signal [7], or from the Gradient image [10]. In the former one, the Laplacian mask was static, the first predefined operation that this mask implemented was the signed scalar difference between the gray-scale intensity value of the center pixel and the intensity value of the neighboring pixels. Some approaches to compute the Laplacian can be found in [62–68].

An approach to the Laplacian of the 3-D field of vectors of the RGB images [7] was also defined based on vector analysis as a “vector whose components are equal to the Laplacian of the individual scalar components of the input vector”.

In contrast, in the current work, the definition of the Gradient-Based Laplacian of a field of vectors is proposed in a mathematically consistent manner to the derivative of the Gradient of a field of vectors. The novel definition and implementation of this Gradient was proposed in the [3] research.

On the other hand, several methods for computing the Gradient of a 3D field of scalars and 3D field of vectors of the point cloud have been developed [69–84]. However, none of these produce a Gradient which can be used to compute the Gradient-Based Laplacian of a field of vectors.

The novel Gradient magnitude definition for a field of vectors was presented in [3], where the vector is handled as solid unit vector, without decomposing it along its components. Also new definitions for the Plane, Ramp, and Step geometrical edges were proposed [3] as follows: (1) a Plane edge corresponds to an area with small rate of change in the direction of its normal vector (i.e., the rate of change is within a given threshold), as shown in figure 4.1(B1); (2) a Step edge corresponds to an area with significant rate of change of the direction of its normal vector (i.e., above a given threshold), as shown in figure 4.1(C1); (3) a Ramp edge corresponds to an area with constant rate change in the direction of its normal vector, as shown in figure 4.1(D1). The change

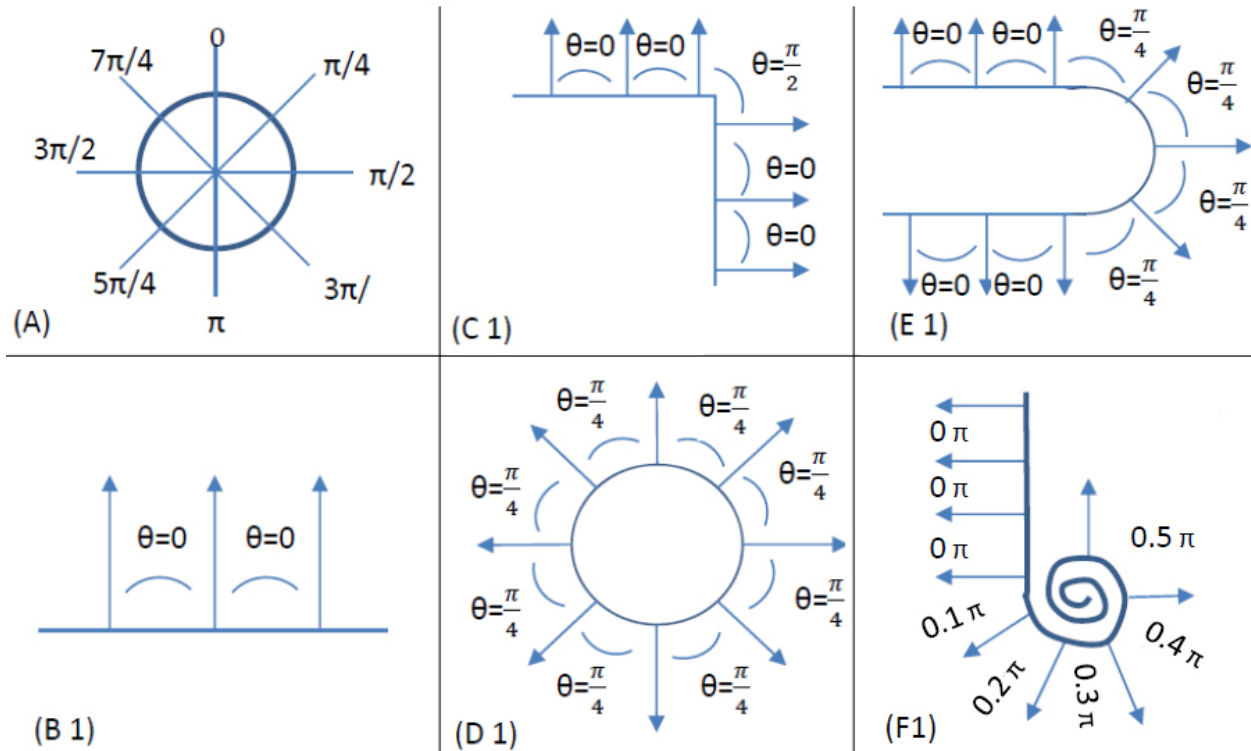


Figure 4.1: Cross section of geometrical shape of the surface of the surfaced point cloud. (A) Angles Direction Measurements. (B1), No Change. (C1), Step. (D1), Ramp. (E1), Onset and End of Ramp. (F1), series of Step edges of Spiral of constant acceleration [3].

in direction is measured by the angle between the normal vectors of the neighbor pixels and the center pixel. It is worth mentioning here that in physics one can define the distance between two pixels, as angle between the normal vectors of the neighbor pixels, the first-order derivative as the speed, and the second-order derivative as the acceleration.

## 4.2 Gradient-Based Laplacian Spatial Filter of a Field of Vectors

In this section, the Gradient-Based Laplacian geometrical edge detector spatial filter of a field of vectors, the second order derivative is described.

### 4.2.1 Definition of the Gradient-Based Laplacian for two-dimensional field of vectors

First, the Gradient-Based Laplacian of a 2-D field of vectors signal, i.e., a cross section of 3-D field of vectors is defined. That is, the behavior of the Gradient-Based Laplacian in Plane areas, onset and end of Step and Ramp areas, and along the Ramp areas is of special interest. By analogy to the 2nd derivative of a 2-D intensity image [7], the 2nd derivative of the 3D surfaced geometrical point cloud must satisfy the following requirements: (1) must be zero in Plane areas; (2) must be non-zero at the onset and end of a geometrical Step or Ramp; (3) must be zero along the Ramp.

The input to the Gradient-Based Laplacian,  $\mathcal{L}_G(f(p))$ , is the Gradient point cloud,  $\mathcal{G}(f(p))$ , which is a field of scalars and which is the first-order derivative of the original signal field of vectors,  $f(p)$ , as shown in equation (4.1).

$$\mathcal{G}(f(p)) = \frac{df(p)}{dp} \quad (4.1)$$

In turn, the Gradient-Based Laplacian,  $\mathcal{L}_G(f(p))$ , is the first-order derivative of the Gradient,  $\mathcal{G}(f(p))$ , and the second-order derivative of the original signal field of vectors,  $f(p)$  as shown in equation (4.2).

$$\mathcal{L}_G(f(p)) = \frac{d\mathcal{G}(f(p))}{dp} = \frac{d^2 f(p)}{dp^2} \quad (4.2)$$

The Gradient-Based Laplacian,  $\mathcal{L}_G(f(p))$ , of only two neighbor pixels is defined by the summation of the outcome of the absolute mathematical scalar difference operations between the Gradient of the center pixel,  $\mathcal{G}(f(P_u))$ , and the Gradient of all the neighbor pixels,  $\mathcal{G}(f(P_{left}))$  and  $\mathcal{G}(f(P_{right}))$ , as shown in equation (4.3). Since the scalars of the Gradient point cloud are in  $[0, 1]$ , the Gradient-Based Laplacian is also in  $[0, 1]$ . The shortest distance over which the geometrical change can occur is between the neighboring pixels, where neighboring pixels are defined as the

closest pixels within a specific radius.

$$\begin{aligned}\mathcal{L}_G(f(P_u)) &= \frac{d\mathcal{G}(f(P_u))}{dP_u} \\ &= |\mathcal{G}(f(P_u)) - \mathcal{G}(f(P_{left}))| + |\mathcal{G}(f(P_u)) - \mathcal{G}(f(P_{right}))|,\end{aligned}\tag{4.3}$$

where in equation (4.3),  $P_u$  is an arbitrary pixel in the  $x, y$  coordinates of the two dimensional signal field of vectors; also,  $P_u$  indicates the center pixel of the signal area under the mask;  $\mathcal{G}(f(P))$  is the Gradient magnitude of its pixel argument;  $P_{left}$  and  $P_{right}$  are the pixels closest to  $P_u$ , to the left and right, respectively, within a specific radius. Letting  $P_{left} = P_1$  and  $P_{right} = P_2$ , equation (4.3) becomes equation (4.4).

$$\begin{aligned}\mathcal{L}_G(f(P_u)) &= \frac{d\mathcal{G}(f(P_u))}{dP_u} \\ &= |\mathcal{G}(f(P_u)) - \mathcal{G}(f(P_1))| + |\mathcal{G}(f(P_u)) - \mathcal{G}(f(P_2))|.\end{aligned}\tag{4.4}$$

More generally, let  $f1_{P_u-P_i}$  denotes the first predefined operation,  $f1_{P_u-P_i}$ , as it will be described in the following sections. Where in equation 4.5,  $n$  is the maximum number of neighbors.  $f1$  is a scalar value that is the absolute difference between the Gradient magnitude of the two neighboring pixels  $P_u$  and  $P_i$ .

$$f1_{P_u-P_i} = |\mathcal{G}(f(P_u)) - \mathcal{G}(f(P_i))|, i = 1, \dots, n,\tag{4.5}$$

For  $N$  neighboring pixels, equation (4.4) can then be rewritten as equation (4.6), which is the second predefined operation to compute the Laplacian.

$$\begin{aligned}\mathcal{L}_G(f(P_u)) &= \frac{d\mathcal{G}(f(P_u))}{dP_u} \\ &= f2(P_u) = \sum_{i=1}^N f1_{P_u-P_i}.\end{aligned}\tag{4.6}$$

**Definition 4.2.1.** The Gradient-Based Laplacian,  $\mathcal{L}_G(f(P_u))$ , is defined as the absolute change between the Gradient values of the center pixel and its neighbor pixels by the change of the center pixel.

The center pixel of the signal area under the mask is changed when the mask visits a new pixel of the signal.

### 4.2.2 Definition of the Gradient-Based Laplacian for three-dimensional field of vectors

Next, the implementation of Gradient-Based Laplacian, the second order derivative, of a 3-D field of vectors of the geometrical surface of the surfaced point cloud is defined. The second-order derivative on the surfaced gradient point cloud is implemented; where the surfaced point cloud is defined as a group of pixels that are arranged in surfaces of one pixel width like geometrical shapes that reflects the environment that is captured by the depth sensor.

The approach to compute the second-order derivative consists of defining a mathematical expression of the second-order derivative and then constructing a surfaced dynamic filter mask based on this mathematical expression. The dynamic mask should be isotropic (i.e., invariant with respect of the filter rotation). Therefore, the Laplacian is an isotropic derivative operator. Given a location  $P_u$ , in the geometrical surface of a point cloud, the Gradient-Based Laplacian is defined by equation (4.7).

$$\mathcal{L}_G(f(P_u)) = \nabla \mathcal{G}(f(P_u)) \quad (4.7)$$

Where  $P_u$ , is the center pixel of the signal area under the mask;  $f(P_u)$  is is the original signal point cloud normal vectors.

More precisely, where  $P_u$  is the center pixel of the signal area under the mask;  $P_{left}$ ,  $P_{right}$  are the closest pixels to  $P_u$  on the left and right respectively. Similarly for  $P_{up}$ ,  $P_{down}$  that are the closest pixels to  $P_u$  on the up and down respectively. The combined discrete Gradient-Based Laplacian for the horizontal and vertical local axes, for the directions *right*, *left*, *up*, and *down* is

defined by equation (4.8).

$$\begin{aligned}
 \mathcal{L}_G(f(P_u)) &= \nabla \mathcal{G}(f(P_u)) = \\
 &|\mathcal{G}(f(P_u)) - \mathcal{G}(f(P_{left}))| \\
 &+ |\mathcal{G}(f(P_u)) - \mathcal{G}(f(P_{right}))| \\
 &+ |\mathcal{G}(f(P_u)) - \mathcal{G}(f(P_{up}))| \\
 &+ |\mathcal{G}(f(P_u)) - \mathcal{G}(f(P_{down}))|
 \end{aligned} \tag{4.8}$$

More generally and because the point cloud is surfaced, for  $N$  number of neighbors, and using  $f1_{P_u-P_i}$  of equation (4.5), equation (4.8) leads to equation (4.9).

**Definition 4.2.2.** Given an arbitrary location,  $P_u(x, y, z)$ , in the geometrical surface of a point cloud, the Gradient-Based Laplacian of a three-dimensional field of vectors is defined by equation 4.9.

$$\begin{aligned}
 \mathcal{L}_G(f(P_u(x, y, z))) &= \nabla g(f(P_u(x, y, z))) \\
 &= f2 = \sum_{i=1}^N f1_{P_u-P_i},
 \end{aligned} \tag{4.9}$$

Where in equation 4.9,  $x, y, z$  are the global coordinate axes of the point cloud.

### 4.3 Design of The Gradient-Based Laplacian Algorithm

The Gradient-Based Laplacian is implemented by the convolution process, the neighborhood search method, the dynamic mask, the two predefined mathematical expressions of equation (4.5) and equation (4.9), and the threshold operation as described below.

The convolution is defined as evaluating a mask of two-dimensional field of scalars over the surface of the three-dimensional field of scalars of the gradient point cloud signal. When the mask visits a new pixel of the signal, the two predefined operations are performed. After the mask slides over all the pixels of the signal, the result is a field of scalars of the Gradient-Based Laplacian of the field of vectors. This output is stored in a newly created point cloud in where the  $x, y, z$  coordinates of the signal are preserved, and each pixel value is its computed Laplacian value.

The neighborhood search method that is used to find the neighbor pixels is the K-D tree method.

Table 4.1: Signal Area Under Mask

$P_1$	$P_2$	$P_3$
$G(P_1)$	$G(P_2)$	$G(P_3)$
$P_4$	$P_u$	$P_5$
$G(P_4)$	$G(P_u)$	$G(P_5)$
$P_6$	$P_7$	$P_8$
$G(P_6)$	$G(P_7)$	$G(P_8)$

Table 4.2: Dynamic Mask

$P_1$	$P_2$	$P_3$
$G(P_u)$	$G(P_u)$	$G(P_u)$
$P_4$	$P_u$	$P_5$
$G(P_u)$	$G(P_u)$	$G(P_u)$
$P_6$	$P_7$	$P_8$
$G(P_u)$	$G(P_u)$	$G(P_u)$

The dynamic mask is shown in table (4.2) where all of the values of its pixels are set equal to the Gradient value of the center pixel of the signal area under the mask. This organization of the mask is required to implement the two predefined mathematical expressions that define the Gradient-Based Laplacian algorithm. The dynamic mask is updated every time it visits a new pixel of the signal by setting the mask pixels values to the Gradient magnitude of that newly visited center pixel of that signal area under the mask. The signal area under the mask is shown in table (4.1) in which every pixel holds the Gradient magnitude value of the Gradient point cloud of the input signal.

The following two predefined operations implement the approximation of the Gradient-Based second-order derivative that satisfies the conditions of the Gradient-Based Laplacian operator of a field of vectors that is defined by equation (4.8).

1. The first predefined operation,  $f1_{P_u-P_i}$ , is the derivative term which is needed in the approximation of equation (4.8) and shown in equation (4.5). It equals the absolute value of the difference between the Gradient magnitude of the center pixel,  $G(P_u)$ , and the Gradient magnitude of every neighboring pixel,  $G(P_i)$ . Therefore, this operation produces a scalar real value for every neighboring pixel  $P_i$ .
2. The second predefined operation,  $f2$ , is the sum of scalars that were produced from the first predefined operation, as shown in equation (4.9). This operation results in the Gradient-Based Laplacian scalar value.

The threshold operation is performed by setting to zero every Gradient-Based Laplacian that is below a certain given threshold value.

### 4.3.1 The Classification

The classification results of the Gradient-Based Laplacian consists of two segments of the pixels of the three-dimensional geometrical surface of the field of vectors, which are: (1) the first segment consists of areas of constant direction of their vectors (e.g. Planes) and areas of constant rate of change in the direction of their vectors (e.g. Ramps); and (2) the second segment consists of areas of varying rate of change of the direction of their vectors (e.g. Step edges).

### 4.3.2 The Pseudo Code and The Complexity

The pseudo code of the Gradient-Based Laplacian algorithm of the field of vectors is shown in algorithm (1). The input is the Gradient magnitude point cloud. The output is the Gradient-Based Laplacian point cloud. The algorithm has two nested for-loops, one controlled by the size of the point cloud,  $|P|$ , the other by the number of neighbors,  $\gamma$ . Therefore, the complexity is of order  $O(|P|\gamma)$ .

## 4.4 Behavioral Analyses

To illustrate the algorithm of the second-order derivative, the Gradient-Based Laplacian of a field of vectors, consider examples B1, C1, D1, E1, and F1 of figure (4.1). Those examples illustrate all possible types of edges and shapes of the geometrical surface of the surfaced 3-D point cloud. Example (B) shows a Plane surfaced area; example (C) shows a Step edged surfaced area; example (D) shows a Ramp surfaced area; example (E), from top-left to bottom-left, shows a series of a three connected surfaces of Plane-Ramp-Plane, for the purpose of describing the behavior of the second-order derivative at the onset and end of the Ramp areas, and at the pixels that precede and come after the onset and end of the Ramp areas; and example (F) shows a series of Step edges of a spiral of constant acceleration that is connected to a Plane area. The Spiral area has decreasing rate of change of direction with constant acceleration. The rate of change of the direction of the last four vectors of the plane area is zero, since they have constant direction. The purpose of this example is



---

**Algorithm 1** The Gradient-Based Laplacian Algorithm

---

1: Input:

1. Gradient Point Cloud of a Field of Vectors= $G(P\{i\})$  that is defined in [3]
2. Neighbors Search Method and Parameters:-
  - *Method*  $\Omega(.) = KDTree$
  - *Search Radius* = (0.05)
  - *Max number of neighbors*  $\gamma = (4)$
3. Laplacian Threshold Value,  $L_{threshold}$ .

2: Output:

- Point Cloud={Output} ▷ The Laplacian

3: **procedure** MAIN ALGORITHM

4:   **for**  $i = 0$  to size  $G(P\{i\})$  **do**

5:     *Laplacian* = 0

6:     **if** ( $P\{i\} = \text{Nan}$ ) **then** continue

7:      $\{Bc\} = \Omega(G(P\{i\}))$

8:     **if** ( $\{Bc\} = 0$ ) **then** continue

9:     **for**  $j = 1, j \leq \gamma, j++$  **do**

10:       **if** ( $P\{i\} \vee G(P\{i\}) \vee Bc\{j\} \vee G(Bc\{j\}) = \text{NAN}$ ) **then** continue ▷ Laplacian Calculation Start

11:        *Difference* =  $|G(P\{i\}) - G(Bc\{j\})|$

12:        *Laplacian* + = *Difference* ▷ Laplacian Calculation End

13:        **if** *Laplacian* <  $L_{threshold}$  **then** *Laplacian* = 0

14:        *Output*{ $i$ }(rgb) = *Laplacian*

15:    Return {Output}

---

to describe the behavior of the second-order derivative at the Spiral areas of constant acceleration. The directions of the vectors of the examples are measured with the standard measurement that is illustrated in figure (4.1)(A), where the vector that points to the top measures zero  $\pi$  radian angle, then the radian angular measurement of the directions of the other vectors increase in the clock-wise direction. Those five examples are explained in the following paragraphs where two cases of the Gradient-Based Laplacian for each example are discussed: (1) the first case is when the absolute difference is used; (2) the second case is when the signed difference is used.

Figures 4.1(B1, C1, D1, E1, F1) illustrate their corresponding geometrical area, specifically to where all the normal vectors point.

Figures 4.2(B2), 4.2(C2), 4.2(D2), 4.2(E2), and 4.2(F2) show the corresponding scan-line of the radian angle measurements of the directions of those normal vectors, and a plot to visualize those measurements.

Figures 4.2(C3), 4.2(D3), 4.2(E3), and 4.2(F3) show the first-order derivative.

Example (B) is illustrated in figure 4.1(B1), and figure 4.2 (B2-B5): (1) in the absolute difference case, the second derivative of Plane areas is zero, as shown by a scan line and a chart in figure 4.2(B4); (2) and in the signed difference case, the second derivative of Plane areas is also zero, as shown by a scan line and a chart in figure 4.2(B5).

Example (C) which is illustrated in figure 4.1(C1), and figure 4.2(C2-C5): (1) in the absolute difference case, the second-order derivative is non-zero at the Step edges, as well as at the pixels that precede and come after this step edges as shown in figure 4.2(C4); (2) and in the signed difference case, the second-order derivative is non-zero at the Step edges, and it has negative values at the pixels that precede and come after those Step edges. This case shows the zero-crossing behavior, that is when the line that connects the Step edge pixels with the pixels that precede and come after them crosses the horizontal line of the zero value and changes its sign from the negative to the positive (or vice versa) as shown in figure 4.2(C5). Gonzales and Woods stated in [7] that the zero-crossing behavior is quite useful for locating edges in the field of scalars gray-scale images. Therefore and similarly, it is suggested that the zero-crossing behavior should be quite useful for

locating geometrical edges in geometrical surfaces of the field of vectors.

Example (D) is illustrated in figures 4.1(D1) and 4.2(D2-D5): (1) in the absolute difference case, the second-order derivative is zero along the Ramp areas as shown in figure 4.2(D4); (2) and in the signed difference case, the second-order derivative is zero along the Ramp areas as shown in figure 4.2(D5).

Example (E) is illustrated in figures 4.1(E1) and 4.2(E2-E5):(1) in the absolute difference case, the second-order derivative at the pixels of the onset and end of the Ramp area is non-zero; and at the pixels that precede and come after the pixels of the onset and end of Ramps respectively is non-zero as well as shown in figure 4.2(E4); (2) and in the signed difference case, the second-order derivative at the pixels of the onset and end of the Ramp area is zero, which violates the rules of the second-order derivatives that state that it must be non-zero. Also, in this case, the second-order derivative at the pixels that precede the onset, and come after the end of Ramps is a negative non-zero value; and at the pixels that come after the onset and precede the end of the Ramp is a positive non-zero value as shown in figure 4.2(E5). This shows that the line that connects the pixels that precede and come after the onset and the line that connects the pixels that precede and come after the end cross the horizontal line of the zero value at the onset and end pixels respectively. As mentioned in the preceding paragraph, this behavior is called the zero crossing which is quite useful in locating edges.

Example (F) is illustrated in figure 4.1(F1), and figure 4.2(F2-F5): (1) in the absolute difference case, the second-order derivative is constant non-zero at the pixels of the Spiral area of constant acceleration. Then the second-order derivative decreases until it becomes zero in the plane areas of zero acceleration as shown with a chart that visualizes it in figure 4.2(F4); (2) and in the signed difference case the second-order derivative at the pixels of the spiral area of constant acceleration is zero, which is a wrong behavior. The second-order derivative is -0.1 for the first two pixels of the Plane area, and it is zero for the last two pixels of the plane area. This is a wrong behavior that the second-order derivative must not show. However, for future work, this behavior may be useful in locating the edges that connect spiral areas of constant acceleration with Plane areas as shown

with a chart that visualizes it in figure 4.2(F5).

In conclusion, when the absolute difference is used in computing the second-order derivative, it can be seen that: (1) it is zero in Plane areas (example B); (2) it is non-zero at Step edges (example C); (3) it is zero along the Ramp areas (example D); (4) it is non-zero at the onset and end of the Ramp areas (example E); (5) it has non-zero constant values at the Spiral areas of constant acceleration, then it decreases until it becomes zero when it transitions from the Spiral to the Plane area (example F). Therefore, in this case, the second-order derivative is consistent with the requirements of the second-order derivative of the field of scalars.

When the signed difference is used to compute the second-order derivative, it can be seen that: (1) it is zero in Plane areas (example B); (2) it is non-zero at Step edges, and it shows the zero crossing behavior (example C); (3) it is zero along the Ramp areas (example D); (4) it is zero at the onset and end of the Ramp areas (example E) which contradicts the conditions of the second-order derivatives, though it also shows the zero crossing behavior at the onset and end of Ramps; (5) it has zero values at the Spiral areas of constant acceleration, then it is -0.1 at the first two pixels of the plane area, when it transitions from the Spiral to the Plane area, and it is zero in Plane areas (example F). Therefore, in this case, the second-order derivative is not consistent with the conditions of the second derivatives of the field of scalars, even though it shows the zero crossing behavior which is useful for locating edges.

## 4.5 Performance Analyses, Comparison Study, and Experimental Results

In this section, performance analyses of the proposed Gradient-Based Laplacian edge detector on the TUM data set are done. After that, comparison study of it on the NYUD data set with the well-known edge detectors in the literature is done. Non of any pre-processing on the data, nor any post-processing on the result were done.

The Precision-Recall frame work that Lejeune *et al.* used in [9] was done, in order to analyze

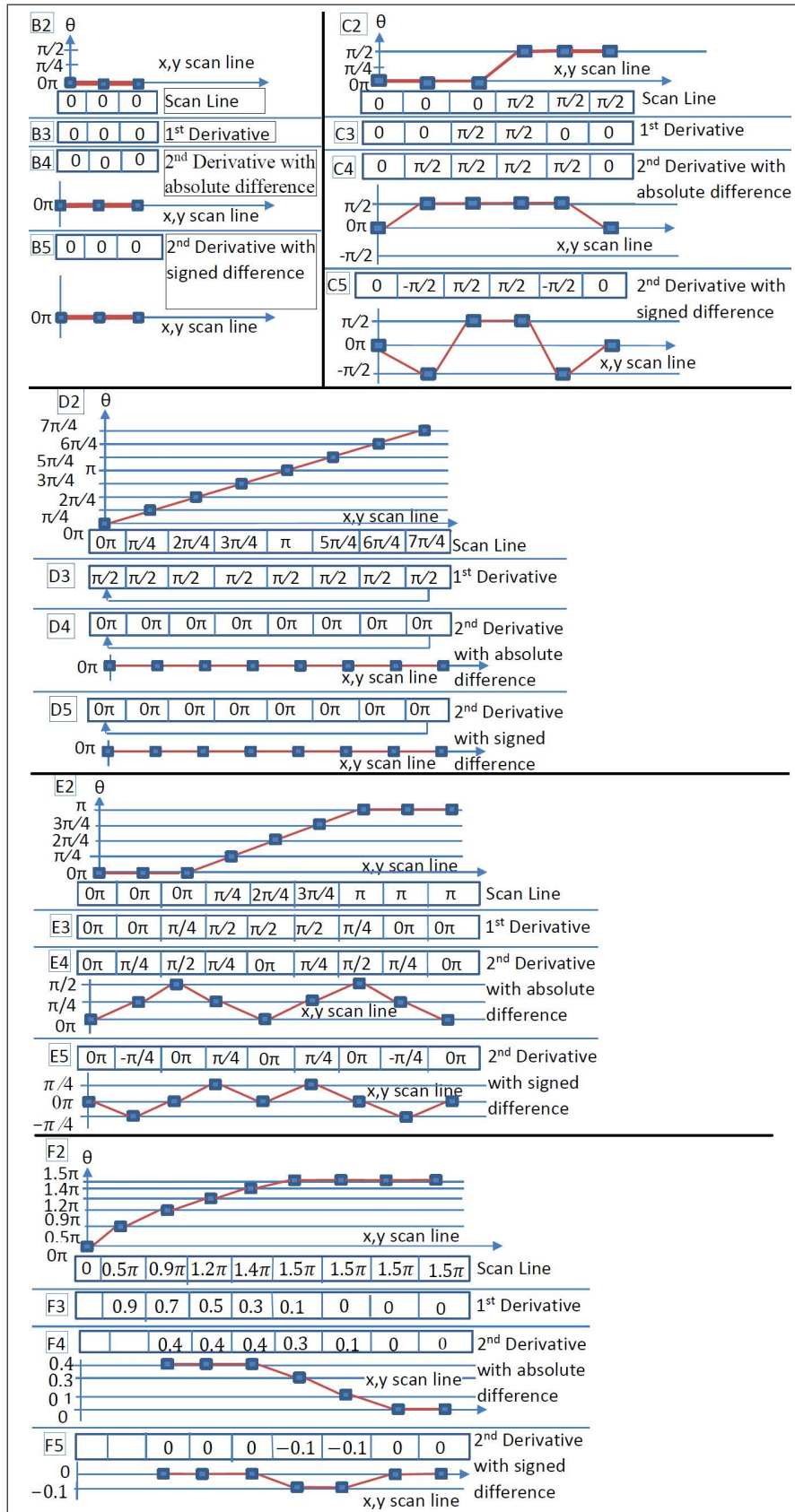


Figure 4.2: Shows examples B, C, D, E, and F.

the performance of the Gradient-Based Laplacian edge detector. This Precision-Recall framework has three indicators: (1) the ODS which is a measure of the best F score of the harmonic mean of the precision and the recall when implemented on the data set with the same set of parameters; (2) the OIS which is the aggregate F score resulted when implemented on the data set with the best set of parameters of each image;(3) and the AP that is the average of the precision along the full recall range that represents the area under the precision-recall curve.

### 4.5.1 Performance Analyses on The TUM data set

First, the performance on the Gradient-Based Laplacian edge detector on two examples of the TUM data set (a box and a sphere objects) [85] was analyzed. Those two examples include all the three types of geometrical edges the plane area of the sides of the box, the step edges of the box edges, and the ramp area of the surface of the sphere. The ground truth of the data set was annotated.

By visual observation of figure 4.3 that shows the Gradient-Based Laplacian edges magnitude for the box and the sphere examples, it can be verified that the Gradient-Based Laplacian classifies two segments: (1) the first segment consists of the planes of the box sides (Plane area), and the ball round shape (Ramp area) which are segmented with light color; (2) and the second segment consists of the box sides intersections (Step edges) with dark color.

Table 4.3 presents the best performance of the Gradient-Based Laplacian detector obtained by extensively searching the threshold parameter space on the TUM data set. The threshold parameter is the most important parameter of the Gradient-Based Laplacian edge detector.

The performance analysis on the ODS, precision, and recall with respect to the threshold parameter scale on the TUM data set show that the threshold  $\theta_{threshold}$  produces the best performance for a value around 0.17.

The duration of the processing time was 79.2402 seconds of total time, including the time spent for the preparation (the neighborhood search method time). The size of the point clouds was 307200 points. The processor was Intel Core i7-4710HQ CPU @2.50 GHz×8. The operating

Table 4.3: Best performance of the proposed Gradient-Based Laplacian edge detector used to detect edges on the TUM data set.

	ODS	OIS	AP
Gradient-Based Laplacian	.951	.962	.982

system was Ubuntu 64 bit. A K-D tree neighborhood search method was used, with  $n = 4$  neighbor points, radius  $r = 0.05$ . The normal vectors estimation is done by the Integral Images method.

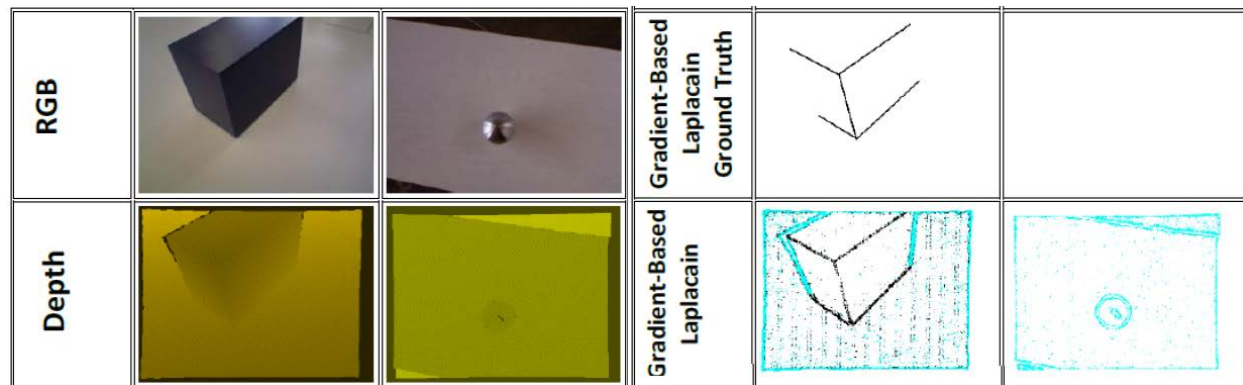


Figure 4.3: Edge detection results of the Gradient-Based Laplacian magnitude operators on the TUM data set. Black, red, green, light blue, and dark blue pixels represent the true positives, false positives caused by algorithm, false negatives caused by algorithm, false positive caused by noise, and false negative caused by noise, respectively.

#### 4.5.2 Comparative Study on the NYUD data set

Second, the performance of the Gradient-Based Laplacian edge detector was compared with the traditional edge detectors on the NYUD data set. Noting that the Gradient-Based Laplacian edge detector detects Plane areas, Ramp areas, and Step edges; in contrast with the traditional edge detectors that detects jump and roof edges that have a different definitions from the Plane, Ramp, and Step edges definitions.

In table 4.4, the performance of the proposed Gradient-Based Laplacian edge detectors on NYUD data set was presented [86]. Also the performances of the state of the art edge detectors were presented, that were obtained from Lejeune *et al.* research [9], of PED 0 [9], PED 1 [9], PED 2 [9], Jiang [87], Lejeune [88], Sobel, Dollar [89], Gupta [90], and Xie [91] on the same data set.

According to Lejeune, the machine learning edge detectors (Dollar, Gupta, Xie) were duplicated to show their complete NYUD model performance and Jump model performance.

As Lejeune *et al.* did, the parameter space of the Gradient-Based Laplacian detector was extensively searched to obtain the maximum scores results presented in table 4.4, and the same matching tolerance of 1.1% was used.

The data that were not counted are the false positive (light blue) and false negative (dark blue) because they are not caused by the Gradient-Based Laplacian detector, but they are caused by the camera depth misreading—such as: infrared reflective pixels (mirrors pixels), infrared penetrative pixels (glass pixels), noisy pixels under ambient light, pixels out of the range of the camera, and infrared shadow border pixels.

Figure (4.4) shows the edge detection magnitude results, produced by the Gradient-Based Laplacian operator, from the same representative selection of images that Lejeune *et al.* used in figure (9) of their research [9] in which they show the edge detection results by PED 1 [9], Lejeune [88], Gupta [90], and Xie [91].

Note that, the ground truth of the Gradient-Based Laplacian operator is different from the ground truth of Lejeune *et al.* detectors in [9].

The performance analysis results presented in table (4.4) show that the Gradient-Based Laplacian detector performs above the machine learning detectors of complete model on the NYUD data set, therefore it is more robust to noises that preserve 3-D surface edges geometrical features. Machine learning detectors perform better than the PED 0, PED 1, and PED2 detectors. PED 1, PED perform better than Dollar, Gupta, Xie of the Jump model. Jiang and Sobel detectors perform the worst.

Although it has a thick edges, but the extra thickness is only one pixel width along each side of the edges; therefore, it is negligible and not disadvantage to further applications. The Gradient-Based Laplacian detector is efficient; more robust to noise. It is not subject to over fitting or under fitting, and it does not require training like the machine learning methods, it does not use any surface model like PED, and it can be used for any depth sensor.





Figure 4.4: This figure shows the edge detection results of the Gradient Based Laplacian magnitude operators on the NYU Depth data set. Black, red, green, light blue, and dark blue pixels represent the true positives, false positives caused by algorithm, false negatives caused by algorithm, false positive caused by noise, and false negative caused by noise, respectively.

Table 4.4: Best performance indicators obtained for each of the 13 detectors on the NYU Depth dataset. The first 12 performance indicators are obtained from Lejeune *et al.* [9].

	ODS	OIS	AP
Jiang [87]	.447	.487	.574
Lejeune [88]	.537	.568	.562
Sobel	.490	.519	.560
Dollar [89] (NYUD model)	.642	.655	.660
Gupta [90] (NYUD model)	.640	.650	.660
Xie [91] (NYUD model)	.682	.695	.702
Dollar [89] (JUMP model)	.519	.529	.389
Gupta [90] (JUMP model)	.541	.561	.507
Xie [91] (JUMP model)	.341	.349	.123
PED 0 [9]	.502	.504	.705
PED 1 [9]	.554	.577	.614
PED 2 [9]	.541	.569	.595
Gradient-Based Laplacian	.683	.695	.703

## 4.6 Conclusions and Future Work

A novel algorithm was proposed to compute the second-order derivative of a field of vectors, Gradient-Based Laplacian for 3-D geometrical edge detection in which the first predefined operation of the convolution was defined as the absolute scalar difference of the Gradient values of the neighbor pixels from Gradient value of the center pixel, in contrast to the existing work [10] of gray-scale edge detection, in which this is defined as the signed Gradient difference operation. This Gradient-Based Laplacian complies with the rules of the second-order derivative. It classifies the pixels of the point cloud of the three-dimensional field of vectors into two classes: (1) pixels of Plane and Ramp areas; and (2) pixels of Step areas. Behavioral analyses were done on the Step edge, Plane, and Ramp areas for two cases when the absolute difference and when the signed difference are used in order to prove that the former is the correct approach (and the latter is the wrong one). Performance analyses were done on TUM data set, and comparison study was done with the state of the art edge detectors on NYUD data set as well. Future work will consider segmentation of the Ramp areas only.

## **Chapter 5**

# **The Unsharp Masking of a Field of Vectors**

## Contents

---

5.1	Introduction	77
5.2	Literature Review	79
5.2.1	Traditional Edge Detection Methods	79
5.2.2	Unsharp Masking of a field of scalars	79
5.2.3	The Gradient and Gradient-Based Laplacian of a field of vectors	80
5.3	The Smoothing Spatial Filter of a Field of Vectors	81
5.4	The Unsharp Masking of a Field of Vectors	84
5.4.1	Definition of The Unsharp Masking	84
5.4.2	Classification	87
5.4.3	Pseudo Code and Complexity	88
5.5	Behavioral analyses	88
5.6	Performance Analyses, Comparison Study, and Experimental Results	90
5.6.1	Performance Analyses on The TUM data set	91
5.6.2	Comparative Study on the NYUD data set	92
5.7	Conclusions and Future Work	93

---

### Abstract

The detection of the edges magnitude and direction of the geometrical surfaces of the indoor environment remains an important problem. This problem has a wide spectrum of applications including object recognition and detection, autonomous navigation systems, and three-dimensional localization and mapping.

The problem statement of the current research is to detect edges magnitude and direction of the geometrical surfaces of the point cloud field of vectors of the indoor environment which is captured by a depth sensor; and the Cartesian Components-Wise mathematical subtraction operation used in the Unsharp Masking of a field of scalars, is not useful to develop the Unsharp Masking of a field of vectors for geometrical edge magnitude and direction detection in point cloud surfaces. The objective of the current research is to propose a novel algorithm of the Unsharp Masking that is applicable to the field of vectors and analogous to the Unsharp Masking of the field of scalars. The **contributions** of the current research includes: (1) proposing a novel algorithm for the Unsharp Masking of a field of vectors that is analogous to the Unsharp Masking of a field of scalars; (2) doing behavioral analyses on the Step edges; Plane areas; and onset, end and along Ramp areas; and (3) doing performance analyses on TUM data set, and comparison study with the state of the art edge detectors on NYUD data set of the indoor environment that was acquired by Microsoft Kinect depth sensor.

## 5.1 Introduction

Geometrical surface edge magnitude and direction detection is still important for the applications of 3-D object recognition and detection, 3-D Simultaneous Localization and Mapping, and many others. The traditional geometrical edge magnitude detectors (such as traditional Gradient and Laplacian of a field of scalars operators) have used the **Component-Wise** vector operations; furthermore they could not be extended to detect geometrical edge directions. According to [7] "Unfortunately, the gradient discussed in section 3.6.4 is not defined for vector quantities... If accuracy is an issue, however, then obviously we need a new definition of the gradient applicable to vec-

tor quantities....As we just mentioned, the gradient we studied in Section 3.6.4 is applicable to a scalar function  $f(x, y)$ ; it is not applicable to vector functions”, where section 3.6.4 is the gradient for the field of scalars. Inspired by the above quotes, J. Al-Anssari *et al.* proposed in [3] their novel mathematical **Solid Vector** Subtraction operation, and based on it, they proposed a novel Gradient of a field of vectors edge detector in [3] as well; Also, I. Naser *et al.* proposed a novel Gradient-Based Laplacian of a field of vectors edge magnitude detector in [1] based on this novel mathematical **Solid Vector** Subtraction operation. The **objective** is to propose a novel Unsharp Masking of a field of Vectors that detects geometrical edge magnitude and direction of the 3-D point clouds, and is analogous to the traditional Unsharp Masking of a field of scalars that is used for the intensity edge detection of the gray-scale images. The **problem statement** is that the Cartesian Components-Wise mathematical subtraction operation used in the Unsharp Masking of a field of scalars, is not useful to develop the Unsharp Masking of a field of vectors for geometrical edge magnitude and direction detection in point cloud surfaces. The **methodology** is to use the mathematical **Solid Vector** Subtraction operation to propose the Unsharp Masking of a field of vectors, instead of the Cartesian Component-Wise subtraction operation that is used in the Unsharp Masking of the field of scalars. The **contributions** of the current research include: (1) Proposing a novel algorithm for the Unsharp Masking of a field of vectors that is based on the Solid Vector subtraction operation, and that is analogous to the Unsharp Masking of a field of scalars; (2) Doing behavioral analyses on the Step edges; Plane areas; and onset, end and along Ramp areas; (3) And doing performance analyses on TUM data set, and comparison study with the state of the art edge detectors on NYUD data set. This chapter is organized as follows: section 5.2 literature review; section 5.3 Smoothing spatial filter of a field of vectors; section 5.4 Unsharp Masking of a field of vectors; section 5.5 behavioral analyses; section 5.6 performance analyses, comparison study, and experimental results; and section 5.7 conclusions and future work.

## 5.2 Literature Review

### 5.2.1 Traditional Edge Detection Methods

There have been four main approaches that the traditional edge detectors followed.

The first approach that was presented in [7], has been implemented on the field of scalars images—such as gray-scale images, in order to detect the intensity edges.

The second approach that was presented in [92] has been implemented on the 2-D field of the 3-D vectors of the RGB color vectors in order to detect RGB color edges in the RGB color images.

The third approach was intended to detect the geometrical edges of the surface of the 3-D images—such as the Point Clouds. Some of these researches presented a 2-D Gradient operator that was implemented on the 2-D field of scalars depth images such as [69–73]; Other researches presented a scanline-based segmentation approach such as [74–78]; Other group of researches presented the layer and skeleton extraction method in order to detect the geometrical edges such as [79].

The fourth approach has been implemented on the three-dimensional field of scalars of the point cloud. Some researches presented the edge-based segmentation such as [70,80,81,83]; Other researches presented the surface-based segmentation such as [74] and [84]; and other researches presented the curvature-based segmentation such as [82].

It is also worth mentioning here that in [93], two derivative operators for the field of vectors were presented: (1) the first is the Divergence which produces a scalar quantity; and (2) the second is the Curl which produces a vector quantity.

Some other approaches that tackled the Laplacian can also be found in [62–68].

### 5.2.2 Unsharp Masking of a field of scalars

The method of the Unsharp Masking of a field of scalars was presented by Gonzales *et al.* in [7]. It works on the field of scalars of the gray-scale images. It applies a smoothing spatial filter on the gray-scale image to smooth it first; then it subtract the smoothed image from the original image to

get a mask. Where the mask emphasizes the edges of the gray-scale image. Then it adds the mask to the original image in order to have a sharper edges.

### 5.2.3 The Gradient and Gradient-Based Laplacian of a field of vectors

In their respective researches [3] and [1], Al-Anssari J. *et al.* and Naser I. *et al.* proposed their novel Gradient and Gradient-Based Laplacian of a field of vectors that are based on their novel mathematical **Solid Vector Subtraction** operation. Also their Gradient and Gradient-Based Laplacian edge magnitude detectors detect edges according to their proposed novel definitions of the Plane, Ramp, and Step edges. In the current research, the novel proposed Unsharp Masking is also based on their novel mathematical **Solid Vector Subtraction** operation as, and detects edges magnitude and direction according to their proposed novel definitions of Plane, Ramp, and Step edges. Therefore, the descriptions of their novel definitions of Plane, Ramp, and Step edges, and their novel mathematical **Solid Vector Subtraction** operation are presented as follows.

#### Edge Types

According to J. Al-Anssari *et al.* research [3], three edge types, geometrical Step edge, geometrical Plane, and geometrical Ramp are defined: (1) the **geometrical Step edge** is defined as the set of points where the rate of change in the local surface properties, direction of the normal vectors, is inconstant (varying) and exceeds a given threshold; (2) the **geometrical Plane** is defined as the set of points where the rate of change in the local surface properties, direction of the normal vectors, is zero, or under a given threshold; (3) the **geometrical Ramp** is defined as the set of points where the rate of change in local surface properties, direction of normal vectors, is constant without increase or decrease over a given threshold.

#### Solid Vector Subtraction Operation

According to J. Al-Anssari *et al.* research [3], the Solid Vector subtraction operation between any two vectors  $V(P_u)$  and  $V(P_i)$  produces two outcomes: (1) the Directional Difference; and (2) the



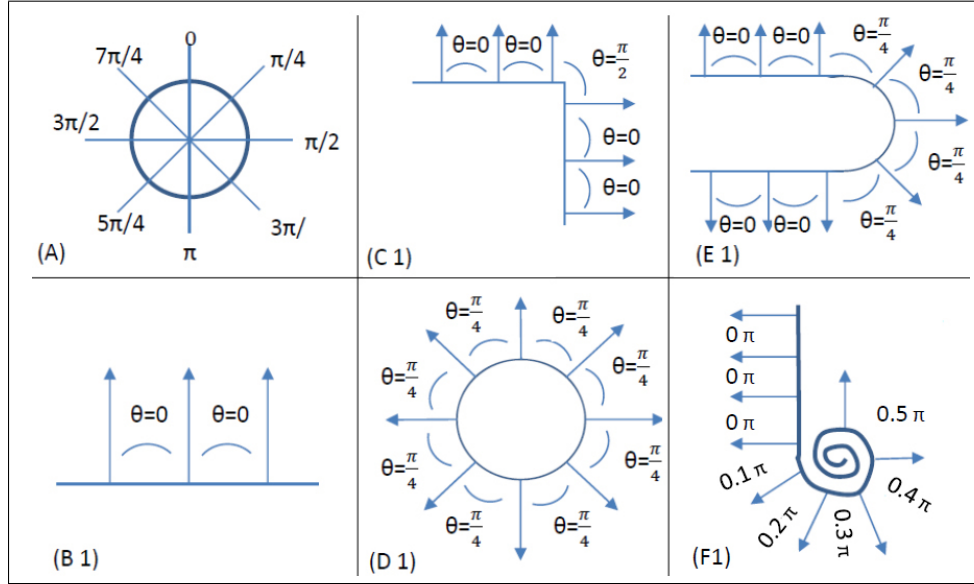


Figure 5.1: Cross section of geometrical shape of the surface of the surfaced point cloud. (A) Angles Direction Measurements. (B1), No Change. (C1), Step. (D1), Ramp. (E1), Onset and End of Ramp. (F1), Step. This figure is cited from [3].

Length Difference.

The Directional difference consists of the Absolute value, that is the radian angle  $\theta$  between the two vectors operands, and the Directional Sign Axis  $VS$ , that is the vector that connects the end point of  $V(P_i)$  to the end point of the projection of  $V(P_i)$  on  $V(P_u)$ . After assigning the absolute value  $\theta$  to the Directional Sign axis  $VS$ , the signed Directional Difference  $VT$  is produced.

The Length difference is the scalar difference between the lengths of the two vectors operands. Figure 5.2 shows this Solid Vector subtraction operation.

### 5.3 The Smoothing Spatial Filter of a Field of Vectors

The smoothing spatial filter, in this research, is applied to the signal field of vectors function,  $f(P_u)$ . The value of the signal field of vectors function at every pixel ( $P_u$ ) of an arbitrary location ( $u$ ) is the normal vector,  $V(P_u)$ , that is perpendicular to the surface surrounding this pixel, as expression (5.1) shows.

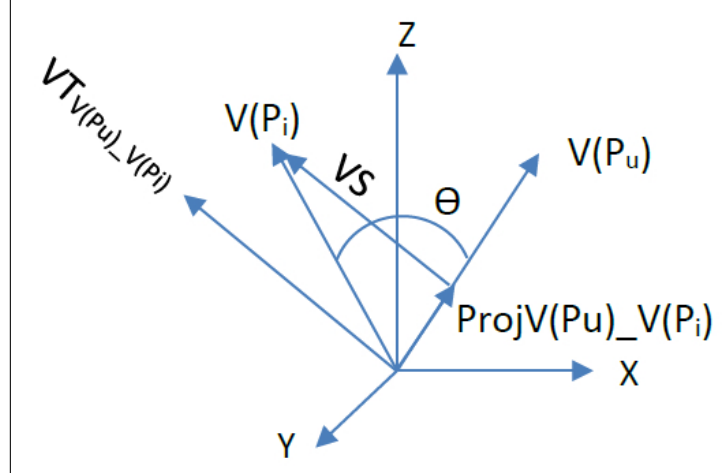


Figure 5.2: This figure shows the **Solid Vector** subtraction operation [3].

$$f(P_u) = V(P_u) \quad (5.1)$$

The smoothing filter, is used for rounding the normal vectors directions of the point cloud of the indoor environment of the field of vectors. This rounding is used for pre-processing task of the Unsharp Masking of a field of vectors that is used for edges magnitude and direction detection.

The response of the smoothing spatial filter of a field of vectors is the **Component-Wise** Average operation of the vectors contained in the signal area under the mask. The average linear spatial filter is also called a low pass filter.

In the smoothing filtering process, the value of every vector in the produced point cloud is replaced with the **Component-Wise** average of the vectors in the signal area under the filter mask. This process results in a point cloud with reduced “sharp” geometrical edges, more round, of the surfaces of the point cloud environment. All non-plane and non-ramp areas of the geometrical surface—such as Step edges, onset and end of Ramp areas are characterized by sharp normal vectors directions transitions, so averaging filters have the effect that they make the edges of the geometrical surfaces more round. One application of the averaging filter is to detect edges using the Unsharp Masking.

A basic definition of the averaging spatial linear filter,  $f(f(P_u))$ , that is applied on pixel  $(P_u)$  of

an arbitrary location ( $u$ ) of the signal function of the field of vectors,  $f$ , is the Cartesian *Component-Wise* summation of all the vectors included in the signal area under the mask. That summation is divided by the number of the vectors, as equation 5.2 shows.

$$f(f(P_u)) = \frac{1}{N + 1} \left( \sum_{n=u,1}^{n=N} f(P_n) \right) \quad (5.2)$$

The general mathematical expression of the averaging spatial filter of equation 5.3 implements the smoothing  $3 \times 3$  mask (filter) of table (5.2). Use of this filter yields the standard average of the vectors under the mask. This filter is called a box filter because all the coefficients are equal.

$$\begin{aligned} f(f(P_u)) &= \frac{1}{9} \times \\ &(1 \times f(P_1) + 1 \times f(P_2) + 1 \times f(P_3) \\ &+ 1 \times f(P_4) + 1 \times f(P_u) + 1 \times f(P_5) \\ &+ 1 \times f(P_6) + 1 \times f(P_7) + 1 \times f(P_8)) \end{aligned} \quad (5.3)$$

The smoothing spatial filter of a field of vectors is implemented by the convolution operation. The convolution is implemented by evaluating a surfaced static mask over the surface of the surfaced field of vectors. The surfaced field of vectors is defined as a group of vectors that their field points are organized in surfaces of one pixel width. The surfaces reflect the shape of the indoor environment that is captured by the depth camera sensor. When the center of the static mask visits a new pixel in the signal field of vectors, the neighborhood search method is used to find the neighbors of the center pixel of the signal, then two predefined mathematical operations are performed between the corresponding pixels of the static mask and the signal area under the mask. Table 5.1 shows the signal area under the mask that is a field of vectors. Table 5.2 shows the static mask. The static mask is a field of scalars where all the scalars are equal to 1 value. Therefore, it is called a box smoothing filter.

Equation 5.4 shows the first predefined operation,  $P1_i(f(P_u))$ . It consists of a unit scalar

Table 5.1: Signal Area Under Mask

$P_1$	$P_2$	$P_3$
$V(P_1)$	$V(P_2)$	$V(P_3)$
$P_4$	$P_u$	$P_5$
$V(P_4)$	$V(P_u)$	$V(P_5)$
$P_6$	$P_7$	$P_8$
$V(P_6)$	$V(P_7)$	$V(P_8)$

Table 5.2: Static Mask

$P_1$	$P_2$	$P_3$
1	1	1
$P_4$	$P_u$	$P_5$
1	1	1
$P_6$	$P_7$	$P_8$
1	1	1

multiplied by every vector of the signal area under the mask.

$$P1_i(f(P_u)) = 1 \times f(P_i), i = u, 1, \dots, N \quad (5.4)$$

The second predefined operation,  $P2(f(P_u))$ , is shown in equation 5.5. It consists of the Cartesian *Component-Wise* vector summation of all the output vectors of the first predefined operation divided by the scalar counts of the number of the signal pixels inside the area under the mask.

$$P2(f(P_u)) = \frac{1}{(N + 1)} \left( \sum_{i=u,1}^{i=N} P1_i(f(P_u)) \right) \quad (5.5)$$

The mask slide on every pixel of the signal field of vectors function,  $f(P)_{Original}$ , and the produced result is stored in a newly created field of vectors function,  $f(P)_{Rounded}$ , stored in a point cloud file format.

## 5.4 The Unsharp Masking of a Field of Vectors

### 5.4.1 Definition of The Unsharp Masking

The Unsharp Masking of a field of vectors is defined as a process that consists of **Solid Vector** subtracting of the original version of the vectors of the point cloud field of vectors from the rounded one. In the current research, this process is used for the detection of the edges magnitude and direction of the three-dimensional geometrical surfaces of the point cloud. This process consists of the following steps:

1. Rounding the vectors of the original point cloud field of vectors using the Smoothing spatial filter of a field of vectors described above.
2. Solid Vector subtracting the original vectors of the point cloud field of vectors from the vectors of the rounded point cloud field of vectors. The resulting Directional difference consists of the edges magnitude and direction.
3. Thresholding the Unsharp Masking edges magnitudes according to a predefined certain scalar threshold value.
4. Assigning the Unsharp Masking edges magnitude scalar values as colors to the same pixels positions of a newly created point cloud.
5. Assigning the Unsharp Masking edges direction vectors quantities as vectors to the same pixels positions of a newly created point cloud.

As the case of the smoothing spatial filter, the Unsharp Masking function is applied to the signal field of vectors function,  $f(P_u)$ . The value of the signal field of vectors function at every pixel ( $P_u$ ) of an arbitrary location ( $u$ ) is the normal vector,  $V(P_u)$ , that is perpendicular to the surface surrounding this pixel, as expressions (5.6) and (5.7) show.

$$f(P_u)_{Original} = V(P_u)_{Original} \quad (5.6)$$

$$f(P_u)_{Rounded} = V(P_u)_{Rounded} \quad (5.7)$$

Letting  $f(P_u)_{Rounded}$  denotes the rounded version of the original signal field of vectors function, where the value of this signal smoothed function at every pixel ( $P_u$ ) of an arbitrary location ( $u$ ) is the rounded (smoothed) vector of pixel ( $P_u$ ).

And letting  $f(P_u)_{Original}$  denotes the original version of the signal field of vectors function, where the value of this original signal function at every pixel ( $P_u$ ) of the same arbitrary location ( $u$ ) is the original normal vector which is perpendicular to the surface of pixel ( $P_u$ ). The Unsharp

Masking of the field of vectors,  $\mathcal{U}(f(P_u))$ , applied on the signal function at pixel ( $P_u$ ) is expressed in the form of equation 5.8.

$$\mathcal{U}(f(P_u)) = (f(P_u)_{Rounded} - f(P_u)_{Original})_{SV\_Signed} \quad (5.8)$$

Where in equation (5.8), the abbreviation  $SV\_Signed$  stands for the mathematical **Signed Solid Vector** Subtraction operation.

In equation (5.9), the Unsharp Masking edge magnitude and direction of pixel ( $P_u$ ) is represented by the Directional vector quantity,  $VT_{V(P_u)_{Rounded}-V(P_u)_{Original}}$  that is produced from the mathematical *Signed Solid Vector* Subtraction operation.

$$\mathcal{U}(f(P_u)) = \left\{ \begin{array}{l} VT_{V(P_u)_{Rounded}-V(P_u)_{Original}} \\ M_{V(P_u)_{Rounded}-V(P_u)_{Original}} \end{array} \right\} \quad (5.9)$$

The Unsharp Masking edge magnitude of pixel ( $P_u$ ),  $Mag(\mathcal{U}(f(P_u)))$ , is represented by the Euclidean length of the vector,  $VT_{V(P_u)_{Rounded}-V(P_u)_{Original}}$ , which is actually the radian angle,  $\theta_{V(P_u)_{Rounded}-V(P_u)_{Original}}$ , as shown in equation (5.10).

$$\begin{aligned} Mag(\mathcal{U}(f(P_u))) &= \\ &= \|VT_{V(P_u)_{Rounded}-V(P_u)_{Original}}\| \\ &= \theta_{V(P_u)_{Rounded}-V(P_u)_{Original}} \end{aligned} \quad (5.10)$$

The Unsharp Masking edge direction of pixel ( $P_u$ ),  $Dir(\mathcal{U}(f(P_u)))$ , is represented by the direction of  $VT_{V(P_u)_{Rounded}-V(P_u)_{Original}}$  vector, which is actually the  $VS_{V(P_u)_{Rounded}-V(P_u)_{Original}}$  Directional Sign Axis, as shown in equation (5.11).

$$\begin{aligned}
 Dir(\mathcal{U}(f(P_u))) &= \\
 &= \frac{VT_{V(P_u)Rounded-V(P_u)Original}}{\|VT_{V(P_u)Rounded-V(P_u)Original}\|} \\
 &= VS_{V(P_u)Rounded-V(P_u)Original}
 \end{aligned} \tag{5.11}$$

In the current research, only the **Signed Directional Solid Vector** Subtraction result is considered, and the **Signed Length Solid Vector** Subtraction result of equation (5.9) is neglected; because it produces the **Signed Length Solid Vector** difference,  $M$ , which is out of the scope of the application of the current research because the interest is only in detecting edges of Plane, Ramp, and Step definitions that can be computed using the Directional difference only.

Although the **Signed Length Solid Vector** difference,  $M$ , ranges from zero value to 1 value; where a zero value of  $M$  indicates that the vectors form a plane surface; and a 1 value of  $M$  indicates that the normal vectors form a polygon shape of  $(N+1)$  number of sides; however the distances between the pixels of the normal vectors of the point cloud surface are normally so small, compared to the size of the point cloud surface, to the degree that it cannot be used to classify the shape of the point cloud surface (how many sides its polygon shape has) based on the value of  $M$ .

$$f(P)_{Unsharp} = \mathcal{U}(f(P)_{Original}) \tag{5.12}$$

After the Unsharp Masking process is performed on the vectors of all the pixels of the original field of vectors signal function point cloud, a newly produced resulting function of the field of vectors of the Unsharp Masking point cloud,  $f(P)_{Unsharp}$  is created, as equation (5.12) shows.

### 5.4.2 Classification

The proposed Unsharp Masking of a field of vectors classifies the signal into two segments: (1) the first segments consists of plane areas and ramp areas; (2) the second segment consists of step

edges areas.

### 5.4.3 Pseudo Code and Complexity

The pseudo code of the Unsharp Masking algorithm, that includes the edge magnitude and direction algorithm, is shown in algorithm 2. The algorithm has two nested for loops, one controlled by the size of the point cloud,  $|P|$ , the other by the number of neighbors,  $\gamma$ . Therefore, the complexity is of order  $O(|P|\gamma)$ .

## 5.5 Behavioral analyses

Examples B, C, D, E, and F explain how the Unsharp Masking works.

Example B is shown in figures 5.1(B1), 5.3(B2,B3,B4,B5,B6). Figure 5.1(B1) shows the vectors directions profile of a plane area of the original signal. Figure 5.3(B3) shows the scan line of the radian angle measurements of the vector directions profile of the original signal. Figure 5.3(B4) shows the scan line of the radian angle measurements of the vectors directions profile of the smoothed (rounded) signal. Figure 5.3(B2) shows the rounded smoothed vectors directions profile (shown solid) superimposed on the original signal (shown dashed) for reference and shows the radian angles between every corresponding original and smoothed vectors which are equal to the edges magnitudes. Figure 5.3(B5) shows the scan line edges magnitudes of the unsharp mask obtained by **Solid Vector** subtracting the original vectors from the smoothed vectors. Figure 5.3(B6) shows a plot of the unsharp mask.

So on for example C which illustrates a Step edge, and is shown in figures 5.1(C1), 5.3(C2, C3, C4, C5, C6); example D which illustrates a Ramp area, and is shown in figures 5.1(D1), 5.3(D2,D3,D4,D5,D6); example E which illustrates the onset and end of Ramp area, and is shown in figures 5.1(E1), 5.3(E2,E3,E4,E5,E6); and example F which illustrates the Spiral area, and is shown in figures 5.3(F1), 5.3(F2,F3,F4,F5,F6).

The Unsharp Masking magnitude classification result is similar to the Gradient-Based Lapla-



**Algorithm 2** Unsharp Masking algorithm

---

1: Input:

1. Point Cloud= $\{P\}$ ,
2. Point Cloud Normal  $\{N\}$  Estimation Mehtod and Parameters:-
  - Normal Estimation Method  $\Psi(.) = \text{Avarage 3d Gradient}$
  - Max Depth Change Factor= $(0.02f)$
  - Normal Smoothing Size= $(10.0f)$
3. Neighbors Search Method and Parameters:-
  - Method  $\Omega(.) = KDTree$
  - Search Radius= $(0.05)$
  - Max number of neighbors  $\gamma = (4)$
4. Unsharp Masking magnitude threshold value  $U_{threshold}$

2: Output:

- Point Cloud= $\{Output\}$  ▷ The Unsharp Masking Edges

3: **procedure** MAIN ALGORITHM

```

4:    $\{N\} = \Psi(P)$ 
5:   for  $i = 0$  to size  $\{P\}$  do
6:      $neighbor\_vectors\_addition = 0$ 
7:      $vectors\_addition = 0$ 
8:     if  $(P\{i\} = \text{Nan})$  then continue
9:      $\{Bc\} = \Omega(P\{i\})$ 
10:    if  $(\{Bc\} = 0)$  then continue
11:    for  $j = 1, j \leq \gamma, j++$  do
12:      if  $(P\{i\} \vee N(P\{i\}) \vee Bc\{j\} \vee N(Bc\{j\})) = \text{NAN}$  then continue
▷ Normal Smoothing Calculation Start
13:       $neighbor\_vectors\_addition+ = N(Bc\{j\})$ 
14:       $vectors\_addition = N(P\{i\}) + neighbors\_vectors\_addition$ 
15:       $smoothed\_vector = vectors\_addition / (\gamma + 1)$  ▷ Normal Smoothing Calculation End
▷ Unsharp Masking Magnitude Calculation Start
16:       $dot\_product = \langle N(P\{i\}), smoothed\_vector \rangle$ 
17:       $radian\_angle = \frac{1}{\pi} \arccos \left( \frac{dot\_product}{\|smoothed\_vector\|} \right)$ 
18:       $unsharp\_magnitude = radian\_angle$  ▷ Unsharp Masking Magnitude Calculation End
▷ Unsharp Masking Direction Calculation Start
19:       $Proj_{N(P\{i\})} smoothed\_vector = dot\_product \times N(P\{i\});$ 
20:       $unsharp\_dir = smoothed\_vector - Proj_{N(P\{i\})} smoothed\_vector$ 
21:       $unsharp\_dir = unsharp\_dir / \|unsharp\_dir\|$  ▷ to make its length equal unit scalar value ▷
Unsharp Masking Direction Calculation End
22:      if  $unsharp\_magnitude < U_{threshold}$  then  $unsharp\_magnitude = 0$ 
23:       $Output\{i\}(rgb) = unsharp\_magnitude$ 
24:       $Output\{i\}(x, y, z) = P\{i\}(x, y, x)$ 
25:       $Output\{i\}(nx, ny, nz) = unsharp\_dir(nx, ny, nz)$ 
26:  Return  $\{Output\}$ 

```

---

cian spatial filter of a field of vectors of [1].

The magnitudes of the Unsharp Masking of the field of vectors is a field of scalars in which its values, at the geometrical surface Step edges and the onset and end of the ramp areas, are above certain threshold value, and emphasized (keep non-zero) after the threshold operation; while its values, at the plane and along ramp areas, are below certain threshold value, and not emphasized (keep zero) after the threshold operation.

Example B proves that there are no edge magnitudes emphasized in plane areas (all are zeros). Example C proves that there are edge magnitudes emphasized in step areas (keep non-zeros). Example D proves that there are no edge magnitudes emphasized along ramp areas (all are zeros). Example E proves that there are edge magnitudes emphasized at the onset and end of ramp areas (keep non-zeros). Example F proves that there are equal edge magnitudes emphasized at the Spiral areas of constant acceleration (keep fixed non-zeros). The edge magnitudes decrease, when the Spiral area changes to plane area. Then the edge magnitude becomes zeros at the plane area.

## 5.6 Performance Analyses, Comparison Study, and Experimental Results

In this section, performance analyses are done of the proposed Unsharp Masking edge detector on the TUM data set. After that, comparison study of it is done on the NYUD data set with the well-known edge detectors in the literature.

The Precision-Recall frame work that Lejeune *et al.* used in [9] is used, in order to analyze the performance of the proposed Unsharp edge detector. This Precision-Recall framework has three indicators: (1) the ODS which is a measure of the best F score of the harmonic mean of the precision and the recall when implemented on the data set with the same set of parameters; (2) the OIS which is the aggregate F score resulted when implemented on the data set with the best set of parameters of each image; (3) and the average precision, AP, that is the average of the precision along the full recall range that represent the area under the precision-recall curve.

There were no pre-processing on the data—such as the median filter, nor any post-processing on the result—such as the edge thinning operation involved in this algorithm.

### 5.6.1 Performance Analyses on The TUM data set

First, the TUM data set was used to analyze the performance on the Unsharp Masking edge detector. The ground truth of the data set was annotated. Two object examples were used from the TUM data set [85] a box example and a sphere example. These two examples include all the three types of geometrical edges the Plane area of the sides of the box, the Step edges of the box edges, and the Ramp area of the surface of the sphere. Six images were used for each object example to analyze the performance of the Unsharp Masking edge detector.

By visual observation of figure 5.5 that shows the Unsharp Masking edges magnitude and direction for the box and the sphere examples, it can be verified that:

- The Unsharp Masking magnitude produces two segments: (1) the first segment consists of the planes of the box sides (Plane area), and the ball round shape (Ramp area) which are segmented with light color; and (2) the second segment consists of the box sides intersections (Step edges) with dark color.
- The Unsharp Masking direction vectors: (1) in the Step edges, point to the direction of the greatest change in the geometrical surface surrounding the center vector; while (2) in Plane areas of the box, and Ramp area of the ball, have zero length (light color), because the Unsharp Masking magnitude is zero.

The most important parameter of the Unsharp Masking edge detector is the threshold  $\theta_{threshold}$  that was analyzed in order to find a default value.

Table 5.3 presents the best performance of the Unsharp Masking detector obtained by extensively searching the threshold parameter space on the TUM data set. The performance analyses on the ODS, precision, and recall with respect to the threshold parameter scale on the TUM data set shows that the threshold  $\theta_{threshold}$  produces the best performance for a value around 0.015.

Table 5.3: Best performance of the proposed Unsharp Masking edge detector used to detect edges on the TUM data set

	ODS	OIS	AP
Unsharp	.953	.964	.984

The duration of the processing time was 0.488391 seconds of total time, including the time spent for the preparation (the neighborhood search method time). The size of the point clouds was 307200 points. The processor was Intel Core i7-4710HQ CPU @2.50 GHz×8. The operating system was Ubuntu 64 bit. A K-D tree neighborhood search method was used, with  $n = 4$  neighbor points, radius  $r = 0.05$ . The normal vectors estimation is done by the Integral Images method.

### 5.6.2 Comparative Study on the NYUD data set

Second, the NYUD data set was used to compare the performance of the proposed Unsharp Masking edge detector with the traditional edge detectors. In table 5.4, the performance of the proposed Unsharp Masking edge detectors on NYUD data set [86] is presented. Also the performance of the state of the art edge detectors is presented, that was obtained from Lejeune *et al.* research [9], of PED 0 [9], PED 1 [9], PED 2 [9], Jiang [87], Lejeune [88], Sobel, Dollar [89], Gupta [90], and Xie [91] on the same data set. According to Lejeune, the machine learning edge detectors (Dollar, Gupta, Xie) were duplicated to show their complete NYUD model performance and Jump model performance.

Figure (5.6) shows the edge detection magnitude and direction results, produced by the proposed Unsharp Masking operator, from the same representative selection of images that Lejeune *et al.* used. These results can be compared with the results of figure (9) of [9]. Note that, the ground truth of the Unsharp Masking operator is different from the ground truth of Lejeune *et al.* detectors in [9].

The performance analyses results presented in table (5.4) show that the Unsharp Masking detector performs above the machine leaning detectors of complete model on the NYUD. The latter performs better than the PED 0, PED 1, and PED2 detectors. PED 1, PED perform better than

Table 5.4: Best performance indicators obtained for each of the 13 detectors on the NYU Depth dataset. The first 12 performance indicators are obtained from Lejeune *et al.* [9].

	ODS	OIS	AP
Jiang [87]	.447	.487	.574
Lejeune [88]	.537	.568	.562
Sobel	.490	.519	.560
Dollar [89] (NYUD model)	.642	.655	.660
Gupta [90] (NYUD model)	.640	.650	.660
Xie [91] (NYUD model)	.682	.695	.702
Dollar [89] (JUMP model)	.519	.529	.389
Gupta [90] (JUMP model)	.541	.561	.507
Xie [91] (JUMP model)	.341	.349	.123
PED 0 [9]	.502	.504	.705
PED 1 [9]	.554	.577	.614
PED 2 [9]	.541	.569	.595
Unsharp	.682	.696	.703

Dollar, Gupta, Xie of the Jump model. Jiang and Sobel detectors perform the worst.

## 5.7 Conclusions and Future Work

In this research, a novel Unsharp Masking of a field of vectors algorithm was proposed that is based on the mathematical **Solid Vector** Subtraction operation, and that is useful for the geometrical edges magnitude and direction detection. Behavioral analyses were done of the Step edge, Plane areas, and Ramp areas. Performance analyses on TUM data set was done, and comparison study with the state of the art edge detectors on NYUD data set was done as well. The Unsharp Masking detector used for geometrical edge detection is fast and efficient. It is not subject to over fitting or under fitting, and it does not required training like the machine learning methods, it does not use any surface model like PED, and it can be used for any depth sensor. Future works can be developing 3-D object recognition and detection applications.

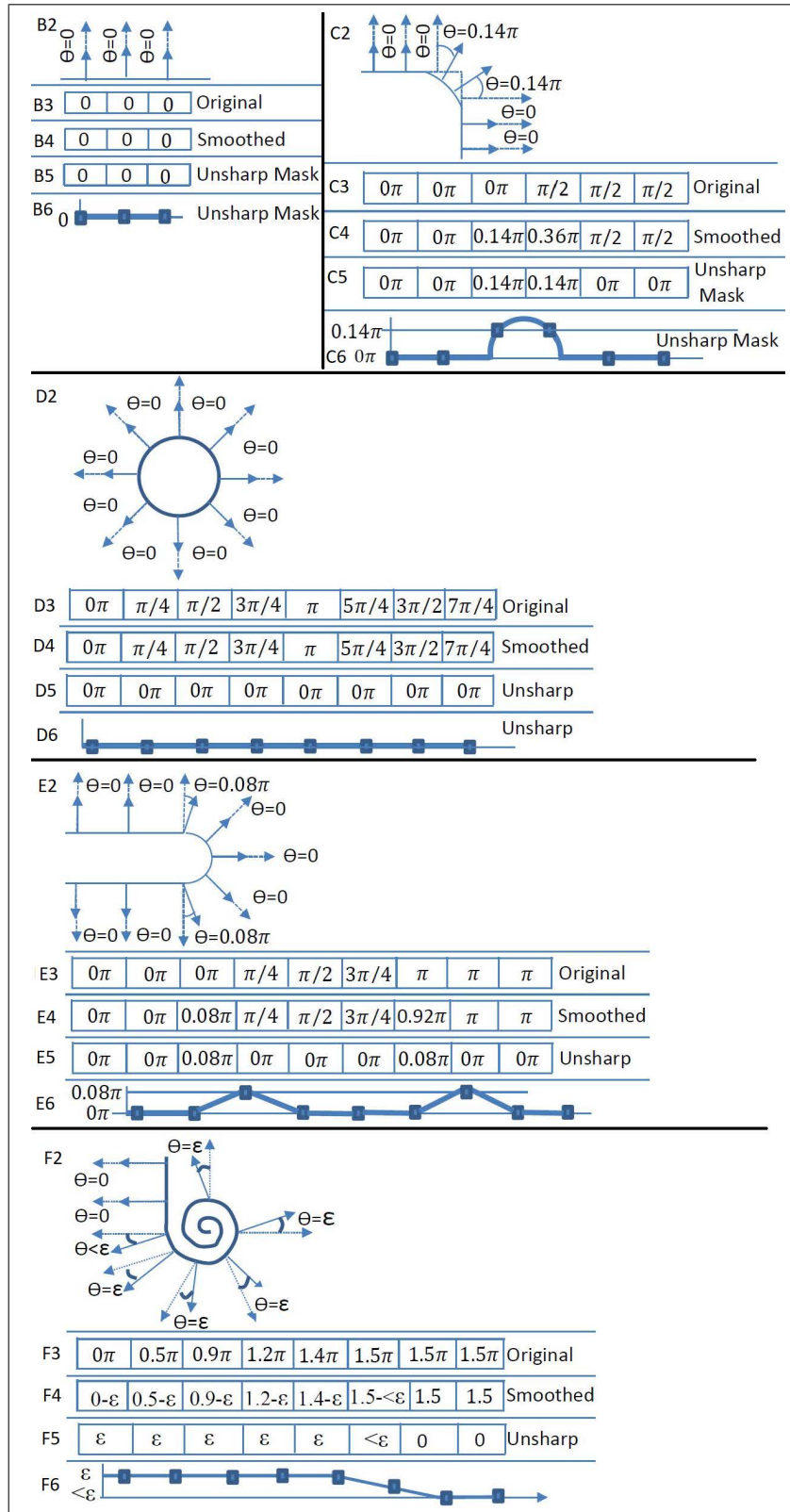


Figure 5.3: Illustrates the behavioral analyses of the Unsharp Masking applied on the areas of examples B,C,D,E and F.

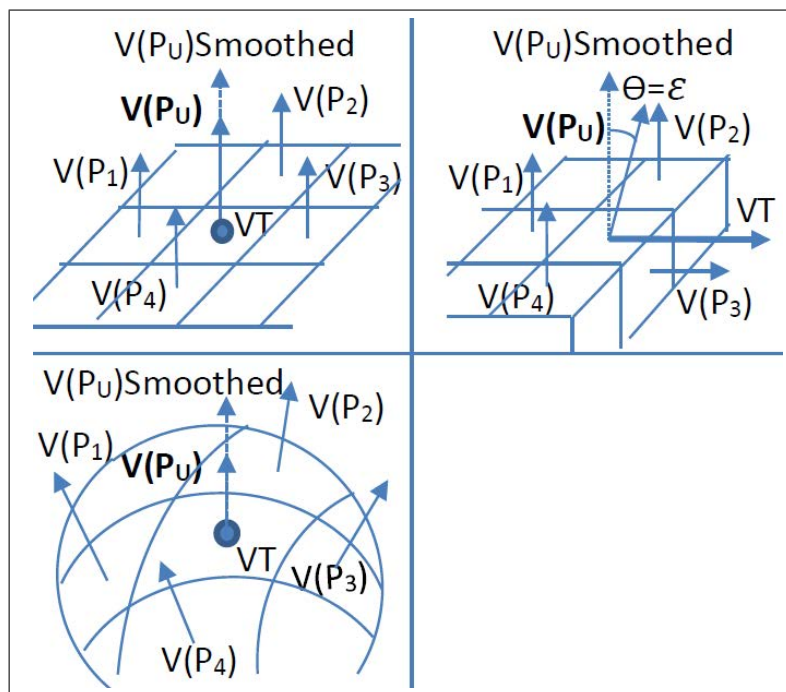


Figure 5.4: Illustrates the Unsharp Masking edge directions of the Plane area (top-left), Step edge (top-right), and Ramp area (bottom-left).

<b>RGB</b>			<b>Unsharp Ground Truth</b>		
<b>Depth</b>			<b>Unsharp</b>		
			<b>Unsharp Direction</b>		

Figure 5.5: This figure shows the edge detection results of the proposed Unsharp magnitude and direction operators on the TUM data set. Black, red, green, light blue, and dark blue pixels represent the true positives, false positives caused by algorithm, false negatives caused by algorithm, false positive caused by noise, and false negative caused by of noise, respectively.

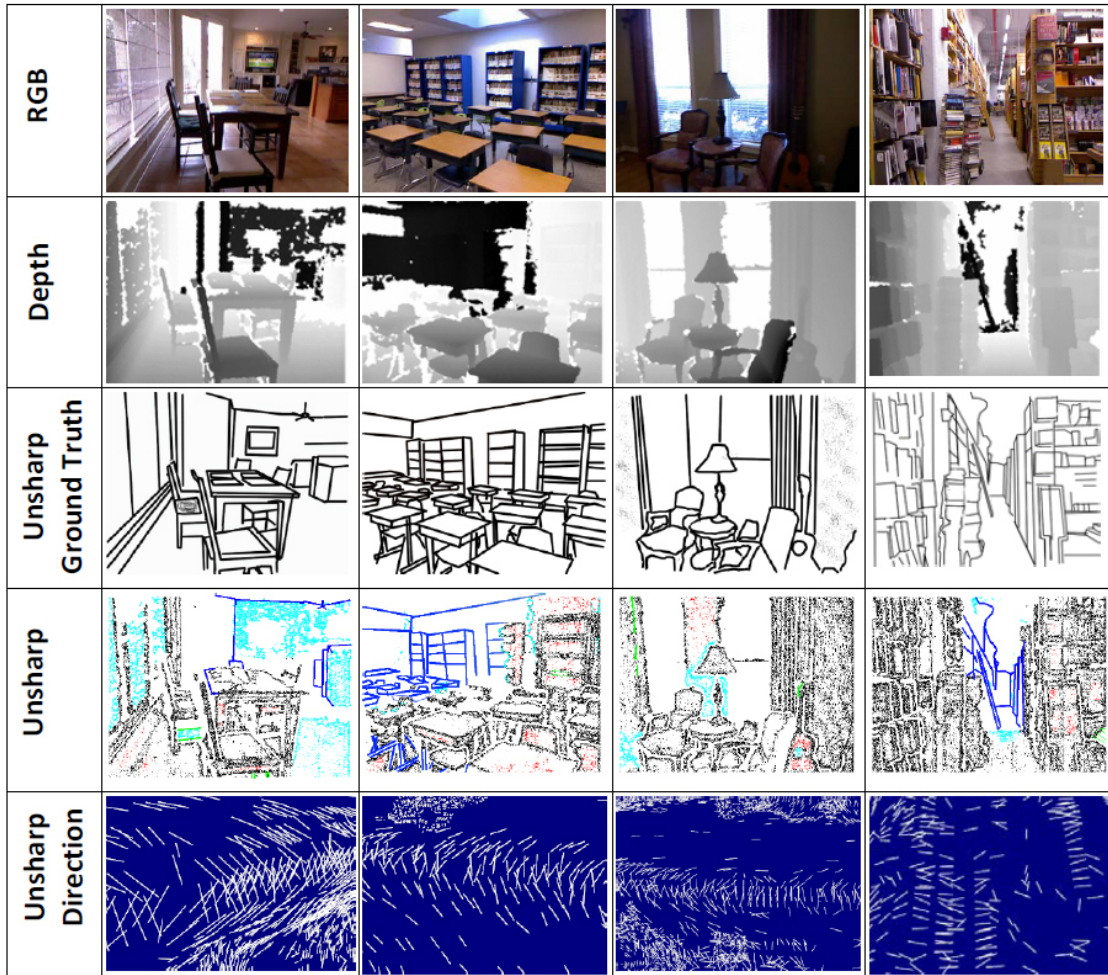


Figure 5.6: This figure shows the edge detection results of the proposed Unsharp Masking magnitude and direction operators on the NYU Depth data set. Black, red, green, light blue, and dark blue pixels represent the true positives, false positives caused by algorithm, false negatives caused by algorithm, false positive caused by noise, and false negative caused by of noise, respectively.



## **Chapter 6**

### **The Conclusions and Future Work**

## Contents

---

6.1	The Conclusions	99
6.2	The future work	101

---

## 6.1 The Conclusions

In the current dissertation, three researches proposals are proposed in three chapters as follows. The **first research** is the Solid Vector operations research to achieve the objective of extending the Solid Vector Subtraction operation by proposing definitions of its coordinate system, its hyper dimensional space, and its other whole complementary set of Solid Vector Addition, Multiplication, Division, Dot Product, and Cross Product operations. That coordinate system and the mathematical operations handle the vector as a one solid quantity unit, in addition to its property of being applicable to the 2d, 3d and hyper dimensional space. Achieving the objective of this research is the answer of the questions that were surfaced from the recent proposal of the Sold Vector Subtraction operation about its coordinate system, the definition of its hyper dimensional space, and the definitions of its other whole complementary set of Solid Vector Addition, Multiplication, Division, Dot Product, and Cross Product operations. The methodologies for achieving the objective of this research are done by: 1. a) making one of the operand vectors of the Solid Vector Subtraction operation as the first axis of the coordinate system that is the Directional Zero vector that replaces the *PoleY* of the Fixed Polar, and that is the lowest starting point of the angular component  $\theta$ ; and b) making the Directional Sign Axis of the Solid Vector Subtraction operation as the second axis of the coordinate system, that replaces the  $x$  axis of the Fixed Polar; in order to get a rotated version of the Fixed Polar that, in the current research, refereed to it as the Rotated Polar coordinate system that is implemented in the 3-D dimensional space and that is not fixed to the still plane of the  $x, y$  axes, 2. specifying a pair of vectors as components of a Hyper Rotated Polar when the Solid Vector Subtraction operations between them and their Directional Zero vector produce the same Directional Sign Axis (including both of its opposite directions the positive and negative), in order to define its extension to the hyper dimensional space, and 3. using the triangulation in order to propose the Solid Vector Addition, Multiplication, Division, Dot Product, and Cross Product operations. The **contributions** of the first research are proposing: 1. Novel definition of a Rotated Polar coordinate system based on two novel axes; the first axis is the dynamic Operation-Based Zero Vector and the second axis is the Directional Sign Axis of the Solid Vector Subtraction op-

eration, 2. Novel definition of the 2-d, 3-d, hyper dimensional Rotated Polar coordinate system space,, 3. Novel definition of the Solid Vector Addition, 4. Multiplication, 5. Division, 6. Dot Product and 7. Cross Product operations.

The **second research** is the second-order derivative, the Gradient-based Laplacian of a field of vectors, 3d geometrical edge detector high pass spatial filter is proposed in an analogous manner to the definition of the Gradient-Based Laplacian of a field of scalars. This proposed Gradient of a field of vectors classifies two types of segments of the geometrical surfaces of the point cloud: the Plane and Ramp areas; and the Step edge areas. And to do that design and implementation, the recently proposed Gradient of a field of vectors high pass spatial filter is used to define the Gradient-Based Laplacian, the 2nd derivative of a field of vectors edges detector. The **contributions** of the second research are: 8. Proposing a novel algorithm of the Gradient-Based Laplacian of a field of vectors (used for detecting 3-D surface geometrical edges) that is based on the novel algorithm of the Gradient of a field of vectors, that was proposed in [3] research. In the Gradient-Based Laplacian algorithm, the first predefined operation of the convolution is defined as the absolute scalar difference of the Gradient values of the neighbor pixels from the Gradient value of the center pixel, in contrast to the existing work [10] of gray-scale edge detection, in which this is defined as the signed Gradient difference operation, 9. Doing behavioral analyses for the Gradient-based Laplacian on Step edge, Plane, and Ramp areas for two cases; when the absolute difference and when the signed difference are used in order to prove that the former is the right one and the latter is the wrong, and 10. Doing performance analyses for the Gradient-based Laplacian on TUM data set, and comparison study with the state of the art edge detectors on NYUD data set.

The **third research** is the Unsharp Masking of a field of vectors which proposes a 3d Unsharp Masking geometrical edge detector high pass spatial filter in an analogous way to the Unsharp Masking of a field of scalars for detecting geometrical edge magnitude and direction of the 3-d point clouds. The Cartesian Components-Wise mathematical subtraction operation used in the Unsharp Masking of a field of scalars is not useful to develop the Unsharp Masking of a field of vectors, so the mathematical Solid Vector Subtraction operation is used to propose the current

Unsharp Masking of a field of vectors. The **contributions** of the third research are: 11. Proposing a novel algorithm for the Unsharp Masking of a field of vectors that is based on the Solid Vector subtraction operation, and that is analogous to the Unsharp Masking of a field of scalars, 12. Doing behavioral analyses for the Unsharp Masking on the Step edges; Plane areas; and onset, end and along Ramp areas, and 13. Doing performance analyses for the Unsharp Masking on TUM data set, and comparison study with the state of the art edge detectors on NYUD data set.

## 6.2 The future work

The above mentioned Solid Vector mathematical operations are applicable to 2d, 3d and hyper-dimensional space; and they handle the vector as only two layers: one directional angular component, and one magnitude component; which make them Solid Vector compliant and act as a foundation for the future work. So the future work for the proposed three proposals the Solid Vector operations, the Gradient-based Laplacian and the Unsharp Masking of a field of vectors can be building 2d, 3d and hyper-dimensional artificial intelligence and machine learning applications—such as 3d object recognition and detection, 3d SLAM, 3d rotation control of virtual and physical objects, 3d virtual sculpturing and segmentation of the Ramp areas only.

# Bibliography

- [1] I. Naser, J. Al-Anssari, and A. Ralescu, “Three-dimensional gradient-based laplacian spatial filter of a field of vectors for geometrical edges magnitude detection in point cloud surfaces,” *2019 Joint 8th International Conference on Informatics, Electronics & Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, pp. 354–361, 2019. [10](#), [30](#), [78](#), [80](#), [90](#)
- [2] —, “Three-dimensional unsharp masking spatial filter of a field of vectors for geometrical edges magnitude and direction detection in point cloud surfaces,” *2019 IEEE International Conference on Humanized Computing and Communication (HCC)*, pp. 68–76, 2019. [10](#)
- [3] J. Al-Anssari, I. Naser, and A. Ralescu, “Three-dimensional gradient spatial filter of a field of vectors for geometrical edges magnitude detection in point cloud surfaces,” *2019 Joint 8th International Conference on Informatics, Electronics & Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, pp. 362–370, 2019. [xi](#), [xii](#), [3](#), [4](#), [5](#), [9](#), [10](#), [19](#), [20](#), [24](#), [29](#), [30](#), [38](#), [39](#), [43](#), [47](#), [55](#), [56](#), [57](#), [58](#), [65](#), [78](#), [80](#), [81](#), [82](#), [100](#)
- [4] —, “Three-dimensional laplacian spatial filter of a field of vectors for geometrical edges magnitude and direction detection in point cloud surfaces,” *2019 IEEE International Conference on Humanized Computing and Communication (HCC)*, pp. 83–93, 2019. [5](#), [11](#)
- [5] A. Kadambi, A. Bhandari, and R. Raskar, *3D Depth Cameras in Vision: Benefits and Limitations of the Hardware*. Cham: Springer International Publishing, 2014, pp. 3–26.

- [Online]. Available: [https://doi.org/10.1007/978-3-319-08651-4\\_1](https://doi.org/10.1007/978-3-319-08651-4_1) xi, 14, 15, 16
- [6] A. Polesel, G. Ramponi, and V. J. Mathews, “Image enhancement via adaptive unsharp masking,” *IEEE Transactions on Image Processing*, vol. 9, no. 3, pp. 505–510, March 2000. xi, 20, 21
- [7] R. C. Gonzalez and R. E. Woods, *Digital Image Processing (3rd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006. xi, 4, 19, 20, 22, 23, 24, 57, 59, 66, 77, 79
- [8] R. Jain, R. Kasturi, and B. G. Schunck, *Machine Vision*. USA: McGraw-Hill, Inc., 1995. xi, 24, 25
- [9] A. Lejeune, J. G. Verly, and M. V. Droogenbroeck, “Probabilistic framework for the characterization of surfaces and edges in range images, with application to edge detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2018. xiv, 68, 71, 72, 74, 90, 92, 93
- [10] “Open CV laplacian operator,” [https://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/laplace\\_operator/laplace\\_operator.html](https://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/laplace_operator/laplace_operator.html), accessed: 2018-09-20. 9, 56, 57, 74, 100
- [11] T. Kanade, A. Yoshida, K. Oda, H. Kano, and M. Tanaka, “A stereo machine for video-rate dense depth mapping and its new applications,” in *Proceedings CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 1996, pp. 196–202. 14
- [12] F. Blais, “Review of 20 years of range sensor development,” *Journal of Electronic Imaging*, vol. 13, no. 1, pp. 231 – 243, 2004. [Online]. Available: <https://doi.org/10.1117/1.1631921> 14
- [13] R. Lange and P. Seitz, “Solid-state time-of-flight range camera,” *IEEE Journal of Quantum Electronics*, vol. 37, no. 3, pp. 390–397, March 2001. 14

- [14] B. FREEDMAN, A. SHPUNT, Y. ARIELI, and M. MACHLINE, “Depth mapping using projected patterns,” 2008. 14
- [15] “Kinect,” accessed in January 2020, wikipedia page. [Online]. Available: <https://en.wikipedia.org/wiki/Kinect> 14
- [16] R. A. Zeineldin and N. A. El-Fishawy, “Fast and accurate ground plane detection for the visually impaired from 3d organized point clouds,” in *2016 SAI Computing Conference (SAI)*, July 2016, pp. 373–379. 14
- [17] “Getting started / basic structures,” accessed in January 2020, point Cloud Documentation. [Online]. Available: [http://pointclouds.org/documentation/tutorials/basic\\_structures.php](http://pointclouds.org/documentation/tutorials/basic_structures.php) 16
- [18] M. Berger, A. Tagliasacchi, L. M. Seversky, P. Alliez, G. Guennebaud, J. A. Levine, A. Sharf, and C. T. Silva, “A survey of surface reconstruction from point clouds,” *Comput. Graph. Forum*, vol. 36, no. 1, p. 301329, Jan. 2017. [Online]. Available: <https://doi.org/10.1111/cgf.12802> 16
- [19] G.-T. Michailidis, G.-T. Michailidis, R. Pajarola, and R. Pajarola, “Bayesian graph-cut optimization for wall surfaces reconstruction in indoor environments,” *The Visual Computer*, vol. 33, no. 10, pp. 1347–1355, 2017. 16
- [20] S. Orts-Escolano, J. Garcia-Rodriguez, M. Cazorla, V. Morell, J. Azorin, M. Saval, A. Garcia-Garcia, and V. Villena, “Bioinspired point cloud representation: 3d object tracking,” *Neural Comput. Appl.*, vol. 29, no. 9, p. 663672, May 2018. [Online]. Available: <https://doi.org/10.1007/s00521-016-2585-0> 16
- [21] A. Sitek, R. H. Huesman, and G. T. Gullberg, “Tomographic reconstruction using an adaptive tetrahedral mesh defined by a point cloud,” *IEEE Transactions on Medical Imaging*, vol. 25, pp. 1172–1179, 2006. 16



- [22] H.-R. Wang, J. Lei, A. Li, and Y.-H. Wu, "A geometry-based point cloud reduction method for mobile augmented reality system," *Journal of Computer Science and Technology*, vol. 33, no. 6, pp. 1164–1177, Nov 2018. [Online]. Available: <https://doi.org/10.1007/s11390-018-1879-3> 16
- [23] S. Holzer, R. B. Rusu, M. Dixon, S. Gedikli, and N. Navab, "Adaptive neighborhood selection for real-time surface normal estimation from organized point cloud data using integral images," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2012, pp. 2684–2689. 16, 17
- [24] J. Ma, H.-Y. Feng, and L. Wang, "Normal vector estimation for point clouds via local delaunay triangle mesh matching," *Computer-Aided Design and Applications*, vol. 10, no. 3, pp. 399–411, 2013. [Online]. Available: <https://www.tandfonline.com/doi/abs/10.3722/cadaps.2013.399-411> 17
- [25] X. Zhan, Y. Cai, H. Li, Y. Li, and P. He, "A point cloud registration algorithm based on normal vector and particle swarm optimization," *Measurement and Control*, p. 2029401985821, 2019. 17
- [26] W. Sun, J. Wang, F. Jin, Z. Liang, and Y. Yang, "Datum Feature Extraction and Deformation Analysis Method Based on Normal Vector of Point Cloud," *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 42.3, pp. 1601–1606, Apr 2018. 17
- [27] K. Klasing, D. Althoff, D. Wollherr, and M. Buss, "Comparison of surface normal estimation methods for range sensing applications," in *2009 IEEE International Conference on Robotics and Automation*, May 2009, pp. 3206–3211. 17
- [28] X. Meng, W. He, and J. Liu, "An investigation of the high efficiency estimation approach of the large-scale scattered point cloud normal vector," *Applied Sciences*, vol. 8, no. 3, 2018. [Online]. Available: <https://www.mdpi.com/2076-3417/8/3/454> 17

- [29] D. OuYang and H.-Y. Feng, “On the normal vector estimation for point cloud data from smooth surfaces,” *Computer-Aided Design*, vol. 37, no. 10, pp. 1071 – 1079, 2005. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S001044850400226X> 17
- [30] D. Holz, S. Holzer, R. B. Rusu, and S. Behnke, “Real-time plane segmentation using rgb-d cameras,” in *RoboCup 2011: Robot Soccer World Cup XV*, T. Röfer, N. M. Mayer, J. Savage, and U. Saranlı, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 306–317. 17
- [31] J. L. Bentley, “Multidimensional binary search trees used for associative searching,” *Commun. ACM*, vol. 18, no. 9, p. 509517, Sep. 1975. [Online]. Available: <https://doi.org/10.1145/361002.361007> 18
- [32] P. Hansen and N. Mladenović, *Variable Neighborhood Search*. Cham: Springer International Publishing, 2018, pp. 759–787. [Online]. Available: [https://doi.org/10.1007/978-3-319-07124-4\\_19](https://doi.org/10.1007/978-3-319-07124-4_19) 18
- [33] M. Khelifa and D. Boughaci, “A variable neighborhood search method for solving the traveling tournaments problem,” *Electronic Notes in Discrete Mathematics*, vol. 47, pp. 157 – 164, 2015, the 3rd International Conference on Variable Neighborhood Search (VNS’14). [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1571065314000638> 18
- [34] A. Divsalar, P. Vansteenwegen, and D. Cattrysse, “A variable neighborhood search method for the orienteering problem with hotel selection,” *International Journal of Production Economics*, vol. 145, no. 1, pp. 150 – 160, 2013. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0925527313000285> 18
- [35] B.-I. Kim, H. Li, and A. L. Johnson, “An augmented large neighborhood search method for solving the team orienteering problem,” *Expert Systems with Applications*, vol. 40, no. 8,

- pp. 3065 – 3072, 2013. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0957417412012638> 18
- [36] K. Svanberg, M. Werme, S. fr teknikvetenskap (SCI), M. (Inst.), O. och systemteori, and KTH, “A hierarchical neighbourhood search method for topology optimization,” *Structural and Multidisciplinary Optimization*, vol. 29, no. 5, pp. 325–340, 2005. 18
- [37] D. Ziou and S. Tabbone, “Edge detection techniques - an overview,” *INTERNATIONAL JOURNAL OF PATTERN RECOGNITION AND IMAGE ANALYSIS*, vol. 8, pp. 537–559, 1998. 19
- [38] S. E. Umbaugh, J. Snyder, and E. A. Fedorovskaya, “Digital image processing and analysis: Human and computer vision applications with cviptools, second edition,” *J. Electronic Imaging*, vol. 20, p. 039901, 2011. 19
- [39] M. J. Milroy, C. Bradley, and G. W. Vickers, “Segmentation of a wrap-around model using an active contour,” *Computer-Aided Design*, vol. 29, no. 4, pp. 299–320, 1997. 20
- [40] M. Yang and E. Lee, “Segmentation of measured point data using a parametric quadric surface approximation,” *Computer-aided design*, vol. 31, no. 7, pp. 449–457, 1999. 20
- [41] W. Fulton, “A few scanning tips, sharpening - unsharp mask,” accessed in January 2020, 1997-2010. Web site. [Online]. Available: <https://www.scantips.com/simple6.html> 20
- [42] G. Deng, “A generalized unsharp masking algorithm,” *IEEE Transactions on Image Processing*, vol. 20, no. 5, pp. 1249–1261, May 2011. 21
- [43] G. Cao, Y. Zhao, R. Ni, and A. C. Kot, “Unsharp masking sharpening detection via overshoot artifacts analysis,” *IEEE Signal Processing Letters*, vol. 18, no. 10, pp. 603–606, Oct 2011. 21
- [44] W. Ye and K. Ma, “Blurriness-guided unsharp masking,” *IEEE Transactions on Image Processing*, vol. 27, no. 9, pp. 4465–4477, Sep. 2018. 21

- [45] A. Rosenfeld and A. C. Kak, *Digital Picture Processing*, 2nd ed. New York: Academic Press, 1982. 22
- [46] “Concept of edge detection,” accessed in January 2020, tutorials Point. Web site. [Online]. Available: [https://www.tutorialspoint.com/dip/concept\\_of\\_edge\\_detection.htm](https://www.tutorialspoint.com/dip/concept_of_edge_detection.htm) 24
- [47] G. Collins, *The Foundations of Celestial Mechanics (Astronomy and Astrophysics Series)*. Pachart Pub House, 1989. 32
- [48] B. Adamczyk, *Foundations of Electromagnetic Compatibility: With Practical Applications*, 1st ed. Wiley Publishing, 2017. 32, 34, 36, 37, 38
- [49] M. Hosseini, H. Hassanabadi, and S. Hassanabadi, “Solutions of the dirac-weyl equation in graphene under magnetic fields in the cartesian coordinate system,” *The European Physical Journal Plus*, vol. 134, no. 1, pp. 1–6, 2019. 32
- [50] G. V. Garca and t. lu, “Parameterization of the ellipse based on the valencia’s sphere, without to use a cartesian coordinate system,” *Annals of the Faculty of Engineering Hunedoara*, vol. 16, no. 4, pp. 59–67, 2018. 32
- [51] Y. A. Grigoriev and A. V. Tsiganov, “On superintegrable systems separable in Cartesian coordinates,” *Physics Letters A*, vol. 382, no. 32, pp. 2092–2096, Aug 2018. 32
- [52] B. . Kim, J. . Sun, S. . Kim, M. . Kang, and S. . Ko, “Cnn-based ugs method using cartesian-to-polar coordinate transformation,” *Electronics Letters*, vol. 54, no. 23, pp. 1321–1322, 2018. 32
- [53] Y. Luo, F. Zhao, N. Li, and H. Zhang, “A modified cartesian factorized back-projection algorithm for highly squint spotlight synthetic aperture radar imaging,” *IEEE Geoscience and Remote Sensing Letters*, vol. 16, no. 6, pp. 902–906, June 2019. 32
- [54] R. Larson, *Elementary Linear Algebra*. Cengage Learning, 2012. [Online]. Available: <https://books.google.com/books?id=rqWCVMYk5mEC> 33

- [55] G. Hay, *Vector and tensor analysis*, ser. Dover Books on Science. Dover Publication, 1953.  
[Online]. Available: <https://books.google.com/books?id=ar3wswEACAAJ> 33
- [56] H. D. Tagare, “Polar coordinates: What they are and how to use them,” accessed in 2019, notes. [Online]. Available: <http://noodle.med.yale.edu/hdtag/notes/coord.pdf> 35
- [57] S. Gai, F. Da, and X. Fang, “A novel camera calibration method based on polar coordinate,” *PloS one*, vol. 11, no. 10, pp. e0165487–e0165487, 2016. 35
- [58] C. Han, H. Chen, G. Alonso, Y. Rao, J. Cubas, J. Yin, and X. Wang, “A linear model for relative motion in an elliptical orbit based on a spherical coordinate system,” *Acta Astronautica*, vol. 157, pp. 465 – 476, 2019. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S009457651830910X> 36
- [59] J. Wang, H. Wanga, J. Li, X. Luo, Y. Shi, and S. K. Jha, “Detecting double jpeg compressed color images with the same quantization matrix in spherical coordinates,” *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1, 2019. 36
- [60] “Hyperplane,” accessed in July 2019, wikipedia page. [Online]. Available: <https://en.wikipedia.org/wiki/Hyperplane> 39
- [61] S. Abbott, “Geometry, by v. v. prasolov and v. m. tikhomirov (trans. o. v. sipacheva). translations of mathematical monographs 200. pp. 257. 69.50. 2001. isbn 0 8218 2038 9 (american mathematical society).” *The Mathematical Gazette*, vol. 86, no. 507, p. 563564, 2002. 39
- [62] C. Luo, I. Safa, and Y. Wang, “Approximating Gradients for Meshes and Point Clouds via Diffusion Metric,” *Computer Graphics Forum*, 2009. 57, 79
- [63] M. Belkin, J. Sun, and Y. Wang, “Constructing laplace operator from point clouds in  $\mathbb{R}^d$ ,” in *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '09. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2009, pp. 1031–1040, <http://dl.acm.org/citation.cfm?id=1496770.1496882>. 57, 79

- [64] ———, “Discrete laplace operator on meshed surfaces,” in *Proceedings of the Twenty-fourth Annual Symposium on Computational Geometry*, ser. SCG ’08. New York, NY, USA: ACM, 2008, pp. 278–287. [Online]. Available: <http://doi.acm.org/10.1145/1377676.1377725> 57, 79
- [65] S. A. Coleman, B. W. Scotney, and S. Suganthan, “Edge detecting for range data using laplacian operators,” *IEEE Transactions on Image Processing*, vol. 19, no. 11, pp. 2814–2824, Nov 2010. 57, 79
- [66] B. Wang and A. Goldenberg, “Gradient-based laplacian feature selection,” *CoRR*, vol. abs/1404.2948, 2014. 57, 79
- [67] M. Belkin and P. Niyogi, “Laplacian eigenmaps for dimensionality reduction and data representation,” *Neural Comput.*, vol. 15, no. 6, pp. 1373–1396, Jun. 2003. [Online]. Available: <http://dx.doi.org/10.1162/089976603321780317> 57, 79
- [68] J. Cao, A. Tagliasacchi, M. Olson, H. Zhang, and Z. Su, “Point cloud skeletons via laplacian based contraction,” in *2010 Shape Modeling International Conference*, June 2010, pp. 187–197. 57, 79
- [69] K. Sugihara, “Range-data analysis guided by a junction dictionary,” *Artificial Intelligence*, vol. 12, no. 1, pp. 41 – 69, 1979. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/0004370279900043> 57, 79
- [70] B. Bhanu, Lee, Sungkee, Ho, Chih-Cheng, and T. Henderson, “Range data processing: Representation of surfaces by edges,” 1986 IEEE. 57, 79
- [71] S. A. Coleman, S. Suganthan, and B. W. Scotney, “Gradient operators for feature extraction and characterisation in range images,” *Pattern Recognition Letters*, vol. 31, no. 9, pp. 1028 – 1040, 2010. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S016786550900364X> 57, 79

- [72] T. M. Khan, D. Bailey, M. A. Khan, and Y. Kong, “Real-time edge detection and range finding using fpgas,” *Optik - International Journal for Light and Electron Optics*, vol. 126, no. 17, pp. 1545 – 1550, 2015. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S003040261500025X> 57, 79
- [73] T. Bao, J. Zhao, and M. Xu, “Step edge detection method for 3d point clouds based on 2d range images,” *Optik - International Journal for Light and Electron Optics*, vol. 126, no. 20, pp. 2706 – 2710, 2015. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0030402615005586> 57, 79
- [74] T. Rabbani, G. Vosselman, and et al., “Segmentation of point clouds using smoothness constraint,” 2006. 57, 79
- [75] E. Natonek, “Fast range image segmentation for servicing robots,” vol. 1, 1998, pp. 406–411 vol.1. 57, 79
- [76] I. Khalifa, M. Moussa, and M. Kamel, “Range image segmentation using local approximation of scan lines with application to cad model acquisition,” *Machine Vision and Applications*, vol. 13, no. 5, pp. 263–274, 2003. 57, 79
- [77] G. Sithole and G. Vosselman, “Automatic structure detection in a point-cloud of an urban landscape,” 2003, pp. 67–71. 57, 79
- [78] A. D. Sappa and M. Devy, “Fast range image segmentation by an edge detection strategy.” IEEE, 2001, pp. 292–299. 57, 79
- [79] V. Kovacs and G. Tevesz, “Edge detection in discretized range images.” IEEE, 2014, pp. 203–208. 57, 79
- [80] D. A. Hafiz, W. M. Sheta, S. Bayoumi, and B. A. B. Youssef, “A new approach for 3d range image segmentation using gradient method,” *Journal of Computer Science*, vol. 7, no. 4, pp. 475–487, 2011. 57, 79

- [81] S. W. Zucker and R. A. Hummel, "A three-dimensional edge operator," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-3, no. 3, pp. 324–331, 1981. 57, 79
- [82] L.-A. Albrecht, "Curvature based analysis of point clouds," 2014, diplom, [Online; accessed 26-September-2017]. 57, 79
- [83] T. W. Lim, P. F. Ramos, and M. C. ODowd, "Edge detection using point cloud data for noncooperative pose estimation," *Journal of Spacecraft and Rockets*, vol. 54, no. 2, pp. 500–505, 2017;2016;. 57, 79
- [84] H. Ni, X. G. Lin, and J. X. Zhang, "Applications of 3d-edge detection for als point cloud," *The International Archives of the Photogrammetry*, pp. 277–283, 2017. 57, 79
- [85] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012. 70, 91
- [86] P. K. Nathan Silberman, Derek Hoiem and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *ECCV*, 2012. 71, 92
- [87] X. Jiang and H. Bunke, "Edge detection in range images based on scan line approximation," *Computer Vision and Image Understanding*, vol. 73, no. 2, pp. 183 – 199, 1999. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1077314298907159> 71, 74, 92, 93
- [88] A. Lejeune, S. Pirard, M. V. Droogenbroeck, and J. Verly, "A new jump edge detection method for 3d cameras," in *2011 International Conference on 3D Imaging (IC3D)*, Dec 2011, pp. 1–7. 71, 72, 74, 92, 93
- [89] P. Dollr and C. L. Zitnick, "Fast edge detection using structured forests," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 8, pp. 1558–1570, Aug 2015. 71, 74, 92, 93



- [90] S. Gupta, R. Girshick, P. Arbellez, and J. Malik, “Learning rich features from rgb-d images for object detection and segmentation,” *Eur. Conf. Computer Vision (ECCV)*, Sept. 2014, 2014. 71, 72, 74, 92, 93
- [91] S. Xie and Z. Tu, “Holistically-nested edge detection,” *International Journal of Computer Vision*, vol. 125, no. 1, pp. 3–18, Dec 2017. [Online]. Available: <https://doi.org/10.1007/s11263-017-1004-z> 71, 72, 74, 92, 93
- [92] S. D. Zeno, “A note on the gradient of a multi-image,,” *Computer Vision, Graphics, and Image Processing*, vol. 33, pp. 116–125, 1986. 79
- [93] J. Coffin, *Vector Analysis: An Introduction to Vector-Methods and Their Various Applications to Physics and Mathematics*. CreateSpace Independent Publishing Platform, 2015, <https://books.google.com/books?id=pmkvjgEACAAJ>. 79