In compliance with the
Canadian Privacy Legislation
some supporting forms
may have been removed from
this dissertation.

While these forms may be included
in the document page count,
their removal does not represent
any loss of content from the dissertation.

DEVELOPMENT OF A REAL-TIME EDGE DETECTION METHOD FOR A LASER RANGE SCANNER SYSTEM

by

DAVID K. MACKINNON, B.SC.(MATH), B.SC.(EE)

A thesis submitted to
Faculty of Graduate Studies and Research
in partial fulfillment of the requirements
of the degree of
MASTER OF APPLIED SCIENCE IN ELECTRICAL ENGINEERING

Ottawa-Carleton Institute of Electrical and Computer Engineering

Faculty of Engineering

Department of Systems and Computer Engineering

Carleton University

Ottawa, Ontario

August 2003

©Copyright

2003, DAVID K. MACKINNON

National Library
of Canada

Bibliothèque nationale
du Canada

Acquisitions and
Bibliographic Services

Acquisitions et
services bibliographiques

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

ISBN: 0-612-88888-6

Our file Notre référence

ISBN: 0-612-88888-6

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

Canada

Abstract

A method of detecting edges using laser range data obtained from closed loop line scans following Lissajous scanning patterns was developed. Raw data from a peak detector and dual galvanometers were used to minimize processing time associated with translation between polar-like UVP and Cartesian coordinate systems. A non-linear model of an auto-synchronizing laser range scanner was developed to perform off-line development of the edge detection algorithms. A median filter was selected as the filter best able to approximate the original signal based on expected noise levels. A product-of-difference and a first derivative edge enhancement method were selected for comparison. Peaks associated with step edges were detected by selecting those with peak base to peak height ratios less than a threshold value. The product-of-difference algorithm was found to perform better than the first derivative algorithm on the simulated system and performed well on a real laser range scanner system.

Acknowledgements

I would like to thank Francois Blais, Research Officer at the National Research Council of Canada, and Dr. Victor Aitken who provided invaluable direction and support during this research. I would also like to thank the Visual Information Technology group of the Institute for Information Technology at the National Research Council of Canada for allowing me to have access to equipment, facilities and resources. I would like to thank the Department of Systems and Computer Engineering for providing facilities and resources during the preparation of this document.

Much of this work would not have been possible without the financial support of the Natural Sciences and Engineering Research Council of Canada. I would also like to thank Neptec Design Group who provided data and feedback for portions of this study.

Finally I would especially like to thank my wife, Coralie, who supported me through this venture.

Contents

Abstract	iii
Acknowledgements	iv
Contents	v
List of Figures	xi
List of Tables	xvii
List of Symbols	xix
1 Introduction	1
1.1 Thesis Objectives and Overview	5
1.2 Boundary Detection	6
1.2.1 Edge Detection	7
1.2.2 Edge Enhancement Methods	9
1.2.3 Noise Filters	12
1.2.4 Edge Localization	15
1.3 Summary	16

2	Random Access Scanner	19
2.1	Overview of Random Access Scanner	20
2.2	Overview of Lissajous Patterns	22
2.3	Application to the Random Access Scanner	25
2.4	Normalized Angular Deflection	27
2.5	Summary	31
3	Random Access Scanner Modelling	33
3.1	Mean Optical Path (MOP)	34
3.1.1	Determination of MOP	35
3.1.2	Calculating Laser Path Deviations	39
3.2	Imaging Axis	42
3.3	Image Detection	45
3.3.1	CCD Imaging	50
3.4	Quantization	54
3.5	Environmental Modelling	56
3.5.1	RAS Response	58
3.6	Scanning Methods	60
3.6.1	Lissajous Scan	60
3.6.2	Raster Scan	64
3.7	Model Calibration	66
3.7.1	Mirror Calibration	67
3.7.2	Peak Detector Calibration	70
3.7.3	Linear Fit	72
3.8	Model Demonstration	73

3.9	Summary	85
4	Edge Mapping	86
4.1	Proposed Edge Detection Method	87
4.1.1	Evaluation Methodology	87
4.2	Phase I: Edge Detector Development	88
4.2.1	Filter Evaluation	93
4.2.2	Edge Enhancement Evaluation	103
4.3	Phase II: Simple Surfaces	114
4.3.1	Edge Detector Evaluation	123
4.3.2	Interpretation of Results	126
4.4	Phase III: Speed Evaluation	126
4.5	Real-time Edge Detection	127
4.6	Summary	128
5	Conclusions and Future Work	131
	References	138
	Appendix A: Scanner Model Development	A-1
A.1	Calculating MOP	A-1
A.1.1	Point of incidence on the fixed output mirror	A-1
A.1.2	Point of incidence on the y-axis mirror	A-4
A.2	Imaging Axis	A-5
A.2.1	Point of incidence on the fixed input mirror	A-6
A.2.2	Point of incidence on the y-axis mirror	A-7

Appendix B: Environmental Model Development	B-1
B.1 Modelling the Environment	B-1
B.2 Laser Vector Representation	B-3
B.3 Planar Intersections	B-5
B.4 Worlds, Regions and Objects	B-6
Appendix C: Model Calibration	C-1
C.1 Model Calibration	C-1
C.2 Mirror Calibration	C-2
C.3 Peak Detector Calibration	C-4
C.4 Linear Fit	C-5
C.5 Calibration Process Results	C-12
Appendix D: Edge Metrics	D-1
D.1 Overview of Edge Metrics	D-1
D.2 Objective Performance Measures	D-2
D.3 Figure of Merit	D-4
D.4 Edge Orientation, Gradient and Angle	D-6
D.5 Signal to Noise ratio (SNR)	D-7
D.6 Subjective Performance Evaluation	D-8
D.7 Selection of Edge Detection Metrics	D-8
Appendix E: Noise Analysis	E-1
E.1 Noise Analysis	E-1
E.2 Noise Distribution	E-1
E.3 Observed Noise	E-7

E.4 Peak Additive Noise	E-16
E.5 Galvanometer Error	E-16
E.6 Peak Spike Rates	E-17
Appendix F: Edge Height at 1-metre	F-1
Appendix G: Edge Height at 2-metres	G-1
Appendix H: Edge Height at 5-metres	H-1
Appendix I: Edge Height at 10-metres	I-1
Appendix J: Edge Slope Effect	J-1
Appendix K: Edge Separation Effect	K-1
Appendix L: RAS Software	L-1
L.1 Overview	L-1
L.2 RAS	L-2
L.3 Calibration	L-3
L.4 Environment	L-5
L.5 Motion	L-6
L.6 Worlds	L-6
L.7 Objects	L-6
L.8 Lissajous	L-7
L.9 Raster	L-8
L.10 Edge	L-9
L.11 C	L-12

L.12 Tools	L-14
L.13 Demos	L-15

List of Figures

1.1	Examples of edge types	2
1.2	Other edge classifications	8
2.1	Raster-mode scanning	20
2.2	Lissajous-mode scanning	20
2.3	RAS Internal structure	21
2.4	Optical path	21
2.5	Target tracking error	22
2.6	Comparison of phase lags	24
2.7	Simplified RAS Model	28
2.8	Simplified laser path	28
3.1	Mean optical path	36
3.2	Intrinsic scanner parameters	37
3.3	Matlab simulation of the RAS	40
3.4	Laser path in (x,y)-plane	41
3.5	Laser path in (y,z)-plane	41
3.6	Matlab simulation of laser path	43
3.7	Imaging axis in (x,y)-plane	44

3.8	Matlab simulation of imaging axis	46
3.9	Point path in the (x,y)-plane	47
3.10	Simplified scanner model in (x,z)-plane	47
3.11	Point projection into CCD array	50
3.12	Matlab simulation of projected lens intersection	52
3.13	Matlab simulated peak values for a 5-m planar surface	53
3.14	Quantization function output	55
3.15	Matlab simulation of environment	59
3.16	Matlab simulation of an unwrapped Lissajous scan	61
3.17	Matlab simulation of a Lissajous scan	62
3.18	Matlab simulation of a Lissajous surface map	63
3.19	Matlab simulation of a raster surface map	65
3.20	Space 40 test configuration	73
3.21	Space 40 test environment	74
3.22	Space 40 Box 1 test	76
3.23	Space 40 Box 1 error	77
3.24	Space 40 Box 2 test	78
3.25	Space 40 Box 2 error	79
3.26	Space 40 Box 3 test	81
3.27	Space 40 Box 3 error	82
3.28	Space 40 Box 4 test	83
3.29	Space 40 Box 4 error	84
4.1	One source of non-zero spikes	92
4.2	One source of zero spikes	92

4.3	Averaging filter window size	95
4.4	Gaussian filter window size	96
4.5	Median filter window size	97
4.6	Processing times for median filter on Calib scanner	98
4.7	Peak values versus range for Space 40 scanner	100
4.8	Comparison of noise filters	102
4.9	A comparison of edge enhancement methods	104
4.10	Effect of increasing window size on Product of Difference	106
4.11	Effect of increasing range on 11-element Product of Difference	107
4.12	Edge heights versus range at $1/4$ -CCD pixel resolution	108
4.13	Selection of PoD threshold level	109
4.14	Selection of Product of Difference threshold level	110
4.15	Effect of increasing range on 3-point Derivative	111
4.16	Effect of increasing edge height on Product of Difference	112
4.17	Effect of increasing edge height on 3-point Derivative	113
4.18	Ridge surface in UVP space	115
4.19	Lissajous scan of a Ridge surface	117
4.20	Effect of range on Lissajous scan of $1/4$ -CCD pixel ridge	118
4.21	Effect of edge height on Product of Difference filtered Lissajous scan of ridge surface	119
4.22	Effect of edge height on derivative filtered Lissajous scan of ridge surface	120
4.23	Ramp surface in UVP space	125
4.24	RAS edge detection of a ridge object	129

A.1	Detailed MOP in (x,y)-plane	A-2
A.2	Detailed MOP in (y,z)-plane	A-2
A.3	Detailed Imaging Axis in (x,y)-plane	A-6
A.4	Detailed Imaging Axis in (y,z)-plane	A-6
C.1	Calib X-mirror position errors	C-16
C.2	Calib Y-mirror position errors	C-17
C.3	Calib peak errors	C-18
D.1	Edge resolution	D-7
E.1	Space 40 Peak normal probability plot	E-7
E.2	Calib Peak normal probability plot	E-8
E.3	Space 40 x-galvo normal probability plot	E-9
E.4	Calib x-galvo normal probability plot	E-10
E.5	Space 40 y-galvo normal probability plot	E-11
E.6	Calib y-galvo normal probability plot	E-12
E.7	Worst-case Calib noise and spike rates	E-14
E.8	Worst-case Space40 noise and spike rates	E-15
F.1	Number Detected for Method by Edge Height (m) at 1-metre . . .	F-2
F.2	Percent True for Method by Edge Height (m) at 1-metre	F-3
F.3	Percent False for Method by Edge Height (m) at 1-metre	F-4
F.4	Noise-to-Signal for Method by Edge Height (m) at 1-metre	F-5
F.5	Mean Width for Method by Edge Height (m) at 1-metre	F-6
F.6	Absolute Deviation for Method by Edge Height (m) at 1-metre . .	F-7
F.7	Pratt FoM for Method by Edge Height (m) at 1-metre	F-8

F.8	van der Heyden FoM for Method by Edge Height (m) at 1-metre .	F-9
G.1	Number Detected for Method by Edge Height (m) at 2-metres . .	G-2
G.2	Percent True for Method by Edge Height (m) at 2-metres	G-3
G.3	Percent False for Method by Edge Height (m) at 2-metres	G-4
G.4	Noise-to-Signal for Method by Edge Height (m) at 2-metres	G-5
G.5	Mean Width for Method by Edge Height (m) at 2-metres	G-6
G.6	Absolute Deviation for Method by Edge Height (m) at 2-metres .	G-7
G.7	Pratt FoM for Method by Edge Height (m) at 2-metres	G-8
G.8	van der Heyden FoM for Method by Edge Height (m) at 2-metres	G-9
H.1	Number Detected for Method by Edge Height (m) at 5-metres . .	H-2
H.2	Percent True for Method by Edge Height (m) at 5-metres	H-3
H.3	Percent False for Method by Edge Height (m) at 5-metres	H-4
H.4	Noise-to-Signal for Method by Edge Height (m) at 5-metres	H-5
H.5	Mean Width for Method by Edge Height (m) at 5-metres	H-6
H.6	Absolute Deviation for Method by Edge Height (m) at 5-metres .	H-7
H.7	Pratt FoM for Method by Edge Height (m) at 5-metres	H-8
H.8	van der Heyden FoM for Method by Edge Height (m) at 5-metres	H-9
I.1	Number Detected for Method by Edge Height (m) at 10-metres .	I-2
I.2	Percent True for Method by Edge Height (m) at 10-metres	I-3
I.3	Percent False for Method by Edge Height (m) at 10-metres	I-4
I.4	Noise-to-Signal for Method by Edge Height (m) at 10-metres	I-5
I.5	Mean Width for Method by Edge Height (m) at 10-metres	I-6
I.6	Absolute Deviation for Method by Edge Height (m) at 10-metres	I-7
I.7	Pratt FoM for Method by Edge Height (m) at 10-metres	I-8

I.8	van der Heyden FoM for Method by Edge Height (m) at 10-metres	I-9
J.1	Number Detected for Method by Edge Slope	J-2
J.2	Percent True for Method by Edge Slope	J-3
J.3	Percent False for Method by Edge Slope	J-4
J.4	Noise-to-Signal for Method by Edge Slope	J-5
J.5	Mean Width for Method by Edge Slope	J-6
J.6	Absolute Deviation for Method by Edge Slope	J-7
J.7	Pratt FoM for Method by Edge Slope	J-8
J.8	van der Heyden FoM for Method by Edge Slope	J-9
K.1	Number Detected for Method by Edge Separation	K-2
K.2	Percent True for Method by Edge Separation	K-3
K.3	Percent False for Method by Edge Separation	K-4
K.4	Noise-to-Signal for Method by Edge Separation	K-5
K.5	Mean Width for Method by Edge Separation	K-6
K.6	Absolute Deviation for Method by Edge Separation	K-7
K.7	Pratt FoM for Method by Edge Separation	K-8
K.8	van der Heyden FoM for Method by Edge Separation	K-9

List of Tables

2.1	Lissajous patterns	23
4.1	Observed sample standard deviations	90
4.2	Observed spike rate	90
4.3	Comparison of Noise filters	101
C.1	Space 40 Galvanometer extents	C-2
C.2	Calib Galvanometer extents	C-3
C.3	Galvanometer calibration results	C-3
C.4	Galvanometer position error	C-4
C.5	Peak Detector extents	C-4
C.6	CCD calibration results	C-5
C.7	Space 40 regression analysis	C-6
C.8	Calib regression analysis	C-6
C.9	Space 40 model confirmation	C-10
C.10	Calib model confirmation	C-11
C.11	Fit improvements	C-12
D.1	Edge Metrics	D-9

D.2	Summary of Edge Metrics	D-12
E.1	Kurtosis values of scanner data	E-3
E.2	skewness of scanner data	E-5
E.3	Observed noise	E-13
E.4	Observed spike level	E-17

List of Symbols

Symbol	Definition
$\times$	Cross product.
$\cdot$	Dot or inner product.
$x(t)$	Continuous time-domain functions use round brackets.
$x[n]$	Discrete time-domain functions use square brackets.
$\tilde{x}$	Quantized values are denoted using a waved overbar.
$\hat{x}$	Estimators are denoted using a circumflex overbar.
$\vec{x}$	Rays are denoted using an arrowed overbar. They consist of an origin and a direction.
$\mathbf{x}$	Vectors are bold-faced and in lower case.
$[\mathbf{x}, \mathbf{y}]$	Terminated rays are represented in angle brackets as a pair of vectors marking the start and end of the ray.
$\mathbf{A}$	Matrices are bold-faced and in upper case.
A	Angles are expressed as either greek symbols or upper case letters. Unless otherwise stated, angles are assumed to be measured in radians.
μ_x	Arithmetic mean of a signal x .

Symbol	Definition
$\bar{x}$	Arithmetic mean values based on a set of measurements are represented with an overbar. Unless otherwise stated, the term mean refers to the arithmetic mean. The term average may be used interchangeably with the term mean.
s_x	Standard deviation of a variable x based on a set of measurements. Unless otherwise stated, the term standard deviation refers to the sample standard deviation.
σ_x	Standard deviation of a signal x . Unless otherwise stated, the term standard deviation refers to the sample standard deviation.
s_x^2	Variance of a variable x based on a set of measurements. Unless otherwise stated, the term variance refers to the sample variance.
σ_x^2	Variance of a signal x . Unless otherwise stated, the term variance refers to the sample variance.
h_x	Distance along the x-axis between the (y,z)-plane and each fixed mirror, generally expressed in millimetres.
h_y	Distance along the y-axis between the (x,z)-plane and the y-axis mirror centre of rotation, generally expressed in millimetres.
h_z	Distance along the z-axis from the (x,y)-plane and the y-axis mirror centre of rotation, generally expressed in millimetres.

Symbol	Definition
D_l	Distance between the (x,y)-plane and the lens plane, generally expressed in millimetres.
f	Distance between the lens plane and the CCD along the z-axis, generally expressed in millimetres.
d	Diameter of lens, generally expressed in millimetres.
β_{CCD}	Angular deviation of the CCD along the y-axis from a plane parallel to the (x,z)-plane, generally expressed in radians.
β_x	Angular deviation of x-axis mirror from the (x,z)-plane galvanometer when $\theta = 0$, generally expressed in radians.
β_y	Angular deviation of x-axis mirror galvanometer from the (x,y)-plane when $\phi = 0$, generally expressed in radians.
β_{out}	Angular deviation of the output fixed mirror from the (x,z)-plane, generally expressed in radians.
β_{in}	Angular deviation of the output fixed mirror from the (x,z)-plane, generally expressed in radians.
λ	Wavelength of light emitted by laser, generally expressed in nanometres.
L_{CCD}	Length of the CCD array, generally expressed in millimetres.
L_{offset}	Distance from the edge of the CCD array of the intersection of the CCD plane with the z-axis, generally expressed in millimetres.

Symbol	Definition
σ_p	Peak detection standard error.
$\theta(t)$	Angular deviation along the (x,z)-plane from the z-axis of a point in space, generally expressed in radians.
$\phi(t)$	Angular deviation along the (y,z)-plane from the z-axis of a point in space, generally expressed in radians.
$\Theta(t)$	Angular deviation of mirror which controls the deviation of the laser point along the x-axis from the z-axis, generally expressed in radians.
$\Phi(t)$	Angular deviation of mirror which controls the deviation of the laser point along the y-axis from the z-axis, generally expressed in radians.
$\alpha_U(t)$	Angular deviation of a detected point on the CCD from the z-axis, generally expressed in radians.
$\Lambda(t)$	Projected point of origin of the optical axis, expressed in millimetres.
$\Gamma(t)$	Projected point of origin of the laser, expressed in millimetres.
$\mathbf{Y}(t)$	Point of intersection of the laser vector with a surface in the environment. This is expressed in millimetres which in the scanner frame of reference and in metres when in the environmental frame of reference.
$k[n]$	Unit vector multiplier used in processing environmental interactions.

Symbol	Definition
$U(t)$	Angular deviation of the output ray along the (x,z)-plane from the positive z-axis, generally expressed in radians.
$u(t)$	Normalized angular deflection of the output ray along the (x,z)-plane from the positive z-axis. This is a unitless variable.
$V(t)$	Angular deviation of the output ray along the (y,z)-plane from the positive z-axis, generally expressed in radians.
$v(t)$	Normalized angular deflection of the output ray along the (y,z)-plane from the positive z-axis. This is a unitless variable.
$U_{OP}(t)$	Angular deviation of the optical path along the (x,z)-plane from the positive z-axis, generally expressed in radians.
$V_{OP}(t)$	Angular deviation of the optical path along the (y,z)-plane from the positive z-axis, generally expressed in radians.
$\tilde{a}_{\Theta}(t)$	Quantized value returned by x-mirror galvanometer.
$\tilde{a}_{\Phi}(t)$	Quantized value returned by y-mirror galvanometer.
$P(t)$	Location of signal peak detected by CCD, generally expressed in millimetres.
$\tilde{P}(t)$	Quantized location of signal peak detected by CCD .
$I(t)$	Intensity of return signal detected by CCD.
$R(t)$	Directed distance from the origin of $\mathfrak{R}_{RAS}$ to a pointed in space, generally expressed in metres.

Symbol	Definition
$\mathbf{u} = [u \ v \ w]^T$	A point defined using the normalized angular deflection (UVW) coordinate system. All three values are unitless.
$\mathbf{u}_R = [u \ v \ R]^T$	A range point defined using the normalized angular deflection (UVR) coordinate system. Only R has units and is generally expressed in metres.
$\mathbf{x} = [x \ y \ z]^T$	A point defined using the Cartesian coordinate system, generally expressed in metres.
$\tilde{\mathbf{a}}_P = [\tilde{a}_\Theta \ \tilde{a}_\Phi \ \tilde{P}]^T$	A peak point defined using the original mirror angles normalized by the maximum angular deflection. These values are referred to as raw samples because they represent values generated by the galvanometers and peak detector.
ω	Frequency at which a Lissajous pattern is repeated expressed in radians/second. This is referred to as the pattern refresh rate.
$(m : n)$	A Lissajous pattern is defined by a ratio of sinusoidal frequencies. $m\omega$ defines the period of the sinusoid along the x-axis. $n\omega$ defines the period of the sinusoid along the y-axis. Both m and n are unitless variables.
$\mathfrak{R}_{RAS}$	Random Access Scanner frame of reference. This frame uses units of millimeters.
$\mathfrak{R}_{World}$	World frame of reference. This frame uses units of metres.
$T_{World RAS}$	Translation from $\mathfrak{R}_{World}$ to $\mathfrak{R}_{RAS}$. Conversion from metres to millimetres are included in this translation. This translation is the inverse of $T_{RAS World}$.

Symbol	Definition
$T_{RAS World}$	Translation from $\mathfrak{R}_{RAS}$ to $\mathfrak{R}_{World}$. Conversion from millimetres to metres are included in this translation. This translation is the inverse of $T_{World RAS}$.
$Y_{World RAS}$	Rotation from $\mathfrak{R}_{World}$ to $\mathfrak{R}_{RAS}$ using the convention Roll, Pitch, then Yaw. Roll refers to a rotation of the (x,y)-plane about the z-axis. Pitch refers to a rotation of the (x,z)-plane about the y-axis. Yaw is a rotation of the (y,z)-plane about the x-axis. All rotations obey the right-hand rule. This rotation is the inverse of $Y_{RAS World}$.
$Y_{RAS World}$	Rotation from $\mathfrak{R}_{World}$ to $\mathfrak{R}_{RAS}$ using the convention Yaw, Pitch, then Roll. Yaw refers to a rotation of the (y,z)-plane about the x-axis. Pitch refers to a rotation of the (x,z)-plane about the y-axis. Roll is a rotation of the (x,y)-plane about the z-axis. All rotations obey the right-hand rule. This rotation is the inverse of $Y_{World RAS}$.
$Q_{World RAS}$	Homogeneous transformation matrix from $\mathfrak{R}_{World}$ to $\mathfrak{R}_{RAS}$. Transformation order is: Translation, Roll, Pitch, Yaw. This transformation is the inverse of $Q_{RAS World}$.
$Q_{RAS World}$	Homogeneous transformation matrix from $\mathfrak{R}_{RAS}$ to $\mathfrak{R}_{World}$. Transformation order is: Yaw, Pitch, Roll, Translate. This transformation is the inverse of $Q_{World RAS}$.

Chapter 1

Introduction

The National Research Council of Canada (NRC) has developed a laser range scanner (LRS) to perform detection and tracking of one or more objects using an auto-synchronized triangulation method developed by Rioux [8]. The scanner generates a single continuous laser signal and detects the position and intensity of the spot formed by the laser intersecting a surface in the environment. This information can be used to characterize one or more target objects based on range or intensity profiles or both. The NRC are currently developing methods to perform robust, generalized object detection and tracking at ranges between 0.5-metres and 10-metres [13].

The NRC's LRS system has been configured to use raster and Lissajous scan patterns to obtain range and intensity information. Lissajous scan patterns have been used to perform real-time object tracking using either intensity or range data [12] [13]. Moreover, Lissajous scans can be performed quickly and provide good coverage of the region of interest (ROI). The system has successfully tracked objects based on intensity information and the NCR as recently began developing

ways to track objects using range information [12]. Currently the system performs range-based tracking by adjusting the position of the Lissajous pattern so that the point of minimum range is centred within the pattern.

Figure 1.1: Examples of two common types of edges

This thesis is concerned with the detection of object boundaries as a precondition to boundary-based object tracking. Object boundaries consist of step edges that, in some cases, may be distorted to appear as steep ramp edges. An example of a step edge and a steep ramp edge can be seen in Figure 1.1. Therefore, the development of a boundary detection method becomes the identification and adaptation of an appropriate edge detection algorithm.

For purposes of discussion, data that appears as a vector is referred to as 1-dimensional (1-D) and data that appears as a matrix is referred to as 2-dimensional (2-D). Range data can be represented geometrically or as a range image. In the former case, the position, shape and orientation of a surface are of primary interest [2, p.1]. The surface is considered to occupy a position in space and may be represented as a surface contour or a series of bounded planar surfaces. In the latter case, the range information is treated in a manner similar to intensity information in that range values are represented as elements in a 1-D or 2-D array. The former represents the format of the data obtained using a Lissajous scan while

the latter represents the format of the data obtained using a raster scan. This thesis considers range information obtained from the laser range scanner as either a range image when obtained from a raster scan or a range profile when obtained from a Lissajous scan. Based on these assumptions, common image processing techniques can be used to perform edge detection. It should be noted that the term 3-dimensional (3-D) is used where it would be more accurate to define a structure as 2.5-dimensional (2.5-D).

Real-time tracking using raster-scanning techniques is not practical because of the amount of time it takes to perform a single raster scan. Non-linear closed-loop line scan patterns like the Lissajous scan pattern can be performed much faster so can be applied to object tracking [13]; however, the range data obtained using a Lissajous scan is sparse so standard 2-D edge detection techniques cannot be performed. As a result only single dimensional edge detection methods would be applicable.

The LRS system used by the NRC obtains raw sensor information and transforms it into Cartesian coordinates; however, calibration introduces non-linear transformations that can make it difficult to detect edges. Moreover, the transformation process is computationally expensive, can introduce round-off errors and more accurate transformation algorithms require more processing time. Therefore, a practical real-time edge detection method should use raw sensor data rather than transformed data.

Under certain conditions the raw sensor data can be corrupted with aperiodic peak signals that deviate significantly from expected peak values, generally referred to as outliers. These outliers appear as spikes representing no range signal (zero spike) or an incorrect range value significantly different than the previous

or next range measurement (non-zero spike). Therefore, the edge detection algorithm must be robust to outliers. Zero and non-zero spikes can be the result of many environmental factors such as occlusion and signal re-reflection but for the purposes of this study are handled as aperiodic noise.

The limitations to an appropriate edge detection method can be summarized as follows:

1. The method can only work with raw sensor data. The UVP coordinate system is introduced in Chapter 2 as a polar-like coordinate system consisting of raw sensor readings. The edge detection process is complicated by the fact that planar surfaces appear as curved surfaces in UVP-space.
2. Only 1-D range data is available. In Chapter 2 the Lissajous scan pattern is described in detail as is the format of the data it generates. Moreover, in Chapter 4 it is shown that the non-linear path followed by the laser during a Lissajous scan can introduce artificial curves into the range data, further complicating the process.
3. The method must be robust to noise typically experienced by a LRS unit. The noise detected in typical LRS data is examined in Chapter 4 and is shown to consist of additive noise and outliers.
4. The method must operate in real-time in a closed loop tracking system. This algorithm will be used for closed loop object tracking so ideally it should be completed in the time it takes for the system to complete the next scan.

There are currently a limited number of LRS units based on the NRC system so a LRS unit could not be devoted exclusively to this project. Before any

development work could be initiated a functional model of the LRS system was developed and is described in Chapter 3. This required a detailed examination of the operation of the NRC's LRS which is described in Chapter 2.

1.1 Thesis Objectives and Overview

The purpose of this thesis is to describe a method of performing real-time edge detection using untransformed sensor readings obtained using a sparse scanning method. There are a limited number of RAS units available so a functional model of the RAS was developed so that algorithms to be used on this system could be developed off-line. The functional model was used to develop and test the edge detection algorithm under simulated conditions. The completed algorithm was then tested using an actual RAS unit at the NRC.

In **Chapter 2** the structure and operation of the RAS are described in general. Lissajous scan patterns are then introduced as an efficient method of quickly scanning a region of interest. This chapter concludes by summarizing challenges posed in converting sensor readings obtained from the RAS into range information.

In **Chapter 3** the functional model of the RAS described in Chapter 2 is presented. This is followed by a description of a model of the environment within which the simulated scanner can operate. Results of calibrating the model to two RAS units located at the NRC are presented and simulated data are compared to data obtained from each of the RAS units. This chapter is based on work originally described by MacKinnon *et al.* [62] but incorporates some improvements and simplifications.

In **Chapter 4** a method of performing edge detection using raw sensor readings

from a sparse scanning method is described. This method was developed using the model described in Chapter 3 before being tested on a RAS unit at the NRC. This chapter is based on work originally described by MacKinnon *et al.* [63] but does not include the planar surface representation.

Discussion arising from the work performed is presented in **Chapter 5**. Applications of the functional model in general and real-time edge detection in particular are discussed. The chapter ends with conclusions drawn from the work described herein.

Some of the topics covered may require more detailed background information. Appendices have been provided to cover selected topics in greater depth. **Appendix A** describes the mathematical basis for the scanner model equations seen in Chapter 3. **Appendix B** shows how the environmental model seen in Chapter 3 was derived using a more mathematically rigorous format. **Appendix C** shows the results of calibrating the simulated scanner using data collected from two scanners at the NRC. **Appendix D** describes the edge metrics used in Chapter 4 in greater detail. **Appendix E** describes the details of the noise analysis and an examination of the noise filters considered in this thesis. **Appendix F** to **Appendix K** contain tabulated results and graphics used to evaluate the edge detection algorithms developed in Chapter 4. **Appendix L** describes the Matlab scripts developed as part of this thesis.

1.2 Boundary Detection

Feature-based object detection can be divided into surface detection and boundary detection. Surface detection involves isolating features based on surface geome-

try such as planar surfaces or curves. This often involves an iterative process to determine the best-fit geometric approximation of the surface. Boundary detection involves locating the edges that define an object's boundaries. Non-iterative techniques are typically used to detect the location of edges. Real-time feature-based object tracking requires the use of techniques that are performed quickly and accurately.

1.2.1 Edge Detection

Trucco and Verri [2, p.69] defined edge points as "...pixels at or around which the image values undergo a sharp variation". Edges, for the purpose of this study, are considered to be significant changes in range data. Edge elements can also be referred to as edgels or edge elements [2, p.69].

Edge detection is described as consisting of three distinct stages: noise filtering, edge enhancement, and edge localization [2, pp.69-71]. Noise filtering involves reducing the effects of additive and impulse noise and removing outliers so that edge elements become clearer. This generally involves balancing noise suppression and edge localization. Edge enhancement involves making the point of transition stand out from the background signal. Edge localization involves ensuring the detected location of the edge closely approximates the actual location of the discontinuity in the original signal.

Edges have been divided into step, ramp, and ridge edges [2, pp.71-72]. Step edges in range images signify abrupt changes in the range of a surface without a significant change in surface slope, an example of which can be seen in Figure 1.1. These edges signify occlusion by an object or part of an object. Ramp edges in range data signify sudden changes in surface slope. Figure 1.1 shows an example

Figure 1.2: Examples of three other classifications of edges

of a steep ramp edge. Ridge edges in range images signify the junction of two surfaces of different slopes. Ridge edges may or may not have a flattened peak, the latter being subclassified as a roof edge. Figure 1.2 shows an example of ridge and roof edges. Jain [64, pp.140-142] also classified a line edge as a ridge edge in which the joining surfaces are invisible to the detector. An example of a line edge can be seen in Figure 1.2. The detection of object boundaries specifically involves the detection of step edges; however, many step edges appear as ramp edges from the perspective of an edge detector. This means that an edge detector must detect either true step edges or ramp edges with steep slopes. Thus, the concept of a steep versus a non-steep edge must be clarified.

An edge detector describes a process used to extract edges from image data. Edge enhancement routines provide algorithms for making edges stand out in a data set. Edge enhancement routines and noise filters are typically combined in the literature under the broad heading of edge detectors. Some may include an edge isolation method but more often this is not discussed. A complete edge detector is therefore defined as a process consisting of noise filtration, edge enhancement and edge localization. The edge detector output will typically generate a single non-zero value for each edge detected.

1.2.2 Edge Enhancement Methods

Three common edge enhancers are the Robert's Cross, Sobel and Prewitt algorithms. All three filters are applied along the x- and y-axes and the two resulting images are added to produce a composite edge image. Thresholding is typically used to eliminate noise. These edge enhancers are sensitive to edge angle but one way to reduce this error is to rotate the detector and select the maximum convolution result at each point [59]. The Compass algorithm [65] uses Sobel and Prewitt or similar convolution masks. Rather than calculate the result of two rotations, the Compass algorithm requires rotating the mask through 8 possible orientations and creating a composite image from the sum of the mask rotations. The maximum result of the convolution at each point is selected. Once again the result is typically thresholded to reduce noise.

Canny [34] suggested an iterative process to optimize the selection of edge elements. The first derivative or Laplacian of the image is calculated and the largest window element above a minimum threshold is selected. Neighbouring elements are selected provided they exceed a second threshold value. Values below the second threshold are set to zero - a process referred to as "non-maximal suppression". This results in thinner edges and better noise handling. Canny typically precedes this process with a Gaussian noise filter, which smoothes the data but introduces some edge blurring [2, pp.71-79] [34] [64, pp.169-173].

All of the edge enhancers mentioned have been designed for 2-D data. Lissajous scans generate a temporal stream of data which is best represented as 1-D data. Lissajous scans represented as 2-D data sets are sparse in that many array elements have neighbours that are undefined. For this reason, Lissajous scans will be represented in this thesis as 1-D data sets; however, this means that the

output of each edge cannot be maximized by changing the mask orientation. A traditional approach to 1-D edge enhancement is to use a derivative filter [64, pp.143-145]. This generates a peak that increases with slope. A step edge has a slope that approaches infinity in the ideal case so step edges should be easily detected using a derivative filter. Discrete differencing is the simplest form of derivative filter and involves calculating the difference between successive adjacent elements [30, pp.87-90] [64, pp.144-145]. A discrete differencing formula using a forward approximation can be expressed as

$$b[x] = a[x + 1] - a[x] \quad (1.1)$$

where $a[x]$ and $a[x + 1]$ are adjacent signal elements and $b[x]$ is the resulting signal element [2, pp.311-314]. One drawback to discrete differencing is that it is not symmetric about the central pixel.

An alternative to discrete differencing is to use an n-point difference approximation. Typically either a 3-point or 5-point approximation is employed and can be configured to be symmetric about the central pixel. A symmetric difference approximation is referred to as a central-difference approximation and has a smaller truncation error than the discrete differencing formula [2, pp.311-314] [60, pp.146-154]. A 3-point symmetric formula has the form

$$b[x] = \frac{a[x + h] - a[x - h]}{2h} \quad (1.2)$$

while a 5-point symmetric formula has the form

$$b[x] = \frac{8a[x + h] - 8a[x - h] + a[x - 2h] - a[x + 2h]}{12h} \quad (1.3)$$

where h represents the step size. The step size can generally be made equal to one to simplify (1.2) and (1.3). The symmetric form of the derivatives were selected

to minimize the error associated with each approximation [2, pp.311-314] [60, pp.149-150].

Typically, a derivative has been combined with a Gaussian filter to enhance edges such as in the Canny edge detector or the Marr-Hildreth edge detector [2, pp.71-79] [30, pp.105-109] [30, pp.111-114] [64, pp.169-173]. The latter method uses the second derivative or Laplacian of the signal and edges are detected as zero crossings. The first derivative stage of the Marr-Hildreth detector can be defined as the edge enhancement stage, and the first derivative of that signal can be defined as the edge localization stage.

A second edge enhancement formula is the product of difference of averages (PoD) [30, pp.103-105]. According to Ritter [30, pp.103-105] this edge enhancer is supposed to be insensitive to noise and better able to preserve edge localization. The formula for PoD is

$$b[y] = \frac{1}{M} \sum_{x=1}^M |a[y+x] - a[y-x]| \quad (1.4)$$

where M is the number of elements in each of the neighbourhoods, y is the location of the central element, and x is an offset to a location in each of the neighbourhoods. A neighbourhood is the set of elements adjacent to an array element so the upper neighbourhood of size M would consist of the elements $\{a[y+1], a[y+2] \dots a[y+M]\}$ and the lower neighbourhood would consist of the elements $\{a[y-1], a[y-2] \dots a[y-M]\}$.

This PoD method can be summarized as finding the average difference between paired elements that are an equal number of steps above and below $a[y]$. Unlike the derivative filters this enhancement filter can be expanded or contracted to optimize edge enhancement for different edge types and noise levels by changing

M . Note that in this form of the PoD filter the centre point has been ignored so that the filter would be symmetric about $a[y]$.

Oliver *et al.* [58] described a modification to the PoD filter in which the averages are compared using a t-test so that window size and observed Gaussian noise levels can be taken into account. They also proposed using a normalized difference estimator (NDE) but stated that a choice of normalization operators exist. The strength of normalizing the PoD is that the resulting measure is independent of scale. The LRS system operates within a range of 0.5 to 10-metres so scaling is expected to be a problem [58].

1.2.3 Noise Filters

Noise filters are used to reduce the effects of random noise on the edge detectors previously described. Noise filters are typically divided into linear and non-linear filters. Linear filters are employed to reduce random noise that follows a particular distribution such as Gaussian noise. Non-linear filters are used to eliminate random noise such as impulse noise. We use the term random to describe any signal being introduced by a source that we are unable to monitor or model.

Independent additive noise can be estimated by assuming that, given enough data points, the noise data in general follows a particular distribution. Often this is a Gaussian distribution. This class of filters have the form

$$b[y] = \sum_{x=1-M}^{M-1} a[y+x]G(x, M) \quad (1.5)$$

where $a[y+x]$ is an element in the original signal, $b[y]$ is an element in the filtered signal, x is an offset to a point in the neighbourhood of y , and $G(x, M)$ is a function defining the filter weights. In this case M is the number of elements

above or below $a[y]$ that have been included in the filter window so the total filter length is $2M + 1$.

Two popular linear filters are the local averaging filter and a Gaussian filter [30, pp.57-60]. These filters smooth the data but may be affected by impulse noise and may blur edges. The function of filter weights for a local averaging filter can be defined as

$$G(x, M) = \frac{1}{2M + 1} \quad (1.6)$$

where M is the number of elements above or below $a[y]$ that have been include in the filter window.

The local averaging filter reduces the variation in range values by replacing the central value with the weighted sum of the range values within the window where the weighting is the inverse of the window size. According to Trucco and Verri [2, p.56] an averaging filter will reduce Gaussian noise but will result in edge blurring . They warned that this could reduce the accuracy of results from the edge localization step. The averaging filter will not remove impulse noise but will diffuse it. From a frequency perspective, the application of a mean filter results in secondary lobes that are a new source of noise introduced into the signal. Pitas [42] compared the response of the arithmetic mean with that of the geometric mean, harmonic mean, L_p mean and contraharmonic mean in response to step noise but performed no formal evaluation.

A Gaussian filter smoothes the data by replacing the central value with the weighted sum of the range values within the window [2, pp.56-62]. The Gaussian filter is not expected to remove impulse noise but should attenuate it [2, p.56]. The benefit of the Gaussian filter over the averaging filter is that the Gaussian

does not generate secondary lobes that form new signal noise in the filtered signal [2, p.62]. The weighting function for a Gaussian filter is defined as

$$G(x, M) = e^{\frac{-x^2}{2(\sigma(M))^2}} \quad (1.7)$$

where $\sigma(M)$ is the spread of the Gaussian function. Trucco and Verri recommended $\sigma(M) = M/5$ [2, pp.59-60].

The second class of filters is the non-linear filters. A popular non-linear filter is the median filter which removes outliers while minimizing the loss of edge information [29, pp.271-273] [30, pp.65-68]. Burian and Kuosmanen [45] stated that the unweighted median filter provides better noise attenuation than a weighted median filter so only unweighted median filters will be considered. Median filters can also be applied recursively to obtain a root signal within a finite time period [45] [46]. The root signal is defined as a signal that does not change after further iterations of the filter. Such filters are referred to as idempotent because the signal generated by the filter is the root signal [45].

A median filter smoothes the data by replacing the central value with the median of the range values within the window. The resulting filtered signal features less blurring and isolated spikes are removed [30, pp.65-66]. Davies [47] reported that under some circumstances the median filter can leave more noise than mean or Gaussian filter noise. Burian and Kuosmanen [45] noted that in the case of a recursive median filter the noise is cumulative. Trucco and Verri [2, pp.62-63] recommend the median filter as a way to sharpen edges. Pitas [42] indicated that a median filter, along with filters based on order statistics and other non-linear filters, are better suited to handling impulse noise and preserving edges. Zou [53] observed that the ability of the median filter to reduce noise depends upon finding

a window size that is neither too large nor too small. Restrepo and Chacon [55] determined that the median filter is effective in smoothing a signal.

A limit to median filter performance is the computational cost of repeatedly sorting each window. Mittermair and Puschner [50] compared the worst-case execution times of eight common sorting algorithms and found that results varied greatly depending upon the window size. Sorting only the initial window, then performing insertion and removal of data points such that the order is preserved, can be used to reduce the computation time. Angelopoulos and Pitas [48] applied this technique successfully to 2-D data using a weighted median filter. Hardware solutions are available to further reduce processing time [51]. A median filter can be defined as

$$b[y] = \text{median}\{W(M, a[y])\} \quad (1.8)$$

where $W(M, a[y])$ is a window of size $2M + 1$ centred on $a[y]$. As stated earlier, M is the number of elements above or below $a[y]$ that have been included in the filter window .

An iterated filter can be defined as a 3-element median filter applied recursively until the resulting signal and the input signal are the same; that is, the resulting signal is a root signal. The signal obtained using this technique is referred to as the signal root or invariant [54] [56]. According to Qiu [57] the recursive median filter has better noise attenuation than standard median filter.

1.2.4 Edge Localization

Edge localization involves edge thinning and distinguishing between noise peaks and peaks denoting edges. Edge thinning is referred to as non-maximum suppres-

sion and peak selection may involve thresholding [2, p.70] [64, pp.170-172]. Jain [64, p.146] divided this step into separate detection and localization steps. These steps corresponded to thresholding and thinning stages respectively.

A common edge localization method involves taking the derivative of the edge-enhanced signal to generate zero-crossings which should correspond to detected edges [30, pp.111-114] [64, p.149-157]. In many cases a derivative filter is applied twice in the edge enhancement stage so that the enhanced edges are not represented as peaks but as zero-crossings [30, pp.111-114] [64, p.149]. One drawback to edge localization is the need to then search for zero crossings as an additional step in the process

An alternative to edge localization using the derivative is to perform peak detection. This method requires a single step to detect local maxima given an appropriate window size. Intuitively this should be faster than edge localization using the derivative. Moreover, the edge detection method can be fine-tuned to improve the localization of edge detection using the slope of the enhanced edge peak.

1.3 Summary

The NRC has developed a LRS system that uses triangulation to obtain range and intensity measurements within a 0.5 to 10-metre range. The system has been configured to use raster and Lissajous scan patterns, the latter being used to perform object tracking. The system currently tracks objects based on intensity measurements and recent work has shown that objects can be tracked using range information, for example, using simple centroid algorithms.

This thesis is concerned with the detection of object boundaries as a precondition to real-time object tracking. Raw sensor information is used to reduce processing time and to avoid possible errors inherent in converting raw data to Cartesian coordinates. Raw sensor data represents flat surfaces as curved and is corrupted by Gaussian noise and spikes. The requirement of real-time operation combined with surface curvature and noise makes edge detection a challenge. Lissajous scans generate 1-D temporal data sets rather than the 2-D data sets generated using a raster scan which places an additional constraint on the methods that can be used. There are also a limited number of LRS units available so a software model is required so that much of the development work can be performed offline.

Boundary detection is essentially a problem in detecting step and steep ramp edges. For purposes of discussion, edge detection is divided into three stages; noise filtering, edge enhancement and edge localization. Typical noise filtration methods include the Gaussian filter, the averaging filter, the median filter and the iterated median filter. Simple edge enhancement methods include the derivative method and the PoD method. Edge localization can be performed using either the detection of the zero crossing of the derivative of the edge-enhanced data or using a peak detection method.

This document has been divided into 5 chapters including this Introduction. The second chapter describes the LRS system designed by the NRC in general and Lissajous scanning in particular. The third chapter describes the software system designed to simulate the behaviour of a typical LRS unit. The fourth chapter describes the determination of an appropriate edge detection method and the implementation of this method in a real-time system. The final chapter summarizes

and discusses the results, and introduces some areas for potential future research.

Chapter 2

Random Access Scanner

The NRC's LRS is able to perform object detection and tracking at ranges between 0.5-metres and 2-kilometres [8]. The system currently performs object tracking by detecting and tracking retro-reflective (Retro), Inconel black-on-white (B/W) and white-on-black (W/B) targets [12] [13]. Geometrical tracking is currently limited to the detection of planar surfaces [12]. The NRC would like to be able to perform robust, generalized object detection and tracking using this system [13].

The NRC's LRS system has been configured to use raster scan patterns such as in Figure 2.1 or Lissajous scan patterns as in Figure 2.2 [13]. The latter can be performed quickly and provides good coverage of the region of interest (ROI). Furthermore, the Lissajous pattern is a closed scan so 1-D edge detection techniques can be used to perform boundary detection. The resulting 2-D sparse edge map (SEM) can be used to perform object detection and, eventually, object tracking.

Figure 2.1: Raster-mode scanning
(reprinted from Figure 4 of [13])

Figure 2.2: Lissajous-mode scanning
(reprinted from Figure 5 of [13])

2.1 Overview of Random Access Scanner

The auto-synchronizing variable-resolution laser range scanner developed by the NRC is commonly referred to as the Random Access Scanner (RAS) and can be seen in Figure 2.3. The RAS system has been designed to be integrated with the Space Vision System (SVS) that is being developed by Neptec Design Group for the Canadian Space Agency (CSA) [6]. The RAS system was designed to operate under a wide range of lighting conditions from full darkness to sun saturation [14].

As can be seen in Figure 2.3, the system employs two precision galvanometers, each controlling a mirror used to direct a laser beam. This beam strikes a target object and is reflected back to the camera. The same system redirects the returning beam (referred to as the point path) onto a linear charge-coupled device (CCD) array. Galvanometers control the angle of incidence of the outgoing laser beam and incoming point path. By adjusting the voltage applied to each

Figure 2.3: Internal structure of Random Access Scanner

Figure 2.4: Optical path (modified from Figure 3 of [11])

galvanometer the horizontal and vertical direction of the laser beam can be controlled. Figure 2.4 is a sketch of the optical path through the camera in Figure 2.3.

Focusing of the laser system determines short (0.5 to 2-meters) and medium (2 to 10-metres) range information using an auto-synchronized triangulation method developed by Rioux [8]. The range is determined by the location of the peak return signal on the CCD array. Long (10 to 2000-metres) range information is obtained using a time-of-flight (TOF) unit. The peak signal intensity is also provided by the system and is used to detect Retro, B/W and W/B targets.

The system has been configured to obtain range and intensity information using either a raster scan pattern or a Lissajous scan pattern. When in imaging mode, the scanner uses a raster scan pattern to obtain dense range or intensity maps. Tracking mode uses one or more Lissajous scanning patterns to obtain sparse range and intensity information [14].

In tracking mode the intensity information is used to determine the centroid

of a target. Position and orientation of the object can then be determined by comparing the range and orientation of detected targets. The Lissajous scanning pattern can track the pattern by centring itself on the centroid of the target intensity data as demonstrated in Figure 2.5. Similarly, the size of the pattern can be automatically adjusted to compensate for the range of the target [14].

Figure 2.5: Target tracking error (reprinted from Figure 4 of Reference [11])

2.2 Overview of Lissajous Patterns

Lissajous patterns are “...a family of curves which are described by a point whose motion is the resultant of two simple harmonic motions in perpendicular directions.” [15]. They were studied by Nathaniel Bowditch in 1815 and later by Jules-Antoine Lissajous around 1850 [16, p.65]. According to [9], a Lissajous pattern is formed by a graph consisting of two sinusoids, one plotted along the x-axis using

$$x(t) = A_x \cos(m \omega t + \phi_x) + X_o, \quad (2.1)$$

and the other plotted along the y-axis using

$$y(t) = A_y \cos(n\omega t + \phi_y) + Y_o. \quad (2.2)$$

The form of the pattern is determined by the frequencies $m\omega$ and $n\omega$, and by the phase lags ϕ_x and ϕ_y of each of the sinusoids [9] [17]. The constants m and n can, in general, be any real value; however, if the m/n is not a rational number then the resulting pattern changes over time. Only static Lissajous patterns are useful in this study so the only values of m and n that are used are those for which m/n is a rational number. This is most easily ascertained for integers so only integer values of m and n are used. Table 2.1 shows a series of low-order patterns generated by only varying the integer ratio ($m : n$) of the frequencies. Each pattern is identified by the combination ($m : n$) which defines how the frequency of each orthogonal sinusoid relates to each other [17] [18].

Table 2.1: Lissajous patterns (a) 1:1, (b) 1:2, (c) 1:3, (d) 1:4, (e) 2:3, (f) 3:4 (g) 4:5 and (h) 5:6

The refresh rate of the scan is $2\pi/\omega$ and the size of the pattern is determined by the amplitudes A_x and A_y . The refresh rate refers to the frequency at which the entire pattern is repeated. The x- and y-axis offsets of the centre of the pattern are specified using X_o and Y_o respectively. The relative phases ϕ_x and ϕ_y are set to $\pi/2$ to obtain the patterns used in this study [9] [15]. This means that $\cos(m\omega t + \phi_x)$ from (2.1) can be replaced with $\sin(m\omega t)$ and $\cos(n\omega t + \phi_y)$ from (2.2) can be replaced with $\sin(n\omega t)$. Figure 2.6 shows the effect of different phase lags on the shape of a typical Lissajous pattern. The upper-right image displays a Lissajous pattern which is symmetrical across the diagonal as a result of using a phase lag of $\pi/2$.

Figure 2.6: Comparison of different phase lags in Lissajous figures

If m/n is a rational number then (2.1) and (2.2) combine to produce stable or non-rotating patterns [16, pp.77-79]. Of particular interest are those patterns that cross at the middle of the graph (patterns (b), (d), (f) and (h) in Table 2.1). These patterns can be used as scanning profiles that will include at least two points in the middle of the scanning region. Lissajous scans used during the course of this study always cross the centre and are always bilaterally symmetric.

2.3 Application to the Random Access Scanner

Lissajous patterns are generated in the RAS by driving each of the galvanometers with a sinusoidal signal of a predefined period and phase $\pi/2$. The x-axis galvanometer signal is generated using (2.1) and the y-axis galvanometer signal is generated using (2.2). This pattern has the benefit of allowing each galvanometer to experience only gradual changes. Meanwhile, the inertia of each mirror ensures that the Lissajous pattern will remain smooth [6]. The galvanometer angular positions are available from the scanner and are represented as $\tilde{a}_\Theta[n]$ and $\tilde{a}_\Phi[n]$ in this study. They represent the angular positions of the x-axis and y-axis mirrors respectively [24]. In this case $\tilde{a}_\Theta[n]$ and $\tilde{a}_\Phi[n]$ each refer to the n -th discrete sample representing the estimated angular position of the mirror measured in galvanometer counts.

Within a single scan period $2\pi/\omega$ the range data is sampled at equal time intervals. The system currently supports sampling resolutions between 128-samples/scan and 4096-samples/scan. The resulting sequence is a 1-D time-domain range sequence $\tilde{P}[n]$ obtained at sampling frequency ω_s . Recall that a single scan repeats itself at a rate ω . The sampling frequency in samples per second can be found

by $\omega_s = N\omega$ where N is the scan resolution in samples per scan. During the course of this study the 16-bit values obtained from the peak detector and the galvanometers are referred to as $\tilde{P}[n]$ and are measured in units of peak detector counts. In this case $\tilde{P}[n]$ represents the n -th discrete sample which is itself a 3-element vector consisting of the estimated angular positions of the x-axis and y-axis mirrors and the estimated location of the peak on the CCD array.

The galvanometer readings $\tilde{a}_\Theta[n]$ and $\tilde{a}_\Phi[n]$, and the peak detector reading $\tilde{P}[n]$, can be combined to represent the location of a point in the environment. This is referred to as the UVP coordinate system and is used extensively in this study. In the UVP coordinate system any point detected by the scanner can be written as $(\tilde{a}_\Theta[n], \tilde{a}_\Phi[n], \tilde{P}[n])$, is abbreviated as $\tilde{\mathbf{a}}_P[n]$ and is in units of counts.

In a calibrated system $\tilde{\mathbf{a}}_P[n]$ can be used to generate a 3-D normalized angular deflection sequence $\hat{\mathbf{u}}[n]$, also referred to as the UVW coordinate system. This sequence contains the position of each sample $(\hat{u}[n], \hat{v}[n], \hat{w}[n])$ as the normalized angular deflection from the camera scan axis. In this system $\hat{u}[n]$ and $\hat{v}[n]$ are dimensionless while $\hat{w}[n]$ is generally expressed in units of mm^{-1} or m^{-1} . The $(\hat{u}[n], \hat{v}[n])$ sequence may be represented as $\hat{\mathbf{u}}_R[n]$ by inverting the $\hat{w}[n]$ term to obtain the directed distance from the origin to the point which is referred to as the range. This sequence is referred to as the UVR coordinate system and is explained in greater detail in section 2.4.

The system can generate a 3-D position sequence $\hat{\mathbf{x}}[n]$ which contains the position of each sample $(\hat{x}[n], \hat{y}[n], \hat{z}[n])$. This represents the Cartesian coordinates of that point in the camera reference frame $\mathcal{R}_{RAS}$. This is the coordinate system typically used when analyzing range data.

The range sequence $\tilde{P}[n]$ can be analyzed as a 1-D line scan; however, no

assumptions can be made about the orientation of the scan with respect to the scene because the scanning direction changes continuously. Standard edge detection techniques appropriate for line scans can be applied to the range sequence to detect boundary transitions. The resulting edge sequence $\mathbf{b}[n]$ can then be combined with $\tilde{\mathbf{a}}_P[n]$ to form a sparse edge map. During the course of this investigation the $\tilde{\mathbf{a}}_P[n]$ sequence is used to perform edge detection and either the $\hat{\mathbf{a}}_P[n]$ or the $\hat{\mathbf{x}}[n]$ equivalent is used to represent the location of the edge in space.

2.4 Normalized Angular Deflection

The normalized angular deflection (UVW) coordinate system was proposed by Blais *et al.* [9] to decouple range measurement error from x- and y-axis position errors. Rather than define the position of each sample using Cartesian coordinates, he proposed using the normalized angular deflections $\mathbf{u} = [u \ v \ w]^T$ where $u = x/z$, $v = y/z$ and $w = 1/z$.

Figure 2.7 shows a simplified model of the RAS optical geometry. We can define $\psi = D_g/R$ where D_g is the separation between the x-axis and y-axis mirrors and R is the range measurement obtained using either triangulation or time-of-flight measurements [9]. Note that R is the directed distance from the origin to any point in space so is not limited to the xz-plane. Note as well that the coordinate system is not equivalent to the spherical coordinate system because ϕ is not a rotation about the origin, rather it is separated from the origin by an offset D_g . According to Blais *et al.* [12] the range measurement R is found by determining the location p of the peak return signal on the linear CCD array. The

Figure 2.7: Geometrical model of the Random Access Scanner (modified with permission from Figure 10 of [9])

Figure 2.8: simplified representation of the laser path from Figure 2.4

relationship is given by

$$R = \frac{f D_x \cos(\theta)}{p} + D_x \sin(\theta) \quad (2.3)$$

where f is the lens focal length and D_x is the triangulation base as shown in Figure 2.8. The triangulation base D_x is generally equal to the distance between the input and output mirrors along the x-axis. This can be used to obtain the Cartesian coordinates of the point using

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R \begin{bmatrix} \sin(\theta) \\ (\cos(\theta) - \psi) \sin(\phi) \\ (1 - \cos(\phi))\psi + \cos(\theta) \cos(\phi) \end{bmatrix} \quad (2.4)$$

where it is important to note that x , y and z all depend upon R . If u , v and w are defined as

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} \frac{x}{z} \\ \frac{y}{z} \\ \frac{1}{z} \end{bmatrix} = \begin{bmatrix} \frac{\sin(\theta)}{(1 - \cos(\phi))\psi + \cos(\theta) \cos(\phi)} \\ \frac{(\cos(\theta) - \psi) \sin(\phi)}{(1 - \cos(\phi))\psi + \cos(\theta) \cos(\phi)} \\ \frac{1/R}{(1 - \cos(\phi))\psi + \cos(\theta) \cos(\phi)} \end{bmatrix} \quad (2.5)$$

then R is removed from u and v with the exception of the ψ term. In general $D_g \ll R$ so $\psi \approx 0$. In this case (2.5) simplifies to

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} \frac{\tan(\theta)}{\cos(\phi)} \\ \tan(\phi) \\ \frac{1}{R \cos(\theta) \cos(\phi)} \end{bmatrix} \quad (2.6)$$

If R is significantly greater than D_g then R will have little effect on u and v . As a result, errors in range measurement do not contribute significantly to position measurement calculation errors [12].

In practice, $\mathbf{u}$ is specified by directing the x- and y-axis galvanometers to move the mirrors by θ and ϕ respectively to change the direction of the laser beam. The constants f and D_x are intrinsic scanner parameters and p is determined from the reflected laser light so (2.3) can be calculated. Equation (2.5) can then be used to calculate the location of the point $\mathbf{u}$ in a UVW coordinate system. Equation (2.4) is then used to convert $\mathbf{u}$ into a Cartesian point $\mathbf{x} = [x \ y \ z]^T$. The benefit of the UVW coordinate system is the ability to fit planar surfaces while minimizing the effect of range error in two of the three position terms [12].

Another way to define the position of the point is using the UVR coordinate system. The variables R , θ and ϕ can be used directly in a polar coordinate system rather than calculate w from R and work in the Cartesian coordinate system. This reduces the number of calculations and so reduces the effect of truncation errors inherent in digital systems. This system is expressed as $\mathbf{u}_R = [u \ v \ R]^T$.

In the following chapter the variables U and V are introduced to represent the deflection of a line from planes parallel to the (x, z) and (y, z) planes respectively. The (u, v) pair defines a line that originates at the origin of the reference frame. The (U, V) pair origin may be at any point within the camera frame of reference. As a result, the UVW and UVR coordinates systems do not use U and V as coordinates but may be used to define the focus point or origin of a line which itself may have deflection angles U and V . In fact, the description of a line with deflection U and V is not complete until the focus point or origin has been specified.

The above discussion provides a summary of the operation of the laser range scanner. However, a more detailed model of the scanner is required to more accurately predict the response of the system to external stimuli. In particular, a simplified model using ray optics formed a basis to examine the behaviour of the system in a dynamic environment.

2.5 Summary

The National Research Council of Canada has developed an auto-synchronizing, variable resolution laser range scanner. The current system is able to perform object detection and tracking of retro-reflective, Inconel and white-on-black targets using Lissajous scanning patterns. Target tracking using feature detection has been limited to the detection of planar surfaces.

Lissajous scan patterns are closed line scans formed using orthogonal sinusoidal patterns. Of particular interest are Lissajous patterns formed by paired integer ratios of sinusoidal periods that cross at the centre of the scanning region, and which are symmetric across the diagonal.

The location of a point in space with respect to the optical axis can be defined using the normalized angular deflection from the z-axis. For large range measurements, a point specified in the UVP coordinate system can be transformed into a point in the Cartesian coordinate system using simple trigonometric relationships. The angular deflection system is preferred to the Cartesian measurements system because it decouples the horizontal and vertical measurement error from the range error.

The system returns the current position of the galvanometers and the location

of the peak on the CCD array as 16-bit signed values. This may be used to perform edge detection which, when using Lissajous scanning patterns, can be used to develop sparse edge maps. Sparse edge maps may utilize either the Cartesian coordinate system or the UVP coordinate system to correlate detected edges to positions within the camera's field of view.

Chapter 3

Random Access Scanner

Modelling

Development of a simplified but accurate model of the RAS was a key component in simulating the robustness of the object detection system. This model uses ray optics to determine the point of intersection of the laser beam with an object surface in the environment. A modified inverting pinhole camera model was used to correlate the intersection point in the camera frame of reference to the position on the CCD array. Noise was then introduced in a controlled manner. This system provided a means to examine edge detection and object detection techniques by comparing results obtained under ideal and controlled-noise conditions.

The first stage in the development of a useful model was the examination of the optical path through the auto-synchronized triangulation system. Model development was restricted to the simulation of short and medium range data collection because the triangulation method is accurate only for ranges less than 10-metres. The simulation was further simplified by only considering the mean

optical path. The model was designed to return only range information. Range data were found as the locations of the theoretical signal peak on a simulated 512-element linear CCD array. The development of the model used in this study was based on the calibration model used by Beraldin *et al.* [5] [7] [8], and Blais *et al.* [9] [12]. MacKinnon *et al.* [62] later applied this work to a simplified model of the scanner. The work presented in this chapter encompasses the results presented in [62] and includes some improvements and simplifications.

The model was developed using Matlab and the simulation environment consisted of a series of modular scripts. The system was designed such that it could be tailored to a range of test conditions. The model represents a simulation of the principles employed by the laser range scanner developed by the NRC and is a simplified model in that it does not include inaccuracies and deviations that would normally be present in an actual system. This means that the behaviour of the scanner can be examined in principle under various test conditions without the need to eliminate or account for many variables. Ideal results can first be obtained using an ideal model, and then noise can be added in a controlled manner. Methods tested and developed using the ideal model can later be employed on the real system where these methods can be further refined to account for real system inaccuracies.

3.1 Mean Optical Path (MOP)

The Mean Optical Path (MOP) defines the series of intersection points and ray vectors that define the path of the laser out of the camera. The path starts in this model at the intersection of the laser with the x-axis mirror as can be seen in

Figure 3.1, defined here as $\mathbf{f}_1(t)$. The vector corresponding to the new laser path is then defined as $\vec{\mathbf{f}}_1(t)$. This vector terminates by striking the fixed mirror at $\mathbf{f}_2(t)$ but the laser is reflected on a new path $\vec{\mathbf{f}}_2(t)$. This vector then terminates by striking the y-axis mirror at a point defined as $\mathbf{f}_3(t)$. The new vector $\vec{\mathbf{f}}_3(t)$ starts at this point and defines the laser path out of the camera and into the environment. Eventually the laser terminates upon striking a surface in the environment at a point $\mathbf{Y}(t)$.

This section mathematically describes variables affecting the change in the laser's path from its point of incidence with the x-axis mirror to the point of incidence with the y-axis mirror. These variables determine the path the laser will follow in the environment with respect to the camera frame of reference. A more rigorous examination of the equations presented in this section can be found in **Appendix A**.

Figure 3.1 shows the MOP of the laser through the RAS system [11]. The origin of the coordinate system visible in Figure 3.2 has been defined to be the point at which the laser beam strikes the x-axis mirror. The z-axis follows the axis of rotation of the x-axis mirror and is directed toward the scanning region. The x-axis is directed along the outgoing path of the laser from the x-axis mirror to the fixed mirror. It crosses the z-axis at the point of incidence of the laser with the x-axis mirror. The y-axis is orthogonal to the x- and z-axes and is directed through the laser.

3.1.1 Determination of MOP

Determining the MOP required defining several intrinsic scanner parameters. Figure 3.1 shows the MOP of the laser through the RAS system [11]. These param-

Figure 3.1: Mean optical path (based on Figure 3 of [11])

ters included the positions of each fixed mirror, the movable mirrors, the lens and the CCD array, the angle of each mirror and the CCD, and the focal length of the lens. All mirrors were assumed to be ideal specular surfaces and both rotating mirrors were assumed to be infinitely thin; that is, the surface of the mirror passes through the axis of rotation.

Figure 3.2: Intrinsic scanner parameters

The distance between the x-axis mirror and the centre of each fixed mirror was defined as h_x as can be seen in Figure 3.2. In this simulation it was assumed that the distance between the x-axis mirror and the outgoing fixed mirror was equal to the distance between the x-axis mirror and the incoming fixed mirror. The outgoing fixed mirror was defined as the fixed mirror along the positive x-axis and the incoming fixed mirror was defined as the fixed mirror along the negative x-axis. The laser was assumed to produce a non-dispersive beam that followed the y-axis and struck the x-axis mirror at the origin of the scanner frame of reference $\mathcal{R}_{RAS}$.

The distance between the x-axis mirror and the y-axis mirror was defined as h_y along the positive y-axis and h_z along the positive z-axis. The axis of rotation of the y-axis mirror was assumed to be strictly parallel with both the (x,y)-plane and the (x,z)-plane. The x-axis mirror plane was assumed to have an angular deviation β_x from the (x,z)-plane. The y-axis mirror plane was assumed to have an angular deviation β_y from a plane parallel to the (x,z)-plane. The outgoing and incoming fixed mirror planes were assumed to have deviation β_{out} and β_{in} respectively from the (x,z)-plane. It was also assumed that the normal of each fixed mirror surface would be parallel with the (x,y)-plane. The distance between the x-axis mirror and the lens was defined as D_l , and the lens focal length was defined as f . The CCD plane was defined as having a deviation β_{CCD} from a plane parallel to the (x,z)-plane. The CCD angular deviation β_{CCD} is referred to as the Scheimpflug angle and guarantees that the image of the laser spot will always be in focus.

The vector defining the path of the laser signal out of the camera was determined by following the path of an ideal laser from the x-axis mirror to the

y-axis mirror. The result was a vector defined by the point of incidence $\mathbf{f}_3(t) = [f_{x3} \ f_{y3} \ f_{z3}]^T(t)$ on the y-axis mirror in $\mathcal{R}_{RAS}$. The direction out of the scanner was defined as a pair of angular deviations from the z-axis. The angular deviation of the vector along the positive x-axis from a vector parallel to the z-axis is defined as $U(t)$. The angular deviation of the vector along the positive y-axis from a vector parallel to the z-axis is defined as $V(t)$. They are presented as a deflection pair $(U(t), V(t))$ with respect to the z-axis.

Figure 3.3 shows the simulated scanner consisting of two fixed mirrors (left and right rectangles), an x-axis mirror (between the fixed mirrors), a y-axis mirror (top of figure consisting of two rectangles), a lens (middle of figure) and a CCD array (bottom of figure). The viewing axis is upward from the camera origin. The axes are measured in millimetres, representing the base units for the scanner frame of reference. The internal dimensions and other intrinsic camera parameters were defined in a calibration file and were based on scanner specifications provided early in the thesis work.

3.1.2 Calculating Laser Path Deviations

The path of the laser (MOP) through the scanner was defined as a series of intersection points $\mathbf{f}_1(t)$ (x-axis mirror), $\mathbf{f}_2(t)$ (fixed output mirror) and $\mathbf{f}_3(t)$ (y-axis mirror). These incidence points were connected by ray paths represented by vectors $\vec{f}_1(t)$ (x-axis mirror to fixed mirror), $\vec{f}_2(t)$ (fixed mirror to y-axis mirror) and $\vec{f}_3(t)$ (y-axis mirror to object). These paths can also be represented as terminated rays $[\mathbf{f}_1(t), \mathbf{f}_2(t)]$, $[\mathbf{f}_2(t), \mathbf{f}_3(t)]$ and $[\mathbf{f}_3(t), \mathbf{Y}(t)]$ where $\mathbf{Y}(t)$ is the termination point of the ray $\vec{f}_3(t)$ in the environment as can be seen in Figure 3.1. This figure has been redrawn as Figure 3.4 which shows the MOP and the intersection points

Figure 3.3: Matlab simulation of the Random Access Scanner

in the (x,y)-plane, and as Figure 3.5 which shows the $f_2(t)$ and $f_3(t)$ intersection points in the (y,z)-plane.

Figure 3.4: Laser path in (x,y)-plane

Figure 3.5: Laser path in (y,z)-plane

The point of incidence with the y-axis mirror can be defined as

$$\mathbf{f}_3(t) = \begin{bmatrix} h_x + h_x \frac{\sin(\varphi(t)) \cos(\beta_{out})}{\sin(\beta_{out} - \varphi(t))} \\ + h_y \tan(U(t)) - h_z \tan(\Phi(t) + \beta_y) \tan(U(t)) \\ h_y - h_z \tan(\Phi(t) + \beta_y) \\ 0 \end{bmatrix} \quad (3.1)$$

which also defines the origin of the path of the laser out of the scanner $\vec{f}_3(t)$. In this equation

$$\varphi(t) = \pi/2 - 2\beta_x - 2\Theta(t) \quad (3.2)$$

where $\Theta(t)$ is the rotation angle of the x-axis mirror and β_x is angle of the x-axis mirror at $\Theta(t) = 0$. This equation also incorporates the deflection angles of the laser path out of the scanner. They are defined as

$$U(t) = \pi - 2\beta_{out} - 2\beta_x - 2\Theta(t) \quad (3.3)$$

and

$$V(t) = 2\beta_y + 2\Phi(t) - \pi/2 \quad (3.4)$$

where β_{out} is the angle of the fixed output mirror, $\Phi(t)$ is the angle of rotation of the y-axis mirror and β_y is the angle of the y-axis mirror when $\Phi(t) = 0$. A more detailed derivation of (3.1), (3.3) and (3.4) can be seen in section A.1 of **Appendix A**. A graphical representation of $U(t)$ can be seen in Figure 3.4 and a graphical representation of $V(t)$ can be seen in Figure 3.5.

It should be noted that $\Theta(t)$ and $\Phi(t)$ are determined by sampling the galvanometer rotation angle. This means that the point of incidence of the laser with the y-axis mirror and the path of the laser out of the scanner can be determined using simple intrinsic scanner parameters and the current galvanometer readings.

Figure 3.6 shows the results of a Matlab simulation of the laser path through the RAS. According to [5] the system has a maximum field of view (FOV) of $30^\circ \times 30^\circ$. The maximum and minimum angular deviations can be defined as 15° and -15° respectively. If it is assumed that $\beta_y = \pi/4$ then $V(t) = 2\Phi(t)$, and if $\pi = 2\beta_{out} + 2\beta_x$ then $U(t) = 2\Theta(t)$. Under these conditions $\Theta_{max}(t) = 7.5^\circ$, $\Theta_{min}(t) = -7.5^\circ$, $\Phi_{max}(t) = 7.5^\circ$, and $\Phi_{min}(t) = -7.5^\circ$. The sizes of the output fixed mirror and the y-axis mirror in this simulation were selected to demonstrate the maximum field of view.

3.2 Imaging Axis

Unlike traditional laser range scanners, the RAS auto-synchronizes the imaging axis path to the output laser path [5]. The time-varying position of the imaging axis must be determined because the location of the signal peak on the CCD

Figure 3.6: Matlab simulation of laser path (solid line) through the Random Access Scanner.

array is defined with respect to the imaging axis. The imaging axis is represented as a series of reflection points as seen in Figure 3.7 starting at the lens ($a_0(t)$) and going to the x-axis mirror ($a_1(t)$), the input fixed mirror ($a_2(t)$) and finally the y-axis mirror ($a_3(t)$). These paths can also be represented as terminated rays $[a_0(t), a_1(t)]$, $[a_1(t), a_2(t)]$ and $[a_2(t), a_3(t)]$. The imaging axis is a reference against which the image of the point of intersection of the laser with an object in the environment is measured.

Figure 3.7: Imaging axis in (x,y)-plane

Figure 3.7 shows the imaging axis from the lens plane to the y-axis mirror. It is assumed that the focal point of the lens lies on the negative y-axis and that the lens plane is parallel to the (x,z)-plane.

The point of intersection of the imaging axis with the y-axis mirror can be

found using

$$\mathbf{a}_3(t) = \begin{bmatrix} -h_x + h_x \frac{\sin(\varphi(t)) \cos(\beta_{in})}{\sin(\varphi(t) + \beta_{in})} \\ + h_y \tan(U_{OP}(t)) - h_z \tan(\Phi(t) + \beta_y) \tan(U_{OP}(t)) \\ h_y - h_z \tan(\Phi(t) + \beta_y) \\ 0 \end{bmatrix} \quad (3.5)$$

where

$$U_{OP}(t) = 2\beta_{in} - 2\beta_x - 2\Theta(t) \quad (3.6)$$

defines the x-axis deflection of the imaging axis. In our configuration the fixed mirrors are generally placed such that $\beta_{out} = \beta_{in} = \pi/4$. In this case $U(t) = U_{OP}(t)$ so the imaging axis and the laser path are parallel. A more detailed derivation of (3.5) and (3.6) can be seen in section A.2 of **Appendix A**.

Figure 3.8 shows the results of a Matlab simulation of the path of the imaging axis through the RAS. In this simulation $\beta_{out} = \beta_{in} = \pi/4$. As in the case of the laser path, the sizes of the fixed axis mirror and the y-axis mirror were selected to encompass the total field of view.

3.3 Image Detection

The outgoing path of the laser (MOP) determines where the laser will intersect a surface in the environment. The imaging path provides a reference against which to measure the location of that intersection point. The task now is to determine the path from the detected point to the lens and then from the lens to the CCD. The point path is defined as an angular deflection $\alpha_U(t)$ from the imaging axis which is then used to determine the position of the point projection on the

Figure 3.8: Matlab simulation of the imaging axis through the Random Access Scanner. Solid lines represent the laser and imaging axis paths through the system.

CCD array. The point path can be represented as terminated rays $[\mathbf{Y}(t), \mathbf{b}_3(t)]$, $[\mathbf{b}_3(t), \mathbf{b}_2(t)]$, $[\mathbf{b}_2(t), \mathbf{b}_1(t)]$, $[\mathbf{b}_1(t), \mathbf{b}_0(t)]$ and $[\mathbf{b}_0(t), \mathbf{P}(t)]$ where $\mathbf{Y}(t)$ is the termination of the ray $\vec{f}_3(t)$ in the environment and $\mathbf{P}(t)$ is the termination of ray $\vec{b}_0(t)$ on the CCD array. Figure 3.9 shows the point path from $\mathbf{Y}(t)$ to the intersection with the lens plane at $\mathbf{b}_0(t)$. The path from $\mathbf{b}_0(t)$ to the intersection with the CCD array can be seen in Figure 3.11 and is described in greater detail in the next section.

It should be noted that the point path considers only the mean optical ray path through the system. The primary signal path is used to determine the position of the signal peak on the CCD array without considering the signal spread which is negligible because of the Scheimpflug angle β_{CCD} .

Figure 3.9: Point path in the (x,y)-plane

Figure 3.10: Simplified scanner model in (x,z)-plane

Consider Figure 3.9 in which a ray passes through the origin of a thin lens model. The ray forms an angle $\alpha_U(t)$ with the imaging axis in the (x,y)-plane. However, this can be simplified by unwrapping the imaging and point paths as

in Figure 3.10. The point at which the imaging and point paths cross in the lens plane can be represented as a virtual point. This reduces the problem to simple planar geometry but requires that the distance between $\mathbf{a}_0(t)$ and $\mathbf{a}_3(t)$ as seen in Figure 3.7 and Figure 3.10 be known or determined.

In Figure 3.10 a point $\mathbf{Y}(t)$ represents the intersection of the laser with an object in the environment. The simplified model can be used to determine the angle $\alpha_U(t)$ that the point makes with the imaging axis. To do this the projected intersection point of the imaging axis and point path is defined as $\Lambda(t)$. The location of $\Lambda(t)$ is a point along a projection of the imaging path through the intersection point $\mathbf{a}_3(t)$ in the y-axis mirror. The distance from $\mathbf{a}_3(t)$ to $\Lambda(t)$ is the distance from $\mathbf{a}_3(t)$ to $\mathbf{a}_0(t)$ through intersection points $\mathbf{a}_1(t)$ and $\mathbf{a}_2(t)$ as can be seen in Figure 3.7. This is found by

$$|\vec{\Lambda}(t)| = |[\mathbf{a}_0, \mathbf{a}_1](t)| + |[\mathbf{a}_1, \mathbf{a}_2](t)| + |[\mathbf{a}_2, \mathbf{a}_3](t)|. \quad (3.7)$$

where $\mathbf{a}_1(t)$ is the origin of the system assuming the surface of the mirror passes through the axis of rotation of the mirror. The point of intersection with the fixed output mirror $\mathbf{a}_2(t)$ is

$$\mathbf{a}_2(t) = \begin{bmatrix} -h_x + h_x \frac{\sin(\varphi(t)) \cos(\beta_{in})}{\sin(\varphi(t) + \beta_{in})} \\ -h_x \frac{\sin(\varphi(t)) \sin(\beta_{in})}{\sin(\varphi(t) + \beta_{in})} \\ 0 \end{bmatrix} \quad (3.8)$$

which has been derived in section A.2 of **Appendix A**.

The origin of the projection can be found by

$$\begin{bmatrix} \Lambda_x \\ \Lambda_y \\ \Lambda_z \end{bmatrix} (t) = \begin{bmatrix} a_{x3}(t) - \Delta\Lambda_z(t) \tan(U_{OP}(t)) \\ a_{y3}(t) - \Delta\Lambda_z(t) \tan(V_{OP}(t)) \\ \Delta\Lambda_z(t) \end{bmatrix} \quad (3.9)$$

where

$$\Delta\Lambda_z = \frac{|\vec{\Lambda}(t)|}{\sqrt{\tan^2(U_{OP}(t)) + \tan^2(V_{OP}(t)) + 1}} \quad (3.10)$$

and $V_{OP}(t) = V(t)$. This last point holds true because the rotation axis of the y-axis mirror is assumed to be parallel to the x-axis. This means that the imaging axis, the MOP and the point path all have the same deflection along the y-axis.

The angle $\alpha_U(t)$ can be found by first locating the point of intersection $\mathbf{b}_3(t)$ with the y-axis mirror. In Figure 3.10 it can be seen that $f_{z3}(t) = a_{z3}(t) = b_{z3}(t)$ because the mirror is parallel to the x-axis in the (x,z)-plane. Similarly $f_{y3}(t) = a_{y3}(t) = b_{y3}(t)$ because the y-axis mirror is also parallel to the x-axis mirror in the (x,y)-plane. All that remains is to find $b_{x3}(t)$ which requires simple planar geometry. It can be shown that

$$\begin{bmatrix} b_{x3} \\ b_{y3} \\ b_{z3} \end{bmatrix} (t) = \begin{bmatrix} \Lambda_x(t) + \frac{(a_{z3}(t) - \Lambda_z(t))(\mathbf{Y}_x(t) - \Lambda_x(t))}{\mathbf{Y}_z(t) - \Lambda_z(t)} \\ a_{y3}(t) \\ a_{z3}(t) \end{bmatrix}. \quad (3.11)$$

In Figure 3.10 it can be seen that the projection of the distance between $\Lambda(t)$ and $\mathbf{b}_3(t)$ onto the x-axis is related to the distance between $\Lambda(t)$ and $\mathbf{Y}(t)$ onto the x-axis through similar triangles. That is,

$$b_{x3} = \Lambda_x(t) + \frac{(a_{z3}(t) - \Lambda_z(t))(\mathbf{Y}_x(t) - \Lambda_x(t))}{\mathbf{Y}_z(t) - \Lambda_z(t)} \quad (3.12)$$

where $a_{z3}(t) - \Lambda_z(t)$ is the projection onto the z-axis of the distance between $a_3(t)$ and $\Lambda(t)$, $\mathbf{Y}_z(t) - \Lambda_z(t)$ is the projection onto the z-axis of the distance between $\mathbf{Y}(t)$ and $\Lambda(t)$, and $\mathbf{Y}_x(t) - \Lambda_x(t)$ is the projection onto the x-axis of the distance

between $\mathbf{Y}(t)$ and $\mathbf{\Lambda}(t)$. The deflection angle $\alpha_U(t)$ can then be found using the cosine law such that

$$\alpha_U = \cos^{-1} \frac{||[\mathbf{a}_3, \mathbf{b}_3](t)||^2 - ||[\mathbf{b}_3, \mathbf{\Lambda}](t)||^2 - ||[\mathbf{a}_3, \mathbf{\Lambda}](t)||^2}{2||[\mathbf{b}_3, \mathbf{\Lambda}](t)||||[\mathbf{a}_3, \mathbf{\Lambda}](t)||} . \quad (3.13)$$

3.3.1 CCD Imaging

Figure 3.11: Point projection onto CCD array through lens plane

An inverting pinhole camera model was used to determine the location of the signal peak on the CCD array [3]. Figure 3.11 shows the intersection of the ray corresponding to the laser point with the CCD array. The angle between the CCD array and a line perpendicular to the imaging axis can be defined as

$$P_\alpha(t) = \pi/2 - \alpha_U(t) - \beta_{CCD} \quad (3.14)$$

where β_{CCD} is the Scheimpflug angle of the CCD with respect to a plane parallel to the lens plane. The location of the point on the CCD array can be found using

the sine law such that

$$P(t) = L_{CCD} + L_{offset} - f \frac{\sin(\alpha_U(t))}{\sin(P_\alpha(t))} \quad (3.15)$$

where f is the focal length of the lens, L_{CCD} is the length of the CCD array, and L_{offset} is the offset of the CCD array from the z-axis. It is assumed that the CCD has been positioned such that the imaging axis intersects the CCD at a distance f from the lens plane along the negative y-axis.

Figure 3.12 shows the results of a Matlab simulation of the laser path and imaging axis as solid lines. The point path from the point of detection to the CCD is shown as a dashed line. The grid lines have been removed so that the optical path can be seen more easily. The sizes of the x-axis mirror, y-axis mirror and the fixed input mirror were refined so that any point within the field of view would reach the lens. Also shown is the projection of the imaging and point paths to the virtual lens intersection.

Figure 3.13 shows peak values obtained for a planar surface 5-metres from the scanner. The surface appears curved because the peak value represents the directed distance between the laser point on the object surface and the projection of the CCD array. Surfaces that are flat when represented using the Cartesian coordinate system will appear curved when represented using the UVP coordinate system. The curvature is most pronounced for surfaces close to the scanner and where the galvanometer deflections are wide. In the case of Figure 3.13, the full galvanometer deflections have been used.

Figure 3.12: Matlab simulation of the Random Access Scanner including projected lens intersection. Solid lines represent the laser path and imaging axis, the dashed lines represent the image path. Virtual paths followed by the imaging axis, laser and image path are also shown as dashed lines.

Figure 3.13: Matlab simulation of peak values for a planar surface 5-metres from the scanner

3.4 Quantization

The galvanometer A-to-D converters and the CCD produce 16-bit signed integer values [5]. Each galvanometer angle has a resolution of 16-bits which limits the values of $\Theta(t)$ and $\Phi(t)$. Similarly, the CCD subsystem is sampled at discrete time intervals to obtain peak and intensity values that are each represented as 16-bit signed integer values.

Each galvanometer can be modelled as a 16-bit element which accepts a value $x[n]$ such that $-1 \leq x[n] \leq 1$ representing the full-scale voltage applied to obtain a rotation between the minimum and maximum deviation angles. Applying a -1 would cause the galvanometer to rotate the mirror to the minimum value of $\Theta[n]$ or $\Phi[n]$. Similarly, applying a voltage of 1 would result in a maximal angular deviation.

The input signal to each galvanometer is quantized based on a truncation model [19, pp.190-193]. The quantization step size Δ_x is defined as

$$\Delta_x = \frac{X_{max} - X_{min}}{2^{16}} \quad (3.16)$$

where X refers to the signal being quantized, in this case either Θ or Φ , and X_{max} and X_{min} are 1 and -1 respectively. The signal $x[n]$ is then quantized using

$$\tilde{x}[n] = X_{min} + \Delta_x \left((x[n] - X_{min}) \text{div}(\Delta_x) \right) \quad (3.17)$$

where $(V_1) \text{div} (V_2)$ returns the integer divisor after dividing V_1 by V_2 . This is the number of divisions of size Δ_x between X_{min} and $x[n]$. The divisor is then multiplied by Δ_x to return to the appropriate level of scaling then added to the minimum possible value of $x[n]$ to obtain the quantized result. This results in a biased quantization model but is sufficient for this model.

The quantized $\tilde{x}[n]$ is used as the signal driving the appropriate galvanometer where $\tilde{x}[n]$ has the range $-1 \leq \tilde{x}[n] < 1$ based on two's-complement binary representation [19, p.192]. Figure 3.14 shows typical output generated by the quantization function using input values from -1.5 to 1.5 as input. This example shows the effect of quantization to a 3-bit signed integer value. We see that the quantized output is truncated at -1 and 1.

Figure 3.14: Matlab output generated by quantization function.

The detected position of the point on the CCD array may be handled in a similar manner. The position of the signal peak on the CCD array is returned as a 16-bit signed integer value [5]. The peak position $P[n]$ is modelled as a quantized

value $\tilde{P}[n]$ such that $-2^{15} \leq \tilde{P}[n] < (2^{15} - 1)$.

The RAS system returns a peak value such that $\tilde{P}[n] = 0$ when the detected point is at the minimum range and $\tilde{P}[n] = 2^{15} - 1$ when the detected point is at the maximum range or undetected. $P[n]$ must first be converted to $P'[n]$ where

$$P'[n] = \frac{P[n]}{L_{CCD}} \quad (3.18)$$

and L_{CCD} is the length of the CCD array. $P'[n]$ represents the fraction of the distance along the CCD at which the point is located and has a value between 0 and 1. First, the step size

$$\Delta_P = \frac{1}{2^{15}} \quad (3.19)$$

is defined, then

$$\tilde{P}[n] = 2^{15} \Delta_P \left((P'[n]) \operatorname{div} (\Delta_P) \right) \quad (3.20)$$

is calculated. The $\left((P'[n]) \operatorname{div} (\Delta_P) \right)$ term returns the number of divisions of size Δ_P between 0 and $P'[n]$. The result is then multiplied by Δ_P to return a value between 0 and 1. This is multiplied by 2^{15} to obtain a quantized value in the range 0 to 2^{15} which value represents the detected peak position.

3.5 Environmental Modelling

An environment consists of a World model and a series of Object models. A World model consists of six planar surfaces that completely enclose the region within which the scanner is located. These surfaces define the outer boundaries for obtaining range information. Object models are also constructed of planar surfaces but can only exist within the bounds of the World model. A spherical

region that defines the maximum extents within which the object can exist bound each object. Rather than search all planar surfaces at each time step, each region is examined to determine whether the laser intersects it. However, intersecting the region does not guarantee an intersection of one of the object planes. A detailed examination of the environmental model can be found in **Appendix B**.

Each plane of each region intersected by the laser $\vec{f}_3[n]$ is tested to determine which is closest to the origin of the ray. Each intersected region is then compared to determine which yields an intersection $\mathbf{Y}_{i,j}[n]$ that was closest to the origin of $\vec{f}_3[n]$. In this nomenclature, i refers to the object in the environment and j refers to the planar surface within the object. If no plane was intersected then each of the planes of the World model are tested to determine which is intersected. The distance between the ray origin $\mathbf{f}_3[n]$ and the intersection point $\mathbf{Y}_{i,j}[n]$ is represented as a multiple $k_{i,j}[n]$ of the unit vector $\vec{f}_u[n]$ in the direction of $\vec{f}_3[n]$. The total distance between $\mathbf{f}_3[n]$ and $\mathbf{Y}_{i,j}[n]$ is $k_{i,j}[n]$ where $k_{i,j}[n]$ is positive. The selected intersection point is the $\mathbf{Y}_{i,j}[n]$ that results in the smallest positive $k_{i,j}[n]$ greater than zero.

Prior to determining the intersection point of the ray $\mathbf{f}_3[n]$ with a surface in the environment, the ray coordinates are transformed from the scanner frame of reference $\mathcal{R}_{RAS}$ to the environmental frame of reference $\mathcal{R}_{World}$. The ray is first rotated about the x-axis (Yaw), y-axis (Pitch) and z-axis (Roll), then translated to the appropriate location in $\mathcal{R}_{World}$. Once the point of intersection is determined then the terminated ray $[\mathbf{f}_3, \mathbf{Y}_{i,j}][n]$ is translated back into $\mathcal{R}_{RAS}$ in order to complete the simulated measurements.

3.5.1 RAS Response

The termination point $\mathbf{Y}[n]$ determined by the intersection of the laser with the environmental model is used to determine the location of the peak $\mathbf{P}[n]$ on the linear CCD array. If the location of $\mathbf{P}[n]$ is outside the CCD array surface then the peak detector returns $\tilde{P}[n] = 0$ indicating that no peak was detected.

In practice the CCD is a linear array with a resolution less than 2^{15} . The systems at NRC generally use a 512-element CCD array so this will be assumed unless otherwise stated. The CCD array returns a signal intensity profile in which the peak location is estimated to sub-pixel accuracy using linear interpolation [1]. The simulated system simplifies this process by calculating the location of the peak $\mathbf{P}[n]$ on the CCD array and converts this to a distance along the CCD $P[n]$.

Figure 3.15 shows a typical environment containing a single planar object. The position of the scanner is the bottom left of the image and the origin of the scanner frame of reference is denoted with a black dot. The line represents the laser path between the $\mathbf{f}_3[n]$ of the scanner frame of reference and the point of intersection with an object in the environment, in this case the back wall. Only the position of the scanner can be changed in the environmental simulation. It is expected that moving objects can be implemented in a future revision of the model; however, moving objects will increase the time required for each simulation and neither scanner nor object motion were needed for the purposes of this simulation. Scanner motion has been included so that the model may be used in future studies involving motion distortion.

Figure 3.15: Matlab simulation of a typical environment with a planar object. The scanner origin is shown as a black dot and the laser path as a solid line. Note that the laser origin is point f_3 , not the scanner origin.

3.6 Scanning Methods

The Random Access Scanner is currently configured to acquire range and intensity data using Lissajous and raster scan patterns. The system accepts a request to perform a complete Lissajous or raster scan of a given width and height, centred at a given location, and obtain samples at a given sample resolution. The simulated system similarly allows the user to define a scan based on sample resolution, scan width and scan height. Currently only range measurements can be obtained but this is sufficient for the purposes of this model.

3.6.1 Lissajous Scan

Lissajous scan patterns can be described by the ratio of x-axis to y-axis frequencies. Larger frequency ratios result in more complex patterns while maintaining good coverage of the field of view. The vector $\tilde{\mathbf{a}}_P[n]$ represents three 16-bit signed integer values obtained directly from the peak detector ($\tilde{P}[n]$) and the galvanometers ($\tilde{a}_\Theta[n]$ and $\tilde{a}_\Phi[n]$) respectively.

A single scan begins by calculating the appropriate voltage signal to be sent to each galvanometer. This results in a change in the position of the x- and y-axis mirrors that in turn affect the orientation of the laser and optical axis. These changes are stored as part of the RAS profile. The time step may also result in changes to the scanner's position within the environment that is stored in the environmental profile. The combination of the RAS profile and the updated environmental profile are used to determine the location of any intersection of the laser with a surface in the environment. The location of the intersection point is then used to determine the peak position $\tilde{P}[n]$ which, combined with the

original galvanometer angles $\tilde{a}_\Theta[n]$ and $\tilde{a}_\Phi[n]$, can be used to estimate the UVR and Cartesian position of the point in space.

Figure 3.16: Matlab simulation of the results of an unwrapped Lissajous scan. The top graph shows the complete peak profile and the bottom graph shows the same peak profile with the first 7 elements removed.

Figure 3.16 shows the results of a typical 256-sample (3,2)-Lissajous scan of the planar object shown in Figure 3.15. The first 7 samples of each Lissajous scan are generally set to zero as can be seen in the top graph so the Lissajous scan method employed in this simulation was configured the same way. The bottom graph shows the same scan with the first seven elements removed.

Figure 3.17: Matlab simulation of the results of a Lissajous scan in UVP-space

Figure 3.17 shows the path of the Lissajous scan in UVP coordinates. The z-axis is in units of raw peak ($\tilde{P}[n]$) values as generated by a generic, non-calibrated simulation. Smaller $\tilde{P}[n]$ values represent points closer to the scanner so an observer must picture themselves as below the (x,y)-plane of the graph and looking up. The x-axis and y-axis values are raw galvanometer values.

Figure 3.18: Matlab simulation of the results of a Lissajous scan presented as a UVP surface map

Figure 3.18 shows the surface map representation of the Lissajous scan in Figure 3.16 and Figure 3.17. Once again, the observer must picture himself or herself looking up from below the image. The scanner was oriented such that

"up" is along the negative y-axis and "right" is along the positive x-axis. The simulated surface was approximately 9-metres from the scanner and the back wall was almost 20-metres from the scanner so the surface curvature expected of a planar surface in UVP-space is not clearly visible.

3.6.2 Raster Scan

A raster scan is typically used to obtain a uniform sample of range or intensity information within a region of interest. The Random Access Scanner performs a raster scan by stepping the y-axis mirror after each x-axis scan has been completed. The x-axis scans alternate in direction so the resulting data must be unwrapped and sorted.

The position of each sample is determined solely by the x- and y-axis galvanometer positions. The peak value ($\tilde{P}[n]$) is the 16-bit signed integer value obtained from the peak detector. The scan resolution is both the number of samples per line and the number of lines scanned. The process of collecting data from each scan line is identical to the process used for the Lissajous scan but is repeated for each step of the y-axis galvanometer.

Figure 3.19 shows a surface map based on a 16 by 16 raster scan of a planar object. The orientation of the scanner and viewpoint of the observer are the same as for the Lissajous scans in the previous section. The flat surface farthest from the scanner (top of the graph) is a simulated back wall several meters behind the box. The curved surface between 30,000 and 31,000-counts is the top of the box that appears curved in the UVP system. The front of the box does not appear curved in this scan because the surface of the box is more than 9-metres from the scanner and the scanning resolution is low.

Figure 3.19: Matlab simulation of the results of a 16×16 raster scan presented as a surface map in UVP coordinates

3.7 Model Calibration

The model as described produces results that approximate those generated by a typical auto-synchronizing laser range scanner. Validation of the model requires comparing the output of the model to that generated by a real scanner. Before validation can take place the output of the model must be scaled to produce results that approximate those generated by a real scanner. The model does not perfectly duplicate the raw data generated by a real scanner because the model is an idealized system. As a result a discrepancy is expected between simulated and raw results. For simulation purposes, the contour of the scans performed by the simulator must approximate the contour of scans performed by the real system.

Calibration was performed using data collected from two laser range scanners located at the NRC. These scanners are referred to herein as the Calib scanner and the Space40 scanner. Calibration consisted of correcting the scanning mirror and peak detector constants before calculating a linear fit between the expected and observed peak values as described by MacKinnon *et al.* [62]. A detailed examination of the scanner calibration method can be found in **Appendix C**. The scanner model described in the paper by MacKinnon *et al.* uses a different method to calculate $\alpha_U[n]$ and $\mathbf{b}_3[n]$ so those results differ slightly from those presented in this thesis.

Section C.5 of **Appendix C** shows the improvement in the scanner model peak value prediction error $s_{\hat{P}|P}$ at each stage in the calibration process. The prediction error, also referred to as the standard error of the estimate, measures the observed deviation of the measured peak values (P) from those predicted by the model ($\hat{P}$). Linear fitting alone did not result in a significant reduction in

prediction error but performing calibration of the galvanometers alone resulted in a significant reduction in prediction error. Peak detector calibration and linear fitting did not result in a significant change in prediction error but have been included in the calibration process for completeness.

3.7.1 Mirror Calibration

Scanning mirror calibration involved correcting the x-axis and y-axis mirror angles β_x and β_y respectively, setting the rotational limits of each mirror, and determining the transformation function from the actual mirror angles $\Theta(t)$ and $\Phi(t)$ to galvanometer values $\tilde{a}_\Theta[n]$ and $\tilde{a}_\Phi[n]$ respectively for each scanner being modelled. For each scanner measurements were obtained at the maximum and minimum galvanometer rotations, and each measurement consisted of UVP values and matching Cartesian coordinates. The Cartesian coordinates were not ground-truth measurements but were generated using a UVP-to-Cartesian transformation developed by the NRC. The discrepancy between simulated and measured results was expected to be significantly larger than the error between ground-truth and calibrated data so calibrated results were deemed sufficient.

MacKinnon *et al.* [62] performed galvanometer calibrations assuming the angular deflection would be 30° and that both β_x and β_y would be $\pi/4$. In practice these values may vary slightly from the expected values. A better approach is to determine β_x , β_y and the maximum angular deflections from the measured data.

Scanning mirror correction was performed for the y-axis mirror followed by the x-axis mirror. From (3.1) and (3.4) it can be seen that the y-position of a point in the environment is dependent only upon the y-axis mirror position $\Phi[n]$ and the distance to the point along the z-axis $Y_z[n]$. Specifically, given a known

distance along the z-axis $\mathbf{Y}_z[n]$ to a point in the environment

$$\hat{y}[n] = (\mathbf{Y}_z[n] - \hat{f}_{z3}[n])\tan(\hat{V}[n]) + \hat{f}_{y3}[n] \quad (3.21)$$

where

$$\hat{V}[n] = 2\beta_y + 2\hat{\Phi}[n] - \pi/2, \quad (3.22)$$

$$\hat{f}_{y3}[n] = h_y - h_z \tan(\hat{\Phi}[n] + \beta_y) \quad (3.23)$$

and $\hat{f}_{z3}[n] = 0$ from (3.1). This assumes h_y , h_z and β_y are known and are correct. This also assumes that the transformation function from $\hat{a}_\Phi[n]$ to $\hat{\Phi}[n]$ is known. However, β_y and the transformation function are not known so the first step is to determine $\hat{\beta}_y$ as an estimator for β_y .

The first stage of the mirror calibration is to determine the $\hat{\Phi}_{min}$ and $\hat{\Phi}_{max}$ values that correspond to the minimum and maximum mirror rotations using the iterative search routine. The measured distance along the y-axis $\mathbf{Y}_y[n]$ to a point $\mathbf{Y}[n]$ is known so $\hat{\Phi}[n]$ can be estimated and iteratively adjusted to minimize the error

$$y_{err} = \mathbf{Y}_y[n] - \hat{y}[n]. \quad (3.24)$$

A new mirror angle $\hat{\beta}_y$ is then obtained using

$$\hat{\beta}_y = (\beta_y + \Phi_{max} - \Phi_{min})/2 \quad (3.25)$$

where Φ_{max} and Φ_{min} were obtained using the iterative search technique. The resulting new mirror angle $\hat{\beta}_y$ is the point equidistant between the maximum and minimum mirror angles. The maximum y-mirror rotation is then defined as

$1/2$ of the total angular deflection between the minimum and maximum angular positions. This is obtained using

$$\hat{\Phi}_{max} = (\hat{\Phi}'_{max} - \hat{\Phi}'_{min})/2. \quad (3.26)$$

The maximum and minimum galvanometer readings $\tilde{a}_{\Phi,max}$ and $\tilde{a}_{\Phi,min}$ are known so the y-mirror rotation can be estimated using the ratio

$$\frac{\tilde{a}_{\Phi}[n] - \tilde{a}_{\Phi,min}}{\tilde{a}_{\Phi,max} - \tilde{a}_{\Phi,min}} = \frac{\hat{\Phi}[n] - (-\hat{\Phi}_{max})}{\hat{\Phi}_{max} - (-\hat{\Phi}_{max})}. \quad (3.27)$$

The term on the left is the ratio of the angular distance between the minimum angular deflection and the target deflection to the total angular deflection measured in galvanometer units. The term on the right is the same ratio expressed in radians. Solving for the target angular deflection in radians yields

$$\hat{\Phi}[n] = 2 \hat{\Phi}_{max} \frac{\tilde{a}_{\Phi}[n] - \tilde{a}_{\Phi,min}}{\tilde{a}_{\Phi,max} - \tilde{a}_{\Phi,min}} - 1. \quad (3.28)$$

The x-mirror rotation can be found using a procedure similar to the one just described. However, it is dependant on both the y-axis mirror and x-axis mirror rotation so can not be determined until the y-axis mirror constants have been determined. Specifically, given a known distance along the z-axis $\mathbf{Y}_z[n]$ to a point in the environment

$$\hat{x}[n] = (\mathbf{Y}_z[n] - \hat{f}_{z3}[n])\tan(\hat{U}[n]) + \hat{f}_{x3}[n] \quad (3.29)$$

where

$$\hat{U}[n] = \pi - 2\beta_{out} - 2\beta_x - 2\hat{\Theta}[n], \quad (3.30)$$

$$\begin{aligned} \hat{f}_{x3}[n] = & h_x + h_x \frac{\sin(\hat{\varphi}[n]) \cos(\beta_{out})}{\sin(\beta_{out} - \hat{\varphi}[n])} \\ & + h_y \tan(\hat{U}[n]) - h_z \tan(\hat{\Phi}[n] + \hat{\beta}_y) \tan(\hat{U}[n]) \end{aligned} \quad (3.31)$$

where

$$\hat{\varphi}[n] = \pi/2 - 2\beta_x - 2\hat{\Theta}[n] \quad (3.32)$$

and $\hat{f}_{z3}[n] = 0$ from (3.1). The mirror angle β_x is calculated as for β_y and the mirror extents are similarly determined. The transformation function for the x-axis mirror proceeds as in (3.28) so it becomes

$$\hat{\Theta}[n] = 2\hat{\Theta}_{max} \frac{\tilde{a}_{\Theta}[n] - \tilde{a}_{\Theta,min}}{\tilde{a}_{\Theta,max} - \tilde{a}_{\Theta,min}} - 1. \quad (3.33)$$

It should be noted that errors in the y-axis mirror calibration result in additional errors in the x-axis mirror calibration. As a result the x-axis mirror calibration should be less accurate than the y-axis mirror calibration.

3.7.2 Peak Detector Calibration

Peak detector calibration involves determining the length of the CCD array, L_{CCD} , and the offset of the edge of the CCD array from the z-axis, L_{offset} . For each scanner, measurements were obtained at the maximum and minimum operational range, and each measurement consisted of UVP values and matching Cartesian coordinates. The Cartesian coordinates were not ground-truth measurements but were generated using a UVP-to-Cartesian transformation developed by the NRC. The discrepancy between simulated and measured results was expected to be significantly larger than the error between ground-truth and calibrated data so calibrated results were deemed sufficient. The calibration process described by MacKinnon *et al.* [62] did not include a peak detector calibration so their results are expected to differ from those presented in this thesis.

The peak detector constants are corrected by calculating where the image would fall on the CCD given a known point of intersection in the environment

$\mathbf{Y}[n]$ and galvanometer readings $(\tilde{a}_\Theta[n], \tilde{a}_\Phi[n])$. This assumes that the x-axis and y-axis mirrors have already been calibrated. Errors inherent in the x-axis and y-axis mirror calibrations will limit the accuracy of the peak detector calibration.

Given a known distance along the z-axis to a point in the environment

$$\hat{L}[n] = f \frac{\sin(\hat{\alpha}_U[n])}{\sin(\hat{P}_\alpha[n])} \quad (3.34)$$

where

$$\hat{P}_\alpha[n] = \pi/2 - \hat{\alpha}_U[n] - \beta_{CCD} \quad (3.35)$$

and $\hat{\alpha}_U[n]$ is obtained using methods previously described. This value becomes

$$\hat{P}[n] = L_{CCD} + L_{offset} \hat{L}[n] \quad (3.36)$$

which would be the same result as obtained using (3.15).

The distances $\hat{L}_{min}$ and $\hat{L}_{max}$ from the z-axis to the point of intersection with the CCD-plane at the minimum and maximum ranges can be determined using (3.34) based on measured values obtained from each scanner. The measured quantized peak values at these ranges can then be used to calculate the length of the CCD array using

$$\hat{L}_{CCD} = 2^{15} \frac{\hat{L}_{max} - \hat{L}_{min}}{\tilde{P}_{max} - \tilde{P}_{min}}. \quad (3.37)$$

The CCD offset can then be estimated using

$$\hat{L}_{offset} = \hat{L}_{min} + \hat{L}_{CCD} \left(\frac{\tilde{P}_{min}}{2^{15}} - 1 \right). \quad (3.38)$$

If all non-linearities were identified in the model and calibration errors were negligible then the calibrated scanner should generate results closely approximating those generated using the real scanner. However, estimation errors exist in

both the mirror and peak detector calibrations. Moreover, not all non-linearities have been included in the model. A further step is required to reduce the discrepancy between simulated and measured results.

3.7.3 Linear Fit

The final stage of the calibration process involves fitting the simulated peak value to the measured peak value for a series of points between the minimum and maximum operational range. Linear regression is used to develop a linear function using simulated peak and galvanometer readings to predict a measured peak value as described by MacKinnon *et al.* [62]. The measured and simulated peak values were not linearly related but were close enough within the operational range that a linear fit provided good results. Ideally the measured and simulated peak values should be linearly related but not all non-linearities have been identified in the model. However, the model fit was close enough for development work.

Matched Cartesian and UVP measurements were collected for each scanner between 1-metre and 10-metres. Linear regression was used to predict the measured peak $\tilde{P}'$ based on simulated UVP measurements, specifically $\tilde{P}$, $\tilde{a}_\Theta$ and $\tilde{a}_\Phi$. A linear correction is then obtained using the regression

$$\tilde{P}'[n] = b_0 + b_1 \tilde{P}[n] + b_2 \tilde{a}_\Theta[n] + b_3 \tilde{a}_\Phi[n] \quad (3.39)$$

where b_0 , b_1 , b_2 and b_3 are regression coefficients which will result in a best fit between the simulated values and the measured peak $\tilde{P}'$. This regression equation explained in section C.4 of **Appendix C**.

3.8 Model Demonstration

The ability of the model to produce results similar to those obtainable from a real scanner was demonstrated by collecting sample data from a calibrated working laser range scanner and comparing to the results generated using the simulated system. This required replicating the scanner environment as a simulated environmental model.

Figure 3.20: Configuration of the Space40 validation test

The Space 40 scanner was selected due to immediate availability to demonstrate the accuracy of the calibrated model. A single box 0.58-metres long, 0.58-metres wide and 0.42-metres high was placed 8.47-metres from the scanner. The centre of rotation of the scanner was estimated to be 1.64-metres from the floor. Four scans were performed representing the front surface (Box 1), bottom edge (Box 2), top edge (Box 3) and upper-right corner (Box 4) of the box. Figure 3.20 shows the physical layout of the test situation. Each scan was repeated 10 times so that interscan variance could be examined and compared to the simulated system.

The environment was simulated using the environmental modelling system

Figure 3.21: Simulated environment for Space 40 calibration test. The black dot represents the origin of the scanner frame and the solid line represents the laser path from f_3 to its intersection with a surface on the object.

and similar scans were performed using the simulated scanner. The position of the scanner in the simulated environment was adjusted so that the Cartesian measurements obtained using the simulation would closely match the Cartesian measurements obtained using the Space 40 scanner. Figure 3.21 shows the position of the box in the simulated environment. A dot in middle of the image represents the position of the laser scanner origin in the environment and the line represents the laser path starting at point f_3 in the laser frame of reference. The box appears thin because each of the axes use a different scale so that the scanner position, laser, and the box could all be seen simultaneously.

Figure 3.22 shows the Cartesian (left) and UV (right) representations of the front surface of the box. The top images display the results of a single scan by the actual laser range scanner. The bottom images show the same scan performed using the simulated system. Slight differences are expected between the images due to slight discrepancies between the simulated and actual scanner position as well as real measurement errors. The differences between the test and simulated UVP are made more pronounced by the scale of the peak axis.

The measured Cartesian data in Figure 3.22 features planar surface with a vertical drop of 100-millimetres between 1.2 and 1.55 metres along the x-axis. In the simulated system the surface is flat with respect to the scanner so the simulated surface deviates from the simulated surface by ± 50 -CCD units. At 9-metres a difference of 90-millimetres represents 16-peak detector counts in the UVP coordinate system. As a result, the discrepancy in real and simulated UVP readings should vary from +8-peak detector units at an X-galvanometer count of -3800 to -8-peak detector units at and X-galvanometer count of -3150. This can be seen as a phase error of approximately 16-peak detector counts in Figure 3.23.

Figure 3.22: Cartesian and UVP representations of Box 1 test using Space 40. The top graphs indicate measured results and the bottom indicate simulated results. Graphs on the left use the Cartesian coordinate system while graphs on the right use the UVP coordinate system.

Figure 3.23: Peak error of Box 1 test using Space 40. The top graph shows results obtained from a real scanner (measured, dashed line) and those obtained from the simulation system (simulated, solid line). The bottom graph shows the absolute error between simulated and measured results.

Figure 3.23 shows the errors in measurement between actual and simulated scan data. The top graph shows the measured (dotted line) and simulated (solid line) measurements obtained. The discrepancy between simulated and measured results at this range is clearly visible. The bottom graph shows the absolute difference between the simulated and measured results. The simulated and real peak values appear to be out of phase due to discrepancy between the simulated and measured positions of the scanner and box discussed in the previous paragraph.

Figure 3.24: Cartesian and UV representations of Box 2 test using Space 40. The top graphs indicate measured results and the bottom indicate simulated results. Graphs on the left use the Cartesian coordinate system while graphs on the right use the UVP coordinate system.

Figure 3.25: Peak error of Box 2 test using Space 40. The top graph shows results obtained from a real scanner (measured, dashed line) and those obtained from the simulation system (simulated, solid line). The bottom graph shows the absolute error between simulated and measured results.

Figure 3.24 shows the Cartesian (left) and UV (right) representations of the bottom edge of the box. The simulated system appears to generate results that follow the same pattern as those obtained using the real system. The results obtained using the real system are less uniform because the object is not perfectly smooth and symmetric, unlike the simulated box.

Figure 3.25 shows the errors in measurement between actual and simulated

scan data. Each of the peaks in the top graph is out of phase due to the phase error discussed previously, and because the surface of the real box was slightly curved whereas the simulated box surface was perfectly smooth. Simulation errors vary between 64 and 200-counts which is acceptable. A similar comparison reported by MacKinnon *et al.* [62] found a discrepancy between measured and simulated peak values of 0.3% of the measured peak value. The discrepancy shown in Figure 3.25 has a maximum discrepancy of 0.48%; however, the simulated scanner position in [62] was based strictly on measurements of the scanner position. In the current demonstration the position of the scanner was adjusted so that the discrepancy between simulated and measured Cartesian values would be minimized.

Figure 3.26 shows the Cartesian (left) and UV (right) representations of the top edge of the box. The simulated system appears to generate results that follow the same pattern as those obtained using the real system.

Figure 3.27 shows the errors in measurement between actual and simulated scan data. The curvature previously noted is now visible as a slight convexity of the valleys in the measured data. Simulation errors are within 200-counts as in the previous simulations.

Figure 3.28 shows the Cartesian (left) and UV (right) representations of the top-right corner of the box. Due to the complexity of the surface the similarity between the simulated and measured results is difficult to judge by eye.

Figure 3.29 shows the errors in measurement between actual and simulated scan data. Inaccuracies in positioning the simulated laser scanner to closely match the position of the real scanner result in a slight shift in the simulated results; however, the simulated system closely approximates the results of the real system. Generally simulation errors are less than 200-counts except where they are caused

Figure 3.26: Cartesian and UV representations of Box 3 test using Space 40. The top graphs indicate measured results and the bottom indicate simulated results. Graphs on the left use the Cartesian coordinate system while graphs on the right use the UVP coordinate system.

Figure 3.27: Peak error of Box 3 test using Space 40. The top graph shows results obtained from a real scanner (measured, dashed line) and those obtained from the simulation system (simulated, solid line). The bottom graph shows the absolute error between simulated and measured results.

Figure 3.28: Cartesian and UV representations of Box 4 test using Space 40. The top graphs indicate measured results and the bottom indicate simulated results. Graphs on the left use the Cartesian coordinate system while graphs on the right use the UVP coordinate system.

Figure 3.29: Peak error of Box 4 test using Space 40. The top graph shows results obtained from a real scanner (measured, dashed line) and those obtained from the simulation system (simulated, solid line). The bottom graph shows the absolute error between simulated and measured results.

by inaccuracies in positioning the simulated laser.

3.9 Summary

A model of the laser range scanner was developed based on knowledge of the operation of the system and published specifications. Quantization error and small-scale noise was added to the system according to published specifications. Raster and Lissajous scanning systems were designed to utilize the scanner and environmental models to generate results that would approximate those which could be obtained using a real scanner. A calibration model was generated for each scanner used in this study so that simulated results would more closely approximate those obtained using a real system. Data were collected using a calibrated scanner and compared to results obtained using the simulated system. The results obtained using the simulated system were acceptably close to those obtained using the real system indicating that the model would be sufficient for this study.

Chapter 4

Edge Mapping

Edge mapping involves determining the spatial location of range discontinuities as seen from the perspective of the scanner. The first stage in developing an edge map is the accurate detection of edges given a typical Lissajous scan under expected noise conditions. Typical Lissajous scans must therefore be examined to develop a noise model that can then be applied to testing the accuracy of potential edge detection methods. The edge metrics that are used to evaluate the selected edge detection algorithms is determined once the noise model has been developed. The second stage is to combine the detected edges with range and position information to orient them in the environment. A goal of this project is to perform edge detection in real-time for control so the edge detection algorithm must complete the processing of one data set before the next one arrives.

4.1 Proposed Edge Detection Method

Oliver *et al.* [58] described a modification to the PoD filter in which the averages are compared using a t-test so that window size and observed Gaussian noise level can be taken into account. They also proposed using a normalized difference estimator (NDE) but stated that a choice of normalization operators exists. The strength of normalizing the PoD is that the resulting measure is independent of scale [58]. In this project the PoD filter output is normalized to the maximum PoD value such that the peak heights rather than the signal values are normalized. In the trivial case, which occurs when the maximum PoD is zero, all signal elements are replaced with zero. The minimum PoD value at which the signal is reduced to zero can be adjusted to eliminate low-level noise.

4.1.1 Evaluation Methodology

Heath *et al.* [37] noted that few comparisons of edge detection methods or their results used statistical methods such as analysis of variance (ANOVA). Most presented tabulated results of the values of different edge metrics without attempting to compare them analytically [37]. Others depend upon graphical methods, one example being the ROC curves employed by Shin *et al.* [36]. In this study results are statistically validated where possible.

For purposes of analysis and discussion edge detection method evaluation is divided into three phases. In Phase I each edge detector is developed and their performance is evaluated using simple step edges and linear surfaces. Edge detectors may be selected or eliminated based on their ability to detect step edges under expected noise conditions. In Phase II the edge detectors are used to scan

simple surfaces under expected noise conditions. Recall that a flat surface is represented as an arc using UVP measurements. It is expected that further edge detector optimization would be required. Moreover, some edge detectors may detect false edges or be unable to detect true edges where many surfaces may be highly curved. In Phase III edge detector performance is then examined using a real scanner to verify that edge detection is being performed in real-time.

Edge detection involves performing noise smoothing, edge enhancement and edge localization [2, p.70]. A comparison of noise smoothing techniques has been performed to determine which methods resulted in a minimum difference between ideal and post-filtered signal. A series of edge enhancement methods were then examined in combination with edge localization techniques to determine which provided the best results based on the selected edge metrics. In this study edges detected using the ideal range model, or ground-truth model to which it is sometimes referred [31], were defined as true edges [29, p.491] [31] or ideal edges [27]. Edges detected using the test range model were defined as detected edges.

Much of the work presented here is based on results published by MacKinnon *et al.* [63]. In some cases this chapter provides a more detailed explanation of the results published in [63]. In other cases the method deviates from that published in [63].

4.2 Phase I: Edge Detector Development

Edge detection involves noise reduction, edge enhancement and edge localization [2, pp.69-70] [64, pp.145-146]. Optimization of the process at each step should provide assurance that a given methodology will maximize the accuracy and pre-

cision of edge detection under real environmental conditions. A source of error that must be considered is speckle error which can be modelled as a Gaussian white noise process [5]. The position error of each galvanometer must also be considered when developing a noise model [7]. In both cases the error appears as variation in the reading obtained from the instrument. An examination of the internal error sources was beyond the scope of this project so will be assumed to be small enough to be ignored.

Prior to evaluating the edge enhancement methods a series of filters must be compared to determine which will perform best given the selected noise model. Based on previous experience with data generated by the laser range scanners at NCR, it was expected that peak detector and galvanometer data would contain additive noise and aperiodic spikes or outliers. A sample of data generated by the Calib and Space 40 scanners was examined in **Appendix E** and it was determined the additive noise could be modelled as a zero-mean Gaussian noise process. Aperiodic spikes were observed and have been classified here as zero-spikes and non-zero spikes. The former refers to spikes with a value of zero and the latter refers to spikes with non-zero values. Spikes were identified by obtaining a series of scans and examining the samples associated with each scan element to determine if any are outliers. The frequency with which zero and non-zero spikes occur is referred to as the spike rate.

Table 4.1 shows the maximum observed sample standard deviations based on ten 256-element scans from the Calib and Space 40 scanners. The Space40 scanner has the largest sample standard deviation; this value is used in this study to represent the worst additive noise levels to which the edge detection algorithm might be exposed. Table 4.2 shows the frequency of occurrence of zero and non-

Table 4.1: Observed peak detector galvanometer standard deviations from **Appendix E**. Peak standard deviations are in units of peak detector counts and galvanometer standard deviations are in units of galvanometer counts.

Source	Calib (counts)	Space40 (counts)
Peak Standard Deviations (s_P)	1.946655	5.164816
X-galvanometer Standard Deviations (s_Θ)	2.480937	3.292068
Y-galvanometer Standard Deviations (s_Φ)	1.413150	2.832741

Table 4.2: Observed zero and non-zero spike rates examined in **Appendix E**. All spike rates represent the fraction of all samples that were identified as outliers. Zero spikes are outliers with a value of zero and non-zero spikes are outliers with non-zero values.

Spike type	Calib	Space40
Zero Spike Rate	0.009766	0.045312
Non-zero Spike Rate	0.000000	0.001172

zero spikes observed in the same data sets obtained from the Calib and Space 40 scanners. Rather than simulate zero and non-zero spikes using the environmental model the highest observed zero and non-zero spike rates based on the Calib and Space40 data were used as examples of possible spike rates. This meant that a simple environmental model could be employed rather than developing a more complex but comprehensive model. Details of the noise analysis can be found in **Appendix E**.

The standard deviation of the peak detector data obtained from the Space40 scanner was used as the standard deviation of the additive noise process. MacKinnon *et al.* [63] examined the observed noise of the Calib scanner and simulated peak detector noise using the standard error of the normalized peak values. The standard error had been selected because it was an unbiased estimator of the population standard deviation [20, pp.303-306]. The standard deviation was selected for this study because it is larger than the standard error so would be more indicative of significant additive noise. If the edge detection algorithm performed acceptably under these noise conditions then it should perform well under normal conditions.

Zero and non-zero spikes were generated according to the zero and non-zero spike rates observed in the peak detector data obtained from the Space40 scanner. The zero-spike rate is the frequency at which a signal is not sufficient to be detected so the peak detector returns a value of zero. These are generally the result of occlusion of the laser spot (as in Figure 4.2) or a surface beyond the range of the scanner. The non-zero spike rate is the rate at which outlier peak signals are detected. These are generally a result of re-reflection causing a second stronger peak (as in Figure 4.1) or a highly reflective surface saturating the CCD array.

Figure 4.1: One possible source of non-zero spikes is signal re-reflection

Figure 4.2: One possible source of zero spikes is occlusion

Figure 4.1 demonstrates the difference between expected and actual signal position on the CCD. These values can be positive indicating an incorrect peak location or negative indicating CCD saturation or multiple peaks of similar height [33]. Figure 4.2 shows one source of zero spikes. The environmental model does not simulate occlusion of the laser termination point so zero values are generally not generated by the simulation. Zero and non-zero spikes in a system where laser intensity and integration time have been appropriately selected can generally be attributed to the geometry of the objects being scanned. From the perspective of the edge detection algorithm they appear as aperiodic impulses so are simulated as random zero and non-zero impulses.

4.2.1 Filter Evaluation

The averaging filter, Gaussian filter, median filter and an iterative median filter were examined as commonly applied noise filters. Zou [53] compared filters by calculating the root mean squared (RMS) error between the filtered results and the expected or ideal output. Rostampour and Reeves [44] evaluated the effectiveness of median filters using the normalized mean squared difference. Pratt [29, p.271] used the sum of squared residuals used to evaluate curve-fitting in regression analysis. MacKinnon *et al.* [63] used the absolute value of the deviation that was obtained by calculating the average of the absolute deviation based on 50 repetitions. Trucco and Verri [2, pp.308-309] referred to this measure as the mean absolute error (MAE) while MacKinnon *et al.* [63] referred to it as the average absolute deviation (AAD). Given a vector $\mathbf{x}_T$ representing an ideal Lissajous scan

and a vector $\mathbf{x}$ representing the same scan with noise, the AAD is found by

$$AAD = \frac{1}{N} \sum_{i=0}^{N-1} |x[i] - x_T[i]| \quad (4.1)$$

where N is the number of elements in the vector [2, p.308].

Each of the filters were examined to determine the optimum filter window size given a 1024-element data set saturated with Gaussian noise and spikes based on the Space40 error model. The AAD based on 10 repetitions was used to determine the expected average deviation at that window size. Only odd filter window lengths are used in this study. Smaller AAD values indicate a better fit between the filtered and ideal signals. The average processing time required by Matlab on a Pentium-III workstation running Windows 2000 was also determined for each window length. Within each filter type, the same data was used for all filter lengths so that differences in performance among window sizes would be strictly due to the filter size and not random variation between data sets.

Figure 4.3, Figure 4.4 and Figure 4.5 show the effect of increasing window size on the mean AAD value based on 10 repetitions. The top graph in each figure shows the change in mean AAD value with increasing window size. The window size displayed on the x-axis of both the top and bottom graphs is the actual number of elements in the filter window. The bottom graph in each figure shows the average processing time to complete the filter operation for each window size. There was a significant visible reduction in AAD for the Gaussian and Median filters between window sizes of 3 and 7 so the 3-element and 5-element window sizes were removed. In this way the AAD reduction for window sizes of 7-elements or greater can be clearly seen.

The AAD results in Figure 4.5 are considerably better than those seen in Figure

Figure 4.3: Mean AAD (top) and processing time (bottom) versus window size for averaging filter ($N=10$).

Figure 4.4: Mean AAD (top) and processing time (bottom) versus window size for Gaussian filter ($N=10$)

Figure 4.5: Mean AAD (top) and processing time (bottom) versus window size for Median filter ($N=10$)

4.3 and Figure 4.4. An increase in window size in the Gaussian filter reduces the AAD to only 65.5 while the same increase in averaging filter window size reduces the AAD to 59. The median filter displays the smallest AAD; however, processing time is higher than for the averaging and Gaussian filters. The averaging and Gaussian filters are eliminated as noise filters due to their poor AAD scores.

Figure 4.6: Mean processing times in milliseconds for median (top) and optimized median (bottom) filters on Calib scanner (N=10).

The scanner operates at a maximum sampling rate of 15-KHz according to [12] and can acquire between 128 and 4096 samples per scan. This means that the system is ideally capable of performing a maximum of $15000/128 = 117.19$ scans

per second so the minimum time required to perform a scan is $1000/117.19 = 9$ -milliseconds per scan. The process of acquiring a 128-element scan and processing it to generate an edge map must be completed in less than 9-milliseconds. In this study 1024-element scans are used so each scan should be completed in require $9(1024/128) = 72$ -milliseconds.

Visual inspection of results obtained from median filters with window sizes between 7-element and 21-elements revealed that there was little visible difference in results for window sizes greater than 9-elements. The top graph of Figure 4.6 shows the results of processing a 1024-element array with random noise added on a 500-MHz Pentium III workstation running the QNX real-time operating system using a median filter. The median filter algorithm can be optimized by performing the median filter once for the initial data window and saving it in an array, then applying a swapping algorithm to the sorted array for the remaining data. The swapping algorithm involves moving the window by a single element, swapping the last element of the previous window out of the sorted array, and inserting the first element of the new window into the sorted array such that the array remains sorted. The optimized median filter generates the same results as the non-optimized median filter but requires substantially less time to complete. The bottom graph shows that processing times required for the optimized median filter are significantly less than those required for the non-optimized median filter. An optimized filter of more than 13-elements requires noticably more time to complete so for the remainder of this study a 13-element optimized median filter is used. Both the optimized and non-optimized median filters required significantly less than 72-milliseconds to process a 1024-element array so neither violate the real-time requirement.

Figure 4.7: Peak values versus range at short (top) and medium (bottom) range for the Space 40 scanner

Each of the filters was compared to determine which method would minimize both Gaussian noise and spikes. The best method would display the smallest AAD value which would mean that the filtered signal that varied the least from the ideal signal. The first 512-elements of a 1024-element array were given a value of 22309 which represented the peak value resulting from a surface approximately 2-metres from the Space 40 scanner. Figure 4.7 shows the peak values as a function of range from which the approximation for the peak value for the 2-metre surface was taken. The remaining elements were given a peak height of $22309 + 2^4$ to represent the minimum detectable edge height. The resulting 1024-element array is referred to as the ideal signal array. Gaussian noise and spike rates using the Space 40 noise model were added to the ideal signal array to produce a test signal array. Each of the filters was applied to the same noisy data set and the AAD was calculated based on 10 repetitions. Each filter was also tested using the QNX real-time operating system on a 500-MHz Pentium III processor with 256-Mb RAM to determine the average time required to process a 1024-element signal containing random noise based on the Space 40 noise model.

Table 4.3: Comparison of the Average Absolute Deviation for each noise filter based on a 1024-element profile (N=10)

Filter type	Window Size (number of elements)	AAD (N=10)	Time (msec) (N=10)
Optimized Median	13	1.65 ± 0.047	2.13 ± 0.19
Iterated Median	3	6.47 ± 1.053	1.85 ± 0.38

Figure 4.8 shows the results of comparing the iterated median and median

Figure 4.8: Comparison of iterated median (bottom) and 13-element median (top)

filters. The results have been summarized in Table 4.3. A lower AAD score is preferred so the median filter is chosen based upon its ability to reconstruct the ideal signal from noisy data. Processing times are shown in the 1st column and represent the mean time required to process a 1024-element array with random noise added on a 500-MHz Pentium III workstation running the QNX real-time operating system. There is little difference in processing times between iterated median and median filters so only the AAD score is used as the selection criteria. Optimization of the median filter eliminated the difference in processing times between the median and iterated median filter so processing time was no longer a relevant selection criteria between these two filters. Moreover, both filters require significantly less than 72-milliseconds to process a 1024-element array so neither violate the real-time requirement. For the remainder of this study the optimized median filter is used.

4.2.2 Edge Enhancement Evaluation

Edge enhancement involves filtering the data such that high-frequency transitions such as step edges can be detected easily. Ideally edge enhancement should not result in a significant loss of edge location information that will be needed in the next step. Moreover, the enhanced edge must be visible over noise that will also be enhanced in this step. The optimal edge enhancement procedure should have a peak that coincides with the edge transition region and should not be unduly affected by the slope of the surfaces around the edge [2, p.70]. The effect of edges other than step edges on edge enhancement should also be examined. Finally, adjustments may need to be made to the noise filtration step to assist in edge enhancement, particularly in screening out high-frequency noise. In this section

the 3-point derivative method (Der) and a PoD method are compared.

Figure 4.9: Results of applying a 3-point Der filter (middle) and 3-point PoD filter (bottom) to a noisy step edge. The top graph shows the ideal step edge with a height of 16-counts ($1/4$ -CCD pixel) as a solid line and the same signal with noise added superimposed as a dashed line. A 13-element median filter was applied to the noise signal prior applying the edge enhancement filter.

Figure 4.9 shows the results of applying a 3-element PoD edge enhancement filter (middle graph) and a 3-point Der filter to noisy step data that has been prefiltered using a 13-element median filter selected in the previous section. The height of the ideal edge is 16-peak counts or $1/4$ -CCD pixel width. There is no

discernable difference between the edge enhancement filters, nor are the peaks corresponding to the two step edges visible. As a result either expected minimum detectable edge height must be increased, or the edge enhancement window needs to be increased, or both.

Figure 4.10 shows the effect of increasing the window size of the PoD filter under noise conditions. The top graph shows the ideal surface consisting of two step edges. The height of the edge in the ideal array is 16-peak counts or $1/4$ -CCD pixel width. The y-axis label indicates the number of elements in the window used in that graph. As the window size increases the edge resolution also increases without unduly increasing the effect of signal noise. Note that the 13-element median filter was used to filter the noisy data prior to applying the PoD filter. The 11-element PoD filter appears to produce the most clear defined peak for each edge and is easily distinguished from peaks resulting from noise. As a result the 11-element PoD filter size used in this study. The middle graph of Figure 4.9 is different than the graph for a 3-element PoD filter in Figure 4.10 because of random variation between data sets. The same noise variance and spike frequencies were used for each graph but a new simulated peak profile was generated for each.

Figure 4.11 shows the resolution of the PoD filter at 1-metre, 2-metres, 5-metres and 10-metres. The ideal surface is the same as the one displayed in the top graph of Figure 4.10 and is used in the remainder of this section. Figure 4.7 was used to determine that for the Space 40 scanner the peak value at 1-metre is 14,357, at 2-metres is 22,309, at 5-metres is 27,084 and at 10-metres is 28,675. These values were used to define the distance in UVP space to the near surface, and the far surface was the base plus 64 peak units. Figure 4.12 shows the minimum detectable edge heights at $1/4$ -CCD pixel sensitivity without noise for the Space

Figure 4.10: Effect of increasing window size on the resolution of the Product of Difference edge enhancement filter. The top graph shows the ideal signal as a solid line. The remaining three graphs show the results of applying a PoD filter using the window size indicated on the y-axis of each graph after. Prior to applying the PoD filter noise was added and the signal filtered using a 13-element median filter.

Figure 4.11: Effect of increasing range on the resolution of the 11-element PoD edge enhancement filter

Figure 4.12: Edge heights versus range at $1/4$ -CCD pixel resolution at short (top) and medium (bottom) range for the Space 40 scanner

40 scanner. At 1-metre the resolution would be 1-millimetres, at 2-metres the resolution would be 4-millimetres, at 5-metres it would be 25-millimetres, and at 10-metres it would be 102-millimetres.

Figure 4.13: Maximum edge enhancement values generated using an 11-element PoD edge enhancement filter of noisy data derived from peak values associated with 1-metre, 2-metre, 5-metre and 10-metre range values

Figure 4.13 and Figure 4.14 show the results of performing a PoD and Der edge enhancement respectively of noisy data generated using peak values associated with 1-metre, 2-metre, 5-metre and 10-metre range values. The plots represent the maximum values obtained for each scan element based on ten rep-

Figure 4.14: Maximum edge enhancement values generated using an 3-point Der edge enhancement filter of noisy data derived from peak values associated with 1-metre, 2-metre, 5-metre and 10-metre range values

etitions. Defining a minimum threshold level $E_{threshold}$ for PoD and Der results can eliminate edge enhancement peaks resulting from additive noise and spikes. The minimum $E_{threshold}$ that would eliminate all the values seen in Figure 4.13 would be 10 and the minimum $E_{threshold}$ that would eliminate all the values seen in Figure 4.14 would be 6. These values represent the minimum threshold level require to eliminate peak enhancement noise due to surface curvature and can be increased if necessary to improve the results of the edge detection process.

Figure 4.15: Effect of increasing range on the resolution of the 3-point Der edge enhancement filter

The resolution of the 3-point Der filter was examined to determine if it could

detect edges under noise conditions. Figure 4.15 shows that the 3-point Der filter appears to generate discernable peaks that correspond to the two step edges in the ideal signal seen in the top graph of Figure 4.10 so is retained as a potential alternative to the PoD edge enhancement method.

Figure 4.16: Effect of increasing edge height on the resolution of the PoD edge enhancement filter

Figure 4.16 and Figure 4.17 show the effect of increasing edge height on the resolution of the PoD and Der filters respectively. In both cases the resolution of the edge enhancement routine increases as edge height increases. The edge detection routine should therefore be most effective in detecting the edges of objects

Figure 4.17: Effect of increasing edge height on the resolution of the 3-point Der edge enhancement filter

against a surface close to the maximum range of the scanner. The nearer the object is to the scanner compared to the background surface the clearer the edges should be.

Edge localization can be performed easily by thresholding the results of the PoD or Der filter when the surfaces are flat in UVP space. It was determined in this section that $E_{threshold} = 10$ and $E_{threshold} = 6$ would be effective in removing PoD and Der peaks respectively that are a result of additive noise and spikes. However, surfaces that are flat in Cartesian space are curved in UVP space so the effect of detecting edges of simple surfaces in UVP space must be examined before a useful edge localization routine can be developed.

4.3 Phase II: Simple Surfaces

Edge localization methods are used to reduce a signal in which the edges have been enhanced such that a single pulse indicates the location of the edge. When the surfaces are flat then simple thresholding can be used to isolate peaks corresponding to edges enhanced by a PoD or Der filter. However, in UVP space flat surfaces rarely occur so an edge localization technique must be able to isolate peaks corresponding to edges from peaks corresponding to surface curvature. Moreover, the effect of noise on a curved surface is somewhat different so the threshold level may need to be adjusted to compensate. Increasing the threshold level of the PoD or Der filter can handle some of the noise but extracting peaks associated with edges from peaks associated with surface curvature requires a different approach.

Figure 4.18 shows the ridge surface used to examine the effect of simulated scanner measurements of a flat surface with a discernable edge on edge detection.

Figure 4.18: Ridge surface in UVP-space. The near surface is 9.5-metres from the scanner and the far surface is 10-metres from the scanner.

This surface was used to generate all the results presented in this section.

Figure 4.19 shows the effect of attempting to perform edge detection of the ridge surface seen in Figure 4.18 using the PoD edge enhancement filter. The top graph shows the locations in the Lissajous scan that correspond to edges crossed during the scan. The second graph from the top shows the unwrapped Lissajous scan of the flat surface after it has been filtered using a 13-element median filter. The curvature of the surface in UVP-space is clearly visible as are the sudden peak value changes corresponding to the crossed edges. It is these sudden changes that the edge detection routine is supposed to detect. The third graph from the top shows the results of filtering the data using an 11-element PoD edge enhancement filter. Notice that most of the sharp peaks correspond to the expected locations of detected edges. The bottom graph shows the results of extracting only the peak values.

Peaks can be extracted by selecting the maximum value of a moving window. Consider a symmetric window $\mathbf{w}$ centred at element k of a vector $\mathbf{x}$. If the half-width of the window is M then the window contains the elements $\{x[k-M] \dots x[k+M]\}$. Element $x[k]$ would be a peak value if $x[k] = \max\{x[k-M] \dots x[k+M]\}$. A vector $\mathbf{y}$ consisting of peak values would be generated using the relationship

$$y[k] = \begin{cases} x[k] & \text{if } x[k] = \max\{x[k-M] \dots x[k+M]\} \\ 0 & \text{otherwise} \end{cases} \quad (4.2)$$

Figure 4.20 shows the effect of range in the Lissajous scan of the ridge surface seen in Figure 4.18 where the ridge height is $1/4$ -CCD pixel width. The vertical lines indicate the expected locations of edges. Sharp peaks corresponding to known edges can only be clearly discerned for a surface 10-metres from the scanner. This

Figure 4.19: Lissajous scan followed by edge detection of the ridge surface seen in Figure 4.18. The top graph shows the expected locations of true edges as vertical lines. The second graph shows the resulting peak profile after applying a 13-element median filter. The third graph shows the results of applying an 11-element PoD filter to the data shown in the second graph. The bottom graph shows the peaks detected in the data seen in the third graph.

Figure 4.20: Effect of increasing range on the Lissajous scan of the ridge surface seen in Figure 4.18. The height of the ridge is $\frac{1}{4}$ -CCD pixel width. Solid vertical lines indicated the expected locations of true edges.

shows the detection of $1/4$ -CCD pixel width edges may not be possible.

Figure 4.21: Effect of increasing peak height on the PoD-filtered Lissajous scan of the ridge surface seen in Figure 4.18. The near surface is located 1-metre from the scanner. Solid vertical lines indicated the expected locations of true edges.

Figure 4.21 shows the effects of increasing edge height on a ridge surface with a near surface 1-metre from the scanner. The top graph shows the effect of increasing the edge height to 4-millimetres which is equivalent to 1-CCD pixel width for a surface 1-metre from the scanner. Most edges are clearly resolved when the edge height is 32-millimetres as can be seen in the bottom graph. At ranges less than 32-millimetres many edges are not clearly discernable. Figure 4.22 shows that

Figure 4.22: Effect of increasing peak height on the Der-filtered Lissajous scan of the ridge surface seen in Figure 4.18. The near surface is located 1-metre from the scanner. Solid vertical lines indicated the expected locations of true edges.

applying a derivative filter has the same result. Edge detection is not expected to achieve a $1/4$ or even 1-CCD pixel width resolution for surfaces 1-metre from the scanner but the resolution should improve with increasing range.

A close examination of Figure 4.19, Figure 4.21 and Figure 4.22 reveals that peaks associated with the expected locations of edges are narrow and tall. MacKinnon *et al.* [63] noted that peaks like these would have a small width with respect to their height. They suggested that one way to identify these peaks would be to determine the width of the peak at some threshold level and divide the width by the height of the peak between the threshold level and the top of the peak. This would also eliminate peaks that correspond to shallow ramp edges [63]. For each peak detected that is greater than the threshold the peak ratio R is calculated using

$$R = \frac{w_p/N}{E_p/E_{max}} \quad (4.3)$$

where E_p is the distance between the threshold and the top of the peak, E_{max} is the maximum peak height in the scan, w_p is the width of the peak the threshold level $E = E_{threshold}$ and N is the number of elements in the scan. If the peak ratio R is less than the ratio threshold $R_{threshold}$ then the peak is retained. This means that a modified peak vector $y'[k]$ can be generated using

$$y'[k] = \begin{cases} y[k] & \text{if } R_{threshold} > \frac{w_p[k]/N}{E_p[k]/E_{max}} \\ 0 & \text{otherwise} \end{cases} \quad (4.4)$$

where $E_p[k]$ is the distance between the threshold and the top of the peak at element k , and $w_p[k]$ is the width of the peak at element k .

In (4.3) the peak height is normalized to the maximum peak value observed in the scan as proposed in [63]. The base width in this case is being normalized

to the signal length N to account for varying scan widths. In this study all scans consist of 1024 elements with the first 7 elements set to zero so typically $N = 1024 - 7 = 1017$ elements. An examination of the effect of varying signal length has not been performed and was not a concern in this study.

The appropriate $R_{threshold}$ depends upon the scanner being used, the edge enhancement method, and the desired sensitivity of the edge detection process. It can only be determined through trial-and-error experimentation. In this study the $R_{threshold}$ was adjusted until all 12 edges of the surface in Figure 4.18 were detected consistently with the minimum number of false edges when the surface was 10-metres from the scanner and the edge height was 1-CCD pixel width. The number of false edges was further reduced by increasing the $E_{threshold}$ level associated with each edge enhancement method. The $R_{threshold}$ for the PoD method was found to be 0.05-counts and the $R_{threshold}$ for the Der method was found to be 0.01-counts based on trial and error experimentation. False edge detection was minimized for the PoD method by increasing the $E_{threshold}$ from 10 to 30 and for the Der method by increasing the $E_{threshold}$ from 6 to 15. MacKinnon *et al.* [63] used threshold values of 60-counts and 14-units for the PoD and Der filters respectively, and peak ratios of 27-units and 3.5-units respectively. However, the values used by MacKinnon *et al.* [63] were not normalized to the length of the signal. The equivalent peak ratios without signal length normalization would have be $0.05(1017) = 50.85$ and $0.01 \times 1017 = 10.17$ for the PoD and Der filters respectively. These are comparable to the threshold values used by MacKinnon *et al.* [63].

Ideally each edge should generate a single, one pixel width, edge for each edge detected. In practice some edges may be wider than one pixel width. After all

edges were detected each was eroded to a width of one-pixel. The edge vector was scanned to determine if any non-zero elements had immediate neighbours that were also nonzero. Each group of contiguous non-zero elements was assumed to denote a single edge. The group was reduced to one-pixel width by repeatedly removing the first and last elements in the group until only one element remained. If the group consisted of or was reduced to two contiguous elements then the lead element was removed.

The vector generated by the PoD or Der edge detection algorithm contains non-zero values only at points corresponding to edges in the original Lissajous scan. The term algorithm in this case and in the rest of this report refers to the process of performing median filtering, edge enhancement and peak detection to convert a Lissajous scan vector into an edge vector. The resulting edge vector is combined with a matrix of galvanometer readings to generate an edge map based on sparse range data, otherwise known as a sparse edge map. Post-processing can be used to assign a pair of peak values to each edge corresponding to the side of the edge nearer the scanner and the side of the edge farther from the scanner. MacKinnon *et al.* [63] demonstrated that this method can be used to locate objects consisting of simple planar surfaces.

4.3.1 Edge Detector Evaluation

The performance of the PoD and Der algorithms are examined under conditions of varying range, edge height, edge slope and edge separation. The data used in this section has been summarized in the following appendices:

Appendix F - This appendix shows the results of varying the height of edges

between 0-metres and 0.25-metres based on a ridge surface at 1-metre from the scanner. Ideally 12 edges should be detected. An example of the ridge surface can be seen in Figure 4.18.

Appendix G - This appendix shows the results of varying the height of edges between 0-metres and 0.5-metres based on a ridge surface at 2-metre from the scanner. Ideally 12 edges should be detected. An example of the ridge surface can be seen in Figure 4.18.

Appendix H - This appendix shows the results of varying the height of edges between 0-metres and 2.5-metres based on a ridge surface at 5-metre from the scanner. Ideally 12 edges should be detected. An example of the ridge surface can be seen in Figure 4.18.

Appendix I - This appendix shows the results of varying the height of edges between 0-metres and 2.5-metres based on a ridge surface at 10-metre from the scanner. Ideally 12 edges should be detected. An example of the ridge surface can be seen in Figure 4.18.

Appendix J - This appendix shows the results of varying the edge slope between 0.0-units and 1.0-units based on a ramp surface with near surface 5-metres from the scanner and far surface 10-metres from the scanner. Ideally 6 edges should be detected. An example of the ramp surface can be seen in Figure 4.23. The slope width is measured in fractions of the total scan width so a slope of 0.5-units corresponds to a ramp edge extending from a galvanometer input value of +0.25 to -0.25 where the maximum range is +1. to -1.0.

Appendix K - This appendix shows the results of varying the separation of

edges between 0.0-units and 1.0-units based on a ridge surface with near surface 5-metres from the scanner and far surface 10-metres from the scanner. Ideally 12 edges should be detected. An example of the ridge surface can be seen in Figure 4.18. The edge separation is measured in fractions of the total scan width so an edge separation of 0.5-units corresponds to two edges separated by a region extending from a galvanometer input value of +0.25 to -0.25 where the maximum range is +1. to -1.0.

Figure 4.23: Ramp surface in UVP-space. The near surface is 9.5-metres from the scanner and the far surface is 10-metres from the scanner.

4.3.2 Interpretation of Results

The PoD algorithm generated fewer false-positives at 1-metre and 2-metres than the Der filter which means that the PoD algorithm generates less edge “noise” than the derivative algorithm at short ranges. At 1-metre both algorithms generate unity Noise-to-Signal (NSR) rates (Figure F.4) and the PoD algorithm is only slightly, but not significantly, better than the Der algorithm at 2-metres (Figure G.4). At 5-metre and 10-metre ranges the PoD and Der algorithms produce roughly the same results with the PoD algorithm generating slightly more false-positives than the Der algorithm. Both methods achieve NSR levels of almost zero as edge height increases (Figure H.4, Figure I.4).

In general the PoD algorithm produces slightly better results based primarily on performance at short range. Note that each of the algorithms can be tuned by adjusting the threshold and peak ratio levels to improve performance. Computation speed can be increased at the cost of increased noise by decreasing the size of the median filter. Increasing the PoD filter window size at the cost of increasing computation time can increase edge sensitivity of the PoD filter.

4.4 Phase III: Speed Evaluation

The noise filtering, edge enhancement and peak detection methods were converted to ANSI-C code and initially tested using Matlab. The edge detector evaluations used in the previous section were based on results obtained from data processed using the ANSI-C code equivalent scripts. This code was transferred to a QNX workstation and integrated into a test system to determine whether the edge detection algorithm could be performed in less than the 72-milliseconds required

to perform a single 1024-element scan.

The edge detection algorithms were ported to the scanner workstation and were compiled under QNX using Watcom-C. The algorithm was performed 100 times and the total processing time was recorded. The averaging time to process a single scan was found by dividing the total processing time by the number of repetitions. It was found that PoD algorithm with the median filter using swapping to optimize speed required an average of 5.49 ± 0.18 -msec to process a 1024-element array based on ten repetitions of the test. The iterated median filter was found to produce results that were not significantly different than those generated by the median filter so it may be used as an alternative to the median filter. The PoD algorithm using the iterated median filter processed a 1024-element array in 5.60 ± 0.40 -msec. The average processing time using the iterated median filter was not significantly different than the average processing time using the median filter with swapping ($p=0.476$) based on a two-tailed t-test [20, pp.330-335] [61, pp.100-107]. For comparison, the PoD algorithm using the non-optimized median filter would have required 18.31 ± 0.23 -msec to process a 1024-element array.

4.5 Real-time Edge Detection

The PoD algorithm was used to detect the edges of two simple objects using the Calib scanner. Figure 4.24 shows edges of a simple planar ridge object 3-metres from the scanner with an edge height of 0.41-metres detected by the Calib scanner under the QNX real-time operating system. The top image shows the original peak profile and the bottom image shows the results of edge detection. The locations of detected edges are shown as crosses on Lissajous pattern in the

bottom image. The locations of the sides of the ridge object are shown as solid lines on the Lissajous scan in the top image. All detected edges coincided with edges of the ridge object. The peak width to height ratio was increased to 1.0 for data generated by the Calib scanner.

4.6 Summary

An 11-element PoD filter was found to perform better than a 3-point Der filter in detecting edges of step and steep ramp surfaces between 1-metre and 10-metres. The PoD algorithm generated fewer false edges than the Der algorithm at short range but at long range both algorithms performed equally well. Both methods detected edges separated by as little as 12.5% of the total scan width. The PoD algorithm will detect edges of slightly shallower ramp edges than the Der filter.

The original Lissajous scans were filtered using a 13-element median filter that was found to be most effective in restoring the original signal under worst-case noise conditions. The iterated median filter was slightly but not significantly less accurate in reconstructing the original ideal signal. Moreover, the processing time for the iterated median filter was not significantly different than the processing time for the median filter optimized using swapping. In practice, either the optimized median filter or the iterated filter could be used as an effective prefilter based on the observed noise levels.

Data produced using the edge enhancement methods were compared to threshold levels and values less than the threshold level were set equal to zero. The base width-to-peak height ratio was then examined and if greater than a threshold level were eliminated as peaks corresponding to edges. The resulting edge vector

Figure 4.24: Results of performing real-time edge detection of a simple ridge object 3-metres from the scanner. The top image shows the Lissajous scan (left) and the peak profile (right) obtained as a screen capture from the QNX workstation controlling the Calib scanner. Vertical lines on the Lissajous pattern indicate the position of the surface in the scan window. The bottom image is a screen capture of the edge detection system using the scan scanner and workstation. Edges can be seen as solid vertical lines (right) and crosses on the Lissajous pattern (left).

can be combined with the original galvanometer readings to form a map of the locations of edges in UVP space, referred to as a sparse edge map.

Chapter 5

Conclusions and Future Work

Two important conclusions can be drawn from this thesis. First, it demonstrated that an edge detection algorithm consisting of a median filter, PoD edge enhancement filter, and a customized peak detector employing a combination of thresholding and width-to-peak height peak selection, can be used to detect edges in real-time. This algorithm was developed and tested primarily using the Random Access Scanner (RAS) model, demonstrating the second conclusion of the thesis; that the model developed as part of this thesis is sufficient to facilitate offline development of algorithms that can be ported to the scanner with minimal changes.

Two important results arose from the work presented here. First, a model was developed that can be used to develop algorithms for use on the RAS system. This will facilitate future research into object detection and tracking using sparse range data. Second, it has been demonstrated that edge detection can be performed using raw data obtained directly from the scanner without requiring a time-consuming conversion into Cartesian coordinates.

The model developed for this thesis is sufficient for rapidly generating test data similar to that which would be obtained from a real scanner. Scanner motion has been included in the current model so it can be used to examine the effects of motion distortion arising from scanner motion. The scanner position is specified as part of the environmental model so one can simulate placing the scanner on a mobile platform or introduce environmental effects such as vibration. Algorithms can be developed based on feedback loops in which the size, position, or format of the next scan can be adjusted based on analysis of the current scan.

The edge detection method developed for this thesis demonstrates that edge detection can be performed using the UVP coordinate system. Using the UVP coordinate system rather than the traditional Cartesian coordinate system means that UVP data does not have to be transformed before edge detection can be performed. The edge detection algorithm is able to process a single Lissajous scan before the next Lissajous scan is obtained so is applicable to real-time feedback control of the scanner. It takes 72-milliseconds to perform a 1024-element Lissajous scan and the edge detection algorithm can be performed in less than 6-milliseconds. This leaves 66-milliseconds in which to evaluate the results of the edge detection algorithm and possibly adjust the position, size or format of the next scan. Future work can examine possible object contour-based control strategies that could be performed within this time constraint.

In the introduction to this thesis it was stated that the reason for developing a method for generating edge maps was as a precursor to object tracking. Sparse edge maps can be used to reduce the number of data points required to track an object. For example an object may generate 10 edges in a 1024-element scan. Each edge would have a near and far range value associated with it so a total of 20

data points would be processed to determine the edge profile of the object. This requires much less time than processing all 1024 samples to determine the object's edge profile. Motion distortion can make it difficult to identify the object's edge profile so it may be difficult to track the object. However, methods currently being developed by the NRC may be applicable to addressing this issue [67]. Feature-based object tracking is an important area of research with a wide range of applications. For example, an object could be tracking based on its edge profile using a Lissajous scan while a raster scan is performed to obtain a more detailed contour image of the object. Edge profiles can also be used in space to track objects that have not been marked with targets.

The scanner model developed in this thesis was sufficient for the purpose of developing an edge detection algorithm and can now be used for studies in motion distortion. However some aspects of the scanner and environmental models can be improved. These areas include the addition of laser signal intensity to the scanner model, adding object motion and occlusion to the environmental model, and modifying the current scanning methods to compute multiple ray paths rather than a single ray path.

The model does not consider the reduction in laser signal intensity or beam spread resulting from the distance between the scanner and the point of intersection with a surface in the environment. This is important when non-Lambertian surfaces are used, and when the laser crosses an edge. A model of the signal profile on the surface of the object being illuminated by the laser can be used to examine the effects of signal distortion by edges. To simplify the process the model can be restricted to simulating an intensity profile along the plane of the CCD-array. The resulting signal can then be processed using the linear interpo-

lation routine developed by Blais and Rioux [1] or new signal processing methods can be examined.

The environmental model was developed so that the scanner could be moved while a scan was taking place. However, this feature was not used during the course of this research. Future research can focus on developing algorithms to handle motion distortion resulting from moving the scanner during a raster or Lissajous scan. Moreover, the ability to move objects during a scan can be added to introduce additional motion distortion. This environmental model could be used to develop algorithms designed to track objects based on features in a dynamic environment.

In this thesis signal occlusion was handled as aperiodic impulse noise. The environmental model can be improved by determining if any surface intersects the image path. The laser may intersect a surface but the point of intersection may not be visible to the CCD array. This should result in no peak signal while the intersection point is occluded. Acquiring a single peak reading takes a small but significant amount of time so if the scanner, objects or both are moving then the signal will be spread over the surface or surfaces. This shifting of the effective signal peak can result in an erroneous range measurement. Future modifications to the RAS-environmental model interaction can explore these issues. Another issue that was mentioned but not examined was that of laser re-reflection that could result in non-zero spikes. A more comprehensive ray-trace model could simulate this effect.

The current environmental model contains only boxes as models of objects. The NRC has acquired 3-D models of objects and many more are available on the Internet. A future project can be to convert a selection of these 3-D models

into objects in the environmental model. The environmental model can also be redesigned to accept time-varying non-linear surfaces so that the problems of scanning deformable surfaces can be explored.

The edge detection method developed in this thesis performed well and met the requirement of real-time performance. Future research can focus on increasing the sensitivity of the edge detection method and testing its performance under dynamic conditions and at a range of scan resolutions.

The peak detector developed for this thesis accepts a peak as potentially arising from an edge by examining the ratio of peak base width to the height of the peak taken at the threshold level. An alternative approach is to specify a peak height limit L_p and determine the width of the base of the peak at a point L_p below the peak. The ratio is then calculated based on the base width divided by L_p . This should allow the peak detector to better differentiate between peaks generated by short edges and peaks generated by shallow ramp edges or surface curvature. This can be combined with some form of threshold, normalization or a combination of the two to reduce the number of erroneous peaks detected.

Experiments using the scanner system at various scan resolutions indicates that normalizing the base width to the scan width may not be effective in generalizing the algorithm to all possible scan resolutions. Future research can focus on developing an algorithm relating the scan resolution to a function of the base width. Ideally a single peak width ratio should apply to any scan resolution between 128-elements and 4096-elements. Currently the ratio must be adjusted when the scan resolution is changed. The results presented in this thesis assumed an operator would manually adjust the edge detection parameters to maximize the number of true edges detected while minimizing the number of false edges

detected. One way to address this issue is to determine the optimal peak width ratio for each scan width between 128 and 4096-elements. The system currently uses only scan widths of 128, 256, 1024, 2048 and 4096-elements so a peak width ratio could easily be assigned to each.

The edge detection algorithm was designed with the assumption that the environment is static but this is not always the case. The noise filters and edge enhancement filters were designed based on the assumption that the preceding and proceeding scans would be effectively identical to this one; that is, the data is assumed to come from a periodic source. This assumption is the basis of techniques such as circular convolution [19, p.571]. This technique was selected to avoid reducing the number of data points in each successive step. For example, if a 128-element signal is filtered using a 11-element median filter without assuming signal periodicity then the resulting signal is truncated at either end by 5 elements leaving 118 usable data elements bounded by 5-element regions on either side consisting of zeros. Applying an 11-element PoD filter would further reduce the number of usable elements to 108. As a result, edges in the 10-element wide regions at the start and end of the scan would not be detected. However, if the scanner or object is moving then the peak height at one end of the scan may be significantly different than the peak height at the other end resulting in a false edge. The current edge detection algorithm should be tested in a dynamic environment to determine if it is able to detect edges with minimal false and missed edges.

In this study a large-window PoD edge enhancement method was compared to a small-window Der filter. Future research can compare large- and small-window PoD enhancement filters, large- and small-window Der enhancement filters, and

compare same-sized PoD and Der filters. These studies would determine whether the better performance of the 11-element PoD enhancement filter over the 3-element Der filter were due to window size, filter type, or both.

References

1. Blais, F., and Rioux, M., "Real-time numerical peak detector," *Signal Process*, Vol.11 (NRC 26561): Elsevier, 1986, pp.145-155.
2. Trucco, E., and Verri, A., *Introductory Techniques for 3D Computer Vision*, Prentice-Hall, 1998.
3. Payeur, P., *Machine Vision*, class notes for Winter term ELG5163: University of Ottawa School of Information Technology and Engineering, 2002.
4. MacLean, S.G., and Pinkney, H.F.L., "Machine Vision in Space," *Canadian Aeronautics and Space Journal*, Vol.39, No.2, June 1993. pp.63-77.
5. Beraldin, J.-A., El-Hakim, S.F., and Cournoyer, L., "Practical range camera calibration," *SPIE Proceedings of Videometrics II*, Vol.2067, Boston, MA. September 7-10 (NRC 35064): SPIE, 1993, pp.21-31.
6. Blais, F., Couvillon, R.A., Rioux, M., and MacLean, S.G., "Real-time tracking of objects for space applications using a laser range scanner," *Proceedings of the AAAI Conference on Intelligent Robots in Field, Factory, Service and Space* Vol.II, Houston, TX. March 20-24 (NRC 37109): 1994, pp.464-472.

7. Beraldin, J.-A., Blais, F., Rioux, M., Cournoyer, L., Laurin, D., and MacLean, S., "Short and medium range 3D sensing for space applications," *SPIE Proceedings of Visual Information Processing VI (Aerosense '97)*, Vol.3074, Orlando, FL. April 21-25 (NRC 40170): SPIE, 2000, pp.29-46.
8. Beraldin, J.-A., Blais, F., Rioux, M., Cournoyer, L., Laurin, D., and MacLean, S.G., "Eye-safe digital 3D sensing for space applications," *Optical Engineering*, Vol.39, No.1, January (NRC 43585): Society of Optical Instrumentation Engineers, 2000, pp.196-211.
9. Blais, F., Beraldin, J.-A., and El-Hakim, S.F., "Range error analysis of an integrated time-of-flight, triangulation, and photogrammetry 3D laser scanning system," *SPIE Proceedings of AeroSense*, Vol.4035, Orlando, FL. April 24-28 (NRC 43649): SPIE, 2000.
10. Blais, F., Beraldin, J.-A., Cournoyer, L., Christie, I., Serafini, R., Mason, K., McCarthy, S., and Goodall, C., "Integration of a tracking laser range camera with the photogrammetry based space vision system," *SPIE Proceedings of AeroSense*, Vol.4025, Orlando, FL. April 24-28 (NRC 43650): SPIE, 2000.
11. Blais, F., Beraldin, J.-A., El-Hakim, S.F., and Cournoyer, L., "Comparison of Pose Estimation Methods of a 3D Laser Tracking System using Triangulation and Photogrammetry Techniques," *SPIE Proceedings of Electronic Imaging 2001, Videometrics and Optical Methods for 3D Shape Measurement VII*, Vol. 4309, San-Jose, CA. January 21-26 (NRC 44152): SPIE, 2001.
12. Blais, F., Beraldin, J.-A., El-Hakim, S.F., and Cournoyer, L., "Real-time

- Geometrical Tracking and Pose Estimation using Laser Triangulation and Photogrammetry,” *Proceedings of the Third International Conference on 3-D Digital Imaging and Modeling*, 28 May-1 June, Quebec, Quebec (NRC 44180): IEEE, 2001, pp.205-212.
13. Blais, F., Beraldin, J.-A., Cournoyer, L., El-Hakim, S.F., Domey, J., and Rioux, M., “The NRC 3D Laser Tracking System: IITs Contribution to International Space Station Project,” *Proceedings of the 2001 Workshop of Italy-Canada on 3D Digital Imaging and Modeling Application of: Heritage, Industry, Medicine, & Land* Padova, Italy, April 3-4 (NRC 44181): 2001.
 14. Blais, F., Beraldin, J.-A., Cournoyer, L., El-Hakim, S.F., Picard M., Domey, J., Rioux, M., Christie, I., Serafini, R., Pepper G., MacLean S.G., and Laurin D., “Target Tracking Object Pose Estimation, and Effect of the Sun on the NRC 3-D Laser Tracker,” *Proceedings of iSAIRAS-2001* Montreal, Quebec, June (NRC 44876): 2001.
 15. Cundy, H. Martyn, and Rollett, A.P., *Mathematical Models*, 2nd edition, Oxford University Press, 1961, pp.242-245.
 16. Brieskorn, Egbert, and Knörrer, Horst, *Plane Algebraic Curves*, Birkhäuser Boston, 1986, pp.65,77-79.
 17. Kantrowitz, Philip, Kousourou, Gabriel, and Zucker, Lawrence, *Electronic Measurements*, Englewood Cliffs, NJ: Prentice-Hall, Inc., 1979, pp.125-126.
 18. Cooper, William David, *Electronic Instrumentation and Measurement Techniques*, Englewood Cliffs, NJ. USA: Prentice-Hall, Inc., 1970, pp.228-234.

19. Oppenheim, Alan V., and Schafer, Ronald W., Buck, John R., *Discrete-Time Signal Processing, 2nd edition*, Upper Saddle River, New Jersey: Prentice-Hall, Inc., 1999.
20. Mendenhall, William, Scheaffer, Richard L., and Wackerly, Dennis D., *Mathematical Statistics with Applications, 3rd edition*, PWS Publishers, 1986.
21. Glantz, S.A., and Slinker, B.K., *Primer of Applied Regression and Analysis of Variance*, McGraw-Hill, Inc., 1990.
22. Larmond, E., *Laboratory Methods for Sensory Evaluation of Food*, Food Research Institute, Ottawa, ON. (Publication 1637): Research Branch, Canada Department of Agriculture, 1982.
23. Stewart, James, *Calculus, Second Edition: Early Transcendentals*, Pacific Grove, California: Brooks/Cole Publishing Company, 1991.
24. Blais, Francois, Personal communication 26-July-2002.
25. Elmes, D.G., Kantowitz, B.H., and Roediger, H.L. III, *Research Methods in Psychology, 4th edition*, St. Paul, MN: West Publishing Company, 1992.
26. Keppel, G., and Saufley, W.H., Jr., *Introduction to Design and Analysis: A Student's Handbook*, W.H. Freeman and Company, 1980.
27. Peli, Tamar; Malah, David, "A study of Edge Detection Algorithms," *IEEE Transactions on Computer Graphics and Image Processing*, 1982, Vol.20, pp.1-21.

28. van der Heyden, F., "Evaluation of edge detection algorithms," *Third International Conference on Image Processing and its Applications*, 1989, pp.618-622.
29. Pratt, W.K., *Digital Image Processing, 3rd edition*, John Wiley and Sons, 2001.
30. Ritter, G.X. and Wilson, J.N., *Handbook of Computer Vision Algorithms in Image Algebra*, 2nd edition, CRC Press, 2000.
31. Heath, M.D., Sarkar, S., Sanocki, T. and Bowyer, K.W., "A robust visual method for assessing the relative performance of edge-detection algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1997, Vol.19, No.12, pp.1338-1359.
32. Larmond, E., *Laboratory Methods for Sensory Evaluation of Food*, Food Research Institute, Ottawa, ON. (Publication 1637): Research Branch, Canada Department of Agriculture, 1977.
33. Blais, Francois, Personal communication March 2002.
34. Canny, J., "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1986, Vol.8, No.6, pp.679-698.
35. Zhou, Y.T., Venkateswar, V. and Chellappa, R., "Edge Detection and Linear Feature Extraction Using and 2-D Random Field Model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1989, Vol.11, No.1, pp.84-95.

36. Shin, M.C., Goldgof, D. and Bowyer, K.W., "Comparison of edge detectors using an object recognition task," *Proceedings of the 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol.1, pp.36-365.
37. Heath, M., Sarkar, S., Sanocki T. and Bowyer, K., "Comparison of edge detectors: a methodology and initial study," *Proceedings of the 1996 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'96)*, pp.143-148.
38. Staunton, R.C., "Measuring image edge detector accuracy using realistically simulated edges," *Electronics Letters*, 1977, pp.2031-2032.
39. Bovik, Alan Conrad, Huang, Thomas S., Munson, David C. Jr., "The Effect of Median Filtering on Edge Estimation and Detection," *Pattern Analysis and Machine Intelligence*, 1987, Vol.9, No.2, pp.181-194.
40. Chin, Roland T. Yeh, Chia-Lung, "Quantitative Evaluation of Some Edge-Preserving Noise-Smoothing Techniques," *Computer Vision, Graphics, and Image Processing*, 1983, Vol.23, pp.67-91.
41. Lyvers, Edward P. Mitchell, O. Robert, "Precision Edge Contrast and Orientation Estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1988, Vol.10, No.6.
42. Pitas, I. and Venetsanopoulos, "Edge Detectors Based on Nonlinear Filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1986, Vol.8, No.4., pp.538-550.

43. Yoo, J., Bouman, C.A., Delp, E.J. and Coyle, E.J., "The Nonlinear Pre-filtering and Difference of Estimates Approaches to Edge Detection: Applications of Stack Filters," *Graphical Models and Image Processing*, 1993, Vol.55, No.2, pp.140-159.
44. Rostampour, A.R. and Reeves, A.P., "2D median filtering and pseudo median filtering," *Proceedings of the Twentieth Southeastern Symposium on System Theory*, 1988, 20-22 March, pp.554-557.
45. Burian, A. and Kuosmanen, P., "Tuning the smoothness of the recursive median filter," *IEEE Transactions on Signal Processing*, 2002, Vol.50, Iss.7, pp.1631-1639.
46. Zhou, Xingwei and Ye, Wanzhou, "A conjecture on iterated three-point median filter," *IEEE Transactions on Signal Processing*, 1996, Vol.44, Iss.9, pp.2347-2350.
47. Davies, E.R., "A remnant noise problem with the median filter," *Proceedings of the 11th IAPR International Conference on Image, Speech and Signal Analysis in Pattern Recognition*, 1992, 30 Aug-3 Sep, Vol.III, pp.505-508.
48. Angelopoulos, G. and Pitas, I., "A fast implementation of two-dimensional weighted median filters," *Proceedings of the 12th IAPR International Conference on Signal Processing in Pattern Recognition*, 1994, 9-13 Oct, Vol.3, pp.140-142.
49. Ranka, S. and Sahni, S. "Efficient serial and parallel algorithms for median filtering" *IEEE Transactions on Signal Processing* 1991, Vol.39, Iss.6, pp.1462-1466.

50. Mittermair, D. and Puschner, P., "Which sorting algorithms to choose for hard real-time applications," *Proceedings of the Ninth Euromicro Workshop on Real-Time Systems*, 1997, 11-13 Jun, pp.250-257.
51. Leeb, S.B., Ortiz, A. and Kirtley, J.L., Jr., "Real-time median filtering with a fast hardware sorter," *Conference Proceedings of the Sixth Annual Applied Power Electronics Conference and Exposition (APEC'91)*, 1991, 10-15 Mar, pp.254-260.
52. Guoping Qiu, "Functional optimization properties of median filtering," *IEEE Signal Processing Letters*, 1994, Vol.1, Iss.4, pp.64-65.
53. Yuan Zou and Dunsmuir, W.T.M., "Generalized max/median filtering," *Proceedings of the International Conference on Image Processing*, 1997, Vol.1, 26-29 Oct, pp.428-431.
54. Wendt, P.D., "Nonrecursive and recursive stack filters and their filtering behavior," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 1990, Vol.38, Iss.12, pp.2099-2107.
55. Restrepo Palacios, A. and Chacon, L., "A smoothing property of the median filter," *IEEE Transactions on Signal Processing*, 1994, Vol.42, Iss.6, pp.1553-1555.
56. Coyle, E.J., Gabbouj, M. and Lin, J.-H., "From median filters to optimal stack filtering," *IEEE International Symposium on Circuits and Systems*, 1991, Vol.1, 11-14 Jun, pp.9-12.
57. Qiu, Gnoping, "An improved recursive median filtering scheme for image

- processing," *IEEE Transactions on Image Processing*, 1996, Vol.5, Iss.4, pp.646-648.
58. Oliver, C.J., Blacknell, D. and White, R.G., "Optimum edge detection in SAR," *IEE Proceedings on Radar, Sonar and Navigation*, 1996, Vol.143, Iss.1, pp.31-40.
59. Davies, E.R. and Celano, D., "Orientation accuracy of edge detection operators acting on binary and saturated grey-scale images," *Electronics Letters*, 1993, Vol.29, Iss.7, pp.603-604.
60. Burden, R.L., and Faires, J.D., *Numerical Analysis, Fourth Edition*, PWS-Kent Publishing Company, 1989.
61. Ott, Lyman, *An Introduction to Statistical Methods and Data Analysis*, Duxbury Press, 1977.
62. David K. MacKinnon, Francois Blais, Victor C.Aitken, "Modeling an Auto-synchronizing Laser Range Scanner," *ACC 2003 American Control Conference*, Denver, Colorado, USA, 4-6 June (NRC 45840): 2003, pp.1-6.
63. MacKinnon, David K., Blais, Francois, Aitken, Victor C., "Object Location using Edge-Bounded Planar Surfaces from Sparse Range Data," *IEEE Canada, CCECE 2003 Canadian Conference on Electrical and Computer Engineering*, Montreal, Canada, 4-7 May (NRC 45831): 2003, pp.1-6.
64. Jain, R., Kasturi, R. and Schunck, B.G., *Machine Vision*, McGraw-Hill, Inc., 1995.

65. Robinson, Guner S., "Edge Detection by Compass Gradient Masks," *Computer Vision and Image Processing*, Vol.6, No.5, 15 Oct, 1977 pp.492-501.
66. Glassner, Andrew S., *Principles of Digital Image Synthesis (Vol.2)*, Morgan Kaufmann Publishers, Inc., 1995.
67. Blais, F., Picard, M., Gaudin, G., "Recursive Model Optimization Using ICP and Free Moving 3D Data Acquisition," *Proceedings of the 3DIM 2003 Conference*, Banff, Alberta, 6-10 October (NRC 45834): 2003.

Appendix A: Scanner Model Development

A.1 Calculating MOP

The MOP defines the path the laser follows out of the scanner. The path changes at three points of incidence defined as $\mathbf{f}_1(t)$, $\mathbf{f}_2(t)$ and $\mathbf{f}_3(t)$. These points of incidence are the deviations on the laser path caused by reflection at the x-axis, fixed output and y-axis mirrors respectively. The laser path away from each of the points is defined as $\vec{f}_1(t)$, $\vec{f}_2(t)$ and $\vec{f}_3(t)$ respectively as can be seen in Figure A.1. The vector $\vec{f}_0(t)$ is defined as the path followed by the laser from the source to its point of incidence $\mathbf{f}_1(t)$ with the x-axis mirror. This point is defined to be the origin of the scanner frame of reference assuming an infinitely thin mirror so $\mathbf{f}_1(t) = [0, 0, 0]^T$.

A.1.1 Point of incidence on the fixed output mirror

The point of incidence $\vec{f}_2(t)$ of the laser with the fixed output mirror as seen in Figure A.1 is found by first calculating the angle of incidence of the laser with the

Figure A.1: Detailed MOP in (x,y)-plane

Figure A.2: Detailed MOP in (y,z)-plane

x-axis mirror. This is found using

$$A(t) = \pi/2 - \beta_x - \Theta(t) \quad (\text{A.1})$$

where $\Theta(t)$ is the rotation of the x-axis mirror caused by the galvanometer. The parameter β_x is the angular offset of the x-axis mirror from the (x,z)-plane when the x-galvanometer rotation is zero.

The vector $\vec{f}_1(t)$ is assumed to be strictly within the (x,y)-plane so only the motion in this plane needs to be considered. Using (A.1) the angular deviation $\varphi(t)$ from the (x,z)-plane can be determined using

$$\varphi(t) = A(t) - \beta_x - \Theta(t) \quad (\text{A.2})$$

which becomes **Equation (3.2)** by substituting (A.1) into (A.2).

The angle of incidence $\gamma_{out}(t)$ of $\vec{f}_1(t)$ with the fixed output mirror is defined as

$$\gamma_{out}(t) = \beta_{out} - \varphi(t) \quad (\text{A.3})$$

where β_{out} is the angular offset of the fixed output mirror from the (x,z)-plane.

The point $\mathbf{f}_2(t)$ is completely within the (x,y)-plane at $z = 0$ so $f_{z2}(t) = 0$. The values of $f_{x2}(t)$ and $f_{y2}(t)$ are found by considering the triangle $K\triangle\mathbf{f}_1\triangle\mathbf{f}_2$ as seen in Figure A.1. The length $c(t)$ of the bisection line of this triangle is defined by

$$c(t) = h_x \sin(\varphi(t)) \quad (\text{A.4})$$

and also by

$$c(t) = r(t) \sin(\gamma_{out}(t)) . \quad (\text{A.5})$$

Let (A.4) equal (A.5) and solve for $r(t)$ to obtain

$$r(t) = h_x \frac{\sin(\varphi(t))}{\sin(\gamma_{out}(t))} . \quad (\text{A.6})$$

As a result

$$f_{y2}(t) = r(t) \sin(\beta_{out}) \quad (\text{A.7})$$

and

$$f_{x2}(t) = h_x + r(t) \cos(\beta_{out}) . \quad (\text{A.8})$$

By combining (A.3), (A.6), (A.7) and (A.8) the point of incidence of the laser path with the fixed output mirror can be defined as

$$\mathbf{f}_2(t) = \begin{bmatrix} h_x + h_x \frac{\sin(\varphi(t)) \cos(\beta_{out})}{\sin(\beta_{out} - \varphi(t))} \\ h_x \frac{\sin(\varphi(t)) \sin(\beta_{out})}{\sin(\beta_{out} - \varphi(t))} \\ 0 \end{bmatrix} . \quad (\text{A.9})$$

A.1.2 Point of incidence on the y-axis mirror

The point of incidence of the laser with the y-axis mirror $\mathbf{f}_3(t)$ as seen in Figure A.2 is found by first calculating the angle of incidence of $\vec{f}_2(t)$ with the y-axis mirror. This is found by

$$B(t) = \pi/2 - \Phi(t) - \beta_y \quad (\text{A.10})$$

where β_y is the angular offset of the x-axis mirror from the (x,z)-plane when the y-galvanometer rotation is zero. The deviation of $\vec{f}_3(t)$ from the a plane parallel to the (x,z)-plane can then be defined as

$$V(t) = 2\beta_y + 2\Phi(t) - \pi/2 \quad (\text{A.11})$$

which is **Equation (3.4)**. Similarly the deviation of $\vec{f}_3(t)$ from a plane parallel to the (y,z)-plane can be defined as

$$U(t) = \pi/2 - (\beta_{out} + \gamma_{out}(t)) . \quad (\text{A.12})$$

Substituting (A.3) into (A.12) results in

$$U(t) = \pi - 2\beta_{out} - 2\beta_x - 2\Theta(t) \quad (\text{A.13})$$

which is **Equation (3.3)**.

The point of intersection of the laser with the y-axis mirror is still in the (x,y)-plane so $f_{z3}(t) = 0$. If h_z is assumed to be non-zero then $f_{y3}(t)$ must be defined before $f_{x3}(t)$ can be determined. It can be seen that

$$f_{y3}(t) = h_y - h_z \tan(\pi/2 - B(t)) \quad (\text{A.14})$$

which becomes

$$f_{y3}(t) = h_y - h_z \tan(\Phi(t) + \beta_y) \quad (\text{A.15})$$

after substituting (A.10) into (A.14). Using $U(t)$ it can be seen that

$$f_{x3}(t) = f_{x2}(t) + f_{y3}(t) \tan(U(t)) . \quad (\text{A.16})$$

By combining (A.3), (A.6), (A.8), (A.15) and (A.16) the point of incidence of the laser path with the y-axis mirror is found to be **Equation (3.1)**.

A.2 Imaging Axis

The imaging axis defines the path from the point of intersection of the laser path with a surface an infinite distance from the scanner to the point intersection with the lens plane. This serves as a reference for calculating the range of an actual point of incidence with a surface in the environment. The path changes at four points of incidence defined as $\mathbf{a}_0(t)$, $\mathbf{a}_1(t)$, $\mathbf{a}_2(t)$ and $\mathbf{a}_3(t)$ as can be seen in Figure A.3. These points of incidence are the point of origin at the lens plane and the deviations of the image axis at the x-axis, fixed output and y-axis mirrors respectively. The image axis can also be defined as a set of vectors $\vec{a}_0(t)$, $\vec{a}_1(t)$, $\vec{a}_2(t)$ and $\vec{a}_3(t)$ respectively. The point of incidence $\mathbf{a}_1(t)$ with the x-axis mirror is defined to be the origin of the scanner frame of reference assuming a the surface of the mirror passes through the axis of rotation of the mirror so $\mathbf{a}_1(t) = [0, 0, 0]^T$. The point of incidence with the lens plane is defined to be a distance D_l along the negative y-axis so $\mathbf{a}_0(t) = [0, -D_l, 0]^T$.

Figure A.3: Detailed Imaging Axis in (x,y)-plane

Figure A.4: Detailed Imaging Axis in (y,z)-plane

A.2.1 Point of incidence on the fixed input mirror

The point of incidence of the Imaging Axis with the fixed output mirror is found by considering a triangle $K\triangle a_1\triangle M$ in Figure A.3. It can be seen that

$$r(t) = h_x \sin(\varphi(t)) \quad (\text{A.17})$$

where $\varphi(t)$ is defined using **Equation 3.2**. A second triangle $K\triangle a_1\triangle a_2$ as seen in Figure A.3 is then used to find

$$p(t) = \frac{r(t)}{\sin(\varphi(t) + \beta_{in})} \quad (\text{A.18})$$

where β_{out} is the angular offset of the fixed output mirror from the (x,z)-plane. Substituting (A.17) into (A.18) yields

$$p(t) = h_x \frac{\sin(\varphi(t))}{\sin(\varphi(t) + \beta_{in})}. \quad (\text{A.19})$$

As a result

$$a_{y2}(t) = -p(t) \sin(\beta_{in}) \quad (\text{A.20})$$

and

$$a_{x2}(t) = -h_x + p(t) \cos(\beta_{in}) . \quad (\text{A.21})$$

The angle of incidence $\gamma_{in}(t)$ of $\vec{a}_1(t)$ with the fixed input mirror is defined as

$$\gamma_{in}(t) = \beta_{in} + \varphi(t) . \quad (\text{A.22})$$

As in the previous section, the imaging axis has been defined to only exist within the (x,y)-plane at $z = 0$ so $a_{z2}(t) = 0$. (A.19), (A.20) and (A.21) can be combined to define the point of intersection of the imaging axis with the fixed input mirror as

$$\mathbf{a}_2(t) = \begin{bmatrix} -h_x + h_x \frac{\sin(\varphi(t)) \cos(\beta_{in})}{\sin(\varphi(t) + \beta_{in})} \\ -h_x \frac{\sin(\varphi(t)) \sin(\beta_{in})}{\sin(\varphi(t) + \beta_{in})} \\ 0 \end{bmatrix} \quad (\text{A.23})$$

which is (3.8).

A.2.2 Point of incidence on the y-axis mirror

The point of incidence of the imaging axis with the y-axis mirror $\mathbf{a}_3(t)$ as seen in Figure A.4 is found by first calculating the angle of incidence of $\vec{a}_2(t)$ with the y-axis mirror. This is found by

$$B(t) = \pi/2 - \Phi(t) - \beta_y \quad (\text{A.24})$$

which is the same as (A.10) because the imaging axis uses the same y-axis mirror as the laser with the same optical arrangement. As a result, $V_{OP}(t) = V(t)$.

Similarly the deviation of $\vec{a}_3(t)$ from a plane parallel to the (y,z)-plane can be defined as

$$U_{OP}(t) = \gamma_{in}(t) - (\pi/2 - \beta_{in}) . \quad (\text{A.25})$$

This simplifies to

$$U_{OP}(t) = \varphi(t) - \pi/2 + 2\beta_{in} \quad (\text{A.26})$$

by substituting (A.22) and can be generalized to **Equation (3.6)** by substituting (A.2). This defines the angular deviation from the (x,z)-plane.

The point of intersection of the imaging axis with the y-axis mirror is still in the (x,y)-plane so $a_{z3}(t) = 0$. If h_z is assumed to be non-zero then $a_{y3}(t)$ must be defined before $a_{x3}(t)$ can be determined. From Figure A.4 it can be seen that

$$a_{y3}(t) = h_y - h_z \tan(\pi/2 - B(t)) \quad (\text{A.27})$$

which becomes

$$a_{y3}(t) = h_y - h_z \tan(\Phi(t) + \beta_y) \quad (\text{A.28})$$

after substituting (A.24). Using $U_{OP}(t)$ it can be seen that

$$a_{x3}(t) = a_{x2}(t) + a_{y3}(t) \tan(U_{OP}(t)) . \quad (\text{A.29})$$

By combining (A.22), (A.19), (A.21), (A.28) and (A.29) the point of incidence of the laser path with the y-axis mirror is found to be **Equation (3.5)**.

Appendix B: Environmental Model Development

B.1 Modelling the Environment

The development of an environmental model required consideration of a number of factors. The RAS frame of reference was measured in units of millimetres and was centred on the point of intersection of the laser with the x-axis mirror. The environmental model must allow the camera to be placed at any position and orientation. At any point in time, the camera would interact with the environment by producing a ray with a given focus and deviation from the camera FOV axis. This ray may intersect any surface in the environment. The point of intersection must then be used to determine if the point is visible from the camera, and at what point, if any, on the CCD array it would be detected. Finally, the model must allow for the possibility of the camera moving during the scanning interval. Moving objects were not required for this simulation but the model has been designed to allow moving models to be implemented. However, only stationary objects and a moving camera were considered in this thesis.

The world frame of reference is defined as $\mathcal{R}_{World}$ and the camera frame of

reference is defined as $\mathfrak{R}_{RAS}$. Recall that the vector $\vec{f}_3$ defines the laser path represents a ray in $\mathfrak{R}_{RAS}$. This vector must be mapped into $\mathfrak{R}_{World}$ using a sequence of translation and rotations. The typical orientation of the scanner is one in which the z-axis of $\mathfrak{R}_{RAS}$ is directed along the x-axis of $\mathfrak{R}_{World}$ and the y-axis of $\mathfrak{R}_{RAS}$ is directed along the negative z-axis of $\mathfrak{R}_{World}$. One way to map $\vec{f}_3 \in \mathfrak{R}_{RAS}$ to $\vec{f}_3 \in \mathfrak{R}_{World}$ is to perform a rotation of the z-axis of $\mathfrak{R}_{RAS}$ about the x-axis, then rotate that about the y-axis. The resulting point can then be translated to an appropriate location in $\mathfrak{R}_{World}$.

Three rotation matrices are defined as

$$\mathbf{Y}_{x,World|RAS} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\xi_x) & -\sin(\xi_x) \\ 0 & \sin(\xi_x) & \cos(\xi_x) \end{bmatrix} \quad (\text{B.1})$$

which is referred to as roll,

$$\mathbf{Y}_{y,World|RAS} = \begin{bmatrix} \cos(\xi_y) & 0 & \sin(\xi_y) \\ 0 & 1 & 0 \\ -\sin(\xi_y) & 0 & \cos(\xi_y) \end{bmatrix} \quad (\text{B.2})$$

which is referred to as pitch, and

$$\mathbf{Y}_{z,World|RAS} = \begin{bmatrix} \cos(\xi_z) & -\sin(\xi_z) & 0 \\ \sin(\xi_z) & \cos(\xi_z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (\text{B.3})$$

also referred to as yaw [3]. The variables ξ_x , ξ_y and ξ_z define the counter-clockwise rotation of the plane about the x-, y- and z-axes respectively. These matrices can be combined to form a rotation matrix such that

$$\mathbf{Y}_{World|RAS} = \mathbf{Y}_{x,World|RAS} \mathbf{Y}_{y,World|RAS} \mathbf{Y}_{z,World|RAS} \quad (\text{B.4})$$

will map a vector from $\mathfrak{R}_{RAS}$ to $\mathfrak{R}_{World}$. A translation vector [3]

$$\mathbf{T}_{World|RAS} = \begin{bmatrix} x_T \\ y_T \\ z_T \end{bmatrix} \quad (\text{B.5})$$

must also be defined to translate the rotated point to a its position in the world frame of reference. The vector $\vec{f}_3$ can be mapped into $\mathfrak{R}_{World}$ using

$$\vec{F}_3[n] = \mathbf{Y}_{World|RAS} \vec{f}_3[n] + \mathbf{T}_{World|RAS} \quad (\text{B.6})$$

and the vector $\vec{F}_3[n]$ can be mapped back from $\mathfrak{R}_{World}$ to $\mathfrak{R}_{RAS}$ using

$$\vec{f}_3[n] = \mathbf{Y}_{RAS|World} \vec{F}_3[n] + \mathbf{T}_{RAS|World} \quad (\text{B.7})$$

which can also be written as

$$\vec{f}_3[n] = \mathbf{Y}_{World|RAS}^{-1} \vec{F}_3[n] - \mathbf{T}_{World|RAS} . \quad (\text{B.8})$$

In this equation,

$$\mathbf{Y}_{World|RAS}^{-1} = \mathbf{Y}_{z,World|RAS}^{-1} \mathbf{Y}_{y,World|RAS}^{-1} \mathbf{Y}_{x,World|RAS}^{-1} \quad (\text{B.9})$$

where all of the matrices are orthogonal so all matrix inverses can be replaced with matrix transpositions

B.2 Laser Vector Representation

The ray $\vec{F}_3[n]$ can be represented as the difference between a vector $\vec{O}[n]$ and a vector $\vec{Y}[n]$ all in $\mathfrak{R}_{World}$. The former defines a vector between the origin of $\mathfrak{R}_{World}$ and the origin of $\vec{F}_3[n]$. The latter defines a vector between the origin of $\mathfrak{R}_{World}$

and $\mathbf{Y}[n]$ which is the point at which $\vec{F}_3[n]$ intersects a surface in the environment.

This can be restated as

$$\vec{F}_3[n] = \vec{\mathbf{Y}}[n] - \vec{O}[n] \quad (\text{B.10})$$

which can be rewritten as

$$k[n] \vec{f}_u[n] = \vec{\mathbf{Y}}[n] - \vec{O}[n] \quad (\text{B.11})$$

where $\vec{f}_u[n]$ is the unit vector of $\vec{F}_3[n]$ [23, p.656]. The variable $k[n]$ is a multiplier defining the length of the vector $\vec{F}_3[n]$ and is the distance between $\mathbf{F}_3[n]$ and $\mathbf{Y}[n]$.

The vector $\vec{F}_3[n]$ is currently expressed as a position and an angular deviation ($U[n], V[n]$). Neither $\vec{\mathbf{Y}}[n]$ nor $\vec{f}_u[n]$ is known but the latter can be calculated by estimating $\mathbf{Y}[n]$ for a surface 1-metre from the scanner along the z-axis. Recall that $F_{z3}[n] = 0$ and

$$\mathbf{Y}'[n] = \begin{bmatrix} F_{x3}[n] + \tan(U[n]) \\ F_{y3}[n] + \tan(V[n]) \\ 1 \end{bmatrix} \quad (\text{B.12})$$

can be defined as a theoretical termination point. The unit vector can therefore be defined as

$$\vec{f}_u[n] = \frac{\vec{\mathbf{Y}}'[n] - \vec{O}[n]}{|\vec{\mathbf{Y}}'[n] - \vec{O}[n]|}. \quad (\text{B.13})$$

Equation (B.11) is only in two unknowns of which $\vec{\mathbf{Y}}[n]$ will be known when a point of intersection with an object in the environment is located. From this $k[n]$ can be calculated to represent the distance to the point of intersection.

Note that the sign of $k[n]$ depends on the direction of $\vec{F}_3[n]$ so it will be positive only for points along the laser path. Negative values indicate a theoretical point of

intersection behind the camera so can be immediately discarded as not applicable. This means that $k[n]$ can have any value between and including 0 and ∞ .

B.3 Planar Intersections

The environment can be defined as a series of planar surfaces of limited extent. Each plane bounded by at least three vectors $\vec{c}_0$, $\vec{c}_1$ and $\vec{c}_2$ representing the extents of the plane. The plane can be defined as a combination of a normal vector and a point on the plane. The normal to the plane is defined as

$$\vec{n} = (\vec{c}_1 - \vec{c}_0) \times (\vec{c}_2 - \vec{c}_0) \quad (\text{B.14})$$

which is the normal at the point $\mathbf{c}_0$ [23, p.660]. Note that this is a plane of infinite extent. Once the point of intersection, if any, is determined between $k[n] \vec{f}_u[n]$ and the plane then it must be determined whether this point lies within the extents of the planar surface.

Suppose the point of intersection $\vec{Y}[n]$ exists on the planar surface. In this case

$$0 = \vec{n} \cdot (\vec{Y}[n] - \vec{c}_0) \quad (\text{B.15})$$

is also true [23, p.659]. From (B.11)

$$\vec{Y}[n] = k[n] \vec{f}_u[n] + \vec{O}[n] \quad (\text{B.16})$$

so (B.15) becomes

$$0 = \vec{n} \cdot (k[n] \vec{f}_u[n] + \vec{O}[n] - \vec{c}_0) . \quad (\text{B.17})$$

Solving for $k[n]$ yields

$$k[n] = \frac{\vec{n} \cdot (\vec{c}_0 - \vec{O}[n])}{\vec{n} \cdot \vec{f}_u[n]} . \quad (\text{B.18})$$

which defines the distance between $\vec{O}[n]$ and $\vec{Y}[n]$, which in turn is the length of $\vec{F}_3[n]$.

If $k[n] \notin \{0, \infty\}$ then $\mathbf{Y}[n]$ exists on the planar surface. If $k[n] \in \{0, \infty\}$ then there are two possible interpretations depending on the value of the numerator. If the numerator of (B.18) is non-zero then there is no solution because $\mathbf{f}_d[n]$ does not both exist on the plane and on the line so $k[n]$ is undefined. If the numerator of (B.18) is zero then there is an infinite number of possible solutions. In either case it can be assumed that the surface was not intersected and $k[n]$ is undefined.

B.4 Worlds, Regions and Objects

The environmental model consists of a World model and one or more Object models. The world model consists of six bounded planes that together completely enclose the area in which the scanner and the Objects exist. Each Object Ω_i consists of a series of m planar surfaces and is enclosed by a spherical Region that defines the area within which all the planar surfaces can be found. The goal is to determine which planar surface results in an intersection with $\vec{F}_3[n]$ and is closest to the scanner. This is accomplished by first determining if $\vec{F}_3[n]$ intersects the Region of any object, then determining if any planar surface of that object is intersected. Region intersection does not guarantee surface intersection so all intersected Regions must be tested.

Consider a Region surrounding an Object Ω_i defined as a sphere of radius r_i

with centre $\vec{c}_i = (x_i, y_i, z_i)$. The centre of the sphere is defined as the average position based on the minimum and maximum axis values. Each Object Ω_i is defined by a set of m vertices given by $\mathbf{V}_i$. Each vertex is a point $\vec{v}_{i,k} = (x_{i,j}, y_{i,j}, z_{i,j})$ where $j \in [1...m]$ is a subscript denoting a given vertex. The centroid position can be found by

$$\mathbf{c}_i^T = \begin{bmatrix} \frac{\max_{j=1}^m \{x_{i,j}\} - \min_{j=1}^m \{x_{i,j}\}}{2} + \min_{j=1}^m \{x_{i,j}\} \\ \frac{\max_{j=1}^m \{y_{i,j}\} - \min_{j=1}^m \{y_{i,j}\}}{2} + \min_{j=1}^m \{y_{i,j}\} \\ \frac{\max_{j=1}^m \{z_{i,j}\} - \min_{j=1}^m \{z_{i,j}\}}{2} + \min_{j=1}^m \{z_{i,j}\} \end{bmatrix}. \quad (\text{B.19})$$

The radius is then found to be

$$r_i = \max_{j=1}^m \{|\vec{v}_{i,j} - \vec{c}_i|\}. \quad (\text{B.20})$$

It is possible for multiple Regions to be intersected so, at each time step, all regions are tested and the $k_{\min,i}[n]$ value of each region is selected. The $k_{\min,i}[n]$ value of each region can be found by determining the closest point of intersection of a ray with a sphere as described in [66, pp.1023-1025]. A sphere enclosing an Object Ω_i can be defined as

$$\Omega_i(\vec{\mathbf{Y}}_i[n]) : r_i^2 = (\vec{\mathbf{Y}}_i[n] - \vec{c}_i) \cdot (\vec{\mathbf{Y}}_i[n] - \vec{c}_i). \quad (\text{B.21})$$

Let

$$\vec{\mathbf{Y}}_i[n] = k_i[n] \vec{f}_u[n] + \vec{O}[n] \quad (\text{B.22})$$

so that

$$\Omega_i(\vec{\mathbf{Y}}_i[n]) : r_i^2 = (k_i[n] \vec{f}_u[n] + \vec{O}[n] - \vec{c}_i) \cdot (k_i[n] \vec{f}_u[n] + \vec{O}[n] - \vec{c}_i) \quad (\text{B.23})$$

which can be expressed as

$$\begin{aligned}\Omega_i(\vec{Y}[n]) : 0 &= k_i^2[n] (\vec{f}_u[n] \cdot \vec{f}_u[n]) \\ &+ 2 k_i[n] (\vec{f}_u[n] \cdot \vec{O}[n] - \vec{f}_u[n] \cdot \vec{c}_i) \\ &+ (\vec{O}[n] \cdot \vec{O}[n] - 2 \vec{O}[n] \cdot \vec{c}_i[n] + \vec{c}_i[n] \cdot \vec{c}_i[n]) - r_i^2.\end{aligned}\tag{B.24}$$

The minimum $k_i[n]$ can be found using

$$k_{min,i}[n] = \min \left\{ \frac{-B[n] \pm \sqrt{B[n]^2 - 4A[n]C[n]}}{2A[n]} \right\}\tag{B.25}$$

where

$$A[n] = \vec{f}_u[n] \cdot \vec{f}_u[n] = 1,\tag{B.26}$$

$$B[n] = 2 (\vec{f}_u[n] \cdot \vec{O}[n] - \vec{f}_u[n] \cdot \vec{c}_i)\tag{B.27}$$

and

$$C[n] = (\vec{O}[n] \cdot \vec{O}[n] - 2 \vec{O}[n] \cdot \vec{c}_i[n] + \vec{c}_i[n] \cdot \vec{c}_i[n]) - r_i^2.\tag{B.28}$$

The $k_{min,i}[n]$ values are sorted with negative and infinite values removed. The former indicates that the Region is not in front of the scanner and the latter indicates that the laser did not intersect the Region [66, pp.1023-1025].

Each Object enclosed by a Region with a positive non-infinite $k_{min}[n]$ is tested to determine if any planes are intersected starting with the Region that yielded the lowest positive non-infinite $k_{min}[n]$ -value. If no plane is intersected then the Region with the next-lowest positive non-infinite $k_{min}[n]$ -value is tested. If no Region yields a planar intersection then each of the planar surfaces comprising the extents of the World model are tested to determine which is intersected.

Appendix C: Model Calibration

C.1 Model Calibration

The model as described produces results that approximate those generated by a typical auto-synchronizing laser range scanner. Validation of the model requires comparing the output of the model to that generated by a real scanner. Before validation can take place the output of the model must be scaled to produce results that approximate those generated by a real scanner. The model does not generate results that duplicate that generated by a real scanner because the model is an idealized system. As a result a discrepancy is expected between simulated and real results.

Calibration was performed using two laser range scanners available at the NRC. We refer to these scanners as the Calib scanner and the Space40 scanner to distinguish between them. A calibration model was generated for each scanner by determining appropriate galvanometer and peak scaling factors for each system.

C.2 Mirror Calibration

Calibration of the galvanometers driving each mirror consisted of collecting data from each scanner at the galvanometer extents, calibrating the y-axis mirror galvanometer settings, then calibrating the x-axis mirror galvanometer settings. Table C.1 and Table C.2 show the galvanometer extents obtained from the Space 40 and Calib scanners respectively. The Cartesian measurement is the position of the detected point in space with respect to the scanner. It is assumed that the origin of the measured data is at the point of intersection of the laser with the axis mirror. In practice the measured origin may vary slightly from this assumption resulting in a small distortion which varies with range and mirror angle.

Table C.1: Space 40 Maximum and Minimum galvanometer measurements

	Cartesian			Galvanometer
	X (mm)	Y (mm)	Z (mm)	Reading (count)
X-mirror Maximum	-998.1	-142.1	3705.8	4643
X-mirror Minimum	988.2	40.9	3733.7	-5154
Y-mirror Maximum	-202.7	1206.1	3513.8	14179
Y-mirror Minimum	8.8	-1326.3	3670.6	-14639

Table C.3 shows the results of calibrating the x-axis and y-axis mirror galvanometers for the Space 40 and Calib scanners respectively. In each case the y-axis mirror galvanometer settings were corrected before calibrating the x-axis mirror galvanometer settings. The maximum and minimum angles are the mirror offset angles with respect to the (x,z)-plane. The maximum deflection is $1/2$ the observed rotational range of the mirror and is used to determine the new mirror

Table C.2: Calib Maximum and Minimum galvanometer measurements

	Cartesian			Galvanometer
	X (mm)	Y (mm)	Z (mm)	Reading (count)
X-mirror Maximum	-960.5	-169.6	4050.0	4505
X-mirror Minimum	504.1	-86.7	1981.2	-4778
Y-mirror Maximum	85.6	642.7	1797.9	11159
Y-mirror Minimum	60.7	-928.5	2599.2	-14666

angle β .

Table C.3: Galvanometer calibration results

	Maximum	Minimum	New Mirror	Maximum
	Angle	Angle	β	Deflection
Space 40 X-mirror	52.729°	38.144°	45.4365°	7.2925°
Space 40 Y-mirror	54.157°	34.747°	44.4525°	9.7050°
Calib X-mirror	51.866°	38.872°	45.3690°	6.4970°
Calib Y-mirror	54.222°	34.719°	44.4705°	9.7515°

Table C.4 shows the error observed when the model was used to predict the position of the points associated with the galvanometer extents. In each case the z-axis measurement of each point and the galvanometer positions were used to drive the model. The x-axis and y-axis measurements generated by the model were then compared to measured values. The x-axis mirror errors are the difference between the simulated and measured x-axis Cartesian measurements. The y-axis mirror errors are the difference between the simulated and measured y-axis Cartesian

measurements. In all cases the observed errors were less than 1-millimetre.

Table C.4: Position error after galvanometer calibration

	X-axis Mirror Maximum	X-axis Mirror Minimum	Y-axis Mirror Maximum	Y-axis Mirror Minimum
Space 40	0.024432-mm	0.097050-mm	0.073635-mm	0.049225-mm
Calib	0.012666-mm	0.009098-mm	0.038361-mm	0.002754-mm

C.3 Peak Detector Calibration

Peak Detector calibration was performed to obtain a set of measurements for the simulated CCD array which would result in peak values that closely approximate those obtained using a real system. The process assumes that the x-axis and y-axis mirror galvanometers are accurately calibrated. The calibration process involved obtaining UVP and distance measurements for points close to the maximum and minimum operating range of the scanner.

Table C.5: Maximum and Minimum peak detector measurements

	UVP			Distance to point (mm)
	a_Θ	a_Φ	P	
Space 40 Peak Maximum	-762	-1395	31052	9501.0
Space 40 Peak Minimum	-470	-1594	18062	1063.9
Calib Peak Maximum	-346	-2451	30448	9261.5
Calib Peak Minimum	1610	-1682	17031	1049.0

Table C.6: Results of CCD calibration

	Minimum Peak Position (mm)	Maximum peak Position (mm)	CDD Array Length (mm)	CCD Array Offset (mm)
Space 40	16.335	1.832	36.583224	-0.083446
Calib	16.744	1.881	36.299579	-0.689280

Table C.5 shows the results of measurements obtained from the Space 40 and Calib scanners. The Distance measurement is the z-axis measurement from the Cartesian representation of the detected point.

Table C.6 shows the results obtained during the calibration process. The minimum and maximum peak positions are the distance along the CCD plane to the point of intersection of the point path from the z-axis. CCD array length is the length of the simulated CCD array which would generate the observed peak values.

C.4 Linear Fit

Ideally the galvanometer and peak detector calibrations should result in a perfect fit between simulated and measured results. The goal of the simulation is to represent the type of output that could be generated by a typical laser range scanner. When calibrated to a given scanner the simulation that should, for a given set of galvanometer readings, generate a peak value that closely approximates the peak value generated by that real scanner. The mirror and peak detector phases of the calibration process should minimize the discrepancy between simulated and measured peak values for a given set of galvanometer readings. However, the model

does not consider all possible variables that could significantly affect the accuracy of the simulation. A linear fit between the simulated and measured peak values provides a way to evaluate the “goodness of fit” between the model and the real system. Moreover, it provides a way to improve that fit.

Table C.7: Regression analysis of linear relationship for Space 40 calibration model

Variable	Beta	Coeff	SE	z-value	Prob(z)
Intercept	b_0	95.056960	2.96	32.07	0.000
Simulated Peak ($\tilde{P}$)	b_1	0.997335	0.00	8718.35	0.000
X-galvanometer (a_Θ)	b_2	0.022589	0.00	29.40	0.000
Y-galvanometer (a_Φ)	b_3	0.004099	0.00	8.46	0.000

Table C.8: Regression analysis of linear relationship for Calib calibration model

Variable	Beta	Coeff	SE	z-value	Prob(z)
Intercept	b_0	143.590417	5.93	24.21	0.000
Simulated Peak ($\tilde{P}$)	b_1	0.994568	0.00	4722.61	0.000
X-galvanometer (a_Θ)	b_2	-0.044113	0.00	-33.24	0.000
Y-galvanometer (a_Φ)	b_3	0.005008	0.00	6.67	0.000

Table C.7 and Table C.8 show the results of fitting the simulated peak and galvanometer values to the measured peak values obtained using the same galvanometer values. The Space 40 results were based on 1285 samples between 1-metre and 9-metres from the scanner. The Calib results were based on 2241 measurements within the same range.

Linear regression generates a “best fit” line through the data by minimizing the

sum of squared error between the observed points and those predicted using the “best fit” line [21, pp.10-16]. The result is a probabilistic model which attempts to predict the peak value P generated by the real scanner [20, p.440]. In this case it is assumed that the peak values $\tilde{P}$ generated by the simulation and the galvanometer values a_Θ and a_Φ used to obtain P and $\tilde{P}$ should be good predictors of P . The peak value $\tilde{P}$ generated by the simulated is referred to as the *simulated peak* and the peak value P obtained from the real scanner is referred to as the *real peak*. The galvanometer values a_Θ and a_Φ were obtained from the real scanner but were also used to drive the simulation.

A linear model has the form

$$y = \sum_{i=0}^k x_i \beta_i + \varepsilon \quad (\text{C.1})$$

where $x_0 = 1$ and ε is a zero-mean random error variable and k is the number of samples independent variables [20, pp.440] [20, pp.448-449]. Equation (C.1) can be re-expressed as matrix equation of the form

$$\mathbf{Y} = \mathbf{X}\beta + \varepsilon. \quad (\text{C.2})$$

If $\mathbf{Y}$ and $\mathbf{X}$ are known then (C.2) can be rearranged to solve for β where $\varepsilon = \mathbf{0}$.

This has the form

$$\hat{\beta} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y} \quad (\text{C.3})$$

where it can be shown that $\hat{\beta}$ is an unbiased estimator of β in the least squares sense [20, pp.449-456]. Specifically, $\hat{\beta}_0$ is the Y-intercept of the linear model and $\hat{\beta}_i; i \in \{1..N\}$ are the slopes of each of the variables in the linear model. The vector $\hat{\beta}$ contains the estimates of the model parameters.

The **Coef** column in Table C.7 and Table C.8 are the slope (b_0) and intercepts (b_1 to b_3) of the regression equation that defines the “best fit” line. These are referred to as the regression coefficients [21, pp.13-22]. Specifically

$$\hat{P} = b_0 + b_1 \tilde{P} + b_2 a_{\Theta} + b_3 a_{\Phi} \quad (\text{C.4})$$

predicts the values of P based on known $\tilde{P}$, a_{Θ} and a_{Φ} values. The simulated peak variable $\tilde{P}$ and galvanometer readings a_{Θ} and a_{Φ} are referred to as the independent variables or parameters and the real peak variable is referred to as the dependant variable. The goal is for the predicted peak values $\hat{P}$ to be as close to P as possible. The coefficients b_0 to b_3 are the parameter estimates generated using (C.3) where $\mathbf{Y}$ is the vector of real peak values $\mathbf{P}$ and the columns of $\mathbf{X}$ are a vector of ones ($\mathbf{1}$), the simulated peak values ($\tilde{\mathbf{P}}$) and the galvanometer measurements ($\mathbf{a}_{\Theta}$ and $\mathbf{a}_{\Phi}$) in that order. This can be expressed as

$$\mathbf{b} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{P} \quad (\text{C.5})$$

where $\mathbf{X} = [\mathbf{1} \ \tilde{\mathbf{P}} \ \mathbf{a}_{\Theta} \ \mathbf{a}_{\Phi}]$.

The **SE** column lists the standard error associated with each of the regression coefficients. It relates the variability of the estimate to the variability of data associated with the regression coefficient [21, pp.22-25]. The standard error of the estimate can be found using

$$s_{y|x} = \sqrt{\frac{SS_{res}}{N - (k + 1)}} \quad (\text{C.6})$$

where N is the number of samples, k is the number of sample groups and

$$SS_{res} = \sum_{i=0}^{N-1} (y_i - \hat{y}_i) \quad (\text{C.7})$$

is the sum of squared residuals, also referred to as the sum of squared error [21, p.57] [20, p.463]. Equation (C.7) can be written in matrix form as

$$SS_{res} = \mathbf{Y}'\mathbf{Y} - \hat{\beta}\mathbf{X}'\mathbf{Y} . \quad (\text{C.8})$$

The variable N is the number of samples used to calculate SS_{res} [20, pp.460-465]. The standard error for the linear model of the scanner can be expressed as $s_{P|X}$ or the standard error of P given $X = [\tilde{P} \ a_{\Theta} \ a_{\Phi}]$. The standard error $s_{P|X}$ is a measure of the variability of the P about $\hat{P}$ so is used in the next section to evaluate each stage in the calibration process.

The standard error of the estimate is used to obtain the standard error of each the regression coefficients. In this case the standard errors are associated with the intercept (s_{b_0}), the simulated peak (s_{b_1}), and the galvanometer values (s_{b_2} and s_{b_3}). Each can then be tested to determine whether they are significantly different than zero using

$$z_i = \frac{b_i}{s_{b_i}} \quad (\text{C.9})$$

for each parameter. The results can be found in the **z-value** column of Table C.7 and Table C.8. Glantz [21, pp.26-27] [21, pp.57-59] recommended using the t-statistic but applied it to data consisting of a small number of samples. The z-statistic approximates the t-statistic closely when samples sizes are greater than 30. In this case there were more than 1000 samples in each data set so the z-statistic was used instead of the t-statistic [20, p.334] [20, p.464]. The z-statistic and the sample size were used to generate the probability that the variable significantly affected the standard error of the estimate. Variables with a probability of less than 0.05 were considered to significantly influence the dependant variable. The probability based on the z-statistic can be found in the **Prob(z)** column.

The standard error of a regression coefficient is found using

$$s_{b_i} = \sqrt{s_{Y|X}^2 c_{i,i}} \quad (\text{C.10})$$

where $s_{Y|X}^2 c_{i,i}$ is the variance of the i^{th} regression coefficient [20, pp.463-466]. The coefficient variances are found along the diagonal of the covariance matrix that can be found using

$$Cov\{\beta\} = s_{Y|X}^2 (\mathbf{X}'\mathbf{X})^{-1} \quad [20, \text{pp.456} - 459]. \quad (\text{C.11})$$

Ideally $\tilde{P}$ should contribute significantly to predicting P . Specifically the coefficient associated with $\tilde{P}$ should be 1 if the model were able to predict P accurately. Moreover a_Θ and a_Φ should not contribute significantly to predicting P . In this case the coefficients associated with a_Θ and a_Φ should be zero. However, all parameters for both scanners were shown to have a significant effect ($p=0.000$ in all cases) at $p < 0.05$ so the galvanometers still could significantly influence the prediction error.

Table C.9: Regression analysis of linear relationship for Space 40 calibration model to confirm results

Variable	Beta	Coeff	SE	z-value	Prob(z)
Intercept	b_0	0.533378	2.97	0.18	0.858
Simulated Peak ($\tilde{P}$)	b_1	0.999999	0.00	8719.71	0.000
X-galvanometer (a_Θ)	b_2	0.000004	0.00	0.00	0.996
Y-galvanometer (a_Φ)	b_3	-0.000013	0.00	-0.03	0.979

A regression equation was developed for each scanner using the regression coefficients to improve results generated by each model. The equation for the

Table C.10: Regression analysis of linear relationship for Calib calibration model to confirm results

Variable	Beta	Coeff	SE	z-value	Prob(z)
Intercept	b_0	0.521604	5.96	0.09	0.930
Simulated Peak ($\tilde{P}$)	b_1	0.999998	0.00	4723.72	0.000
X-galvanometer (a_Θ)	b_2	-0.000029	0.00	-0.02	0.982
Y-galvanometer (a_Φ)	b_3	0.000001	0.00	0.00	0.998

Space 40 scanner was

$$\hat{P} = 95.056960 + 0.997335\tilde{P} + 0.022589a_\Theta + 0.004099a_\Phi \quad (\text{C.12})$$

and for the Calib scanner the equation was

$$\hat{P} = 143.590417 + 0.994568\tilde{P} - 0.044113a_\Theta + 0.005008a_\Phi . \quad (\text{C.13})$$

The regression analysis was run a second time using the new peak values. Table C.9 and Table C.10 show that all parameters except the peak value are non-significant at $p < 0.05$. Specifically, the probabilities associated with a_Θ and a_Φ are $p = 0.996$ and $p = 0.979$ for the Space 40 data in Table C.9, and $p = 0.982$ and $p = 0.998$ for the Calib data in Table C.10. This means that a_Θ and a_Φ no longer significantly influence the variability of the prediction. Moreover, the peak parameter has a coefficient of approximately 1.0 indicating that $\tilde{P}$ has a 1-to-1 relationship with P .

C.5 Calibration Process Results

The model improvement resulting from each stage in the calibration process was observed by noting the reduction in the standard error of the estimate ($s_{P|X}$) obtained by fitting a linear model to the data using linear regression. The data does not follow a strictly linear distribution but the standard error provides an estimate of the unexplained variance remaining in the model.

Table C.11: Improvement in model fit at each stage in the calibration process

	Space 40		Calib	
	$s_{P X}$	(N=1285)	$s_{P X}$	(N=2241)
Uncorrected results	21.138	N/A	52.531	N/A
After galvanometer calibration	17.470	p= 0.000	40.044	p= 0.000
After peak detector calibration	17.482	p= 0.495	40.027	p= 0.504
After linear fitting	17.480	p= 0.501	40.018	p= 0.502
Using only linear fit	21.140	p= 0.501	52.512	p= 0.497

The first row of Table C.11 shows the $s_{P|X}$ for each scanner after fitting a linear model to the uncalibrated model data. This was used as a reference point for comparison to the calibrated results. The last row shows the results of fitting a linear model to model data which has been corrected using only the method described on Section C.4. The standard error of the linear fit results were compared to the $s_{P|X}$ of the uncorrected data using an F-test and were found to not be significantly different at $p < 0.05$.

The third, fourth and fifth rows show the results of implementing each stage in the calibration process. The third row shows the results after only performing the

mirror calibration stage (Section C.2). This stage showed a significant reduction in $s_{P|X}$ for both scanners when compared to the $s_{P|X}$ of the uncorrected results at $p < 0.05$.

The third row shows the results of calibrating the peak detector (Section C.3) after performing the galvanometer calibration. Implementing the galvanometer calibration stage did not result in a significant decrease in $s_{P|X}$ at $p < 0.05$ when compared to the $s_{P|X}$ after performing only the mirror calibration.

The fourth row shows the results of performing linear fitting (Section C.4) after performing galvanometer and peak detector calibrations. Linear fitting (Section C.4) did not result in a significant reduction in $s_{P|X}$ at $p < 0.05$ when compared to the combination of galvanometer and peak detector calibration.

The effect of each stage of calibration can be seen more clearly by examining the results at each stage of the calibration process. Figure C.1 and Figure C.2 show the effects of performing the mirror calibration stage (Section C.2) and the final results after all stages have been completed. These are compared to the results of only performing linear fitting (Section C.4). Figure C.3 shows the effects of performing peak detector calibration after the mirror calibration stage (Section C.2) and the final results after all stages have been completed. These are compared to the results of only performing linear fitting (Section C.4). Results have only been shown for the Calib scanner but the explanations are equally applicable to Space 40 scanner results.

Graph (a) in Figure C.1 shows the x-axis mirror position errors for the Calib scanner without performing any calibration. Graph (b) shows the improvement resulting from performing only the galvanometer calibration. This can be compared to the graph (d) that shows the results of performing only a linear fit. Graph

(c) shows the final position errors after the all stages of the calibration process have been completed. Little difference is expected to be seen between graph (b) and graph (c) because most of the galvanometer errors are handled in the mirror calibration stage.

A comparison of the graphs (a) and (d) of Figure C.1 reveals that linear fitting along has little discernable effect which supports the results in Table C.11. However, a comparison of the graph (a) with the graph (b) shows that the position error is noticeably reduced, supporting the results in Table C.11. A comparison of the graph (b) with graph (c) from the top shows that performing the mirror calibration followed by the peak detector calibration and linear fitting does not significantly affect the results as seen in Table C.11.

Graph (a) in Figure C.2 shows the y-axis mirror position errors for the Calib scanner without performing any calibration. Graph (b) shows the improvement resulting from performing only the galvanometer calibration. This can be compared to the graph (d) that shows the results of performing only a linear fit. The graph (c) shows the final position errors after the all stages of the calibration process have been completed. The conclusions drawn from comparing these graphs are identical so those discussed for the X-axis mirror.

Graph (a) of Figure C.3 shows the peak detector errors for the Calib scanner without performing any calibration. Peak errors are measured in units of CCD pixel widths which are $1/64$ of the peak detector value. This is based on the assumption that the scanners use a 512-element CCD array. Graph (b) shows the improvement resulting from performing only the peak detector calibration. This can be compared to the graph (d) that shows the results of performing only a linear fit. Graph (c) shows the final position errors after the all stages of the

calibration process have been completed.

A comparison of the graph (a) and graph (d) of Figure C.3 reveals that linear fitting alone does significantly reduce peak fit errors. This supports results presented by MacKinnon *et al.* [62] in which only a linear peak fit was performed. A comparison of the graph (a) with the graph (b) from the top shows that performing a peak detector calibration followed by peak detector calibration results in a significant reduction in peak errors. Graph (c) from the top shows that linear fitting results in further small reduction in peak errors.

Figure C.1: Calib X-axis mirror position errors

Figure C.2: Calib Y-axis mirror position errors

Figure C.3: Calib peak errors

Appendix D: Edge Metrics

D.1 Overview of Edge Metrics

Edge Metrics are measures of the effectiveness or performance of an edge detection method. Pratt [29, pp.485-498] indicated that edge detection performance could be measured using objective or subjective metrics. Objective performance measures provide numerical ratings of performance under specific conditions based on quantitative experimental results. Subjective measures involve a human evaluating the performance of an edge detector based on some qualitative measure. Peli and Malah [27] demonstrated the application of both objective and subjective measures in evaluating edge detection methods in their 1982 study of edge detection algorithms.

The edge detection metrics described in this chapter are generally used to evaluate edge detection methods applied to 2-D data. In this thesis they are only applied to 1-D range data.

D.2 Objective Performance Measures

Trucco and Verri [2, pp.80-81] proposed that the most important evaluation criteria that could be applied to edge detectors was whether they improved the performance of the system. They divided edge detection performance evaluation procedures into theoretical and experimental evaluations. The latter would include the number of false edges detected, the number of true edges that were missed, and the error in position and orientation between the true and detected edges.

Jain *et al.* [64, pp.176-179] proposed a more extensive list of edge performance criteria. Criteria for the evaluation of edge detection performance were divided into three classifications. The first classification was performance as an edge detector that consisted of measuring the probability of detecting false edges and of missing true edges. The second classification was performance in locating edge position and orientation that consisted of measuring the error of edge angle estimate and the mean square difference between the detected and true edge. The third classification performance when the edges did not match the ideal model which consisted of measuring the tolerance of the edge detector to feature complexities such as edge distortion, corners and junctions.

Jain *et al.* [64, pp.176-179] outlined a two-stage approach to performing a simple evaluation of edge detection algorithms. These stages corresponded to the first and second classifications of edge detection methodology.

The first stage consisted of counting the number of false edges detected in each scan and the number of true edges that were missed. The edges detected using a synthetic ideal image is used to provide the number of true edges. The ideal

image is then modified by adding noise or through distorting the edge positions. The edges detected in the test image are matched in a 1-to-1 correspondence to edges detected using the ideal image. Any detected edges remaining unmatched would be classified as false edges and any true edges remaining unmatched would be classified as missed edges [64, pp.176-179].

Jain *et al.* [64, pp.423-428] advocated using disparity analysis to perform matching between true and detected edges. The disparity between points would be defined as a displacement vector that would define a correspondence between two points. The process of matching would be guided by three properties. The discreteness property emphasized that the features under examination should be isolated. This means that the points should be unique compared to its immediate neighbours. For example, a point on a line would not be unique but a corner would be unique. The similarity property would measure of how much two features resemble each other. The consistency property emphasized that the match should conform to previous matches. Within the context of the thesis, the similarity between two points is considered to be the absolute or mean-squared distance between each true edge and the detected edges closest to it on either side. A threshold level is selected to eliminate those edges that are greater than some predefined minimum distance from the true edge.

The second stage of Jain's approach was to compare the locations of each of the matched edges detected in the first stage with the locations of their corresponding true edges. The error variance associated with a particular image would be the sum of squared errors divided by $n - 1$ where n was the number of edges that were matched [64, pp.423-428].

Pratt [29, p.485] emphasized distinguishing between information that must be

obtained from the edge detector and information that could be obtained from the edge detector. For example, pixel location was defined as mandatory information whereas edge height could be considered optional information. Pratt [29, pp.486-487] proposed that edge localization be an important criterion in measuring the performance of edge detectors. Applying the edge detection method to a unit width ramp edge and measuring the displacement from the centre of the ramp was suggested as one way to evaluate edge localization.

D.3 Figure of Merit

Jain *et al.* [64, pp.423-428] described a figure of Merit (*FoM*) originally proposed by Pratt [29, pp.490-493] that would combine the three types of errors commonly encountered in edge detection. These errors are missed edge error, edge localization error and misclassification error. Missed edge error is associated with true edges that were not detected. Edge localization error is associated with difference between the true edge location and the location of its matched detected edge. Misclassification error is associated with the detection of edges that do not match any true edge. The proposed *FoM* was summarized as

$$FoM = \frac{1}{\max(I_A, I_I)} \sum_{i=1}^{I_A} \frac{1}{1 + d_i \alpha^2} \quad (D.1)$$

where I_A was defined as the number of detected edges, I_I was defined as the number of true edges, d was defined as distance between matched detected and true edges and α was a penalization parameter to weight the effect of the error associated with displaced edges. The signal-to-noise ratio (*SNR*), computed using (D.3), would then be plotted against the *FoM* to display the performance of the

detector under varying noise conditions. Pratt [29, pp.490-493] used as an example a weighting parameter of $\alpha = 1/9$ to demonstrate application of the *FoM* to evaluating edge detectors.

Van der Heyden [28] suggested that the *FoM* proposed by Pratt [29, pp.490-493] was based on an incomplete set of criteria. First, the *FoM* could not be used to tune the edge-detection method to find an optimal balance between edge detection and noise suppression. Second, the criteria had not been designed to test edges formed by curved contours. Finally, the *SNR* is actually a function of the range, but the *FoM* did not account for the variability in the *SNR*. Van der Heyden suggested a more complex *FoM* based upon the standard deviation of a number of parameters. These parameters corresponded to the granularity of the image, the distribution of grey-level variation, the image blur and the signal noise level and a weighting factor for these errors.

To evaluate the performance of an edge detector, a reference image was selected and edge detection was performed to provide a reference edge image. A low-pass filtered Gaussian noise process with a threshold level of zero was added to the image to simulate granularity. The luminance was then varied by a systematic adjustment to the grey level of each image region. A Gaussian filter was then applied to the result to provide blurring along with an additive Gaussian noise process. Edges were detected in the resulting image to provide a test edge image [28].

The reference and test edge images were combined using a classifier. The classifier assigned edges that were detected correctly a value of zero, missed edges a value of 1 and false edges a value of -1. The classified image was then convolved with a Gaussian filter and the result was squared, summed and divided by the

number of pixels. The resulting formula would be as

$$FoM = \frac{1}{NM} \sum_{n=1}^N \sum_{m=1}^M g^2(n, m) \quad (D.2)$$

where N and M referred to the number of pixels defining the image width and height and the function $g()$ referred to the classified image. In range measurements the equivalent of variation in luminance would be variation in the height of the edge [28].

D.4 Edge Orientation, Gradient and Angle

The orientation of an edge with respect to the edge detector is expected to have a significant effect on the performance of the edge detector [29, pp.485-486] [27] [40]. As a result, evaluation of the detected versus the actual edge angle is another measure of edge detector performance [38] [41]. However, this measure is useful only when applied to 2-D window edge operators. The orientation of an edge with respect to a 1-D line scan is entirely dependant upon the path of the scan line, not the choice of edge operator or operator characteristics. For this reason, detected edge angle would not be useful measures of edge detector performance when applied to the results of a Lissajous scan.

Another factor that is affected by the orientation of the edge is the perceived slope of the edge. This may be measured by comparing the detected gradient of an edge to the slope of a line normal to the edge [38] [40] [41]. The 1-D equivalent of the gradient is the derivative that may be used to evaluate the perceived edge slope. Once again, the perceived slope is a function of the scan path so is not a valid method to compare edge detectors using a Lissajous scan pattern.

D.5 Signal to Noise ratio (SNR)

Van der Heyden [28] and Pratt [29, p.469] defined the SNR based on the square of the edge height divided by the noise variance. In this case the SNR was defined as

$$SNR = \frac{h^2}{\sigma^2} \quad (D.3)$$

where h was the height of the edge and σ was the standard deviation of the Gaussian white-noise process added to the ideal image. Staunton [38] preferred to describe the SNR in units of decibels. The latter SNR measure is the more common approach.

Figure D.1: Edge detection resolution at 1/4-CCD pixel

Before one can evaluate the performance of edge detectors one must determine the minimum height to be detected. One of the benefits of using peak values to detect edges is that the peak value has been automatically scaled for range

variations. Figure D.1 shows the relationship between minimum detectable edge height and range for a peak resolution of 1/4-CCD pixel. For purposes of this study we have selected this value as a reasonable minimum level of edge detection.

D.6 Subjective Performance Evaluation

Pratt [29, p.494] suggested that in some cases the best measure of edge detection performance was how well the edge boundaries matched with perceived image boundaries. Peli and Malah [27] defined three qualitative criteria to evaluate edges of which two would be applicable to the current study. One test involved classifying the results of edge detection as either a single edge or a double edge. Another test involved classifying the results of edge detection as distorted or undistorted.

D.7 Selection of Edge Detection Metrics

Edge detection metrics that could be applied to 1-D range data were selected for this study. Metrics evaluating edge orientation and slope were not being considered because both are dependant upon the angle of the scan path with respect to the edge normal. 2-D edge detectors can be designed to account for edge orientation, one example being the Canny edge detector [2, p.77] [34]. In these cases edge orientation can be determined within the context of the viewing window because multiple edge points can be detected. A 1-D edge detector has a viewing window that is restricted by the scan path. The edge orientation cannot be determined from a single intersection point so variations in edge slope are

irrelevant.

Peli and Malah [27] defined an edge as the midpoint of the edge slope. They used a variety of quantitative measures to evaluate edge detector performance. The measures have been summarized in Table D.1.

Table D.1: Measures of edge detection metrics used by Peli [27]

Formula	Description
$P_{True} = 100 \frac{I_{True}}{I_I}$	Percent detected edges coinciding with true edges
$P_{False} = 100 \frac{I_{False}}{I_I}$	Percent of detected edges not coinciding with true edges
$NSR_{True} = \frac{I_{False}}{I_{True}}$	Noise to Signal Ratio of detected edges that coincide with true edges
$MW = \frac{I_A}{I_I}$	Mean width of detected edge
$R_1 = \frac{1}{\max(I_A, I_I)} \sum_{i=1}^{I_A} \frac{1}{1 + d_i \alpha^2}$	Deviation of a detected edge from the true edge
$R_2 = \frac{1}{I_A} \sum_{i=1}^{I_A} d_i^2$	Average squared deviation of detected edge from true edge
$R_3 = \frac{1}{I_A} \sum_{i=1}^{I_A} d_i $	Average absolute deviation of detected edge from true edge

In Table D.1 I_{True} was the number of detected edges that coincide with a true edge, I_{False} was the number of detected edges that did not coincide with a true edge, I_I was the number of true edges, I_A was the number of edges detected, d was the distance between the detected edge and the true edge, and α was the

weighting coefficient for Pratt's *FoM* [27]. In this study the weighting factor was defined to be $\alpha = 1/9$.

Metrics examining edge slope or gradient were not considered. In a 2-D edge detector the edge gradient can be determined within the context of the window. A 1-D edge detector is limited to determining the first derivative of the edge slope that is dependant upon the scan path. Furthermore, the scan path of a Lissajous figure is non-linear so the determination of the edge slope is not a simple matter.

The effectiveness of an edge detector is generally measured according to Canny's [34] criteria. These criteria are good detection, good edge localization and only a single response for each edge [2, pp.73-74] [29, pp.490-491].

To provide good detection an edge detector must exhibit both a low probability of missing true edges and a low probability of reporting false edges [34] [2, pp.73-74]. Peli and Malah [27] in their 1982 survey of edge detection algorithms used the percent of ideal edges detected to measure the former and the percent of unmatched detected edges to measure the latter quantitatively. Optimally the former should be 100% while the latter should be 0%. Peli and Malah [27] also measured the Noise-to-Signal ratio (*NSR*) of unmatched detected edges to matched detected edges. Optimally this ratio should be 0 corresponding to the case of only ideal edges being detected. They also measured the mean width of the detected edge as the ratio of number of detected edges to the number of ideal edges [27]. Optimally the mean width should be 1 corresponding to the case of no more or less edges being detected than would exist in the ideal model. In this study these measures are referred to as accuracy metrics because they measure how close the number of detected edges is to the number of ideal edges.

Good localization involves ensuring that the locations of the detected edges

are as close as possible to the locations of the ideal edges [34] [2, p.74]. Peli and Malah [27] used both the average squared deviation between detected and ideal edges and the mean absolute deviation between detected and ideal edges points. In both cases the optimal value would be zero corresponding to no difference between detected and ideal edges [27]. In this study we refer to these measures as precision metrics because they measure the proximity of detected edges to true edges. The average squared deviation and the absolute deviation are functionally equivalent so we use the absolute deviation to simplify interpretation of the results.

Pratt [29, pp.490-494] proposed a figure of merit to combine edge precision measurement and edge accuracy measurement [27]. Optimally this measure would have a value of 1 when all true edges were detected, those detected edges coincided spatially with true edges, and no false edges were detected. Van der Heyden [28] proposed a more comprehensive evaluation system in which test images were specifically designed to test the robustness of an edge detector. Optimally this figure of merit should have a value of 0 when all edges are matched. In this study edge detector performance was generalized using Pratt's *FoM* and Heyden's *FoM*. Robustness was not evaluated using randomized images but the performance of each edge detector under tightly controlled conditions was performed.

Each detector was characterized by its response to varying edge height, varying surface distance, edge separation, and edge slope. The Figures of Merit were used to compare edge detector performance in general while proximity and accuracy measures were used to evaluate specific aspects of their performance. The criterion of a single response for each true edge was examined qualitatively rather than quantitatively. Peli and Malah [27] used a test in which subjects classified each edge as a single or double edge.

Table D.2: Summary of Edge Metrics used in this report

Test	Description
Percent True	Accuracy metric. Percent of all edges detected edges that coinciding with a true edge. Ideally this should be 100%.
Percent False	Accuracy metric. Percent of all edges detected that do not coinciding with any true edge. Ideally this should be 0%.
Noise-to-Signal Ratio	Accuracy metric. Ratio of the total number of detected edges that do not coincide with a true edges to the total number of detected edges that do coincide with a true edge. Ideally this should be 0.
Mean Signal Width	Accuracy metric. Ratio of the number of edges detected to the number of true edges. Ideally this should be 1.
Absolute Deviation	Precision metric. The absolute Euclidean distance in UVP-space between a detected edge and it's closest possible corresponding true edge. Distance is expressed as a fraction of the total scan width so the ideally maximum value is 1. The practical maximum is determined through observation. Ideally this should be 0.
Pratt's Figure of Merit	Overall measure of the "goodness" of the edge detector. Ideally this should be 1.
van der Heyden's Figure of Merit	Overall measure of the "goodness" of the edge detector. Ideally this should be 0.

Zhou et al [35] indicated that comparisons of edge detectors based on simulated results alone were of limited benefit so they used visual inspection to make a final selection. Heath *et al.* [31] [37] proposed a method to compare the results of edge detectors based on a seven-point rating scale. Panellists were provided with a series of images, each consisting of a single object in a familiar orientation, and were asked to rate the results of an edge detection operation [37]. Qualitative evaluation using object recognition [36] was not considered because even the ideal edge profile generated using a Lissajous scan would not provide enough information for a person to accurately identify an object. A qualitative evaluation of edge detector performance was not performed in a quantifiable manner but its application to edge detector evaluation has been included for completeness. Table D.2 shows the edge detection metrics used in this report.

Appendix E: Noise Analysis

E.1 Noise Analysis

The peak and galvanometer noise levels and distributions were examined for each of the scanners at the NRC. Noise was observed to consist of both additive noise and outliers that represent aperiodic impulses, referred to here as spikes. The observed noise for each scanner was first examined to determine whether the additive noise could be modelled as a Gaussian white noise process. The observed noise was then examined to determine the noise variance, in the case of additive noise, or rate, in the case of outliers.

E.2 Noise Distribution

Noise is generally modelled as a zero-mean Gaussian white noise process [2, p53]. The spread of a noise process can be modelled using the standard deviation of the observed noise process. According to the Central Limit Theorem if the observed noise consists of independent random samples from the same distribution then

$$U = \frac{\sqrt{n}(\bar{x} - \mu)}{\sigma} \tag{E.1}$$

where μ is the expected value of the noise process and σ is the expected spread of the noise process, then the distribution function of will converge to a Gaussian or standard normal distribution [20, p.281]. This means that the observed noise generated by each of the scanners must be examined to determine if they can be approximated using a Gaussian distribution.

Pearson kurtosis is one measure commonly used to examine the normality of a data set. However, kurtosis is considered to be highly optimistic so it is only useful to indicate serious departures from normality. Typically a t or z-test is used to compare the kurtosis to the expected value of 3. Specifically, for a large sample, given a kurtosis κ_4 and sample size N

$$z = \frac{\kappa_4 - 3}{\sqrt{24/N}} \quad (\text{E.2})$$

where 24 represents the variance of a kurtosis metric. Kurtosis can be found using

$$\kappa_4 = \frac{N \sum_{i=1}^N (x_i - \bar{x})^4}{\left(\sum_{i=1}^N (x_i - \bar{x})^2 \right)^2} \quad (\text{E.3})$$

where $\bar{x}$ is the mean of a sample set of size N . If $\kappa_4 > 3$ then the distribution is flat and broad while if $\kappa_4 < 3$ then the distribution is tall and narrow. Testing the hypothesis that $\kappa_4 = 3$ is generally only useful if the data set contains more than 1000 samples as it does in this case [61, pp626-627].

Table E.1 shows the peak, x-galvanometer and y-galvanometer kurtosis values. The p-value shown in brackets is the probability that the kurtosis measure is significantly different than the expect value of 3 using a z-test. In most cases the observed kurtosis values are significantly larger than the expected value of 3 indicating that the distribution is broader with a higher peak than a Gaussian distribution. However, this does not mean indicate that the data cannot

Table E.1: Kurtosis values of Calib and Space 40 data

Attribute	Sample	Space40	Calib
Peak	1	3.662785 (p=0.000)	3.645296 (p=0.000)
	2	4.544629 (p=0.000)	3.464397 (p=0.000)
	3	3.503421 (p=0.000)	6.680700 (p=0.000)
	4	3.628982 (p=0.000)	2.943741 (p=0.687)
X-galvo	1	4.443151 (p=0.000)	3.078661 (p=0.212)
	2	3.815054 (p=0.000)	3.327248 (p=0.003)
	3	4.139378 (p=0.000)	3.385538 (p=0.000)
	4	3.130003 (p=0.108)	3.285444 (p=0.007)
Y-galvo	1	3.199840 (p=0.021)	5.121627 (p=0.000)
	2	3.631624 (p=0.000)	3.009119 (p=0.469)
	3	4.225809 (p=0.000)	4.225164 (p=0.000)
	4	3.526358 (p=0.000)	3.161473 (p=0.080)

be approximated using a Gaussian distribution because the kurtosis measure is extremely sensitive to non-normal data. Generally the kurtosis measure used to warn that it is possible the data may not fit a normal distribution but other measures should also be used.

Pearson skewness is another measure used to warn of substantial deviations from normality. This measure is also considered to be highly optimistic so it is only useful to indicate serious departures from normality. Typically a t or z-test is used to compare the skewness to the expected value of 0. Specifically, for a large sample, given a skewness κ_3 and sample size N

$$z = \frac{\kappa_3}{\sqrt{6/N}} \quad (\text{E.4})$$

where 6 represents the variance of a skewness metric. Skewness can be found using

$$\kappa_3 = \sqrt{\frac{N \left(\sum_{i=1}^N (x_i - \bar{x})^3 \right)^2}{\left(\sum_{i=1}^N (x_i - \bar{x})^2 \right)^3}} \quad (\text{E.5})$$

where $\bar{x}$ is the mean of a sample set of size N . If $\kappa_3 > 0$ then the distribution is skewed toward the largest values while if $\kappa_3 < 0$ then the data is skewed toward the smallest values. Generally this test is only performed for data set with more than 150 samples such in the data sets used in this section [61, pp625-626].

The skewness and kurtosis measures are generally described as the third and fourth moments about the mean. Specifically

$$\kappa_3 = \frac{E\{(x - \mu)^3\}}{\sigma^3} \quad (\text{E.6})$$

and

$$\kappa_4 = \frac{E\{(x - \mu)^4\}}{\sigma^4} \quad (\text{E.7})$$

where μ is the population mean. In each case the expected value is scaled by the population variance σ [61, pp624-625].

Table E.2: skewness of Calib and Space 40 data

Attribute	Sample	Space40	Calib
Peak	1	0.049118 (p=0.159)	-0.133843 (p=0.997)
	2	-0.103898 (p=0.969)	0.067926 (p=0.126)
	3	0.201170 (p=0.000)	0.254355 (p=0.000)
	4	-0.008781 (p=0.566)	0.022335 (p=0.349)
X-galvo	1	-0.040226 (p=0.793)	0.041303 (p=0.201)
	2	0.262999 (p=0.000)	-0.180984 (p=0.999)
	3	-0.029639 (p=0.691)	0.064178 (p=0.106)
	4	0.022505 (p=0.334)	-0.034577 (p=0.726)
Y-galvo	1	0.379494 (p=0.000)	-0.030384 (p=0.732)
	2	0.549682 (p=0.000)	0.124974 (p=0.017)
	3	0.384213 (p=0.000)	0.089338 (p=0.041)
	4	0.390892 (p=0.000)	-0.069408 (p=0.886)

Table E.2 shows the peak, x-galvanometer and y-galvanometer skewness. The p-value shown in brackets is the probability that the skew measure is significantly different than the expect value of zero using a z-test. The skewness of most data sets is not significantly different than zero which indicates that the data are generally not skewed positively or negatively.

Most statistical tests tend to be somewhat robust with regards to deviations from normality. Typically it is sufficient to observe the frequency histogram for

a data set and if it appears to follow a normal or "bell-shaped" distribution then it should be sufficient for most statistical tests. Another way to examine the normality of a data set is to examine the normal probability plot. This test is performed by sorting the normalized data from smallest to largest, then calculating the cumulative frequency using

$$F_i = \frac{i - 0.5}{N} \quad (\text{E.8})$$

where N is the number of samples. The result is plotted on a normal probability plot. If the data is normally distributed then the result is an ascending straight line. A t-test or z-test can be performed to test the probability that the data does not follow a straight line [21, pp125-130].

Figure E.1 and Figure E.2 show the normal probability plots for peak data obtained from the Space 40 and Calib scanners. All data sets except Sample 2 of the Space 40 data appear to closely follow the normal line. That data set shows a slight curvature but it does not appear significantly curved so can still be considered normally distributed.

Figure E.3 and Figure E.4 show the normal probability plots for x-galvanometer data obtained from the Space 40 and Calib scanners. All data sets except Sample 1 and Sample 3 of the Calib data appear to closely follow the normal line. That data set shows a slight curvature but it does not appear significantly curved so can still be considered normally distributed.

Figure E.5 and Figure E.6 show the normal probability plots for y-galvanometer data obtained from the Space 40 and Calib scanners. All data sets except Sample 2, Sample 3 and Sample 4 of the Space 40 data appear to closely follow the normal line. The distributions do not appear significantly curved so can still be

considered normally distributed.

The data obtained from the Calib and Space 40 scanner do not follow a perfect Gaussian distribution but are close enough that a Gaussian can be used to represent additive noise for simulation purposes.

Figure E.1: Normal probability plot for Space 40 peak data

E.3 Observed Noise

The data used to calibrate the models for the Space40 and Calib scanners were used to obtain a model of the noise types and levels for each scanner. This data

Figure E.2: Normal probability plot for Calib peak data

Figure E.3: Normal probability plot for Space 40 X-galvanometer data

Figure E.4: Normal probability plot for Calib X-galvanometer data

Figure E.5: Normal probability plot for Space 40 Y-galvanometer data

Figure E.6: Normal probability plot for Calib Y-galvanometer data

comprised consisted of range values from 1-metre and 9-metres as well as a variety of galvanometer angles. Each sample point consisted to 10 measurements from which a sample standard deviation was obtained. Only peak values greater than 0 were included in the sample. If a peak value of 0 was observed then the sample was not included in the calculation and the count of zero spikes was incremented.

Table E.3: Observed peak and galvanometer noise levels

Source	Calib	Space40
Peak Standard Deviations (s_P)	1.946655	5.164816
X-galvo Standard Deviations (s_Θ)	2.480937	3.292068
Y-galvo Standard Deviations (s_Φ)	1.413150	2.832741

Peak data can contain outliers and the variance was observed to increase with the signal mean. Observation of the normalized data indicated that outliers could be identified as having a normalized variance of less than 3.5×10^{-3} . When a normalized variance greater than this level was observed then each of the samples was removed and the normalized variance re-calculated. The sample resulting in the greatest decrease in normalized variance was deemed the outlier. If the normalized variance was still greater than the maximum normalized variance then the process was repeated. Each time a value was removed the count of non-zero spikes was incremented. Table E.3 shows the maximum standard deviation observed for each scanner based on ten scans of 256 samples each.

The zero and non-zero spike rates were obtained by dividing the number of spikes of each type observed by the number of samples. This resulted in a measure of the fraction of samples observed with peak values of zero or outlier peak values.

Table E.4 shows the observed spike rates for the Calib and Space 40 scanners.

Figure E.7: Maximum observed noise levels for the peak detector (top) and galvanometers (middle), and spike rates (bottom) for the Calib scanner

Peak additive noise levels were the maximum observed normalized standard deviations. Galvanometer noise levels were the maximum observed sample standard deviation. Table E.3 shows the maximum observed or worst-case noise levels for each device. Figure E.7 and Figure E.8 show the worst-case levels as a function of data set examined. Samples 1 to 4 represent samples in which the galvanometer values were varied significantly. Samples 5 to 9 represented variance at a series of different range values between 1- meter and 9-metres. Typically the standard er-

Figure E.8: Maximum observed noise levels for the peak detector (top) and galvanometers (middle), and spike rates (bottom) for the Space40 scanner

ror is used as an unbiased estimator of the population standard deviation but the worst-case variances were obtained from data sets containing 10 or less samples [20, pp.303-306]. Moreover, the goal of this exercise was to obtain a worst-case estimate of the noise and the standard deviation of the observed data would be larger than the standard error and, thus, provide a margin of error in the estimate.

E.4 Peak Additive Noise

According to Beraldin *et al.* [5], laser speckle noise limits peak detection. Baribeau and Rioux described the noise in simplified form as

$$\sigma_p = \frac{f \lambda}{d \cos(\beta_{CCD} \sqrt{2} \pi)} \quad (\text{E.9})$$

where σ_p is the standard deviation of a zero-mean Gaussian noise process. This can be simulated by adding a Gaussian noise process to the detected peak value prior to quantization. For purposes of simulation, speckle noise is assumed to be included in the observed peak additive noise. This is modelled by adding a random value generated using a zero-mean Gaussian distribution with a spread estimated using the observed peak standard error. The random value is truncated to a range of ± 5 standard deviations about mean to avoid excessively large values [20, pp.317-318].

E.5 Galvanometer Error

Blais [7] and Beraldin [8] indicated that the worst-case position error of each galvanometer was $50 - \mu\text{rad}$ between 15-m and 47-m. Earlier work by Beraldin [5] cited an error of $60 - \mu\text{rad}$. However, this refers to the error associated with

the expected position of the laser point on a surface versus the actual position. Furthermore the type of error distribution was not specified.

For purposes of simulation, galvanometer error is assumed to be included in the observed galvanometer additive noise. This is modelled by adding a random value generated using a zero-mean Gaussian distribution with a spread estimated using the observed galvanometer standard error. The random value is truncated to a range of ± 5 standard deviations about mean to avoid excessively large values [20, pp317-318].

E.6 Peak Spike Rates

Impulse or “salt-and-pepper” noise consists of random values spanning a single pixel width that are significantly higher or lower than neighbouring values [2, pp53-55]. When applied to laser range scanners, impulses are referred to as outliers. In this thesis the term outlier and spike will be used interchangeable, the latter being preferred to distinguish between outliers in raw peak signals and statistical outliers observed in data being fit using a linear model.

Table E.4: Observed zero and non-zero spike levels

Spike type	Calib	Space40
Zero Spike Rate	0.009766	0.045312
Non-zero Spike Rate	0.000000	0.001172

Spike rates were calculated as the fraction of samples in which a spike occurred. The number of zero and non-zero spikes occurring in each Lissajous scan was determined and the maximum number was saved as the worst-case number of

spikes. This was divided by the number of samples in each scan to obtain the spike rate as a fraction of sampling rate. Table E.4 shows the spike rates for each of the scanners. MacKinnon *et al.* reported similar zero and non-zero spike rates for the Calib scanner [63]. Figure E.7 and Figure E.8 show the worst- case spike rates as a function of data set examined.

For purposes of simulation, spikes were added if a random value based on a uniform distribution with range 0 to 1 generated a value less than the observed spike rate. Replacing peak values with zeros simulated zero spikes. Non-zero spike values were generated by replacing the peak value with a random value taken from a uniform distribution with range -1 to $2^{15} - 1$.

Appendix F: Edge Height at 1-metre

The following figures represent an edge metric measured for each of two edge enhancement methods as edge height is changed. The edge enhancement methods are product-of-difference (PoD, circle) and derivative (Der, triangle). Vertical lines through each point represent the Scheffee confidence interval. Intervals that do not overlap indicate that the sample groups are significantly different at $p < 0.05$. In this test the edge height was varied between 0-metres and 0.25-metres above a background 1-metre from the scanner.

Figure F.1: Number Detected for Edge Enhancement Method by Edge Height (m) at 1-metre range

Figure F.2: Percent True for Edge Enhancement Method by Edge Height (m) at 1-metre range

Figure F.3: Percent False for Edge Enhancement Method by Edge Height (m) at 1-metre range

Figure F.4: Noise-to-Signal for Edge Enhancement Method by Edge Height (m)
at 1-metre range

Figure F.5: Mean Width for Edge Enhancement Method by Edge Height (m) at 1-metre range

Figure F.6: Absolute Deviation for Edge Enhancement Method by Edge Height (m) at 1-metre range

Figure F.7: Pratt Figure of Merit for Edge Enhancement Method by Edge Height (m) at 1-metre range

Figure F.8: van der Heyden Figure of Merit for Edge Enhancement Method by Edge Height (m) at 1-metre range

Appendix G: Edge Height at 2-metres

The following figures represent an edge metric measured for each of two edge enhancement methods as edge height is changed. The edge enhancement methods are product-of-difference (PoD, circle) and derivative (Der, triangle). Vertical lines through each point represent the Scheffee confidence interval. Intervals that do not overlap indicate that the sample groups are significantly different at $p \leq 0.05$. In this test the edge height was varied between 0-metres and 0.5-metres above a background 2-metres from the scanner.

Figure G.1: Number Detected for Edge Enhancement Method by Edge Height (m) at 2-metre range

Figure G.2: Percent True for Edge Enhancement Method by Edge Height (m) at 2-metre range

Figure G.3: Percent False for Edge Enhancement Method by Edge Height (m) at 2-metre range

Figure G.4: Noise-to-Signal for Edge Enhancement Method by Edge Height (m)
at 2-metre range

Figure G.5: Mean Width for Edge Enhancement Method by Edge Height (m) at 2-metre range

Figure G.6: Absolute Deviation for Edge Enhancement Method by Edge Height (m) at 2-metre range

Figure G.7: Pratt Figure of Merit for Edge Enhancement Method by Edge Height (m) at 2-metre range

Figure G.8: van der Heyden Figure of Merit for Edge Enhancement Method by Edge Height (m) at 2-metre range

Appendix H: Edge Height at 5-metres

The following figures represent an edge metric measured for each of two edge enhancement methods as edge height is changed. The edge enhancement methods are product-of-difference (PoD, circle) and derivative (Der, triangle). Vertical lines through each point represent the Scheffee confidence interval. Intervals that do not overlap indicate that the sample groups are significantly different at $p \leq 0.05$. In this test the edge height was varied between 0-metres and 2.5-metres above a background 5-metres from the scanner.

Figure H.1: Number Detected for Edge Enhancement Method by Edge Height (m) at 5-metre range

Figure H.2: Percent True for Edge Enhancement Method by Edge Height (m) at 5-metre range

Figure H.3: Percent False for Edge Enhancement Method by Edge Height (m) at 5-metre range

Figure H.4: Noise-to-Signal for Edge Enhancement Method by Edge Height (m)
at 5-metre range

Figure H.5: Mean Width for Edge Enhancement Method by Edge Height (m) at 5-metre range

Figure H.6: Absolute Deviation for Edge Enhancement Method by Edge Height (m) at 5-metre range

Figure H.7: Pratt Figure of Merit for Edge Enhancement Method by Edge Height (m) at 5-metre range

Figure H.8: van der Heyden Figure of Merit for Edge Enhancement Method by Edge Height (m) at 5-metre range

Appendix I: Edge Height at 10-metres

The following figures represent an edge metric measured for each of two edge enhancement methods as edge height is changed. The edge enhancement methods are product-of-difference (PoD, circle) and derivative (Der, triangle). Vertical lines through each point represent the Scheffee confidence interval. Intervals that do not overlap indicate that the sample groups are significantly different at $p \leq 0.05$. In this test the edge height was varied between 0-metres and 2.5-metres above a background 10-metres from the scanner.

Figure I.1: Number Detected for Edge Enhancement Method by Edge Height (m)
at 10-metre range

Figure I.2: Percent True for Edge Enhancement Method by Edge Height (m) at 10-metre range

Figure I.3: Percent False for Edge Enhancement Method by Edge Height (m) at 10-metre range

Figure I.4: Noise-to-Signal for Edge Enhancement Method by Edge Height (m)
at 10-metre range

Figure I.5: Mean Width for Edge Enhancement Method by Edge Height (m) at 10-metre range

Figure I.6: Absolute Deviation for Edge Enhancement Method by Edge Height (m) at 10-metre range

Figure I.7: Pratt Figure of Merit for Edge Enhancement Method by Edge Height (m) at 10-metre range

Figure I.8: van der Heyden Figure of Merit for Edge Enhancement Method by Edge Height (m) at 10-metre range

Appendix J: Edge Slope Effect

The following figures represent an edge metric measured for each of two edge enhancement methods as the distance between near and far ends of a ramp edge is changed. The edge enhancement methods are product-of-difference (PoD, circle) and derivative (Der, triangle). Vertical lines through each point represent the Scheffee confidence interval. Intervals that do not overlap indicate that the sample groups are significantly different at $p \leq 0.05$. In this test the width of the ramp edge was varied between 0-units and 1-units where a unit is a twice the fraction of the total galvanometer rotation. This means that 2 units refers to a 100% galvanometer rotation.

Figure J.1: Number Detected for Edge Enhancement Method by Edge Slope

Figure J.2: Percent True for Edge Enhancement Method by Edge Slope

Figure J.3: Percent False for Edge Enhancement Method by Edge Slope

Figure J.4: Noise-to-Signal for Edge Enhancement Method by Edge Slope

Figure J.5: Mean Width for Edge Enhancement Method by Edge Slope

Figure J.6: Absolute Deviation for Edge Enhancement Method by Edge Slope

Figure J.7: Pratt Figure of Merit for Edge Enhancement Method by Edge Slope

Figure J.8: van der Heyden Figure of Merit for Edge Enhancement Method by Edge Slope

Appendix K: Edge Separation Effect

The following figures represent an edge metric measured for each of two edge enhancement methods as the distance between edges on a ridge is changed. The edge enhancement methods are product-of-difference (PoD, circle) and derivative (Der, triangle). Vertical lines through each point represent the Scheffee confidence interval. Intervals that do not overlap indicate that the sample groups are significantly different at $p \leq 0.05$. In this test the edge separation was varied between 0-units and 1-units where a unit is a twice the fraction of the total galvanometer rotation. This means that 2 units refers to a 100% galvanometer rotation.

Figure K.1: Number Detected for Edge Enhancement Method by Edge Separation

Figure K.2: Percent True for Edge Enhancement Method by Edge Separation

Figure K.3: Percent False for Edge Enhancement Method by Edge Separation

Figure K.4: Noise-to-Signal for Edge Enhancement Method by Edge Separation

Figure K.5: Mean Width for Edge Enhancement Method by Edge Separation

Figure K.6: Absolute Deviation for Edge Enhancement Method by Edge Separation

Figure K.7: Pratt Figure of Merit for Edge Enhancement Method by Edge Separation

Figure K.8: van der Heyden Figure of Merit for Edge Enhancement Method by Edge Separation

Appendix L: RAS Software

L.1 Overview

The RAS simulator consists of a simplified model of the Random Access Scanner (RAS) developed by the National Research Council of Canada (NRC) and a simulation environment which allows the scanner to interact with virtual objects. The RAS model generates a vector which defines the path of a laser out of the scanner. The vector is then converted from the RAS frame of reference into the environmental frame of reference so the closest point of intersection with a surface in the environment can be determined. The position of the point of intersection is then converted back into the RAS frame of reference so that the RAS model can determine the peak value the RAS would generate in response to this information. The environmental model stores the location of the RAS in the simulation environment as well as the locations of all planar surfaces in the environment.

The RAS simulator is initialized prior to operation using the **sysInit.m** file which adds the appropriate path information to Matlab search path. All scripts used to perform simulations are run from the RAS home directory. The following sections summaries the contents and importance of each of the subdirectories contained within the RAS home directory.

L.2 RAS

RASInit.m - Initializes the RAS simulator based on an initialization file supplied as part of the function call. For example,

```
myRAS = RASInit( 'initGeneric' );
```

creates an RAS model object **myRAS** using the initialization file **initGeneric.m**. Initialization files are stored in the **Calibration** subdirectory. The resulting galvanometer position is stored in **myRAS.galvoA** variable.

RASoutput.m - Accepts a RAS model object and a pair of galvanometer “voltages” and returns an updated RAS model object containing the new laser output vector. The vector is defined by the origin of the ray stored in **myRAS.f3** and the x-axis and y-axis mirror deflections respectively stored in **myRAS.deflection**. This script also generates a vector defining the optical axis as an origin **myRAS.a3** and the x-axis and y-axis mirror deflections respectively stored in **myRAS.OP**. For example,

```
myRAS = RASoutput( myRAS, thetaV, phiV );
```

updates the RAS object **myRAS** based on the new x-galvanometer position **thetaV** and the new y-galvanometer position **phiV**. The variables **thetaV** and **phiV** must be in the range 1 to -1.

RASinteract.m - Accepts a RAS model object, an environmental model object and a time step in seconds to determine the nearest point of intersection of the laser with a surface in the environment. The closest point of intersection of the laser with the surface is stored in the **myRAS.Pdetected** variable. For example,

```
[myRAS, myEnviro] = RASinteract( myRAS, myEnviro, 0.01 );
```

updates the RAS object `myRAS` and the environmental model `myEnviro` with a time step of 0.01-seconds.

The RAS model object stores the laser information and the environmental model object stores the position of the scanner and all objects in the environment. This script updates the position of the scanner in the environment prior to determining the laser intersection. Environmental models are stored in the **Environment** directory.

RASrespond.m - Accepts a RAS model object and generates the resulting peak value which is stored in `myRAS.P`. The final UVP coordinate contains the peak value and the galvanometer readings as a single variable `myRAS.UVP`. For example,

```
myRAS = RASrespond( myRAS );
```

updates the RAS object `myRAS`.

L.3 Calibration

initGeneric.m, **initCalib.m**, **initSpace40.m** - Initialization files for specific scanners. The command

```
myRAS = RASInit('initGeneric');
```

initializes the RAS model based on the intrinsic and calibration parameters and for the scanner modelled in **initGeneric.m**. Scanner calibration must take place in the following order:

1. Run the script **galvoCalibrate.m** and copy the new settings into the appropriate initialization file. The **initGeneric.m** file can be used as a template. This step assumes samples have been obtained from the maximum galvanometer extents of the real scanner being modelled.
2. Run the script **peakCalibrate.m** and copy the new settings into the appropriate initialization file created in Step 1. This step assumes samples have been obtained between 1-metre and 10-metres from the real scanner being modelled.
3. Run the script **linearPeakFit.m** and copy the new settings into the appropriate initialization file created in Step 1 and updated in Stage 2.
4. Run the script **confirmFit.m** to ensure that the model generates results that are linearly related to the results obtained from the real scanner.

galvoCalibrate.m - Used to perform calibration of the x-axis and y-axis mirror positions and determines the galvanometer transformation parameters.

peakCalibrate.m - Used to perform calibration of the peak detector and determines the peak detector transformation parameters.

linearPeakFit.m - This script generates the linear model coefficients to fit the model data to measured data from the real scanner.

confirmFit.m - Used to perform linear fitting of the peak values generated by the model to peak values obtained from a real scanner.

normTest.m - Determines the kurtosis and skewness of each data set, then tests the significance of those parameters versus the hypothesis that the data comes

from a Gaussian distribution. This script also generates a normal probability plot.

normPeak.m - Called by **normPeak.m** to process each data set.

noiseTest.m - Determines the standard deviation, and zero and non-zero spike rates, of data collected from a real scanner.

noiseModel.m - Called by **noiseTest.m** to process each data set.

L.4 Environment

envInit.m - Creates an environmental model object according to a specified environmental model initialization file. For example

```
myEnviro = envInit('roomModel');
```

creates an environmental model object **myEnviro** using the initialization file **roomModel.m**. Initialization files are stored in this subdirectory.

spaceModel.m, **roomModel.m**, **space40Model.m** - Environmental initialization files. These scripts combine scanner motion information from scripts in the **Motion** subdirectory, the outer extents of the “world” encompassing the environment from the **Worlds** subdirectory, and objects stored in the **Objects** subdirectory.

envDraw.m - Draw the environment. For example

```
envDraw( myEnviro );
```

draws the environment based on the settings currently stored in the environmental model object `myEnviro`.

L.5 Motion

`genericStatic.m` - Generic scanner motion model in which the scanner is stationary.

`genericMotion1.m`, `genericMotion2.m` - Generic scanner motion models in which the scanner is moving during the scanning process.

`space40Static.m`, `space40Motion.m` - Scripts based on a real test structure using the Space 40 scanner.

L.6 Worlds

`spaceWorld.m`, `roomWorld.m`, `space40World.m` - Examples of world models which define the outer extents of the scanner environment.

L.7 Objects

`genericBox.m`, `Space40Box.m` - Examples of objects that can exist in a scanner environment.

L.8 Lissajous

lissajousInit.m - Creates a Lissajous scan object with a set of scan parameters entered as part of the function call. For example

```
myLissScan = lissajousInit( Atheta, Aphi, m, n, resolution, ...  
                           uInit, vInit );
```

creates a Lissajous scan object **myLissScan** with x-galvanometer scan width **Atheta**, y-galvanometer scan width **Aphi** and using a (m:n) Lissajous pattern. The scan will obtain **resolution** samples during a single scan and will be centred at **uInit** along the x-axis and **vInit** v-axis. **Atheta**, **Aphi**, **uInit** and **vInit** must be in the range 1 to -1.

lissajousScan.m - Performs a Lissajous scan using the scan parameters stored in the Lissajous scan object specified as part of the function call. For example

```
[myLissScan, myRAS, myEnviro] = lissajousScan( myLissScan, myRAS, ...  
                                              myEnviro );
```

performs a Lissajous scan based on the settings stored in **myLissScan** using the RAS model **myRAS** in the environment described by **myEnviro**. The peak values obtained during the Lissajous scan are stored in the variables **myLissScan.P** and the corresponding galvanometer readings are stored in **myLissScan.A**. Note that the first 7 elements of the the scan will be set to zero.

plotObject.m, **plotSurface.m**, **plotRange.m**, **plotScan.m** - Scripts to plot data obtained using a Lissajous scan. For example


```
plotObject( myLissScan.P(8:resolution), myLissScan.A(8:resolution,:) );
```

plots the data stored in myLissScan.

L.9 Raster

rasterInit.m - Creates a raster scan object with a set of scan parameters entered as part of the function call. For example

```
myRasterScan = rasterInit( Atheta, Aphi, resolution, xInit, yInit );
```

creates a raster scan object myRasterScan with x-galvanometer scan width Atheta and y-galvanometer scan width Aphi. The scan will obtain resolution samples during a single scan and will be centred at uInit along the x-axis and vInitv-axis. Atheta, Aphi, uInit and vInit must be in the range 1 to -1.

rasterScan.m - Performs a raster scan using the scan parameters stored in the raster scan object specified as part of the function call. For example

```
[myRasterScan, myRAS, myEnviro] = rasterScan( myRasterScan, myRAS, ...  
                                              myEnviro );
```

performs a raster scan based on the settings stored in myRasterScan using the RAS model myRAS in the environment described by myEnviro. The peak values obtained during the Raster scan are stored in the variables myRasterScan.P and the corresponding galvanometer readings are stored in myRasterScan.A.

plotRaster.m, imgRaster.m - Scripts to plot data obtained using a Raster scan. The **imgRaster.m** script requires a raster scan of at least 64-bit resolution to display a useful image. For example

```
plotRaster( myRasterScan.P, myRasterScan.A );
```

plots the data stored in myRasterScan.

L.10 Edge

lissEdgeScan.m - Performs a Lissajous scan using the simple surface entered as part of the function call rather than an environmental model. For example

```
[myLissScan, myRAS] = lissEdgeScan( myLissScan, myRAS, 'rampEdge', ...  
                                   dMin, dMax, width );
```

performs a Lissajous scan based on the settings stored in myLissScan using the RAS model myRAS and the surface described by the script **rampEdge.m**. dMin, dMax and width are variables handed to the script generating the simple surface.

boxEdge.m, **rampEdge.m**, **ridgeEdge.m**, **roofEdge.m** - Simple surfaces used to test edge detection routines.

averagingFilter.m - Averaging noise filter. For example

```
[myLissScan.P] = averagingFilter( myLissScan.P, 5 );
```

filters the data stored in myLissScan.P using a $5 + 5 + 1 = 11$ element averaging filter.

gaussianFilter.m - Gaussian noise filter. For example

```
[myLissScan.P] = gaussianFilter( myLissScan.P, 5 );
```

filters the data stored in `myLissScan.P` using a $5 + 5 + 1 = 11$ element Gaussian filter.

medianFilter.m - Median noise filter. For example

```
[myLissScan.P] = medianFilter( myLissScan.P, 5 );
```

filters the data stored in `myLissScan.P` using a $5 + 5 + 1 = 11$ element median filter.

iteratedFilter.m - Iterated median noise filter. For example

```
[myLissScan.P] = averagingFilter( myLissScan.P );
```

filters the data stored in `myLissScan.P` using an iterated median filter.

pod.m - Product of Difference edge enhancement filter. For example

```
[myEdgeScan.P] = pod( myLissScan.P, 5 );
```

generates a new Lissajous scan object `myEdgeScan` by filtering the data stored in the variable `myLissScan.P` using a $5 + 5 + 1 = 11$ element Product of Difference edge enhancement filter.

derive3.m, derive5.m - Three and 5-point derivative edge enhancement filter. For example

```
[myEdgeScan.P] = derive3( myLissScan.P );
```

generates a new Lissajous scan object `myEdgeScan` by filtering the data stored in the variable `myLissScan.P` using a 3-point derivative edge enhancement filter.

getPeaks.m - Peak extraction routine. For example

```
[myEdgeScan.P] = getPeaks( myEdgeScan.P, myLissScan.P, minThreshold, ...  
                           minRatio );
```

updates the Lissajous scan object **myEdgeScan** such that each non-zero element of the variable **myEdgeScan.P** indicates a single peak value that is greater than the threshold **minThreshold** and subject to the constraint that the peak base to peak height ratio of each peak is greater than **minRatio**. All peaks have been trimmed to ensure they are no more than one pixel wide.

edgeExtract.m - Extract the smallest and largest ranges around each detected edge. For example

```
edgeModel = edgeExtract( myEdgeScan, myLissScan, window );
```

returns an edge model object **edgeModel** generated by pairing smallest and largest range values around the edges stored in **myEdgeScan** using the peak values stored in **myLissScan** within the window defined by **window**. Both **myEdgeScan** and **myLissScan** are Lissajous scan objects but **myEdgeScan** contains non-zero peak values only where an edge has been detected.

edgeMetrics.m - Generates a series of test statistics based on the ideal and measured edge profiles.

L.11 C

The **C** directory stores C and C++ code that can be used to pre-test software intended to be used on a real scanner. The scanners at the NRC are controlled using QNX workstations so modules can be developed in C or C++ and tested using the RAS simulator prior to being integrated into one or more of the existing systems.

NRCaveraging.c - Averaging noise filter. For example

```
[myLissScan.P] = double( NRCaveraging( int16(myLissScan.P), 5 )' );
```

filters the data stored in `myLissScan.P` using a $5 + 5 + 1 = 11$ element averaging filter.

NRCgaussian.c - Gaussian noise filter. For example

```
[myLissScan.P] = double( NRCgaussian( int16(myLissScan.P), 5 )' );
```

filters the data stored in `myLissScan.P` using a $5 + 5 + 1 = 11$ element Gaussian filter.

NRCmedian.c - Median noise filter. For example

```
[myLissScan.P] = double( NRCmedian( int16(myLissScan.P), 5 )' );
```

filters the data stored in `myLissScan.P` using a $5 + 5 + 1 = 11$ element median filter.

NRCmedianQ.c - Median noise filter optimized using swapping. For example

```
[myLissScan.P] = double( NRCmedianQ( int16(myLissScan.P), 5 )' );
```

filters the data stored in `myLissScan.P` using a $5 + 5 + 1 = 11$ element median filter optimized using swapping.

NRCiterated.c - Iterated median noise filter. For example

```
[myLissScan.P] = double( NRCiterated( int16(myLissScan.P) )' );
```

filters the data stored in `myLissScan.P` using an iterated median filter.

NRCpod.c - Product of Difference edge enhancement filter. For example

```
[myEdgeScan.P] = double( NRCpod(int16(myLissScan.P'), 5)' );
```

generates a new Lissajous scan object `myEdgeScan` by filtering the data stored in the variable `myLissScan.P` using a $5 + 5 + 1 = 11$ element Product of Difference edge enhancement filter.

NRCder.c - Three-point derivative edge enhancement filter. For example

```
[myEdgeScan.P] = double( NRCder(int16(myLissScan.P'))' );
```

generates a new Lissajous scan object `myEdgeScan` by filtering the data stored in the variable `myLissScan.P` using a 3-point derivative edge enhancement filter.

NRCgetPeaks.c - Peak extraction routine. For example

```
[myEdgeScan.P] = double( NRCgetPeaks(int16(myEdgeScan.P'), ...  
                                int16(myLissScan.P'), minThreshold, minRatio)' );
```

updates the Lissajous scan object `myEdgeScan` such that each non-zero element of the variable `myEdgeScan.P` indicates a single peak value that is greater than the threshold `minThreshold` and subject to the constraint that the peak base to peak height ratio of each peak is greater than `minRatio`. All peaks have been trimmed to ensure they are no more than one pixel wide.

L.12 Tools

`addNoise.m`, `addSpike.m` - Add noise to a vector.

`lineDef.m` - Define a line.

`RotY.m`, `RotX.m`, `RotZ.m`, `translate.m`, `lineMove.m`, `invLineMove.m` -
Used to translate a line between coordinate frames.

**`drawpoint.m`, `displayPoint.m`, `drawlens.m`, `drawline.m`, `drawplane.m`,
`drawsquare.m`** - Assorted drawing routines.

`quantize.m` - Quantize a value to a specified number of bits.

`randTrunc.m` - Generate a random value from a Gaussian distribution truncated to a given number of standard deviations.

`boxExtents.m` - Calculate the planes enclosing a box of a given set of dimensions.

`planeInit.m`, `planeIntersection.m`, `withinPlane.m` - Define a plane and determine if a line intersects it.

`regression.m`, `compareToMean.m`, `zCDF.m`, `zP.m` - Assorted statistical routines.

enclose.m, **sphereIntersect.m** - Used to define a spherical region and determine if a line intersects it.

L.13 Demos

The **Demos** directory contains demonstration programs that can form the basis of test programs. Programs that will not use the **RASinteract** script can use the **flatDemo.m** script as a basis. Test that will use Lissajous or Raster scan patterns can use the **lissajousDemo.m** or **rasterDemo.m** scripts respectively.

lissajousDemo.m - Demonstration of a Lissajous scan.

rasterDemo.m - Demonstration of a raster scan.

comboDemo.m - Demonstration of combining a Lissajous scan with a raster scan.

edgeDemo.m - Demonstration of using a Lissajous scan to perform edge detection.

flatDemo.m - Demonstration of scanning a flat surface rather than using an environmental model.