A LADAR-BASED POSE ESTIMATION ALGORITHM FOR DETERMINING RELATIVE MOTION OF A SPACECRAFT FOR AUTONOMOUS RENDEZVOUS AND DOCK

by

Ronald C. Fenton

A thesis submitted in partial fulfillment
of the requirements for the degree

of

MASTER OF SCIENCE

in

Electrical Engineering

Approved:

Dr. Rees Fullmer Major Professor	Dr. Scott Budge Committee Member
Dr. Charles Swenson Committee Member	Dr. Byron R. Burnham Dean of Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2008

Abstract

A Ladar-Based Pose Estimation Algorithm for Determining Relative Motion of a Spacecraft for Autonomous Rendezvous and Dock

by

Ronald C. Fenton, Master of Science

Utah State University, 2008

Major Professor: Dr. Rees Fullmer
Department: Electrical and Computer Engineering

Future autonomous space missions will require autonomous rendezvous and docking operations. The servicing spacecraft must be able to determine the relative 6 degree-of-freedom (6 DOF) motion between the vehicle and the target spacecraft. One method to determine the relative 6 DOF position and attitude is with 3D ladar imaging. Ladar sensor systems can capture close-proximity range images of the target spacecraft, producing 3D point cloud data sets. These sequentially collected point-cloud data sets were then registered with one another using a point correspondence-less variant of the Iterative Closest Points (ICP) algorithm to determine the relative 6 DOF displacements. Simulation experiments were performed and indicated that the mean-squared error (MSE), angular error, mean, and standard deviations for position and orientation estimates did not vary as a function of position and attitude and meet most minimum angular and translational error requirements for rendezvous and dock. Furthermore, the computational times required by this algorithm were comparable to previously reported variants of the point-to-point and point-to-plane-based ICP variants for single iterations when the initialization was already performed.

(89 pages)

Dedicated to my beautiful wife, Lynda, my mother, and father for always believing in me!

Acknowledgments

I would like to thank the Utah State University-Skunkworks program, the Center for Advanced Imaging Ladar (CAIL), and the SDL IRAD program for funding this exciting research topic. Without the funding for this project, none of this work would have been possible. Next, I would like to acknowledge and thank many times over Dr. Rees Fullmer for his excitement, expertise, and experience with the 3D ladar pose estimation project over the past several years. With his guidance and experience, I was finally able to finish this project that took much longer than I had expected. He was there for me as a colleague and a friend during a very difficult time in my life, and I cannot thank him enough. Additionally, I would like to thank him for all his patience in taking on a free-spirited soul such as myself, and trying to mold me into a successful research assistant. I would like to thank my loving wife, Lynda, who has stuck by me through thick and thin as I have started a new chapter in my life. Finally, I would like to acknowledge my family. To my mother, you have been my emotional rock through all my life, and you have been there to give me support when no one else could. To my father who passed away this year, I hope that I will be half the man you were as a golfer, a friend to others, my hero, and foremost as a father. You have both shaped and molded me into the successful young man that I have become these past 27 years. Thank you all!

Ronald C. Fenton

Contents

List of Tables

List of Figures

Notation

Events

$\vec{q}$	quaternion
$R(\vec{q}), R(\psi, \theta, \phi)$	6-DOF rotation matrix
ψ, θ, ϕ	euler angles
$\vec{p}(x, y, z)$	6-DOF cartesian translation vector
$O()$	computational order
R_1, R_2	3D point cloud data sets
$d(x_i, y_i)$	point-to-point distance
$d({}^I R_{point}, A_I)$	point-to-plane distance
x_P, y_P, z_P	3D cartesian coordinate system - plane/mesh fixed
x_I, y_I, z_I	3D cartesian coordinate system - ladar fixed
$T_{I \rightarrow P}$	ladar to plane/mesh rotational coordinate transformation
${}^I \rho_{trans}$	ladar to plane/mesh cartesian translation vector
e_x, e_y, e_z	unit eigen vectors
${}^P X_{point}, {}^I X_{point}$	single point-cloud point projected onto plane/mesh
$\vec{\mu}_{R_1}, \vec{\mu}_{R_2}$	R_1, R_2 point-cloud centroids
$\Sigma_{R_1 R_2}$	3x3 cross covariance matrix
$\vec{n}$	plane/mesh surface normal
$R_{R_1 R_2}(\psi, \theta, \phi)$	rotation Matrix from R_1 to R_2
$\vec{q}_r$	quaternion from R_1 to R_2

Acronyms

S/C	spacecraft
DOF	degree-of-freedom
3D	three dimensional
MSE	mean squared error
ICP	Iterative Closest Point
CAIL	Center for Advanced Imaging Ladar
LADAR	Laser Detection and Ranging
LIDAR	Light Detection and Ranging
RVD	rendezvous and docking
GNC	guidance, navigation, and control
EO	electro optics
FOV	field-of-view
SNR	sinal-to-noise ratio
CCD	charge coupled device
CMOS	complementary metaloxidesemiconductor
ARD	Automated Rendezvous and Dock Sensor
ISS	International Space Station
RLS	Rendezvous Ladar System
FPA	Focal Plane Array
LARS	Laser Range Scanner
CSA	Canadian Space Agency
LDRI	Laser Dynamic Range Imager
NEAR	Near-Earth Asteroid Rendezvous-Shoemaker
NLR	NEAR Laser Range Finder

LAMP	LAser MaPper
AVGS	Advanced Video Guidance System
RVR	rendezvous laser radar
NASADA	National Space Development Agency of Japan
NASA	National Aeronautics and Space Administration
MSFC	Marshall Space Flight Center
FPGA	field-programmable gate array
DSP	Digital Signal Processing
ROI	regions of interest
ETS	Engineering Test Satellite
DART	Demonstration of Autonomous Rendezvous Technology
MUBLCOM	Multiple-Path Beyond-Line-of-Site Communications
RSCS	Premier Rendezvous and Sample Capture System
ARCSS	Autonomous Rendezvous and Capture Sensor System
HST	Hubble Space Telescope
HRV	Hubble Robotic Service Vehicle
SVD	singular-value decompositon
EVD	eigen-value decompositon
RMS	root mean squared

Chapter 1

Introduction

1.1 Motivation

In the future, organizations such as NASA, Lockheed-Martin, and Boeing will have the capabilities for autonomous space missions. For missions without man-in-loop control, these spacecraft will require autonomous rendezvous and docking (RVD) operations with either cooperative or non-cooperative spacecraft. In order to maneuver and dock, a servicing spacecraft must be able to determine the relative 6 degree-of-freedom (6 DOF) motion/orientation between the vehicle and the target spacecraft.

The critical tool required for these RVD maneuvers was a relative navigation sensor system, which determined the relative position and orientation of the controlled spacecraft with respect to the target spacecraft. Current relative position sensing methods used 2D visual cameras and extensive image processing to determine the position and range of a cooperative target. The limitations of 2D systems are that the size, shape, and range to objects are all derived from interpretations of the surface reflectivity leaving them sensitive to lighting conditions. Over the last few decades, 3D sensing technologies have matured greatly and have proven to be more reliable. Thus, we have chosen to use a ladar system for pose orientation. This required using a pose estimation algorithm, which used 3D point clouds (3D cartesian XYZ range information) as an input, and using only that point cloud (i.e. no feature extraction, etc).

Conceptually, a ladar-based autonomous RVD system would be able to produce accurate 3D point cloud models in real time. Then two separate point clouds would be matched together with each other, or independently matched to a solid model of the spacecraft. From this match, the system could determine the necessary relative position and orientation between the vehicle and target spacecraft in the form of a quaternion.

Therefore, the goal of this thesis was to find a precise mathematical algorithm, suitable for 6 DOF range and orientation determination, such as the ICP algorithm. In this project, our contribution was to analytically determine whether the ICP algorithm was accurate enough for relative motion determination for autonomous rendezvous and dock using only ladar-based point-clouds. This algorithm required the use of sequentially acquired and accurate 3D LADAR (Laser Detection and Ranging), LIDAR (Light Detection and Ranging), or Laser Radar point cloud models, in order to estimate 6 DOF orientation.

1.2 Approach of Research

The approach of this thesis project started by building a 3D model of a satellite and using it as an input into the software simulator LadarSIM. Using the 3D model, LadarSIM was able to build and generate point cloud models which provided complete 3D information about the solid model. These point clouds were taken sequentially as time moved forward from one instant to the next. The point clouds were then seeded into two variants of the Iterative Closest Point (ICP) algorithm. After seeding two sequentially acquired point clouds, the point-to-point and point-to-plane ICP variants were used to calculate the 6 DOF rotation (quaternion or rotation matrix), as well as the cartesian translation. To study the ICP algorithm, a noise sensitivity analysis was performed by incorporating RMS range error noise (0 cm, 2 cm, and 14 cm) into the point cloud models to estimate the algorithms computational accuracy. Also, tests were performed to discover what improvements in initialization must be made for real time requirements, in order to incorporate the pose estimation software into a closed loop system for autonomous RVD. All results were graphically analyzed, and tabularized for a quick quantitative analysis. Finally, methods for future initialization implementation, alternative methods that could be incorporated into the ICP framework to increase pose estimation accuracy, and future plans for experimental hardware testing.

1.3 Thesis Organization

This thesis was organized into five separate chapters. Following the introductory chapter, the second chapter will review the literature describing autonomous RVD, several space based ladar systems and hardware, and 6 DOF pose estimation algorithms using 3D ladar point-cloud data. In the third chapter, a full discussion of the Iterative Closest Point (ICP) algorithm will be given for the point-to-point and point-to-plane variations. The fourth chapter will discuss how simulated data was obtained in LadarSIM, and show sensitivity analysis results from the point-to-point and point-to-plane variations. Finally, the fifth chapter will discuss sensitivity analysis conclusions and provide a road map for future research endeavors.

Chapter 2

A Review of the RVD Problem, Ladar Imaging, and Pose Estimation

2.1 Autonomous Rendezvous and Dock

On March 16, 1966, the first rendezvous and dock (RVD) between two spacecraft occurred when Neil Armstrong and Dave Scott manually performed a rendezvous in a Gemini spacecraft and then docked with an unmanned Agena target vehicle. Over a year later on October 30, 1967, the first man/computer controlled autonomous RVD took place when the Soviet vehicles Cosmos 186 and 188 docked with one another. Since then, the Russians have been performing autonomous RVD between support craft such as the Soyuz and the Mir space station since the 1990s until its closure in 2001 [1–3]. In general, the rendezvous and docking process consists of a series of orbital maneuvers and trajectories that successfully bring an active satellite/spacecraft (S/C) (chaser) into the vicinity, and eventually into contact with a passive satellite/spacecraft (target) through a set of planned maneuvers (See fig. 2.1) [4]:

- Launch: injection into orbital plane of target and achievement of stable orbital conditions;
- Phasing: reduction of orbital phase angle between chaser and target spacecraft;
- Far Range Rendezvous: transfer from phasing orbit to first aim point in close vicinity of target spacecraft;
- Close Range Rendezvous: reduction of relative distance to target acquisition of final approach, approach to capture point, and achievement of capture conditions; and

- Mating: (docking or berthing) prevention of escape of capture interfaces and attenuation of shock and residual motion, as well as, insertion into structural latch interfaces and achievement of rigid structural connection.

For the case of docking, the chaser satellite has an active docking mechanism controlled by the guidance, navigation, and control (GNC) system allowing the chaser spacecraft to rigidly attach itself to the passive docking mechanism on the target spacecraft. Similarly, berthing allows the chaser vehicle to arrive within a certain proximity of the target satellite with zero relative velocities and angular rates. At this point, a robotic manipulator arm extends from the chaser vehicle and orients itself until it interfaces with a berthing port on the target satellite/spacecraft (See fig. 2.2) [1].

The next several sections in the thesis provide an overview the literature. Section 2.2 discusses 3D ladar scanning sensor technologies; sec. 2.3 discusses space-based sensors range sensor scanners; sec. 2.4 discusses many of the space-based autonomous rendezvous and dock missions and systems; sec. 2.5 discusses the historical background of 3D pose estimation algorithms and the path that was followed to develop the ICP. The final section provides a quick summary of the literature review.

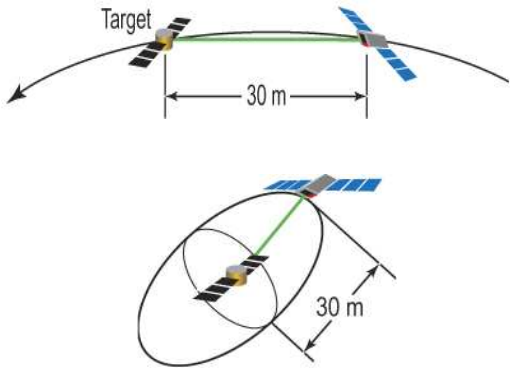

Fig. 2.1: Rendezvous and docking process (short range rendezvous and docking).

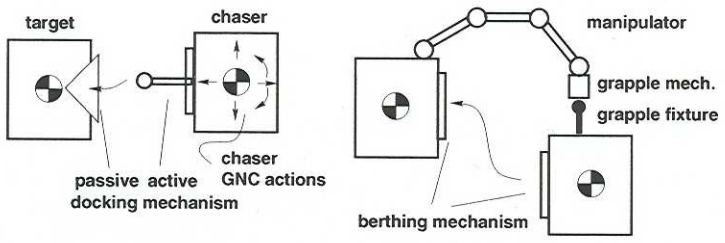

Fig. 2.2: Docking and berthing.

2.2 3D Ladar Scan Sensor Technologies

Ladar systems combine the capabilities of radar and optical systems allowing simultaneous measurement of range, reflectivity, velocity, temperature, azimuth, and elevation angles. Unlike traditional camera-based electro-optic (EO) systems, ladar systems are independent of ambient lighting conditions in providing accurate range measurements, although with coarser resolutions than EO systems. Ladar instruments will usually use one of two techniques when calculating range measurements to an object: (1) continuous wave ladar, which sends out modulated signals and measures the phase difference between transmitted and received signals; or (2) pulsed ladar, which sends out individual laser pulses and measures time-of-flight between transmitted and received signals [3].

Many ladar systems typically consist of a pulsed laser, scan and steering mechanism, and a detector coupled to a high-speed clock to measure the laser pulse time-of-flight. Ladars operate at wavelengths in the ultraviolet, visible, near, mid and far-infrared regions (i.e., wavelengths between $0.5 * 10^{-6}$ to $1 * 10^{-5}$ m or frequencies between 600K THz to 30K THz). Range accuracies are a function of the pulse energy, width, receiver sensitivity, and clock resolution, providing range accuracies from 1 to 3 cm over kilometer distances with 0.05° angular resolutions. Image scan time, field-of-regard, and angular accuracy are dependent upon laser pulse repetition frequency, scanner speed, range of motion, and resolution. Primarily, the raw data obtained from any ladar instrument would be simultaneous range, azimuth, and elevation angles.

3D point-cloud images are generated by either scanning the laser beam over two dimensions or flashing numerous laser pulses over two dimensions at one time. The system converts the data into a traditional 3D cartesian format. These 3D images contain significantly more navigational information than a traditional 2D imager such as: area-inertias, volumes, depth, etc. [5].

Over the last several decades, these ladar technologies have been developed so that data could be acquired through a variety of methods. Blais [6] provided a brief review of the past 20 years of development in the field of 3D laser imaging. This review discusses range sensor development of single point laser scanners, slit scanners, pattern projection, time-of-flight, and flash systems.

2.2.1 Single Point, Slit Scanner, and Pattern Projection

Single point 3D laser scanners use fast scanning mirrors which synchronize the optical detector with the laser projection enabling the use of long focal length lens and zoom configurations. However, complex scanning systems and components increase the cost of laser scanners, while decreasing system speeds [6].

Slit scanners tend to be the most widely used triangulation-based 3D laser cameras because of their cost. As newer CCD and complementary metaloxidesemiconductor (CMOS) arrays are developed, the lateral resolution increases because of a larger number of pixels, but range performance remains constant even with a decrease in the number CMOS detectors and flawless internal optics.

Pattern projection techniques use multiple stripes or patterns projected on the object rather than using a single laser line, mechanically scanning the scene, and processing independent range profiles. Drawbacks to this technology are de-focusing of the projected pattern, smaller depth of focus, and reduction in dynamic range in intensity [6].

2.2.2 Time-of-Flight

For larger structures and longer ranges, time-of-flight (TOF) 3D scanners are the preferred choice for long range measurements. Their range accuracy remains relatively constant

over the whole scan volume. Because these systems require detection of the time that light propagates though air, measurements will be affected mostly by drift and jitter in electronics and discrimination methods. A pulse time-of-flight scanner detects the time a laser pulse is reflected back to a receiving detector. Pico-second resolution requires extremely sensitive electronics with high band-width, constant group-delays, and thermal stability. To reduce noise, multiple pulses are averaged and resolutions on the order of 0.5 to 1 cm can be measured. High frequency bandwidth in the electronics is needed to amplify the large frequency spectrum associated with the laser pulses. Electronics are more complex for the pulsed technology, but reduced bandwidth provides better range resolution on the order of 3 to 5 mm [6]. Time-of-flight ladar nominally have minimum ranges typically from 10 to 100 m due to the short flight period and high return intensity, with maximum ranges as far as 10 km. Ladar systems are virtually immune to the varied lighting and viewing conditions encountered in space, including both specular and diffuse reflecting surfaces.

2.2.3 Flash

Flash ladars use the time-of-flight approach for range determination. The laser transmitter will flood illuminate a target area with a relatively short pulse ($\sim$10 ns). The time-of-flight pulses are measured in a per pixel fashion and are commonly represented by stacks of pixels that represent wave front time series for each pixel in the focal plane array (FPA). A new frame is triggered by each laser pulse, and each corresponding time series is initiated by the reflected wavefront. The position of the detecting pixel yields the angular position of the target element, and the time-of-flight yields the third dimension range. Flash ladars, like point TOF ladar, also have the ability to track targets as far out as 10 kilometers with range accuracies of 10-15 cm. The advantage to the flash technology is that with a single shot, the complete 3D point-cloud of a target is able to be captured. Currently, Ball Aerospace has designed a fourth generation flash ladar which they couple with FPGA processors to quickly process 3D imaging ladar point-cloud data [7, 8].

2.3 Space-Based Range Sensor Scanners

Vision systems that have been developed fall into one of two catergories, either passive (video, stereo) or active (3D laser projection). Passive systems offer simplicity in hardware, typically requiring only conventional video cameras. However, passive 2D video images do not deliver direct 3D measurements that are required for most autonomous RVD situations. Active systems, such as ladar scanners, provide direct range measurement to an object. Also, the use of laser light offers strong advantages for discriminating against unwanted backgrounds. So, over the years, many different institutions have used both 3D ladar technologies and 2D video cameras for the purpose of obstacle avoidance, inspection, and rendezvous and dock [9–18].

2.3.1 Laser Range Scanner (LARS)

The Laser Range Scanner (LARS), developed by the Canadian Space Agency (CSA) [9, 10], was a versatile 3D sensor for space applications capable of doing surface imaging, target ranging, and tracking. It was capable of short range (0.5 m to 20 m) and long range (20 m to 10 km) sensing using time-of-flight methods. At short ranges, the resolution was sub-millimeter and drops gradually with distance (2 cm at 10 m). For long range the TOF provides a constant resolution of ± 3 cm, independent of range. LARS was immune to sunlight and adverse lighting and has clear advantages over a microwave ladar system in terms of size, mass, power, and precision. The scanner had the ability to register high-resolution 3D images of range and intensity (up to 4000 x 4000 pixels) and to perform point target tracking as well as object recognition and geometrical tracking. For a single target, refresh rates up to 137 Hz could be obtained. The digitizing and modeling of human subjects, cargo payloads, and environments was possible with the LARS camera. LARS was used to obtain target-based measurements, feature-based measurements, and, image-based measurements like differential inspection in 3D space and surface reflectance monitoring of the ISS on the STS-52 shuttle flight in October 1992.

2.3.2 Laser Dynamic Range Imager (LDRI)

The Laser Dynamic Range Imager (LDRI), developed by Sandia National Laboratories [11], was used to remotely measure vibration of the international space station (ISS) structure and determine the structural mode frequencies and amplitudes. LDRI's specifications included six modes of operation, a 40° FOV, range resolution of 0.1 inches, images of 640 x 640 pixels, and a 7.5 Hz update rate. The sensor flew on the space shuttle in December 2000, and provided range video of the newly installed P6 truss and solar array panels during thruster firing. The measured vibration spectra captured the desired mode frequencies and amplitudes with a resolution of 0.02 to 0.1 inches. Additional measurements of curvature in the solar array panels demonstrated the potential for on-orbit characterization or inspection of structures.

2.3.3 NEAR Laser Rangefinder (NLR)

The NLR, developed by Johns Hopkins University Applied Physics Laboratory [12], was a direct-detection TOF laser altimeter that determined the range from the NEAR spacecraft to Eros surface by measuring the time it took for a round-trip of laser pulses with 0.312 m range resolution (single count). To determine the shape, mass, and density of Eros, as well as to characterize local and regional scale topography, NLR data was used to produce high resolution and high accuracy topographic grids and profiles. During the one-year observation of Eros, approximately 11 million measurements were obtained from NLR and combined with navigational information provided range measurements with 31.2 cm resolution and less than 6 m accuracy. Given this precision and accuracy for the NLR data, the mass of Eros was determined to be within 0.0001%, while mass density was determined to within 0.1%. NLR data products included a global topographic grid having 250 m resolution with 23.2 m rms best-fit radial accuracy and regional scale topographic models with 5 m accuracy.

2.3.4 LAser MaPper (LAMP)

LAMP, developed by the NASA Jet Propulsion Laboratory [13], was a flight qualified laser radar that could form 3D images by emitting high power, short duration laser pulses, which were directed by an internal gimbaled mirror in azimuth and elevation. The 2-axis gimbal allowed programmable, 2-axis motion (azimuth and elevation) of a 5 cm diameter beryllium mirror. It had an angular motion of 10° on both axes and a sweep rate of 10°/sec with the slow axis and up to 1000°/sec using the fast axis. Thus, it could scan a 10° x 10° area in 1 second. LAMP's processor is a 12.5 MHz, 32-bit MIPS R3000 Synova processor with memory and RS422 interface. The processor controlled the gimbal, monitors/controls, temperature, reads the laser levels, reads the timing chips, etc. LAMP was proposed as a sensor for guidance and navigation for several different space missions: capture of a Mars sample in Mars orbit, hazard avoidance sensor during smart landing on Mars, transverse planning for Mars rover, rendezvous or docking with another spacecraft in Earth orbit, and small body landing/exploration. In 2005, LAMP was the sensor onboard for the ST6/XSS11 ARX (autonomous rendezvous experiment). The objective of the ARX experiment was to demonstrate and characterize an autonomous rendezvous system that autonomously locates and rendezvous with a passive object [14].

2.3.5 Advanced Video Guidance Sensor (AVGS)

AVGS, developed by NASA/MSFC [15], was the continuation and advancement of the Video Guidance Sensor, which flew successfully on STS-87 and STS-95. AVGS was designed to be an autonomous docking sensor with updated electronics, increased range, reduced weight, and improved dynamic tracking ability. AVGS was designed to provide line-of-sight bearing from greater than one kilometer and provide 6 DOF relative position and attitude data from 300 hundred meters to dock. AVGS's sensor has two hardware modes and several software modes that equate into various operations. AVGS required an initial range estimate relative to the target (±20%) which was used to set firing parameters of laser power, integration time, and imager threshold. AVGS collects image data by firing an 808 nm laser at the target and capturing the returned image with a 1024 x 1024 CMOS imager

camera. The target's retroflectors have a filter, which is opaque to 808 nm frequency, and are not captured in the background image. Next, the AVGS fires an 850 nm laser at the target and captures that image with the same CMOS imager camera, including the retroflectors returns in its FOV. Then the two images are subtracted from one another leaving only the retroflector images. The resultant pixels that are touching one another (three or more) are called spots or blobs, and are preprocessed by an FPGA to pass these blob data to the DSP processor. The DSP processor determines pattern size, pattern validity for tracking, and computes the centroids for each set of image data. In tracking mode, the CMOS image can is reduced to regions of interest (ROI) around all of the target image reducing physical scan time allowing centroids of the target image to be processed at an increased rate. The ROI's are sized to allow the relative position of the target to shift in the FOV at a rate of at least 2°/sec.

2.3.6 Rendezvous Laser Radar (RVR)

The RVR, developed by the National Space Development Agency of Japan (NASDA) [16], detects the reflected light using a CCD camera and an Avalanche Photo Diode. The RVR estimates the line-of-sight angle by processing the CCD image and computes the relative range by comparing phase difference between transmitted and received beam. The RVR laser radar can measure out to a relative range of 660 m and within a line-of-sight angle of 4°. The unique feature of the RVR is its laser transmission method: it expands its laser beam in the specified angle and has no scanning system ("static type" laser radar). The RVR can function under the optical interference of the Sun and active optical sensors. The RVR measurement accuracy requirements for bias error were 10 cm in range and 0.05° in line-of-sight. The range differences between in-orbit data and ground data were within 2 cm and the line-of-sight differences were within 0.05°. Finally, the RVR conducted and properly targeted acquisitions at the docking position and handover point from GPS-relavtive rendezvous and dock at 500 m and 150 m holding positions for the EST-VII mission in 1998.

2.3.7 Triangulation and Lidar (TriDAR)

Neptec has developed a 3D Automated Rendezvous and Docking Sensor (ARD) system, under the 3D Automatic Target Recognition (3D ATR) project with the Canadian Department of National Defence, which includes the TriDAR 3D sensor which combines multiple ranging principles into a single scanning mechanism rather than a single ranging principle in multiple scanning techniques [17, 18]. The TriDAR combines a short range, high-precision auto-synchronous triangulation sensor with a mid- to long-range time-of-flight LADAR sensor in the same unit for frequencies of 1-5 Hz. The triangulation principle provides high-precision range measurements at close range; but the nature of the triangulation geometry means that imprecision grows approximately with the square of range; and even the best systems tend to only be practical up to the order of 10-20 meters. The TriDAR design takes advantage of the auto-synchronous scanning approach to keep both the detectors viewing the projected lasers while maintaining a small instantaneous field-of-view (FOV). The instantaneous FOV can be scanned through a much larger field-of-regard up to 30° using dual-axes scanning mirrors. This scanning capability makes the TriDAR a random access flying spot scanner. Coupling the sensor with Neptec's Intelligent 3D software there was a reduction in the amount of data collected and processed, drastically reducing the computational overhead of working in 3D. The result is 6 DOF real-time tracking and pose estimation at ranges from 150 meters to docked. An extended range LADAR module is also in development that will extend 6 DOF tracking out to 200 meters and also provides 3 DOF tracking at ranges up to two kilometers. Focused on recognizing known objects from a knowledge database and estimating their 6 DOF pose from long-range 3D data, this sensor system provided extensive characterization of the algorithms and was completed using both simulated and field data acquired with three different time-of-flight LADAR's. Neptec also developed a simulator for time-of-flight LADAR systems for the Canadian Department of National Defence.

2.4 Space-Based Autonomous Rendezvous and Dock Mission and Systems

An ideal rendezvous and docking system would provide relative 6 DOF pose to the guidance, navigation, and control system (GNC); operate autonomously; and provide multi-functional capability. This need for autonomous rendezvous capability has been recognized for some time and will be needed to meet and connect two spacecraft in orbit in the future. Several different systems have been designed, flown, and discussed throughout the literature.

2.4.1 Engineering Test Satellite-VII

On July 7, 1998, the Engineering Test Satellite-VII (EST-VII) [16] successfully performed the first autonomous rendezvous docking between uninhabited spacecraft. EST-VII performed autonomous RVD so that a navigation function measures and estimates the relative position and velocity between the chaser and the target as needed. ETS-VII had three separate navigation methods. Each of these navigation methods was selected automatically depending on the distance between the two satellites: the GPS relative navigation was used in the relative approach phase (from 10 km to 500 m), the RVR, a camera-type proximity sensor was used in the final approach phase (from 500 m to 2 m), and the PXS was used in the docking phase (within 2 m). RVD flight FP-1 was successfully completed on July 7, 1998, becoming the first autonomous RVD. In the FP-1 experiment, the chaser was pushed out from the target at a rate of 1.8 cm/sec. The chaser spacecraft then started to control relative position and attitude automatically and separated up to a 2 m holding point. The two satellites continued formation flight for 15 minutes keeping a constant distance of 2 m with range accuracy of a few centimeters using PXS navigation and relative 6 DOF control. In the docking approach phase, requirements for position control accuracy had to be within 10 cm, and actual control errors were about 2 cm. Attitude control requirements had to be within 2°, and control errors were within 0.5° on each axis. Random errors were within 0.2 mm in relative position and less than 0.1° in relative attitude. The authors speculated that the EST-VII project could be applied to advanced space vehicles such as the H-II Transfer Vechile (HTV), the OSV, and lunar or Mars explorers.

2.4.2 MRO Premier Rendezvous and Sample Capture System (RSCS)

The MRO Premier Rendezvous and Sample Capture System (RSCS), developed by CNES/NASA [19], was designed for a Mars premier orbiter mission. The target carried a radio beacon, enabling a suite of radio and optical sensors to locate, track, and autonomously rendezvous with the released target. The first objective was to deliver four Netlander science stations to the surface of Mars, and provide at least a year of science-telemetry. The second objective was to demonstrate rendezvous technologies for future Mars-Sample-Return missions. The rendezvous system used one-way radiometric and optical observables for distant measurements and optical methods exclusively for close observations. The hardware configuration of this system consisted of two cameras, a duplicate of the Mars Reconnaissance Orbiter (MRO) Optical Navigation camera, and a wide angle camera, thought to be a version the the Mars Exploration Rover lander camera. The RSCS was a combined ground and flight software system, and when combined with the optical sensing instruments onboard, provided a means of manual ground-in-loop tracking and maneuvering. It also provides automatic closed loop tracking and maneuvering, and in the latter case, accomplished close-proximity operations including simulation of a capture. In conjunction with the optical measurements, one-way doppler measurements were taken with the Electra instrument onboard the carrier spacecraft. However, the system still required a combined ground and flight software to accomplish the rendezvous and sample capture process.

2.4.3 Demonstration of Autonomous Rendezvous Technology (DART)

Demonstration of Autonomous Rendezvous Technology (DART) program was developed to demonstrate technologies required for spacecraft to locate and rendezvous with another spacecraft without direct human guidance [20]. The vehicle was scheduled to perform a series of orbit transfers to arrive at a point near a target satellite and demonstrate a collision avoidance maneuver extremely close to the target vehicle using the AVGS hardware. On April 15, 2005, DART was launched aboard a Pegasus rocket and reached vicinity of the target spacecraft where AVGS was used for navigation to demonstrate many different rendezvous and docking capabilities. Some of these tests were to demonstrate AVGS

capabilities, demonstrate various approach techniques, demonstrate station keeping on the nadir axis at a distance of 50 m from the target, demonstrate station keeping on the velocity axis at a distance of 15 m from the target, demonstrate station keeping on the docking port axis at a distance of 5 m from the target, and demonstrate autonomous operations following loss of AVGS lock and requisition. DART successfully completed the location and rendezvous phases of its operations, closing to within approximately 92 m (300 ft) of the mission's target: the Multiple-Path Beyond-Line-of-Site Communications (MUBLCOM) satellite. The MUBLCOM satellite design had retroflectors for use with the AVGS installed along the edge of the central ring. DART's AVGS instrument was also able to acquire the MUBLCOM satellite, accomplishing one of the mission's key RVD objectives. However, the DART spacecraft was unable to complete all of its close proximity and circumnavigation operations near the MUBLCOM satellite due to a depleted fuel situation. The entire sequence was designed to be accomplished under autonomous control.

2.4.4 Orbital Express

NASA was investigating a fully integrated repair and replacement program for satellite systems in the Orbital Express program, scheduled for launch and flight-testing in October 2006 [21]. The Orbital Express Demonstration system program has the goal of validating the technical feasibility of robotic, autonomous on-orbit refueling and reconfiguration satellites. To accomplish this, a prototype servicing satellite (ASTRO) and surrogate next-generation serviceable satellite (NEXTSat) have been designed. The key element to enabling the ASTRO satellite to successfully approach and capture the NEXTSat client satellite is the Autonomous Rendezvous and Capture Sensor System (ARCSS). The ARCSS system provides relative state information for NEXTSat from ranges of hundreds of kilometers until capture. ARCSS consists of three imaging sensors: a narrow field of view acquisition and tracking sensor (VS1), a mid to short-range wide field of view visible track sensor (VS2), and an infrared sensor (IRS) for continuous situational awareness during day and nighttime operations. ARCSS is designed to acquire a client satellite at ranges extending beyond 200 km by using its narrow field of view visible sensor, where the client satellite appears as

a point source. Eventually, the client satellite will come into range of the infrared sensor and laser rangefinder. The laser-based tracking system, adopted form DART, was used to independently measure the attitude, range, and bearing of NEXTSat for short-range and capture operations of less than 200 m. The Orbital Express system will demonstrate for the first time: 1) fully autonomous rendezvous out to 7 km with a capability that could support rendezvous at separation distances up to 1,000 km; 2) soft capture and sub-meter range autonomous station-keeping; and 3) on-orbit refueling and component replacement as well as other robotic operations using as its primary sensor AVGS. The Orbital Express mission lifted off on 08 March, 2007, from Cape Canaveral Air Force Station in Florida with an agenda to demonstrate autonomous RVD. Upon successful demonstration of all scenarios, Orbital Express will provide the foundation for developing an operational system that can provide routine on-orbit servicing of existing and future space assets. The Orbital Express completed its final and most challenging unmated rendezvous and capture scenario on 02 July, 2007. The operation was completely autonomous, with the two satellites operating at distances of up to 7 km apart, and often with only passive optical and infrared imaging for guidance. The mission marked the second successful grapple and capture of the NextSat by the ASTRO, using its robotic arm. Finally, on the 22 July, 2007, the Orbital Express satellites performed their end-of-life maneuvers and have been decommissioned [22].

2.4.5 XSS-11

The purpose of the AFRL XSS-11 demonstration mission, launched April 11, 2005, was the development and on-orbit verification of guidance, navigation, and control capabilities that would enable a micro-satellite to safely and autonomously rendezvous with multiple space objects [23]. Utilizing a scanning Ladar to acquire and track the object, XSS-11 automatically transitions from rendezvous into closed loop proximity operations. The ground or on-board planner can select from a menu of GNC modes that command the vehicle to station-keep at way points, or circumnavigate the object using natural or forced-motion trajectories. Attitude determination was initially achieved with the Lost-in-Space algorithm which utilizes between 5 and 7 stars and performs pattern matching of

the presented star measurement information using a star catalog that is stored onboard. Attitude guidance for all slew modes is done with a rate limited slew about the eigen-axis of the attitude error quaternion. The attitude error quaternion is formed by differencing the desired quaternion form the actual body quaternion. Relative navigation is performed by combining ladar range and bearing measurements with the propagated inertial states in an extended Kalman filter. The EKF consisted of three orbit propagators that keep track of absolute XSS-11 relative states, and filtered non-truth position and velocity information in Earth Centered Inertial coordinates. The system also includes on-board collision avoidance system that can trigger an abort burn due to a safety constraint violation. The mission duration was slated to be 12-18 months and by the fall of 2005 it had performed over 20 rendezvous maneuvers with the Minotaur 4th stage, several days of circumnavigation operations, and hours of active, closed-loop station keeping at various ranges and vantage points. Current plans suggest that XSS-11 will conduct rendezvous and proximity maneuvers with several U.S.-owned dead or inactive target objects near its current orbit [3, 23].

2.4.6 Hubble Robotic Service Vechicle (HRV)

NASA proposed using multiple 2D cameras and a ladar sensor as the baseline system for the Relative Navigation Sensor for Hubble Space Telescope (HST) Robotic Vehicle (HRV) [24]. The NASA HRV mission had two primary objectives. The primary objective of the HRV was to dispose safely of the HST at the end of its science mission by providing a controlled re-entry capability to the HST. The secondary objective was to extend the scientific life of the HST through ORI/ORU replacement or augmentation. The HRV consisted of two modules: the De-Orbit module and the Ejection Module. The HRV was going to be capable of an approach and capture of the HST that is in an inertial hold control mode or uncontrolled and tumbling at a rate of up to 0.22°/sec per axis. After HST attitude rates are determined, the HST capture axis would be propagated in time in order to select a relative point in space and a corresponding time from which to initiate the capture approach. In the case of a tumbling HST, the analysis showed that accurate propagation of the capture axis was extremely sensitive to the magnitude and accuracy of

the estimated HST angular rates. The Relative Navigation Sensor System would provide the relative position and attitude data during the proximity operatives and capture phases. It consisted of a Ladar system, laser camera system, and vision processing unit that utilizes nine digital cameras and halogen illuminators, but the project was canceled.

Summing up this section of the literature review, several different range sensor technologies have been discovered leading to exciting low and high-cost sensors that have been used in space applications. Using these sensors, several aerospace companies have developed systems using sensors such as AVGS, LAMP, etc, to try and perform autonomous rendezvous and dock to date.

2.5 3D Pose Estimation Algorithms

3D pose estimation is the process of determining the position and orientation (rotation matrix/quaternion and cartesian translation) between two separate objects or the same object at two sequential times. Many position and orientation algorithms require point correspondences (the relationship between points in a point-cloud from one time instance to the next) to determine the 6 DOF of a rigid body (pose) using 3D range images gathered from ladar systems.

In 1984, Blostein and Huang [25], presented four different algorithmic methods for determining general 3D motion (rotation and translation) based upon rigid body motion mechanics and point correspondences. The first was the direct linear method that set up four 3D systems of linear equations in order to solve for the twelve unknown motion parameters of the ordered pair $(R(\vec{q}), p(x, y, z))$ where $R(\vec{q})$ was the rotation matrix and $p(x, y, z)$ was the translation vector. In order to solve these linear equations, four non-coplanar pairs of point correspondences were needed to solve the system of equations. The second method was based upon translation invariants. A 3D unit vector was constructed between any two points on the rigid body. Therefore, only three point correspondences were needed to solve directly for the rotation matrices parameters, and consequently the translation matrix was obtained through direct substitution. The third technique, adapted spherical projection, used the assumption that the rotation matrix can be represented as a

eigen-axis unit vector and a rotational angle about that vector. Using the 3D unit vectors constructed in the previous algorithm with the rotation axis, one could simply solve for the unknown parameters. The fourth approach looked at the 3-D motion with parameters based upon a screw decomposition of the rotation matrix, requiring knowledge of only three point correspondences. The four different algorithms derived could be used at different times for different cases depending on whether coordinate frames and origins could be established, as well as the number of point correspondences which could be determined from the range data.

In 1986, Huang, Lin, and Lee [26] presented a way to find 3D point correspondences in position and orientation estimation based upon a direct linear method. This algorithm utilizes a matrix eigen-decomposition to calculate four possible solutions to the rotation matrix with the understanding that the two centroids at different instances in time are governed by the correct rotation matrix solution. In the algorithm, point correspondences were assumed known, but in order to calculate the centroids at both time instances this knowledge did not need to be known. To determine the correct solution out of the possible four solutions, the points are rearranged until the euclidean norm between the two data sets was minimized.

Also in 1986, Huang et al., as well as Blostein and Margerum [27], developed two separate point-cloud-based algorithms. The first was based on the frequency domain Fourier transform, and the second algorithm was based upon a least squares estimation which required point correspondences. The reason that point correspondences were not necessary in the Fourier approach was because they are not needed to calculate the Fourier transform. The algorithm sets up two functions based on the 3D coordinates given, and takes into account that the Fourier transforms of these two functions must undergo the same rotation and translation. The second algorithm solves for the rotational and translational components by setting up a least squares estimate term based on the six motion parameters (three for translational and three Euler angles for rotation(ψ, θ, ϕ)). This iterative algorithm takes into account that the rotation matrix can be decoupled into three successive rotations about

the z, y, and x axis by the corresponding Euler angles. In this process two of the Euler angles were held constant at initial guesses while the other angle was solved for through an iterative process until the least squared error was minimized.

In 1987 Arun, Huang, and Blostein [28] developed a least-squares algorithm that used the singular value decomposition (SVD) of a 3 x 3 covariance matrix constructed from the two point sets to compute the rotation and translational components. In 1991, Umeyama [29] combined and modified the SVD algorithm by checking certain terms calculated by the SVD to ensure that the degenerate (reflection) case did not happen when the data was either noisy or co-linear. Both algorithms required that point correspondences be known.

In 1989, Huang and Lee [30] developed two position and orientation algorithms using 3D point cloud orthographic projections. In the case where two orthographic views were known, the rotation matrix was solved for by simply moving both sets of objects points to the origin without loss of generality. The second algorithm used orthographic projections once again, but takes them at three different time instances instead of two. Once again, linear combinations were constructed to solve for the rotation. In the case where two orthographic projections were used, there are an infinite number of solutions off by only a scaling factor. When three orthographic projections and four point correspondences were used, it ensured a unique solution plus the existence of reflection for the rotational component.

At the end of the decade (1989) and into the next decade (1991), engineers and scientists began looking at the impact noise might have upon motion estimation algorithms [31, 32]. Huang, Weng, and Ahuja developed a closed form solution for motion parameters by exploiting redundancies in the data to obtain better estimates when noise was present. To account for noise in the algorithm, the error in the motion parameters was estimated based upon the standard deviation of the error and the variance of the errors.

In 1987 [33–35], Horn, Hilden, Negahdaripour, Walker, et al. developed three least-squares pose estimation algorithms. The first of these algorithms used the SVD closed form solution, but instead of using Euler angle, rotation axis and angle, or screw decomposition they used the quaternion representation for rotation. The second least-squares technique

used orthonormal matrices like the technique developed by Weng, Huang, and Ahuja [36]. Both these algorithms incorporate the necessary rigid body constraints, use all the data collected, and calculate the optimal motion transformation.

Smith and Nandhakumar in 1994 [37], developed a new and innovative algorithm that took into account that laser radars do not instantaneously acquire the range images, but sequentially scan an object through single laser shots. The method developed was an iterative, linear, feature-based approach which used the non-zero image acquisition time constraint to accurately recover motion parameters from the distorted structure of 3D range images. A set of linear equations was set up to solve for an estimate of the rotational and translational components. Then, the velocity estimates looped back into the iterative process to refine the rotational and translational components until the change in any one motion parameter resided below some threshold; however, their method still required point-correspondences.

Liu and Rodrigues in 1999 used a geometrical technique to estimate motion from range images [38]. In this geometrical algorithm, a reflected correspondence pair, a correspondence vector, and reflected correspondence vector are defined such that the pole of displacement vector and rotation axes can be calculated. Because the data collected most likely contained noise, the algorithm had a fuzzy reasoning approach blended into it to correct the solutions and more accurately calculate motion parameters. Zhang, Liu, and Huang in 2002 [39] presented an algorithm which showed that the 3D structure of a corner could be recovered by introducing a new coordinate system, and by knowing only this one corner and two point correspondences over two separate views was sufficient enough to determine the rotational and translational components. Because many spacecraft are shaped like giant rectangular boxes, this algorithm looked to be promising because of its use of orthonormal relations and 3D corner structure, but was not investigated for this thesis because the author was looking into an algorithm with more generalization for 3D objects.

2.6 Literature Review Summary

From the literature review, a determination was made that the best algorithm suited for this type of work was using the Singular Value Decomposition (SVD) or the Eigen-Value Decomposition (EVD) methods; however, both methods required point correspondences between the sequential point sets. In 1992, two independent groups [40, 41] developed and completely described an iterative algorithm using the SVD and EVD to calculate pose known as the Iterative Closest Point (ICP) Algorithm where point correspondences did not have to be known. The algorithms and literature will now be reviewed in detail in the following chapter.

Chapter 3

Iterative Closest Point (ICP) Algorithm

3.1 Taxonomy of ICP Variants

The Iterative Closest Point (ICP) algorithm has become one of the dominant registration methods in the literature for aligning a pair of range images or globally aligning several 3-D point based range images. Originally developed by Chen and Medoni [41] and Besl and Mckay [40], the ICP algorithm takes two sequentially acquired range images and calculates the registration parameters: specifically the quaternion (describing the rotation matrix) between the two targets and the cartesian translation vector. In 1994, [42] used both the range information and the intensity images in a version of the ICP algorithm. In 1994, [43] developed zippered polygon meshes which became extremely important in the ICP algorithm when constructing meshes of solid objects and calculating point-to-plane distances. In 1996, [44] described how the ICP algorithm could be used in medical applications to build 3D representations of bones at different times throughout a surgery. In 1996, [45] described a method to incorporate the k-d tree structure into point matching in order to speed up point correspondence computational times. In 1996, [46] used the ICP algorithm as a basis for multiple image matches and derived a numerical algorithm using the point-plane error metric. In 1997, [47] developed a global image registration method based upon the ICP algorithm. Then in 1998 [48], he developed a similar global image registration method using the eigen-value decomposition (EVD) method as opposed to the singular value decomposition (SVD) method. In 1997, [49] performed a sensitivity analysis of the ICP algorithm in the presence of data corrupted by noise for the first time. In the same year, he also developed a system which used the ICP algorithm and meshing mathematics to merge multiple range images together for image registration [50]. In 1997, [51] used the traditional ICP algorithm and modified it to use the color information provided

by the imager. In 1997, [52] used luminance and depth images incorporated into the ICP algorithm in order to determine pose. In 1997, [53] discussed the mathematics behind how to calculate the distances from points to surfaces, and one surface to another. In 1999, [54] used an ICP bootstrapping method to optimally estimate rigid body motion. The same year, [55] used the ICP algorithm for multiple registration of extremely large data sets not seen or used previously. In 2000, [56] used the algorithm on the statue Michelangelo to build a full 3D CAD image. In 2002, [57] discussed the trimmed ICP algorithm which used a trimmed least squares error metric. In 2002, [58] developed a multi-resolution ICP scheme to determine registration. In 2003, [59] refined the point-to-point ICP algorithm to be more selective in the point-to-point matches determined in the ICP algorithm by use of weighting matrices. In 2003, [60] used the ICP algorithm to model non-rigid objects from multiple range images. In 2004, [61] used a sensor projection method to improve the ICP algorithm for registering range images globally.

3.2 ICP Algorithm Overview

Since the ICP algorithms introduction, several different variants have been developed improving or modifying basic concepts and steps within the algorithm. Rusinkiewicz, for his dissertation [62], studied the computational effect that theses variants had on each step and iteration within the ICP algorithm. The steps within the ICP algorithm are as follows: (1) initialization, (2) point selection, (3) calculating point-to-point or point-to-plane correspondences, (4) calculating registration parameters, and (5) minimizing an error metric (see fig. 3.1).

3.3 Initialization

Given a pair of range images, the first step in general was to initialize the algorithm by generating an initial alignment between the two point clouds. One constant difficulty in using the ICP algorithm revolves around the algorithm calculating registration parameters that provide a global minimum, not a local minimum. Several initial alignment methods have been used when trying to assure global convergence of the ICP algorithm, such as

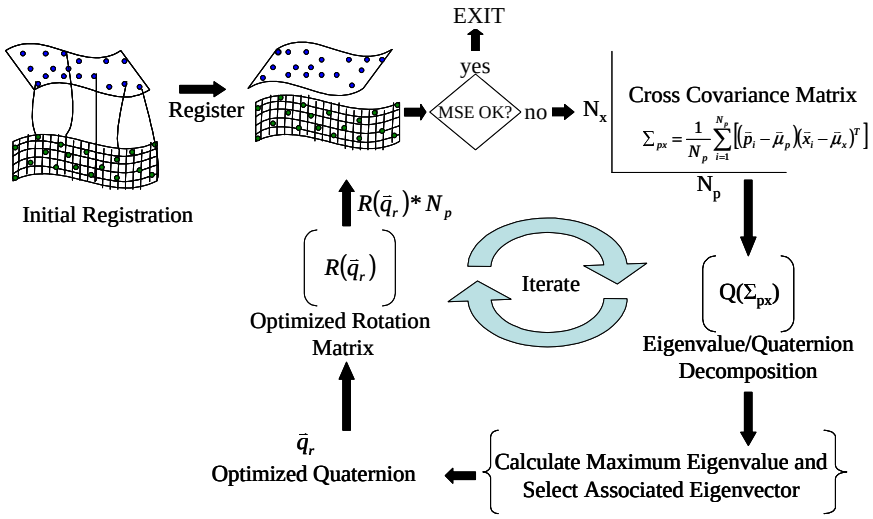

Fig. 3.1: Iterative closest point algorithm flow chart.

tracking scanner position and indexing surface features [63,64], spin-image surface signatures [65, 66], computing principal axes of scan of ladar scanned range images [50], searching for corresponding points between images [67], calculating image corners, [39], and manually inputting the initial alignment.

However, in this thesis, manually inputting the initial registration quaternion was chosen because of its simplicity in application; however, this becomes a brute force approach to finding the initial registration. Thus, many equally spaced initial quaternions were constructed and used within the ICP algorithm. Because the point clouds used contain such large numbers of points with an associated range error, a set of 312 initial registration quaternions were constructed from all normalized combinations of the following quaternion sets $q_0 = \{1, 0.5, 0\}, q_1 = \{1, 0.5, 0, -0.5, -1\}, q_2 = \{1, 0.5, 0, -0.5, 1\}$, and $q_3 = \{1, 0.5, 0, -0.5, 1\}$. The quaternion can be described as a 4D vector $[q_0; q_1; q_2; q_3]$ that describes the rotation of a rigid body in 3D space. For the purpose of the ICP algorithm, each term q_0 to q_3 was given a value equally spaced from -1 to 1 in increments of 0.5. This subdivides the infinite number of possible rotations in 3D space. The All combinations in which q_0 was negative were thrown out because they were equal to another quaternion

combination where the q_0 term was positive (i.e., the 3D rotation transformation was the same).

For example, the initial quaternion has the form:

$$\vec{q} = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix}. \tag{3.1}$$

Now choose one value from each of the sets listed above :

$$\vec{q} = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}. \tag{3.2}$$

Finally, the quaternion has to be normalized (3.3) such that $q_0^2 + q_1^2 + q_2^3 + q_3^2 = 1$ and perform the rotational transform to the first point set data:

$$\vec{q}_{norm} = \vec{q}/\|\vec{q}\|. \tag{3.3}$$

This normalized quaternion can now be used for an apriori guess as to the rigid body rotation. Therefore, the normalized apriori quaternion can be applied to the first of the two point clouds (R_1) to be used in the ICP algorithm using eq. 3.4 where $R(\vec{q}_{norm})$ is the rotation matrix corresponding to the normalized quaternion.

$$R_1 = R(\vec{q}_{norm}) * R_1 \tag{3.4}$$

Now by exhaustively using all initial alignment possibilities, the algorithm can successively search for the global minimum to the registration problem at the expense of computational time. This process is accomplished as follows: (1) given two point clouds there is a set of registration parameters relating the two point clouds to each other, (2) apply one of the 312 apriori guess to the first point cloud, (3) run all steps in each iteration of the ICP algorithm discussed in secs. 3.4 - 3.7 for this single apriori guess to obtain a solution, (4) continue applying n apriori guesses repeating step 3 until all 312 apriori guesses have been used, and (5) select the solution that provides the global minimum.

3.4 Point Selection

Once an initial alignment has been found, the ICP algorithm chooses the number of points from each point-cloud to be used for point-to-point matching, or building meshes for point-to-plane matching. This step of the algorithm can be the simplest step, but has the greatest impact on computational speeds. Thus, three possible strategies for 3-D range images are discussed. First, there is the case where all the points provided are used, which has the worst computational speeds, with a computational order (algorithms computational burden) of $O(N_s N_p)$ [40]. Next, there are two separate sub-sampling strategies, which can be used to ease the computational burden of the point selection process. These are uniform random sampling [43] (or random sampling [45]), and normal-space sampling. Uniform random sampling simply selects points in a uniformly random way, without regard to any feature information. Normal-space sampling first sorts the points into bins according to their surface normals. Then these bins are uniformly sampled so that the resulting subset has a more uniform distribution of surface normals. Uniform random sampling on a surface has order $O(N_s)$, where N_s was the number of points and normal-space sampling is $O(N_s^2)$.

Once the selection of points has been performed, a k-d tree structure was used to store range images in this structure [68]. The k-d tree is a generalization of a binary-search tree for efficient search in high dimensional (multi-dimensional) spaces. The k-d tree was created by recursively splitting a point-cloud data set down the middle of its dimensions of greatest variance until the leaf nodes contain a small enough number of data points (see fig. 3.2) [69]. The figure below represents a general point cloud in 3D space (e.g., feature points on satellite). In the figure, the white lines show how the feature points in the point-cloud have been used to create bins for the rest of the points within the point-cloud. The k-d tree structure can then be stored into a buffer or computer memory. This allows nearest-neighborhood searches to be accomplished more efficiently by matching the appropriate branch/bin with the corresponding point or plane. This reduces the number of point matches that have to be done during each iteration because once a point or plane has been matched to a specific bin/branch. The algorithm continues to go back to the

points within that branch/bin and no other for the remainder of the ICP algorithm. The expected number of operations for the nearest neighbor search was $O(logN)$ as compared to the $O(N_pN_x)$ which was the computation order when comparing every point in one data set with every point in the other data set.

3.5 Point Matching

After selecting an appropriate number of points and applying an initial rotation, the ICP algorithm finds the closest or most compatible set of matches between the points in the two sets of data $R_1 = (x_1, x_2, \cdots, x_n)$ and $R_2 = (y_1, y_2, \cdots, y_m)$ obtained from the object (e.g., satellite). In order to match points in the first data set to the points or planes in the second data set, either a point-to-point distance can be utilized (sec. 3.5.1), or a point-to-plane distance calculation can utilized (sec. 3.5.2).

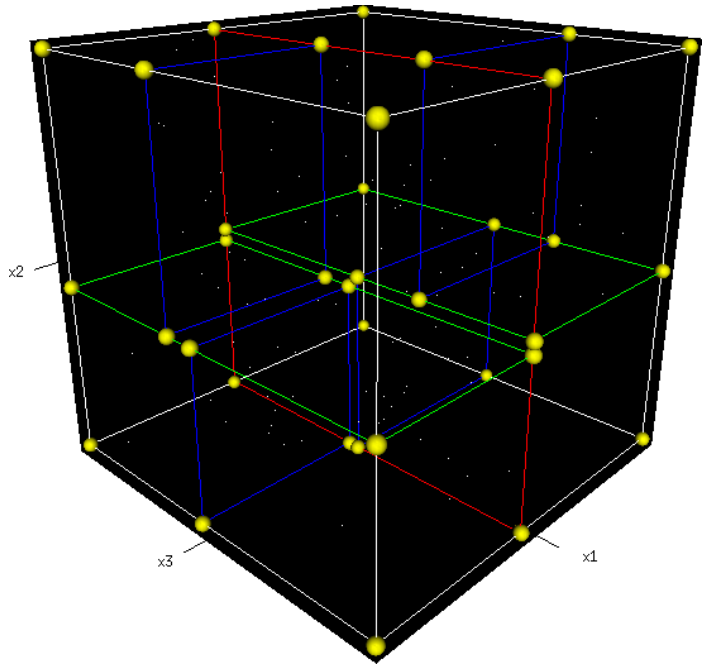

Fig. 3.2: Example of a 3D k-d tree being constructed from a 3D point cloud data set.

3.5.1 Point-to-Point Correspondences

The first method, point-to-point matching, begins by calculating the distance from one point in the first point cloud to those points in the second point cloud by (3.5). Remember though, each point has been matched mathematically through the k-d tree construction and search algorithm to a specific branch/bin. Therefore, the point matching distance calculation occurs between each point in the first point cloud and only those points in the second point cloud residing in the specific bin or branch (See fig. 3.3.). The point matching selection is accomplished by selecting the point match with the minimum distance between the two points (i.e., $min(d(x_i, y_i))$ for all i, j. During this process, several points in the first point cloud have the possibility of being matched to the same point in the second point cloud during one more iterations of the ICP algorithm.

$$d(x_i, y_j) = \sqrt{(x_{i1} - y_{j1})^2 + (x_{i2} - y_{j2})^2 + (x_{i3} - y_{j3})^2} \text{ for } i, j = 1 : max(m, n) \qquad (3.5)$$

When each point in R_1 had a corresponding matched point (see fig. 3.3) in R_2, the algorithm proceeds to the next step in the algorithm: calculating the corresponding registration between the matched point sets. However, the point-to-point matching operation has to be done during each iteration of the ICP algorithm until an error metric between the two data sets has been minimized. Applying this technique leads to longer computa-

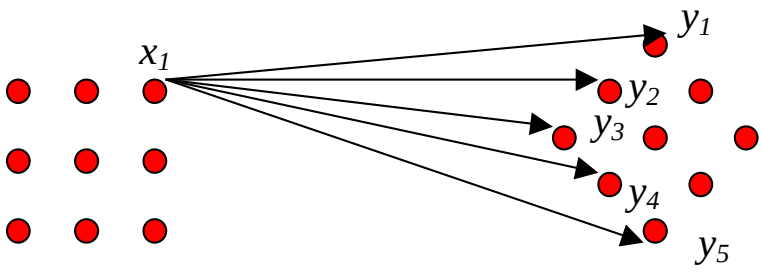

Fig. 3.3: Example of point-to-point matching.

tional times and a greater mean-squared error then the point-to-plane matching according to an analysis done by Rusinkiewicz [62]. These methods are outlined by past authors have used a variety of methods that incorporate some variation on the point to plane methodology [41, 49, 50, 53, 56].

3.5.2 Point-to-Plane Correspondences

The point-to-plane correspondence step is dependent upon whether the 3D rigid object did or did not have a 3D CAD model/finite element model with all the appropriate triangle vertice information. If the object has a 3D CAD model, most software programs can directly import the finite element model elements with all associated triangle vertices. For the case where the object geometry was unknown,no finite element/3D CAD model, the algorithm needs to create and store triangular mesh information for each range image to build up a database of 3D mesh information that can be used in the point-to-plane matching methodology. Two simple yet different software tools were used for constructing these triangular meshes: (1)"trimesh" a MATLAB command, and (2) "trimesh2" a C++ library constructed by Rusinkiewicz for construction and manipulation of triangular meshes. Once one of these two programs was employed, a triangular mesh of the point-cloud data might appear as follows (see fig. 3.4):

In the calculations outlined below, there are two separate coordinate frames that are used for these calculations. The first coordinate system is the "inertial" coordinate frame represented with variables sub/superscripted with a capital "I" (e.g., X_I is the x-axis unit vector for the inertial coordinate system). The first coordinate system is centered at the location of the ladar, and the second coordinate system is centered around one of the three vertices. The second coordinate system is the "plane" coordinate frame constructed from three vertices on the plane (A_I, B_I, and C_I) by (3.6). These variables are represented with a sub/superscripted capital "P" (e.g., X_P is the x-axis unit vector for the plane coordinate frame). In the description of this methodology, it is always been stated that there are two separate point clouds that the algorithm calculates the position and orientation between. Whereas, in the description of the meshing algorithm, an explanation is given that suggests

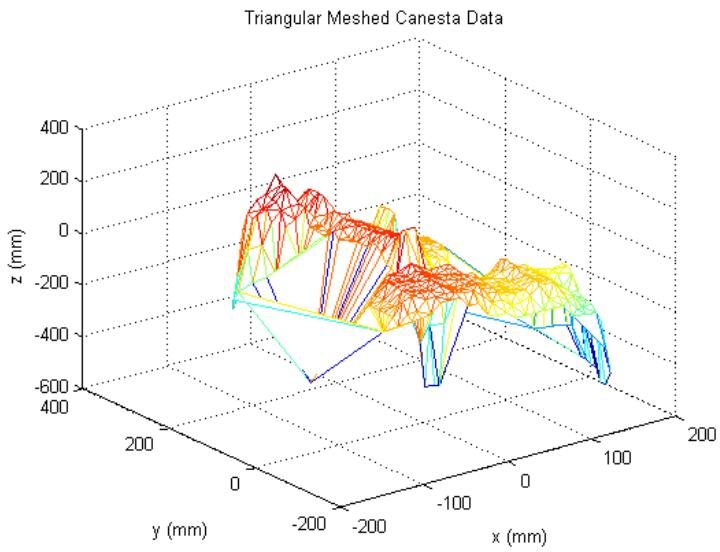

Fig. 3.4: Example of a triangular mesh constructed from data taken by a canesta ladar system.

the building of database of meshes for this rigid object that are used instead. They coincidentally are one in the same. If no database was initially provided, then the algorithm begins with the first two point clouds, and the first point cloud becomes the initial database. Now, once the pose of the 3D rigid object is calculated and a global minimal solution calculated, the previous two point clouds can be saved together in the database or the second point cloud can be stored as the only point cloud in the database. The second point cloud then becomes the first point cloud in the next pose estimation when a third point cloud is obtained. After meshing the first two 3-D point-clouds and building a database, the point-to-plane distances and correspondences are calculated using the steps outlined below.

First, choose one of the planes from the first 3D point-cloud mesh (labeled "plane" in fig. 3.5) where $\vec{A_I}$, $\vec{B_I}$, and $\vec{C_I}$ are the three triangular vertices for that plane in the inertial coordinate system. Then, choosing a point $({}^{I}R^{i}_{point})$ from the second point cloud mesh where $i = 1 : N$ and N is the number of points in the second point cloud for the point-to-plane distance calculation in (3.6). This point-to-plane distance calculation can then be done for each plane/mesh in the first point cloud with the points in the second point cloud.

Also, the k-d search methodology is used to speed up computations by telling the algorithm, which points in the second point cloud should be matched with each individual plane as is done in the point-to-point matching method. After the point-to-plane distance calculation is done, each plane was matched to a point in the second point cloud by selecting the match that satisfied $min(d({}^{I}R^{i}_{point}, A_I))$ for all i and each plane/mesh.

$$d({}^{I}R^{i}_{point}, A_I) = \vec{n} \cdot ({}^{I}R^{i}_{point} - B_I), \tag{3.6}$$

where $\vec{n}$ is the chosen planes surface normal.

In order to perform the registration step of the algorithm, the algorithm needs to have two sets of points matched to one another to construct the cross-covariance matrix among other mathematical terms described in sec. 3.6. However, the following method matches points-to-planes; therefore, a point on the surface of the plane had to be mathematically calculated to have two sets of matched points. So to begin the second coordinate system, generally a triad of orthogonal coordinates, which defines the a plane/mesh coordinate system, was constructed using the three plane vertices A_I, B_I, and C_I in (3.7).

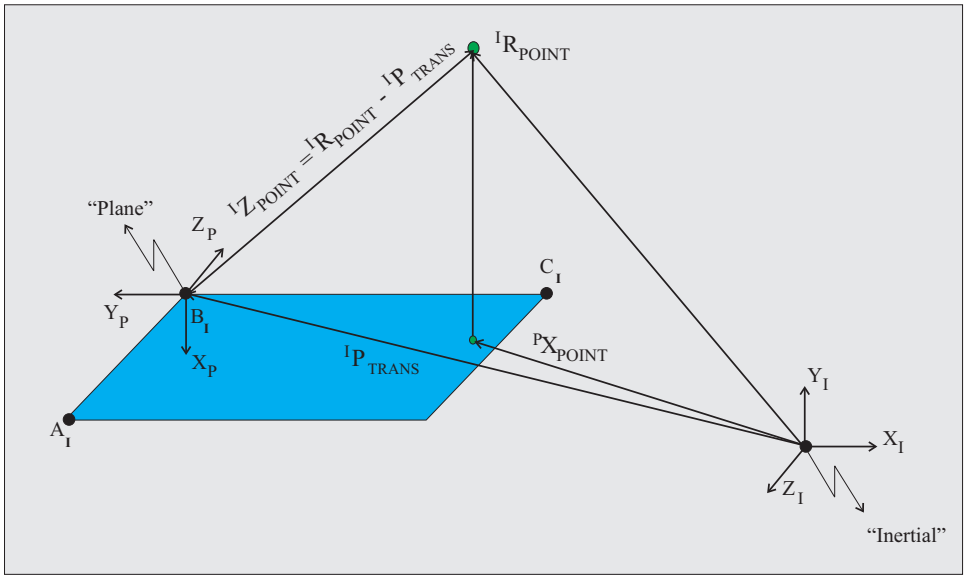

Fig. 3.5: 3D Representation of a triangular mesh and point-cloud point used for point-to-plane distance calculation.

$$Z_P = \pm\frac{\left(\vec{A_I} - \vec{B_I}\right)}{\left|\vec{A_I} - \vec{B_I}\right|};\ X_P = \pm\frac{\left(\vec{A_I} - \vec{B_I}\right) \times \left(\vec{C_I} - \vec{B_I}\right)}{\left|\left(\vec{A_I} - \vec{B_I}\right) \times \left(\vec{C_I} - \vec{B_I}\right)\right|};\ Y_P = \vec{Z_P} \times \vec{X_P} \tag{3.7}$$

Using a specific combinations of these three vectors, $T_{I\rightarrow P}$ the transformation (rotation) matrix between the plane/mesh fixed coordinate system and ladar-fixed coordinate system is constructed.

$$T_{I\rightarrow P} = \begin{bmatrix} X_P^T \\ Y_P^T \\ Z_P^T \end{bmatrix} \tag{3.8}$$

Now, that the two coordinate systems have been calculated an the orientation between those coordinate systems, the process of calculating the point on the planes surface can be accomplished. Assuming that $\vec{B_i}$ is the center of the plane coordinate system, then the vector ${}^I p_{trans}$ is the distance between the center of the inertial coordinate system and the plane coordinate system. The vector ${}^I Z^i_{point}$ is the difference between the vector $\vec{B_i}$ and the point ${}^I R^i_{point}$ and can be calculated by (3.8). This point was then transformed or expressed in the plane coordinate frame eq. 3.7 by using the transformation matrix constructed in (3.9), and now the variable has the notation ${}^P Z^i_{point}$ because it is viewed in the plane coordinate frame. Finally, by projecting the vector ${}^P Z^i_{point}$ onto the plane, the point ${}^P X^i_{point}$ in the plane coordinate frame has been found. The projection of ${}^P X^i_{point}$ is accomplished and calculated by (3.10).

$${}^I Z^i_{point} = {}^I R^i_{point} - {}^I p_{trans} \tag{3.9}$$

$${}^P Z^i_{point} = T_{I\rightarrow P}\left[{}^I R^i_{point} - {}^I p_{trans}\right] = T_{I\rightarrow P}\left[{}^I Z^i_{point}\right] \tag{3.10}$$

$${}^P X^i_{point} = \begin{bmatrix} 0 \\ e_y \cdot {}^P Z^i_{point} \\ e_z \cdot {}^P Z^i_{point} \end{bmatrix} \tag{3.11}$$

Finally, rotate and translate ${}^{P}X^{i}_{point}$ from (3.11) into the inertial coordinate system via (3.12) to obtain ${}^{I}X^{i}_{point}$. Therefore, for each minimum selected point-to-plane match that is done through iteration, there is now a corresponding point-to-point match for ${}^{I}R^{i}_{point}$ and ${}^{I}X^{i}_{point}$.

$$ {}^{I}X^{i}_{point} = {}^{I}\rho_{trans} + T_{P\rightarrow I}\left[{}^{P}X^{i}_{point}\right] \tag{3.12} $$

The above derivation was done with more of an engineer's perspective. However, this point-to-plane distance calculation step was also presented by Neugebauer [53], Chen [41], and Besl/McKay [40] assuming minimal knowledge in mathematics or computer science. Each iteration k in the literature was described as finding the closest point in a point set T for each point to a surface $S_{(k-1)}$, which represents the initial position of the registration surface for this iteration. Let the point sets $s_{(k-1)i}$ and $t_{(k-1)i}$ be the points selected respectively from $S_{(k-1)}$ and T, where $s_{(k-1)i}$ is paired with $t_{(k-1)i}$ for $i = 1, 2, ..., N$. The set $s_{(k-1)i}$ was chosen to be the vertices in $S_{(k-1)}$, then the corresponding $t_{(k-1)i}$ is the point contained on one of the triangles in T that has a minimum distance to $s_{(k-1)i}$. If we let t be a triangle in T defined by the vertices $\vec{t_u}$,$\vec{t_v}$, and $\vec{t_w}$, the distance between $s_{(k-1)i}$ and t is given by (3.13):

$$ d(s_{(k-1)i}, t) = \min_{u+v+w=1} \|u\vec{t_u} + v\vec{t_v} + w\vec{t_w} - s_{(k-1)i}\|, \tag{3.13} $$

where $u, v, w \in [0, 1]$. To find the point $t_{(k-1)i}$ to be paired with each $s_{(k-1)i}$, the triangles in the targeted surface T are searched through for the triangle τ that gives the minimum distance. The point $t_{(k-1)i}$ is then selected as

$$ t_{(k-1)i} = u\vec{\tau_u} + v\vec{\tau_v} + w\vec{\tau_w}, \tag{3.14} $$

where $d(s_{(k-1)i}, \tau)$ was minimized.

After using either the point-point or point-plane distance calculations a distance threshold was used, which allows certain point matches that could be completely incorrect to be

rejected. Similarly, a point pair culling percentage can be used, which culls a given percentage of point matches that have the greatest distance between points according to some multiple of the standard deviation [45]. Also, point pair matches can be rejected when they fall on the boundaries of the triangular meshes [43]. By selecting small subsets of the point matches available, the computational burden of the algorithm can be greatly reduced by rejecting point matches that are outliers.

3.6 3D Relative Pose Estimation (Registration)

Registration is the process of calculating the rotation matrix (quaternion) and cartesian translation between two objects (see fig. 3.6). Registration of multiple point clouds and planar surfaces is an important research topic in the areas of computer vision, computer graphics, robotics, and medicine for the past several years. These registration algorithms and methods developed are used in a variety of tasks ranging from complicated surgeries to constructing accurate 3-D CAD models of priceless artifacts such as Davids Michelangelo and the Great Buddha [56, 68].

Once a set of compatible point correspondences has been found, the registration problem is solved directly through matrix based computations. To simplify the problem, the mean of both data sets is calculated and subtracted from every point in the corresponding range image so that only the rotational component needs to be calculated. In the literature [40, 41, 53, 62], several authors emphasize that there needs to be around a 40-60%

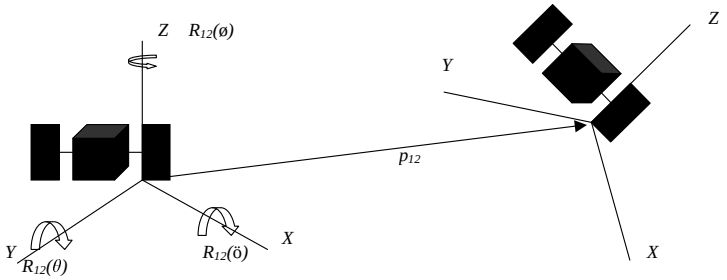

Fig. 3.6: Simulated demonstration of a spacecraft moving through 3D space with the registration parameters (rotation matrix and cartesian translation).

overlap between the point-clouds for the algorithm to converge. The two methods most commonly used in finding the rotation matrix (quaternion) in the ICP algorithm are either the Singular-Value Decomposition (SVD) or the Eigen-Value Decomposition (EVD)-based methods. The singular value decomposition (SVD) method calculates the rotation matrix by performing the SVD decomposition on the cross-covariance matrix between the point-to-point matched data sets [33, 35]. The two methods, SVD and EVD, provide the same pose estimation values but in different formats (rotation matrix and quaternion).The quaternion-based method calculates the quaternion equal to the eigenvector associated with the maximum eigenvalue of a 4 x 4 matrix constructed from the cross-covariance matrix between the point-to-point matched data sets [28, 29, 34]. In this thesis, both methods were initially studied, but the EVD method was chosen for the final pose estimation results, so knowing whether one is more accurate is important. In the literature, the author Kalman [70] shows that both the EVD and SVD can be derived from the other, and presents applications and performance for the SVD while mentioning that identical results were obtained when using the EVD method. These two methods are described in secs. 3.6.1 and 3.6.2.

3.6.1 Singular Value Decomposition (SVD) Algorithm

The first mathematical method used to calculate the registration was the SVD method. In order to do this, the two point-sets centroids μ_{R_1} and μ_{R_2} are calculated from (3.15) and (3.16) where R_1 and R_2 have size N_1 and N_2.

$$\vec{\mu}_{R_1} = \frac{1}{N_1} \sum_{i=1}^{N_1} \vec{R_1^i} \tag{3.15}$$

$$\vec{\mu}_{R_2} = \frac{1}{N_2} \sum_{i=1}^{N_2} \vec{R_2^i} \tag{3.16}$$

Then the 3 x 3 cross-covariance matrix between the two matched point-sets was calculated from the two point clouds in (3.17). Even though both point sets may have different sizes, the matched data sets contain the same number of data points. This was because when matching the first data set, which might contain more points, each point has a match

with the second point cloud even if there were duplicate matches.

$$\Sigma_{R_1R_2} = \frac{1}{N}\sum_{i=1}^{N}\left[\left(\vec{R_1^i} - \vec{\mu}_{R_1}\right)\left(\vec{R_2^i} - \vec{\mu}_{R_2}\right)^T\right] \tag{3.17}$$

As mentioned above, the SVD of the cross-covariance matrix was calculated. From this calculation, three separate matrices U, Σ, and V were calculated containing the eigenvalues and eigenvectors of the cross-covariance matrix. According to the theory of the SVD [71], every matrix A $\in \mathbb{A}^{mxn}$ can be factored as

$$A = U\Sigma V^H, \tag{3.18}$$

where U $\in \mathbb{A}^{mxm}$ and V $\in \mathbb{A}^{nxn}$ are both unitary matrices and Σ has the form,

$$\Sigma = diag(\sigma_1, \sigma_2, \cdots, \sigma_p), \tag{3.19}$$

where $p = \min(m, n)$ and the diagonal elements of Σ are called the singular values of A and are usually ordered from the largest singular value down to the smallest singular value. The matrices U and V are formed from the terms u_i and v_i which are eigenvectors of AA^T, and A^TA, respectively, as given by (3.20) and (3.21).

$$U = [\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_m] \tag{3.20}$$

$$V = [\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n] \tag{3.21}$$

After taking the singular value decomposition of the cross-covariance matrix calculated in (3.17), the pose estimation parameters $R_{R_1R_2}(\psi, \theta, \phi)$ and $p_{R_1R_2}(x, y, z)$ can be calculated. The rotation matrix is calculated by multiplying the matrices U, V, and S (identity matrix) in the specific order shown in (3.22). This derivation and order are described completely in Appendix A outlining the need for the identity matrix.

$$R_{R_1R_2}(\psi, \theta, \phi) = VSU^T \tag{3.22}$$

Because of the geometry of certain 3D objects, such as perfect spheres or the case of a triangle centered along its midpoint, the SVD algorithm will fail to work. However, Umeyama [29] solved the problem of the degenerate case. In the degenerate case, the determinate of the cross-covariance matrix goes to -1; and in order to correct for this degeneracy, the last term of the identity matrix used in the rotation matrix calculation was set equal to -1. To ensure that the degenerate case would not happen in software, the registration calculation used a matrix S separated into the following two cases as follows:

$$S = \begin{cases} I & \text{if } det(U)det(V) = 1 \\ diag(1, 1, \cdots, 1, -1) & \text{if } det(U)det(V) = -1. \end{cases} \tag{3.23}$$

To calculate the translation vector component of the pose, it is necessary to use both the rotation matrix calculated in (3.22) and the mean/centroid values from (3.14) and (3.15). The ICP algorithm described throughout the thesis outlines an algorithm that provides a solution that performs a least squares optimization for only the rotation matrix component of the pose. Because the two point-clouds may only overlap by 40-60%, the centroid values calculated in (3.15) and (3.16) will not represent the same physical point on the object. Therefore, the translation vector being calculated in this algorithm will represent the translation of the object in addition to algorithmic error caused by these two centroid values not representing the same physical point. Since, the algorithm does not optimize for the translation vector independently, it will be dependent entirely on the centroid values and the rotation matrix and can be calculated as follows:

$$p_{R_1R_2}(x, y, z) = \mu_{R_2} - R(\psi, \theta, \phi)\mu_{R_1}. \tag{3.24}$$

Described below is an entire step by step method for using the SVD algorithm.

3.6.2 Eigen-Value Decomposition (EVD) Algorithm

In the EVD algorithm described by Horn and Walker [33–35], the covariance matrix is calculated in (3.16). However, several different vectors and matrices are used to calculate

Algorithm 3.1 Singular Value Decomposition (SVD) Algorithm

Input:

Point set P with N_p,
Point set X with N_x,

Initialization:

Let m = number of spatial dimensions
Calculate μ_p and μ_x /* Mean Vectors of P and X data sets*/
Calculate Σ_{px} /* Covariance matrix between P and X data sets */

Begin

Calculate the singular value decomposition of $\Sigma_{px}(U, D, V)$
and use the identity matrix for S.

If rank $(\Sigma_{px}) > m - 1$
Calculate the registration motion parameters as follows:

$R_{R_1R_2}(\psi, \theta, \phi) = VSU^T$ /* 3x3 Registration Rotation Matrix */
$p_{R_1R_2}(x, y, z) = \mu_x - R(\psi, \theta, \phi)\mu_p$ /* 3x1 Translation Matrix */
Else rank $(\Sigma_{px}) \leq m - 1$

$$S = \begin{cases} I & \text{if } det(U)det(V) = 1 \\ diag(1, 1, \cdots, 1, -1) & \text{if } det(U)det(V) = -1 \end{cases}$$

Calculate the registration parameters: $R_{R_1R_2}(\psi, \theta, \phi)$
and $p_{R_1R_2}(x, y, z)$ using the adjusted identity matrix.
End
End

Output:

$R_{R_1R_2}(\psi, \theta, \phi)$ and $p_{R_1R_2}(x, y, z)$ /* Final Pose Estimation Registration Parameters */

the rotation matrix. The first term A_{ij} is an anti-symmetric matrix formed from the cross-covariance matrix (where i and j are indices for elements in the matrix A being constructed), and then the elements of A_{ij} are used to construct the column vector Δ. Finally, this column vector Δ and the cross-covariance matrix are used to construct a symmetric matrix $Q(\Sigma_{R_1R_2})$.

$$A_{ij} = (\Sigma_{R_1R_2} - \Sigma^T_{R_1R_2})_{ij} \tag{3.25}$$

$$\Delta = [A_{23}A_{31}A_{12}]^T \tag{3.26}$$

$$Q(\Sigma_{R_1R_2}) = \begin{bmatrix} trace(\Sigma_{R_1R_2}) & \Delta^T \\ \Delta & \Sigma_{R_1R_2} + \Sigma^T_{R_1R_2} - trace(\Sigma_{R_1R_2})I \end{bmatrix} \tag{3.27}$$

From the matrix $Q(\Sigma_{R_1R_2})$ above, the eigen-values (λ_i) and eigenvectors (x_i) of $Q(\Sigma_{R_1R_2})$ are calculated, then $\overrightarrow{q}_r$ was selected as the unit eigenvector (x_i) corresponding to the maximum eigenvalue (λ_i). This quaternion is used to calculate $R_{R_1R_2}(\psi, \theta, \phi)$ from the quaternion $\vec{q}_r$ in (3.28) in (3.29). The translation vector $p_{R_1R_2}$ from (3.24) was calculated using the rotation matrix found in (3.29) with the same problems explained above (see Appendix B).

$$\vec{q}_r = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix} \tag{3.28}$$

$$R_{R_1R_2}(\vec{q}_r) = \begin{bmatrix} q_0^2 + q_1^2 - q_3^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 + q_2^2 - q_1^2 - q_3^2 & 2(q_2q_3 + q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 + q_3^2 - q_1^2 - q_2^2 \end{bmatrix} \tag{3.29}$$

Described below is an entire step by step method for using the EVD algorithm.

Algorithm 3.2 Eigen-Value Decomposition (EVD) Algorithm

Input:

Point set P with N_p,
Point set X with N_x,

Initialization:

Let m = number of spatial dimensions
Calculate μ_p and μ_x /* Mean Vectors of P and X data sets*/
Calculate Σ_{px} /* Covariance matrix between P and X data sets */

Begin

Calculate the anti-symmetric matrix $A_{ij} = (\Sigma_{px} - \Sigma_{px}^T)_{ij}$
Form the vector $\Delta = [A_{23} A_{31} A_{12}]^T$
Calculate the matrix $Q(\Sigma_{px})$:

$$Q(\Sigma_{px}) = \begin{bmatrix} trace(\Sigma_{px}) & \Delta^T \\ \Delta & \Sigma_{px} + \Sigma_{px}^T - trace(\Sigma_{px}) I \end{bmatrix}$$

Calculate the eigenvalues (λ_i) and eigenvectors (x_i) of $Q(\Sigma_{px})$
Select $\overrightarrow{q}_r$ as the unit eigenvector (x_i) corresponding to the maximum eigenvalue (λ_i)
Calculate $R_{R_1 R_2}(\psi, \theta, \phi)$ from $\vec{q}_r$
Calculate $p_{R_1 R_2} x, y, z = \vec{\mu}_x - R_{R_1 R_2}(\psi, \theta, \phi) \vec{\mu}_p$

End

Output:

$R_{R_1 R_2}(\psi, \theta, \phi)$ and $p_{R_1 R_2} x, y, z$ /* Final Pose Estimation Registration Parameters */

3.7 Error Metric Minimization

3.7.1 Point-to-Point Error Metric

The final step in the ICP algorithm is to compute an error metric that the algorithm uses as a measure to the success of the calculated registration at each iteration. For most applications, a mean square error (MSE) or sum of the squared distance between corresponding points after applying the calculated registration is used as represented in (3.30).

$$MSE = \frac{1}{N}\sum_{i=1}^{N} \|\vec{x}_i - R_{R_1R_2}(\psi,\theta,\phi)\vec{y}_i - p_{R_1R_2}x,y,z\|^2 \tag{3.30}$$

3.7.2 Point-to-Plane Error Metric

Similarly, the final step in the ICP algorithm (point-plane variation) is to compute an error metric that the algorithm uses as a measure of the success of the calculated registration at each iteration. For most applications, a mean square error (MSE) or sum of the squared distance between corresponding point and mesh/plane after applying the calculated registration was used where $\vec{n}$ was the surface normal as represented in (3.31).

$$MSE = \frac{1}{N}\sum_{i=1}^{N} \left\|\vec{n}\cdot({}^{I}R^{i}_{point} - R_{R_1R_2}(\psi,\theta,\phi)B_I - p_{R_1R_2}x,y,z)\right|^2 \tag{3.31}$$

The overall ICP algorithm has been condensed and is shown on the next page (see Appendix C).

Algorithm 3.3 Iterative Closest Point (ICP) Algorithm

Input:

Point set R_1 with N_p,
Point set R_2 with N_x,

Initialization:

Let k= 0
Let $\tau = 0.001$ /*Error Tolerance*/
Let $MSE_k = 1 * 10^8$ /*Initial Mean-squared Error*/
Define Normalized Initial Rotational and Translational States
Calculate the cross-covariance matrix from (3.17)

Begin

For

Let $R_{1,k} = R_{init}(\psi, \theta, \phi)R_1 + p_{init}$
While $MSE_{k+1} - MSE_k \geq \tau$

- Calculate correspondences for point set $R_{2,k}$ from $(R_{1,k}, R_{2,k}$ from (3.5) (point-point) or (3.6-3.12) (point-plane)
- Compute the registration parameters: $R_{12,k}(\psi, \theta, \phi), p_{12,k}$ between $R_{1,k}$ and $R_{2,k}$ using either (3.22-3.23) or (3.24-3.28,3.23)
- Apply the registration: $R_{1,k+1} = R_{12,k}(\psi, \theta, \phi)R_{1,k} + p_{12,k}$
- Calculate MSE_{k+1} and let $k = k + 1$

End

End

Output:

$R_{final}(\psi, \theta, \phi)$ and $p_{final}(x, y, z)$
/* Final Pose Estimation Registration Parameters */

Chapter 4

ICP Implementation and Simulation Results

4.1 Simulated Experimental Data Collection Procedure

This project employed the direct use of a 6 DOF pose estimation algorithm (i.e., ICP algorithm), which uses sequentially gathered range images, also known as point clouds, to calculate 6 DOF relative pose and orientation. To study the robustness (i.e., the characteristic where the output or response is insensitive to the variation of the inputs) of the ICP algorithm and its future implementation into real time hardware applications, range images, or point clouds, were generated by simulation only. The method used to generate point cloud models was through a simulation program developed at the Center for Advanced Imaging Lidar (CAIL) at Utah State University. This software package, LadarSIM, incorporates performance-related parameters associated with the laser, detector, signal processing, scanner dynamics, platform dynamics, navigation errors, and scene properties to provide general system analysis, error source modeling, and 3-D points clouds [72]. The data gathered from this simulation program is gathered simulated in manner where all points are gathered all at once similar to a flash ladar technology. The only information that was used from these images was range information (x,y,z information) for every single point in each point-cloud image. There was no INS/GPS data on the target, and the EVD least squares estimation algorithm is used on the point-clouds. No feature extraction methods are used to help obtain point matches that are necessary for the EVD algorithm. Finally, all the point-cloud images are for a simple solid model of a spacecraft, as shown in fig. 4.1. The simulated point cloud were generated assuming the range between the ladar and the target was approximately 1 km. The angular resolution between shots was 100 rad with an RMS range error of 0 cm, 2 cm or 14 cm, based on realistic ladar transceiver error models.

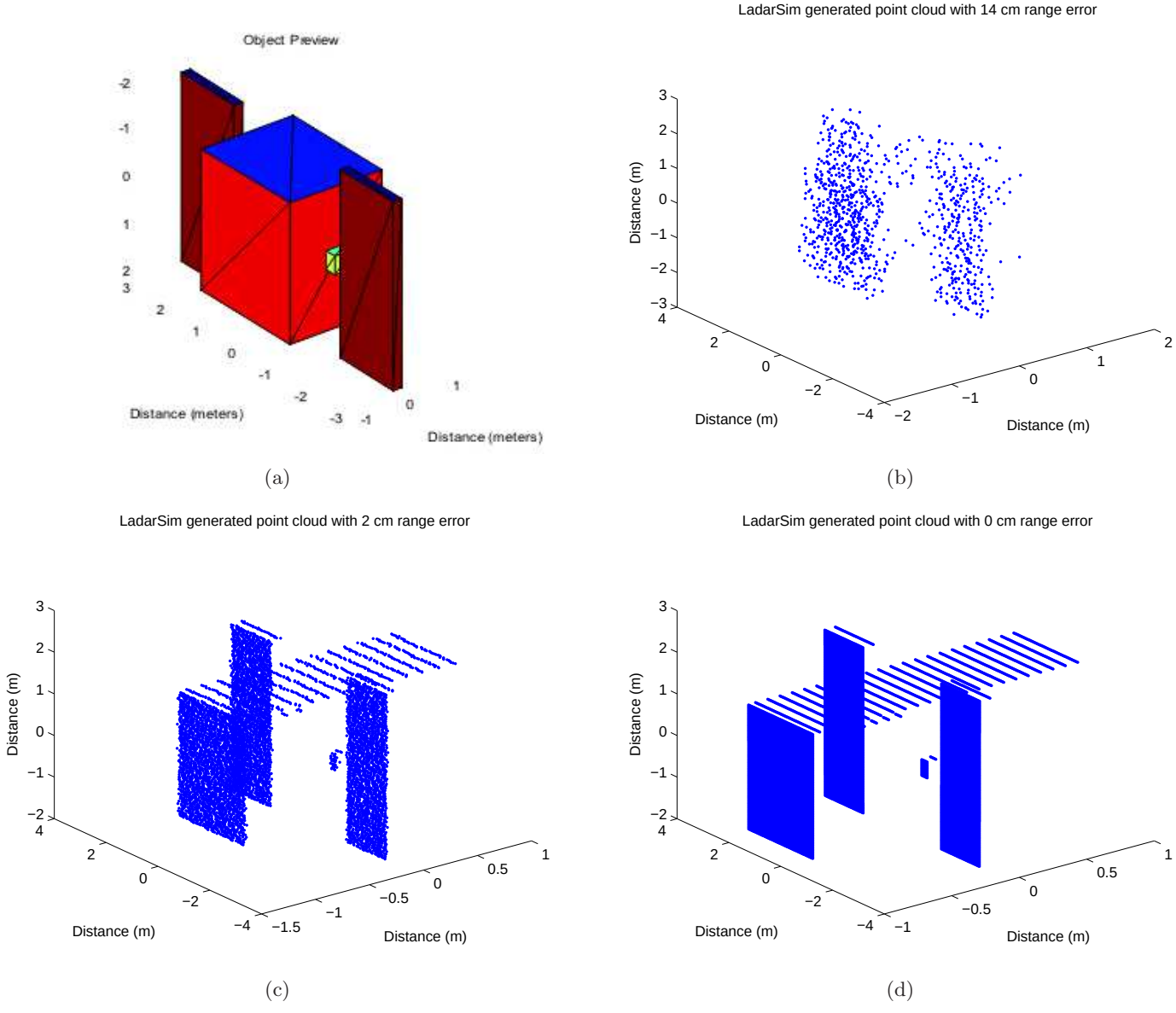

Fig. 4.1: LADARSIM model of a (a) 3D solid model, (b) point-cloud with 14 cm range error, (c) point-cloud with 2 cm range error, and (d) point-cloud with zero cm range error noise.

4.2 Point-to-Point ICP Algorithm Results

The purpose of this project was to determine a suitable registration algorithm that would accurately, quickly, and robustly find the position and orientation from a pair of range images. For this thesis project, the point-to-point ICP algorithm simulations were performed on simulated data that had 0 cm range error and 14 cm range error taken at a distance of approximately 1 km. Results were obtained from the simulated range images of a spacecraft going through varying angular rotations such as: 1°, 5°, 10°, 20°, 40°, and 60°, about one or three orthogonal rotational axes at the same time. The point-to-point ICP algorithm was then used to calculate the registration parameters using only 10% of the range data points (i.e., anywhere between 50 points and 2500 points) to speed up computation times. To test the sensitivity of the ICP algorithm, a Monte Carlo-based test strategy was performed by running 50 individual runs of the ICP point-to-point algorithm. For each of these 50 separate runs, two different point clouds were chosen. The second point cloud was rotated about a single-axis or about all three axes. The ICP algorithm was then used to compare these two individual point-clouds. Previously, it was stated that the translation vector when estimated is a combination of the actual translation between the two point-clouds, plus the additional error caused as a result of centroids being located at different physical positions. To estimate the error due to this misalignment, the translation applied to the second point cloud was zero along all three axes. Since there is no translation involved, the estimated translation vector represents the error directly related to the centroid misalignment. After all these simulations were set up many values were analyzed. The measured values included euler angular errors, eigen-axis angular errors, algorithm times, MSE, translational errors, etc. The data from these simulations was processed to find values such as the mean and standard deviation values over the 50 trials for several of the important pose parameters above.

Of the parameters obtained from simulation, eigen-axis angular errors, mean-squared error (MSE), and translational errors were determined to be most relevant to rendezvous and dock capabilities. Eigen-axis error is the error between the true rotation angle used to

move the second point cloud and the angle calculated by the ICP algorithm. The next value is the translational error, which is the error between the true translation that was used when moving the second point cloud and the translation calculated using the ICP algorithm. Since no translation is used for these simulations, the error related only to centroid misalignment. The final value was the MSE, which is the error between the individual points in the two point clouds, meaning each point in the first point cloud ends up matched to a point in the second point cloud after the final iteration and the sum of the distances between these points gives us the MSE. Each of these values was calculated for 50 separate trials, for each of the six rotations about one or three rotation axes. In each of the tables (Table 4.1 to Table 4.6), for both single-axis and three-axis rotations, the mean and standard deviation values were calculated from the 50 experimental trials for each of the six varying rotation angles using a point-to-point error metric.

Table 4.1: Eigen-axis angular error means and standard deviations values between actual rotation and point-to-point ICP algorithm for 14 cm RMS range error point cloud data.

True Rotation Angle(∘)	Z-Axis Rotation		ZYX-Axis Rotation	
	Mean Error (∘)	Standard Deviation (∘)	Mean Error (∘)	Standard Deviation(∘)
1	2.05	1.30	2.22	2.21
5	2.22	2.22	1.85	1.36
10	1.87	1.35	2.22	1.51
20	1.85	1.09	2.54	2.23
40	2.31	2.29	2.44	2.26
60	1.81	1.36	1.89	1.37

Table 4.2: Cartesian translation error means and standard deviation values between actual rotation and point-to-point ICP algorithm for 14 cm RMS range error point cloud data.

True Rotation Angle(∘)	Z-Axis Rotation		ZYX-Axis Rotation	
	Translation Error(cm)	Standard Deviation (cm)	Translation Error(cm)	Standard Deviation(cm)
1	18.1	11.41	15.6	8.19
5	15.5	7.89	15.6	7.63
10	15.5	8.84	16.6	8.01
20	17.6	8.81	14.1	8.39
40	15.3	9.01	14.9	8.64
60	15.0	7.09	16.2	7.44

Table 4.3: Mean-squared alignment error means and standard deviations between registered point sets for point-to-point ICP algorithm for 14 cm RMS range error point cloud data.

True Rotation Angle(∘)	Z-Axis Rotation		ZYX-Axis Rotation	
	MSE (cm)	Standard Deviation (cm)	MSE (cm)	Standard Deviation(cm)
1	12.3	6.82	11.6	5.67
5	11.5	5.68	11.0	5.48
10	11.0	5.49	11.9	5.41
20	12.5	6.27	11.5	5.63
40	11.5	5.65	11.5	5.65
60	11.0	5.50	11.0	5.48

Table 4.4: Eigen-axis angular error means and standard deviations values between Actual rotation and point-to-point ICP algorithm for 0 cm RMS range error point cloud data.

True Rotation Angle(∘)	Z-Axis Rotation		ZYX-Axis Rotation	
	Mean Error (∘)	Standard Deviation (∘)	Mean Error (∘)	Standard Deviation(∘)
1	1.53	1.19	1.72	1.18
5	1.51	1.09	1.71	1.30
10	1.28	0.94	1.71	1.51
20	1.83	1.41	1.79	1.21
40	1.64	1.29	1.79	1.14
60	1.74	1.23	1.46	1.57

Table 4.5: Cartesian translation error means and standard deviation values between actual rotation and point-to-point ICP algorithm for 0 cm RMS range error point cloud data.

True Rotation Angle(∘)	Z-Axis Rotation		ZYX-Axis Rotation	
	Translation Error(cm)	Standard Deviation (cm)	Translation Error(cm)	Standard Deviation(cm)
1	15.9	9.14	14.8	8.34
5	14.1	7.47	15.2	8.31
10	15.8	9.60	14.8	8.17
20	15.2	6.75	13.9	6.88
40	15.3	8.34	15.9	7.83
60	14.8	7.34	14.8	7.58

Table 4.6: Mean-squared alignment error means and standard deviations between registered point sets for point-to-point ICP algorithm for 0 cm RMS range error point cloud data.

True Rotation Angle(∘)	Z-Axis Rotation		ZYX-Axis Rotation	
	MSE (cm)	Standard Deviation (cm)	MSE (cm)	Standard Deviation(cm)
1	9.8	1.89	9.6	1.68
5	9.8	1.45	9.6	1.76
10	9.7	1.35	9.7	1.88
20	9.8	1.84	9.7	1.70
40	9.8	2.02	9.6	1.26
60	9.8	1.53	9.8	1.90

All the previous data was processed and condensed down further into four smaller tables which were more easily discernable (Table 4.7 to Table 4.10) for the rotational and translational error values. For single-axis rotation, angular errors fell between 1.28° and 1.83° for noiseless and 1.81° to 2.31° for noisy data; whereas for three-axis rotation, the angular error fell between 1.46° and 1.79° for noiseless and 1.85° to 2.54° for noisy data. Similarly, the translational errors fell between 14.1 to 15.9 cm for noiseless and 15.0 to 18.1 cm for noisy data; whereas, for three-axis rotation the translational errors were 13.9 to 15.2 cm for noiseless and 14.1 to 16.6 cm for noisy data.

Rotational results suggest that for all rotations the estimated rotation was around 2°, which although quite accurate is too large for RVD operations that require precise pose estimates. The translational results were in the 10-20 cm range, and these were compared to a satellite that was 6 m x 4 m x 2 m. The translational error calculated was a root-sum-squared (RSS) error where each term in the x, y, and z axis were squared, summed together, and the square root taken. The translation error then was between 3-10% of the actual size of the spacecraft depending on the dimension to which the comparison is made. Because the centroids are not the same in the two point-clouds, the relative error due to this misalignment has been bounded at 3-10% the object size and can now be modeled. Another question posed as a result of this analysis is whether or not that 3-10% relative error would remain constant as the object decreased or increased in size. Also, as long as there is a 40-60% overlap among the point clouds and for rotations up to 60° all the calculated values

remain constant. The MSE values were used as the stopping criterion for the ICP algorithm. The MSE value represented a RSS values between the points in the first point cloud and the points in the second point cloud. A metaphor for this description would be taking two pieces of tin foil and crinkling them up separately, and folding them up into the shape of some object. By crinkling up the tin foil separately, there are different places where the tin foil peaks like the points of a point-cloud. Then, say you match up the two pieces of tin foil and perform point-to-point distance calculations between matched points on the tin-foil, sum all those distances together, take the square root, and this would essentially be the MSE value calculated by the ICP algorithm. For the point-to-point simulations, the MSE were between 9-12 cm being about 3-5% of the spacecraft's relative size, which directly relates to how well the point clouds over lie one another. Couple all the values from rotational error, translational error, and MSE together the point-to-point ICP variant was moderately accurate when it came to estimating pose, but was not accurate enough for rendezvous and dock proximity operations; however, the large range errors did not significantly increase these errors using the point-point method.

A special simulation was performed that took two point-clouds and sub-sampled the data between 0%-90% of the total number of points. Then, a single simulation was performed for a 20° rotation about the z-axis. This experiment showed that as the number of points increased from 0%-90%, angular error only saw a 1 decrease for a 100% increase in the algorithm time; however, this data was obtained using only one trial which might explain the trough at 50% points used (see fig. 4.2). To more accurately understand this relationship, it is recommended that at least 50 trials be used to smooth out the angular error data. The important take home message from this experiment though is that number of points does drastically affect computational time; whereas, rotational error does not increase as the number of points used decreases. Also, one may conclude that if a range image is gathered with a small resolution with enough overlap between the two-point sets, pose estimation can be accurately done with the ICP algorithm.

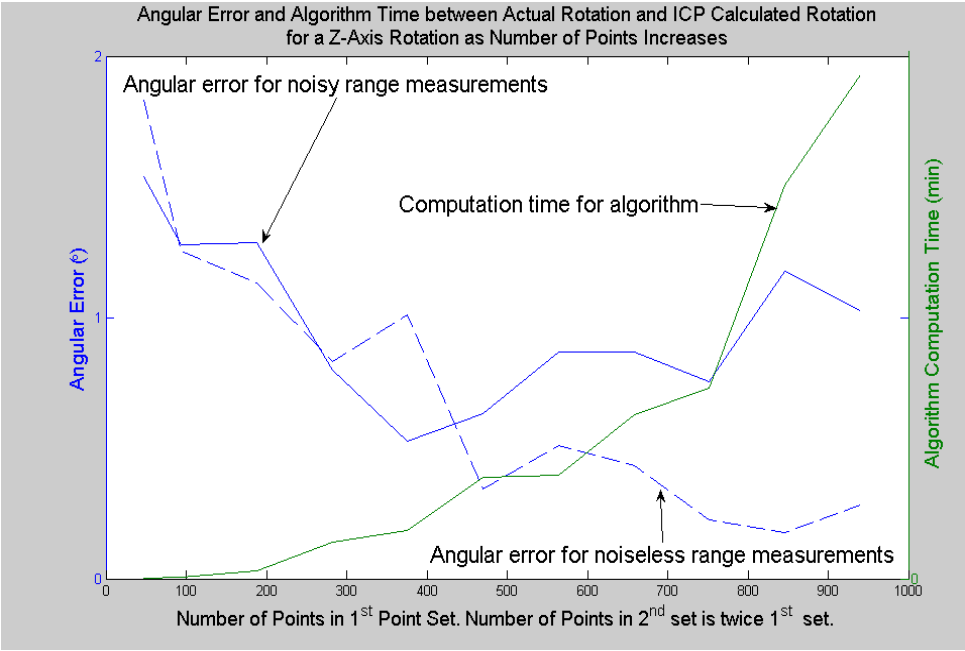

Fig. 4.2: Angular error and algorithm time between actual rotation and ICP calculated rotation for a z-axis rotation as number of points increased.

4.3 Point-to-Plane ICP Algorithm Results

As mentioned above, the purpose of this project was to determine a suitable registration algorithm that would accurately, quickly, and robustly find the position and orientation from a pair of range images. The point-plane variation of the ICP algorithm described by Rusinkiewicz [62] was evaluated within this thesis to reduce the mean-squared error during each iteration and speed up computations. In his dissertation, it was shown that for each iteration, the point-plane variant MSE converged much faster than the point-to-point variant that was used before. However, the dissertation never discussed how accurate the algorithm could calculate the registration parameters and how computational times would suffer without an initial registration parameter given. For this thesis, the point-to-plane ICP algorithm simulations were performed on simulated data that had 0 cm range error, 2 cm range error, and 14 cm range error taken at a distance of approximately 1 km as before. Simulation results were obtained from the simulated range images of a spacecraft going through varying angular rotations such as: 1°, 5°, 10°, 20°, 40°, and 60° about one

or three orthogonal rotational axes at the same time. Thus, the experiment selects two separate 3D point clouds randomly from a set of range images that were collected with LadarSIM. At this point, the second point cloud was rotated by one of the angles listed earlier either about one axis, or about all three axes using the same rotation angle about all the axes. The point-to-plane ICP algorithm was then used to calculate the registration for comparison using only 1% of the possible data points ranging between 1000 points and 3000 points. For this set of simulations, instead of using only 0 cm and 14 cm noise range error data, a third set of data was collected at 2 cm noise range error all using the point-to-plane error metric (see Tables 4.7-4.15).

Table 4.7: Eigen-axis angular error means and standard deviations values between actual rotation and point-to-plane ICP algorithm for 0 cm RMS range error point cloud data.

True Rotation Angle(∘)	Z-Axis Rotation		ZYX-Axis Rotation	
	Mean Error (∘)	Standard Deviation (∘)	Mean Error (∘)	Standard Deviation(∘)
1	0.031	0.017	0.052	0.033
5	0.043	0.024	0.044	0.025
10	0.047	0.033	0.054	0.022
20	0.057	0.055	0.062	0.029
40	0.074	0.045	0.075	0.024
60	0.079	0.050	0.220	0.270

Table 4.8: Cartesian translation error means and standard deviation values between actual rotation and point-to-plane ICP algorithm for 0 cm RMS range error point cloud data.

True Rotation Angle(∘)	Z-Axis Rotation		ZYX-Axis Rotation	
	Translation Error(cm)	Standard Deviation (cm)	Translation Error(cm)	Standard Deviation(cm)
1	2.1	0.72	2.1	0.67
5	2.1	0.72	2.1	0.84
10	2.1	0.72	1.8	0.70
20	2.0	0.60	1.9	0.83
40	2.1	0.61	3.0	1.81
60	2.0	0.68	7.8	8.05

Table 4.9: Mean-squared alignment error means and standard deviations between registered point sets for point-to-plane ICP algorithm for 0 cm RMS range error point cloud data.

True Rotation Angle(∘)	Z-Axis Rotation		ZYX-Axis Rotation	
	MSE (cm)	Standard Deviation (cm)	MSE (cm)	Standard Deviation(cm)
1	0.33	0.22	0.45	0.27
5	0.34	0.20	0.38	0.23
10	0.35	0.18	0.47	0.23
20	0.47	0.25	0.47	0.25
40	0.48	0.24	0.79	0.35
60	0.45	0.21	1.37	2.25

Table 4.10: Eigen-axis angular error means and standard deviations values between actual rotation and point-to-plane ICP algorithm for 2 cm RMS range error point cloud data.

True Rotation Angle(∘)	Z-Axis Rotation		ZYX-Axis Rotation	
	Mean Error (∘)	Standard Deviation (∘)	Mean Error (∘)	Standard Deviation(∘)
1	0.053	0.024	0.070	0.053
5	0.051	0.026	0.065	0.053
10	0.056	0.034	0.095	0.062
20	0.055	0.038	0.186	0.120
40	0.079	0.056	0.490	0.697
60	0.129	0.084	0.654	0.715

Table 4.11: Cartesian translation error means and standard deviation values between actual rotation and point-to-plane ICP algorithm for 2 cm RMS range error point cloud data.

True Rotation Angle(∘)	Z-Axis Rotation		ZYX-Axis Rotation	
	Translation Error(cm)	Standard Deviation (cm)	Translation Error(cm)	Standard Deviation(cm)
1	2.1	0.72	2.1	0.67
5	2.1	0.72	2.1	0.84
10	2.1	0.72	1.8	0.70
20	2.0	0.60	1.9	0.83
40	2.1	0.61	3.0	1.81
60	2.0	0.68	7.8	8.05

Table 4.12: Mean-squared alignment error means and standard deviations between registered point sets for point-to-plane ICP algorithm for 2 cm RMS range error point cloud data.

True Rotation Angle(∘)	Z-Axis Rotation		ZYX-Axis Rotation	
	MSE (cm)	Standard Deviation (cm)	MSE (cm)	Standard Deviation(cm)
1	0.33	0.22	0.45	0.27
5	0.34	0.20	0.38	0.23
10	0.35	0.18	0.47	0.23
20	0.47	0.25	0.47	0.25
40	0.48	0.24	0.79	0.35
60	0.45	0.21	1.37	2.25

Table 4.13: Eigen-axis angular error means and standard deviations values between actual rotation and point-to-plane ICP algorithm for 14 cm RMS range error point cloud data.

True Rotation Angle(∘)	Z-Axis Rotation		ZYX-Axis Rotation	
	Mean Error (∘)	Standard Deviation (∘)	Mean Error (∘)	Standard Deviation(∘)
1	0.39	0.26	0.35	2.21
5	0.44	0.25	0.37	1.36
10	0.41	0.28	0.44	1.51
20	0.38	0.26	0.39	2.23
40	0.43	0.25	0.42	2.26
60	0.49	0.30	0.89	1.37

Table 4.14: Cartesian translation error means and standard deviation values between actual rotation and point-to-plane ICP algorithm for 14 cm RMS range error point cloud data.

True Rotation Angle(∘)	Z-Axis Rotation		ZYX-Axis Rotation	
	Translation Error(cm)	Standard Deviation (cm)	Translation Error(cm)	Standard Deviation(cm)
1	2.23	1.08	2.25	1.12
5	2.29	1.31	2.28	1.47
10	2.78	1.80	2.79	1.78
20	2.73	1.49	2.83	2.87
40	2.30	1.74	5.07	7.34
60	2.84	2.20	7.05	9.63

Table 4.15: Mean-squared alignment error means and standard deviations between registered point sets for point-to-plane ICP algorithm for 14 cm RMS range error point cloud data.

True Rotation Angle(∘)	Z-Axis Rotation		ZYX-Axis Rotation	
	MSE (cm)	Standard Deviation (cm)	MSE (cm)	Standard Deviation(cm)
1	7.1	2.20	7.1	2.25
5	7.1	2.19	7.2	2.24
10	6.9	2.17	7.1	2.26
20	7.3	2.22	7.2	2.21
40	7.2	2.24	7.2	2.18
60	6.9	2.19	7.4	2.23

The data from Tables 4.7 to 4.15 was processed as before in order to determine how well the algorithm performed. For the single-axis rotation and three-axis rotation, angular error fell between 0.031° and 0.890° for 0, 2, and 14 cm RMS range error point-clouds. Similarly, the translational errors fell between 2.00 to 8.05 cm for 0,2, and 14 cm RMS range error data sets for the single- and three-axis rotational cases.

Here for the point-to-plane experiments the rotational error have values that are all nearly below 1°, which are much more accurate than the best point-to-point rotational errors seen. Notice that the only rotation angle that had larger errors then most other cases was the 60° case. This happens because the ICP algorithm requires approximately 60% overlap between point clouds, and when the point clouds are extremely similar, like those obtained through simulation, that point ends up near a rotation angle of 60°. Translational errors decreased to the point where they are 1-5% the size of the satellite implying that the point-plane variation of the ICP algorithm is able to deal with the centroid misalignment better. The MSE error values for the point-to-plane variation experiments were down around 0.50 cm, which are an order of magnitude lower than the previous point-point experiments. This can be best explained by the previous tin-foil explanation. Instead of using the peak points on the second piece of tin-foil, meshes or planes can be constructed from those peaks using the peaks as vertices. Then the points in the first piece of tin-foil match to meshes instead of points that might contain significant amounts of noise. Therefore, the MSE value can be reduced ten-fold by being able to match a set of points over a meshed object and not a cloud

of points per se. Taking a look at Tables 4.16-4.19, one can conclude that the point-to-plane ICP variant was extremely accurate when it came to pose estimation even when 2 or 14 cm range error was present.

Table 4.16: Single-axis rotational errors for point-to-point ICP Algorithm.

Range Error (RMS)	No Noise	2 cm	14 cm
single-axis Rotation	(◦)	(◦)	(◦)
Maximum Mean Angular Error	0.079	0.129	0.494
Minimum Mean Angular Error	0.031	0.024	0.383
Average Mean Angular Error	0.055	0.070	0.423
Average 1σ - Standard Deviation	0.037	0.044	0.267

Table 4.17: Single-axis translational errors for point-to-plane ICP Algorithm.

Range Error (RMS)	No Noise	2 cm	14 cm
single-axis Rotation	(cm)	(cm)	(cm)
Maximum Mean Translational Error	2.10	2.10	2.84
Minimum Mean Translational Error	2.00	2.00	2.23
Average Mean Translational Error	2.09	2.09	2.53
Average 1σ - Standard Deviation	0.67	0.67	1.60

Table 4.18: Three-axis rotational errors for point-to-plane ICP Algorithm.

Range Error (RMS)	No Noise	2 cm	14 cm
Three-axis Rotation	(◦)	(◦)	(◦)
Maximum Mean Angular Error	0.220	0.654	0.890
Minimum Mean Angular Error	0.044	0.065	0.349
Average Mean Angular Error	0.054	0.260	0.477
Average 1σ - Standard Deviation	0.067	0.283	1.824

Table 4.19: Three-axis translational errors for point-to-plane ICP Algorithm.

Range Error (RMS)	No Noise	2 cm	14 cm
Three-Axis Rotation	(cm)	(cm)	(cm)
Maximum Mean Translational Error	7.76	7.76	8.05
Minimum Mean Translational Error	1.76	1.76	2.25
Average Mean Translational Error	3.09	3.09	3.71
Average σ - Standard Deviation	2.15	2.15	4.01

4.4 Results Summary

Finally, after running all the experimental cases through the version of the ICP algorithm written in MATLAB based on all the previous literature, the ICP version that was coded using a previous students work [73] and the Rusinkiewicz ICP code from his dissertation work. Using the newly formed MATLAB simulation, several different figures were put together showing two simulated cases and how the pose was correctly calculated for the two point clouds. In each case, the figure shows the two point clouds in their initial orientation and each in their final orientation after running the ICP algorithm through several iterations. Although the data gathered from the experiments showed accurate pose estimation, being able to see that the point-plane variation of the algorithm works helps drive home the point at least one more time (See fig. 4.3).

The final analysis that was performed for this thesis was to summarize and compare the point-to-point variant and the point-to-plane variant. Table 4.20 summarizes all the data from tables 4.1-4.19. In the end, the point-to plane variant of the ICP algorithm is an order of magnitude better at estimating both rotational and translational pose parameters. Whether the rigid body motion was about one axis or three axes, as well as the rotational magnitude, both methods show considerable robustness to estimating pose parameters. Finally, one set of results that was not analyzed in depth was the considerable problem with having to brute force through the initialization process. When Monte Carlo simulations were being performed, having to do 50 experiments with 312 initial guess, plus the number of iterations per initial guess begin to add up and required days to perform. On orbit, this type of algorithm computation time would be diastrous and completely unreasonable for real-time RVD. Overall, the ICP algorithm has great potential to be able as a RVD pose estimation algorithm given only point-clouds images and nothing else, but will require further investigation regarding initialization methods.

Table 4.20: Final comparison of point-point and point-plane ICP algorithm for single-axis and three-axis cases.

	Point-Plane	Point-Point
Single-axis Rotation		
0 cm (Rotational Error)	0.055° ± 0.037°	1.59° ± 1.19°
0 cm (Translation Error)	2.09 cm ± 0.67 cm	15.2 cm ± 8.1 cm
14 cm (Rotational Error)	0.423° ± 0.267°	2.02° ± 1.60°
14 cm (Translational Error)	2.53 cm ± 1.60 cm	16.1 cm ± 8.8 cm
Three-axis Rotation		
0 cm (Rotational Error)	0.054° ± 0.067°	1.69° ± 1.32°
0 cm (Translation Error)	3.09 cm ± 2.15 cm	14.9 cm ± 7.9 cm
14 cm (Rotational Error)	0.477° ± 1.82°	2.21° ± 1.82°
14 cm (Translational Error)	3.71 cm ± 4.01 cm	15.5 cm ± 8.1 cm

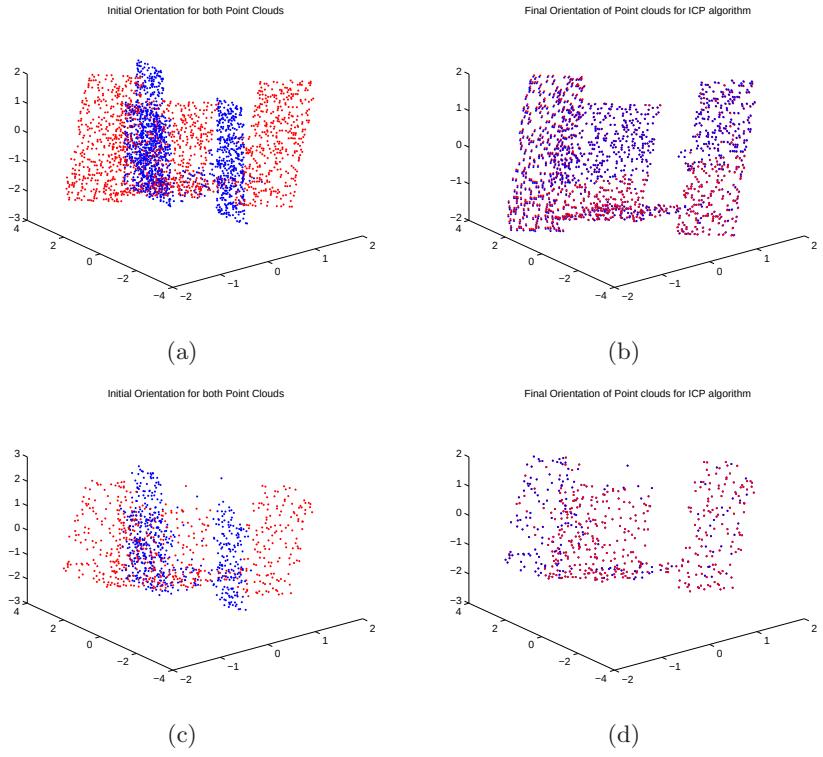

Fig. 4.3: Two separate point clouds prepared for ICP Algorithm (a) initial orientation 2 cm range error, (b) final orientation 2 cm range error range error, (c) initial orientation 14 cm range error, and (d) final orientation 14 cm range error.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

In this thesis, the three goals that were expected to be completed were:

- determine the accuracy the ICP algorithm when determining relative position and attitude of a spacecraft from ladar data, and
- determination of requirements needed to build a real-time relative navigation sensor (future work).

Experimental testing was been done using ladar-based navigation methods to locate the position and orientation of a nearby spacecraft in simulation. Position and attitude were determined by comparing the sequentially acquired point-clouds generated from solid model of the spacecraft with each other. The generic point-cloud matching algorithm known as the Iterative Closest Point (ICP) method resulted in modest accuracy for the point-to-point variant, but exhibited much better accuracy for the point-to-plane variation (See Table 4.20).

The nominal requirements for most RVD systems and projects listed in the literature mentioned are less than 1° in orientation and less than 1 m in translational error for autonomous RVD. Results show that the MSE, angular error, mean, and standard deviations for position and orientation registration between the ranges of $1 - 60^\circ$ did not vary ($< 2.5^\circ$ - point-to-point and $< 0.5^\circ$ - point-to-plane). This implied that pose accuracy of the ICP algorithm was independent of orientation for a symmetric spacecraft. The principal algorithmic results to date have shown a major success in implementing the point-plane and point-point variants of the ICP algorithm with single iteration computation times less than 1 sec comparable to iteration times shown by Rusinkiewicz in his disseration [62], as well

as other papers in the literature, but still with the problem of brute force initialization in this thesis.

Ladar imaging systems can be used as relative 3D navigation sensors for robotic spacecraft. These sensors provide extremely accurate, lighting independent relative position maps of their targets. They also have a major role in spacecraft identification for situational awareness missions, as demonstrated by the current AFRL XSS-11 mission. However, this study has pointed out that a more efficient processing algorithm would be needed for real-time applications. Flying spot ladar systems with tight scan range control of the type flown on missions like XSS-11 [23] are expensive and difficult to gain access to.

5.2 Future Work

The principal analysis work for this project was to thoroughly study the algorithms that determine position and orientation of the target spacecraft from 3D ladar data. Specific problems, which need to be addressed, include improving the initial orientation estimate, improved point-cloud matching methods, and integrating non-point cloud methods.

5.2.1 Future Work on Initialization

This thesis has identified several problems needing to be overcome before a ladar relative navigation system could be completed. First, the computational effort required for accurate navigation solutions was excessive; which leads us to the conclusion that it is "too slow" for real-time implementation in a navigation situation. The algorithms, in their current implementation, simply take too much time for use in real-time work. This is primarily because in the initialization of the algorithm, where the apriori knowledge is non-existent an initial orientation was picked from 312 different possibilities. This has been acceptable so far, since the focus of the research has been on demonstrating that an accurate solution could be found.

Therefore, new innovative algorithms must be added to the current work so that computational requirements can be extensively reduced to the level where they can run in real-time, or at approximately 2 to 5 Hz. Based on current simulation work, the primary

delay comes in the initialization of the relative location of the target. Currently, it takes several minutes for initialization on a standard PC. Future initialization research is recommended to attack this problem in several ways such as: a correlation method such as MACH filtering, spin filters, neural networks, geometry exploitation (feature extraction), etc. [74].

5.2.2 Future Work on Alternative Methods

One idea would be to construct an algorithm that exploits the geometry of the 3D point cloud, such that, a reasonable guess for a initial alignment can be obtained and re-run all experiments described above on a truth model. Secondly, since the point-to-point ICP variant was robust under all registrations and it makes sense that the point-to-plane variant will behave similarly. This may lead to quaternion estimates with sub-degree accuracy and computational times similar to those in Rusinkiewicz [30], Simon [9], and ZinBer [17] and others. Finally, an extended Kalman filter will be implemented for real-time hardware implementation with the Canesta ladar and experiments because it will provide the initial alignment on all 3-D registrations following the first one; as well as, providing both spacecrafts position, rates, and velocities continuously between all sample times and registration updates.

One exceptionally other promising method is the direct integration of high-resolution 2D Electro-Optic (e.g., CMOS imager) data directly with the coarser 3D ladar data. SDL/USU and CAIL has developed and patented a combined ladar/Electro-Optic (EO) TEXEL sensor that combines the advantages of both an EO camera and a ladar sensor (US Patent Number 6,664,529, held by SDL). This camera provides real-time spatially correlated ladar/EO data sets. The correlated TEXEL ladar/EO data may be able to provide higher accuracy measurements for orientation than either method alone. The 3D ladar data would then be used for orientation measurements by using Iterative Closest Point (ICP) algorithms to match the ladar point-cloud to a known solid model of the spacecraft. Simultaneously, real-time 2D image matching algorithms are used and the higher resolution (100 to 1) EO camera data is used to align relative displacement between ladar shots, improving the accuracy of the point cloud image. Once a position and orientation estimate is made from the ladar data,

an extended Kalman filter (EKF) based on the 12th order nonlinear dynamics described by Hills and Eulers equations [4] could be used to propagate and update the dynamic state of the spacecraft. Note that the output of this filter is used to seed the initial orientation used by point-cloud analysis, reducing the computational times required.

References

[1] W. Fehse, *Automated Rendezvous and Docking of Spacecraft*. Cambridge University Press, 2003.

[2] D. Zimpfer, P. Kachmar, and S. Tuohy, "Autonomous rendezvous, capture and in-space assembly: Past, present, and future," in *1st Space Exploration Conference: Continuing the Voyage of Discovery*, Jan. 2005.

[3] D. C. Woffinden and D. K. Geller, "Navigating the road to autonomous orbital rendezvous," *Journal of Spacecraft and Rockets*, vol. 44, no. 4, pp. 898–909, Aug. 2007.

[4] R. Fullmer, P. Patterson, and R. Pack, "The conceptual design of the guidance, navigation, and control system for a maintenance and repair spacecraft," in *AAS/AIAA Astrodynamics Conference*, Aug. 2001.

[5] A. Jelalian, *Laser Radar Systems*, ch. 1. Artech House, 1992.

[6] F. Blais, "A review of 20 years of range sensor development," *SPIE Electronic Imaging Videometrics VII*, vol. 5013, pp. 62–79, 2003.

[7] M. J. Halmos, M. Jack, J. Asbrock, C. Anderson, S. Bailey, G. Chapman, E. Gordon, P. Herning, M. Kalisher, L. Klaras, K. Kosai, V. Liquori, V. Randall, R. Reeder, J. Rosbeck, S. Sen, P. Trotta, and P. Wetzel, "3D flash ladar at Raytheon," *SPIE Laser Radar Technology and Applications VI*, vol. 4377, pp. 84–97, 2001.

[8] R. Craig, D. I. Gravseth, D. R. P. Earhart, J. Bladt, S. Barnhill, L. Ruppert, and C. Centamore, "Processing 3D flash ladar point-clouds in real-time for flight applications," in *Sensors and Systems for Space Applications*, vol. 6555, pp. 1–9, May 2007.

[9] J.-A. Beraldin, F. Blais, M. Rioux, L. Cournoyer, D. Laurin, and S. G. MacLean, "Short and medium range 3D sensing for space applications," *SPIE Visual Information Processing VI*, pp. 29–46, 1997.

[10] D. Laurin, J.-A. Beraldin, F. Blais, M. Rioux, and L. Cournoyer, "A three-dimensional tracking and imaging laser scanner for space operations," *SPIE Conference on Laser Radar Technology and Applications IV*, vol. 3707, pp. 278–289, Apr. 1999.

[11] C. L. Smithpeter, R. O. Nellums, S. M. Lebien, G. Studor, and G. James, "LADAR measurements of the international space station," *SPIE Laser Radar Technology and Applications VI*, vol. 4377, pp. 65–72, 2001.

[12] T. D. Cole, M. Zuber, G. Neuman, A. F. Cheng, A. Reiter, Y. Guo, and D. Smith, "Analysis of laser radar measurements of the asteroid, 433 Eros," *SPIE Laser Radar Technology and Applications VI*, vol. 4377, pp. 163–174, 2001.

[13] C. C. Liebe, A. Abramovici, R. K. Bartman, R. L. Bunker, J. Chapsky, *et al.*, "Laser radar for spacecraft guidance applications," in *Aerospace Conference*, pp. 2647–2662, Mar. 2003.

[14] R. Kornfeld, R. Bunker, G. Cucullu, J. Essmiller, F. Hadaegh, C. Liebe, C. Padgett, and E. Wong, "New millennium ST6 autonomous rendezvous experiment (ARX)," in *2003 IEEE Aerospace Conference*, vol. 1, pp. 1–12, Mar. 2003.

[15] S. Van Winkle, "Advanced Video Guidance Sensor (AVGS) project summary."

[16] I. Kawano, M. Mokuno, T. Kasai, and Takashi, "Result of autonomous rendezvous docking experiment of Engineering Test Satellite-VII," *Journal of Spacecraft and Rockets*, vol. 38, no. 1, pp. 105–111, Jan. - Feb. 2001.

[17] C. English, S. Zhu, C. Smith, S. Ruel, and I. Christie, "TriDAR: A hybrid sensor for exploiting the complementary nature of triangulation and lidar technologies." ISAIRAS 2005 Conference, Sept. 2005.

[18] Neptec, "http://www.neptec.com/neptecspace.html," Oct. 2006.

[19] J. E. Riedel, J. Guinn, M. Delpech, J. Dubois, D. Geller, and P. Kachmar, "A combined open-loop and autonomous search and rendezvous navigation system for the cnes/nasa mars previer orbiter," pp. 1–16, Feb. 5-9 2003.

[20] T. E. Rumford, "Demonstration of autonomous rendezvous technology (DART) project summary," in *SPIE Space Systems Technology and Operations Conference*, vol. 5088, pp. 10–19, 2003.

[21] T. Weismuller and M. Leinz, "GNC technology demonstrated by the Orbital Express autonomous rendezvous and capture system," in *29th Annual AAS Guidance and Control Conference*, pp. 1–9, Feb. 2006.

[22] O. Express, "http://www.darpa.mil/orbitalexpress/missionupdates.html," Feb. 2008.

[23] I. T. Mitchell, T. B. Gordon, K. Taskov, M. E. Drews, D. Luckey, M. L. Osborne, L. A. Page, H. L. Norris, and S. W. Shepperd, "GNC development of the XSS-11 micro-satellite for autonomous rendezvous and proximity operations," in *29th Annual AAS Guidance and Control Conference*, pp. 1–17, Feb. 4-8 2006.

[24] P. Miotto, D. Zimpfer, and H. Mamich, "Autonomous mission manager and GNC design for the hubble robotic vehicle de-orbit module," in *29th Annual AAS Guidance and Control Conference*, pp. 1–22, Feb. 4-8 2006.

[25] S. D. Blostein and T. S. Huang, "Estimating 3D motion from range data," in *1st Conference on Artificial Intelligence Applications*, pp. 246–250, 1984.

[26] Z.-C. Lin, H. Lee, and T. S. Huang, "Finding 3D point correspondences in motion estimation," in *8th International Conference on Pattern Recognition*, pp. 303–305, Oct. 1986.

[27] Z.-C. Lin, T. S. Huang, S. D. Blostein, H. Lee, and E. A. Margerum, "Motion estimation from 3D points sets with and without correspondences," in *IEEE Conference on Computer Vision and Pattern Reconition*, pp. 194–201, June 1986.

[28] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3D point sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI 9, no. 5, pp. 698–700, Sept. 1987.

[29] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol. 13, no. 4, pp. 376–380, Apr. 1991.

[30] T. Huang and C. Lee, "Motion and structure from orthographic projections," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 5, pp. 536–540, May 1989.

[31] J. Philip, "Estimation of three-dimensional motion of rigid objects from noisy observations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 1, pp. 61–66, Jan. 1991.

[32] J. Weng, T. S. Huang, and N. Ahuja, "Motion and structure from two perspective views: Algorithms, error analysis, and error estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 5, pp. 451–476, May 1989.

[33] B. K. P. Horn, "Closed-form solution of absolute orientation using quaternions," *Journal of the Optical Society of America*, vol. 4, no. 4, pp. 629–642, Apr. 1987.

[34] B. K. P. Horn, H. M. Hilden, and S. Negahdaripour, "Closed-form solution of absolute orientation using orthonormal matrices," *Journal of the Optical Society of America*, vol. 5, no. 7, pp. 1127–1135, July 1988.

[35] M. W. Walker, L. Shao, and R. A. Volz, "Estimating 3D location parameters using dual number quaternions," *CVGIP: Image Understanding*, vol. 54, no. 3, pp. 358–367, Nov. 1991.

[36] J. Weng, N. Ahuja, and T. S. Huang, "Optimal motion and structure estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 9, pp. 864–884, Sept. 1993.

[37] P. W. Smith and N. Nandhakumar, "Accurate structure and motion computation in the presence of range image distortions due to sequential acquisition," in *IEEE Computer Vision and Pattern Recognition Conference*, pp. 925–928, June 1994.

[38] Y. Liu and M. A. Rodrigues, "Correspondenceless motion estimation from range images," in *International Conference on Computer Vision*, pp. 654–659, Sept. 1999.

[39] X. Zhang, Y. Liu, and T. S. Huang, "Determining 3D structure and motion of manmade objects from image corners," in *5th IEEE Southwest Symposium on Image Analysis and Interpretation*, pp. 26–30, Apr. 2002.

[40] P. J. Besl and N. D. McKay, "A method for registration of 3D shapes," *IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–255, Feb. 1992.

[41] Y. Chen and G. Medoni, "Object modeling by registration of multiple range images," *Image and vision Computing*, vol. 10, no. 3, pp. 145–155, 1992.

[42] G. Godin, M. Rioux, and R. Baribeau, "Three-dimensional registration using range and intensity information," *SPIE Videometrics III*, vol. 2350, pp. 279–290, 1994.

[43] G. Turk and M. Levoy, "Zippered polygon meshes from range images," in *SIGGRAPH*, 1994.

[44] D. A. Simon, *Fast and Accurate Shape-Baed Registration*. Ph.D. dissertation, Carnegie Mellon University, Dec. 1996.

[45] T. Masuda, K. Sakaue, and N. Yokoya, "Registration and integration of multiple range images for 3D model construction," in *13th International Conference on Pattern Recognition*, pp. 879–883, Aug. 1996.

[46] A. Stoddart and A. Hilton, "Registration of multiple point sets," in *13th International Conference on Pattern Recognition*, pp. 40–44, 1996.

[47] R. Benjemaa and F. Schmitt, "Fast global registration of 3D sampled surfaces using a multi-z-buffer technique," in *IEEE International Conference on Recent Advances in 3D Digital Imaging and Modeling*, pp. 113–120, May 1997.

[48] R. Benjemaa and F. Schmitt, "A solution for the registration of multiple 3D point sets using unit quaternions," in *5th European Conference on Computer Vision*, pp. 34–50, June 1998.

[49] C. Dorai, J. Weng, and A. K. Jain, "Optimal registration of object views using range data," *IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol. 19, no. 10, pp. 1131–1138, Oct. 1997.

[50] C. Dorai, G. Wang, A. K. Jain, and C. Mercer, "Registration and integration of multiple object views for 3D model construction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 83–89, Jan. 1998.

[51] A. E. Johnson and S. B. Kang, "Registration and integration of textured 3D data," in *International Conference on Recent Advances in 3D Digital Imaging and Modeling*, May 1997.

[52] S. Weik, "Registration of 3D partial surface models using luminance and depth information," in *International Conference on Recent Advances in 3D Digital Imaging and Modeling*, pp. 93–100, May 1997.

[53] P. J. Neugebauer, "Geometrical cloning of 3D objects via simultaneous registration of multiple range images," in *International Conference on Shape Modeling and Applications*, p. 130, 1997.

[54] B. Matei and P. Meer, "Optimal rigid motion estimation and performanc evaluation with bootstrap," in *IEEE Conference of Computer Vision and Pattern Recognition*, pp. 339–345, 1999.

[55] K. Pulli, "Multiview registration for large data sets," in *IEEE International Conference on 3D Imaging and Modeling*, pp. 160–168, Oct. 1999.

[56] M. Levoy, S. Rusinkiewicz, M. Ginzton, J. Ginsberg, K. Pulli, D. Koller, S. Anderson, J. Shade, B. Curless, L. Pereira, J. Davis, and D. Fulk, "The digital michelangelo project: 3D scanning of large statues," in *SIGGRAPH*, pp. 131–144, 2000.

[57] D. Chetverikov, D. Svirko, D. Stepanov, and P. Krsek, "The trimmed iterative closest point algorithm," in *International Conference on Pattern Recognition*, 2002.

[58] T. Jost and H. Hugli, "A multi-resolution scheme ICP algorithm for fast shape registration," in *1st International Symposium on 3D Data Processing Visualization and Transmission*, June 2002.

[59] T. ZinBer, J. Schmidt, and H. Niemann, "A refined ICP algorithm for robust 3D correspondence estimation," in *IEEE International Conference on Image Processing*, Sept. 2003.

[60] D. Hahnel, S. Thrun, and W. Burgard, "An extension of the ICP algorithm for modeling nonrigid objects with mobile robots," in *International Joint Conference on Artifical Intelligence*, 2003.

[61] S.-H. Kim, Y.-H. Hwang, H.-K. Hong, and M.-H. Choi, "An improved ICP algorithm based on the sensor projection for automatic 3D registration," in *Third Mexican International Conference on Artificial Intelligence*, pp. 642–651, Apr. 2004.

[62] S. M. Rusinkiewicz, *Real-Time Acquistion and Rendering of Large 3D Models*. Ph.D. dissertation, Standford University, Aug. 2001.

[63] F. Stein and G. Medioni, "Structural indexing: efficient 3D object recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 125–145, Feb. 1992.

[64] O. Faugeras and M. Herbert, "The recognition and locating of 3D objects," *International Journal of Robotic Research*, vol. 5, no. 3, 1986.

[65] A. E. Johnson and M. Herbert, "Surface registration by matching oriented points," in *International Conference on Recent Advances in 3D Digital Imaging and Modeling*, pp. 121–128, May 1997.

[66] D. Huber, "Automatic 3D modeling using range images obtained from unknown viewpoints," in *Proceedings of the Third International Conference on 3D Digital Imaging and Modeling*, pp. 153–160, May 2001.

[67] C.-S. Chen, Y.-P. Hung, and J.-B. Cheng, "RANSAC-based DARCES: A new approach to fast automatic registration of partially overlapping range images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 11, pp. 1229–1234, Nov. 1999.

[68] K. Nishino and K. Ikeuchi, "Robust simultaneous registration of multiple range images," in *5th Asian Conference on Computer Vision*, pp. 1–8, Jan. 2002.

[69] "http://en.wikipedia.org/wiki/Image:3dtree.png," Aug. 2006.

[70] D. Kalman, "A singularly valuable decomposition: The SVD of a matrix," in *The American University*, Feb. 2002.

[71] T. K. Moon and W. C. Stirling, *Mathematical Methods and Algorithms for Signal Processing*. Prentice Hall, 2000.

[72] U. S. University, *USU LADAR SIM Public Release 1.0 Users Manual*, Space Dynamics Lab, Apr. 2003.

[73] K. Melville, "Integration of simulated ladar data with INS and GPS to obtain accurate position and orientation estimates of a sensor," Master's thesis, Utah State Universtiy, May 2003.

[74] D. W. Carlson, *Optimal Tradeoff Composite Correlation Filters*. Ph.D. dissertation, Carnegie Mellon University, Oct. 1996.

Appendices

Appendix A

Singular Value Decomposition (SVD) Algorithm Proof

In the text book written by Moon and Stirling [71], a proof behind how the SVD can be used to solve least squares problems was presented. Because the translational component has been removed from the point clouds R_1 and R_2, the pose estimation problem can be considered the least-squares solution of the generalized equation $A\mathbf{x} = \mathbf{b}$ where A would be the rotation matrix for our case. Now the problem becomes minimizing the norm of the error, $\|A\mathbf{x} - \mathbf{b}\|$. Now, substituting in the SVD for A

$$\min \|A\mathbf{x} - \mathbf{b}\| = \min \|U\Sigma V^H\mathbf{x} - \mathbf{b}\| = \min \|\Sigma V^T\mathbf{x} - U^H\mathbf{b}\|. \tag{A.1}$$

The latter equality follows from the fact that U is unitary, and multiplication by a unitary matrix does not change the length of a vector. Let $\mathbf{v} = V^H\mathbf{v}$ and $\hat{b} = U^H\mathbf{b}$. Then the least squares problem can be written as $\min \|\Sigma\mathbf{x} - \hat{\mathbf{b}}\|$. Thus, the solution to the problem is

$$\mathbf{v} = \Sigma^{\dagger}\hat{\mathbf{b}}. \tag{A.2}$$

The solution for $\hat{\mathbf{x}}$ comes from working the problem backwards:

$$V^T hat\mathbf{x} = \Sigma^{\dagger}U^H\hat{\mathbf{b}}, \tag{A.3}$$

$$\hat{\mathbf{x}} = V\Sigma^{\dagger}U^H\hat{\mathbf{b}}. \tag{A.4}$$

The matrix $V\Sigma^{\dagger}U^H$ is the pseudoinverse of A

$$A^{\dagger} = V\Sigma^{\dagger}U^H. \tag{A.5}$$

The final step would be to incorporate the identity matrix which takes care of the degenerate case and dropping the singular value matrix because it just scales the rotation.

$$A^{\dagger} = VSU^{H} \tag{A.6}$$

Note: All subscripts are subject to variable changes provided in the thesis because subscripts used for the following derivation come directly from the research text or paper!

Appendix B

Eigen-Value Decomposition (SVD) Algorithm Proof

Horn [33] has a proof for why the eigenvector associated with the maximum eigenvalue ends up being the optimal rotation. To find the rotation that minimizes the sum of squares of the errors, the quaternion $\dot{q}$ that maximizes $\dot{q}Q\dot{q}$ subject to the constraint that $\dot{q}\dot{q} = 1$ where Q was the symmetric matrix constructed from the cross-covariance matrix. The symmetric 4x4 matrix Q will have four real eigenvalues, say, λ_1, λ_2, , λ_3, and λ_4. A corresponding set of orthogonal unit eigenvectors $\dot{e}_1$, $\dot{e}_2$, $\dot{e}_3$, and $\dot{e}_4$ can be constructed such that

$$Q\dot{e}_i = \lambda_i \dot{e}_i \text{ for } i = 1, 2, 3, \text{ and } 4. \tag{B.1}$$

The eigenvectors span the 4D space, so an arbitrary quaternion $\dot{q}$ can be written as the linear combination in the form

$$\dot{q} = \alpha_1 \dot{e}_1 + \alpha_2 \dot{e}_2 + \alpha_3 \dot{e}_3 + \alpha_4 \dot{e}_4. \tag{B.2}$$

Since the eigenvectors are orthogonal

$$\dot{q} \cdot \dot{q} = \alpha_1^2 + \alpha_2^2 + \alpha_3^2 + \alpha_4^2. \tag{B.3}$$

Knowing that this has to equal one, and since $\dot{q}$ is a unit quaternion the following happens and since $\dot{e}_1$, $\dot{e}_2$, $\dot{e}_3$, and $\dot{e}_4$ are eigenvectors of N the conclusion is the second equation.

$$Q\dot{q} = \alpha_1 \lambda_1 \dot{e}_1 + \alpha_2 \lambda_2 \dot{e}_2 + \alpha_3 \lambda_3 \dot{e}_3 + \alpha_4 \lambda_4 \dot{e}_4 \tag{B.4}$$

$$\dot{q}^T Q\dot{q} = \dot{q} \cdot (Q\dot{q}) = \alpha_1^2\lambda_1 + \alpha_2^2\lambda_2 + \alpha_3^2\lambda_3 + \alpha_4^2\lambda_4 \tag{B.5}$$

Now suppose that we have arranged the eigenvalues in the order so that

$$\lambda_1 \geq \lambda_2 \geq \lambda 3 \geq \lambda_4. \tag{B.6}$$

Then we see that

$$\dot{q}^T Q\dot{q} \leq \alpha_1^2\lambda_1 + \alpha_2^2\lambda_2 + \alpha_3^2\lambda_3 + \alpha_4^2\lambda_4 < \lambda_1, \tag{B.7}$$

so that the quadratic from cannot become larger than the most positive eigenvalue. Also, the maximum is attained when we choose $\alpha_1 = 1$ and $\alpha_2 = \alpha_3 = \alpha_4 = 0$, that is, $\dot{q} = \dot{e}_1$. Thus the conclusion is that the unit eigenvector associated to the most positive eigenvalue maximizes the quadratic form $\dot{q}^T Q\dot{q}$,

$$\dot{q} = \dot{e}_1. \tag{B.8}$$

Note: All subscripts are subject to variable changes provided in the thesis because subscripts used for the following derivation come directly from the research text or paper!

Appendix C

Convergence Theorem of ICP Algorithm

Besl and McKay [40] have shown the convergence of the ICP algorithm. The key ideas were that: 1) least squares registration generically reduces the average distance between correspondences during each iteration, and 2) the closest point determination generically reduces teh distance for each point individually.

Theorem: The iterative closest point algorithm always converges monotonically to a local minimum with respect to the mean-square distance objective function.

Proof: Given $P_k = \vec{p}_{ik} = \vec{q}_k(P_o)$ and X, compute the set of closest points $Y_k = \vec{y}_{ik}$ as prescribed above given the internal geometric representation of X. The mean-squared error e_k of the correspondence is given by

$$e_k = \frac{1}{N} \sum_{i=1}^{N} \|\vec{y}_{ik} - \vec{y}_{ik}\|^2 . \tag{C.1}$$

Then the Q operator was applied to get $\vec{q}_k$ and d_k,

$$d_k = \frac{1}{N} \sum_{i=1}^{N} \|\vec{y}_{ik} - R(\vec{q}_{kR})\vec{p}_{i0} - \vec{q}_{kT}\|^2 . \tag{C.2}$$

It is always the case that $d_k \leq e_k$. Suppose that $d_k > e_k$. If this were so, then the identity transformation on the point set would yield a smaller mean square error than the least squares registration, which cannot possibly be the case. Next, let the least squares registration $\vec{q}_k$ be applied to the point set P_0, yielding the point set P_{k+1}. If the previous correspondence to the set of points Y_k were maintained, the the mean square error is still d_k as follows:

$$d_k = \frac{1}{N} \sum_{i=1}^{N} \|\vec{y}_{ik} - \vec{p}_{i,k+1}\|^2 . \tag{C.3}$$

However, during the application of the subsequent closest point calculation, a new matched point set Y_{k+1} would be obtained. Therefore, $\|\vec{y}_{i,k+1} - \vec{p}_{i,k+1}\| \leq \|\vec{y}_{ik} - \vec{p}_{i,k+1}\|$ for each i =1,N_p because the point $\vec{y}_{ik}$ was the closest point prior to transformation by $\vec{q}_k$. If $\vec{y}_{i,k+1}$ were further from $\vec{p}_{i,k+1}$ than $\vec{y}_{ik}$, then this would directly contradict the closest point operation being performed. Therefore, the mean square errors e_k and d_k must obey the following inequality:

$$0 \leq d_{k+1} \leq e_{k+1} \leq d_k \leq e_k, \tag{C.4}$$

for all k.

After all that, the lower bound occurs since the mean-squared errors cannot be negative. Because the mean-squared error sequence would be non-decreasing and bounded, the algorithm as stated above must converge monotonically to a minimum value.