A Vision Based Control System for Autonomous Rendezvous and Capture of a Mars Orbital Sample

By

Vishnu Jyothindran

B.S. Aerospace Engineering

University of Texas at Austin, 2010

SUBMITTED TO THE DEPARTMENT OF AERONAUTICS AND ASTRONAUTICS IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE IN AEROSPACE ENGINEERING AT THE MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2013

Signature of Author: _____
Department of Aeronautics and Astronautics
August 5, 2013

Certified by: _____
David W. Miller
Professor of Aeronautics and Astronautics
Thesis Supervisor

Certified by: _____
Alvar Saenz-Otero
Principal Research Scientist
Thesis Supervisor

Accepted by: _____
Eytan H. Modiano
Professor of Aeronautics and Astronautics
Chair, Graduate Program Committee

A Vision Based Control System for Autonomous Rendezvous and Capture of a Mars Orbital Sample

by

Vishnu Jyothindran

Submitted to the Department of Aeronautics and Astronautics
on August 2, 2013, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

NASA's Mars Sample Return (MSR) mission involves many challenging operations. The current mission scenario utilizes a small Orbiting Sample (OS) satellite, launched from the surface of Mars, which will rendezvous with an Earth Return Vehicle (ERV) in Martian orbit. One of the highest-risk operations is the guidance of the OS into the capture mechanism on the ERV. Since the OS will most likely be passive (with no attitude or propulsion control), the ERV must determine the OS' location in Martian orbit, and maneuver itself to capture it. The objective of this project is to design and develop a vision-based tracking and capture system using the SPHERES test bed. The proposed Mars Orbital Sample Return (MOSR) system uses a SPHERES satellite to emulate the combined motion of the capture satellite and the OS. The key elements of the system are: (1) a modified SPHERES satellite with a white shell to match the optical properties of the OS; (2) a capture mechanism; and (3) an optical tracking system. The system uses cameras mounted on the capture mechanism to optically track the OS. Software on the capture mechanism computes the likely maneuver commands for a capture satellite, which are then translated into relative motions to be performed by a SPHERES satellite, acting as the OS.

The focus of this thesis is on the vision-based algorithms and techniques used to ensure accurate 3-DOF ranging of the OS. The requirements of the OS tracking system are severe and require robust tracking performance in challenging illumination conditions without the use of any fiduciary markers(on the OS) to assist as a point of reference. A brief literature survey of common machine vision techniques for generic target tracking (in aerospace and other fields) is presented. In the proposed OS tracking system, two different methods are used for tracking and ranging of the OS. A Hough Transform algorithm is used to ensure accurate tracking of the OS in the 'near' field within all possible illumination regimes. A Luminosity based tracking algorithm is used to track the OS in the 'far' and 'near' field. Results from testing at MIT's Flat Floor Facility are presented to show the performance of these algorithms in an integrated Kalman Filter. Lastly, a new Model Predictive controller design is proposed for the fuel-optimal capture of the OS. Implementation and testing of the controller in the SPHERES satellite is presented and the comparisons to the SPHERES PD control system are revealed to highlight its strengths.

Thesis Supervisor: David W. Miller & Alvar Saenz-Otero

Acknowledgements

First, I would like to thank my thesis supervisor Professor David Miller and Dr. Alvar Saenz-Otero for their guidance and advice and for challenging me during my two years at MIT. Being part of the SPHERES team under their supervision was an honor and a privilege.

I would also like to thank everyone from the SPHERES team who helped me tirelessly with the SPHERES hardware. Shoutout to the team: Chris(Silent Sim assassin), David(Beacon buster), Bryan(Flat floor fanatic), Dustin(Lab clean-up czar), Brent(VERTIGO virtuoso), Jake(SPHERES Estimator guru), Alex(the Simulation king) and Greg(Battery guy). Your assistance and guidance will not be forgotten. Also, I could not have completed the project without the expert direction of Andrea Valmorbida and his remarkable efforts with MPC. Andrea, grazie per avermi dato l'opportunita' di lavorare con te. So bene che a volte non e' stato facile lavorare sull'hardware mentre eri a migliaia di kilometri di distanza a debuggare codice via Skype. Nonostante cio, MPC e' finalmente completato per SPHERES e speriamo di mandarlo presto sulla ISS.

My gratitude goes out to Dehann Fourie who was instrumental in teaching me the mechanism of the Kalman Filter. Special thanks to my UROPs Spencer and Maggie for helping me run the SPHERES hardware. At certain times, I know that the work I assigned was incredibly mundane but I appreciate your patience and diligence in doing everything I required of you.

MIT would not have been the same without the Sidney Pacific crowd and my closest friends Tamas, Alexander and Jameson. Special shoutout to my entire rowing crew who taught me more about life at 6 am in the morning than I could have ever imagined. You all surely contributed to recreating a home away from home. Thank you to my parents, and my sister for their encouragements, support, and love. I would not have made it here without them and the strong foundation they give to my life.

Contents

1 Introduction 13

1.1 Motivation 13

1.2 Base architecture 13

1.3 Previous work on vision based rendezvous navigation in space 14

1.4 The Hough Transform 17

1.4.1 Hough Transform Theory 17

1.4.2 Applications of Hough Transform 19

1.5 Astronomical photometry 21

1.5.1 Astronomical Photometry theory 21

1.5.2 Applications of Photometry 23

1.6 Model Predictive Control (MPC) 23

1.6.1 Model Predictive Control Theory 24

1.6.2 Applications of Model Predictive Control 25

1.7 Outline of thesis 25

2 The MOSR RDOS System 28

2.1 Introduction 28

2.2 Existing capabilities 29

2.2.1 OS Capture Mechanism 29

2.2.2 Synchronized Position Hold Engage & Reorient Experimental Satellites (SPHERES) as OS 30

2.2.3 OS Shell 32

2.2.4 Camera system 35

2.3 Software Architecture 36

3 Vision Algorithms and Kalman Estimator 39

3.1 Introduction 39

3.2 Hough Transform 39
3.2.1 Brief Review 39
3.2.2 The role of Canny edge detection 40
3.2.3 The Hough Transform applied to MOSR 42
3.2.4 Absolute Position estimation using the Hough Transform 47
3.2.5 Conclusion 52
3.3 Astronomical Photometry 53
3.3.1 Review 53
3.3.2 Astronomical photometry for MOSR 53
3.3.3 Conclusion 56
3.4 Kalman Estimator 57
3.4.1 Review 57
3.4.2 The Use of the Kalman Filter in MOSR 58
3.5 Kalman Filter Results 61
3.6 Conclusion 64
4 Model Predictive Control 66
4.1 Overview 66
4.2 Definition of the Control Strategy 66
4.3 Controller constraints 67
4.3.1 Field of view constraint 67
4.3.2 Limited control authority 67
4.3.3 Terminal constraints 68
4.3.4 Attitude constraints 68
4.4 MPC Software Architecture 69
4.5 Definition of the Reference frames 69
4.6 MPC Results 70
4.6.1 Offline MPC time delay calculation 71

4.6.2 MPC results in the SPHERES simulator 71
4.6.3 MPC results using Global Metrology on SPHERES hardware 72
4.6.4 PD and MPC Control comparison results using Global Metrology on SPHERES Hardware 74
4.7 Conclusion 77
5 Conclusion 80
5.1 Summary of Thesis 80
5.2 List of contributions 81
5.3 Future work 81
References 84
6 Appendix A 88
6.1 MPC Simulation results using PD and MPC Control 88
6.2 PD control test on SPHERES hardware 92
6.3 MPC Test on SPHERES hardware 97

List of Figures

Figure 1: Artist's rendition of Mars sample launching from MSR lander; (right) MSR Orbiter performing OS target search and acquisition in Mars orbit 14
Figure 2: DARPA's Orbital Express showing both spacecraft ASTRO and NEXTSAT[4] 15
Figure 3: Artist's rendition of the Hayabusa spacecraft landing on the Itokawa asteroid 16
Figure 4: Polar transform as used by Line Hough transform 17
Figure 5: Circular Hough Transform 19
Figure 6: Circular fiduciary markers on the ISS 21
Figure 7: Graphical representation of MPC[18] 24
Figure 8: SPHERES-MOSR test bed performing OS contact dynamics experiments on reduced gravity flight 29
Figure 9: Boresight view of SPHERES-MOSR testbed 30
Figure 10: SPHERES onboard the International Space Station[26] 31
Figure 11: Communications architecture for SPHERES MOSR system 32
Figure 12: OS Shell disassembled 33
Figure 13: OS Shell (with SPHERE for comparison) 33
Figure 14: OS Shell showing both light and dark configurations 34
Figure 15: OS on an air bearing support structure 34
Figure 16: uEye LE camera (lens not shown) 36
Figure 17: Overall Software Control architecture 37
Figure 18: Edge Detection using the Canny edge Detection algorithm 41
Figure 19: Raw Image of OS shell on the flat floor 43
Figure 20: Thresholded OS image 44
Figure 21: OS image after Canny Edge detection 45
Figure 22: Completed Hough Transform OS image 46
Figure 23: Partially occluded OS image processed with the Hough Transform 47
Figure 24: Camera reference frame perpendicular to image plane 48
Figure 25: Camera Reference frame along image plane 49
Figure 26: Time History plot of Hough Transform for a stationary OS in the 'near' field 50

Figure 27: Time History plot of Hough Transform for a stationary OS in the 'far' field 51
Figure 28: Relationship between OS size vs. depth 52
Figure 29: Number of Pixels on target scales with range 54
Figure 30: Linear fit plot for astronomical photometry 55
Figure 31: Inverse Quartic relationship of astronomical photometry 56
Figure 32: Comparison of Reference frames (Global Metrology and Camera) 61
Figure 33: Vision Estimator Convergence vs. Raw measurement 62
Figure 34: Position Convergence Comparison in the GM Reference frame 63
Figure 35: Velocity Convergence Comparison in the GM Reference frame 64
Figure 36: MPC Constraints 68
Figure 37: Online Communication Framework with MPC 69
Figure 38: Camera and Global Metrology reference frames 70
Figure 39: Time Delay computation for MPC Computation 71
Figure 40: X and Y Position Errors for MPC engine 72
Figure 41: Velocity vs. Time using the SPHERES testbed 73
Figure 42: Control accelerations vs. Time using the SPHERES testbed 74
Figure 43: Cumulative Δv requirement in MPC vs. PD on SPHERES testbed 75
Figure 44: Velocity profile comparison for MPC and PD controller on SPHERES testbed 76
Figure 45: XY Trajectory comparisons of PD vs. MPC 77
Figure 46: SPHERES Simulator PD MPC Comparison – Position Error vs. Time 88
Figure 47: SPHERES Simulator PD MPC Comparison – Control Accelerations vs. Time 89
Figure 48: SPHERES Simulator PD MPC Comparison – Δv requirements vs. Time 90
Figure 49: SPHERES Simulator PD MPC Comparison – Control accelerations vs. Time 91
Figure 50: SPHERES hardware PD Comparison – Control accelerations vs. Time 92
Figure 51: SPHERES hardware PD Comparison – Cumulative Δv vs. Time 93
Figure 52: SPHERES hardware PD Comparison – X and Y Position vs. Time 94
Figure 53: SPHERES hardware PD Comparison – Velocities vs. Time 95

Figure 54: SPHERES hardware PD Comparison – XY Trajectory 96
Figure 55: MPC Trajectory using SPHERES Hardware.....97
Figure 56: Cumulative Δv requirement for MPC using SPHERES hardware.....98
Figure 57: XY Trajectory for MPC using SPHERES hardware 99

Chapter 1

1 Introduction

1.1 Motivation

Ever since the beginning of planetary exploration on Mars, there's been significant interest in a Sample Return Mission. A Mars Sample Return Mission (MSR) would be a spaceflight mission to collect rock and dust samples from the Martian surface and return it to Earth. It is widely considered to be a very powerful form of exploration since the analysis of geological samples is free from the typical spacecraft constraints such as time, budget, mass and volume. According to R.B Hargraves[1], it is considered to be the "holy grail" of robotic planetary mission because of its high scientific return.

However, due its high system complexity (and high financial needs), all MSR-type missions have not passed any planning phases. The goal of this thesis is to demonstrate a control system for the rendezvous and capture of a passive Martian sample in Martian orbit. This is considered to be a high-risk maneuver since rendezvous and capture of a completely passive object has not been attempted past Earth orbit.

1.2 Base architecture

For the purposes of this thesis, the base sample return architecture used is that of Mattingly et. al[2]. This architecture consists of a small passive Orbiting Sample satellite (OS) and chaser satellite. The OS is launched from Martian surface and contains Martian geological samples recovered by a caching rover. The OS is completely passive during the entire rendezvous and capture process and it is possibly fitted with a radio beacon for long-range tracking. The chaser satellite uses a visual band camera system for final rendezvous and capture and 'the need for risk mitigation in this system is critical'[3]. This is the primary focus of this thesis.

Figure 1 shows an artist's rendition of this architecture.

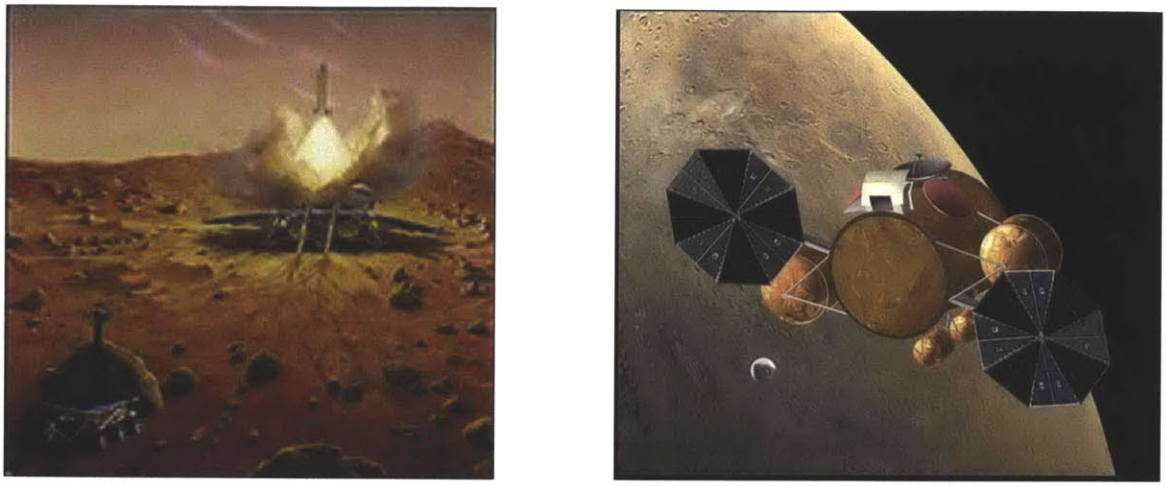

Figure 1: Artist's rendition of Mars sample launching from MSR lander; (right) MSR Orbiter performing OS target search and acquisition in Mars orbit

1.3 Previous work on vision based rendezvous navigation in space

Vision based navigation for rendezvous and capture has not been well used in spacecraft in the past. The first major use of machine vision for rendezvous was DARPA's Orbital express[4]. The project hoped to demonstrate several satellite servicing operations and technologies including rendezvous, proximity operations, station keeping, capture and docking. The Advanced Video Guidance sensor was used for docking between two spacecraft. This sensor consisted of a laser diode to illuminate a reflective visual target that was processed by vision algorithms on board the spacecraft. A relative state estimate consisting of relative spacecraft attitudes and positions was computed by the on-board vision system. Figure 2 shows the two spacecraft used in the Orbital Express mission.

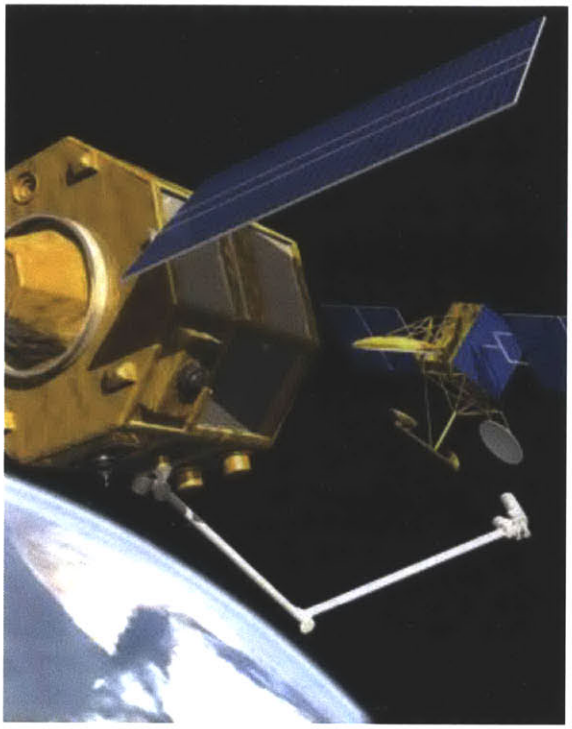

Figure 2: DARPA's Orbital Express showing both spacecraft ASTRO and NEXTSAT[4]

One of the more recent missions to exhibit vision based spacecraft navigation was the Hayabusa mission by the Japanese Aerospace Exploration Agency(JAXA) which performed a sample return mission of the Itokawa asteroid[5]. The Hayabusa spacecraft attempted multiple vision based landings on the asteroid to gather samples on the surface. However, due to the unknown local terrain geometry and the visual appearance of the asteroid, many of the landings were not ideal[5]. The spacecraft navigated to the landing sites by tracking features on the surface of the asteroid. Figure 3 shows an artist's rendition of the Hayabusa satellite attempting to land on the surface of the asteroid.

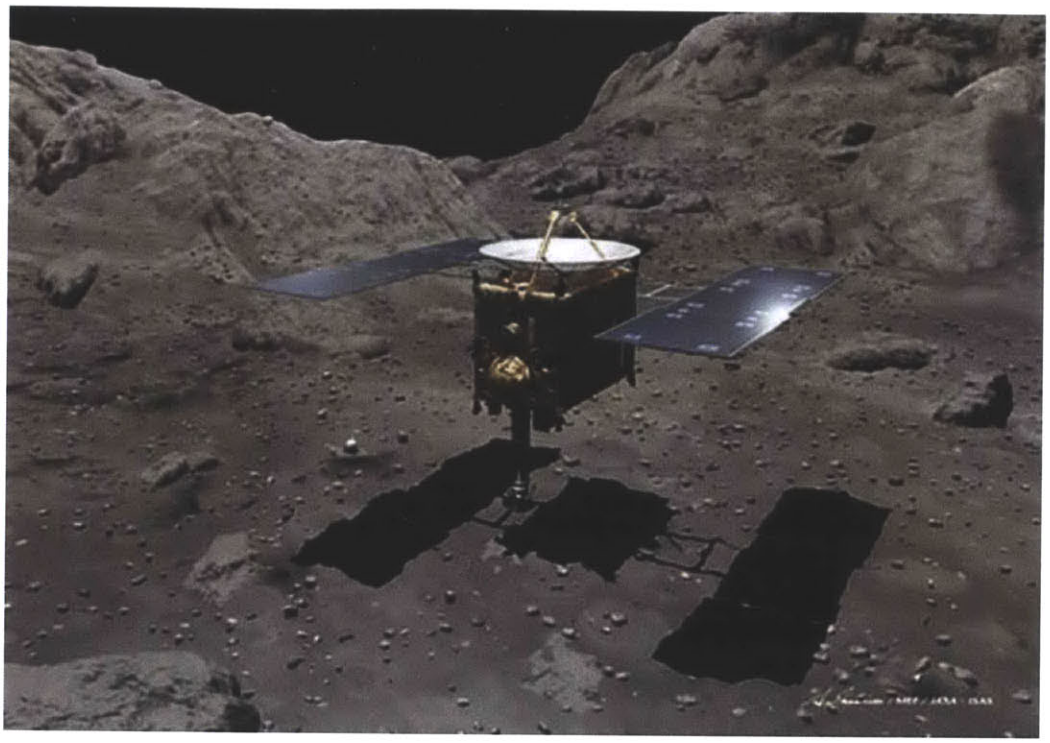

Figure 3: Artist's rendition of the Hayabusa spacecraft landing on the Itokawa asteroid

A small yet significant application of optical navigation was done by the Mars Reconnaissance Orbiter (MRO). From 30 days to 2 days prior to Mars Orbit Insertion, the spacecraft collected a series of images of Mars' moons Phobos and Deimos. By comparing the observed position of the moons to their predicted positions relative to the background stars, the mission team was able to accurately determine the position of the orbiter in relation to Mars. While not needed by Mars Reconnaissance Orbiter to navigate to Mars, the data from this experiment demonstrated that the technique could be used by future spacecraft to ensure their accurate arrival. Adler et al, have also proposed using such a system for an MSR mission[6].

The common feature between both the missions mentioned above was the tracking of fiduciary markers (Orbital Express) or the use of known features (Hayabusa and MRO). Both of these navigation techniques track individual known reference points and use this to update state estimates. However, the use of generic shape detection algorithms, where the only assumption made is that of the target's outer shape has not been done in

spacecraft. This project attempts to show vision based rendezvous and docking using such generic vision techniques. Two such techniques used are: (1) The circular Hough Transform and (2) astronomical photometry. This also allows the use of a single camera instead of using a stereoscopic (two or more cameras) system.

1.4 The Hough Transform

1.4.1 Hough Transform Theory

The Hough transform is a feature extraction technique used in image analysis and machine vision. The purpose of the technique is to find imperfect instances of objects within a certain class of shapes by a voting procedure.

The Hough transform was invented by Richard Duda and Peter Hart[7]. There is some debate as to who the real inventor is as a patent was filed shortly(or around the same time as the publication of the paper by Duda and Hart) by Paul Hough[8] who the algorithm is named after.

The simplest case of Hough transform is the linear transform for detecting straight lines. In the image space, a line can be described as y = mx + b where the parameter m is the slope of the line, and b is the y-intercept. In the Hough transform, however, this can be simplified further using polar coordinates, as shown in Figure 4 and Equation 1 and proposed by Duda[7].

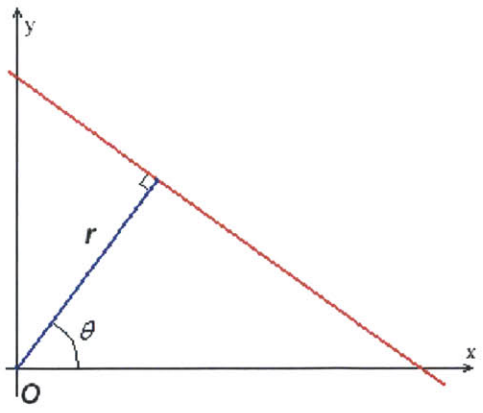

Figure 4: Polar transform as used by Line Hough transform

$$x\cos\theta + y\cos\theta = r \tag{1}$$

Extending this to discretized points(x_i, y_i) on an image, it is easy to see that the each coordinate pair now transforms into sinusoidal curves in the $\theta - r$ plane as defined by Equation 2:

$$\rho = x_i\cos\theta + y_i\cos\theta \tag{2}$$

One can also see that for a collinear set of points (x_i, y_i) and (x_j, y_j); the curves corresponding to these points intersect at a unique point (θ_o, r_o). Therefore, the problem for finding collinear points is converted to finding concurrent curves. This can be done quickly computationally by any equation solver.

This concept can also be extended to the circle as proposed by Kierkegaard et. al [9]. Here the parametric representation of the equation of a circle can be written in the form as given in Equation 3.

$$(x - a)^2 + (y - b)^2 = c^2 \tag{3}$$

For any arbitrary point (x_i, y_i), this can be transformed into a surface in three dimensional space with the dimensions being (a,b,c) as defined in Equation 4:

$$(x_i - a)^2 + (y_i - b)^2 = c^2 \tag{4}$$

Each sample point (x_i, y_i) is now converted into a right circular cone in the (a,b,c) parameter space. If multiple cones intersect at one point (a_o,b_o,c_o), then these sample points now exist on a circle defined by the parameters a_o,b_o, and c_o.

The above process is repeated over all the sample points in an image and votes are gathered for each set of parameters (a_o,b_o,c_o). This method is extremely computationally intensive and has a memory requirement on the order of O(m n c) where mn is the total number of sample points(i.e. image size) and c is the number of possible radii. This memory requirement is needed for the accumulator array where the columns of the

accumulator array are the possible radii in the image and each row is the unique set of (a,b) points for the coordinate of the centers.

However, the computational burden of this method can be significantly lower if the radius c is known a priori. This reduces the 2-D accumulator array into a single column vector for a specific c_o or range of radii ($c_i - c_j$). Thus, if the radius is known a priori, the algorithm's memory requirement reduces to O(mn) time, since fewer cells in the accumulator array have to be updated for each sample point. Figure 5 and shows a purely graphical representation of the circular Hough transform.

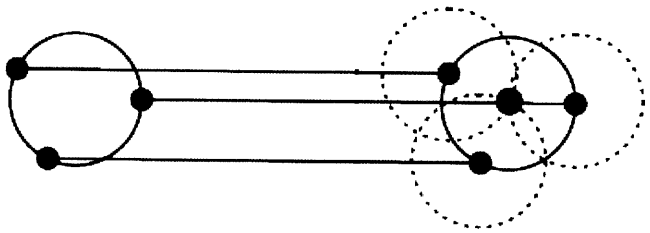

Figure 5: Circular Hough Transform

The process described above only applies to certain simple shapes like lines and curvilinear shapes like circles and ellipses. However, this can be extended to detect shapes in any orientation as proposed by Ballard et. al [10]. This key advancement in the theory shows its possible use in detecting generic objects in a space environment.

1.4.2 Applications of Hough Transform

Uses of the Hough Transform are spread over a variety of fields but their use in aerospace applications is very limited.

One of the most widespread uses of the Hough transform is in the medical industry. Zana et. Al [11] proposed using the Hough transform to detect blood vessels and lesions in retinal scans. This method is used on multiple retinal scans at different orientations and common blood vessels and lesions are marked in each so that physicians can follow

the progress of a certain disease or general retinal health over time. This also improves the identification of some lesions and allows easy comparison of images gathered from different sources.

Another use of the Hough transform is in traffic monitoring. As suggested by Kamat et. al[12], the Hough transform is used for the general problem of detection of vehicle license plates from road scenes for the purpose of vehicle tracking. It describes an algorithm for detecting a license plate from a road scene acquired by a CCD camera using image processing techniques such as the Hough transform for line detection (the shape of the license plates is defined by lines). Here the Hough transform is more memory efficient since the dimensions of the license plates are known.

Lastly, the use of the Hough transform is also fairly common in the geological industry for detecting features in terrain. As proposed by Karnieli et. al[13], the Hough transform is used for detecting geological lineaments in satellite images and scanned aerial photographs.

As previously mentioned, the use of the Hough transform is severely limited in the aerospace industry. Only one application is known at this time. Casonato et. al[14] proposed an automatic trajectory monitoring system designed for the rendezvous between the automatic transfer vehicle (ATV) and the International Space Station (ISS). During the final approach phase, a TV camera on the ISS currently provides images of ATV visual targets to be used by ISS crew for visual monitoring. The proposed monitoring system, based on the Hough transform is applied to these TV images of the approach, and is intended to autonomously and automatically determine relative ATV-ISS position and attitude. Artificial intelligence techniques for edge detection, Hough transform and pattern matching are used to implement a recognition algorithm, able to fast and accurately indicate ATV visual targets position and orientation with respect to ISS. Those values are then processed in order to calculate the full set of relative ATV navigation data. However, in this case, the Hough transform is used only in a supervisory role and not as part of a vision based control system. It is also important to note that the circular Hough transform is used to detect fiduciary markers (in the shape of circles) that already exist on the outer facade of the ISS (shown in Figure 6).

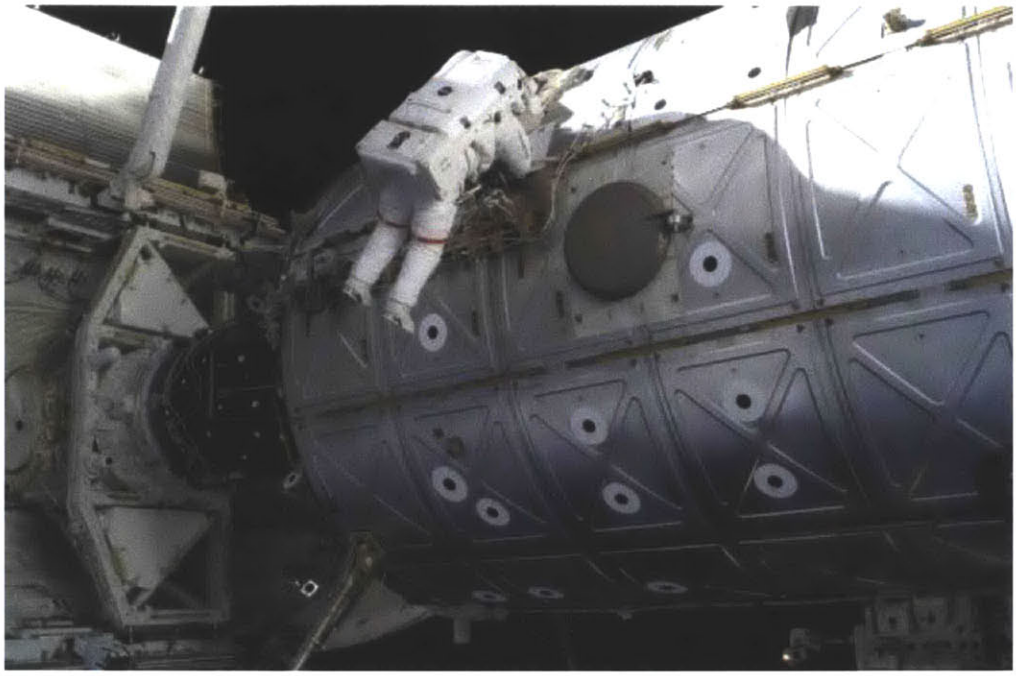

Figure 6: Circular fiduciary markers on the ISS

1.5 Astronomical photometry

1.5.1 Astronomical Photometry theory

Photometry is an astronomical technique concerned with measuring the flux, or intensity of an astronomical object's electromagnetic radiation. In this case, the spectrum of electromagnetic radiation is limited to the visible light spectrum.

When using a CCD camera to conduct photometry, there are a number of possible ways to extract a photometric measurement from the raw CCD image. The observed signal from an object will typically be smeared (convolved) over many pixels. When obtaining photometry for a point source the goal is to add up all the light from the point source and subtract the light due to the background. This means that the apparent brightness or luminosity (F) of a point source in an image is a simple sum of pixel brightness $B(x_i,y_i)$ of all the pixels across the point source(with dimensions m, n pixels in each direction) as shown in Equation 5.

$$F_{point} = \sum_{i=0,j=0}^{m,n} B(x_i, y_i) \tag{5}$$

The pixel brightness $B(x_i,y_i)$ is the CCD value for that pixel in a grayscale (monochrome) camera and is the average across all three red, green, blue channels for a color camera as shown in Equation 6 and Equation 7.

$$B(x_i, y_i) = CCD(x_i, y_i) \quad for\ monochrome\ CCDs \tag{6}$$

$$B(x_i, y_i) = \frac{RED(x_i, y_i) + BLUE(x_i, y_i) + GREEN\ (x_i, y_i)}{3} \quad for\ color\ (RGB)\ CCDs \tag{7}$$

The flux density or apparent brightness of a point source can also be given by the flux density relationship in Equation 8:

$$F_{point} = \frac{L}{r^2} \tag{8}$$

Here, L is the true luminosity of the source and r is the distance away from the source.

In this application, the target is not a point source. For Equation 8 to apply, each pixel is regarded as a point source. Therefore, the apparent luminosity F_{target} is simply the sum of the apparent luminosity over all points on the target as shown in Equation 9.

$$F_{target} = \sum F_{point\ i} \tag{9}$$

It is interesting to note, that one can calculate the distance away from the target using this method as shown in Equation 10.

$$r = \sqrt{\frac{L}{F_{target}}} \tag{10}$$

However, the true luminosity L needs to be known first. In this application, the true luminosity is pre-calculated from a previous state estimate given to the algorithm from another ranging algorithm (such as the radio beacon described in Section 1.2). This algorithm also shows an increased sensitivity to depth measurements since the distance to the target given by r scales quadratically to the flux density (or apparent brightness) of the target.

1.5.2 Applications of Photometry

As mentioned before, photometry is limited to the field of astronomy. The preeminent source on the subject is Astronomical Photometry: A guide by Christiaan Sterken and Jean Manfroid[15].

The first major application to ranging was by Shapley et. al[16] who used astronomical photometry to measure the distance to star clusters. This was done by measuring the apparent brightness of Cepheid stars whose true luminosities are known. The use of photometry in a spacecraft application is limited to one. Reidel et. al[17] proposed using such a ranging technique for the Mars CNES Premier Orbiter mission but this has not been tested in hardware.

1.6 Model Predictive Control (MPC)

An equally important role is played by the control algorithms in this system. Sections 1.4 to 1.5 presented the vision algorithms used while Section 1.6 will describe the control system and its applications.

1.6.1 Model Predictive Control Theory

Model predictive controllers rely on dynamic models of the process, most often linear empirical models obtained by system identification. The main advantage of MPC is that it allows the current timeslot to be optimized, while keeping future timeslots in account. This is achieved by optimizing a finite time-horizon, but only implementing the current timeslot. MPC has the ability to anticipate future events and can take control actions accordingly. PID controllers do not have this predictive ability. MPC is based on an iterative, finite horizon optimization of a plant model. At time t the current plant state is sampled and a cost minimizing control strategy is computed (via a numerical minimization algorithm) for a relatively short time horizon in the future, (t - t+T). An online or on-the-fly calculation is used to explore state trajectories that emanate from the current state and find a cost-minimizing control strategy until time t+T. Only the first step of the control strategy is implemented, then the plant state is sampled again and the calculations are repeated starting from the now current state, yielding a new control and new predicted state path. The prediction horizon keeps being shifted forward. Figure 7 shows a graphical representation of MPC and the receding horizon control strategy.

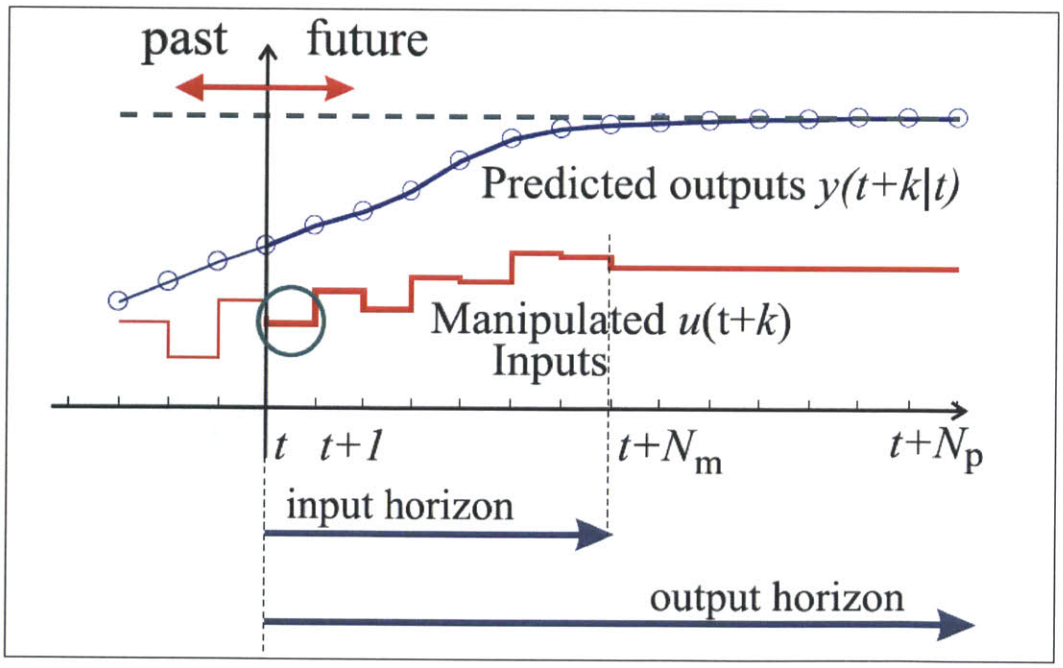

Figure 7: Graphical representation of MPC[18]
Receding horizon strategy: only the first one of the computed moves u(t) is implemented.

MPC also maintains a history of past control actions. This history is used in a cost optimization to optimize a certain parameter of the control system. In this application, the cost function relates directly to thruster fuel efficiency. For details about this system, the reader is advised to refer to [19].

1.6.2 Applications of Model Predictive Control

As briefly mentioned in Section 1.6, MPC is commonly used in the chemical and oil industries.

Though the ideas of receding horizon control and model predictive control can be traced back to the 1960s[20], interest in this field started to develop in the 1980s after dynamic matrix control (DMC) [21]. DMC was conceived to tackle the multivariable constrained control problems typical for the oil and chemical industries and it had a tremendous impact in those industries.

In spacecraft applications, MPC has been proposed and theorized before. However, no hardware applications are known at this point. Manikonda et. al[22] proposed an application of a model predictive control-based approach to the design of a controller for formation keeping and formation attitude control for the NASA DS3 mission[23]. Richards et. al [24] presented an MPC controller for a spacecraft rendezvous approach to a radial separation from a target. Both Richards and Manikonda used MPC to reduce thruster fuel or power usage. Lavaei et. al[25]further extended the use of MPC to multiple cooperative spacecraft with communication constraints.

1.7 Outline of thesis

The work presented in this thesis is part of an overall research initiative to develop computer vision based rendezvous and docking algorithms for an MSR-type mission. This thesis presents the three steps in this program: the development of a research testbed to be tested onboard the ISS, the verification of this research testbed through the implementation of a vision tracking algorithms (based on the Hough transform and astronomical photometry), and the preliminary analysis of an MPC based controller for rendezvous and capture.

Chapter 2 presents the design and development of a testbed for computer vision based navigation onboard the ISS. This testbed utilizes the Synchronize Position Hold Engage Re-orient Experimental Satellites (SPHERES) along with new hardware such as cameras and lights. This hardware was developed in a partnership with Aurora Flight Sciences as part of the Mars Orbital Sample Return Rendezvous and Docking of Orbital Sample (MOSR) program.

Chapter 3 discusses the performance and analysis of the machine vision algorithms outlined in Section 1.4 and Section 1.5. Experimental results using these algorithms are shown along with suggested improvements and integration into a Kalman filter. An approach to a linearly coupled Kalman filter is proposed and experimental comparisons with the SPHERES Global Metrology system (used as ground truth) are presented.

Chapter 4 presents the Model Predictive Controller. Simulation and experimental results are shown to compare its performance with classic PID controllers.

Chapter 5 summaries the conclusions and contributions of this thesis and discusses future work.

Chapter 2

2 The MOSR RDOS System

2.1 Introduction

The current MSR mission scenario utilizes a small Orbiting Sample (OS) satellite, launched from the surface of Mars, which will rendezvous with a chaser spacecraft. The guidance of the OS into the capture mechanism on the chaser satellite is critical. Since the OS will most likely be passive—possibly outfitted with a radio beacon for long-distance detection, but with no means of active propulsion or attitude control—the chaser spacecraft must determine the location of the OS in Martian orbit, and maneuver itself to capture it. As proposed earlier in Chapter 1, the chaser spacecraft will rely on optical tracking using a single visual-band camera to perform the final rendezvous and capture operation.

Using the Synchronize Position Hold Engage Re-orient Experimental Satellites (SPHERES) satellites (created at the Massachusetts Institute of Technology), the existing MOSR testbed is augmented to incorporate optical tracking and control. This augmentation will facilitate the repeated trial and testing of new control and vision based algorithms to be used on a final launch MSR mission. The goal of the MOSR testbed is to provide a research platform where such algorithms and controls can be tested in a relatively 'risk-free' environment onboard the International Space Station.

This thesis focuses on the Phase II work of the MOSR project. The Phase II effort focuses on the adaptation of the SPHERES MOSR test bed to address the "last few meters" GN&C problem of Mars Sample Return Orbiting Sample capture - integrating and demonstrating the tracking and relative maneuver capabilities with respect to the passive OS. There are two main technical objectives for the Phase 2 effort: (1) provide visual tracking for the MOSR test bed by integrating the required cameras and lighting and (2) emulate the combined chaser/OS dynamics using a SPHERES satellite by

integrating the relative motion controls into the MOSR baseplate and SPHERES software.

2.2 Existing capabilities

2.2.1 OS Capture Mechanism

The existing SPHERES MOSR test bed currently has the capability to analyze the contact dynamics between the OS and the OS capture mechanism, using a SPHERES satellite as a surrogate OS and incorporating an instrumented baseplate to measure the contact loads between the OS and the capture mechanism. The MOSR test bed is also equipped with boresight and side-view cameras to provide video data to correlate the contact dynamics data.

Figure 8 shows the capture mechanism being tested onboard a reduced gravity flight.

Figure 9 shows the boresight camera view which is the primary capture view used by the vision system.

Figure 8: SPHERES-MOSR test bed performing OS contact dynamics experiments on reduced gravity flight

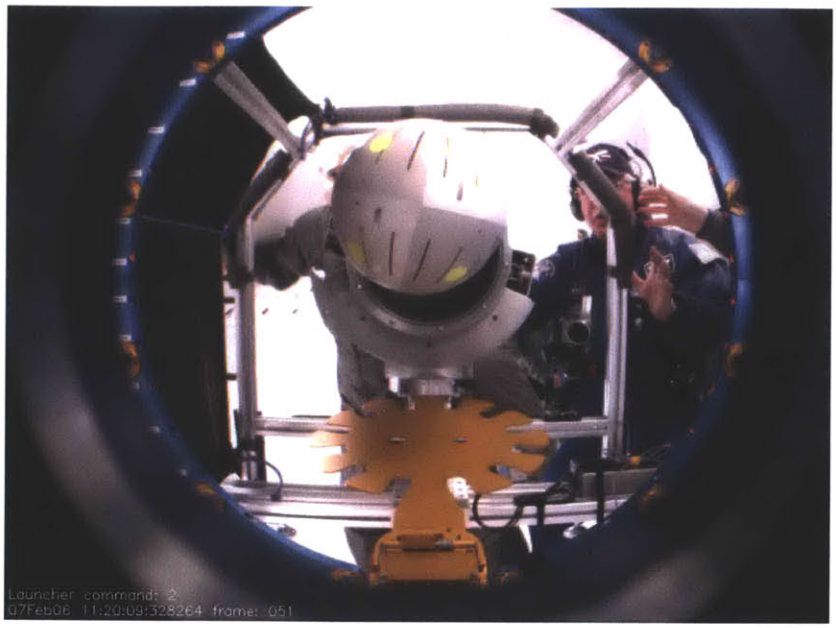

Figure 9: Boresight view of SPHERES-MOSR testbed

2.2.2 Synchronized Position Hold Engage & Reorient Experimental Satellites (SPHERES) as OS

The SPHERES test bed is a test bed for the development of multi-satellite GN&C algorithms. Currently, there are three SPHERES used in 3DOF ground testing and three SPHERES on the ISS for full 6DOF microgravity testing, shown in Figure 10. This test bed provides a unique opportunity for researchers to test algorithms, analyze data, then refine and uplink new algorithms in a relatively short period of time. This iterative process allows algorithms to be matured in a low-cost, low-risk environment.

Each satellite is equipped with a processor, two communication transceivers, battery power, 12 cold-gas thrusters, and 24 ultrasound receivers. These ultrasound receivers are used in a time-of-flight metrology system analogous to GPS. Using ultrasound beacons placed in a designated volume around the SPHERES, the satellites individually measure their respective positions and attitudes with an accuracy of a few millimeters and 1-2 degrees, respectively (Miller et. al[26]).

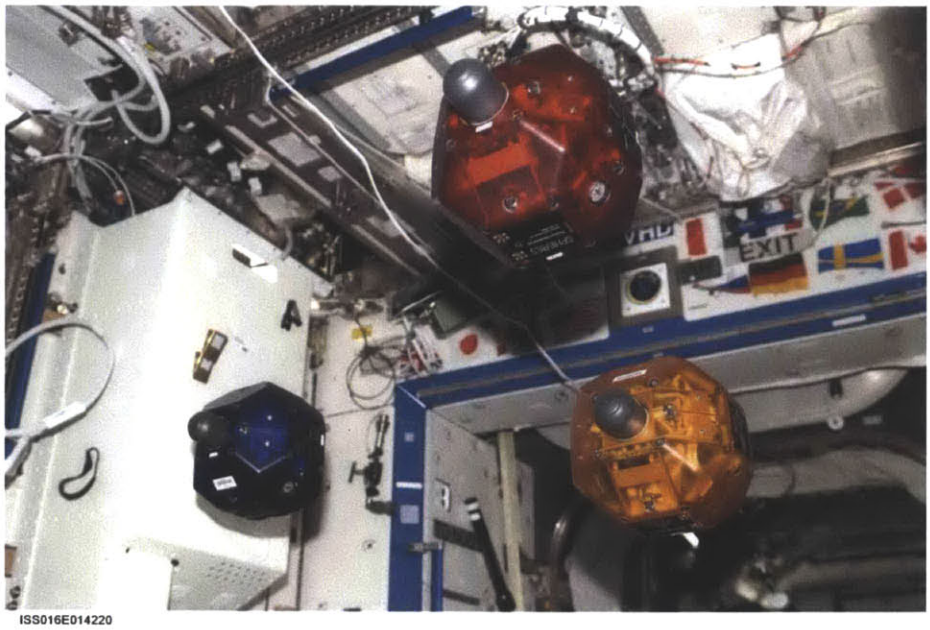

Figure 10: SPHERES onboard the International Space Station[26]

One of these SPHERES will be modified to serve as a surrogate OS. While the actual OS will not have the same capabilities as SPHERES, many of the SPHERES subsystems will aid the development, testing and verification of the SPHERES MOSR test bed. First, the propulsion system will enable the surrogate OS to maneuver and perform motions equivalent to the chaser being actuated. Using this reversed-roles maneuvering precludes the development of a larger chaser satellite that would have to maneuver with the capture cone. Second, the communication system will allow the OS to receive the control commands necessary to execute the reversed-roles maneuvering. Third, the ultrasound metrology system provides the OS with the ability to determine its position and attitude. These independent measurements will provide a means to validate the visual tracking algorithm's accuracy. Figure 11 shows a brief communications overview in this reversed-roles model.

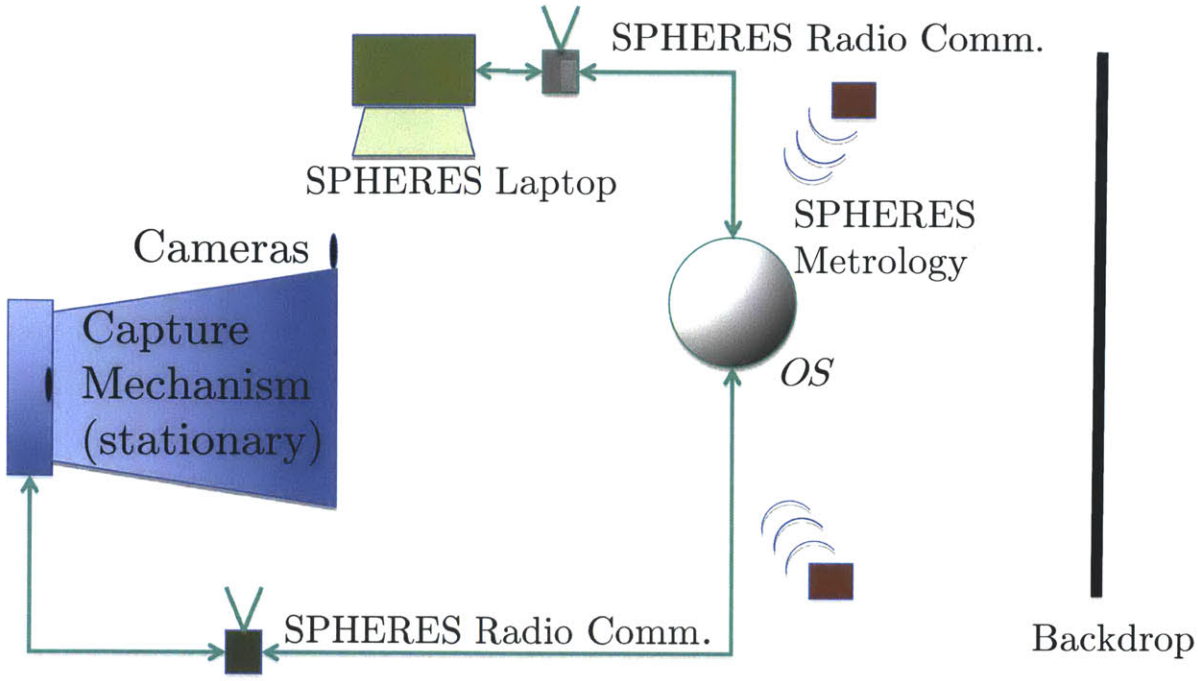

Figure 11: Communications architecture for SPHERES MOSR system

2.2.3 OS Shell

As shown in Figure 10, the SPHERES satellites are of different colors. In order to make the SPHERES MOSR system not dependent on a single satellite, a removable OS shell prototype is built. The prototype (shown disassembled in Figure 12) is fabricated from black ABS-M30 plastic using Fused Deposition Modeling (FDM) rapid prototyping technology. It consists of six parts – two hemispheres that encase the satellite, connected via four size 4-40 button head cap screws to two disks, one on each side that locks the assembly in place. The hemispheres each contain four extrusions on its interior surface that fit snugly over the satellite, preventing the shell from moving relative to it. Two additional disks rotate into the two hemispheres with a quarter-lock mechanism to provide access to the satellite's battery compartments. Figure 13 shows the assembled shell next to a SPHERES satellite to illustrate the mounting orientation.

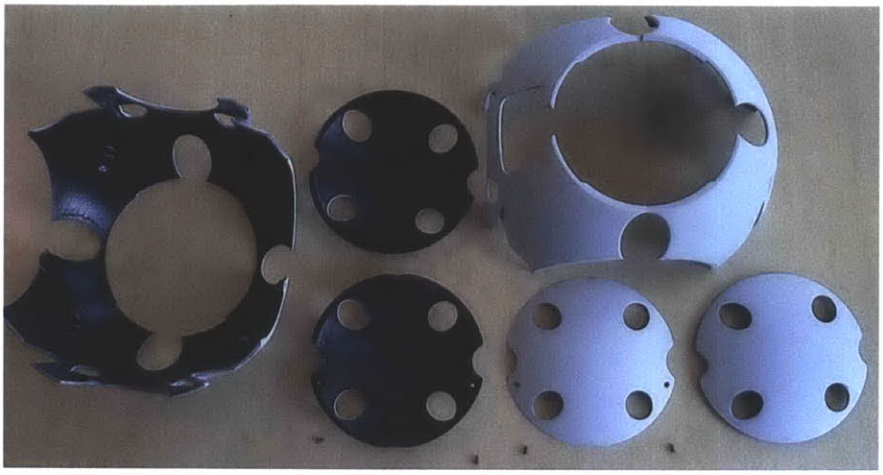

Figure 12: OS Shell disassembled

Figure 13: OS Shell (with SPHERE for comparison)

The color scheme selected for the shell involves a quarter of the sphere being painted flat black and the rest of the sphere painted flat white, as shown in Figure 14. This allows for testing of the OS in a variety of lighting conditions that simulate that of on orbit operations, where the majority of the OS will be illuminated by the capture spacecraft during rendezvous. Lighting of the OS will range from fully illuminated (only the white section being visible) to the worst case lighting condition where the black section of the OS will cover half of the OS surface orientated towards the tracking cameras so only half of the OS is visible, as seen in Figure 15.

Figure 14: OS Shell showing both light and dark configurations

During flat floor testing, the orbiting sample is mounted on a SPHERES air bearing support structure that is suspended on a cushion of CO2 gas, allowing it to move around the polished flat floor surface with minimal friction, thus simulating motion in a 2-D space environment. Figure 15 shows the OS on the MIT flat floor facility.

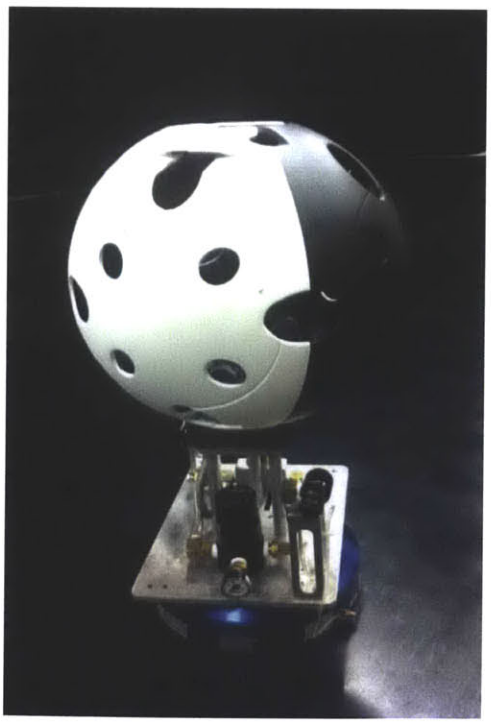

Figure 15: OS on an air bearing support structure

2.2.4 Camera system

The visual tracking of the OS utilizes at least two cameras based on the already NASA-approved VERTIGO cameras, uEye UI-1225LE-M-HQ. These gray-scale cameras have a resolution of 752x480 pixels with 8-bits of depth over a USB 2.0 interface. One camera is mounted inside the baseplate looking along the bore sight of the capture cone. This enables video from the final approach and capture of the OS. The second camera is mounted offset from the capture cone and provides a wider field of view. This enables visual tracking of the OS tens-of-meters away from the capture system and provides the mechanism for the rendezvous and approach of the chaser satellite with the OS, as emulated by the movements of the SPHERES module. Table 1 presents the camera's specifications. Figure 16 shows the camera with the lens detached. The focal length of the lens used is 5 mm.

Table 1: Specifications of uEye LE Camera

Sensor	1/2" CMOS with Global Shutter
Camera Resolution	640 x 480 pixels
Pixel Size	6.0 μm, square
Lens Mount	CS-Mount
Frame Rate	87 FPS (Max), 10 FPS (Typical)
Exposure	80 μs - 5.5 s
Power Consumption	0.65 W each
Size	3.6 cm x 3.6 cm x 2.0 cm
Mass	12 g

Figure 16: uEye LE camera (lens not shown)

2.3 Software Architecture

The overall software architecture is shown in Figure 17. The chaser's visual tracking algorithm provides the relative positions and velocities of the chaser and the OS. The chaser then, using the controller and model described in Section 1.6 and Chapter 4, computes its required forces and torques to rendezvous and capture the OS. For the purposes of the MOSR Chaser/OS emulation, the chaser's baseplate computer then converts these into the forces required to move the SPHERES satellite, representing the OS, in order to emulate the combined motions. The baseplate then transmits these to the SPHERES satellite, whose internal controller then actuates its respective thrusters to execute the motion. This architecture expects to transmit the x, y, and z forces (or accelerations) and relative location components from the chaser's baseplate to the OS at a rate greater than or equal to 1Hz.

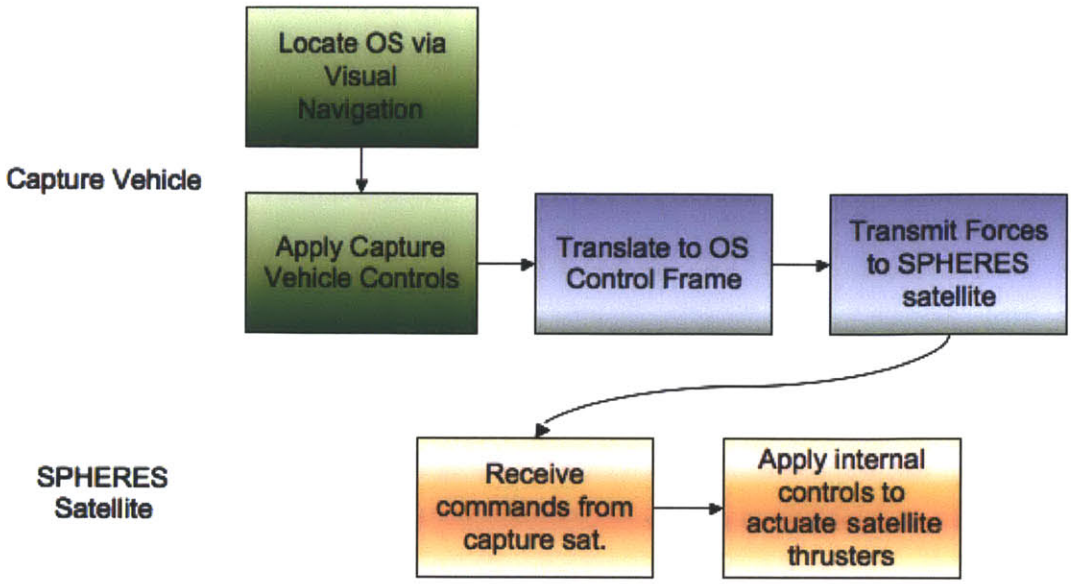

Figure 17: Overall Software Control architecture

Chapter 3

3 Vision Algorithms and Kalman Estimator

3.1 Introduction

Chapter 2 has described the overall SPHERES MOSR system. This chapter will describe the algorithms used in developing the software on this testbed. First, the reader is introduced to the Hough Transform and how it integrates into the MOSR system. Equations describing key fundamental relationships are shown and raw data from the algorithm are shown. The process is repeated for the astronomical photometry algorithm and raw data is shown. Key results and breakdowns are shown in both algorithms and the need for a Kalman estimator is revealed. Finally, the application of a Kalman filter is shown and the results are presented. Comparisons are made with the SPHERES global metrology system to gauge the accuracy of a single camera vision system.

3.2 Hough Transform

3.2.1 Brief Review

Revisiting the Hough Transform shown in Section 1.4.1, the parametric representation of the equation of a circle can be written in the form as given in Equation 11.

$$(x-a)^2 + (y-b)^2 = c^2 \tag{11}$$

For any arbitrary point (x_i, y_i), this can be transformed into a surface in three dimensional space with the dimensions being (a,b,c) as defined in Equation 12:

$$(x_i-a)^2 + (y_i-b)^2 = c^2 \tag{12}$$

Each sample point (x_i, y_i) is now converted into a right circular cone in the (a,b,c) parameter space. If multiple cones intersect at one point (a_o,b_o,c_o), then these sample points now exist on a circle defined but the parameters a_o,b_o, and c_o.

The above process is repeated over all the sample points in an image and votes are gathered for each set of parameters (a_o,b_o,c_o). This method is of course extremely computationally intensive and has a memory requirement on the order of O(m n c) where mn is the total number of sample points(i.e. image size) and c is the number of possible radii. This memory requirement is needed for the accumulator array where the columns of the accumulator array are the possible radii in the image and each row is the unique set of (a,b) points for the coordinate of the centers. However, the computational burden of this method can be significantly lower if the radius c is known a priori. This is reduces the 2-D accumulator array into a single column vector for a specific c_o or range of radii ($c_i - c_j$). Thus, if the radius is known a priori, the algorithm's memory requirement reduces to O(mn) time, since fewer cells in the accumulator array have to be updated for each sample point.

3.2.2 The role of Canny edge detection

Based on the above method, a similar algorithm based on Yuen et. al [27] is used. Here the key difference is an edge detector such as that proposed by Canny[28]is used. This eliminates all the sample points in the image that do not lie on an edge.

Canny edge detection[28] works by computing derivatives in the horizontal and vertical direction. These are performed on slightly blurred images in order to discard any discontinuities due to CCD sensor noise. These derivatives are the change in pixel brightness along those directions. Equation 13 shows the derivative G(also known as the edge gradient) which the magnitude of the G_x and G_y.

$$G = \sqrt{G_x^2 + G_y^2} \tag{13}$$

One can also calculate an edge direction θ which is given as:

$$\theta = tan^{-1}\left(\frac{G_x}{G_y}\right) \tag{14}$$

Once these gradient magnitudes are found, a localized search of the area is done to find if the gradient magnitude is a local maximum. If it indeed lies in a local maximum, the sample point lies on an edge and is kept. Any other sample points in the localized search area are discarded.

This process is repeated until all the edge points are located. The edge points are then set to max brightness (max pixel value in a monochrome camera). The points that do not lie on an edge are set to pixel level 0. Figure 18 shows the original blurred image followed by an image with Canny edge detection performed.

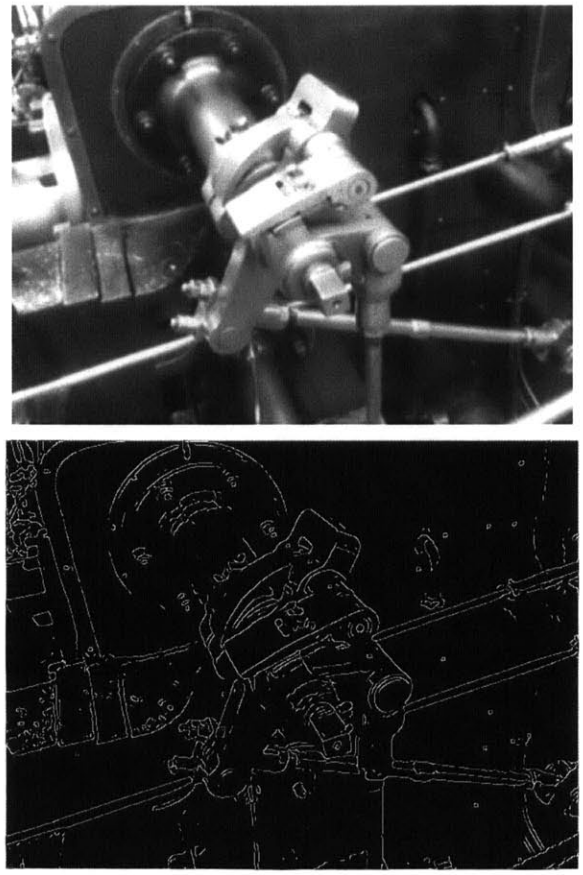

Figure 18: Edge Detection using the Canny edge Detection algorithm

3.2.3 The Hough Transform applied to MOSR

Yuen et. al[27] proposed using an edge detection algorithm to reduce the storage and computational demands of circle finding. Using edge detection, the problem of circle finding can now be decomposed into two stages consisting of a 2D Hough Transform to find the circle centers, followed by a 1D Hough Transform to determine radii. Since the center of the circle must lie along the gradient direction of each edge point on the circle, then the common intersection point of these gradients identifies the center of the circle. A 2D array is used to accumulate the center finding transform and candidate centers are identified by local peak detection. This can be viewed as an integration along the radius axis of all values of the Hough Transform at a single value of (a,b). The second stage of the method uses the center of the first stage to construct histogram values of possible radii values. The radius value with the most number of edge points lying on it is considered the radius of the circle.

A potential problem with this modified Hough Transform method is that any 3D information from the image in the edge detection process is lost. For example, any circles that do not lie parallel to the camera image plane are completely lost since their gradients might not lie in the vertical and horizontal directions. This problem is not a concern for MOSR since the OS is assumed to be a "few meters" away from the capture mechanism. The cross-section of a spherical object will always remain parallel to the image plane.

Using the method described above, the algorithm is applied to the MOSR OS in steps:

Step 1 consists of gathering a raw monochrome image as shown in Figure 19.

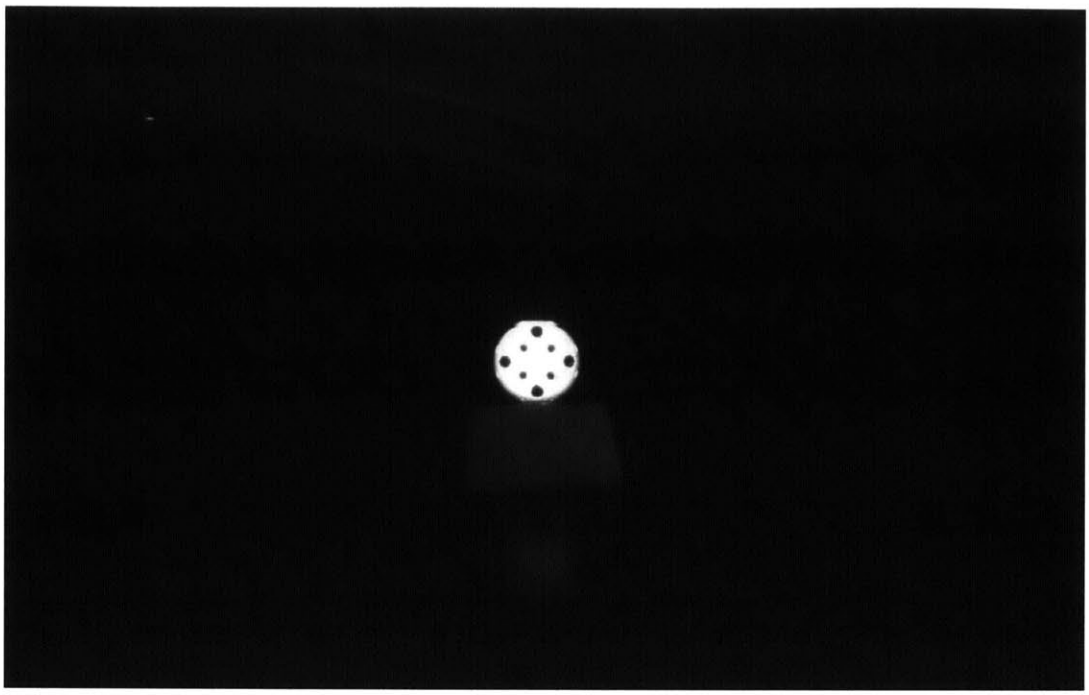

Figure 19: Raw Image of OS shell on the flat floor

It is important to note that the air bearing support structure as shown in Figure 15 is covered by grey foam so as not to interfere with the algorithms accuracy. Spurious circles may be detected in the air bearing mechanism. In this image, the OS is 1.5 m away from the camera.

Step 2 consists of a simple thresholding algorithm which sets any pixel brighter than a certain threshold to full 'white' (pixel value 255 in an 8-bit CCD sensor) and conversely, sets any pixel dimmer than a certain threshold to full 'black' (pixel value 0). This can be seen in Figure 20.

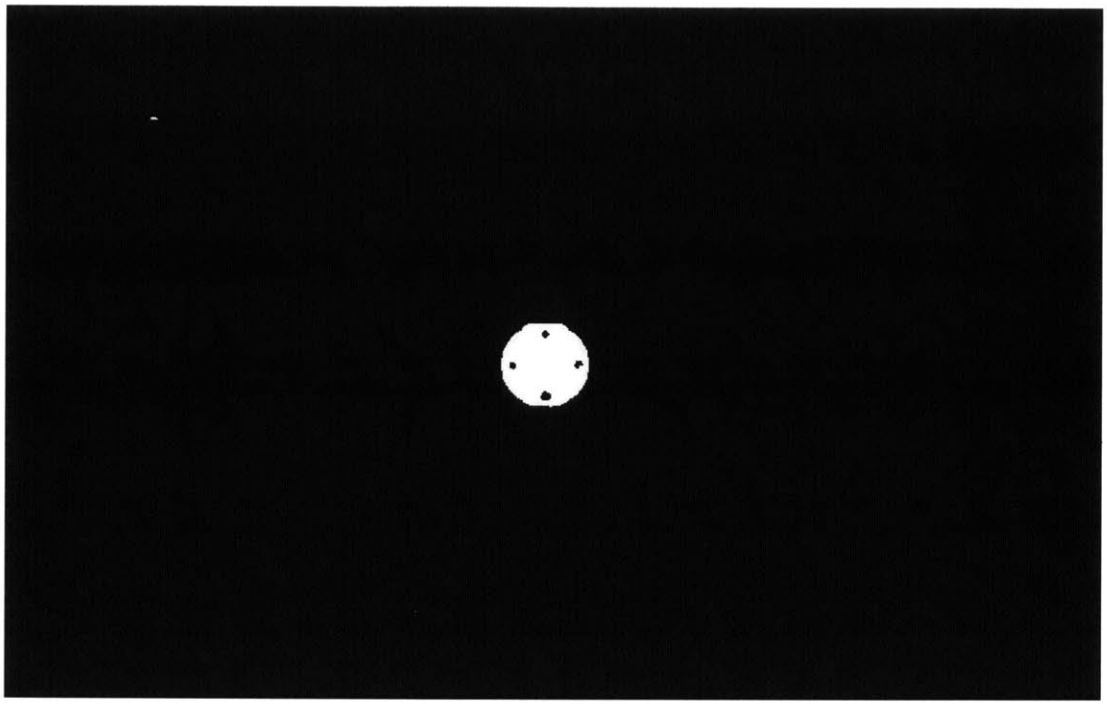

Figure 20: Thresholded OS image

It is important to note that in this image; most of the room plumbing in the background has now disappeared. A bright speck in the top left corner passed the thresholding algorithm but this should not be of concern since it is not circular or of the right diameter in shape.

Step 3 consists of the image then being processed by a Canny edge detector as shown in Figure 21.

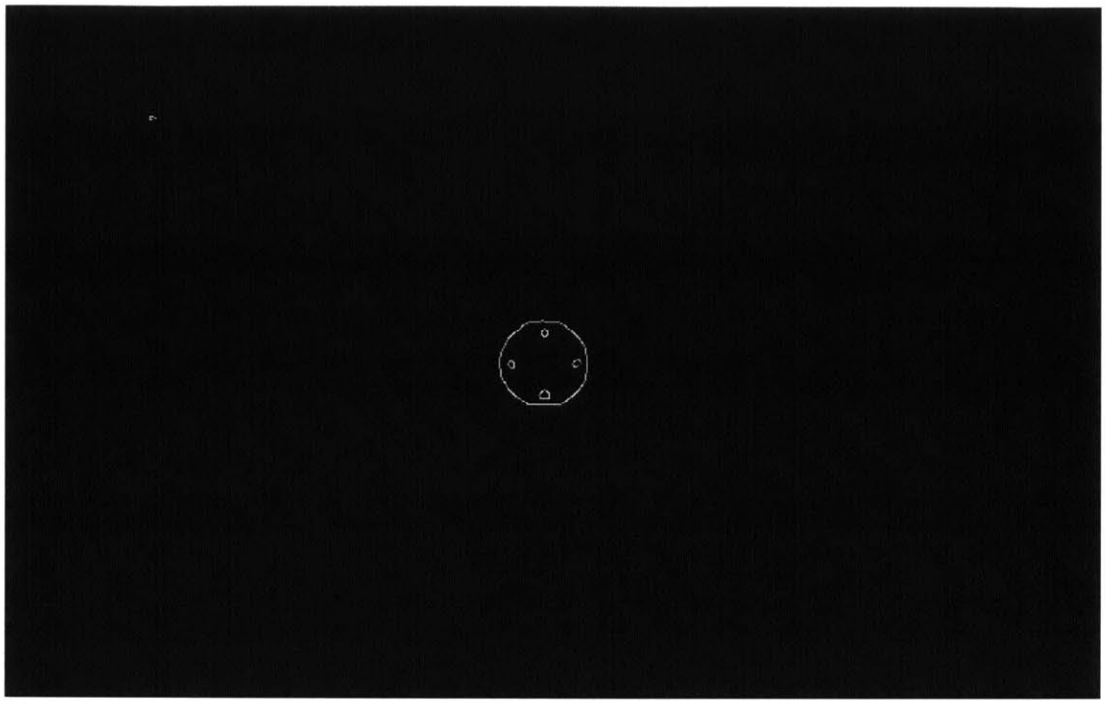

Figure 21: OS image after Canny Edge detection

Again, it is important to note that the speck in the top left still exists (albeit with only its edges intact). The Canny edge detector has now removed all the pixels internal to the OS and kept the outer edge intact.

It can now be seen that a large non-zero 752 by 480 pixel image (total pixels: 360,960) is reduced to an image with only several hundred non-zero pixels. This offloads the computation done by the Hough Transform in Step 4.

Step 4 consists of running the image through the Hough transform algorithm with an initial guess for a diameter. As mentioned in Section 1.4.1, this initial estimate for the diameter will be given by another ranging algorithm (possibly using the long range radio beacon) in the real MSR mission.

The result after Step 4 can be seen in Figure 22.

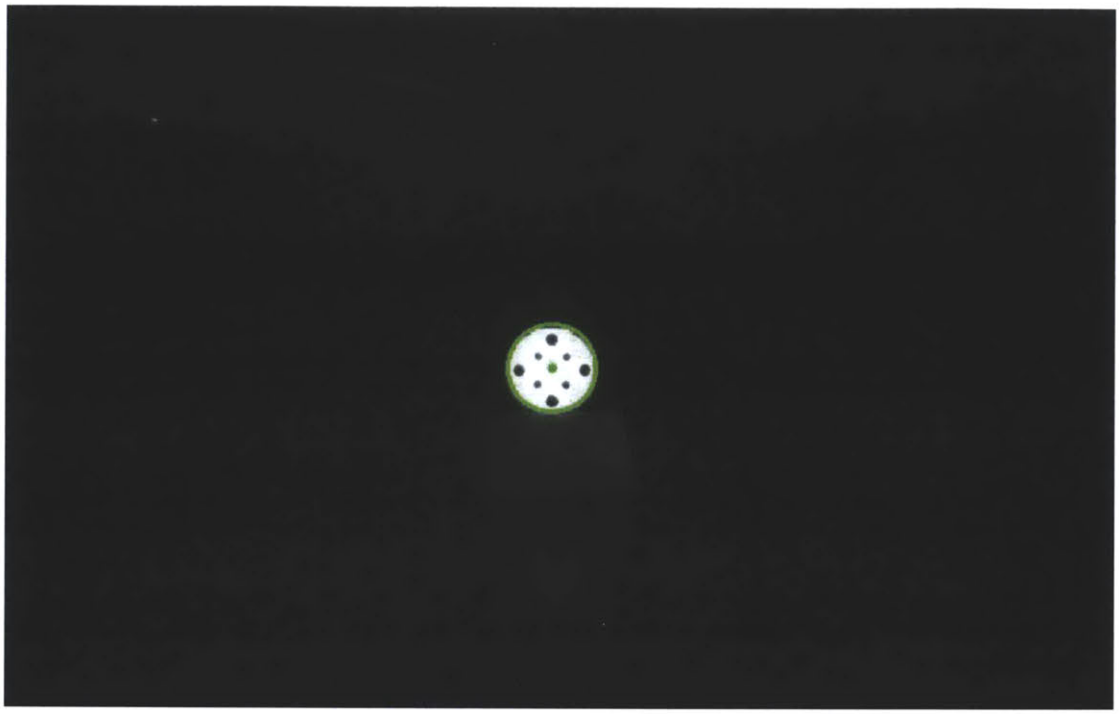

Figure 22: Completed Hough Transform OS image

In the interest of clarity, the solution is superimposed on the original non-processed image. The green circle signifies the radius that the transform computed, and green dot in the center signifies the center of that circle. It is important to note that the smaller circles were not recognized since the radii of these circles were not within acceptable bounds (within ± 10% of the initial estimate). The new circle radius is then fed into the following iteration as the new initial 'guess'. Here, it is assumed that the relative motion of the OS perpendicular to the image plane is not high enough to exceed the 10% tolerance mentioned above.

Step 1-4 can be reapplied to a partially shielded OS as shown in Figure 23.

Figure 23: Partially occluded OS image processed with the Hough Transform

The OS shown above was partially occluded by rotating the dark side of the OS to face the camera (as shown in Figure 15).

In Figure 23, the circle radius is not as accurate. This is mainly due to the lower number of edge points in the image. These circular edge points now generate fewer votes thereby decreasing the accuracy of the algorithm.

3.2.4 Absolute Position estimation using the Hough Transform

As shown in Figure 22 and Figure 23, the Hough Transform generates a robust radius and center measurement in pixels and pixel positions respectively. This section will cover the transformation of the pixel positions and values to absolute position states (in meters).

Figure 24 shows the geometry of the OS in the camera reference frame.

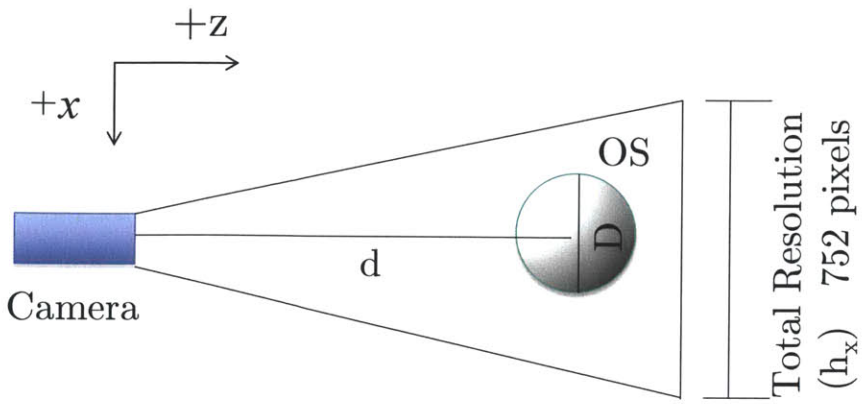

Figure 24: Camera reference frame perpendicular to image plane

In Figure 24, the depth 'd' of the OS in the camera reference frame can be derived from basic trigonometric principles and is presented in Equation 15. The reference axes directions are also given for clarity.

$$d = z = \frac{D\, h_x}{4 \tan\left(\frac{FOV_x}{2}\right) r_p} \tag{15}$$

Here, FOV_x is the field of view in degrees of the camera, r_p is the estimated radius in pixels of the OS, D is the actual diameter of the OS shell in meters(0.277 m) and h_x is total resolution of the image in that direction (752 pixels).

The same calculation can be done in the perpendicular image direction using total resolution h_y and Field of View FOV_y. An identical depth z is found.

Looking at the center location of the OS, Equation 16 and 17 can now be used to calculate the x and y coordinate of the OS position using α and β (azimuth and elevation angles) with respect to the center of the image plane(as the origin).

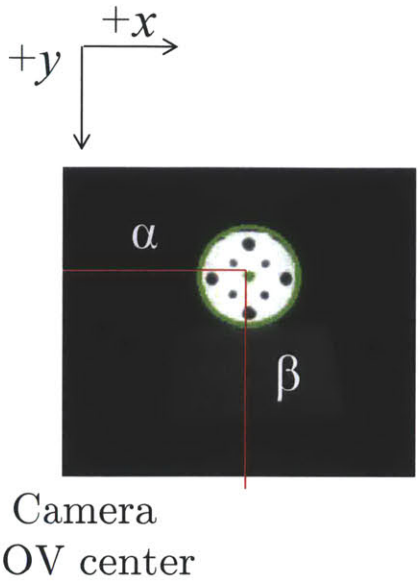

Figure 25: Camera Reference frame along image plane

Using Figure 25, x and y locations can be derived using basic trigonometric principles shown in Equations 16 and 17.

$$x = z\,tan\left(\frac{x_p\,FOV_x}{h_x}\right) \tag{16}$$

$$y = z\,tan\left(\frac{y_p\,FOV_y}{h_y}\right) \tag{17}$$

Once again, z is the depth estimate computed from Equation 15, FOV_x and FOV_y are the Fields Of View in the x and y directions, x_p and y_p are the pixel positions of the center of the OS with respect to a origin centered at the FOV center and h_x and h_y are the total pixel resolutions in the x and y directions respectively. It is noted that z appears in both equations and so the depth measurement z is crucially important in state estimation.

The next step is to plot the time history of the OS depth z in this camera reference frame. In this case, the OS is completely stationary.

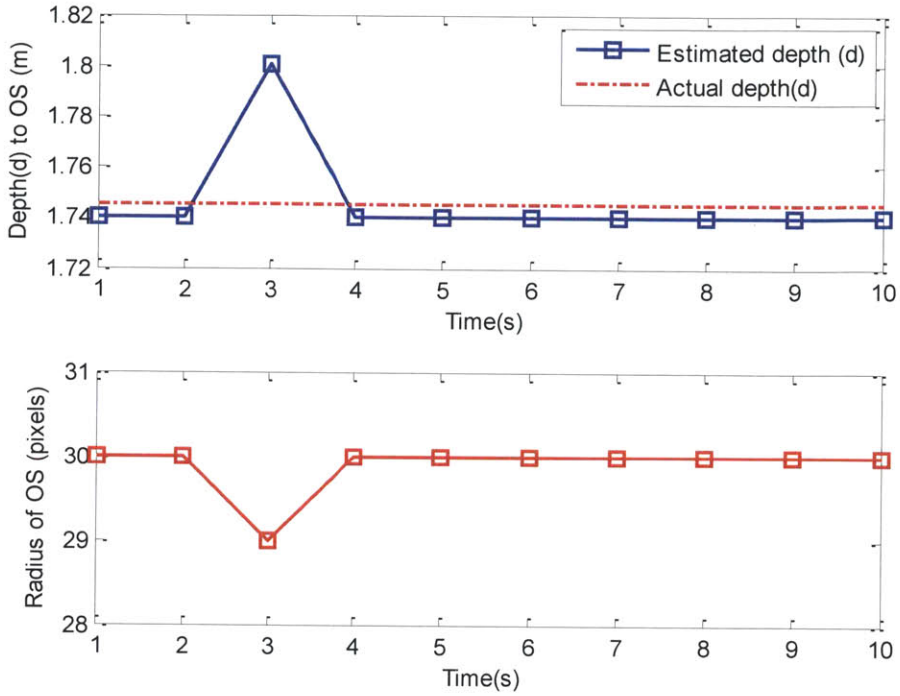

Figure 26: Time History plot of Hough Transform for a stationary OS in the 'near' field

In Figure 26, the time history shows a relatively low noise level for a stationary OS in the 'near' field (actual depth is 1.75m). In ten samples, the noise level remains between ±1 pixel which is nearly ideal for the Hough transform[29].

In Figure 27, the same time history plot is taken for the 'far' field.

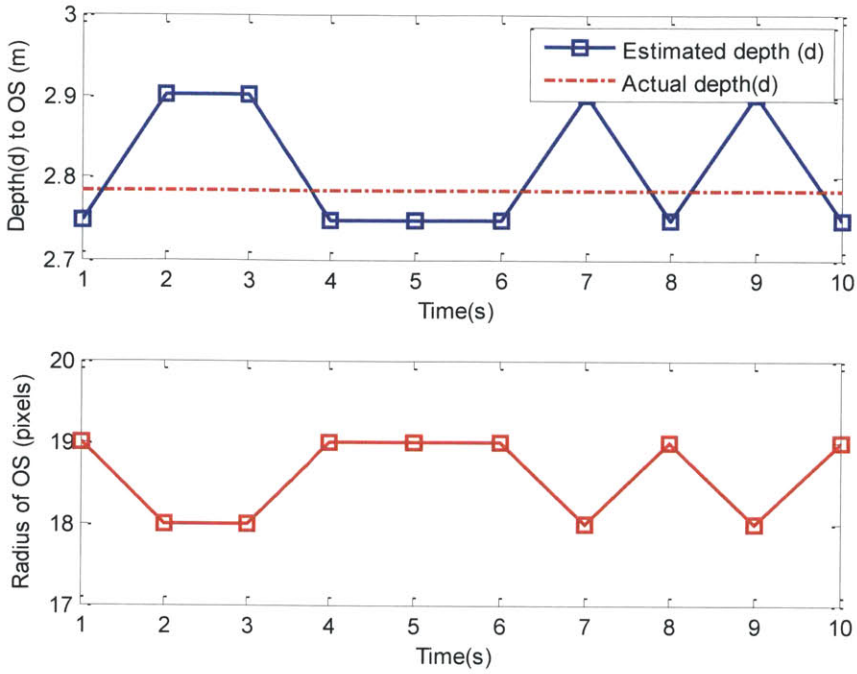

Figure 27: Time History plot of Hough Transform for a stationary OS in the 'far' field

In Figure 27, the noise level is higher than in the 'near' field case. Here the OS is placed at 2.78 m. This is due to the larger thresholding errors and fewer edge points on the circle which accumulate fewer votes for a given radii. Again, a pixel error of ±1 pixel is expected for the Hough transform.

However, a very curious relationship can be seen. For a change in 1 pixel of radius estimate, the resulting difference in depth is much higher in the 'far' field case (~15 cm) than in the 'near' field case (~6 cm). This relationship can be seen more easily when Equation 15 is plotted. Here there is a clear non-linear (inverse) relationship between r_p(pixel radius) and depth z.

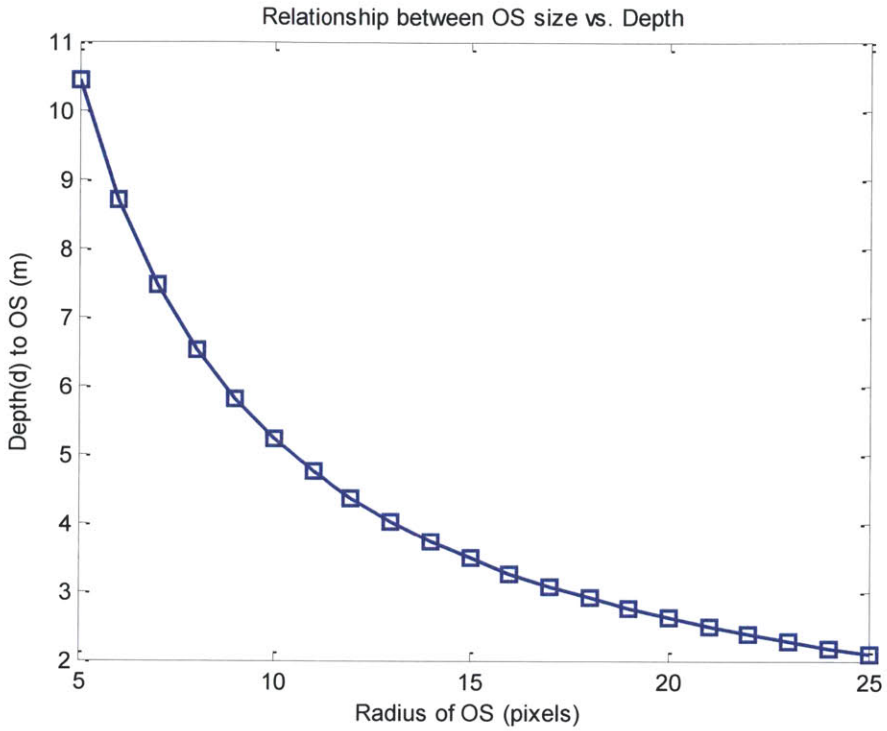

Figure 28: Relationship between OS size vs. depth

In Figure 28, the blue boxes indicate discrete pixel values whose depths can be resolved. As the depth decreases, the radius of the OS r_p in pixels increases, and the difference between two successive pixel estimates(i.e. the vertical space between each consecutive pixel estimate) decreases. This relationship is what is being seen in Figure 26 and Figure 27.

3.2.5 Conclusion

As a consequence of these results, it is evident that the Hough transform algorithm can only generate depths at discrete pixel values. Therefore, a method which does not rely on discrete pixel estimates needs to be used in order to increase fidelity in the depth measurement. This algorithm is based on astronomical photometry.

3.3 Astronomical Photometry

3.3.1 Review

From Section 1.5.1, the apparent brightness or luminosity (F) of a point source in an image is a simple sum of pixel brightness $B(x_i,y_i)$ of all the pixels across the point source(with dimensions m, n pixels in each direction) as shown in Equation 18.

$$F_{point} = \sum_{i=0,j=0}^{m,n} B(x_i, y_i) \tag{18}$$

The flux density or apparent brightness of point source can also be given by the flux density relationship in Equation 19:

$$F_{point} = \frac{L}{z^2} \tag{19}$$

Here, L is the true luminosity of the source and z is the distance away from the source.

3.3.2 Astronomical photometry for MOSR

In this application, the target (OS) is not a point source. For Equation 5 to apply, each pixel is regarded as a point source. Therefore, the apparent luminosity F_{target} is simply the sum of the apparent luminosity over all points on the target as shown in Equation 20.

$$F_{target} = \sum F_{point\ i} \tag{20}$$

However, the total number of pixels on the OS increases inverse-quadratically with respect to depth. Figure 29 presents this relationship graphically.

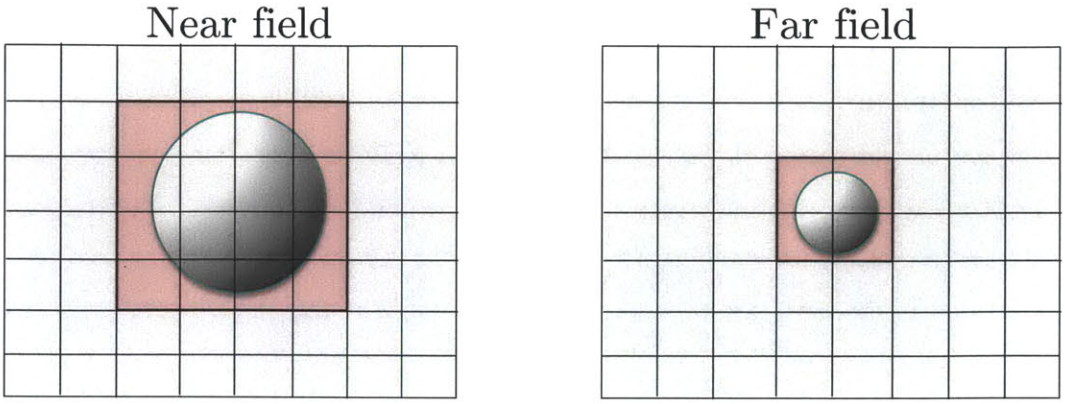

Figure 29: Number of Pixels on target scales with range

$$Number\ of\ pixels\ across\ OS\ \propto \frac{1}{z^2} \tag{21}$$

Combining equations 20 and 21, the relationship between total apparent brightness F_{target} and depth z is now an inverse quartic relationship with C_1 being a new constant of proportionality that needs to be solved a priori.

$$F_{target} = \frac{C_1}{z^4} \tag{22}$$

Equation 22 now presents an inverse quartic relationship between F_{target} and depth z. This increased fidelity to the z measurement can now be seen in Equation 23.

$$z = \sqrt[4]{\frac{C_1}{F_{target}}} \tag{23}$$

Equation 23 is a fourth root function and can be converted to the log scale as seen in Equation 24.

$$\log z = a \log(C_1) + a\ log\left(F_{target}\right) \tag{24}$$

A linear curve fit can be done between z and F_{target} to calculate the values of 'a' and C_1.

Figure 30 shows a log plot with a linear fit to Equation 24. In this plot, z is known using alternate methods (a laser range finder) and F_{target} is computed by a simple summation of the pixel values as shown in Equation 20.

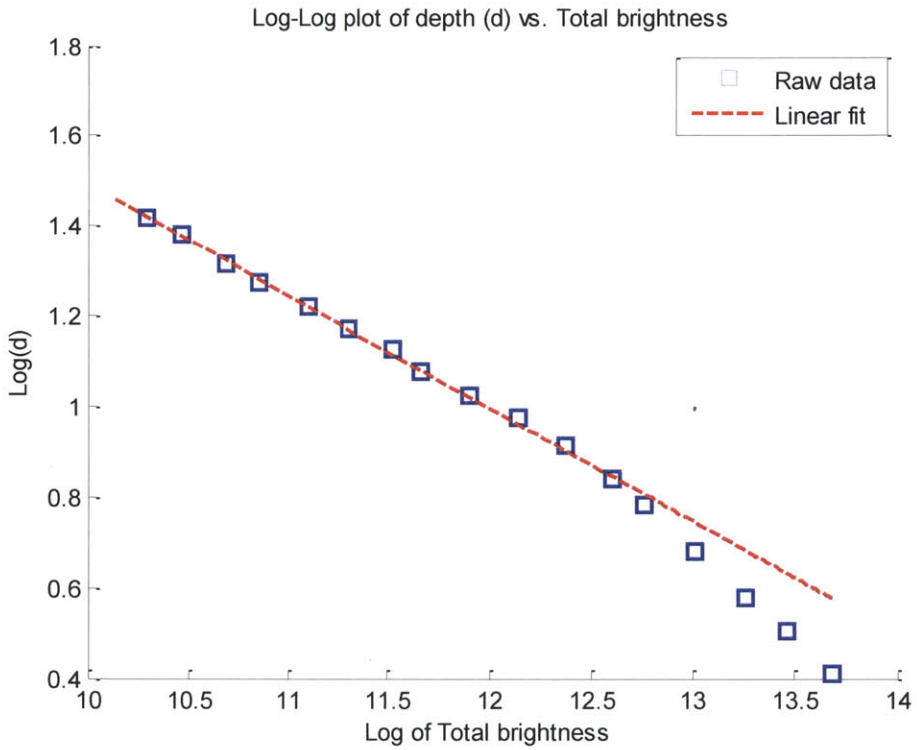

Figure 30: Linear fit plot for astronomical photometry

In Figure 30, the linear fit coefficients are computed as follows:

$$-(a)^{-1} = b = 4.01 \qquad (25)$$
$$C_1 = 8.257E7$$

For this linear fit, b is the power on z and is expected to be 4. However, the linear fit does not correlate well for OS locations closer than 2 m (these points have been

excluded from the fit). This discrepancy can be seen clearly when the plot is converted back to non-log axes as presented in Figure 31.

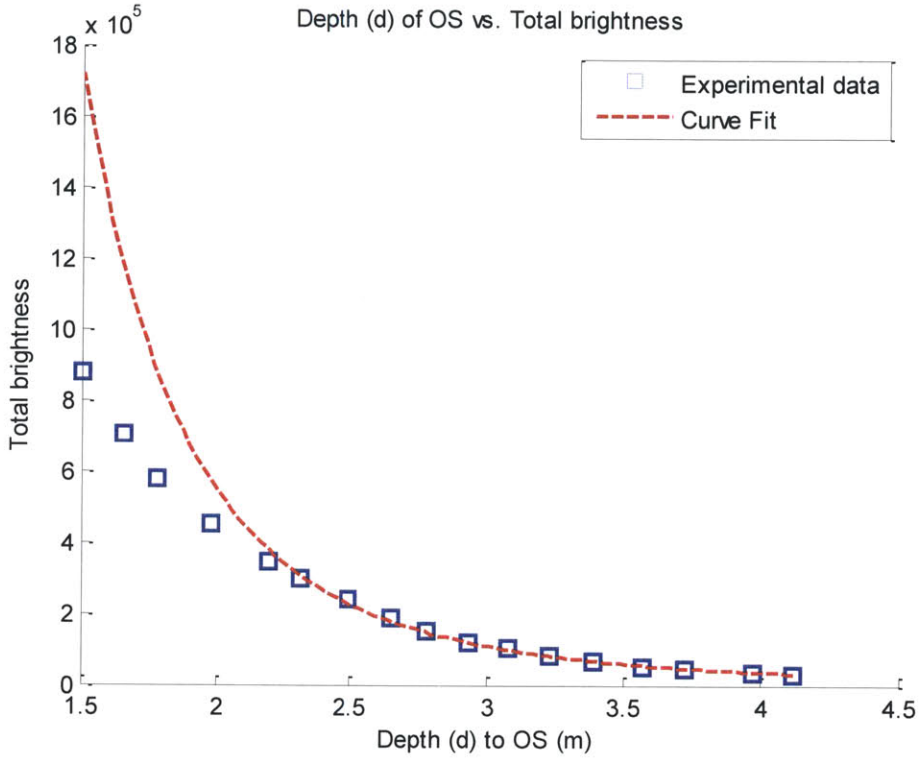

Figure 31: Inverse Quartic relationship of astronomical photometry

The discrepancy in the result presented in Figure 31 is due to CCD sensor saturation. As the OS approaches the camera, the pixel values cannot get higher than 255(the max pixel value for an 8-bit sensor) and the inverse quartic relationship starts to break down. However, the performance of this algorithm is fairly robust in the far field.

3.3.3 Conclusion

The photometry algorithm described in this section features strong performance in the far field and relatively poor performance in the near field. On the flip side, the Hough transform begins to breakdown in the far field, and it is this complementary

relationship between the two algorithms that can be exploited by an integrated filter such as the Kalman Filter.

3.4 Kalman Estimator

3.4.1 Review

The discrete time Kalman Filter equations are reviewed here. For further detail, the reader is advised to consult numerous textbooks such [30] and [31].

The state estimate vector is x_k and the measurements are z_k. If there are no known external disturbances (as is assumed in this case), the Kalman Filtering equation is simply given by Equation 26.

$$x_k = x_k^- + K_k(z_k - Hx_k^-) \tag{26}$$

In Equation 26, x_k^- (the predicted state estimate vector) can be rewritten as shown in Equation 27. In this equation, A is the system model matrix(which is known and is constant and x_{k-1} is the state estimate from the previous iteration.

$$x_k^- = Ax_{k-1} \tag{27}$$

From Equation 28, the Kalman gain, K_k is calculated in the manner described in Equation 28 and Equation 29.

$$P_k^- = AP_{k-1}A^T + Q \tag{28}$$

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \tag{29}$$

Equation 28 and 29 introduces four new terms: H(the measurement model matrix), R(the covariance matrix of the measurement noise), P_k^-(the predicted error covariance matrix), and Q (the process noise matrix).

Finally, the error covariance matrix P_k can be updated using Equation 30:

$$P_k = P_k^- - K_k H P_k^- \tag{30}$$

The accuracy of the Kalman Filter heavily depends on the matrices that define the state model, the measurement model and the noise model. These matrices (A, Q, R and H) are vitally important and are application specific. They are discussed in Section 3.4.2.

3.4.2 The Use of the Kalman Filter in MOSR

The Kalman Filter is used to integrate the measurements described in Sections 3.2.4 and 3.3.2. The measurement vector z_k consists of position estimate as given in Equation 31.

$$z_k = [x_k, y_k, z_k]^T \tag{31}$$

The position estimates described here are the same for both algorithms, with the key difference being that z is different for both algorithms (and so their corresponding x and y estimates also differ from each other).

The state estimate x_k is a 9x1 column vector with position, velocity and accelerations shown in Equation 32. While position and velocity estimates are the only required estimates for the controller described in Chapter 4, accelerations are included to take into account process noise. Since the Kalman Filter process noise matrix Q assumes zero mean, Gaussian noise, the acceleration terms are required to account for variations in velocity. This will become more apparent in Equation 35.

$$x_k = [x_k, \dot{x}_k, \ddot{x}_k, y_k, \dot{y}_k, \ddot{y}_k, z_k, \dot{z}_k, \ddot{z}_k]^T \tag{32}$$

The state model A is assumed to be simply based on Newton's second law. Here A is a 3x3 vector consisting of 3 3x3 terms along the diagonal resulting in a 9x9 system as shown in Equation 33 and 34.

$$A_o = \begin{bmatrix} 1 & T_s & 0.5T_s^2 \\ 0 & 1 & T_s \\ 0 & 0 & 1 \end{bmatrix} \tag{33}$$

$$A = \begin{bmatrix} A_0 & \dots 0 & \dots 0 \\ 0 \dots & A_0 & \dots 0 \\ 0 \dots & 0 \dots & A_o \end{bmatrix} \tag{34}$$

In Equation 33, T_s is the length of the time step between measurements; in this case, it is 1 second.

As mentioned previously, the motion of the OS is assumed to have zero-mean acceleration along the entire maneuver. Therefore, the non-discretized form of the process noise matrix is presented in Equation 35.

$$Q = \varphi \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{35}$$

Discretizing Q from Equation 35 using the convolution integral results in Equation 36 for Q_o.

The process noise matrix Q_k is a 9x9 matrix which consists of 3 3x3 Q_o matrices along the diagonal.

$$Q_0 = \varphi \begin{bmatrix} \frac{T_s^5}{20} & \frac{T_s^4}{8} & \frac{T_s^3}{6} \\ \frac{T_s^4}{8} & \frac{T_s^3}{3} & \frac{T_s^2}{2} \\ \frac{T_s^3}{6} & \frac{T_s^2}{2} & T_s \end{bmatrix} \tag{36}$$

$$Q_k = \begin{bmatrix} Q_0 & \dots 0 & \dots 0 \\ 0 \dots & Q_0 & \dots 0 \\ 0 \dots & 0 \dots & Q_0 \end{bmatrix} \tag{37}$$

In Equation 35 and 36, φ is the spectral noise density. This is a constant and is tuned via trial and error when testing the Kalman Filter.

H, the measurement matrix is the mapping matrix from z_k to x_k. Since the z_k terms exist in x_k without any numeric calculations, the H matrix is simply of the form shown in Equation 38.

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{38}$$

The measurement noise matrix R is tuned via trial and error (using the constant R_0) and has the form shown in Equation 39.

$$R = R_0 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{39}$$

Given the above constant matrices, A, Q, H and R, the Kalman filter can be integrated in MATLAB using Equations 26 to 30.

3.5 Kalman Filter Results

With proper tuning of the parameters φ and R_0, Figure 33 to Figure 35 are the results of the filter.

It is important to note that comparisons are made to Global Metrology system (used as ground truth). The camera reference frame is converted to the Global metrology reference frame as shown in Figure 32.

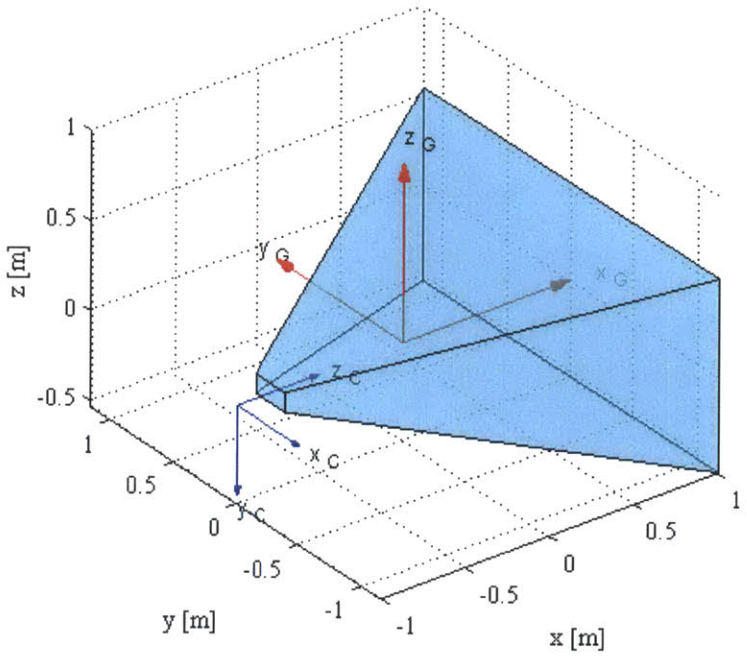

Figure 32: Comparison of Reference frames (Global Metrology and Camera)

Figure 33 shows the comparison between the raw measurement and the filtered state estimate.

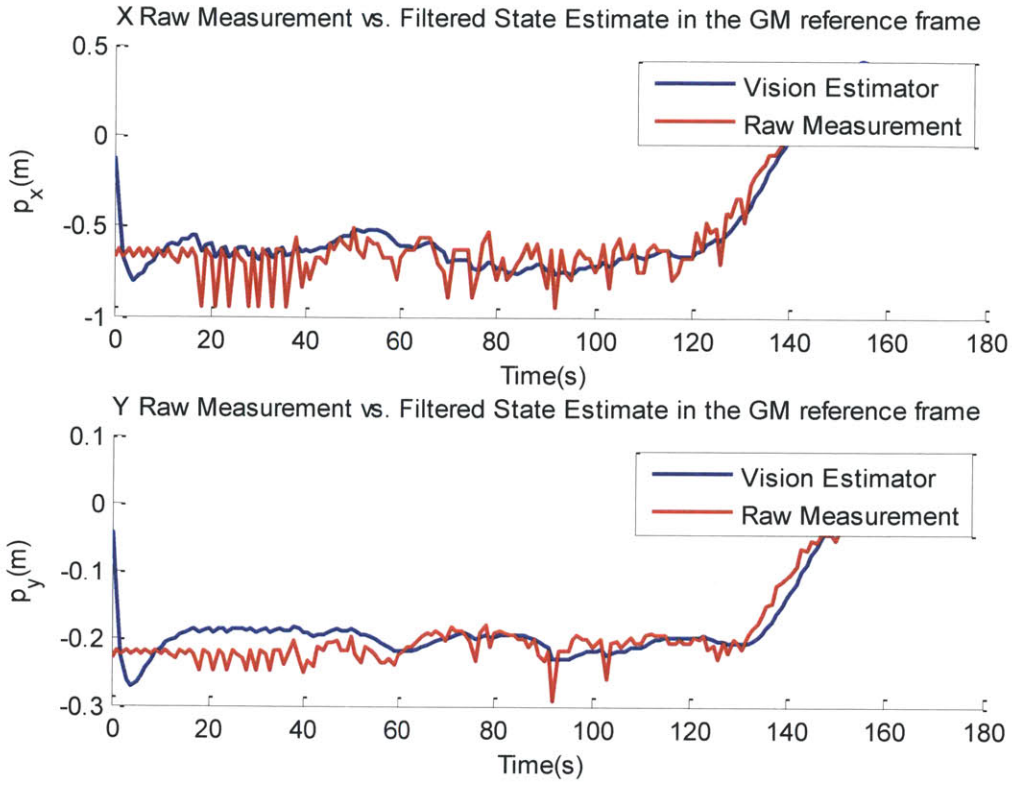

Figure 33: Vision Estimator Convergence vs. Raw measurement

The X and Y position estimate from the Kalman filter estimator data in Figure 27 shows a clear 'smoothing' capability. As mentioned in Section 3.2, there is significant noise at the beginning of the maneuver in the 'far' field and this is smoothed out by the Kalman Filter. Towards the end of the maneuver (when the OS is closest to the camera), the raw data is much less noisy as expected. The Kalman Filter works well in tracking the mean value of the measurement while smoothing the results as is needed in this application.

Figure 34 and Figure 35 compare the Kalman Filter state estimate generated by the vision system with the Global Metrology data from the SPHERES system. It is important to note here that the Global Metrology system had significantly more noise than is usually expected. This problem was diagnosed as being due to interference from the MOSR shell geometry. The MOSR shell blocks the infrared synch pulse from the SPHERES satellite to the external Global metrology system. A workaround was made

where the exterior back half of the MOSR shell was removed for the test. This workaround was only successful for a few tests.

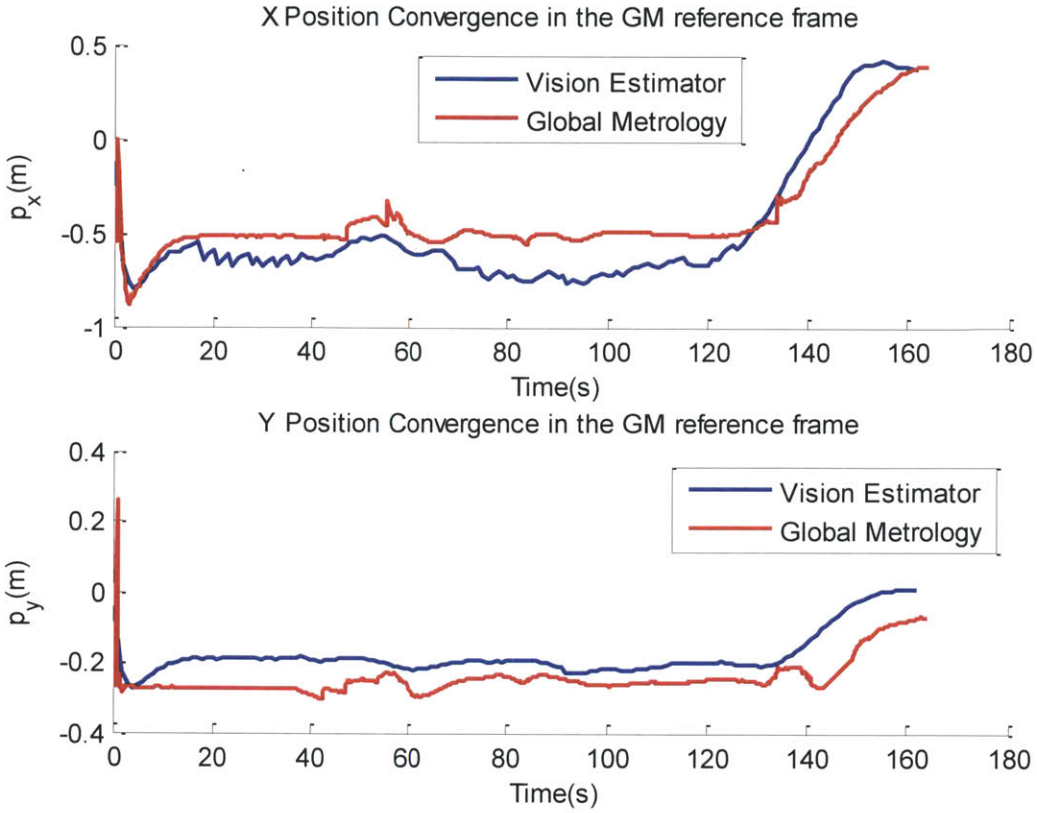

Figure 34: Position Convergence Comparison in the GM Reference frame

The X and Y Position data presented in Figure 34 show a bias that varies with time. It is conceivable that this bias maybe due to a small pixel error in the Hough Transform in the 'far' field. This 'bias' then gets propagated through the filter and slowly recovers as the raw measurement error decreases and the OS travels closer to the camera. This bias varies between 15 cm and 5 cm.

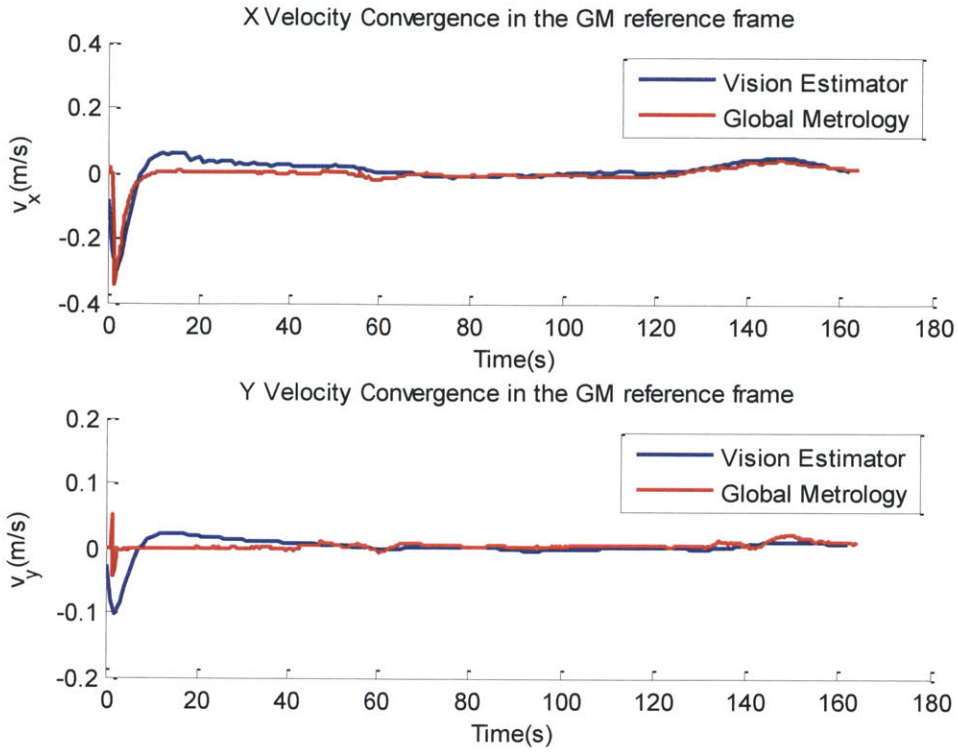

Figure 35: Velocity Convergence Comparison in the GM Reference frame

The X and Y Velocity data generated by the vision estimator in Figure 35 shows remarkable accuracy when compared the to the Global Metrology data. The velocity error is less than 1 cm/s which is more than acceptable for the purposes of docking and capture.

3.6 Conclusion

This chapter presents an approach for close proximity navigation and state estimation using a single camera system. The details of this approach (including both algorithms used in this approach) are described and experimental results using a laser range finder and the SPHERES Global Metrology system are discussed. From these results, it can be concluded that the system has upper bounds on accuracy of around 15 cm for position and 1 cm/s for velocity. While these accuracies are not ideal when compared to a stereoscopic system laid out by Tweddle [32], it is remarkable for a single camera system.

Chapter 4

4 Model Predictive Control

4.1 Overview

The rendezvous and docking scenario for the MOSR mission involves a chaser spacecraft with an on-board vision based navigation system and a capture mechanism system already described in Chapters 2 and 3. In order to capture the relative motion between the chaser spacecraft and the target, tests are performed on the Flat Floor facility (as shown in Figure 15). A fixed device with the vision navigation system placed on one side of the Flat Floor operative area and a SPHERES satellite is used to perform the rendezvous and capture maneuver. This rendezvous and capture maneuver is performed using a Model Predictive Controller as described in Section 1.6. In order to define how the controller works, different controller strategies can be considered.

4.2 Definition of the Control Strategy

There are two possible control strategies that can be considered:

1. In the first strategy, a planning algorithm is used to generate a (safe) reference state trajectory, starting from an initial estimated state of the chaser spacecraft and ending at a target location. A controller, such as an MPC-based one, could be used to track the reference trajectory.

2. In the second strategy, an MPC-based controller is used to both compute a reference trajectory at each time step and track it at the same time.

For this work, the second control strategy is adopted with the aim of testing the capability of MPC to compute and track a reference trajectory for the close-proximity phase of the rendezvous and capture maneuver.

4.3 Controller constraints

In the reversed roles model proposed in Chapter 2, it is important that the controller maintain certain constraints in order to correctly emulate the OS-Chaser mechanics. The constraints are as follows.

4.3.1 Field of view constraint

It is required that the chaser remains within the FOV cone of the vision system during the entire rendezvous maneuver. The FOV in the horizontal plane is 90 degrees and the FOV in the vertical plane is 65 degrees.

4.3.2 Limited control authority

The thruster system on the SPHERES satellite has a maximum control force than can be actuated. Each thruster on-board the SPHERES satellite can perform a max force of 0.112N. Taking into account the thruster configuration (two thrusters along each direction), the maximum force that can be applied in any direction is $F_{max} = 0.224$N.

The mass of the SPHERES is assumed constant and equal to the SPHERE and the air bearing support structure shown in Figure 15. The mass is 11 kg, and therefore the theoretical maximum acceleration is 0.02 m/s^2. The MPC algorithm assumes a Piece Wise constant control acceleration profile, with a control period of 1 second. Due to interference issues between the thruster and global metrology system, thrusters can only be fired during 0.2 sec of the 1 sec controller duty cycle.

4.3.3 Terminal constraints

Terminal constraints are applied to the relative dynamic state of the OS in the proximity of the target position for safety purposes and to guarantee certain conditions for nominal capture mechanism function.

The following terminal constraints are placed on the SPHERES MPC controller:

1. The relative velocity has to be less than 3 cm/s in the z direction (in the camera reference frame).
2. The relative velocity along the other two directions must be almost 0.

Figure 36 shows these constraints graphically.

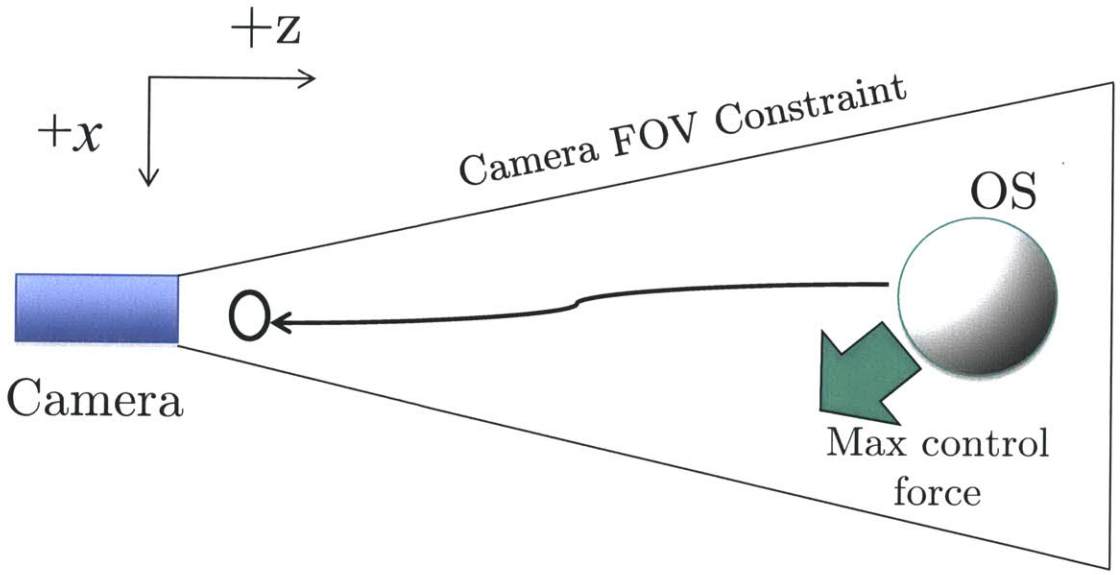

Figure 36: MPC Constraints

4.3.4 Attitude constraints

The attitude of the chaser spacecraft is maintained at a target orientation during close proximity operations. This is done so that the MOSR shell (described in Section 2.2.3) can be tested under different lighting conditions independently.

A quaternion based PD controller is used to regulate the attitude of the SPHERES satellite during the entire maneuver. MPC does not control the attitude of the OS.

4.4 MPC Software Architecture

Due to its high computational time when run on-board the SPHERES satellite, the MPC Engine cannot be used directly on-board SPHERES when a control frequency of 1Hz is required. In these situations, the MPC problem is solved using the laptop control station and the computed control forces are transmitted back to the SPHERES satellite for the actuation. To manage the time delay due to offline MPC computation and communication lag, a new state estimate for the SPHERES satellite is propagated with a constant assumed time delay. This time delay must be much less than 1 second in order for new control accelerations to be output to the SPHERES satellite before a new control cycle has begun.

Figure 37 shows the layout of the MPC Communications framework. The communication between the offline MPC Engine and the SPHERES satellite takes place using the regular SPHERES communication protocol.

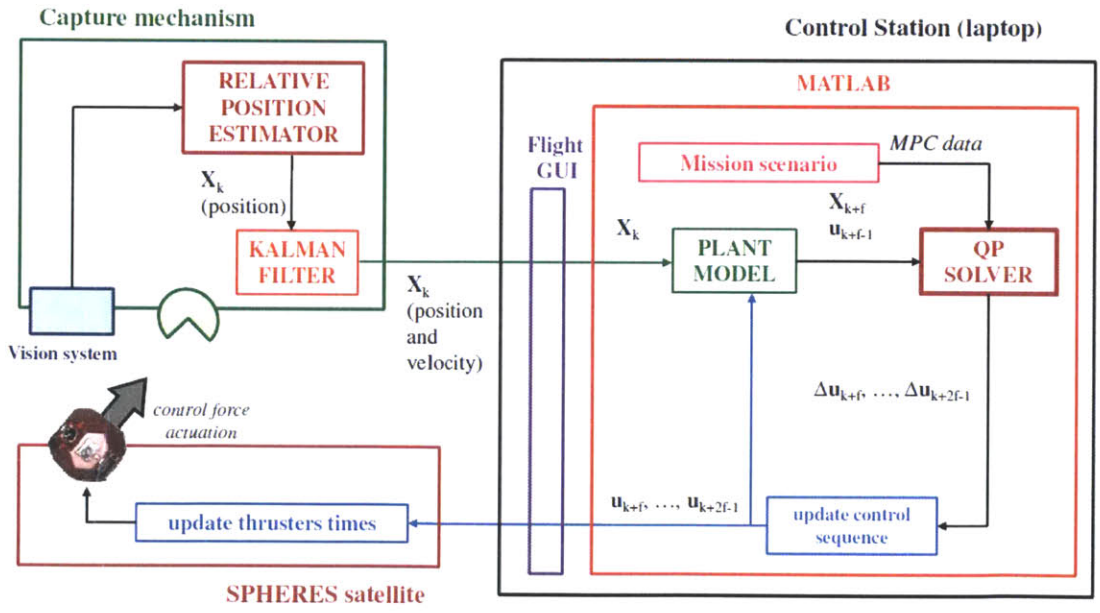

Figure 37: Online Communication Framework with MPC

4.5 Definition of the Reference frames

The following Reference frames are defined:

1. Camera Reference Frame: x and y axes are in the camera image plane, z axis along the camera focal length axis. This is identical to the camera reference frame defined in Section 3.5.

2. Global Metrology reference frame: This is the reference frame used by the SPHERES ultrasound Global Metrology system.

Figure 38 shows these reference frames in a graphical format. The teal cone is FOV constraint as it is applied to the MPC Controller.

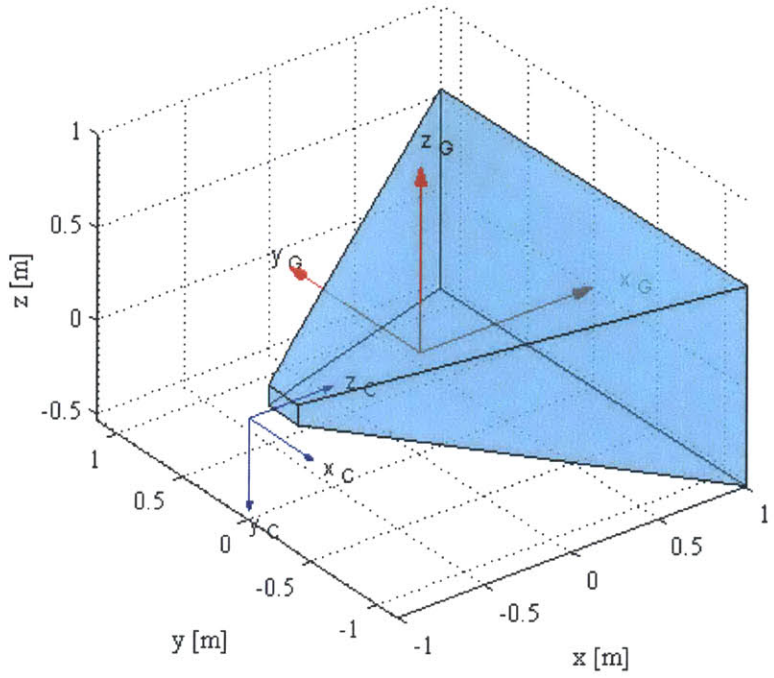

Figure 38: Camera and Global Metrology reference frames

4.6 MPC Results

For the purposes of this thesis, the exact mechanisms of the MPC controller are not shown. The reader is advised to refer to [19] whose author is a collaborator on this project.

4.6.1 Offline MPC time delay calculation

In order to check the viability of offloading the MPC Engine to MATLAB, a test was done to calculate the time delay between the SPHERES satellite requesting a new control action from the offline MPC Engine and the accurate delivery of the new control action back the SPHERES satellite. Figure 39 shows this result:

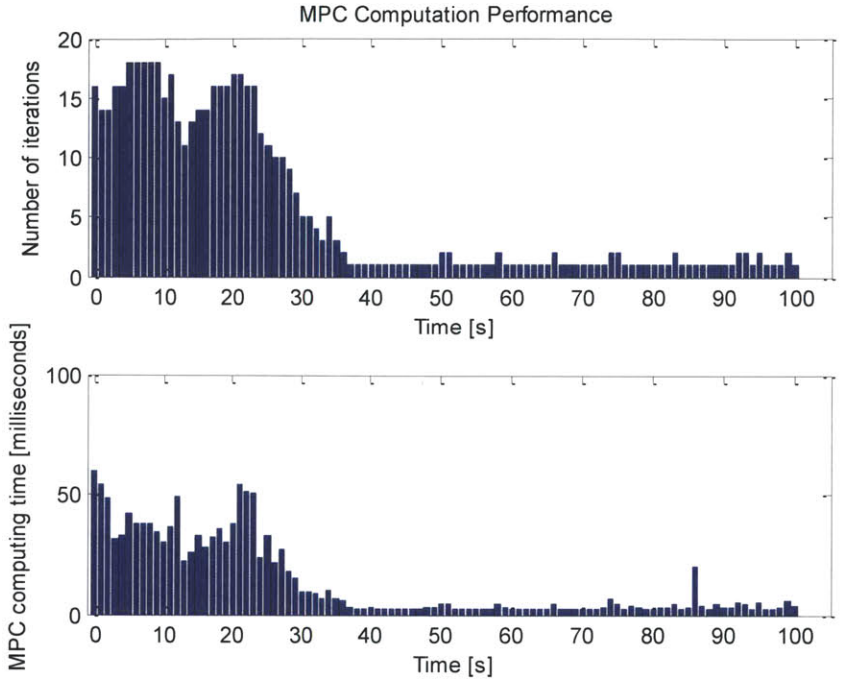

Figure 39: Time Delay computation for MPC Computation

Figure 39 presents the time delay and number of optimizer iterations required to generate a new MPC control action. The time delay is much lower than the required 1 second (~50 milliseconds) and this proves the viability of offloading the MPC engine to the control station.

4.6.2 MPC results in the SPHERES simulator

To ensure proper code functionality, the MPC code was run in the SPHERES MATLAB simulator. The MPC engine was compared with a well-tuned PD controller (having the same settling time as the MPC controller) and the same PD gains were used

in the test using the SPHERES hardware. Refer to Appendix A for more information of the simulation results.

4.6.3 MPC results using Global Metrology on SPHERES hardware

Figure 40 to Figure 42 show the MPC tests on the flat floor using the Global Metrology for state estimation. Comparisons are made to the SPHERES simulator results.

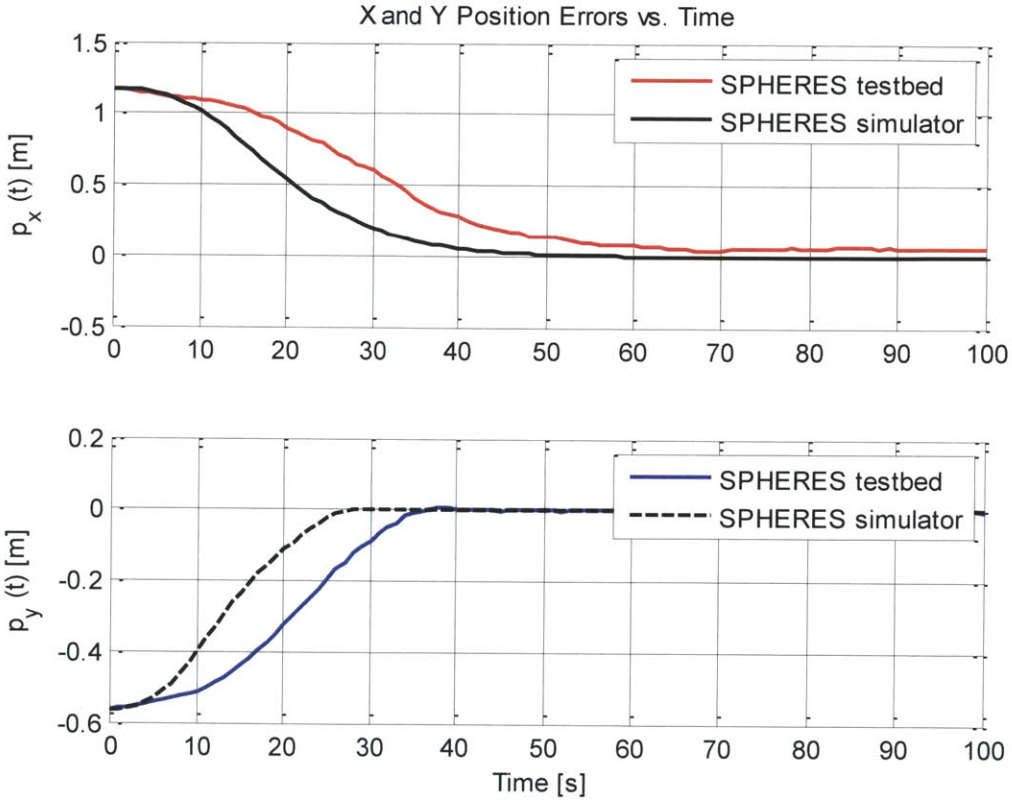

Figure 40: X and Y Position Errors for MPC engine

In Figure 40, the position errors are higher than the simulation. It is conceivable that the flat floor air bearing setup used for this test is not frictionless as the simulator assumes. This build-up of position errors leads to a build-up of control accelerations as shown in Figure 42. As the test progresses, the position errors tend to decay and this behavior matches the simulator results. A non-zero X-position error is seen towards the trailing edge of the test. This is possibly due to excessive friction in one of the air pucks

that the SPHERES satellite floats on for this 2D test. This is routinely seen on SPHERES tests on the flat floor and the error is not unexpected.

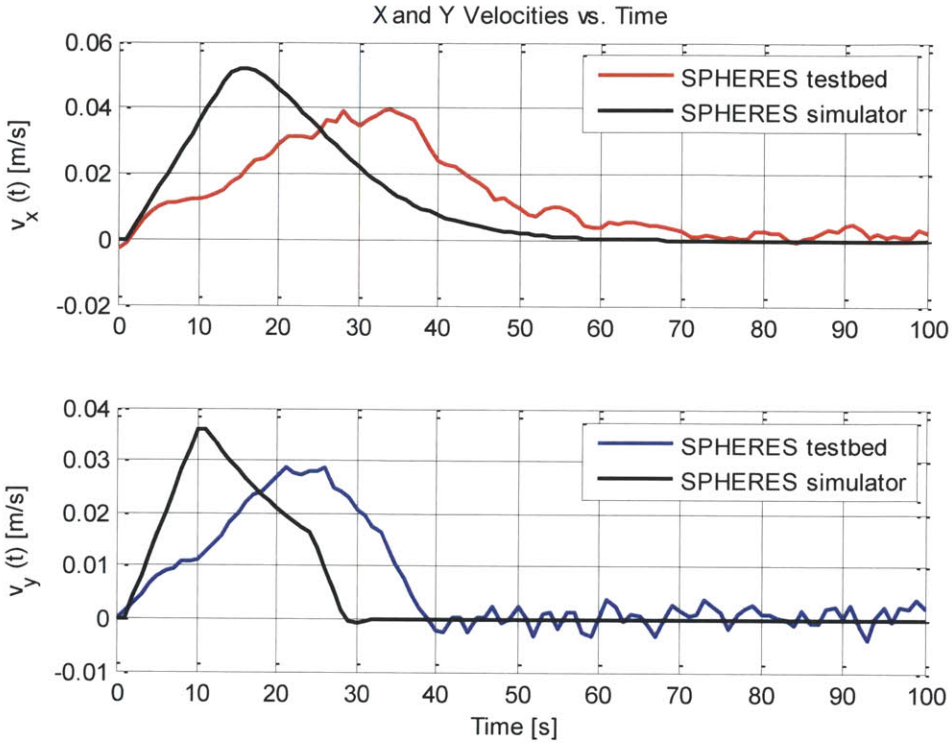

Figure 41: Velocity vs. Time using the SPHERES testbed

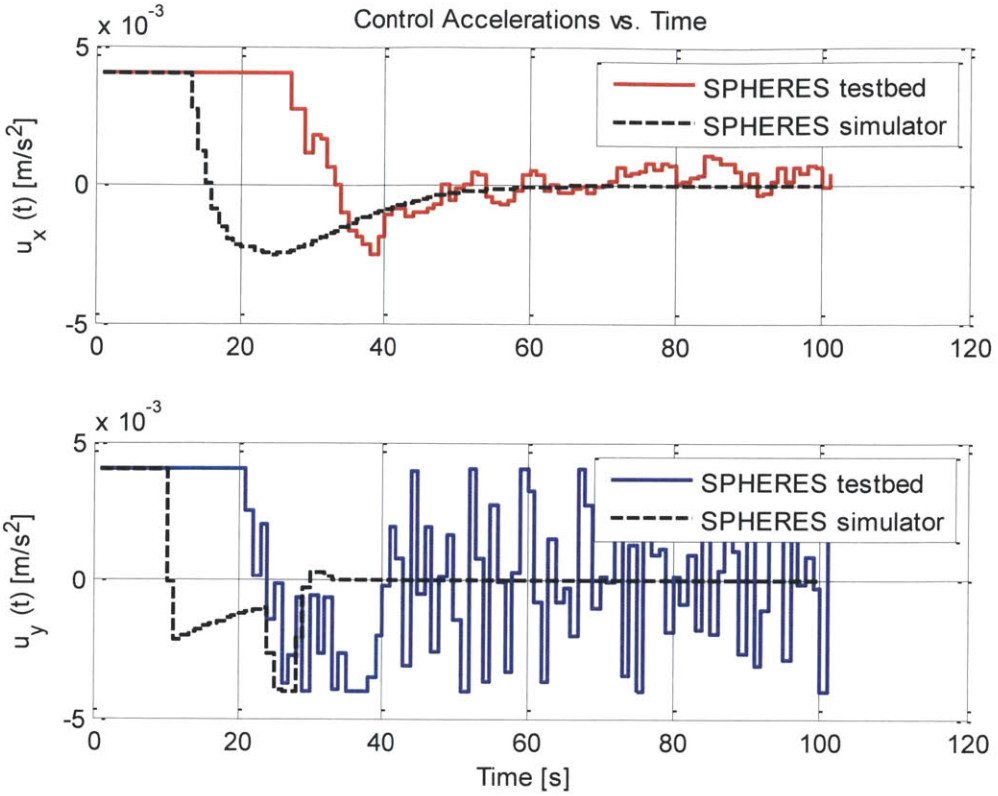

Figure 42: Control accelerations vs. Time using the SPHERES testbed

In Figure 42, there is a noticeable increase in control accelerations along the x and y axes. This is a direct consequence of the large position errors that build up during the start of the test.

However, the control accelerations in the y-direction show large fluctuations. This could be possibly due to the lack of y-axis convergence from the Global Metrology estimator. It is uncertain if these large fluctuations in the control acceleration are a result of velocity fluctuations seen in Figure 41 or these control accelerations are causing the velocity fluctuations in the y-direction.

4.6.4 PD and MPC Control comparison results using Global Metrology on SPHERES Hardware

Figure 43 and Figure 44 are shown to compare a standard PD controller and the MPC controller. As the results in Appendix A show, the MPC controller should show a 10-11% decrease in fuel consumption.

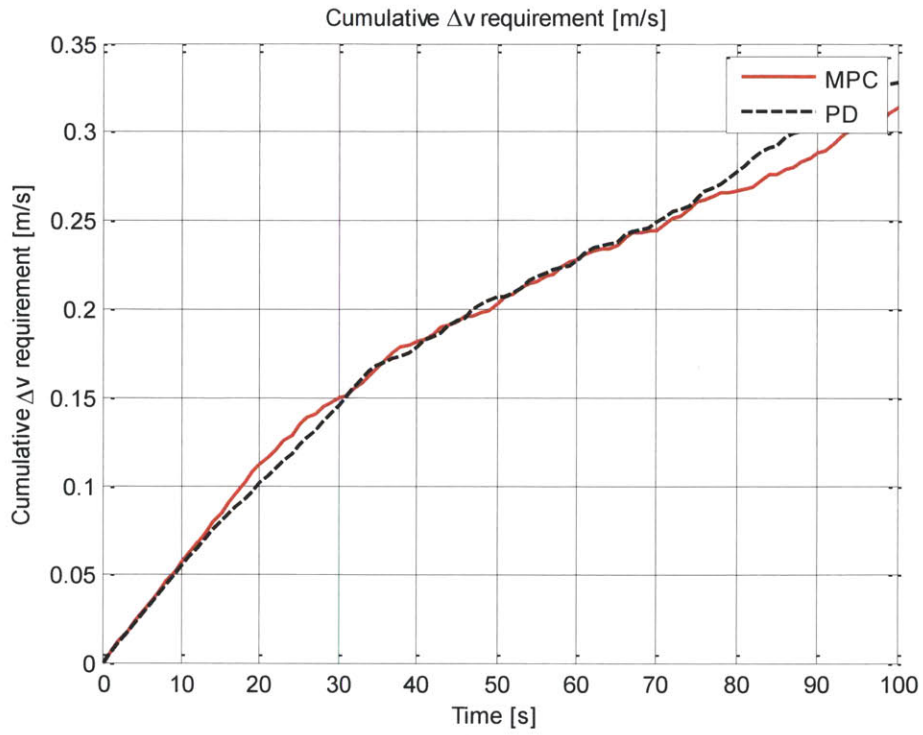

Figure 43: Cumulative Δv requirement in MPC vs. PD on SPHERES testbed

The MPC data displayed in Figure 43 shows a clear advantage with the PD controller. While the Δv is not as significant as the 11% seen in the simulations (refer to Appendix A), the MPC controller is still fuel more fuel efficient than a well-tuned PD controller. This advantage is due to the predictability features of the MPC algorithm and the thruster saturation handling ability. As more and more time elapses, the ability of the MPC controller to predict its state increases thereby increasing its handling ability.

These advantages discussed here can also be seen in Figure 44.

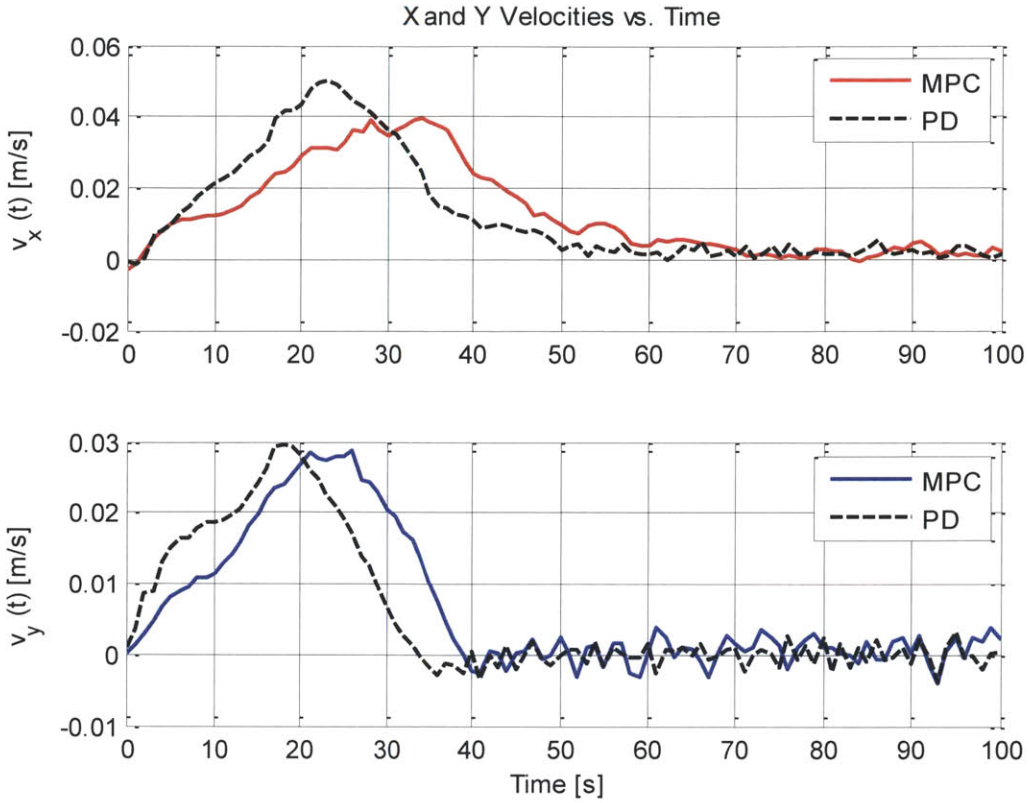

Figure 44: Velocity profile comparison for MPC and PD controller on SPHERES testbed

Figure 44 presents the MPC advantages in a slightly different form. The velocity in the x-direction is significantly lower for MPC throughout the entire maneuver even though both satellites reach the end of the maneuver before the test ends. This relationship is easier to see in the x-y trajectory mapped out in Figure 45.

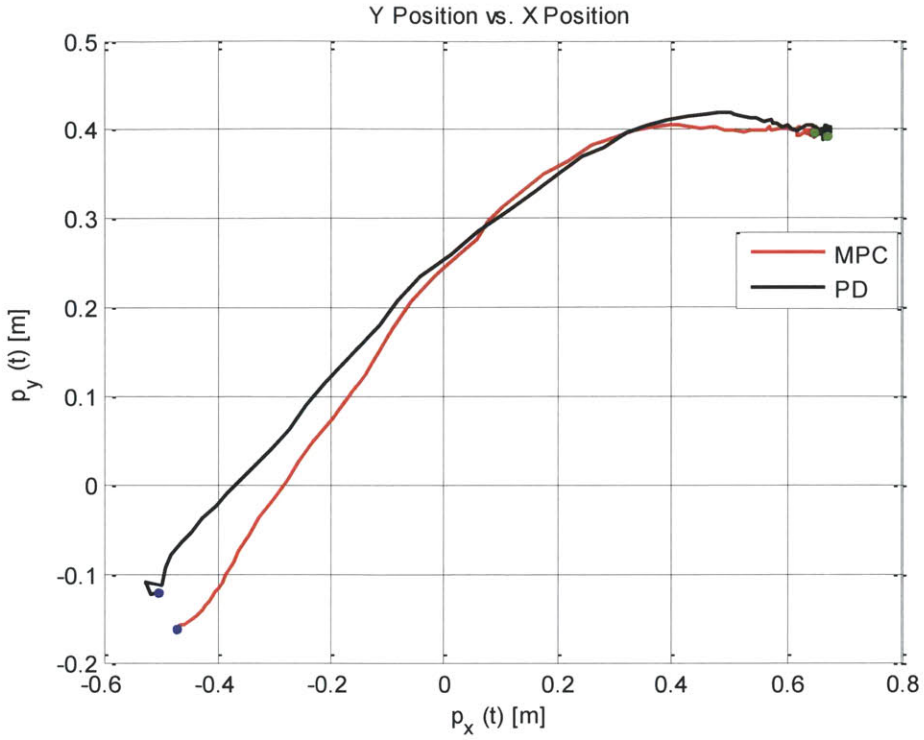

Figure 45: XY Trajectory comparisons of PD vs. MPC

The green points in Figure 45 refer to the final ending locations of the SPHERES satellites. Even though both controllers had different starting points, the PD controller overshot the target x-location causing it to backtrack to find the final target location. In contrast, the MPC controller took a much smoother approach to the ending location with nearly zero overshoot. This shows that the MPC controller is indeed more fuel efficient than the PD controller. See Appendix A for more results from the PD-MPC test.

4.7 Conclusion

This chapter presents a different controller approach for close proximity rendezvous and capture. The details of the controller architecture and its communication framework are described and experimental results with a PD controller using the SPHERES hardware are shown. From these results, it can be concluded that the MPC controller is a viable controller for use in such a rendezvous application with clear fuel efficiency benefits over

PD control. However, no tests were performed with the vision estimator in the loop due the MOSR shell interference issues described in Section 3.5.

Chapter 5

5 Conclusion

5.1 Summary of Thesis

This thesis presents the first steps of a larger research initiative to investigate the problem of rendezvous and capture of a completely passive object around Mars orbit.

Chapter 2 presents the overall MOSR SPHERES system which is currently being developed in order to perform a full 6-DOF test on-board the International Space Station. This testbed consists of a capture mechanism with at least two independent non-stereoscopic cameras used in succession. A bi-color MOSR shell is designed and built to increase the accuracy and robustness of the vision algorithm.

Chapter 3 presents the two algorithms used in the vision system. This system does not use any fiduciary markers or known surface features to compute the 3 DOF state estimate. The system assumes only a constant circular cross-section of given diameter. First, the mechanism of the Hough transform is discussed. The inherent drawbacks of using such an algorithm are explained and possible mitigating solutions are discussed. Second, an algorithm based on astronomical photometry is presented. The increased fidelity in measuring target range is noted and special attention is paid to the weaknesses of each of these algorithms (when used in parallel). Finally, the use of a Kalman Filter is discussed as a possible 'smoother' to integrate both these algorithms. Once again, the mechanism of the linearly-coupled Kalman filter is discussed and experimental data is shown with appropriate comparisons to the Global Metrology system (which is assumed to be ground truth).

Chapter 4 presents the controller used in this project. The controller is based on Model Predictive control as outlined in [19] and is used for rendezvous and capture of the OS from 'few meters' away. The system specific constraints are listed and the communications framework of this controller for this application is revealed. Evidence is

presented that strongly suggests that MPC can be a better alternative to PD based controllers for close proximity rendezvous and capture applications.

5.2 List of contributions

The following is a list of contributions of this thesis:

- Design and development of a vision system that can be used for close proximity rendezvous applications.
- Implementation and experimental evaluation of a vision system that uses no fiduciary markers or surface features for relative state estimation. This vision system also uses a single camera and can be an advantage over more sophisticated stereoscopic systems.
- Identification of the weaknesses of the approach to relative state estimation using the vision techniques described. A Kalman filter is introduced to integrate both these algorithms and experimental evaluation of this estimator is presented.
- Implementation and testing of a new Model Predictive Controller for space-based rendezvous and capture applications. This particular application of MPC has not been done before and there is sufficient evidence to show that MPC is a more fuel-efficient alternative to PD.

5.3 Future work

The main focus of the future work on the SPHERES MOSR project will involve more integrated testing of the estimator and the MPC controller. As is noted before, the MOSR shell described in Section 2.2.3 interferes with the IR synch pulse sent from the SPHERES satellite to the Global Metrology beacons. Modifications to this shell need to be made before more integrated testing can be done to test the estimator and the controller in a closed-loop environment.

Future work on the vision navigation system should include incorporation of the Kalman Filter into the Hough Transform initial estimate. This ensures that the algorithm can deliver OS ranges with maximum accuracy possible. The ability of the Kalman filter to predict future states of the OS can be invaluable to the accuracy of the

Hough transform. This approach to integrate the estimator into the working of the Hough transform is described in Mills et. al[33]. The astronomical photometry algorithm can also be upgraded to compute new estimates for the proportionality constants on the fly. Since the range estimate of the photometry algorithm depends heavily on the constants of the inverse-quartic curve fit, updating these constants over time can lead to better precision and accuracy of the range measurement. Finally, the Kalman Filter can be upgraded to an Extended Kalman Filter to take into account of some of the inherent non-linearities and cross coupling of the measurement model. While this upgrade will not offer a significant increase in estimator accuracy (since the accuracy of the estimator is driven by the 'sensor' algorithms), it might allow the initial state convergence to be smoother than what is seen now.

A number of areas exist for the preliminary results presented in Chapter 4. The MPC controller can be tested with the vision estimator to test closed-loop performance of the entire system. Additionally, a test with no control inputs can be conducted to diagnose the unexpected noise seen along the y-axis. This test will reveal the true source of noise in the system along the y-axis; this could be sensor noise through the Global Metrology system or inherent noise in the controller design which might need to be accounted for, for future tests using this control system.

References

[1] Hargraves, R. B., Knudsen, J. M., Madsen, M. B., and Bertelsen, P., "Finding the right rocks on Mars," *Eos*, vol. 82, no. 26, pp. 292–292, 2001.

[2] R. Mattingly, "Continuing evolution of Mars sample return," *2004 IEEE Aerospace Conference Proceedings*, pp. 477–492, 2004.

[3] Aurora Flight Sciences, "SPHERES MOSR RDOS Phase 2 NASA Proposal," 2010.

[4] "DARPA Orbital Express." [Online]. Available: http://archive.darpa.mil/orbitalexpress/index.html. [Accessed: 01-Aug-2013].

[5] "JAXA | Asteroid Explorer 'HAYABUSA' (MUSES-C)."[Online]. Available: http://www.jaxa.jp/projects/sat/muses_c/index_e.html. [Accessed: 01-Aug-2013].

[6] M. Adler, W. Owen, and J. Riedel, "Use of MRO Optical Navigation Camera to Prepare for Mars Sample Return," *LPI Contributions*, 2012.

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

[8] P. Hough, "METHOD AND MEANS FOR RECOGNIZING COMPLEX PATTERNS," 1960.

[9] P. Kierkegaard, "A method for detection of circular arcs based on the Hough transform," *Machine Vision and Applications*, vol. 5, pp. 249–263, 1992.

[10] D. H. Ballard, "Generalizing the Hough transform to detect arbitrary shapes," *Pattern Recognition*, vol. 13, pp. 111–122, 1981.

[11] F. Zana and J. C. Klein, "A multimodal registration algorithm of eye fundus images using vessels detection and Hough transform.," *IEEE Transactions on Medical Imaging*, vol. 18, pp. 419–428, 1999.

[12] V. Kamat and S. Ganesan, "An efficient implementation of the Hough transform for detecting vehicle license plates using DSP'S," *Proceedings RealTime Technology and Applications Symposium*, pp. 58–59, 1995.

[13] A. Karnieli, A. Meisels, L. Fisher, and Y. Arkin, "Automatic extraction and evaluation of geological linear features from digital remote sensing data using a Hough transform," *Photogrammetric Engineering and Remote Sensing*, vol. 62, pp. 525–531, 1996.

[14] G. Casonato and G. B. Palmerini, "Visual techniques applied to the ATV/1SS rendezvous monitoring," *2004 IEEE Aerospace Conference Proceedings IEEE Cat No04TH8720*, vol. 1, 2004.

[15] C. Sterken and J. Manfroid, *Astronomical Photometry: A Guide*. Springer, 1992, p. 272.

[16] H. Shapley, "Studies based on the colors and magnitudes in stellar clusters. VI. On the determination of the distances of globular clusters.," *The Astrophysical Journal*, vol. 48, p. 89, Sep. 1918.

[17] J. Riedel, J. Guinn, M. Delpech, and J. Dubois, "A combined open-loop and autonomous search and rendezvous navigation system for the CNES/NASA Mars Premier Orbiter mission," Pasadena, CA, 2003.

[18] A. Bemporad and M. Morari, "Robust Model Predictive Control: A Survey," *Robustness in Identification and Control*, vol. 245, pp. 207–226, 1999.

[19] A. Valmorbida, "Model Predictive Control (MPC) strategies on the SPHERES test-bed for the Mars Orbital Sample Return (MoSR) scenario," University of Padova, 2013.

[20] C. E. Garcia, D. M. Prett, and M. Morari, "Model Predictive Control : Theory and Practice a Survey," *Automatica*, vol. 25, pp. 335–348, 1989.

[21] C. E. Garcia and A. M. Morshedi, "Quadratic programming solution of dynamic matrix control (QDMC)," *Chemical Engineering Communications*, vol. 46, pp. 73–87, 1986.

[22] V. Manikonda, P. O. Arambel, M. Gopinathan, R. K. Mehra, and F. Y. Hadaegh, "A model predictive control-based approach for spacecraft formation keeping and attitude control," *Proceedings of the 1999 American Control Conference Cat No 99CH36251*, vol. 6, pp. 4258–4262, 1999.

[23] R. Linfield and P. Gorham, "The DS3 Mission: A Separated Spacecraft Optical Interferometer," *The American Astronomical Society*, p. 3, 1998.

[24] A. Richards and J. P. How, "Model predictive control of vehicle maneuvers with guaranteed completion time and robust feasibility," *Proceedings of the 2003 American Control Conference 2003*, vol. 5, pp. 4034–4040, 2003.

[25] J. Lavaei, A. Momeni, and A. G. Aghdam, "A Model Predictive Decentralized Control Scheme With Reduced Communication Requirement for Spacecraft Formation," *IEEE Transactions on Control Systems Technology*, vol. 16, pp. 268–278, 2008.

[26] D. Miller, A. Saenz-Otero, and J. Wertz, "SPHERES: a testbed for long duration satellite formation flying in micro-gravity conditions," *AAS/AIAA Space Flight Conference*, 2000.

[27] H. K. Yuen, J. Princen, J. Illingworth, and J. Kittler, "Comparative study of Hough transform methods for circle finding," *Image and Vision Computing*, vol. 8, pp. 71–77, 1990.

[28] J. Canny, "A computational approach to edge detection.," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 679–698, 1986.

[29] C. M. Brown, "Inherent bias and noise in the hough transform.," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 5, pp. 493–505, 1983.

[30] P. Kim, *Kalman Filter for Beginners: with MATLAB Examples*. CreateSpace Independent Publishing Platform, 2011, p. 232.

[31] P. Zarchan and H. Musoff, *Fundamentals of Kalman Filtering: A Practical Approach*, vol. 190. 2005, p. 2010.

[32] B. Tweddle, "Computer vision based navigation for spacecraft proximity operations," no. February, p. 225, 2010.

[33] S. Mills, T. P. Pridmore, and M. Hills, "Tracking in a Hough Space with the Extended Kalman Filter," *Procedings of the British Machine Vision Conference 2003*, pp. 18.1–18.10, 2003.

6 Appendix A

6.1 MPC Simulation results using PD and MPC Control

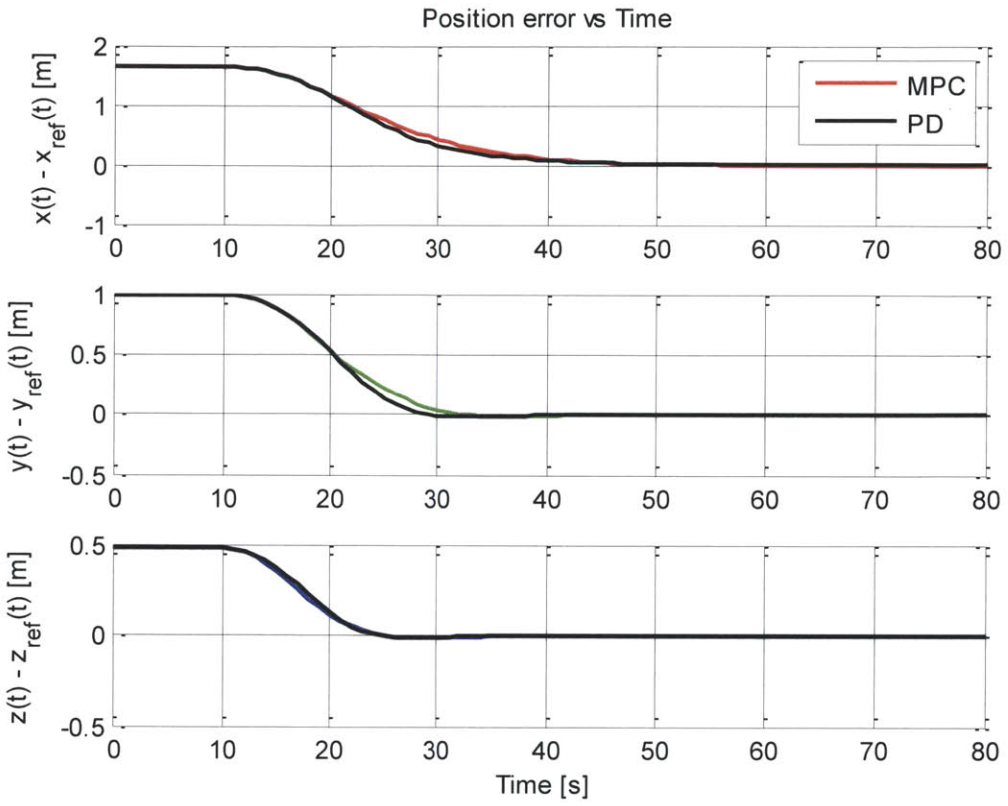

Figure 46: SPHERES Simulator PD MPC Comparison – Position Error vs. Time

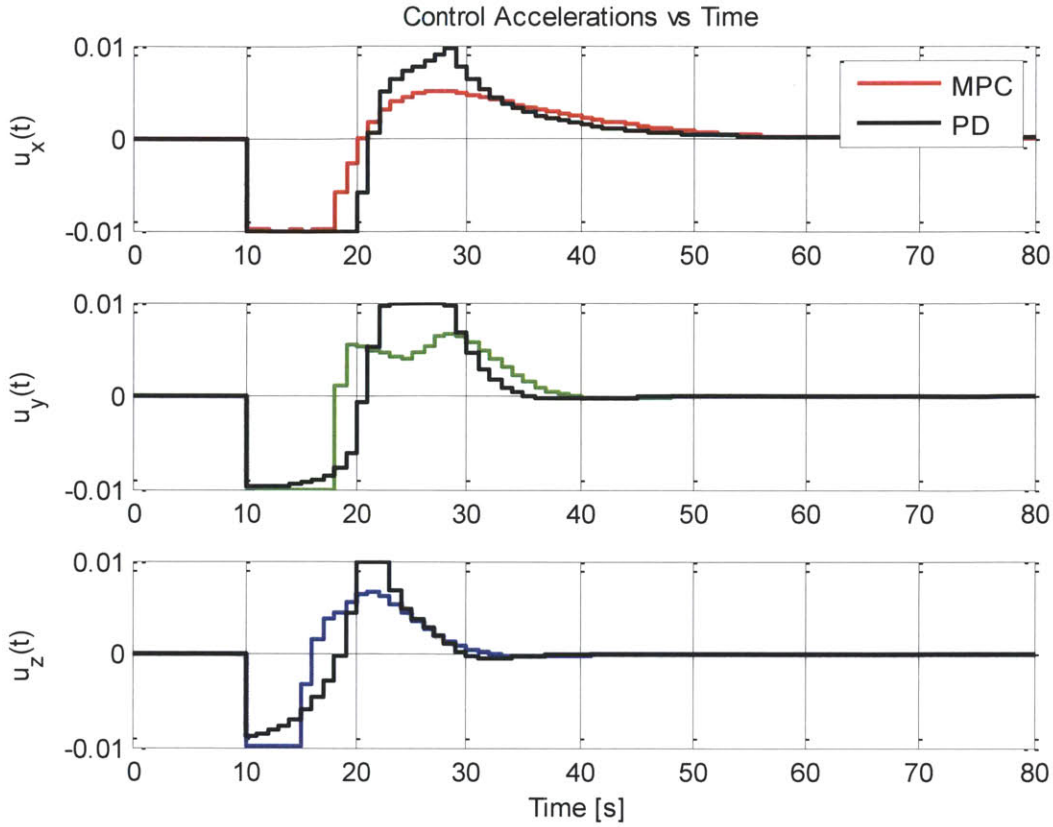

Figure 47: SPHERES Simulator PD MPC Comparison – Control Accelerations vs. Time

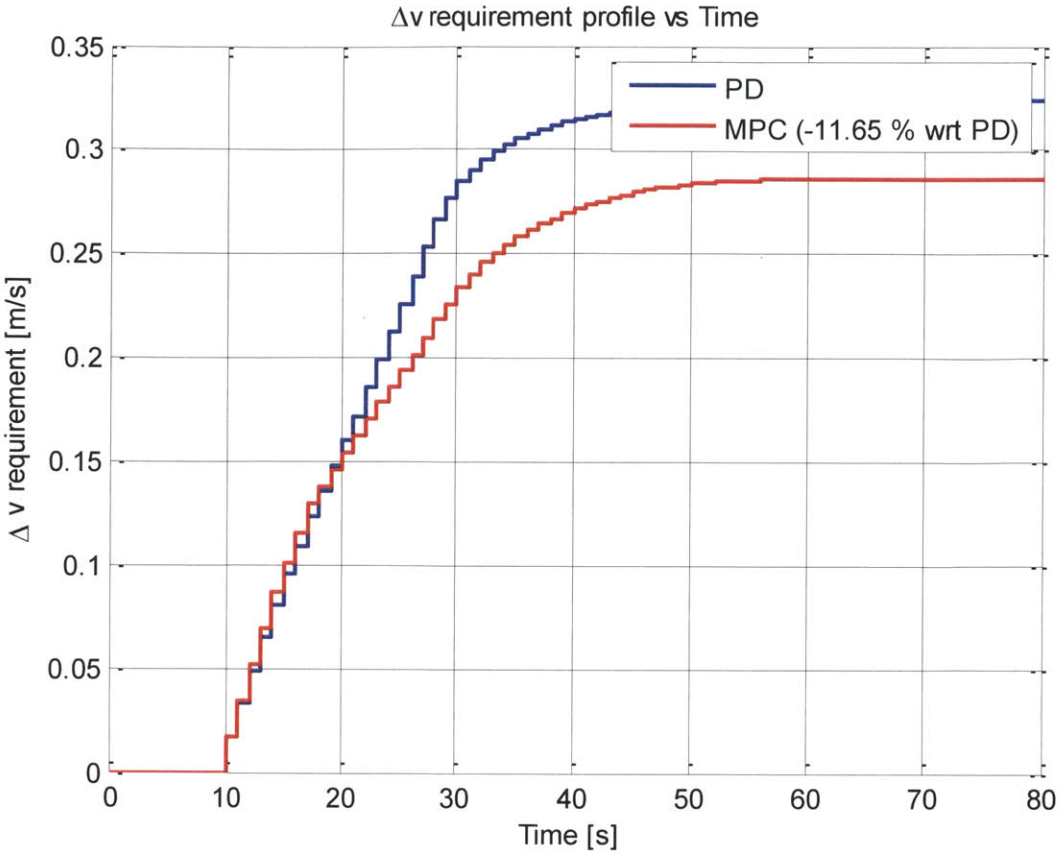

Figure 48: SPHERES Simulator PD MPC Comparison – Δv requirements vs. Time

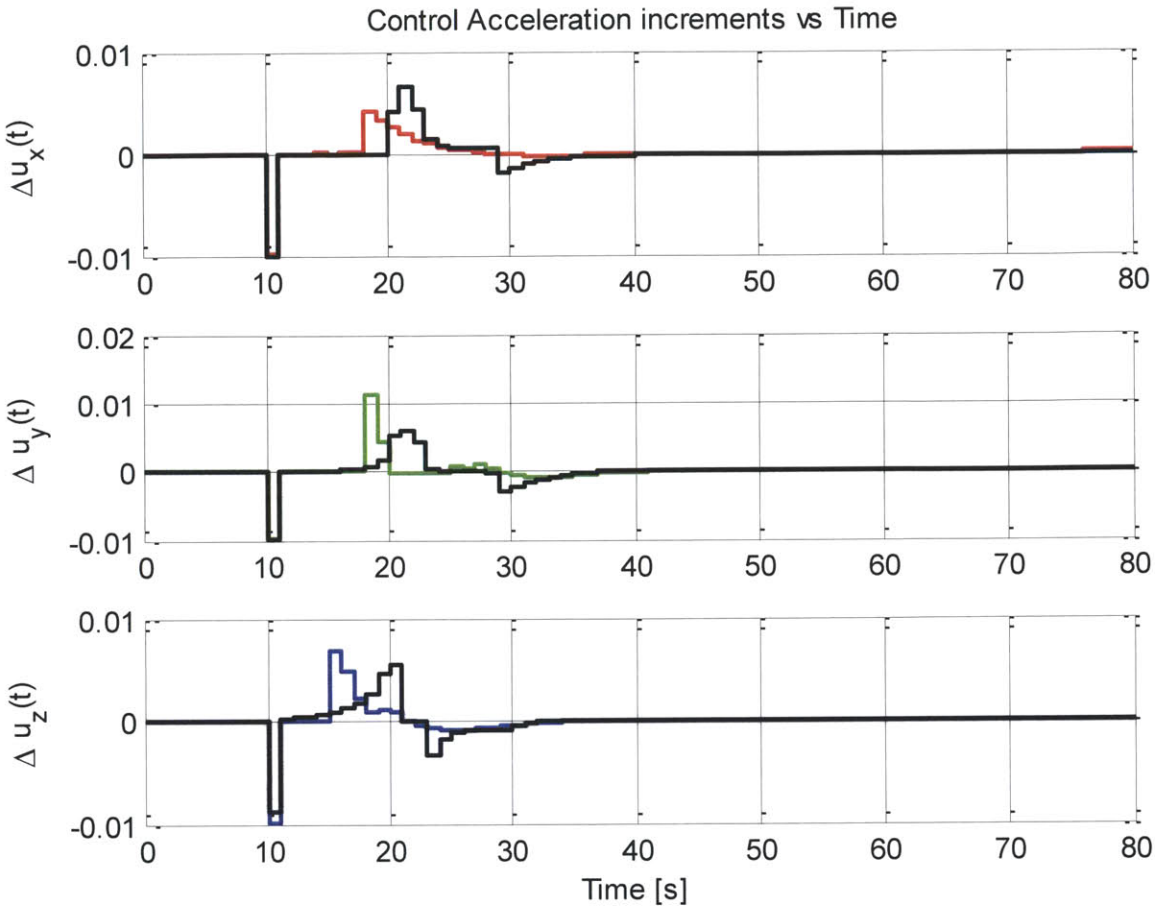

Figure 49: SPHERES Simulator PD MPC Comparison – Control accelerations vs. Time

6.2 PD control test on SPHERES hardware

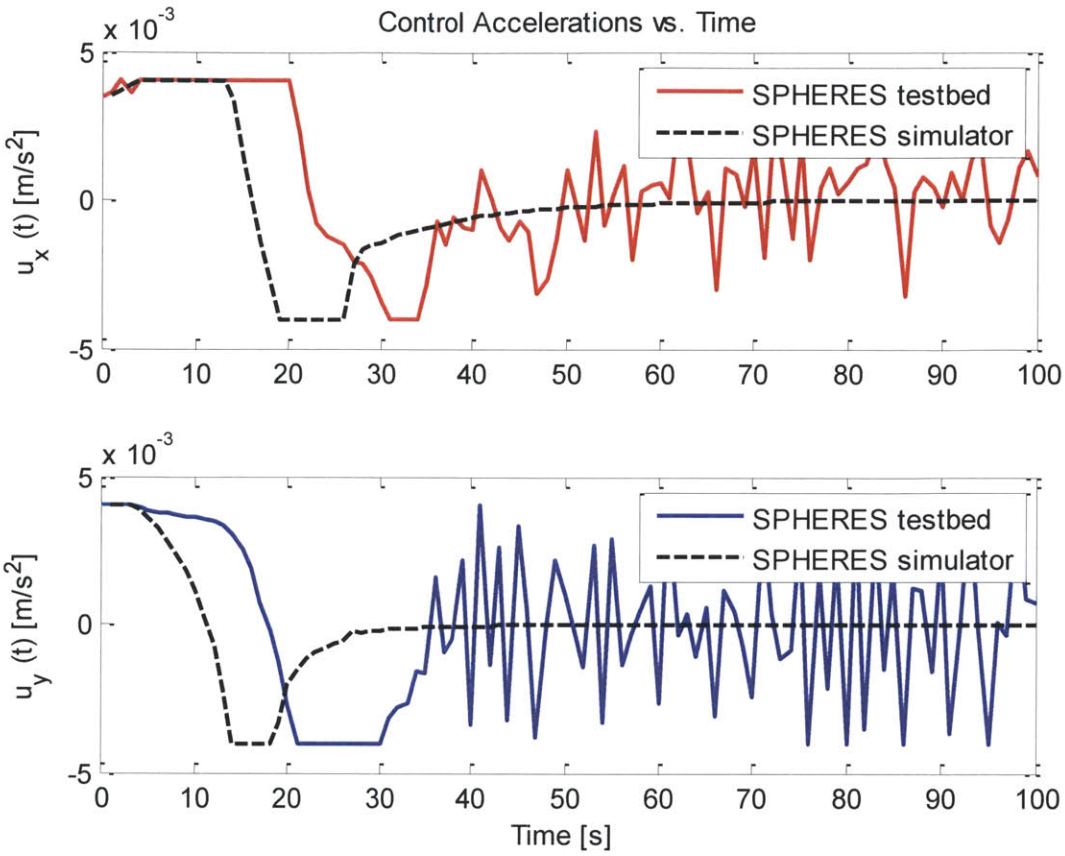

Figure 50: SPHERES hardware PD Comparison – Control accelerations vs. Time

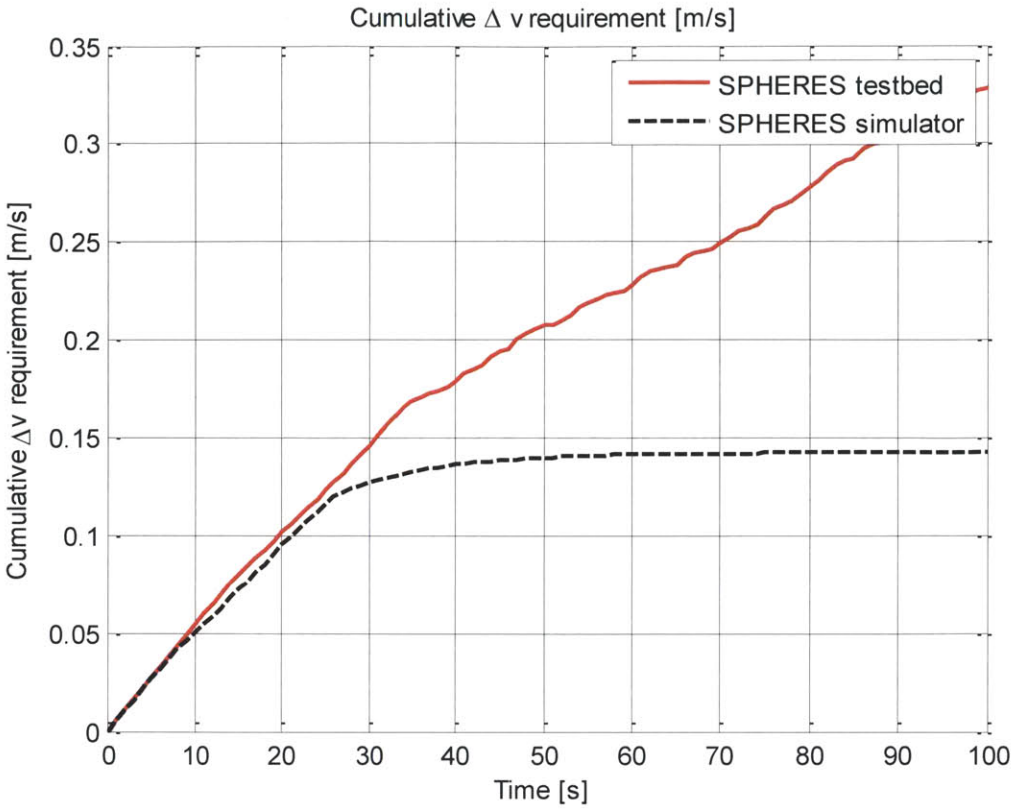

Figure 51: SPHERES hardware PD Comparison – Cumulative Δv vs. Time

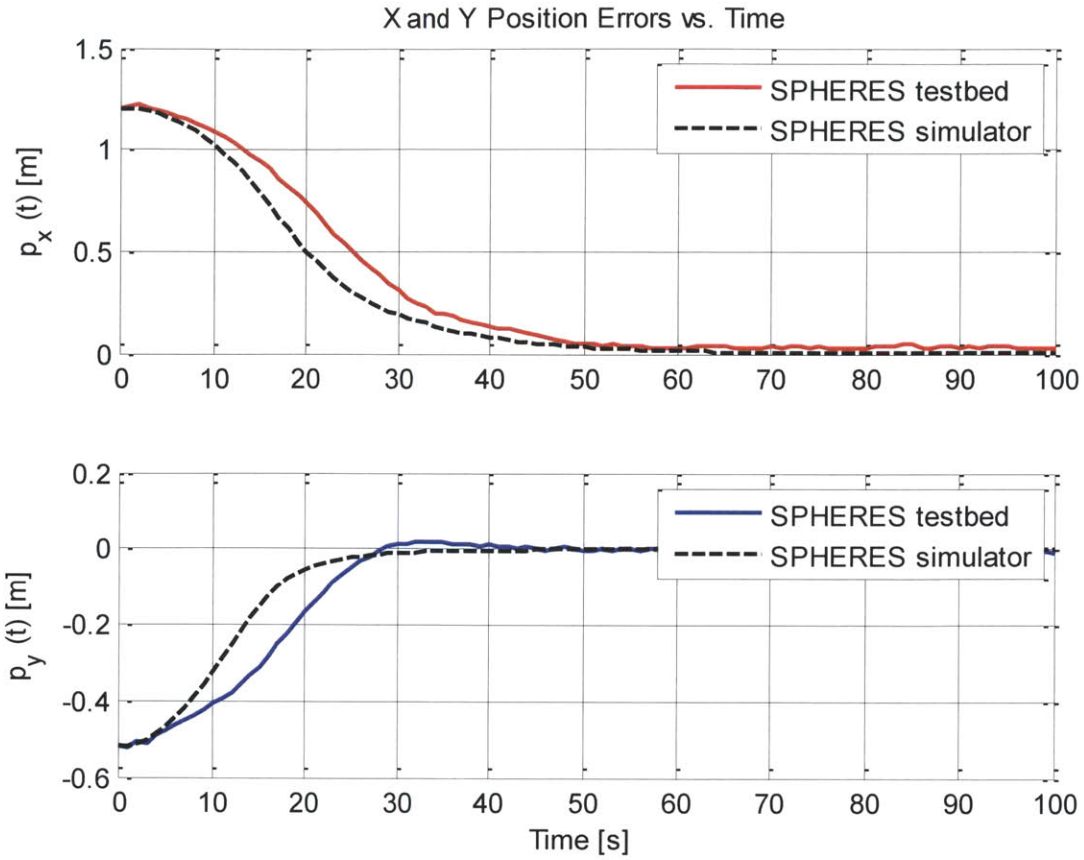

Figure 52: SPHERES hardware PD Comparison – X and Y Position vs. Time

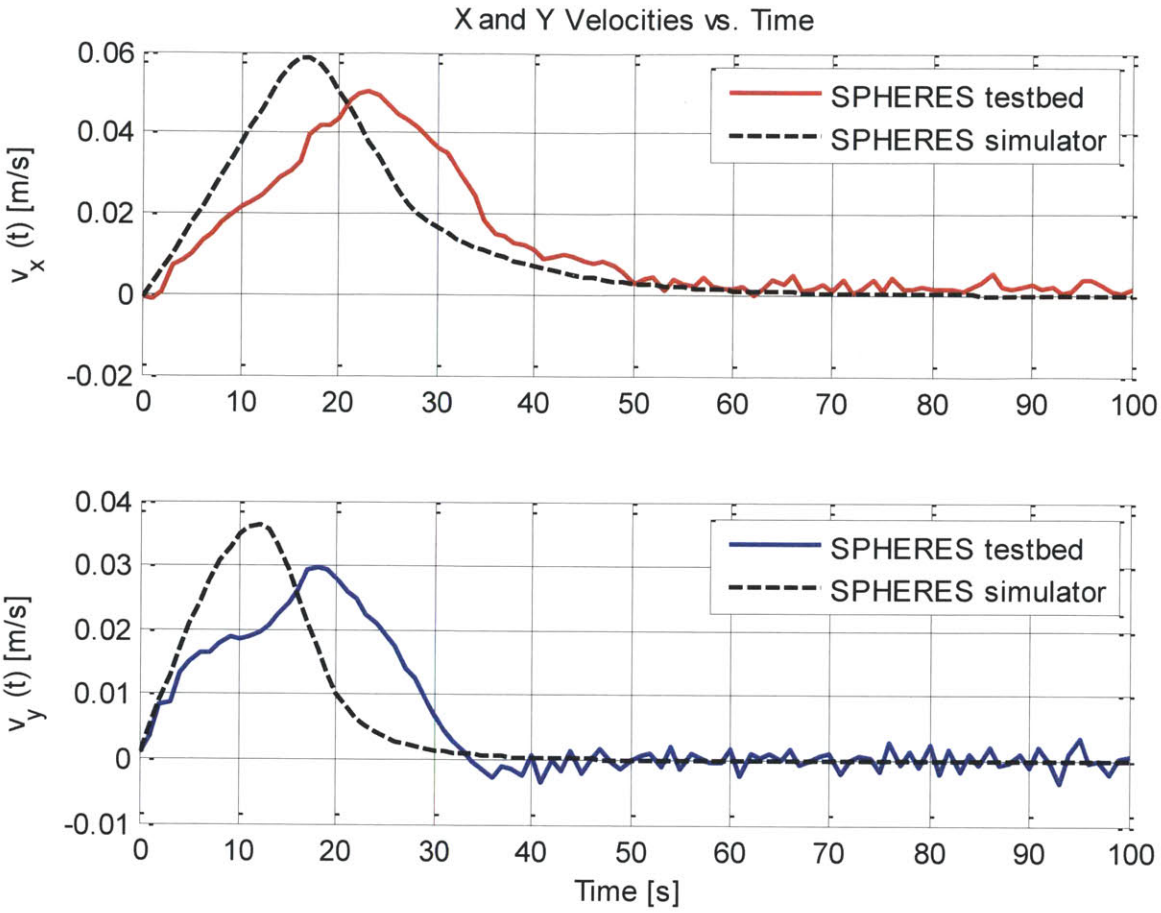

Figure 53: SPHERES hardware PD Comparison – Velocities vs. Time

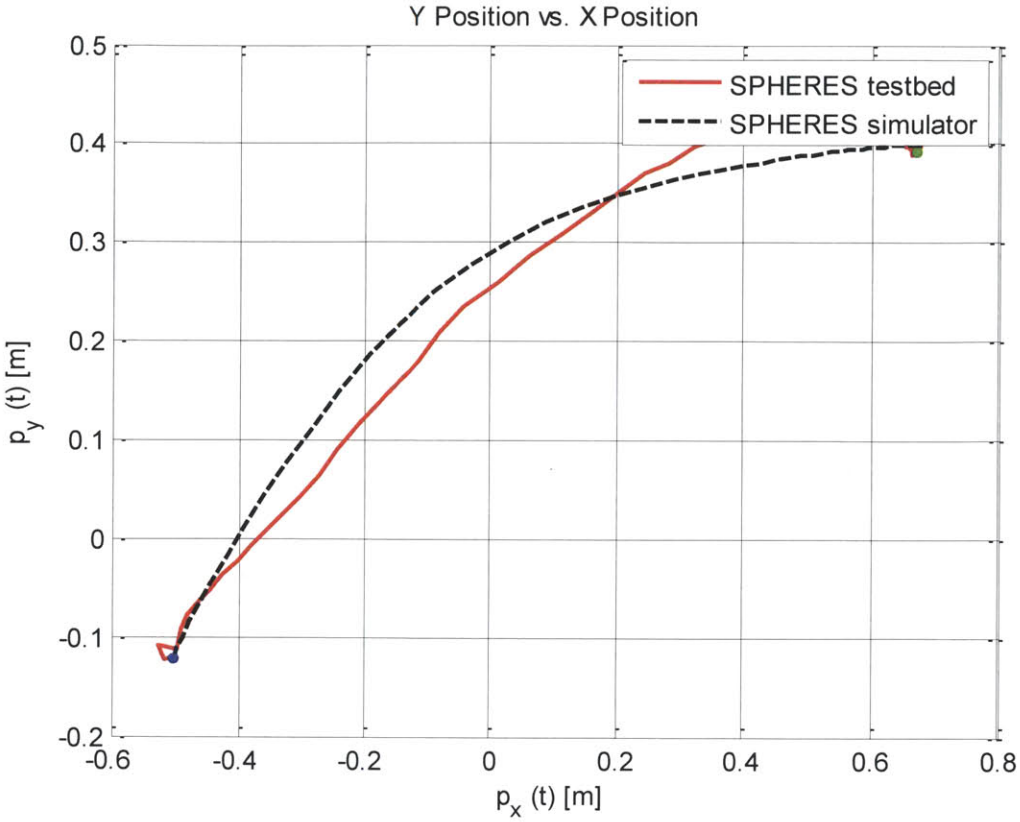

Figure 54: SPHERES hardware PD Comparison – XY Trajectory

6.3 MPC Test on SPHERES hardware

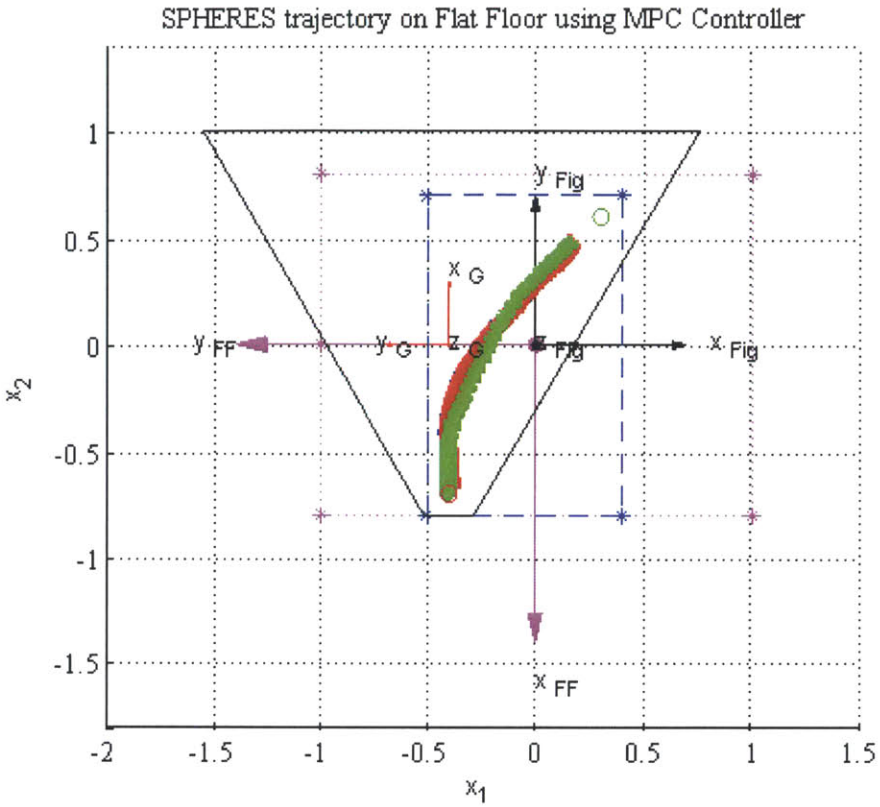

Figure 55: MPC Trajectory using SPHERES Hardware

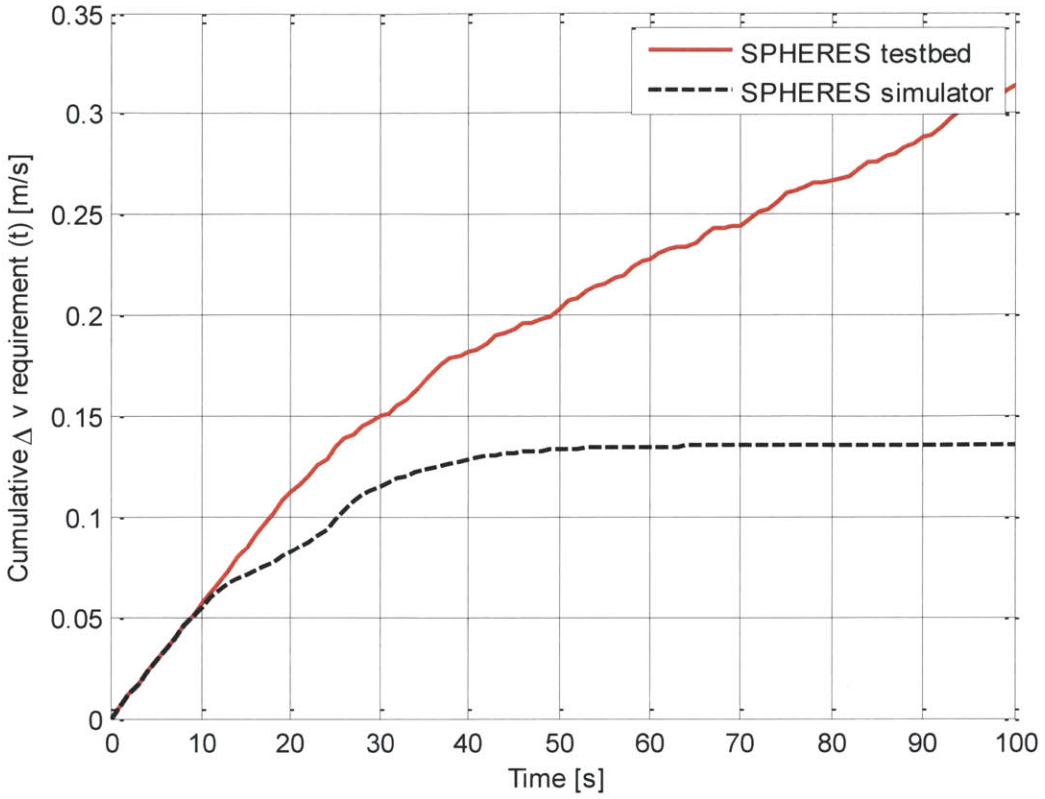

Figure 56: Cumulative Δv requirement for MPC using SPHERES hardware

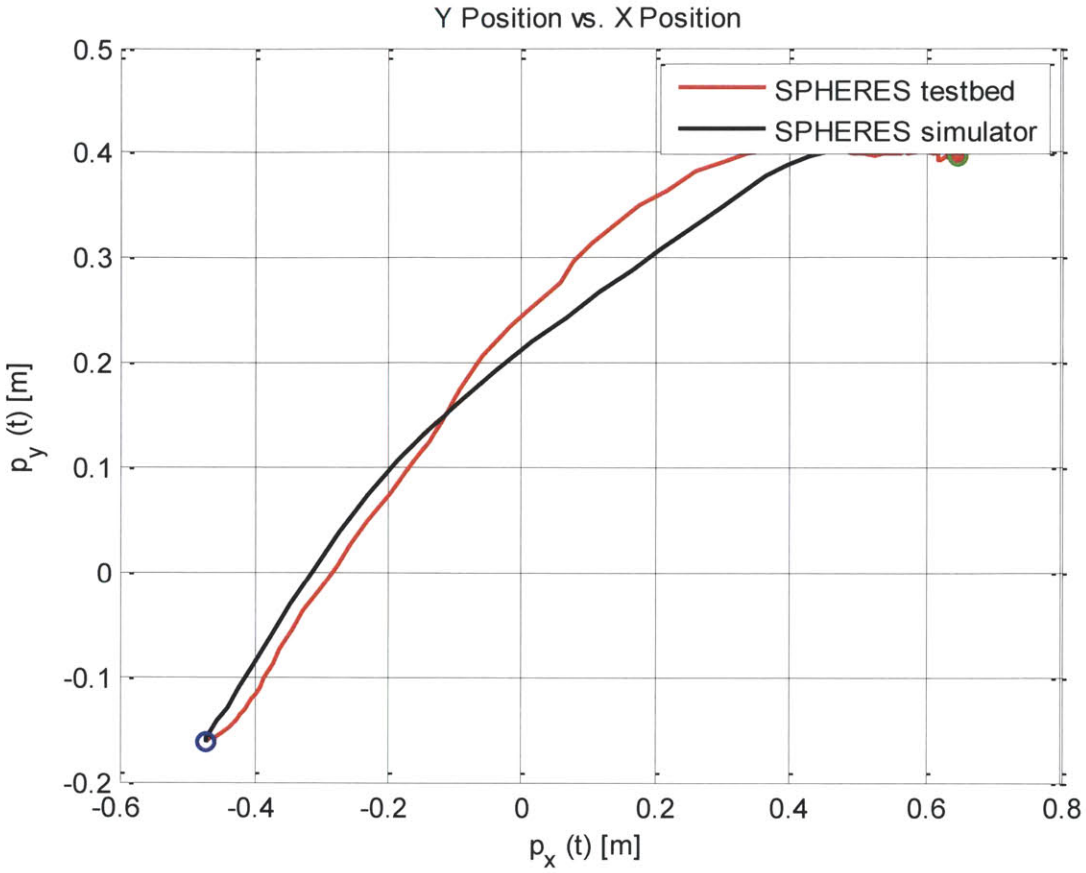

Figure 57: XY Trajectory for MPC using SPHERES hardware