# Virtual Simulation for Testing a Vision-Based Autonomous Docking and Navigation Algorithm

John Karasinski, Christopher Lorenzen, and Melanie Stich
Department of Mechanical & Aerospace Engineering
University of California, Davis
Davis, CA, USA

***Abstract*—With small failures leading to the loss of entire spacecraft, innovations in inspection methods are required. Determining the viability of a proposed small satellite inspection free flyer with machine vision navigation is discussed and methods of testing the robustness of the system are developed. Some past examples in the field are presented as well as their applicable contributions. A baseline system using a traditional laser range finding position determination system is used as a comparison to rate the success of the proposed machine vision position determination system. Two design aspects of the navigation control system are defined so that their alterations to the behavior of the craft can be identified. A possible test environment and simulation methods are introduced to serve as a method of acquiring data to develop empirical cause and effect relationships between the different suggested control design parameters.**

***Index Terms*—robots, satellite**

## I. Introduction

### A. Background

In recent years the small satellite platform has increased in popularity. With the ability to launch multiple small satellites as a secondary payload and the relatively low development cost involved, they are one of the most viable testbeds for new technologies. One great example of this is the class of satellites that adhere to the CubeSat standard [1]. Although the small size limits their capabilities, CubeSat initiatives have made many projects possible due to the minimal costs required. Higher fidelity projects usually require a larger volume to contain all desired systems. There are several projects that use slightly larger free flyers that have produced unique navigation demonstrations such as AERCam Sprint [2], Mini-AERCam [3], and SPHERES (SPHERES was also used for many subsequent projects due to its modular nature) [4].An obvious limitation to small satellites is the ability to fit all required architecture within the size restrictions set by the design. One way to reduce the space required is to choose your subsystems such that some may fill multiple roles. An example of this is to use a science instrument to also collect navigation data. More specifically, to use an optical sensor that is needed to take stereoscopic visual recordings for inspection purposes to also determine position and attitude relative to a target. To do this the image captured must be used in conjunction with a machine vision algorithm in order to estimate navigational parameters.

The decision to use stereoscopic visual sensors to inspect was largely in part to increase ease of inspection for a human operator. Using stereo vision and full visual light spectrum, a more familiar virtual image is produced. Technicians are able to view the subject as if it was in front of them allowing for quick and informed decision making.

Additionally if the inspector had the ability to dock to the target craft, this could greatly increase the effective mission operation time. The inspector could refuel, recharge, and download data. Instead of a small disposable craft that has a single use, it could become a long term asset and an integral part of the target crafts diagnostic routines. Most current docking procedures utilize a laser range finder for determining distance. Laser range finders have flight heritage and have moderate accuracy [5].

### B. Motivation

The advantages presented by the addition of an inspector satellite to a larger craft show the necessity for the development of this technology. Whether the target craft is inhabited by astronauts or not it can still increase the probability of completing a mission successfully.

There are two main types of damage of concern. The first cause of alarm is physical damage due to collision with orbital debris or micrometeorites. The second thing to monitor is component failure. Either of these events can be disastrous to a mission. The best case scenario is that the craft is not inhabited and you only lose a multimillion dollar piece of equipment. With inspection capabilities, damage and failures can be identified in a timely manner. With more information about the problem area and additional time to deal with the issue, the possibility of salvaging the mission increases greatly. There are even additional opportunities that arise if inspection capability was integrated with other projects like DARPAs Phoenix Satellites [6]. If damage can be known or if the specific component that failed can be identified then a satellite similar to the Phoenix system can dock with and repair or repurpose a damaged satellite.

Using the International Space Station as an example, NASA has showed concern for both damage types that could be seen with a visual inspector. There have already been studies done trying to quantify the danger involved with debris collision [7]. As seen in Figure 1, the impact risk for different parts of the station have already been determined and an inspection satellite could be made to center its efforts in patrolling high risk areas. There have also been close calls due to component

failures. One such incident that required emergency extra-vehicular activity was when an ammonia pump developed a leak in may of 2013 [8].

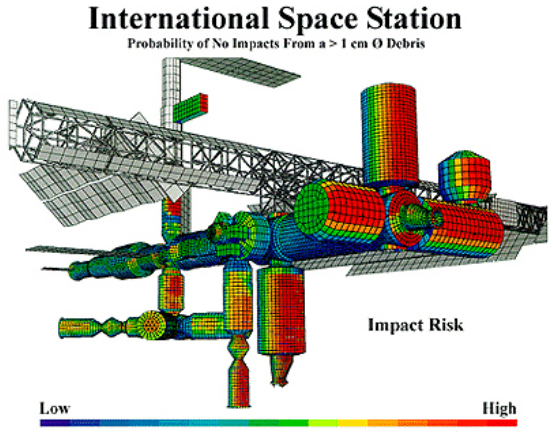


Fig. 1. Debris impact risk by region of the International Space Station

Beyond the overall need for an inspection resource, there is also a driving factor to be as efficient as possible. With increased navigational efficiency, it extends the life of the inspector as well as increases the data that is able to be gathered with each excursion. As the inspection capability of the free flyer increases, the overall feasibility of the inspection mission increases.

### C. Objective

The purpose of this exercise is to create a simulation to test the viability and robustness of a machine vision based navigation system. Several aspects will be modified to see if using stereoscopic cameras in conjunction with a machine vision algorithm is a viable option for a low cost inspection satellite. To reduce costs, accuracy can be traded to reduce computational capabilities, battery power, and propellant needed. When accuracy is reduced, chance of the inspector colliding with the subject increases, so a collision avoidance algorithm must also be tested. With respect to the collision algorithm, the position error of the machine vision system must be analyzed to determine if a more accurate ranging device is required.

To test the viability of the machine vision system, a basic machine vision system will be compared to a more advanced laser range finding system. Because of the limited processing power available, advanced machine vision systems are not possible. The processing power required for a laser range finding system is minimal, so more advanced versions fit within the scope of the inspection craft.

When the study has been completed, we would like to answer:

> Given a small satellite with machine vision navigation, how does it perform versus a laser range finder based navigation and how does varying the path correction deadband width and position update frequency affect the fuel and time efficiency of traveling to a specific location near an object with a constant attitude and constant orbit?

The path correction deadband refers to the deviation that is allowed from the target trajectory before a correction burn is made while the position update frequency is how often the machine vision system determines, updates, and compares the inspection satellites position with the predicted position produced by the system dynamics and the inertial navigation system. Under normal conditions, the only drift from the intended trajectory is from error or disturbances, but in this situation we also have drift due to the inspector and the target occupying different orbits. This means that even with perfect planning and a perfect initial trajectory, corrections are required. Since there is no way to circumvent course corrections, we must make our navigation scheme as efficient as possible.

With the concepts of deadband and update frequency explained, cause and effect relationships can be hypothesized but the weight of the interactions can only be seen from simulation. There will be a lot of trade offs happening when the deadband width or update frequency are changed. For example, a wider deadband will mean that the craft will make correction burns less often, but it will drift further off course than a slim deadband so that the corrections required will be larger.

## II. Literature Review

### A. Overview of Small Inspection Satellites

Over the past 20 years, there have been incredible advancements within the realm of semi-autonomous satellites. Beginning in 1997, the Autonomous Extravehicular Activity Robotic Camera Sprint (AERCam Sprint) was the first semi-autonomous satellite to demonstrate the use of a free-flying prototype camera aboard the International Space Station (ISS). While operating alongside STS-87 Mission Specialist Winston Scott, the AERCam Sprint flew under the remote-control guidance of Steve Lindsey for approximately 75 minutes, and relayed live television images to Columbia's Mission Control [2], [3]. After successfully completing this experiment, researchers and analysts decided to incorporate a higher level of autonomy, and produced a second prototype known as the Mini AERCam in 2000. While this satellite never made it to space, the Mini AERCam underwent multiple tests on an air-bearing table and in an orbital test simulation facility at Johnson Space Center. This newly designed satellite was given automatic position hold, point-to-point maneuvering, and an additional camera to provide an orthogonal view, allowing astronauts to navigate the Mini AERCam with respect to the ISS. Through these multiple additions, researchers expanded the satellite's capability to encompass supervised autonomous and/or remotely piloted operations [3], [9].

Later, in 2006 the first Synchronized Position Hold Engage Reorient Experiment Satellites (SPHERES), a self-contained nanosatellite made by MIT's Space Systems Laboratory, was launched to the ISS and taken to the US Laboratory. Since that time, this semi-autonomous satellite has been joined by two additional SPHERES, making this system the first

consistent experimental nanosatellite testbed aboard the ISS. Unlike the AERCam Sprint and the Mini AERCam, SPHERES is a modular satellite where each system is self-contained in individual capsules. This configuration allows SPHERES to easily incorporate system expansions onto specific platforms, such as navigation, without needing to reconfigure the entire craft. Furthermore, its modularity helps researchers efficiently address system failures, making it easier for astronauts to perform on-site repairs. To navigate SPHERES within the ISS, the system utilizes wall-mounted ultrasonic beacons and corresponding ultrasonic receivers attached to the nanosatellite [4]. [When SPHERES looks to determine its location, it emits an infrared flash. Once emitted, the satellite waits for the wall-mounted beacons to emit corresponding ultrasonic pulses. After receiving these ultrasonic pulses, the satellite measures its range based on the pulse's time of flight, and can then calculate its relative position, attitude, and angular velocity [4], [10].] This unique navigation system allows SPHERES to emulate a "pseudo-GPS" time-of-flight sensing system, and ultimately estimate its position, angular velocity, and attitude without the potential for signal interference and noise – a challenge that has been previously encountered with GPS systems [10]. Through this autonomous navigation and modular design, the SPHERES testbed has become a versatile platform for developing vision-based navigation algorithms, as well as anti-collision and formation flying algorithms. By allowing research teams to create algorithms that can then be uplinked to the SPHERES test system aboard the ISS, researchers can receive live feedback, and ultimately find the exact areas within their algorithms that need improvement.

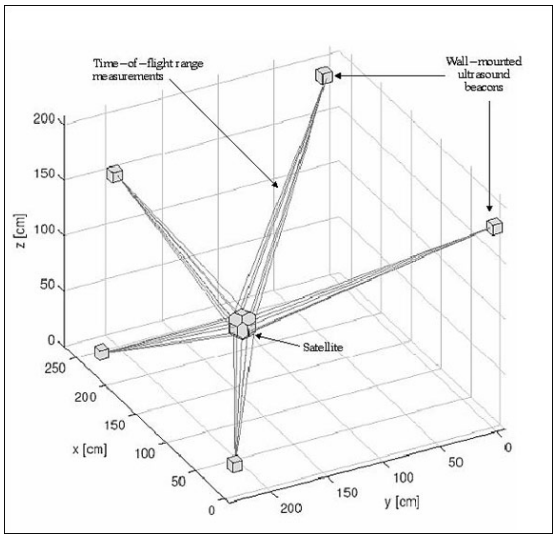


Fig. 2. SPHERES time-of-flight range measurements and wall-mounted ultrasound beacons.

### B. SPHERES VERTIGO

In 2008 the MIT Space Systems Laboratory began building an upgrade to the SPHERES system, known as the Low Impact Inspection Vehicle (LIIVe), as part of the Visual Estimation and Relative Tracking for Inspection of Generic Objects (VERTIGO) program. Once completed, this upgrade would later be attached to the existing SPHERES system and act as VERTIGO "goggles," allowing SPHERES to perform vision-based navigation experiments in the microgravity environment aboard the ISS. After adjusting these VERTIGO Goggles to suit the ISS environment, the final system was upgraded to include two monochrome stereo cameras, two illuminating LEDs, a Via 1.2 GHz Via Nano processor, a 802.11n network card, and optics that included a larger aperture lens in a synchronized stereo configuration [4]- [11].

When the modified SPHERES VERTIGO was ready for experimentation, numerous flight algorithms were tested to demonstrate the spacecraft's complete autonomy. After ISS Expedition 34, it was confirmed that SPHERES VERTIGO was capable of autonomously conducting a circular orbit about an uncooperative object, while simultaneously maintaining a constant relative position between SPHERES VERTIGO and the target. This objective was achieved through the primary use of inertial sensors and cameras, and was considered an unprecedented success [11], [12].

While the SPHERES VERTIGO has already made considerable progress over the past few years, this advanced system continues to exhibit a perpetuating navigational limitation through the use of ultrasonic beacons and receptors. As seen in Figure 2, the navigational hardware requires a total of five wall-mounted ultrasound beacons, confining the SPHERES VERTIGO system to one room. This ultimately limits its overall use, and essentially forces SPHERES VERTIGO to remain as an experimental testbed [4], [10]–[12]. Today, researchers are working to expand the system's navigational range by incorporating Google's Project Tango – a mobile device that can track 3D motion allowing autonomous navigation within a building [13]. Researchers hope that once this hybrid Project Tango and SPHERES system, also known as "Smart SPHERES," has been successfully implemented, the SPHERES nanosatellite will ultimately be able to traverse the entire ISS, performing interior maintenance and inspections.

### C. Collision Avoidance and Docking Algorithms

Since the SPHERES nanosatellite's first microgravity guidance, navigation, and control experiment, there have been three classes of algorithms pertaining to collision avoidance and docking that have emerged: metrology, control, and autonomy [14].

*1) Metrology Algorithms:* The metrology algorithms were implemented using a SPHERES-specific interface, and utilized a series of Extended Kalman Filters to obtain the system's state vector from the sensor outputs. While this approach has been typically utilized in position, attitude, and determination systems, recent literature has suggested that the second and third classes have demonstrated greater success and popularity in the areas pertaining to collision avoidance and docking algorithms [14].

*2) Control Algorithms:* The control algorithm class involves both closed-loop controls and path-planning algorithms. One prominent control algorithm that has been frequently tested is the glideslope algorithm. This algorithm is a hybrid

between a path-planning and a velocity-control algorithm, where the incoming spacecraft is given commands to slow its velocity as it approaches its target [14]–[17]. The glideslope algorithm was the first autonomous docking algorithm to successfully attach an incoming spacecraft to its tumbling target, and its simple, yet robust, controller makes this algorithm easy to store aboard SPHERES [15]. Another algorithm that has also seen success is the "safe" trajectory algorithm. This innovative algorithm computes a pre-planned trajectory using the solution from a Mixed-Integer Linear Program, and, using this pre-computed trajectory, is able to optimize fuel and avoid incoming obstacles [15]. However, while this algorithm is guaranteed to produce a safe trajectory, its overall complexity requires it to be computed on an external computer. Then once the computations have completed, the final trajectory is transferred to SPHERES. This entire computational process creates an approximate nine second delay, and can potentially create a catastrophic outcome if the spacecraft is tumbling towards an immediate collision. To remedy this solution, researchers have begun to trade trajectory and fuel optimality for computational time, and can reduce the total computation time to about $0.17$ seconds [15]. Lastly, the close point of approach algorithm has been demonstrated to be both compact and computationally efficient, and has served as a background safety routine for the high school SPHERES Zero Robotics program [17]. While all three aforementioned algorithms have accurately performed numerous tests pertaining to collision avoidance and docking, each algorithm is associated with its own specific set of pros and cons. As it currently stands, researchers have yet to find a way to optimize fuel usage, pre-planned trajectories, and computational power, and thus must decide which factors are most important for any given mission [14]–[17].

*3) Autonomous Algorithms:* Finally, the autonomous algorithm class is used to execute the control class algorithms and determine the current mode of operation [14]. As a result, the glideslope, "safe" trajectory, and close point of approach algorithms all utilize autonomy to properly perform their respective procedures. For this particular project, the authors recognize that autonomous collision avoidance and docking algorithms are an integral part in achieving a successful vision-based autonomous docking simulation. However, due to time limitations, the authors have chosen to simply focus on developing a virtual simulation for autonomous docking, and will readdress collision avoidance at a future date.

### *D. Visual Simulations*

When testing a computer-vision based navigational algorithm, the first step is to simulate the algorithm in a comparable environment to test the system's design. For example, the research team at the West Virginia Robotic Technology Center (WVRTC) facility created a virtual environment to attempt to test their vision based pose estimation system, consisting of a monocular camera mounted to the tip of a robotic manipulator [17]. In doing so, the designers were able to exhibit how their system parameters effected their system's performance, and make dynamic modifications to their computer vision algorithms. Similarly, when the NASA JPL scientists simulated NASA's Mars Sample Return Mission, particularly the high-risk operation of capturing the Orbiting Sample, this research team performed numerous virtual simulations to test as many situations as possible, before implementing the final algorithm [18]. As a result, advanced virtual simulations have become an integral procedure in emulating the system's environment, using tools such as EDGE and OpenCV, to aid in the detection of algorithmic deficiencies. The authors recognize that this is an optimal way for testing a vision-based autonomous navigation algorithm, and will thus utilize a virtual simulation to test and demonstrate the navigational and docking algorithm's capability.

### *E. Literature Review Conclusions*

Since the AERCam Sprint, there have been numerous advancements pertaining to small semi-autonomous and autonomous satellites, particularly in the areas regarding guidance, navigation, and control. Through the glideslope, safe trajectory, and close point approach algorithms, researchers have attained many viable options for docking and collision avoidance, but there still remains room for improvement. With the need for both computational efficiency and vehicle safety, there remains a wide area of computer science that researchers are continuing to explore, to ultimately find a solution to this problem. However, the most intriguing aspect of these programs is the fact that all satellites, excluding the AERCam Sprint, have been solely implemented in interior environments, when the AERCam Sprint was originally designed to assist astronauts in Extra Vehicular Activities (EVA) [2], [3]. While the MIT Space Systems Laboratory is the closest team to attaining a functioning exterior inspection satellite through their SPHERES-X proposal, there still remains a relatively open field that has yet to be explored [4]. In this paper, the authors hope to take preliminary steps towards creating an exterior inspection satellite by developing machine vision navigation algorithms that could later be applied to a completed satellite system.

## III. Design Specifications

Assume that there is a preexisting small satellite with machine vision navigation capabilities, and the following features are attached to implement the autonomous machine vision algorithms:

- Two color stereo cameras attached to the side of the satellite.
- A wide angle 2.8mm f/1.3 CCTV lens with manual iris and focus, capable of a $96^\circ$ horizontal angle and a $71^\circ$ vertical field of view.
- Two illuminating light-emitting diodes.
- A computer capable of 1.2 GHz x86 Via Nano Processor, 4GB RAM, 1 MB L2 Cache, SSE3.
- A 802.11n network card.

[11], [12], [17], [19]

For this simulation, assume that the physical dynamics of the satellite have been previously implemented in secondary algorithms, and thus have no impact on the machine vision based algorithm. Furthermore, assume that this small satellite

has reaction control wheels for attitude control and a sufficient supply power to properly conduct all virtual test simulations. Although the inspector will also have enough fuel, it will be considered finite to determine accuracy of the different control schemes.

*****ANYTHING ELSE YOU GUYS THINK WE SHOULD ADD???

## IV. The Environment

### A. The Spacecraft Environment

Space is an extremely dangerous environment with many challenges of different natures and magnitudes. Some effects on spacecraft can arise from radiation, space debris and meteoroid impact, upper atmospheric drag, spacecraft electrostatic charging, and many other factors. While the resulting dynamics of these phenomena have various degrees of effect on small spacecraft, these are outside the scope of this work. For the purposes of the study, all of these factors are ignored. It is assumed here that the spacecraft is operating under nominal conditions in an otherwise empty environment.

### B. Dynamical Equations of Motion

The Clohessy-Wiltshire equations were used to model the dynamics of the spacecraft while it was in close proximity to the target vehicle. The Clohessy-Wiltshire equations describe a simplified model of orbital relative motion, in which the target is in a circular orbit, and the chaser spacecraft is in an elliptical or circular orbit. This model gives a first-order approximation of the chaser's motion in a target-centered coordinate system. It is used here in planning rendezvous of the chaser with the target [20].

These equations of motion can be expressed as

$$\begin{aligned} \ddot{x} &= 3n^2 + 2n\dot{y} \\ \ddot{y} &= -2n\dot{x} \\ \ddot{z} &= -n^2 z \end{aligned} \tag{1}$$

and have the closed form solution given by

$$\begin{aligned} x(t) &= (4 - 3\cos nt)x_0 + \frac{sinnt}{n}\dot{x}_0 + \frac{2}{n}(1 - \cos(nt))\dot{y}_0 \\ y(t) &= 6(\sin nt - nt)x_0 + y_0\frac{-2}{n}(1 - \cos nt)\dot{x}_0 \\ &\quad + \frac{4\sin nt - 3nt}{n}\dot{y}_0 \\ z(t) &= z_0 \cos nt + \frac{\dot{z}_0}{n}\sin nt \end{aligned} \tag{2}$$

where

$$n = \sqrt{\frac{\mu}{a_t^3}} \tag{3}$$

and $a_t$ is the semi-major axis of the target vehicle's orbit and $\mu$ is the standard gravitational parameter.

## V. Simulation

The virtual simulation was driven with a variety of software packages. The dynamics and control systems were driven by several Python modules, while the visualization was rendered in EDGE. EDGE is a graphics display tool developed at NASA's Johnson Space Center that combines key elements from graphics software developed for the space shuttle and the International Space Station programs and adapts them for integration with other engineering simulations and facilities [21].

EDGE makes use of a node tree to structure data, objects, and models. Each node has many properties, the most notable of which are position, orientation, and the node's parent. Each node's position and orientation are defined relative to it's parent's position and attitude. A node's parent can be changed to a different node, at which time the node's position and attitude are automatically updated to the correct values.

Three Python Python 2.7.9 scripts were developed for use in the virtual simulation:

***cv***
: A computer vision image processing module which takes advantage of OpenCV-Python version 2.4.9.

***dynamics***
: A dynamics and controls module.

***dcomm***
: A wrapper library developed to interface with EDGE's C++ DCOMM library.

### A. Python Modules

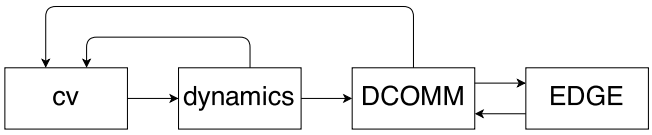


Fig. 3. Block diagram of software package interactions

*1) cv:* The *cv* module takes advantage of Python bindings for OpenCV. OpenCV (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision. This module takes the live video feed from EDGE and attempts to identify features visible on the target spacecraft. The features of the target are supplied to the module a priori, and these descriptions were used to estimate the 3D pose of the target vehicle relative to the spacecraft's camera. This pose information is passed on to the *dynamics* module

*2) dynamics:* The *dynamics* module drives the dynamics and control of the simulation. The movement of the spacecraft around the target craft is modeled by the Clohessy-Wiltshire equations, see section IV-B. This module controls how the spacecraft operated in its different modes. Depending on pilot input, the spacecraft can navigate purely on line of sight, laser guided sensors to approach and hold a distance from a target, make use of the optical camera and machine vision to identify and dock with docking ports, or display guidance for a human pilot.

*3) DCOMM:* The *DCOMM* module was previously developed to network between Python scripts and EDGE. The Python DCOMM interface allows a user to call various C++ functions from EDGE's DCOMM module and communicate with an EDGE server. Users can move and rotate nodes, and can also change a node's parents, units, and principal axis definitions. Commands to set the spacecraft's attitude and position were sent from the *dynamics* module through DCOMM and passed on to EDGE. This module was also responsible for requesting and handing off the video feed from the EDGE server.

## VI. Summary

The conclusion goes here.

## References


[1] R. Nugent, R. Munakata, A. Chin, R. Coelho, and D. J. Puig-Suari, "The cubesat: The picosatellite standard for research and education," 2008. [Online]. Available: http://gwww.cubesat.org/images/More_Papers/cps2008.pdf

[2] K. Dismukes, "AERCam Sprint," 2003. [Online]. Available: http://spaceflight.nasa.gov/station/assembly/sprint/

[3] NASA, "NASA Johnson Space Center's Miniature Autonomous Extravehicular Robotic Camera (Mini AERCam)," 2002. [Online]. Available: aercam.jsc.nasa.gov_aercam.pdf

[4] H. J. Kramer, "ISS utilization: SPHERES (Synchronized Position Hold Engage Reorient Experiment Satellites)," 2002. [Online]. Available: https://directory.eoportal.org/web/eoportal/satellite-missions/i/iss-spheres

[5] M. Machula and G. Sandhoo, "Rendezvous and docking for space exploration," January 30 - Februrary 1, 2005 2005.

[6] Darpa.mil, "Developing technologies for more flexible, cost-effective satellite operations in geo," 2015. [Online]. Available: http://www.darpa.mil/Our_Work/TTO/Programs/Phoenix.aspx

[7] J. W. Thomas Guay, "International space station - quantifying and reducing risk following orbital debris penetration," September 24-26, 1996, 2008 1996.

[8] NASA.gov, "Astronauts complete spacewalk to repair ammonia leak, station changes command," 2013. [Online]. Available: http://www.nasa.gov/mission_pages/station/expeditions/expedition35/e35_051113_eva.html

[9] S. E. Frederickson, S. Duran, and J. D. Mitchell, "Mini AERCam inspection robot for human space missions," 28-30 Sept. 2004 2004.

[10] B. E. Tweddle, E. Muggler, A. Saenz-Otero, and D. W. Miller, "The SPHERES VERTIGO Goggles: Vision based mapping and localization onboard the International Space Station," in *Proceedings of i-SAIRAS (International Symposium on Artificial Intelligence, Robotics and Automation in Space)*, 2012, Conference Proceedings. [Online]. Available: http://ssl.mit.edu/spheres/library/tweddle-isairas12.pdf

[11] D. Fourie, B. E. Tweddle, S. Ulrich, and A. Saenz-Otero, "Flight results of vision-based navigation for autonomous spacecraft inspection of unknown objects," *Journal of Spacecraft and Rockets*, vol. 51, no. 6, p. 10, 2014.

[12] ——, "Vision-based relative navigation and control for autonomous spacecraft inspection of an unknown object," August 19-22, 2013 2013.

[13] M. Williams, "Google's Project Tango headed to International Space Station," 2014. [Online]. Available: http://www.cio.com/article/2377746/smartphones/google-s-project-tango-headed-to-international-space-station.html

[14] E. M. Kong, A. Saenz-Otero, S. Nolet, D. S. Berkovitz, and D. W. Miller, "SPHERES as a formation flight algorithm development and validation testbed: Current progress and beyond," 2004. [Online]. Available: http://ssl.mit.edu/spheres/library/SPHERESFFPaper72.pdf

[15] A. Fejzic, "Results of SPHERES microgravity autonomous docking experiments in the presence of anomalies," in *Proceeding of the 59th IAC (International Astronautical Congress)*, 2008, Conference Proceedings. [Online]. Available: http://ssl.mit.edu/spheres/library/IAC_2008_AmerFejzic.pdf

[16] A. Saenz-Otero, G. Aoude, M. M. Jefrey, S. Mohan, A. Fejzic, J. Katz, C. Edwards, and D. W. Miller, "Distributed satellite systems algorithm maturation with SPHERES aboard the ISS," in *Proceedings of the 59th IAC (International Astronautical Congress)*, 2008, Conference Proceedings. [Online]. Available: http://ssl.mit.edu/spheres/library/SPHERES_IAC2008.pdf

[17] A. F. Velasquez, G. Marani, T. Evans, M. R. Napolitano, J. A. Christian, and G. Doretto, "Virtual simulator for testing a vision based pose estimation system for autonomous capture of satellites with interface rings," pp. 1597–1602, 2013 2013. [Online]. Available: http://www.csee.wvu.edu/~gidoretto/publications/velasquezMENCD13med.pdf

[18] B. E. Tweddle, J. McClellan, G. Vulikh, J. Francis, and D. Miller, "Relative vision based navigation and control for the Mars sample return mission: Capturing the Orbiting Sample," May 18-20, 2011 May 2011. [Online]. Available: http://ssl.mit.edu/spheres/library/tweddle_MOSR_SFFMT_2011.pdf

[19] Fujinon, "Fujinon yf2.8a-2 wide angle 2.8mm f/1.3 cctv lens with manual iris and focus, for 1/3-inch and 1/4-inch ccd cameras with cs-mount," 2015. [Online]. Available: http://www.bhphotovideo.com/c/product/404186-REG/Fujinon_YF28A2_YF2_8A_2_Wide_Angle_2_8.html

[20] Ocampo, "Clohessy-Wiltshire equations," 2015. [Online]. Available: http://courses.ae.utexas.edu/ase366k/cw_equations.pdf

[21] NASA, "Enhanced graphics tools for advanced 3D simulations," 2015. [Online]. Available: http://www.nasa.gov/centers/johnson/techtransfer/technology/MSC-24663-1-doug-edge.html

[22] Opencv.org, "Opencv," 2015. [Online]. Available: http://opencv.org

[23] F. Hoots and J. Thomas F. Starchville, "Debris risk assesment process," August 18-21, 2008 2008.