SEMI-AUTONOMOUS INSPECTION WITH A NEUTRAL BUOYANCY FREE-FLYER

by

Catharine Linet Rea McGhan

Paper submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Science
2006

Advisory Committee:
Assistant Professor Ella M. Atkins, Advisor
Associate Professor Robert M. Sanner

Abstract

The ability to perform autonomous in-space inspection and servicing activities is of significant importance to the future space program. The University of Maryland has developed a pair of free-flying robotic platforms that operate in a neutral buoyancy environment to support such space-based activities. Numerous tests have been performed with this platform, including demonstration of a robust closed-loop controller for the rotational degrees of freedom. This rotational controller has been observed to substantially reduce operator workload through stable maintenance and tracking of attitude commands. However, in previous work the operator has directly controlled translational motion due to the absence of an inertial navigation system such as GPS in the underwater environment. Recently, a Visual Positioning System (VPS) has been developed for the neutral buoyancy environment that provides translational position and velocity estimates of the free-flying inspection vehicle. This paper describes the development and testing of a six degree-of-freedom control system for this inspection vehicle and analyzes its performance during a variety of stationkeeping, pilot-assisted, and autonomous trajectory-tracking tasks.

I. Introduction

Space inspection and servicing are important technologies that must be developed to support future space operations. Robotic systems are not yet prepared to perform these tasks without operator assistance, particularly given the safety-critical proximity operations environment in which an inspection or servicing robot would be expected to maneuver. Direct teleoperation of a free-flying robotic assistant requires extensive real-time feedback and generates a very high workload for the operator(s), as was demonstrated during the flight of the AERCam system[1] that was directly controlled by astronauts with an unobscured view of the operating environment. Researchers[2,3,4] have begun to incorporate closed-loop controllers into free-flying robotic platforms to augment stability, resulting in a significant decrease in operator workload.[5] This paper describes the implementation and testing of a full six degree-of-freedom control system for the Supplemental CAMera Platform (SCAMP), a free-flying neutral buoyancy robotic vehicle. The goal of this research is to enable fully-autonomous three-dimensional station-keeping and traversal between specified waypoints, a capability that will enable a variety of complex robotic and collaborative human-robot space operational scenarios to be explored in a neutral buoyancy environment.

The neutral buoyancy simulation concept was developed to provide a long-term three-dimensional work environment to train astronauts and more recently to demonstrate robotic systems. Alternative environments, including lower-friction air bearing tables and KC-135 zero-g simulation, do not experience the overhead of operating underwater but cannot provide long-term, three-dimensional, (simulated) zero-gravity conditions. History has repeatedly demonstrated the importance of experimentally validating operational concepts prior to launch. Robotic systems are being developed to support human space exploration by performing complex inspection and servicing tasks alone or potentially in collaboration with astronauts. These robots will operate in proximity to safety-critical hardware, and ultimately, they will likely operate in the same physical environment as the astronauts they support. The SCAMP vehicle pair has been developed at the University of Maryland to support inspection tasks or to simply provide a maneuverable camera view for an astronaut or controller of a larger-scale manipulation-based robotic system. In future tests, it is anticipated that SCAMP will also operate collaboratively with astronauts (divers) performing tasks analogous to those performed during Extra-Vehicular Activity (EVA) operations. However, to support such operations, SCAMP must first be equipped with the ability to accurately follow a trajectory, the topic of this paper.

Other robotic systems such as AERCamII and the Personal Satellite Assistant (PSA) designed for interior operations have been augmented to accurately compute inertial position and orientation estimates for use in closed-loop control systems.[6,7,8] Research on these platforms has spanned modes ranging from teleoperation through autonomous task planning, but with limited hardware-based demonstration of fully-autonomous operations.[6,9] AERCamII was designed to rely upon GPS and other support sensors to navigate on-orbit.[6] Conversely, PSA requires fiducial markings on the internal walls of a habitat to determine its state.[8] The SPHERES system[2] also has a sophisticated controller, but its primary focus is on demonstration of satellite formation flight operations rather than inspection, servicing, or collaborative operations, thus relative navigation of the SPHERES "spacecraft" is emphasized.

In previous work,[3] a variety of standard and adaptive attitude closed-loop controllers were deployed and evaluated on SCAMP and its predecessor platforms using measurements from an onboard inertial sensor suite. However, translational closed-loop control did not previously exist, because accurate position and velocity estimates for the underwater neutral buoyancy environment have only recently become available via a Visual Positioning System (VPS) in which a series of calibrated cameras affixed to the tank walls provide real-time translational state estimates.[10] Below, we overview the SCAMP system and its support equipment, including VPS and the vehicle control station. The operational modes of the vehicle – open loop, stationkeeping and pilot-assisted flight, and trajectory-tracking – are then discussed. Next, the algorithms for rotational and translational closed-loop controllers are presented. Gain selection for the translational closed-loop controller is then discussed. Finally, three test cases in trajectory tracking are presented, and the results are given in terms of controller stability and system accuracy.

II. SCAMP Test Platform and Support Equipment

Figure 1 shows one of the two SCAMP vehicles in the University of Maryland's Neutral Buoyancy Research Facility – a 50 ft. diameter, 25 ft. deep heated tank allowing unrestricted diver and robotic activity. SCAMP, the test vehicle for this work, is equipped with an onboard PC-104 computer stack running the QNX real-time operating system. Both SCAMP vehicles have six thrusters aligned with the primary vehicle axes, an onboard camera, and a full inertial measurement unit (IMU) sensor suite. The sensor suite consists of a triaxial magnetometer, triaxial accelerometer, and three angular rate gyros, which measure the orientation of the vehicle using a TRIAD algorithm.[4] With this system, rotational closed-loop control is

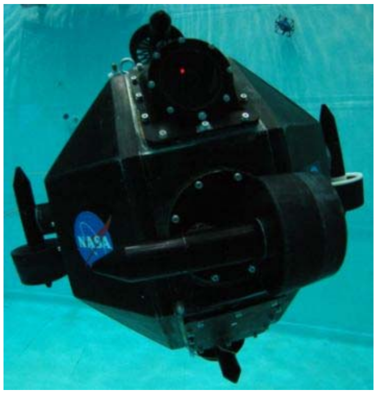
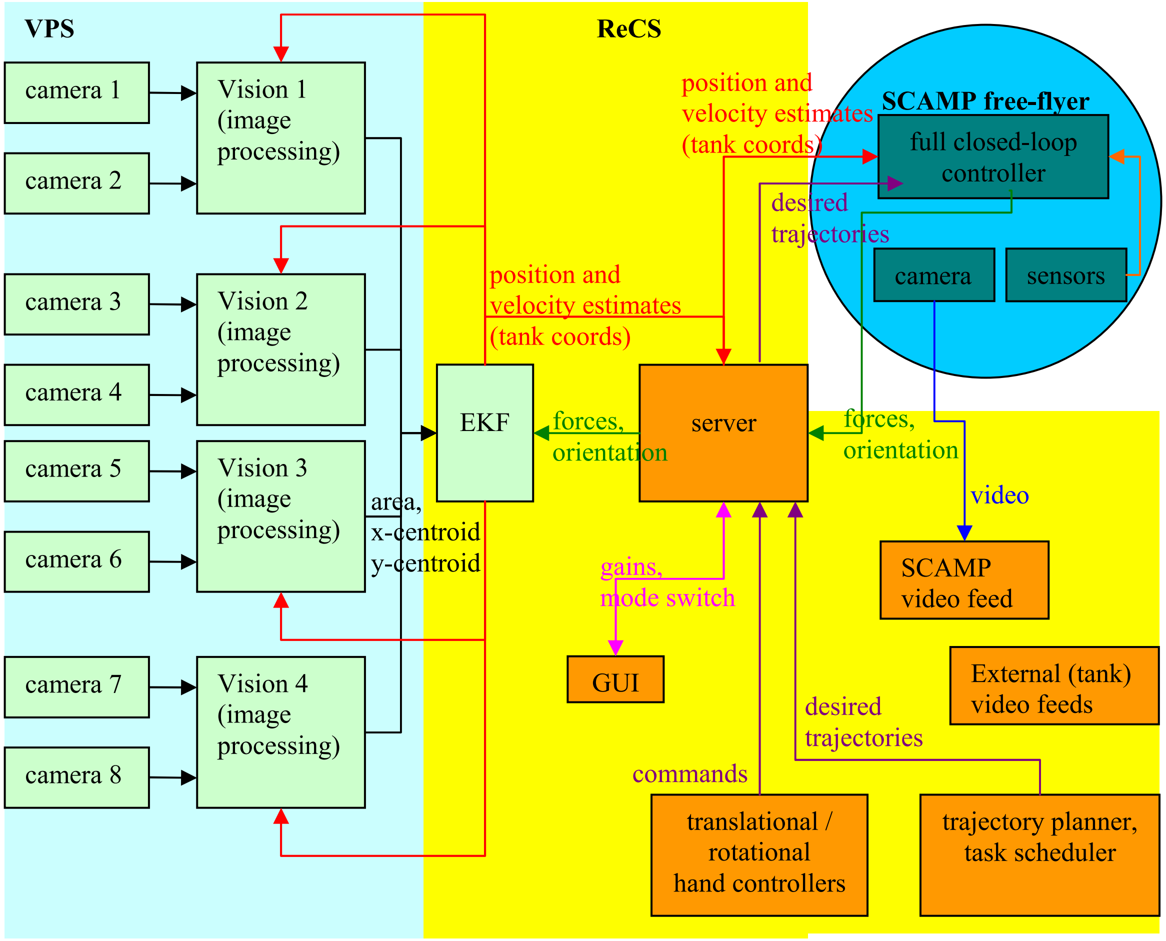


Figure 1: SCAMP Vehicle

performed completely onboard the vehicle. The vehicle sends and receives telemetry over a fiber optic Ethernet link, and transmits the onboard video feed over another fiber. Physically, the vehicle is approximately symmetric, which allows for simplification of the model and controller design.



VPS
ReCS
camera 1
camera 2
Vision 1 (image processing)
camera 3
camera 4
Vision 2 (image processing)
camera 5
camera 6
Vision 3 (image processing)
camera 7
camera 8
Vision 4 (image processing)
area, x-centroid y-centroid
EKF
position and velocity estimates (tank coords)
forces, orientation
server
gains, mode switch
GUI
commands
translational / rotational hand controllers
desired trajectories
trajectory planner, task scheduler
SCAMP free-flyer
full closed-loop controller
camera
sensors
position and velocity estimates (tank coords)
desired trajectories
forces, orientation
video
SCAMP video feed
External (tank) video feeds

Figure 2. Full System Diagram with Closed-Loop Control

The Reconfigurable Control Station (ReCS) provides the interface between operator and the SCAMP vehicle pair. Illustrated in Figure 2, ReCS hosts a pair of standard space shuttle 3-DOF hand controllers (one translation, one rotation) and a suite of monitors displaying SCAMP's onboard video as well as a series of video feeds from around the neutral buoyancy tank. The hand controllers are used to command vehicle motion in open-loop and pilot-assist modes but are unused when in autonomous or semi-autonomous operational modes. The ReCS computer generates a GUI with real-time telemetry data and an operator command interface. ReCS communicates with SCAMP and also executes the Extended Kalman Filter (EKF) that computes SCAMP's position and velocity estimates based on VPS camera data. This computer system is also responsible for logging all test data and managing all SCAMP activities, either from user commands or discrete waypoint-based trajectories to autonomously follow.

Also shown in Fig. 2, the University of Maryland Neutral Buoyancy Research Facility has been outfitted with a Visual Positioning System (VPS) consisting of eight CCD cameras in two tiers around the tank and four dedicated image-processing computer systems that sample and process video frames at 10-12 Hz per camera. An EKF (on ReCS) is used to combine data extracted from the processed images in the individual camera frames, while propagating the vehicle state with the thruster force commands in the (inertial) tank frame. Previous research has demonstrated that VPS provides centimeter-level accuracy of the vehicle's inertial position.[10]

A robust SCAMP attitude controller[11,12] has already been developed and extensively tested. Because previous work determined that a non-adaptive attitude controller performed comparably to a variety of adaptive control strategies, the work described in this paper does not utilize SCAMP's attitude parameter adaptation capabilities but instead uses a PID (proportional-integral-derivative) attitude control law. This work focuses on the development of a simple translational control law and vehicle model, and then gives a comprehensive performance evaluation of both.

III. Vehicle Guidance, Dynamics, and Control

This section describes SCAMP's guidance modes, dynamic model, and control system design. The attitude dynamic model, numerical parameters, and control algorithms have been adopted from previous work,[4] with minor numerical value adjustments made in this work to account for vehicle equipment upgrades. Work to develop a translational model and evaluate it via simulation has been reported.[13] However, because of recent upgrades to higher-output thrusters, voltage-to-force thrust calibration curves required reevaluation using previously obtained measurements. A far more accurate translational model for the vehicle was then constructed using experimental data. This model was used to refine the controller gains and later served as a baseline comparison for experimental data. Controller gains for translational closed-loop control were determined experimentally through real-time observation and offline data analysis, as will be described below.

Navigation data is supplied from two processes: (1) 3-D translational position and velocity estimation performed at the surface control station by an EKF that relies on VPS data, and (2) attitude estimation from IMU data via a local (onboard) software thread. A variety of operational guidance modes are available, including 1) open-loop teleoperated, 2) stationkeeping and/or pilot-assist (augmented stability), and 3) autonomous operation (in which a reference trajectory is followed). Each of these modes is described below, followed by a discussion of the rotational and translational control law designs.

A. Control Modes, Guidance

As shown in Fig. 2, the surface control station (ReCS) is responsible for providing guidance commands to the SCAMP vehicle. In open-loop and pilot-assist modes, a pilot "flies" the vehicle with the two space shuttle hand controllers, while for station-keeping and autonomous trajectory tracking modes, ReCS maintains and updates the commanded reference trajectory based on waypoints (e.g., station to hold) and trajectories currently provided from time history files.

1. Open-loop

Open-loop piloting of the vehicle is the first mode that was implemented on the vehicle. In this mode, desired forces and torques (in body coordinates) are directly proportional to the deflections of the translational and rotational hand controllers, respectively. The operator can then fly the vehicle directly as a pilot using the onboard video view.

2. *Pilot-assist (closed-loop, semi-autonomous)*

In this mode, hand controller deflections are proportional to the desired angular and linear velocities for the vehicle (defined in the body frame). These velocities, through integration, implicitly specify a trajectory defined as a sequence of positions and orientations. A feedback control algorithm resident on the vehicle compares the estimated positions and velocities to the desired quantities at each instant, and automatically computes the forces and torques required to drive the difference between the estimated and desired states to zero. When no commands are received from the pilot, this mode defaults to stationkeeping.

3. *Reference trajectory tracking (autonomous)*

Reference trajectory tracking mode is the final capability necessary to enable completely autonomous operation of SCAMP. For this work, the desired angular and linear velocity, quaternion, and position time histories (or a subset of these) are provided by an input file; ultimately, we will generate these trajectories with autonomous path planning and task scheduling algorithms based on mission goals, interaction with other astronaut/robotic entities, and contingencies to which the system must respond. Regardless of how it is created, however, the desired trajectory is then tracked by the onboard feedback control algorithm as in the pilot-assist mode described above. The principle difference between these two modes is the source of the trajectory to be followed: in pilot-assist, the pilot directly specifies the trajectory to fly via hand controller inputs. In autonomous-operations mode, these trajectories are either pre-specified or computed by software from higher level goal descriptions, safety considerations, and real-time task-level feedback.

B. State Feedback, Control

In modes (2) and (3) described above, forces and torques are automatically computed from the specified desired positions, orientations, and corresponding velocities using the following feedback control algorithms. Torque is computed from:

$$\begin{aligned} {}^B\tau &= -K_{Dr} * {}^B\tilde{\omega} - K_{Pr} * {}^B\tilde{e} - K_{Ir} * {}^B\tilde{l} \\ {}^B\tilde{\omega} &= {}^B\omega - {}^B_D C(\tilde{q}) \cdot {}^D\omega_d \end{aligned} \tag{1}$$

where: ${}^B\tau$ is torque applied to vehicle (body frame)

K_{Xr} are rotational control gains, *X*→{*D*=derivative, *P*=proportional, *I*=integral}

${}^B\omega$ is the current angular velocity of the vehicle (body coordinates)

${}^D\omega_d$ is current desired angular velocity

$\tilde{q}$ is the quaternion representation of the orientation of the vehicle relative to the desired orientation

${}^B\tilde{e}$ is the "vector part" of $\tilde{q}$

${}^B\tilde{l}$ is the integral of ${}^B\tilde{e}$

${}^B_D C(\tilde{q})$ is the rotation matrix which converts from the current desired frame to the current body frame

Force is computed from:

$$\begin{aligned}
{}^{B}f &= {}^{B}_{I}C(q)\cdot\left(-K_{Dt}*{}^{I}\tilde{v} - K_{Pt}*{}^{I}\tilde{x} - K_{It}*{}^{I}\tilde{z}\right)\\
{}^{I}\tilde{v} &= {}^{I}v - {}^{I}v_{d}\\
{}^{I}\tilde{x} &= {}^{I}x - {}^{I}x_{d}
\end{aligned} \tag{2}$$

where: ${}^{B}f$ is the force applied to vehicle (body frame)

K_{Xt} are translational control gains, X→{D=derivative, P=proportional, I=integral}

${}^{I}v$ is current translational velocity of vehicle (tank frame)

${}^{I}v_{d}$ is desired translational velocity

${}^{I}x$ is current position of vehicle (tank frame)

${}^{I}x_{d}$ is desired position

${}^{I}\tilde{z}$ is the integral of ${}^{I}\tilde{x}$

q is quaternion representation of vehicle orientation relative to tank frame

${}^{B}_{I}C(q)$ is the rotation matrix which converts from tank frame to current body frame

These are essentially PID control strategies, with some nonlinearities to convert between the multiple reference frames involved. The rotational controller uses (Wie, et al)[14] to form a "proportional" error from the quaternion representation of the attitude deviation, and the corresponding derivative term is the angular velocity deviation resolved into a common body frame. The integral term in the rotation controller is evaluated directly from the attitude deviation used for the proportional error. The translation controller computes the tracking errors in tank coordinates, then rotates these vectors into body coordinates to determine the required body axis forces.

Once the forces and torques are found, they are converted to specific forces to be exerted by each of the six individual thrusters. This conversion is a vehicle-specific function of thruster locations and offsets.

IV. Results

A variety of neutral buoyancy experiments were conducted to evaluate the performance of the SCAMP 6-DOF controller in all operational modes described above. The results shown below are primarily indicative of position tracking only – results for the attitude controller have been obtained and previously presented elsewhere,[3,4,5] but are included for completeness. From previous experiences, it has been determined that during open loop operation the vehicle was controllable but only with substantial effort by the pilot, particularly with the existence of buoyancy offsets.

A. Translational Gains Tuning

The translational gains were tuned by creating a translational model based on experimental data and determining several viable gain sets from the model response; final tuning and gain selection was accomplished through vehicle testing. The translational model was assumed to approximate a second-order linear system given by:

$$f = m*\ddot{x} + D_{t}*\dot{x} \tag{3}$$

where: m is the mass of the vehicle

D_t is the translational drag

This corresponds to the transfer function:

$$G_p(s) = \frac{1}{m * s^2 + D_t * s} \tag{4}$$

If $f = K_{Pt} * (x_d - x)$, as is the case for a purely proportional response, then the controlled response has the standard second order transfer function:

$$T(s) = \frac{K_{Pt}}{m * s^2 + D_t * s + K_{Pt}} \tag{5}$$

and, by observing the features of a typical step response, such as in Figure 3 below, one can back out m and D_t:[15]

$$\begin{aligned} T(s) &= \frac{K_{Pt}/m}{s^2 + (D_t/m) * s + (K_{Pt}/m)} = \frac{K}{s^2 - 2 * \sigma * s + (\omega_d^2 + \sigma^2)} \\ \sigma &= \ln(M_p)/t_p \\ \omega_d &= \pi / t_p \\ m &= K_{Pt}/(\omega_d^2 + \sigma^2) \\ D_t &= -2 * \sigma * m \end{aligned} \tag{6}$$

where: ω_d is the damped frequency of the system in radians/second

M_p is the overshoot (measured directly from experiment)

t_p is the peak time in seconds (measured directly from experiment)

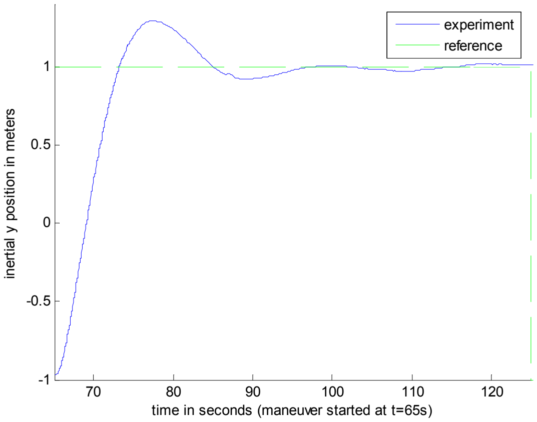


Figure 3. Step Response to K_{Pt}=16 in Inertial y Direction Over Time

For this response, M_p and t_p are ~0.1465 and ~12.4 seconds. Several identical proportional-only experiments were conducted, with different values of Kp, and the computed masses and drags ranged primarily in the intervals m=[140,160] and

D=[30,50]. The values used in the final model (m=150 kg, D_t=40 kg/s, see Equation 7 below) were the average of this range of computed values. This model was used for all three axes due to the physical symmetry of the vehicle.

$$G_p = \frac{1}{150s^2 + 40s} \tag{7}$$

At this point, several promising sets of PID gains were identified in Matlab using this model, with an emphasis on stability and response speed; these sets were experimentally tested and refined, and a final set of gains was selected after offline comparison of experimental data. Table 1 lists the final tuned gains used in the SCAMP controller. Note that these gains were tuned to step responses and not trajectory tracking (comparable to commanding a series of much smaller step responses).

Table 1. Controller Gains for SCAMP

	K_{Pr} (N m/rad)	K_{Dr} (N m/rad/s)	K_{Ir} (N m/rad-s)	K_{Pt} (N/m)	K_{Dt} (N/m/s)	K_{It} (N/m-s)
Gain value	15.0	10.0	3.0	30.0	65.0	0.625

Below are results compiled from experiments in autonomous trajectory tracking mode (3). It is assumed for these results that VPS is giving an accuracy of ~2cm in the position data (under good lighting conditions), consistent with previously reported results.[10]

B. Vehicle Response to Step Command

Figure 4 shows the estimated vehicle response over time in the inertial y-direction for a typical step command in relation to the response predicted by the vehicle model in Equation 7. Overall, the overshoot averages ~10% or less and the position error after 60 seconds still averages ~3% of the step size, which is mirrored by the model. It is notable that there is still a slight discrepancy between the model and the actual response near the peak value: this is probably due to the fact that the actual drag force is nonlinear, while we are using a linear model for comparison.

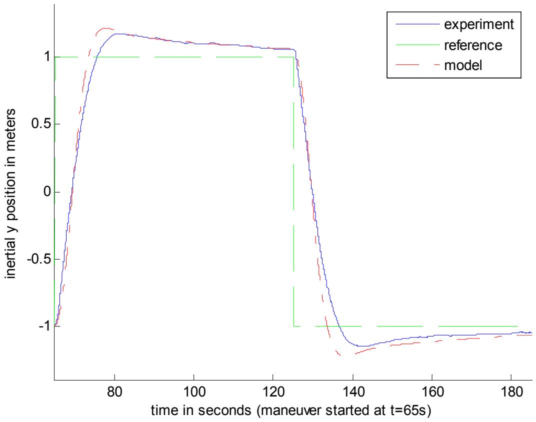


Figure 4. Step Response in Inertial y Direction Over Time

PID control was chosen over PD or P control for translational motion primarily to offset the effects of buoyancy in the z-axis (up/down) direction. This creates a challenge for creating a stable response which both minimizes overshoot and settling time, and remains within the feasible force range of the vehicle's thrusters. Although superior responses could have been achieved in x and y by omitting the integral term, for simplicity the same controller was used for all three translational axes in the experiments reported below.

C. Vehicle Response to General Trajectories

In order to characterize the controller's response to general trajectories, three types of tests were performed: a test of the controller itself by evaluating responses to precomputed trajectories stored onboard the vehicle (simple, continuous), a test of the higher-level trajectory specification mechanism by evaluating responses downlinked to the vehicle in real-time (simple, discrete), and a test of the general vehicle response by commanding translation+rotation trajectories downlinked to the vehicle in real-time (complex, discrete).

1. Internal trajectory input

First, to test just the controller, circular trajectories were executed purely inside the onboard controller; that is, the desired trajectories were generated continuously aboard the vehicle, as opposed to being downloaded discretely from the control station. Figure 5 and Figure 6 show the response to a trajectory command that traces a circle in the x-y plane with constant z position, while Figure 7 and Figure 8 show a similar command in the y-z plane with constant x position. The vehicle starts by moving to the point, [0.0,-0.25,1.50], and then travels the circle counter-clockwise in the plane. Desired positions and velocities were computed for these trajectories; a constant desired orientation (vehicle body frame aligned with the tank inertial frame) is tracked over the trajectory. It is clear from these responses that the controller is internally capable of following simple trajectories with little error (~3-6cm over all three axes). This is significantly higher than the ~2cm error inherent in the accuracy of the VPS system, however the observed tracking is close to that predicted by the mathematical model identified above. Note that the PID gains above were tuned for step response performance, not for tracking. Future work will retune the gains with emphasis on tracking.

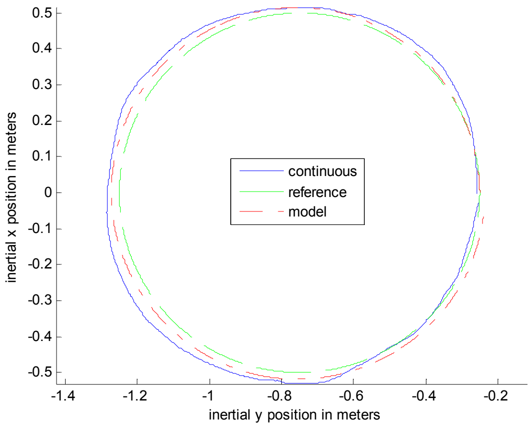


Figure 5. Inertial Position in x-y Plane for Continuous Tracking, Positions and Velocities

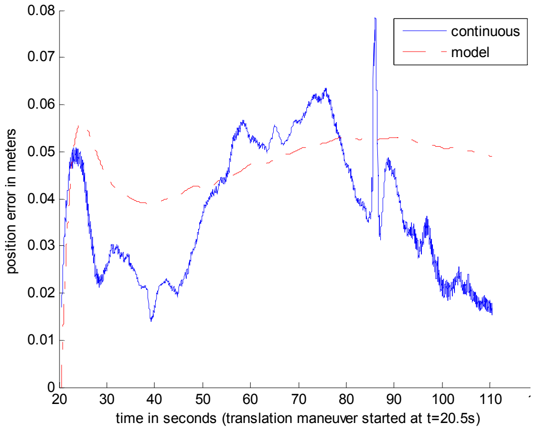
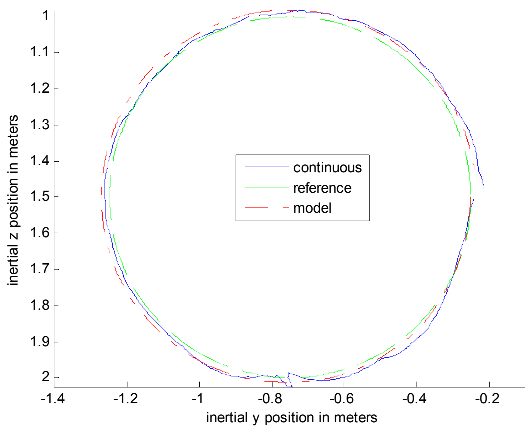


Figure 6. Euclidean Error in Inertial Position Over Time for Continuous Tracking (x-y)



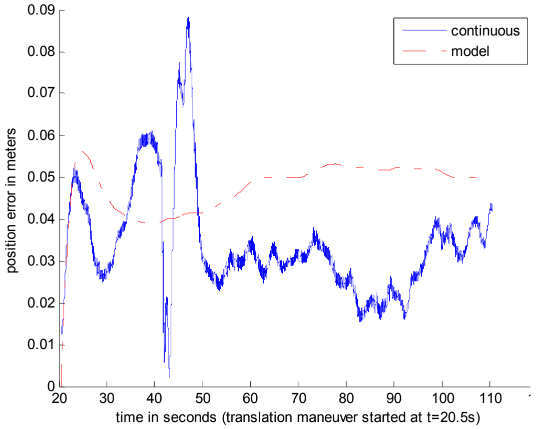
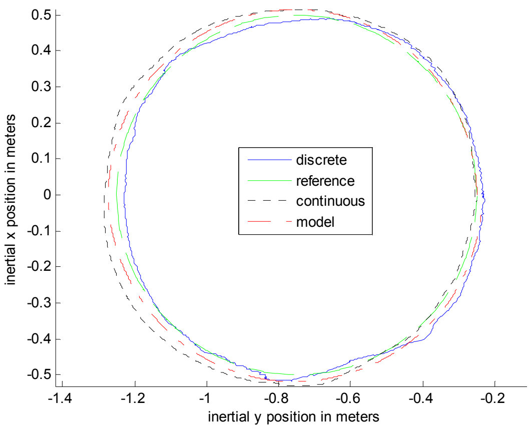


Figure 7. Inertial Position in y-z Plane for Continuous Tracking, Positions and Velocities

Figure 8. Euclidean Error in Inertial Position Over Time for Continuous Tracking (y-z)

2. *External trajectory input*

Next, the same trajectory in the x-y plane was tested, but this was generated as discrete waypoints downlinked in real-time from the control station.



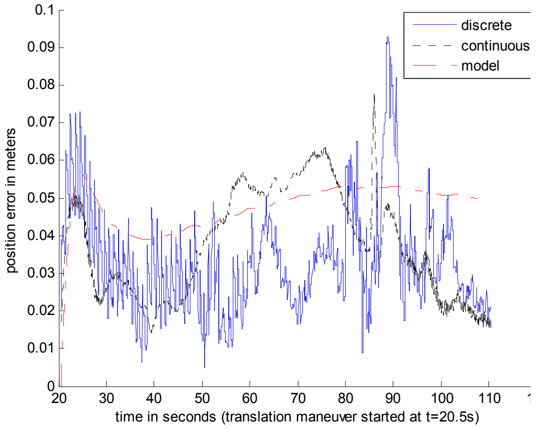


Figure 9. Inertial Position in x-y Plane for Discrete vs. Continuous Tracking

Figure 10. Euclidean Error in Inertial Position Over Time for Discrete vs. Continuous Tracking

Figure 9 and Figure 10 show the vehicle response to the downlinked trajectories overlaid with the results from the continuous and model responses shown in the previous section. The average error is slightly larger in this case due to the time delay, but comparable to the previous response. The high frequency periodic spikes in position error are mainly the result of the discretization of the trajectory. This trajectory specification mechanism – sending a discretized, piecewise-linear trajectory down to the vehicle – offers much more flexibility, but also introduces more error.

3. *General vehicle response*

Finally, the circle trajectory in the x-y plane is performed again, but with nontrivial motion in both translation and rotation – the vehicle first yaws to point towards the center of the circle trajectory in the x-y plane it is about to execute; then, as the vehicle translates counter-clockwise around the circle, it also continues to rotate so that it is continually pointing towards the center of the circle trajectory during the entire movement, simulating an autonomous inspection trajectory. Desired positions, velocities, quaternions, and angular velocities are computed for this trajectory.

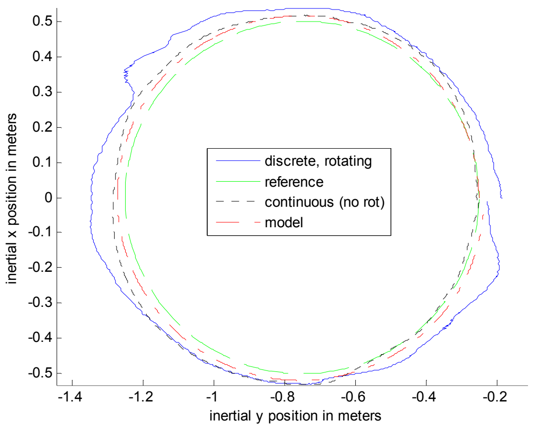


Figure 11. Inertial Position in x-y Plane for Complex Trajectory Tracking

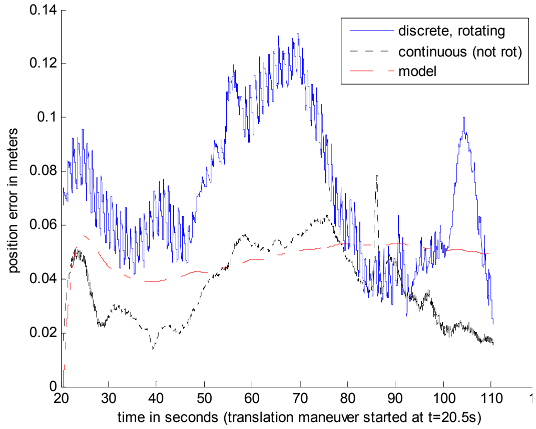


Figure 12. Euclidean Error in Inertial Position Over Time for Complex Trajectory Tracking

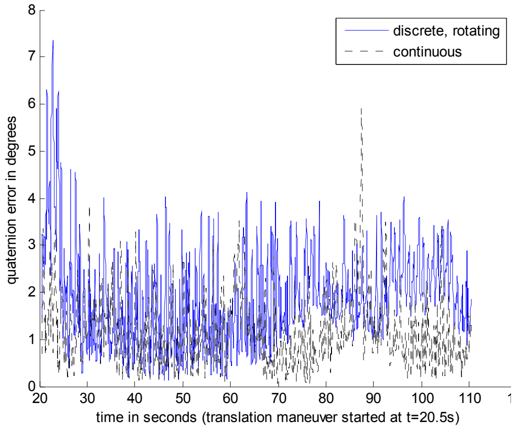


Figure 13. Orientation Error Over Time During Translation Movement

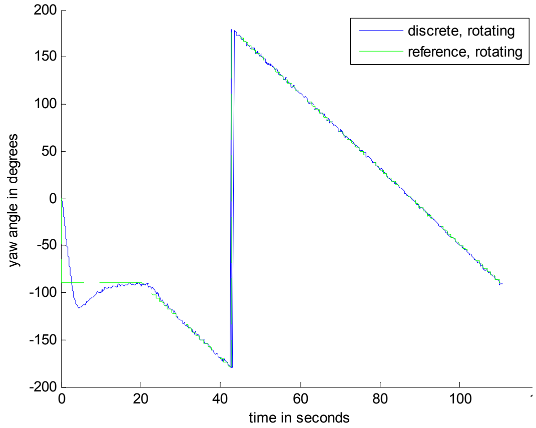


Figure 14. Yaw Angle Over Time for Complex Trajectory Tracking

As seen in Figure 12, the position and orientation errors are approximately twice as large as seen during the translation-only maneuvers. Although the control system has assumed decoupled translation-rotation dynamics, in fact these motions are coupled, primarily due to variations in the individual thrusters. The model (and control algorithm) assumes that each of the eight vehicle thrusters provide identical force output at a given propeller speed. In fact, however, slight variations in the ducting, gearing, and propeller of each thruster result in small differences in force output, creating unwanted coupling between translation and rotation. Thus, translation introduces rotation perturbations and vice-versa, which the controller is not designed to accommodate. Achieving greater accuracy for coupled maneuvers will require a more accurate model of each individual thruster, perhaps determined in real-time using adaptive control techniques, together with a more robust control algorithm capable of correctly compensating for the coupling in the dynamics. There is also the problem of a discrepancy between the orientation of the inertial frame used by VPS, and the orientation of the inertial frame as computed by the vehicle sensors; this is due to separate calibration of the VPS and the onboard sensor system frames of reference, and could also contribute to the error when combined with variations in the individual thrusters.

V. Conclusion and Future Work

In this paper, we presented and evaluated an architecture which utilizes the new VPS system[10] to enable full 6-DOF control for the SCAMP family of neutral buoyancy space simulation vehicles. Previous (pre-VPS) results showed that rotational closed-loop control significantly reduces the workload of pilots performing simulated inspection and docking tasks.[5] The new results with the 6-DOF controller set show that it can further reduce workload in both stationkeeping and in pilot-assist mode, by providing accurate tracking of commanded translational maneuvers. The problem of the coupling between the translational and rotational motion does still need to be resolved by refining the system identification and implementing a more capable control algorithm to increase the tracking accuracy to the limits of the VPS system.

The combination of VPS and the autonomy software currently implemented enables experimental investigation into new strategies for autonomous rendezvous and docking, inspection and damage detection, space repair tasks, and space-based human-robot collaboration. However additional enhancements, currently in progress, are still required before SCAMP will be capable of fully-autonomous operation. First, a three-dimensional trajectory planner must be implemented that meets the dynamic constraints but also provides obstacle avoidance capability. Next, potential coupling issues between these higher-level components and the lower-level controller architecture must be explored. Additional work is also needed to include the ability to re-plan or repair trajectories and reevaluate task sets in highly uncertain environments.

Acknowledgments

The author would like to thank Rob Sanner, Ella Atkins, Dave Akin, Michael Chinn, Rebecca Besser, and members of the University of Maryland Space Systems Laboratory for their support throughout this project. This research was supported in part by an NSF Graduate Research Fellowship.

References

[1] Williams, T., and Tanygin, S., "On-orbit engineering tests of the AERCam Sprint robotic camera vehicle," *Proceedings of the AAS/AIAA Space Flight Mechanics Meeting*, pp. 1001-1020. 1998.

[2] Chung, S.-J., Kong, E., and Miller, D., "Dynamics and Control of Tethered Formation Flight Spacecraft Using the SPHERES Testbed," *2005 AIAA Guidance, Navigation and Control Conference*, San Francisco, CA, August 2005.

[3] Henshaw, C. G., "Nonlinear and Adaptive Control Strategies for Improving the Performance of Teleoperated Spacecraft and Underwater ROVS," M.S. Thesis, Dept. of Aerospace Engineering, University of Maryland, College Park, MD, 1998.

[4] Hossaini, L. S., "The Design and Analysis of a Second Generation Free Flying Underwater Camera Platform," M.S. Thesis, Dept. of Aerospace Engineering, University of Maryland, College Park, MD, 2000.

[5] Henshaw, C. G., and Sanner, R., “Hybrid AI/control system interactions and analysis,” *Journal of Experimental & Theoretical Artificial Intelligence*, 16(4), October-December 2004.

[6] Wagenknecht, J., Fredrickson, S., Manning, T., and Jones, B., "Design, Development and Testing of the Miniature Autonomous Extravehicular Robotic Camera (Mini AERCam) Guidance, Navigation, and Control System," *26th Annual American Astronautical Society Guidance and Control Conference*, February 5-9, 2003.

[7] Dorais, G. A., and Gawdiak, Y. “The personal satellite assistant: an internal spacecraft autonomous mobile monitor,” *Proceedings of IEEE Aerospace Conference*, 2003, Vol. 1.

[8] Maluf, D. A., Gurram, M. M., Dorais, G., and Tran, P. B, "Active Sensor Vision Sensing for Spacecraft Mobile Robot Navigation," *Proceedings of the Systematics, Cybernetics, and Informatics (SCI) on Computer Science & Engineering Conference, 2003*.

[9] Hexmoor, H. and Vaughn, J., “Computational adjustable autonomy for NASA Personal Satellite Assistants,” *Proceedings of the 2002 ACM symposium on Applied computing*, 2002, pp. 21-26.

[10] Smithanik, J., Atkins, E., and Sanner, R. (2005). "A visual positioning system for an underwater space simulation environment," *AIAA Journal of Guidance, Control, & Dynamics*, in press.

[11] Sanner R. and Vance. E., “Adaptive control of free-flying space robots using neural networks,” *Proceedings of the American Control Conference*, 1995.

[12] Sanner, R. M. and Slotine, J. J, “Gaussian networks for direct adaptive control”, *IEEE Trans Neural Networks*, 1992.

[13] McGhan, C. L. R., “A Trajectory Planner for Autonomous Structural Assembly,” Undergraduate Research Project, Dept. of Aerospace Engineering, University of Maryland, College Park, MD, 2004.

[14] Wie, B., Weiss, H., and Arapostathis, A., “Quaternion Feedback Regulator for Spacecraft Eigenaxis Rotations,” *AIAA Journal of Guidance, Dynamics, and Control*, 12(3), May-June 1989.

[15] Nise, N. S., *Control Systems Engineering*, 3rd ed., Wiley & Sons, New York, 2000.