Experimental Demonstrations of Semi-Autonomous Control

Mark Campbell, Raffaello D'Andrea, Jin-woo Lee, Eelco Scholte

Department of Mechanical and Aerospace Engineering
Cornell University, Ithaca NY, 14853
{mc288,rd28,jl206,es244}@cornell.edu

***Abstract*— This paper describes two experimental testbeds which focus on real time verification and validation of semi-autonomous cooperative control. RoboFlag is an experimental and real time simulation of a capture the flag game using robots. The SeaScan is a long endurance UAV designed for military and commercial applications. For each testbed, details of hardware, simulators, current results, and experiment plans are given.**

I. Introduction

Research supporting applications where operators control multiple vehicles has been on the rise in recent years. A wide variety of applications are fueling the work, such as battlefield management, search and rescue, and space exploration. The challenges with these problems include: 1) developing "cooperation" amongst the vehicles, 2) developing higher level strategies and resource management, 3) interfacing the vehicles with the operator in a predictable manner, 4) Understanding and modeling the interface between the operator and vehicles, and 4) verifying the results using high fidelity simulations and experiments.

Ref. 1 proposes to use a hierarchical control structure (Figure 1). This allows fundamentally different approaches to be developed for cases such as M=1:N=3, M=2:N=10, or even M=5:N=150 (M is the number of operators, N is the number of vehicles). In each case, the level of automation must change as a function of N, M and tasks assigned. Lower levels of the hierarchy, such as estimation and path planning, are defined by formal algorithms with hard guarantees. Mid- and high-levels of the hierarchy, such as team strategizing and composition, are defined by optimization and/or randomized algorithms that allow teams to perform operator tasks successfully, but not predictably. Operator interfaces can occur at multiple points in the hierarchy.

The authors have developed a series of experimental testbeds that make use of multiple vehicles to validate a wide array of technologies I. The Cornell RoboCup team and its robots have won the international competition four of the past five years.[2] RoboFlag[3], [4] utilizes fourth generation of RoboCup robots to play games of capture the flag, complete with user interfaces and layers of technology. The SeaScan is a long endurance UAV designed and built by the Insitu Group;[5] the Cornell team developed the payload for mid-level autonomous control.[6] The ION-F system[7] consists

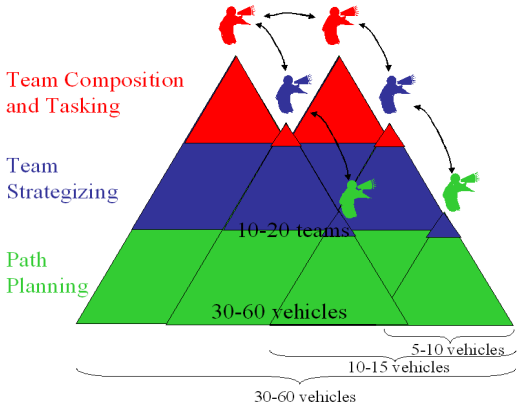

Fig. 1. Hierarchical, semi-autonomous control of a complex, multiple vehicle system.

of three fully functional spacecraft designed to validate formation flying in space. The ICE CUBE spacecraft[8] are two identical spacecraft used to measure scintillations in the ionosphere. The FF experiment[9] is used to test techniques for controlling spatially interconnected systems. The Cornell AFV is used to explore the in-flight formations of phased array antennas.[10]

This paper details the RoboFlag and SeaScan testbeds, and their role in verifying and validating semi-autonomous control technologies. First, the SeaScan vehicles are described. Hardware and simulators are detailed, along with the software architectures and initial experimental results. The RoboFlag testbed is then described, including the hardware, real time simulator, and experimental results. Finally, experimental plans for these testbeds are presented.

TABLE I
Cornell Multiple Vehicle Experiments.

	# Veh	InterVeh Coord	InterVeh Comm	OnBoard Control	OnBoard Auton
SeaScan 1.2	2	Y	N	Y	Y
RoboCup 6.0	6	Y	N	Y	Y
RoboFlag 2.0	11	Y	N	Y	Y
ION-F 1.0	3	Y	Y	Y	Y
ICE CUBE 3.0	2	N	N	Y	N
FF 3.0	4	Y	N	N	N
AFV 1.0	4	Y	Y	Y	Y

Fig. 2. Transparent view of the SeaScan UAV (top), and the SeaScan prepped for launch (bottom).

II. The SeaScan UAV

The SeaScan[5] is a long endurance unmanned aerial vehicle (UAV) platform developed by Insitu Group for a variety of applications including search-and-rescue, fishing reconnaissance, coastal patrol and other applications. Figure 2 shows the SeaScan platform. The dimension, weight and performance of the SeaScan are shown in Tables II, III, and IV respectively.

Unlike many small UAV's which rely solely upon batteries, the SeaScan system includes an electrical generator powered by the engine and is capable of generating up to 140 Watts. SeaScan launches at about 50 kt; on land, a cartop cradle or a low-pressure pneumatic catapult can be used, and at sea, only the latter is used. Figure 3(left) shows the catapult launch. SeaScan retrieval/landing is typically accomplished using the Skyhook, an Insitu-patented technique, where the aircraft flies into a single suspended line. A hook on the wingtip grabs the line and quickly stops the aircraft. Figure 3(right) shows the SeaScan caught by the Skyhook at a 60 kt approach.

The structure of the avionics is shown in Figure 4, while Figure 5 shows the internal layout of the electronics. The primary functions of the onboard avionics include:

Flight Control: The SE555 provides inner loop control and data management. The SE555 receives data from aircraft sensors and sends commands to control surfaces. Flight path characteristics can be determined from pre-programmed or in-flight commands.

Air-to-Ground Communication: This data link is used to communicate aircraft status, control, and mission data, as well as to relay messages from payload modules.

Avionics-to-Payload Communication: This data link is used to send sensor reports to the payload, and receive commands from the payload.

Video Data: The SE555 communicates serially with an on-board video turret, issuing camera positioning commands as well as aircraft attitude and stabilization data. The video signal bypasses the SE555 and is sent directly to a ground receiver via an onboard RF link.

GPS Receiver: The SE555 communicates serially with an onboard GPS receiver, receiving differential GPS data from the ground to improve position and velocity solutions.

Sensor Management: The onboard sensors include roll, pitch and yaw rate gyro, vertical, lateral and longitudinal accelerometer and temperature, external temperature, relative pressures of pilot, alpha, beta, gamma, and absolute pressures of barometric and manifold.

TABLE II
SeaScan Dimensions.

Wing span	3.04 m
Wing area	0.72 m^2
Wing sweep	23 deg
Fuselage diameter	0.18 m
Overall length	1.2 m

TABLE III
SeaScan Mass Properties.

Airframe	3.7 kg
Avionics/Payload	3.2 kg
Power plant	2.4 kg
Empty weight	11.0 kg
Max fuel weight	4.3 kg
Max launch weight	15.4 kg
I_{xx}	3.7560 kg m^2
I_{yy}	1.3447 kg m^2
I_{zz}	4.6774 kg m^2
I_{xz}	0.0225 kg m^2

TABLE IV
SeaScan Performance.

Ambient temperature range	-10° C to +40° C
Max level speed	68 knots
Cruise speed @ max wt	48 knots
Min speed max wt	42 knots
Max S/L climb max wt	2.5 m/s
Service ceiling @ max wt	5000 m
Still-air range, no reserves	1500 km
Navigation	GPS (inc differential)
Communication	9600 baud half-duplex UHF U/L,D/L; 1.7 GHz

Fig. 3. SeaScan UAV ready for catapult launch (left). SeaScan "landing" using the skyhook system.

The Cornell developed payload module is designed to implement mid- and high-level control commands. State/parameter estimation, fault detection, and path planning are realized within this module. The payload is an Embedded Planet 8260 board with a Motorola PPC8260 processor. The processor performs at 570MIPS, 300Mhz, and includes 64MB RAM, 32MB Flash, two serial ports, and is VxWorks 5.4 compatible. The power specifications are 1.5Amps, 5V max.

A. Simulators

Two simulators exist for developing and testing algorithms before flight. The first is a software only simulator in C++. Validation at this level tests interfaces to the flight code, as well as the relative speed of the implementation. It does not, however, verify the real time implementation of the coding. The hardware in the loop simulation includes not only the full avionics suite, but also a set of servo mechanisms that mimic the on-board actuators, RF communication for air-to-ground, and a high fidelity aircraft dynamics simulator.

B. Sample Experimental Results

The focus of the research implemented on the SeaScan vehicles includes on-line model development using nonlinear estimation techniques, estimating and tracking system faults, reconfiguration of the control after a fault, path planning, and coordination of multiple SeaScans. In Nov 2003 and Jan 2004, the Insitu/Cornell team demonstrated the first systems level validation of the SeaScan/Cornell payload system. The launch of the SeaScan is shown in Figure 6.

The Cornell payload was fed raw sensor data at 10Hz from the primary CPU, and a nonlinear sigma point filter was used to estimate the aerodynamic model on-line.[11] A path planner on the payload board was also implemented. The planner selected "orbits" (R, X, Y, Z) and bow tie patterns based on on-board, and delivered commands to the SeaScan CPU. Figure 7 shows the 3D position estimates of the filter over time. The planner based orbits, orbits at different altitudes, and bow tie patterns can each be seen.

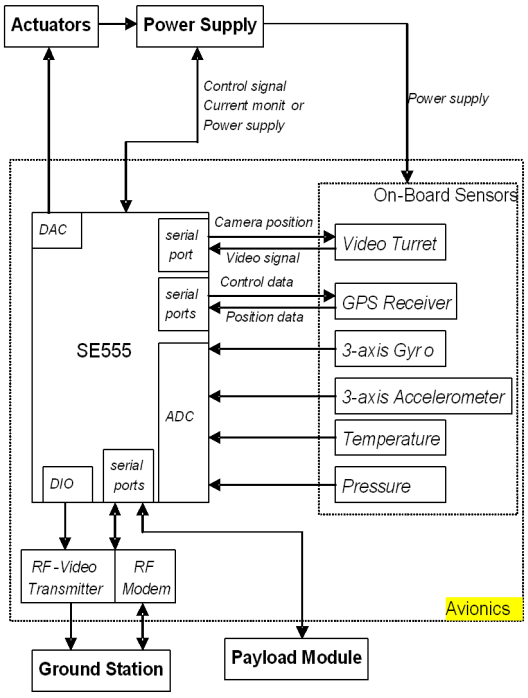

Fig. 4. SeaScan Avionics structure.

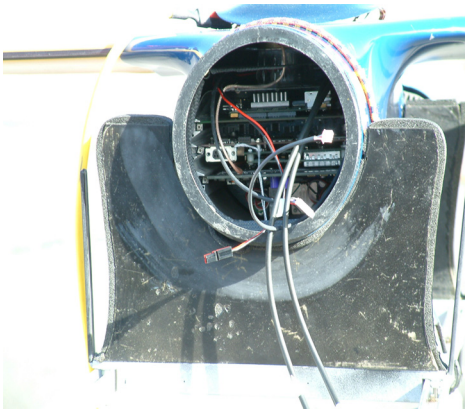

Fig. 5. Nose off view of the SeaScan: Parallel boards plug into a "shaped" backplane in the form of the fuselage (top to bottom): Cornell payload, CPU+GPS+modem, Power.

C. Experiment Plans

In the near term, two sets of flight experiments are planned, demonstrating:

Model Estimation: Single vehicle aerodynamics will be estimated during normal operations and during an aileron fault. Nonlinear estimators (SPF's,[11] ESMF's,[12] and hybrid estimator[13]) will be implemented. Integration with on-line model predictive control[14] will be tested.

Path Planning for Target Pursuit: A streamline based path planner, as well as an evolutionary path planner will be implemented. Basic path planning around obstacles, with risk, and in target pursuit

Fig. 6. SeaScan UAV just after launch.

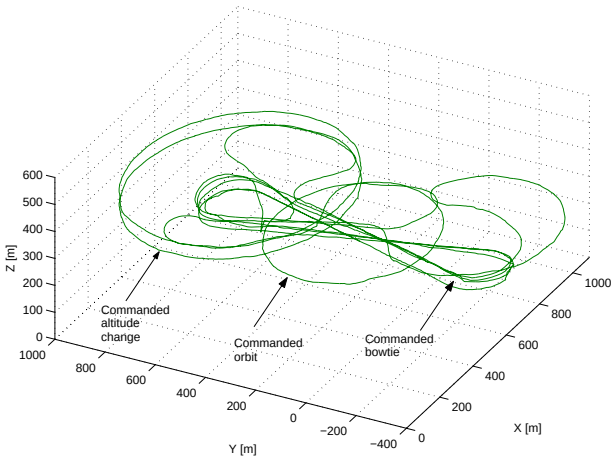

Fig. 7. SeaScan UAV position tracking (Jan 2004).

will be explored.

Cooperative Control: Cooperative planning and control of two SeaScan vehicles for coupled objectives. The primary mission will be a cooperative reconnaissance mission.[15]

Far term work will focus more exclusively on multiple vehicle work, with an emphasis on reconnassaince missions and possibly operator controls.

The final implementation of both the matrix based estimation and control will take advantage of state of the art tools in matric operations. Current computer architectures rely heavily on the use of cache and memory structure for speed. To take advantage of these features a logical choice for matrix operations on an embedded system is the use of optimized libraries. The most common library for matrix and vector operations is BLAS.[16] A C++ implementation of level 1, 2 and 3 BLAS is provided by the uBLAS[17] package, which enables the user to call all BLAS functions with the use of expression templates in C++.

TABLE V

TIME REQUIRED FOR MATRIX LIBRARIES (PIII, 900 MHZ, WIN2K).

Tool	$P^2_{250\times250}$	$P^2_{1000\times1000}$
Array (standard C++)	0.15	n/a
uBLAS, Level 3 BLAS	0.05	28.61
uBLAS, Level 3 ATLAS (tuned)	0.20	2.91

Although a very time intensive process, BLAS is often tuned specifically for a given platform in order to create efficient code. However, another option is available, called ATLAS.[18] ATLAS is an automatically tuned BLAS version, which provides a high-performance library for most all platforms. In addition it includes several LAPACK[19] functions for Linear Algebra. Table V compares these tools on several sample problems. For smaller matrices, the uBLAS works the best. As the matrix size becomes larger and the density of the matrices becomes an important factor, the increased speed using ATLAS is prominent.

III. ROBOFLAG

RoboFlag, is an experimental testbed[4] with autonomous, fast-moving teams of vehicles, and is therefore an excellent system to aid in the development and evaluation of realistic solutions for semi-autonomous control with $N >> M$. The objective of the RoboFlag competition[3] is to venture into opponent territory, locate and capture the "flag," and return to the "home base." This has many key aspects of these future systems, including a human operator, team dynamics, different levels of tasking, cooperative planning, and uncertainties such as incomplete information, latency, and an intelligent adversary.

RoboFlag exists in both hardware (real robots), and software (a real time simulator). The approach is to use the simulator for all development work, and then move to the hardware for demonstrations and "focused" tests. The RoboFlag hardware is based on the 2001 RoboCup robots. The RoboFlag field is significantly larger than the regulation RoboCup field, which necessitated the use of a two camera vision system. Each camera images half the field and feeds its view into a vision computer. The vision computer then uses colored dots located on top of the robots to identify each robot, the its location, which team it is on, and its orientation. The algorithmic software (path/strategy/resource planning) resides on a set of parallel computers for the entities. The arbiter computer takes in all information (from both the vision and entity computers), evaluates game constraints, and sends commands to each of the vehicles. Within the arbiter, game conditions can be updated and changed, such as vehicle and sensor types, communication cones, weather patterns, etc. The arbiter relays commands to the actual robots via a wireless RF link. The hardware setup is depicted in Figure 8, while the software is depicted in Figure 9.

Figure 10 shows a GUI with the field, as seen by the Blue operator. The field areas (Home Zone, the Defense Zone, and the Attack Zone) are easily seen, as are 1) robot status

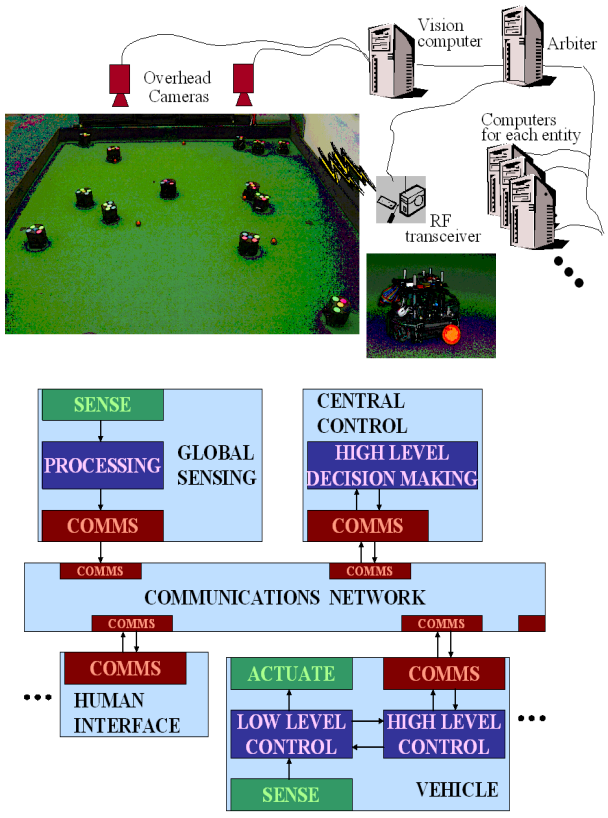

Fig. 9. Software architecture for RoboFlag.

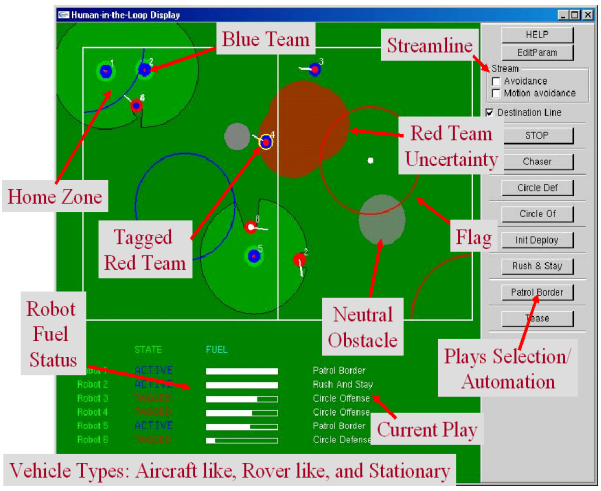

Fig. 10. RoboFlag interface, showing a current game, from the perspective of the Blue operator.

including fuel and play, 2) neutral obstacle locations, 3) path planner selection, and 4) higher level autonomy (play) selection. The GUI is updated as new plays are developed.

A. Simulator

In addition to the hardware, RoboFlag exists as a software only experiment. The software simulator is an excellent substitution for the hardware because it mimics the interfaces and speed of the true system, from the communications to the distributed implementation of the algorithms to the arbiter. All algorithms and software are developed on the simulator.

B. Sample Experimental Results

Research on the testbed has focused on algorithms at each level of the hierarchy (Figure 1). Path planning, such as streamline based planners,[20], [21] have worked well in both theory and experiment. Optimization based cooperative strategies have been explored for reconnaissance,[15] defense,[22] and time critical tasks.[1]

Perhaps the most important work has been in the area of human in the loop testing of operators controlling multiple vehicles. The Cornell team led a set of studies using the hardware and software simulator,[23] and varied dependent variables such as the number and speed of the vehicles. Sample results are shown in Figures 11-12. This work showed particular trends, such as 1) multiple operators on one terminal cooperate to improve situation awareness and improve command selection; multiple operators scored higher in the game total than single operators; and 2) automations, such as autonomously sending vehicles home to refuel, were used much more as the vehicle speeds and numbers increased. The AFRL/HE group, led by Dr. Scott Galster, has also implemented a series of HitL studies using RoboFlag[24], [25] using the AFRL subject pool. These HitL studies examined human performance differences and subjective ratings of mental workload and situation awareness.

C. Experiment Plans

The following areas are currently or will be explored using the RoboFlag system:

Cooperative Reconnaissance: Further work in decentralizing the estimation process, as well as the use of clustering targets will be implemented.

Strategy Generation: An approach to developing higher level plays and strategies is being explored that utilizes genetic evolution of base vehicle plays (defend, attack, recon, etc.).

Human in the Loop: Operator based studies will continue, with a focus on developing data for decision modeling, and exploring the interface between the operators and the multiple vehicles.

IV. Lessons Learned

The success of the Cornell multiple vehicle testbeds can be traced to three important educational areas at Cornell: 1) the implementation of the systems engineering program, 2) the culture of project teams, and 3) the health of the Masters of Engineering program. The on-campus students typically come from one of many majors (M&AE, ECE, CS, ORIE, CE), and must take a project in addition to systems engineering courses.

In addition, several of the important lessons learned on these testbeds and others include:

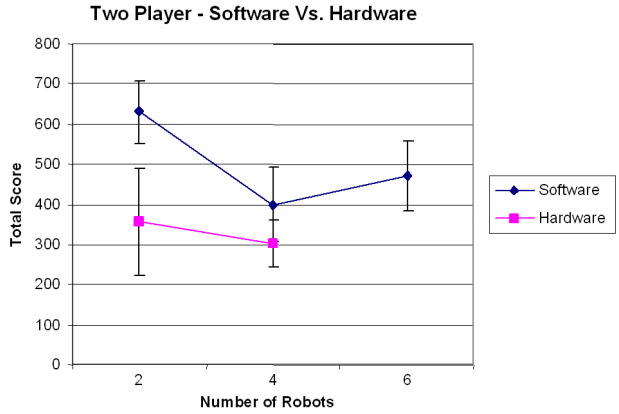

Fig. 11. Human in the loop scores, comparing hardware vs. software based tests as a function of the number of robots in the game.

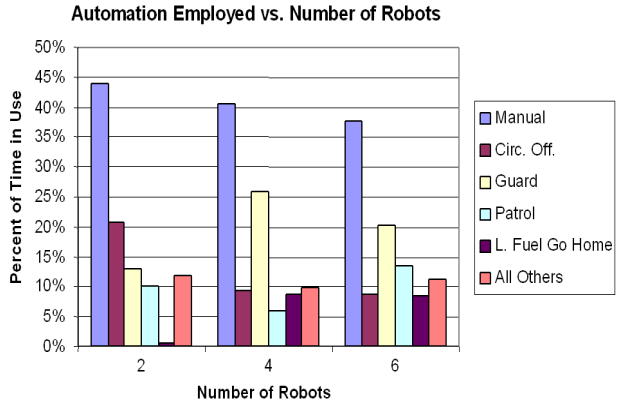

Fig. 12. Percentage of play automations used in the human in the loop tests, as a function of the number of robots in the game.

- The use of systems engineering processes (requirements definition, interface control, etc.) is a critical part of the experimental development.
- Development and construction of a series of prototypes is valuable to the learning process and eventual success of the final experiment.
- Teams of MAE, ECE, and CS students work very well together, especially once the language barrier has been overcome.
- Having one or several mentors with "corporate knowledge" is critical, especially when there is a high turnover of students.
- Hardware in the loop testing is critical to the final success of the experiment.

V. Acknowledgments

The SeaScan work was supported by the DARPA SEC program (#F33615-99-C-3612), with Dr. Helen Gill and Dr. John Bay as DARPA Program Monitors and Mr. Raymond Bortner and Mr. James McDowell from AFRL as Contract Monitors. The RoboFlag work was supported by the DARPA MICA program (#F30602-01-2-0577), with Maj. Sharon Heise, PhD as DARPA Program Monitor and Mr. Carl DeFranco AFRL/Rome Labs as Contract Monitor.

References

[1] M. Campbell, R. D'Andrea, D. Schneider, A. Chaudhry, S. Waydo, J. Sullivan, J. Veverka, and A. Klochko, "RoboFlag Games using Systems Based, Hierarchical Control," in *Proceedings of the American Control Conference*, 2003.

[2] P. Stone, M. Asada, T. Balch, R. D'Andrea, M. Fujita, B. Hengst, G. Kraetzschmar, P. Lima, N. Lau, H. Lund, D. Polani, P. Scerri, S. Tadokoro, T. Weigel, and G. Wyeth, "RoboCup-2000: The fourth robotic soccer world championships," *AI Mag*, vol. 22(1), p. 11-38, 2001.

[3] R. D'Andrea and R. Murray, "The RoboFlag Competitions," in *Proceedings of the American Control Conference*, 2003.

[4] R. D'Andrea and M. Babish, "The RoboFlag Testbed." in *Proceedings of the American Control Conference*, 2003.

[5] "The SeaScan UAV," *The Insitu Group*, www.insitugroup.net.

[6] M. E. Campbell, J. Han, J. Lee, E. Scholte, and J. Ousingsawat, "Validation of active state model based control using the seascan uav," in *AIAA Unmanned Unlimited Conference*, Sept 2003.

[7] M. Campbell, C. Swenson, R. Fullmer, and C. Hall, "The ionospheric observation nanosatellite formation - ion-f," in *Small Satellite Systems and Service Conference*, La Baule, France, June 2000.

[8] S. Waydo, D. Henry, and M. Campbell, "Cubesat design for leo-based earth science missions," in *IEEE Aerospace Conf*, March 2002.

[9] J. M. Fowler and R. D'Andrea, "A formation flight experiment for research in control of interconnected systems," *Control Systems Magazine*, 2003, to appear.

[10] S. Breheny and R. D'Andrea, "Using airborne vehicle-based antenna arrays to improve communications with uav clusters," in *Conference on Decision and Control*, 2003.

[11] S. Brunke and M. Campbell, "Square root sigma point filtering for aerodynamic model estimation," *AIAA Journal of Guidance, Control, and Dynamics*, vol. Mar-Apr, 2004.

[12] E. Scholte and M. Campbell, "A nonlinear set-membership filter for on-line applications," *International Journal of Nonlinear and Robust Control*, vol. 13, no. 15, pp. 1337–1358, Dec 2003.

[13] P. Otanez and M. Campbell, "Bounded model switching in uncertain hybrid systems," in *American Control Conference*, 2004.

[14] E. Scholte and M. Campbell, "Robust nonlinear model predictive control with partial state information," in *AIAA Guidance, Navigation and Control Conference*, 2003.

[15] J. Ousingsawat and M. Campbell, "On-line estimation and path planning for multiple vehicles in an uncertain environment," *International Journal of Nonlinear and Robust Control*, (to appear, 2004).

[16] C. L. Lawson, R. J. Hanson, D. Kincaid, , and F. T. Krogh, "Basic linear algebra subprograms for fortran usage," *ACM Trans. Math. Soft.*, vol. 5, pp. 308–323, 1979.

[17] http://www.boost.org/libs/numeric/ublas/doc/index.htm, "ublas."

[18] R. C. Whaley, A. Petitet, and J. J. Dongarra, "Automated empirical optimization of software and the ATLAS project," *Parallel Computing*, vol. 27, no. 1–2, pp. 3–35, 2001 (`www.netlib.org/lapack/lawns/lawn147.ps`).

[19] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen, *LAPACK Users' Guide*, 3rd ed. Philadelphia, PA: Society for Industrial and Applied Mathematics, 1999.

[20] S. Waydo and R. M. Murray, "Vehicle Motion Planning Using Stream Functions," in *IEEE Int'l Conf on Robotics and Automation*, 2003.

[21] J. Sullivan, S. Waydo, and M. Campbell, "Using stream functions to generate complex behavior," in *AIAA Guidance, Navigation and Control Conference*, 2003.

[22] M. Earl and R. D'Andrea, "A study in cooperative control: The roboflag drill," in *American Control Conference*, 2002.

[23] J. Veverka and M. Campbell, "Experimental study of information load on operators in semi-autonomous systems," in *AIAA Guidance, Navigation and Control Conference*, 2003.

[24] R. Parasuraman, S. Galster, , and C. Miller, "Human control of multiple robots in the roboflag simulation environment," in *IEEE Int'l Conference on Systems, Man, and Cybernetics*, Oct. 2003.

[25] P. N. Squire, S. M. Galster, and R. Parasuraman, "The effects of levels of automation in the human control of multiple robots in the roboflag simulation environment," in *Human Performance, Situation Awareness, and Automation Conference*, 2004.