

One challenge facing coordination and deployment of unmanned aerial vehicles (UAVs) today is the amount of human involvement needed to carry out a successful mission. Currently, control and coordination of

UAVs typically involves multiple operators to control a single agent. The aim of this article is to invert this relationship, enabling a single pilot to control and coordinate a group of UAVs. Furthermore, decision support is provided to the pilot to facilitate effective control of the UAV team. In the scenario envisioned in this article, the human operator (the pilot) is operating along-side a team of UAVs. The pilot communicates with the UAV team remotely and controls the UAV team to execute a surveillance mission.

An important aspect of this is the question of how much the pilot should interact with the UAV team and how much aid should be provided to the pilot without overloading the pilot with data and support. We address this issue by allowing two major modes of operations, namely autonomous mode and pilot-controlled mode. In both of these modes, the UAV team is controlled in a leader-follower manner, and the leader UAV is assigned by the pilot, where the followers are positioning themselves with respect to the other UAVs in the network. In the autonomous mode, the leader UAV executes the mission without intervention of the pilot. At any time, the pilot is allowed to take over and directly control the leader vehicle. Hence, the pilot can interrupt the mission to investigate an area or avoid certain threats. The pilot can also release control of the UAV, and the UAV team automatically resumes the execution of the given mission.

The problem of controlling multiple agents in a coordinated fashion to achieve a set of goals, such as maintaining desired formations, ensuring coverage



© U.S. AIR FORCE PHOTO/MASTER SGT. ROBERT W. VALENCA

Digital Object Identifier 10.1109/MRA.2009.932526

# Executive Decision Support

## *Single-Agent Control of Multiple UAVs*

BY XU CHU DING, MATTHEW POWERS, MAGNUS EGERSTEDT,  
SHIH-YIH (RYAN) YOUNG, AND TUCKER BALCH

**The goal of the real-time optimal control module is to update the switching time vector at each time-step, in such a way that it gets incrementally closer to the optimal solution.**

of an area, or selection of the best leader for a group of agents, has received considerable attention during the last decade. In particular, multiple approaches have been developed to allow the inclusion of a human operator. In [1] and [2], the human operator provides commands to all the agents in the network either by total control of the robot or supervisory control, which modifies the behaviors of the robots. In either case, the same commands are provided to all the agents in the network simultaneously. Another work [3] developed a central interface to display the information of a swarm of mobile robots. In [4], the effect of information and decision aids provided to a human operator is examined in an abstract setting to address the question of how much interaction with the human should be provided.

The main novelty of the work in this article lies in the decision support aid provided to the pilot in the form of leader selection mechanisms. During operation of the system, the pilot can reconfigure the UAV network topology by assigning a different leader vehicle. When the system is in autonomous mode of operation, a desired leader selection is provided to the pilot. This selection is computed by a receding horizon, real-

time optimal control algorithm that evaluates relative merits of different leader selections toward execution of the mission. It should be noted that we always insist on including the human operator in the loop for making high-level control decisions. Although this article focuses on a single-leader, multiple-followers scenario, the system introduced here can be generalized to a system with multiple leaders and groups.

### System Architecture

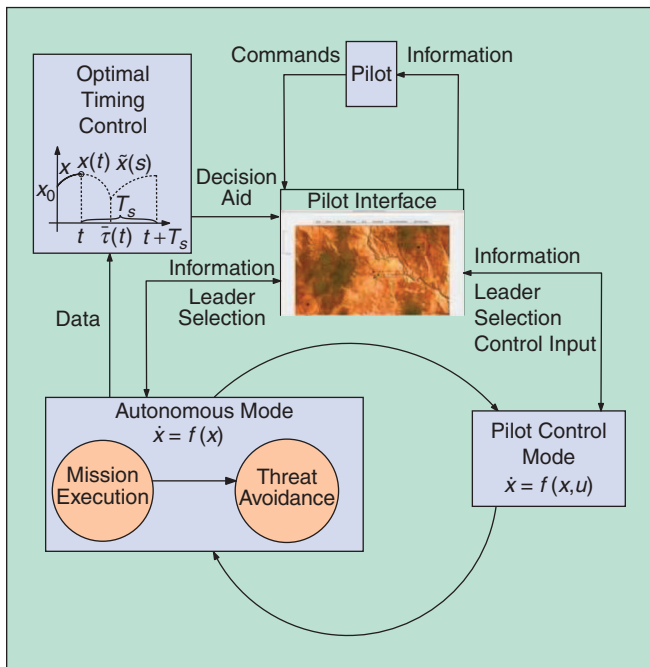
The support system developed here is designed with the philosophy in mind that the aid and support are provided but not forced. The mission implemented to demonstrate the pilot decision-support system is a surveillance mission, and an a priori mission plan is generated at configuration time using pilot input. Once the mission is planned, it is executed autonomously. The pilot can take control at any time, during which the mission is paused. This is expected since the pilot may choose to investigate an area of interest or avoid certain threats. The overview of the pilot decision-support system is shown in Figure 1. The arrows represent information and interactions between different components of the framework.

Information and support is designed to be delivered to the pilot in a way so that data can be understood and acted upon quickly. Hence, the central component of the system is a graphical user interface (GUI). An example of the GUI during operation of a mission is shown in Figure 2. All data and information are congregated and exchanged at the GUI, including local information about the individual UAV agents and decision support computed by the optimal control module. This interface also provides the mean of interaction between the pilot and UAV agents. The pilot selects and controls the leader vehicle by interacting with the GUI. In addition, the GUI displays data (location, altitude, etc.) of the network graphically so that the information can be easily absorbed by the pilot. The background map shown on the GUI corresponds to the environment of the UAVs, which is a virtual three-dimensional (3-D) world constructed in a Player/Gazebo simulation environment [5], as shown in Figure 3. This simulation environment is described more in detail in the next section.

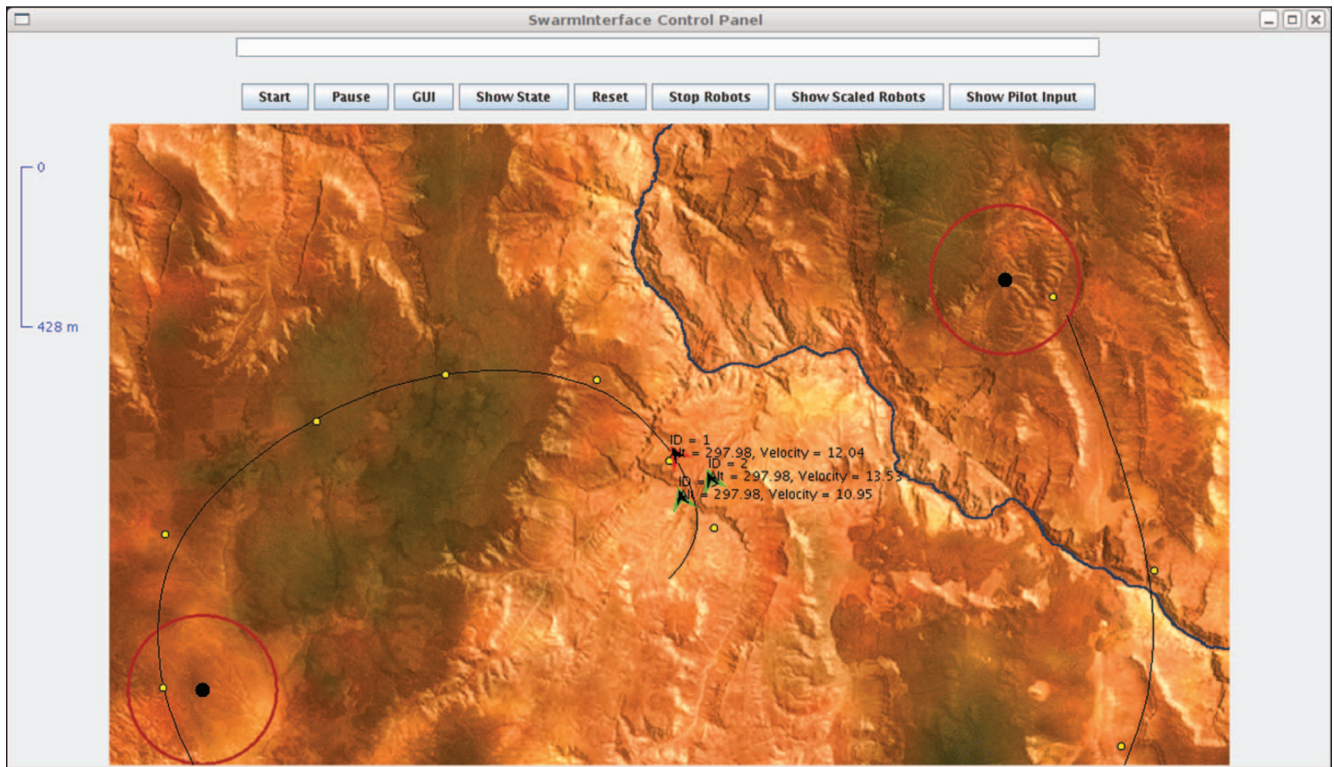
The system supports autonomous mode and pilot-control mode of operation. The pilot can switch the system between these two modes at any time. For the majority of time in autonomous mode, the autonomous controller is designed to execute the surveillance mission. This mode of autonomous operation is referred to as the *mission-execution mode*. There may be situations when an imminent threat is detected by local sensors on the UAVs, in which case the system switches to the threat-avoidance mode, in which threat avoidance is carried out in higher priority. Furthermore, the optimal timing control module is designed to provide decision aid based on state information when the system is in mission-execution mode. Every time the system switches to the mission-execution mode, the decision aid is recomputed in a real-time fashion.

### Autonomous Mode

One important goal of the proposed system is to provide autonomous operation of the UAV team for a given mission. In



**Figure 1.** Pilot decision-support system architecture and different modes of operation.

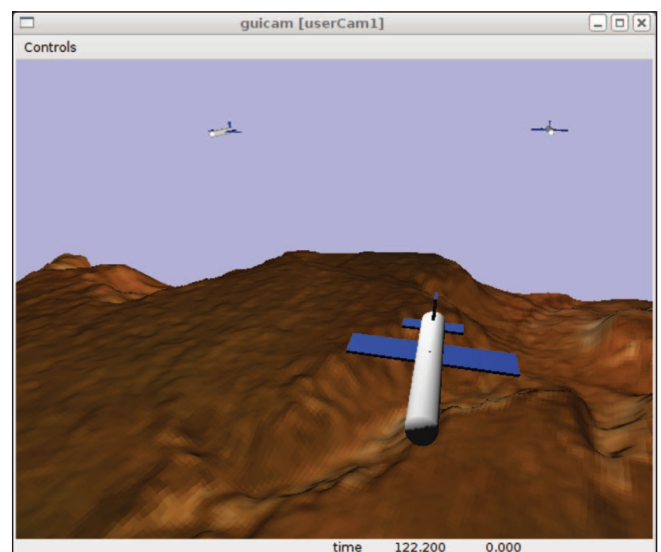


**Figure 2.** GUI of the pilot decision-support framework.

autonomous operation, it is vital to choose a proper formation and topology for the multiagent network in response to changes in the environment or human intervention. For example, [6] suggests using a spread-out formation in an open space and a closely grouped formation in a tight area. In the proposed framework, the UAV team is designed to be controlled in a leader–follower fashion. This has been an increasingly important approach to control multiagent network systems, and it is proven to be successful for many tasks (see [7] and [8], for example). Using the leader–follower control approach, the topologies of the network depends on the identity of the leader agent. After the pilot releases control, it may be necessary to reevaluate the network topology so that a better candidate vehicle may be chosen as the leader. To this end, it is natural to characterize the network as a hybrid system that switches among a number of topologies (see [9] for a similar idea).

For the clarity of presentation, we consider a UAV team consisting of three individual UAVs. One of the UAVs is designated as the leader vehicle, and the remaining UAVs act as followers. Each follower maintains a proper distance with respect to the leader and the other follower. Thus, in this UAV network, it is possible for the pilot to switch between three distinct subsystems, each defined by designating a different agent as the leader. At the planning level, the state of the system is defined as  $x = [x_1^T, x_2^T, x_3^T]^T$ , where  $x_i \in \mathbb{R}^3$  is the position of the  $i$ th robot in the network. Since the mission will be specified as evolving on the plane, it is useful to defined the operator  $Px_i = [x_{i1}, x_{i2}]^T$ .

The mission of the UAV network is defined as a surveillance task, and the goal of the mission is to provide coverage for an area. One way to achieve this task is to create a plan at configuration time using either a path planner or input from the pilot. The controller of the UAV network is designed to navigate through this path. This path is denoted as  $p(t) \in \mathbb{R}^2$ . We let this path be generated by first laying down a set of way-points on the map of the area, then computing a smoothing spline curve based on the way-points. The



**Figure 3.** UAVs in Player/Gazebo 3-D simulation environment.



## **The behavior arbitrator generates the appropriate behavior so that each agent moves away from the threat while performing its individual task.**

smoothing spline guarantees the continuity of  $p(t)$ , while balancing the curvature of the path and the distance from the path to the given way-points.

Let the identity of the leader be  $l$ , where  $l \in \{1, 2, 3\}$ . At a high level of abstraction, we let the dynamics of the networked system in autonomous and mission-execution mode be given by

$$\begin{aligned}\dot{x}_l(t) &= \begin{bmatrix} p(t) - Px_l(t) \\ 0 \end{bmatrix}, \\ \dot{x}_i(t) &= \alpha \sum_{j \neq i} (\|x_i(t) - x_j(t)\| - k)(x_j(t) - x_i(t)), \\ i &= \{1, 2, 3\} \setminus \{l\}.\end{aligned}$$

The leader controller is thus a simple proportional compensator that drives the leader agent to the desired path. This way, at least one UAV in the network is regularly covering the area over the given path. The followers execute a weighted consensus dynamics. This controller is designed to maintain a fixed distance  $k$  from the follower to the other two UAVs, driving the UAV network to a triangular formation. The vertical component is moreover kept constant to be used only when directly controlled by the pilot or when avoiding threats. An example of the UAV network information with the leader tracking a path is shown in Figure 2. In the autonomous mode, the leader vehicle is colored red, and the followers are colored green.

In addition to mission execution, the proposed framework also provides threat-detection capabilities. When an external threat is perceived by sensors, the UAVs must work to avoid the threat while navigating nominally along the defined path. This mode of operation within autonomous mode is referred to as the threat-avoidance mode. Within this mode, the dynamics of the agents are determined by the behavior arbitrator of each agent. The behavior arbitrator generates the appropriate behavior so that each agent moves away from the threat while performing its individual role in a weighted fashion. Once the threat is perceived to be out of range, then the system returns to the mission-execution mode. This switching behavior is illustrated in the autonomous-mode module in Figure 1.

### **Optimal Timing Control**

The objective of the optimal control module is to compute optimal switching times that minimize a cost function,

which is formulated as a measure of the progress of the surveillance task. As a result, this decision aid is only provided when the pilot decision-support system is in autonomous and mission-execution mode. The switching times information is then fed-back to the pilot, and it is up to the discretion of the pilot whether or not to switch according to the decision aid. For the pilot decision-support framework, the network is reconfigured constantly during system run-time, and the switching times need to be constantly recomputed online and updated. Furthermore, since the final mission time can be long and optimal control requires simulation of the state trajectory, a receding horizon approach is used. In this approach, a look-ahead window is selected, and the merits of switching to different leaders are evaluated over this time horizon.

We first attack this problem in a rather general setting by considering an autonomous switched dynamical system in the form of

$$\dot{x} = f_i(x), t \in [\tau_{i-1}, \tau_i], \quad i = 1, \dots, N+1, \quad (1)$$

with a given time horizon length  $T_s$  and initial condition  $x_0$ . The index  $i$  denotes the mode of the hybrid system and  $\tau_i$  denotes the switching time instant between mode  $i$  and mode  $(i+1)$ , assuming that  $\tau_0$  is the initial time and  $\tau_{N+1} = \tau_0 + T_s$  is the final time. The vector  $\bar{\tau} = [\tau_1, \tau_2, \dots, \tau_N]^T$  is called the switching time vector, and the optimization task is to minimize a cost function as a function of  $\bar{\tau}$ . As this is normally a hard problem, what we aim for is a real-time algorithm that improves the switching time vector at each time step.

Since we define the surveillance task in terms of tracking a given path, the instantaneous measure of task progress can be formulated as the average distance of the UAV network to the path. This average distance is the distance between the centroid of the network to the path, and we use  $\text{centroid}(x(t))$  to denote the centroid of the network. Hence, the real time optimal control problem is that of minimizing the centroid of the UAV team to the given path over the sliding window. This is a cost-to-go functional since it depends on the true time  $t$ , and is evaluated over a state trajectory over the time interval  $[t, t + T_s]$ . This state trajectory has to be simulated since it happens in the future. This future trajectory is denoted by  $\tilde{x}(s)$ , where  $s \in [t, t + T_s]$  denotes the simulated time. The dynamics of the simulated trajectory is identical to the system equation given in (1), with the initial condition  $\tilde{x}(t) = x(t)$ .

As such, the cost-to-go functional is given by

$$J(t, x(t), \bar{\tau}(t)) = \int_t^{t+T_s} L(\|\text{centroid}(\tilde{x}(s)) - p(s)\|_2^2) ds. \quad (2)$$

This performance measure ensures that we select the UAV that minimizes the average distance between the UAV network to the given path.

The goal of the real-time optimal control module is to update the switching time vector at each time-step, in such a way that it gets incrementally closer to the optimal solution. The iterative process used for this is a Newton-Raphson-based

real-time algorithm. This algorithm was first derived and proposed in [10], and a convergence analysis for a similar Newton-Raphson-based real-time algorithm was presented in [11], which showed that the convergence rate is quadratic. Quadratic convergence rate of this class of algorithms ensures the optimal switching times can be quickly obtained.

### Pilot-Control Mode

As previously mentioned, at any time during the mission, the pilot may take control of the leader vehicle. In this mode of operation, the dynamics  $\dot{x} = f(x)$  of the subsystem corresponding to  $l$ -th UAV being the leader of the UAV network becomes

$$\dot{x}_l(t) = u(t),$$

where  $u(t)$  is the control input (direction and speed) supplied by the pilot. The followers in this mode use the exact same dynamics as the mission-execution mode (maintaining a distance  $k$  from each other), but the leader is controlled directly by the pilot.

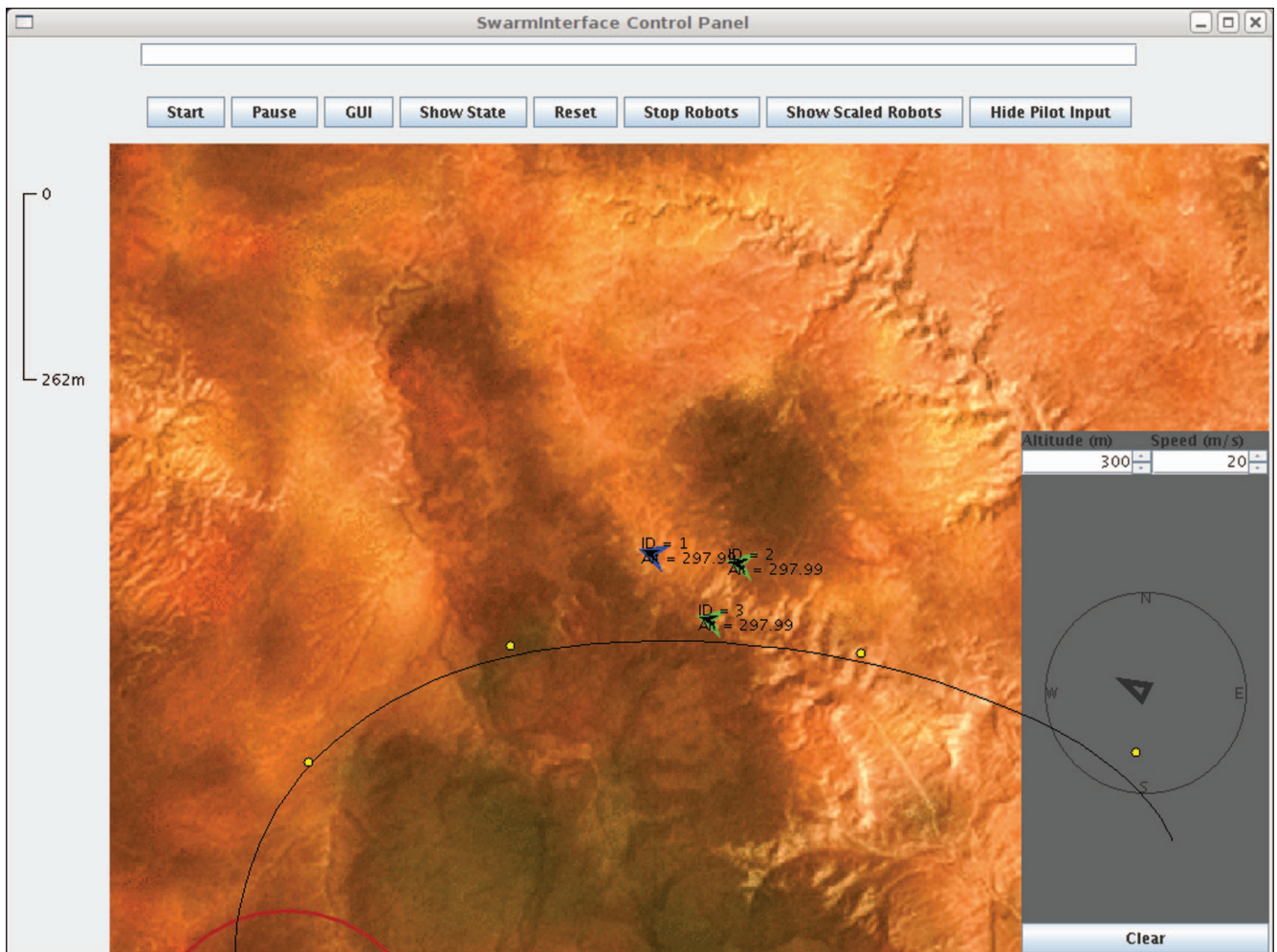
The pilot enters the mode by clicking on the pilot virtual joystick as shown in the lower-right corner in Figure 4.

***The mission of the UAV network is defined as a surveillance task, and the goal of the mission is to provide coverage for an area.***

The mouse click is used to generate the direction, and the speed of the vehicle can be adjusted by the virtual dial on the joystick panel. When the pilot is in control of a UAV, the vehicle is colored blue. If the pilot wishes to release control to autonomous mode, the pilot needs to click on the *clear* button.

### System Logic

Since all the components of the system architecture are described, we are ready to explain the logic flow of the pilot decision-support system for a surveillance mission. At every cycle (time step), the system goes through a decision tree as



**Figure 4.** GUI when in pilot-control mode of operation.

## At any time, the pilot is allowed to take over and directly control the leader vehicle.

shown in Figure 5 and switches to a different mode if necessary. The human pilot always has total control in decision making, including leader selection and movement of the UAV team. Furthermore, in the autonomous mode, threat avoidance has higher priority than mission execution. Therefore, the system switches to mission execution only when there is no threat perceived. It is moreover only during mission execution that the decision support will be computed and delivered to the pilot. This ensures that computational resources are not wasted. When the decision support is provided, it is up to the discretion of the pilot on whether or not to abide by the computed suggestion.

### Implementation

The pilot decision-support system is implemented in a modular and extendible way. A third-party open-source (Player/Gazebo) simulation environment is used to generate a 3-D environment where physical interactions can be simulated realistically. Furthermore, this simulation environment allows for a seamless transition from simulation to robotic hardware so that the system may be implemented on real UAVs in the future.

### Agent Implementation

The UAV agents are implemented using a modular, behavior-based design. Primitive and integrated perceptual

information from a variety of sources are directed through a set of behaviors that calculate output preferences for control set points in three dimensions: translational curvature within the  $x$ - $y$  plane ( $k$ ), translational velocity within the  $x$ - $y$  plane ( $v$ ), and altitude ( $z$ ). The output from the individual behaviors is combined in a behavioral arbitration layer, resulting in final  $k$ ,  $v$ , and  $z$  set points for the vehicle's low-level controllers.

### Perceptual Sources

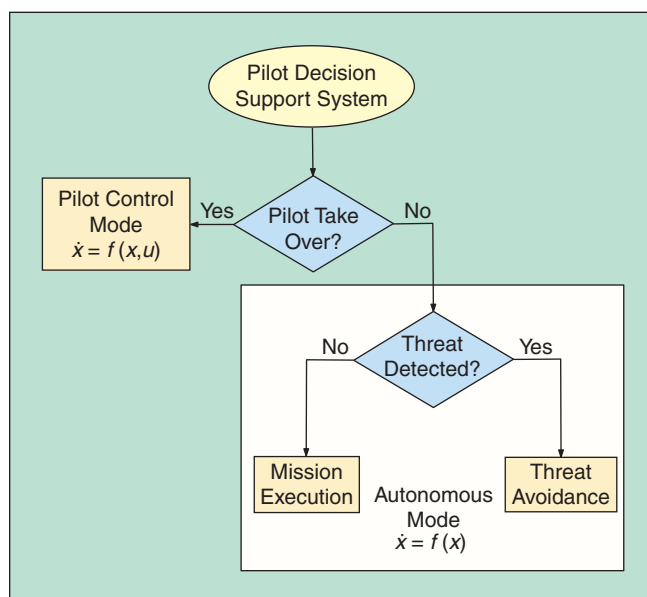
Each robot in the simulation is implemented as an independent agent, acting either as a leader robot or a follower robot, according to input from the pilot. The robots act on information from several sources. The following are the sources included.

- 1) *A priori plan*: as described in the "Autonomous Mode" section, a plan is generated at configuration time for the UAVs to follow, which leads them through the simulated environment. The plan is represented by a time-parameterized smoothing spline curve based on a set of way-points provided by the pilot.
- 2) *External perception*: via an interface with the simulation, the robots are able to observe their own relative and global positions. Additionally, they are aware of features within the environment, including external threats, which must be avoided.
- 3) *Inter-robot communication*: the robots are able to communicate their internal states, including whether they are currently acting as a leader or follower robot, with each other. The robots are also able to communicate their global positions to aid in the formation maintenance.
- 4) *Pilot input*: the human pilot may communicate to the robots, choosing one to act as leader within the group, and, optionally, taking remote control of the leader.

### Behavioral Control

The UAVs navigate using a behavior-based control system based on the distributed architecture for mobile navigation (DAMN) (see [12] and [13], for an extended version). In this architecture, each behavior represents a particular interest of the agent (e.g., avoid threats, follow the plan, and follow the pilot's input). The behaviors' output is expressed as preferences or votes, across a set of discrete options for the next control set point. Arbitration between these behaviors is performed by finding the option with the highest preference from a weighted sum of all behaviors.

In addition to expressing preferences across a set of possible  $k$  values, each behavior may set a maximum allowable  $v$  and a minimum allowable  $z$ , given each  $k$  value. Once a  $k$  value is chosen by the arbiter, the minimum of the maximum allowable  $v$ 's associated with the chosen  $k$  is set as the translational velocity set point. Likewise, the maximum of the minimum allowable  $z$ 's associated with the chosen  $k$  is set as the altitude set point. The  $k$ ,  $v$ , and  $z$  set points are passed on to low-level controllers, which control the actuation of the vehicle to achieve the commanded set points. The set of available behaviors are as follows:



**Figure 5.** Logic of the pilot decision-support system at each cycle.

- 1) *Leader behavior*: At any given time, only one robot may act as a leader. The leader's task is to follow the a priori plan using a simple proportional compensator, as described in the "Autonomous Mode" section.
- 2) *Follower behavior*: The follower robots act to maintain an appropriate distance from the leader and from each other by weighted consensus, as described in the "Autonomous Mode" section.
- 3) *Threat-avoidance behavior*: All robots work individually to avoid detected external threats. Threat detection is implemented using a local sensor model. The threat-avoidance behavior only expresses a preference if an external threat is detected by the local sensor model.
- 4) *Pilot-guidance behavior*: The pilot may choose to take remote control of the leader robot. This behavior follows the input given by the pilot, as described in the "Autonomous Mode" section.

### Pilot Interface

The GUI (Figure 2) is implemented separately from the simulation environment, and it communicates with each agent via TCP/IP. It also communicates with the optimal control module to send state information and receive decision support. The GUI retains a copy of the mission (seen as the black curve in Figure 2) and initial way points set by the pilot to generate the mission plan (yellow dots in Figure 2). The roles of the UAV agents are distinguished by color. In autonomous mode, the leader UAV is red. In pilot-control mode, it is blue. Followers are always green, unless one is suggested as leader, in which case it becomes black.

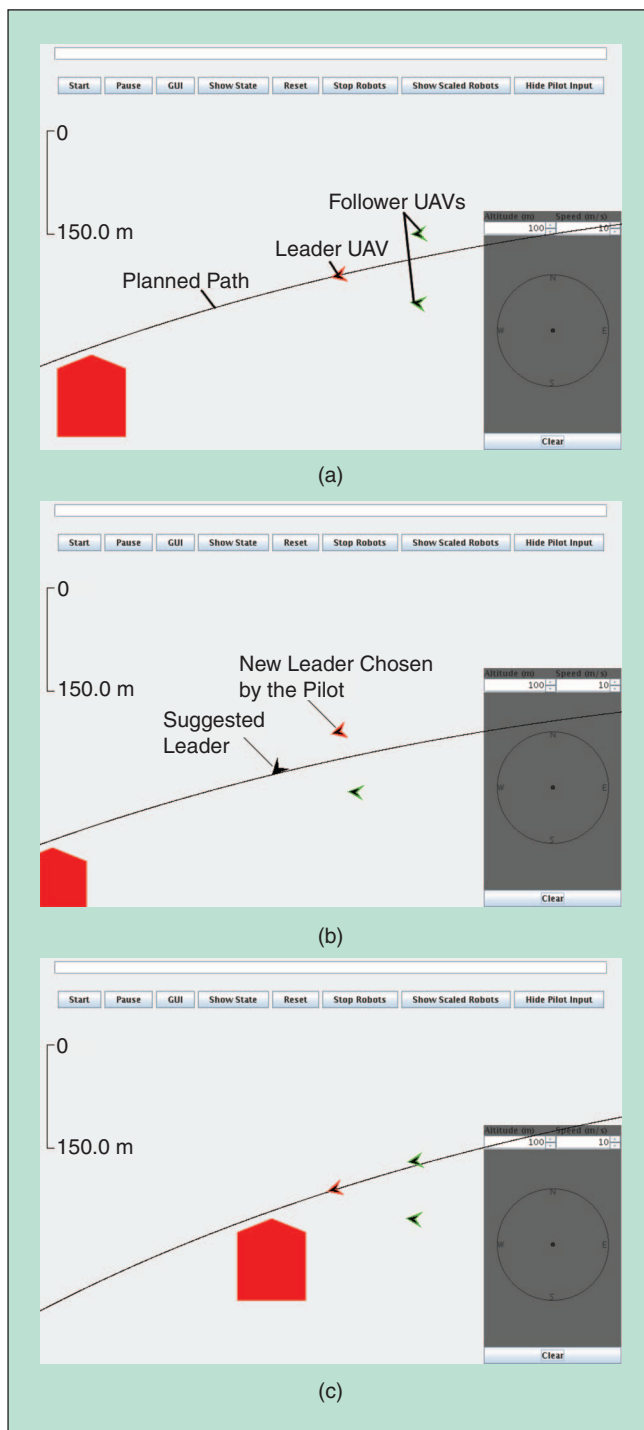
### Results

Demonstrations of the above-described system are organized to highlight three features: 1) leader selection, 2) decision support, and 3) pilot control. Background images are removed to show the results more clearly. For clarification, the roles of UAVs are marked on the figures.

Figures 6 demonstrates the pilot's ability to designate a leader UAV from the team of UAVs. Figure 6(a) depicts the pilot interface showing a team of three UAVs navigating along a predefined route. The designated leader is in front (marked and colored red), whereas the other two UAVs follow behind (marked and colored green). In Figure 6(b), the pilot selects one of the two follower UAVs to become the new leader. Immediately, the selected UAV moves to take its place at the front of the formation, whereas the former leader moves to take its new place as a follower in the formation. In Figure 6(c), the new steady-state behavior with the newly designated leader in the front of the formation is shown.

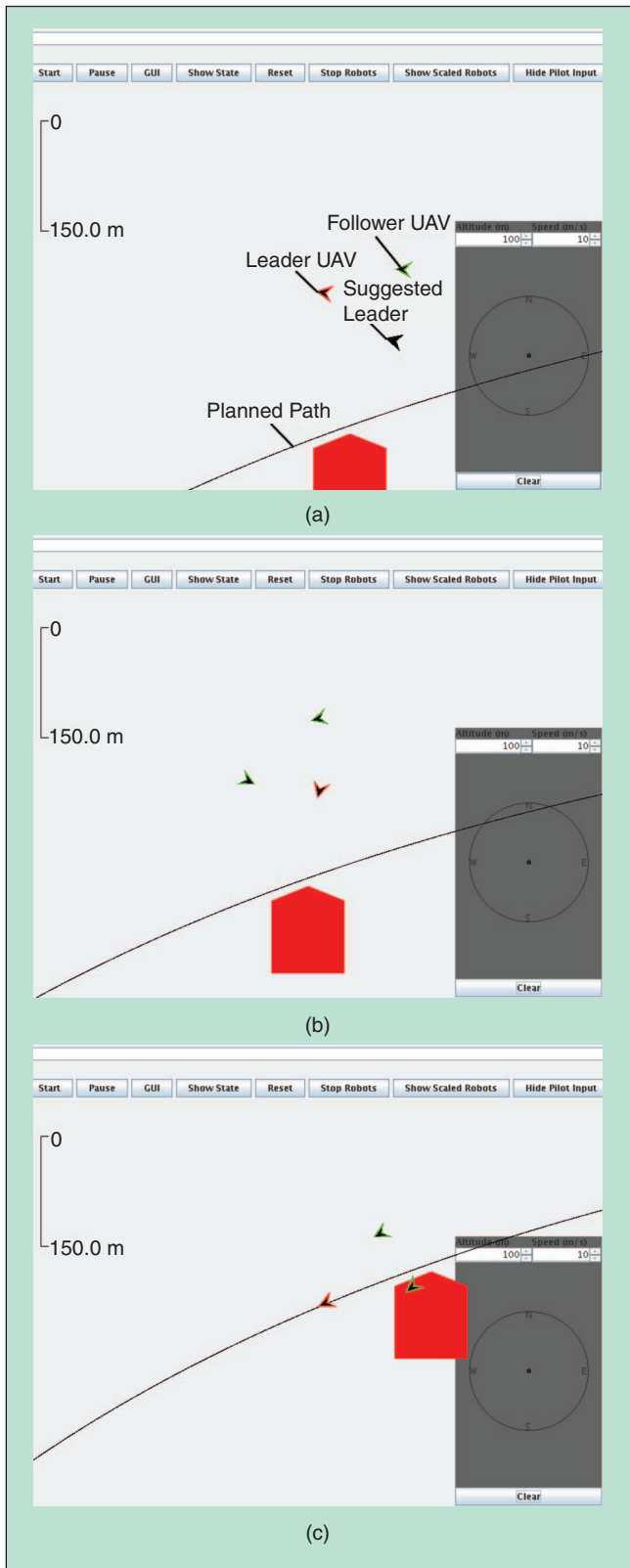
Figure 7 shows the pilot decision-support system in action. In Figure 7(a), the UAVs are in formation but offset from their commanded route. In this state, the UAV lowest in the screen would make the best leader, as evaluated by the pilot decision-support system. This is communicated by coloring the center of the UAV in question black. In Figure 7(b), the state of the UAVs immediately after the pilot has selected the suggested UAV to be the new leader. The new

*The optimal timing control module is designed to provide decision aid for the pilot.*



**Figure 6.** Demonstration of the pilot designating a new leader for the formation of UAVs. (a) The formation of UAVs follow the planned path. (b) The pilot chooses a new leader. (c) The new leader takes its position at the front of the formation.

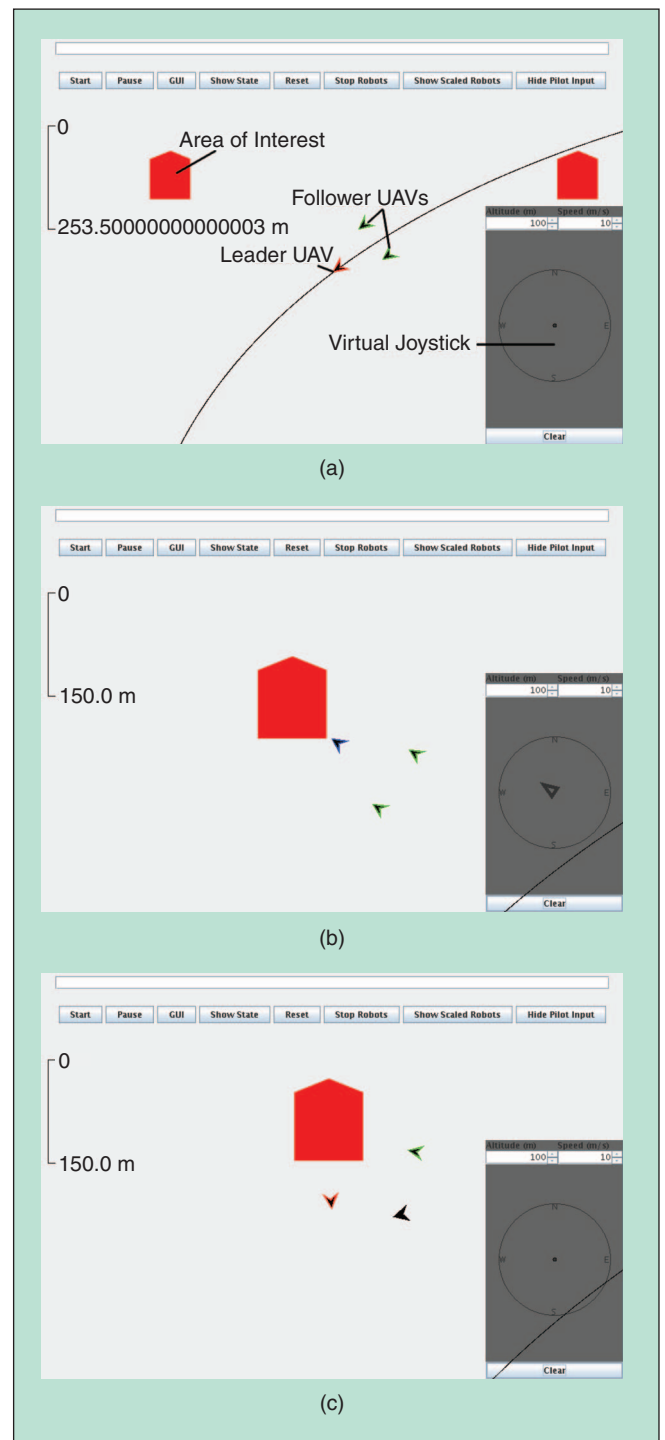




**Figure 7.** Demonstration of the decision-support system aiding the pilot by suggesting a new leader. (a) The formation of UAVs is off the planned path. The pilot decision-support system suggests a new leader to get back to the path faster. (b) The pilot takes the suggestion of the decision-support system, selecting the suggested UAV to be the new leader. (c) The formation moves back to the planned path.

leader moves to lead the other UAVs toward commanded route. Finally, in Figure 7(c) shows the UAVs in steady-state formation shortly thereafter.

The affordance for the pilot to manually navigate the leader UAV is demonstrated by Figure 8: Figure 8(a) shows the



**Figure 8.** Demonstration of the pilot exploring an area of interest by remote controlling the leader UAV. (a) The UAVs move along the planned path. An area of interest to the pilot lies to the northwest of the formation. (b) The pilot remote controls the leader to the area of interest. (c) The pilot returns the leader to autonomous control. The formation moves back to the planned path.



formation of UAVs following their commanded route. An area of interest to the pilot (marked by the red polygon) is to the northwest of the formation. To explore the area of interest, the pilot uses the virtual joystick (shown in the bottom right corner of the interface) to pilot the leader UAV, as shown in Figure 8(b). The UAV under the pilot's control is colored blue. The other two UAVs continue to operate autonomously, albeit as followers. In this manner, the pilot is essentially in remote control of the entire formation. Once the pilot is satisfied with the exploration achieved, the leader UAV is returned to autonomous operation. Figure 8(c) shows the fully autonomous formation returning to its mission.

## Acknowledgment

This work was sponsored by a grant from Rockwell Collins Advanced Technology Center.

## Keywords

Autonomous aerial vehicles, formation control, decision support.

## References

- [1] R. C. Arkin, "Cooperation without communication: Multiagent schema-based robot navigation," *J. Robot. Syst.*, vol. 9, no. 3, pp. 351–364, 1992.
- [2] R. C. Arkin, "Reactive control as a substrate for telerobotic systems," *IEEE Aerospace Electron. Syst. Mag.*, vol. 6, no. 6, pp. 24–31, 1991.
- [3] J. McLurkin, J. Smith, J. Frankel, D. Sotkowitz, D. Blau, and B. Schmidt, "Speaking swarmish: Human-robot interface design for large swarms of autonomous mobile robots," presented at the AAAI Spring Symp., Stanford, CA, Mar. 2006.
- [4] M. L. Cummings, "Human supervisory control of swarming networks," presented at the 2nd Annual Swarming: Autonomous Intelligent Networked Systems Conf., Arlington, VA, June 2004.
- [5] T. H. J. Collett, B. A. MacDonald, and B. P. Gerkey, "Player 2.0: Toward a practical robot programming framework," presented at the Australasian Conf. on Robotics and Automation, Sydney, Australia, Dec. 2005.
- [6] J. P. Desai, J. Ostrowski, and V. Kumar, "Controlling formations of multiple mobile robots. Proceedings of the 1998," in *Proc. IEEE Int. Conf. Robotics and Automation*, Lueven, Belgium, May 1998, pp. 2864–2869.
- [7] M. Ji, A. Muhammad, and M. Egerstedt, "Leader-based multi-agent coordination: Controllability and optimal control," in *Proc. IEEE American Control Conf.*, Minneapolis, MN, June 2006, pp. 1358–1363.
- [8] R. Olfati-Saber, J. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," in *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, 2007.
- [9] H. Axelsson, A. Muhammad, and M. Egerstedt, "Autonomous formation switching for multiple, mobile robots," in *Proc. IFAC Conf. Analysis and Design of Hybrid Systems*, Brittany, France, June 2003, pp. 147–152.
- [10] X. C. Ding, M. Powers, M. Egerstedt, and R. Young, "An optimal timing approach to controlling multiple UAVs," to be presented at the IEEE American Control Conf., St. Louis, MO, June 2009.
- [11] Y. Wardi, X. C. Ding, M. Egerstedt, and S. Azuma, "On-line optimization of switched-mode systems: Algorithms and convergence properties," in *Proc. IEEE Conf. Decision and Control*, New Orleans, LA, Dec. 2007, pp. 5005–5010.
- [12] J. Rosenblatt, "DAMN: A distributed architecture for mobile navigation," *J. Exp. Theor. Artif. Intell.*, vol. 9, no. 2, pp. 339–360, 1997.

- [13] D. Wooden, M. Powers, M. Egerstedt, H. Christensen, and T. Balch, "A modular, hybrid system architecture for autonomous, urban driving," *J. Aerospace Comput. Inform. Commun.*, vol. 4, no. 12, pp. 1047–1058, 2007.

**Xu Chu Ding** received his B.S. degree in 2004 and M.S. degree in 2007 in electrical and computer engineering and is pursuing his Ph.D. degree from the Georgia Institute of Technology. He is currently working as a graduate research assistant at the Georgia Robotics and Intelligent Systems Laboratory. His research interests include real-time optimal control of hybrid systems and multiagent networked systems.

**Matthew Powers** received his B.S. degree in mechanical engineering and computer science from Northwestern University in 2001. He is currently a Ph.D. candidate in the College of Computing at the Georgia Institute of Technology. His research interests include hybrid-control architectures, machine learning, and field robotics.

**Magnus Egerstedt** received his M.S. degree in engineering physics and Ph.D. degree in applied mathematics from the Royal Institute of Technology, Stockholm, in 1996 and 2000, respectively. He is currently an associate professor in the School of Electrical and Computer Engineering at the Georgia Institute of Technology, where he has been on the faculty since 2001. His research interests include optimal control and modeling and analysis of hybrid and discrete event systems, with an emphasis on motion planning and control of (teams of) mobile robots.

**Shih-Yih (Ryan) Young** received his M.S. and Ph.D. degrees in aerospace engineering from the University of Texas at Arlington in 1991 and 1994, respectively. He is currently a principal systems engineer with the Advanced Technology Center of Rockwell Collins Inc. His areas of expertise include integrated GPS/inertial navigation systems, GPS integrity algorithms, precision approach navigation, and intelligent guidance.

**Tucker Balch** completed his B.S. and Ph.D. degrees in computer science from Georgia Tech. He was a postdoctoral fellow for one year, and then a member of the research faculty at Carnegie Mellon before he returned to Georgia Tech in 2001. He is currently an associate professor in the School of Interactive and Intelligent Computing at Georgia Tech. His research interests include collaboration and communication in multirobot teams, and activity recognition and modeling in live social agent systems.

**Address for Correspondence:** Xu Chu Ding, School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA. E-mail: ding@ece.gatech.edu.