

Group Health Management of UAV Teams With Applications to Persistent Surveillance

Brett Bethke, Jonathan P. How, and John Vian

Abstract—Unmanned aerial vehicles (UAVs) are well-suited to a wide range of mission scenarios, such as search and rescue, border patrol, and military surveillance. The complex and distributed nature of these missions often requires teams of UAVs to work together. Furthermore, overall mission performance can be strongly influenced by vehicle failures or degradations, so an autonomous mission system must account for the possibility of these anomalies if it is to maximize performance. This paper presents a general health management methodology for designing mission systems that can anticipate the negative effects of various types of anomalies on the future mission state and choose actions that mitigate those effects. The formulation is then specialized to the problem of providing persistent surveillance coverage using a group of UAVs, where uncertain fuel usage dynamics and strong interdependence effects between vehicles must be considered. Finally, the paper presents results showing that the health-aware persistent surveillance planner based on this formulation exhibits excellent performance in both simulated and real flight test experiments.

I. INTRODUCTION

Unmanned aerial vehicles (UAVs) are becoming increasingly sophisticated in terms of hardware capabilities. Advances in sensor systems, onboard computational platforms, energy storage, and other enabling technologies have made it possible to build a huge variety of UAVs for a range of different mission scenarios [1], [2]. Many of the mission scenarios of interest, such as persistent surveillance, are inherently long-duration and require coordination of multiple cooperating UAVs in order to achieve the mission objectives. In these types of missions, a high level of autonomy is desired due to the logistical complexity and expense of direct human control of each individual vehicle. Currently, autonomous mission planning and control for multi-agent systems is an active area of research [3]–[7]. Some of the issues in this area are similar to questions arising in manufacturing systems [8], [9] and air transportation [10]–[16]. While these efforts have made significant progress in understanding how to handle some of the complexity inherent in multi-agent problems, there remain a number of open questions in this area.

This paper investigates one important question that is referred to as the *health management problem for multi-agent systems* [17], [18]. Designs of current and future UAVs increasingly incorporate a large number of sensors for

monitoring the health of the vehicle’s own subsystems. For example, sensors may be installed to measure the temperature and current of electric motors, the effectiveness of the vehicle’s control actuators, or the fuel consumption rates in the engine. On a typical UAV, sensors may provide a wealth of data about a large number of vehicle subsystems. By making appropriate use of this data, a health-aware autonomous system may be able to achieve a higher level of overall mission performance, as compared to a non-health-aware system, by making decisions that account for the current capabilities of each agent. For example, in a search and track mission, utilization of sensor health data may allow an autonomous system to assign the UAVs with the best-performing sensors to the search areas with the highest probability of finding the target.

Utilization of the current status of each vehicle is an important aspect of the health management problem. Another important aspect is the ability not only to react to the current status, but to consider the implications of future changes in health status or failures on the successful outcome of the mission. This predictive capability is of paramount importance, since it may allow an autonomous system to avoid an undesirable future outcome. For example, if a UAV tracking a high value target is known to have a high probability of failure in the future, the autonomous system may be able to assign a backup vehicle to track the target, ensuring that the tracking can continue even if one of the vehicles fails.

This paper addresses these health management issues and develops a general framework for thinking about the health management problem. It then specializes the discussion to the persistent surveillance problem with a focus on group fuel management when there is uncertainty in the fuel consumption dynamics. This work builds on previous health management techniques developed for the persistent surveillance problem [17]. While the previous work focused on embedding health-aware heuristics into an already-existing mission management algorithm, this paper develops a new formulation of the problem in which health-aware behaviors emerge automatically. Finally, simulated and real flight results are provided which demonstrate the effectiveness of the new formulation.

II. HEALTH MANAGEMENT IN MULTI-AGENT SYSTEMS

The term *health management* is often used in different contexts, so it can be difficult to define exactly what it means for a multi-agent system to incorporate health management techniques. To make the problem more precise, we can

B. Bethke is a Ph.D. Candidate, Dept. of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA 02139
bbethke@mit.edu

J. P. How is a Professor of Aeronautics and Astronautics, MIT,
jhow@mit.edu

J. Vian is a Technical Fellow, Boeing Phantom Works, Seattle, WA
john.vian@boeing.com

define a number of general properties that we would like a multi-agent system to exhibit. Once these properties are defined, potential design methodologies for incorporating health management into such systems can be evaluated.

In the context of multi-agent systems, health management refers to accounting for changing resource or capability levels of the agents. These changes may be caused by failures, degradations, or off-nominal conditions (of actuators, sensors, propulsion systems, etc), or by unpredictable events in the environment. Broadly speaking, we would like a multi-agent system to exhibit the following properties:

- 1) The system should be proactive. That is, the system should be capable of “looking into the future” to anticipate events that are likely to occur, and given this information, actively select a course of action that leads to a desirable state and/or avoids an undesirable state. In contrast, a reactive system is incapable of making such future predictions (or cannot effectively use such predictions if they are available). Thus, a reactive system can only respond to failures after they occur, instead of trying to avoid them in the first place.
- 2) The system should manage health information at the group, not just the individual, level. In most multi-agent mission scenarios, there are strong coupling effects between vehicles that must be accounted for. For example, in the multi-UAV task assignment problem, failure of a single UAV may necessitate reassigning all the other UAVs to different targets in order to continue the mission. These coupling effects may be very complex, depending on the mission. Nevertheless, they must be considered if the system is to be robust to changing conditions.

A. Design Considerations

Given the above properties, we now consider some of their implications for design of multi-agent systems. First, Property 1 implies that the system must have a model of its environment in order to anticipate the system state in the future. This model should account for how the current control action will influence the future state. Furthermore, since many of the events of interest in the health management problem - such as failures - cannot be predicted with certainty, the model should be stochastic, and the system should account for the relative probability of possible future events in selecting actions. In general, then, the system model will be of the form:

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) \quad (1)$$

where \mathbf{x} is the system state, \mathbf{u} are the control inputs, \mathbf{w} are random variables that capture the uncertainty in the system dynamics, and k is a discrete time index.

If the system is to proactively choose control actions that lead to desirable future states, it therefore must have some method of ascertaining whether a given state is desirable or not. A natural way to accomplish this is to define a cost function $g(\mathbf{x})$ which maps states \mathbf{x} to a scalar cost value.

Property 2 also has implications for the design of health-enabled systems. Normally, the interdependence between agents plays a large role in overall mission performance. Therefore, the system model $\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k)$ and cost function $g(\mathbf{x})$ should capture this interdependence so that knowledge of how health related events such as failures can be exploited.

The considerations presented here lead naturally to the idea of applying dynamic programming (DP) techniques to the multi-agent health management problem. DP provides an attractive method for achieving proactive behavior, since the problem solution method involves calculating the actions that minimize not only the current cost, but also the expected future cost [8]. Furthermore, formulating the health management problem as a dynamic program allows the interdependence between agents to be encoded naturally in the system model (Equation 1).

Consideration must be given to computational issues in the formulation of the system model. The choice of too complex a model may lead to an unnecessarily large state space, rendering the problem difficult to solve and implement in real-time. On the other hand, selecting a model which is too simplified may not adequately capture important aspects of the problem, leading to poor performance. Thus, a balance between model complexity and computational tractability must be struck in the formulation of the health management problem. In the following sections, we use the multi-vehicle persistent surveillance problem as an example to show how an appropriate dynamic program capturing the important aspects of the problem can be formulated and solved, and show that the resulting control policy exhibits the desirable properties discussed in this section.

III. PERSISTENT SURVEILLANCE WITH GROUP FUEL MANAGEMENT

Health management techniques are an enabling technology for multi-vehicle persistent surveillance missions. In the model of this scenario considered here, there is a group of n UAVs equipped with cameras or other types of sensors. The UAVs are initially located at a base location, which is separated by some (possibly large) distance from the surveillance location. The objective of the problem is to maintain a specified number r of requested UAVs over the surveillance location at all times.

The UAV vehicle dynamics provide a number of interesting health management aspects to the problem. In particular, management of fuel is an important health management concern. The vehicles have a certain maximum fuel capacity F_{max} , and we assume that the rate \dot{F}_{burn} at which they burn fuel may vary stochastically during the mission due to aggressive maneuvering that may be required for short time periods, engine wear and tear, adverse environmental conditions, etc. Thus, the total flight time each vehicle may achieve on a full tank of gas is a random variable, and we would like to account for this uncertainty in the problem. If a vehicle runs out of fuel while in flight, it crashes and is lost. Finally, when the vehicle returns to base, it begins

refueling at a rate \dot{F}_{refuel} (i.e. the vehicles take a finite time to refuel).

Another health management concern we will model in the problem is the possibility for randomly-occurring vehicle failures. These may be due to sensors, engines, control actuators, or other mission-critical systems failing in flight.

A. Dynamic Program Formulation

Given the description of the persistent surveillance problem, a suitable dynamic program can now be formulated. The program is defined by its state vector \mathbf{x} , control vector \mathbf{u} , state transition model $\mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k)$, cost function $g(\mathbf{x})$, and discount factor α .

1) *State Space*: The state of each UAV is given by two scalar variables describing the vehicle's flight status and fuel remaining. The flight status y_i describes the UAV location,

$$y_i \in \{Y_b, Y_0, Y_1, \dots, Y_s, Y_c\} \quad (2)$$

where Y_b is the base location, Y_s is the surveillance location, $\{Y_0, Y_1, \dots, Y_{s-1}\}$ are transition states between the base and surveillance locations (capturing the fact that it takes finite time to fly between the two locations), and Y_c is a special state denoting that the vehicle has crashed.

Similarly, the fuel state f_i is described by a discrete set of possible fuel quantities,

$$f_i \in \{0, \Delta f, 2\Delta f, \dots, F_{max} - \Delta f, F_{max}\} \quad (3)$$

where Δf is an appropriate discrete fuel quantity.

The total system state vector \mathbf{x} is thus given by the states y_i and f_i for each UAV, along with r , the number of requested vehicles:

$$\mathbf{x} = (y_1, y_2, \dots, y_n; f_1, f_2, \dots, f_n; r)^T \quad (4)$$

The size of the state space N_{ss} is found by counting all possible values of \mathbf{x} , which yields:

$$N_{ss} = (n+1) \left((Y_s + 3) \left(\frac{F_{max}}{\Delta f} + 1 \right) \right)^n \quad (5)$$

2) *Control Space*: The controls u_i available for the i^{th} UAV depend on the UAV's current flight status y_i .

- If $y_i \in \{Y_0, \dots, Y_s - 1\}$, then the vehicle is in the transition area and may either move away from base or toward base: $u_i \in \{+, -\}$
- If $y_i = Y_c$, then the vehicle has crashed and no action for that vehicle can be taken: $u_i = \emptyset$
- If $y_i = Y_b$, then the vehicle is at base and may either take off or remain at base: $u_i \in \{\text{"take off"}, \text{"remain at base"}\}$
- If $y_i = Y_s$, then the vehicle is at the surveillance location and may loiter there or move toward base: $u_i \in \{\text{"loiter"}, \text{"-"}\}$

The full control vector \mathbf{u} is thus given by the controls for each UAV:

$$\mathbf{u} = (u_1, \dots, u_n)^T \quad (6)$$

3) *State Transition Model*: The state transition model (Equation 1) captures the qualitative description of the dynamics given at the start of this section. The model can be partitioned into dynamics for each individual UAV.

The dynamics for the flight status y_i are described by the following rules:

- If $y_i \in \{Y_0, \dots, Y_s - 1\}$, then the UAV moves one unit away from or toward base as specified by the action $u_i \in \{+, -\}$ with probability $(1 - p_{crash})$, and crashes with probability p_{crash} .
- If $y_i = Y_c$, then the vehicle has crashed and remains in the crashed state forever afterward.
- If $y_i = Y_b$, then the UAV remains at the base location with probability 1 if the action "remain at base" is selected. If the action "take off" is selected, it moves to state Y_0 with probability $(1 - p_{crash})$, and crashes with probability p_{crash} .
- If $y_i = Y_s$, then if the action "loiter" is selected, the UAV remains at the surveillance location with probability $(1 - p_{crash})$, and crashes with probability p_{crash} . Otherwise, if the action "-" is selected, it moves one unit toward base with probability $(1 - p_{crash})$, and crashes with probability p_{crash} .
- If at any time the UAV's fuel level f_i reaches zero, the UAV transitions to the crashed state ($y_i = Y_c$) with probability 1.

The dynamics for the fuel state f_i are described by the following rules:

- If $y_i = Y_b$, then f_i increases at the rate \dot{F}_{refuel} (the vehicle refuels)
- If $y_i = Y_c$, then the fuel state remains the same (the vehicle is crashed)
- Otherwise, the vehicle is in a flying state and burns fuel at a stochastically modeled rate: f_i decreases at the rate \dot{F}_{burn} with probability $p_{fnominal}$ and decreases at the rate $2\dot{F}_{burn}$ with probability $(1 - p_{fnominal})$.

4) *Cost Function*: The cost function $g(\mathbf{x})$ has three distinct components due to loss of surveillance area coverage, vehicle crashes, and fuel usage, and can be written as

$$g(\mathbf{x}) = C_{loc} \max\{0, (r - n_s(\mathbf{x}))\} + C_{crash} n_{crashed}(\mathbf{x}) + C_f n_f(\mathbf{x})$$

where:

- $n_s(\mathbf{x})$: number of UAVs in surveillance area in state \mathbf{x} ,
- $n_{crashed}(\mathbf{x})$: number of crashed UAVs in state \mathbf{x} ,
- $n_f(\mathbf{x})$: total number of fuel units burned in state \mathbf{x} ,

and C_{loc} , C_{crash} , and C_f are the relative costs of loss of coverage events, crashes, and fuel usage, respectively.

IV. DYNAMIC PROGRAM SOLUTION AND SIMULATION RESULTS

In order to solve the DP, a software framework written in Python was developed. The framework allows for generic DPs to be programmed by specifying appropriate system transition, action, and cost functions. Once programmed, the framework applies the value iteration algorithm to iteratively solve Bellman's equation for the optimal cost-to-go $J^*(\mathbf{x}_k)$.

The results of this computation are stored on disk as a lookup table. Once $J^*(\mathbf{x}_k)$ is found for all states \mathbf{x}_k , the optimal control action $\mathbf{u}^*(\mathbf{x}_k)$ can be quickly computed by choosing the action which minimizes the expected future cost [8]:

$$\mathbf{u}^*(\mathbf{x}_k) = \arg \min_u E_{\mathbf{x}_{k+1}} [g(\mathbf{x}_{k+1}) + \alpha J^*(\mathbf{x}_{k+1})] \quad (7)$$

While the controller runs very quickly once $J^*(\mathbf{x}_k)$ is known, a potentially large amount of computation is required to find $J^*(\mathbf{x}_k)$. In the experiments done to date, the largest DP solved was a three-vehicle problem ($n = 3$) with $s = 1$, $F_{max} = 16$, and $\Delta f = 1$, resulting in a state space size (Equation 5) of approximately 1.2 million states. For this problem, the value iteration algorithm took roughly 36 hours to complete on a 2.4GHz Pentium 4 PC.

To test the performance of the health-enabled control strategy, $J^*(\mathbf{x}_k)$ was calculated for several different sets of problem parameters, and the resulting optimal control law $\mathbf{u}^*(\mathbf{x}_k)$ was found. The system dynamics were then simulated under the action of $\mathbf{u}^*(\mathbf{x}_k)$. Unless otherwise specified, many of the problem parameters were held fixed at the following values: $Y_s = 1$, $p_{crash} = 0.01$, $p_{f_{nominal}} = 0.90$, $\Delta f = 1$, $\dot{F}_{burn} = 1$, $C_{loc} = 8.0$, $C_{crash} = 50.0$, and $C_f = 1.0$.

Figure 1 shows a simulation result for the three-vehicle ($n = 3$) case with two vehicles requested ($r = 2$). In this experiment, the fuel capacity F_{max} was 12 and the refuel rate \dot{F}_{refuel} was 4. The figure shows the flight status y_i for each of the UAVs in the lower graph (where $-2 =$ crashed, $-1 =$ base location, $0 =$ transition area, and $1 =$ surveillance location) and the fuel state f_i for each UAV in the top graph. These results exhibit a number of desirable behaviors. First, note that the system commands two UAVs to take off at time $t = 0$ and fly immediately to the surveillance area to establish initial coverage. If the system were to leave these two UAVs in the surveillance area until both were close to the minimum fuel level needed to return to base, they would both have to leave at the same time, resulting in coverage gap. However, because the system anticipates this problem, it instead recalls the green UAV to base well before it has reached the minimum fuel level. In addition, it launches the blue UAV at the right moment so that the blue UAV arrives at the surveillance location precisely when the green UAV is commanded to return to base, resulting in continuous coverage throughout the vehicle swap. This initial command sequence allows the system to set up a regular pattern of vehicle swaps which results in the greatest possible coverage. Another desirable feature of the solution is that the system tries to arrange for the UAVs to return to base with a small reserve quantity of fuel remaining. This behavior is a proactive hedge against the uncertainty in the fuel burn dynamics, reducing the probability that the vehicle will run out of fuel before reaching base due to one or more higher-than-average fuel burn events.

Note that in this example, the parameters were chosen so that the ratio of fuel capacity to refueling rate of the UAVs was slightly less than the minimum necessary to maintain perfectly continuous coverage. In other words, it

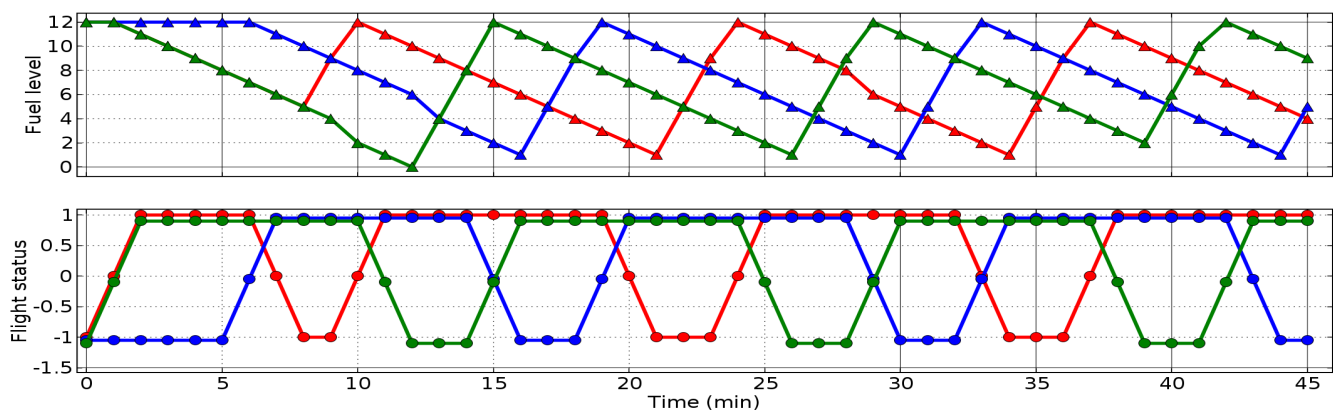
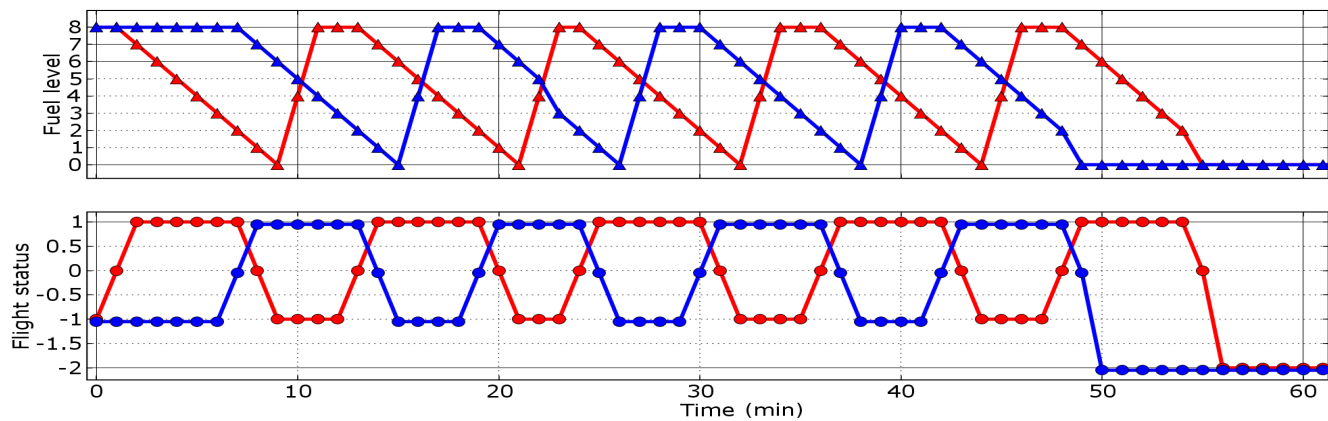
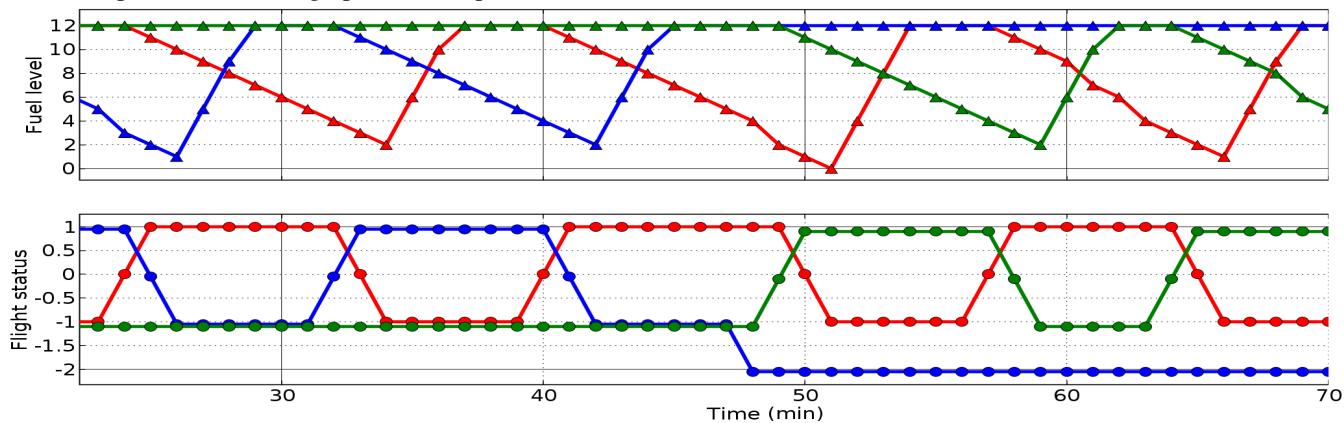
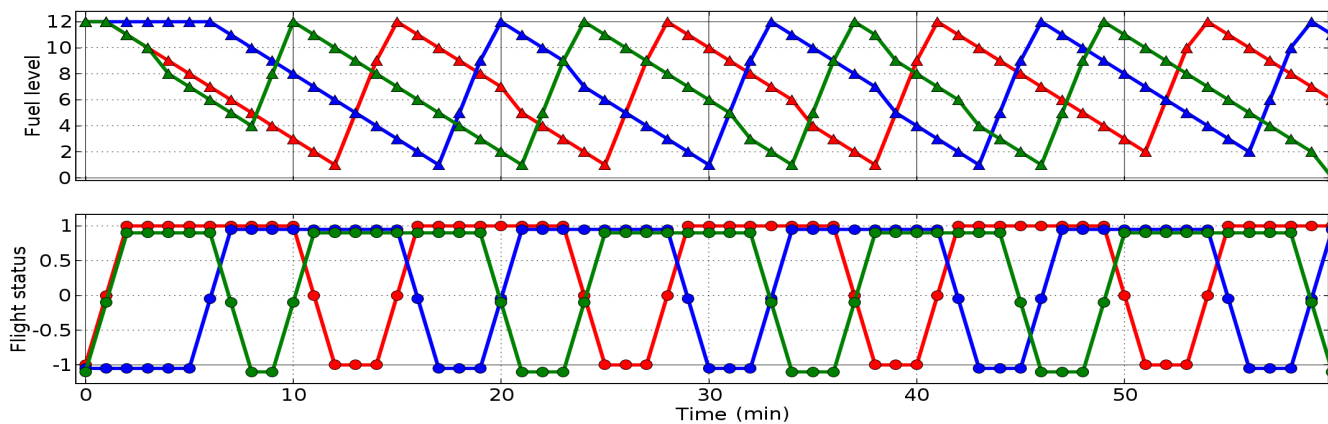
was a “hard” problem where even an optimal solution must exhibit small periodic coverage gaps, such as the one seen at time $t = 20$. In spite of the intentionally challenging problem setup, the system performed extremely well. It is especially interesting to note that, as discussed, the system exhibits a number of distinct and relatively sophisticated behaviors, such as bringing back vehicles early and the hedge against fuel burn uncertainty. These behaviors emerge naturally and automatically due to the formulation and solution method applied to the problem.

A further example is shown in Figure 2. This experiment was an “easy” problem where three UAVs ($n = 3$) with high fuel capacity and refuel rates ($F_{max} = 12$, $\dot{F}_{refuel} = 4$) were available to satisfy a single request ($r = 1$). In this example, the system initially sets up a regular switching pattern between only the red and blue UAVs. The green UAV is left in reserve at the base location since it is not needed to satisfy the requested number of vehicles, and commanding it to fly would consume extra fuel (recall that fuel burn is a small part of the cost function $g(\mathbf{x})$). However, at time $t = 47$, the blue UAV suffers a random failure during take off and crashes (this failure was a result of having a nonzero value p_{crash} for the probability of random crashes, which the system can do nothing to prevent). The system immediately commands the green UAV to take off to replace the blue UAV, while simultaneously leaving the red UAV in the surveillance area for an extra time unit to provide coverage until the green UAV arrives on station. After that point, the switching pattern is maintained using the green and red UAVs. Even though a catastrophic failure occurs in this example, coverage is maintained at 100% at all times.

A final example, shown in Figure 3, illustrates the value of the problem formulation presented here, which accounts for the inherent uncertainty in the fuel burn dynamics. In this example, an optimal policy for the purely deterministic problem ($p_{f_{nominal}} = 1$), which does not account for this uncertainty, was calculated. This policy was then simulated using the true, uncertain, dynamics ($p_{f_{nominal}} = 0.90$). The fuel state graph in Figure 3 reveals that the proactive hedging behavior is lost with the policy based on deterministic dynamics; the policy always brings vehicles back to base with exactly zero fuel remaining. Unfortunately, this strategy is highly risky, since any off-nominal fuel burn events which occur when the vehicle is returning to base are guaranteed to produce a crash. In the simulation shown, both vehicles crash after a short period of time. Indeed, in every simulation run to date, both UAVs end up crashing after a short period of time. The reason for this is that for each vehicle cycle, the probability of a crash is $1 - p_{f_{nominal}} = 0.100$, as opposed to the control policy based on uncertain fuel dynamics, which effectively reduces this probability to $(1 - p_{f_{nominal}})^3 = 0.001$. When viewed in this light, the uncertainty-based controller has a clear advantage.

V. FLIGHT RESULTS

A series of flight experiments using the MIT RAVEN testbed [19] were carried out to evaluate the performance



of the health enabled, DP-based control scheme developed in this paper. In these experiments, the controller had access to $n = 3$ quadrotor UAVs and was commanded to maintain $r = 2$ UAVs over a surveillance area. The parameters of the experiments were adjusted to make the problem intentionally hard for the controller. That is, the distance from base to the surveillance area was large relative to the amount of fuel each UAV could carry, thus necessitating rapid swapping of the UAVs in order to maintain the desired coverage.

Flight results for a typical experiment are shown in Figure 4. Despite the difficulty of the problem, the performance of the controller is excellent, providing nearly continuous (97%) coverage over the 45-minute duration of the mission. Furthermore, the qualitative behavior of the flight controller is identical to that seen in simulation. This demonstrates that the desirable properties exhibited by the DP-based controller can be successfully implemented on a real hardware system.

VI. CONCLUSIONS

Health management in multi-agent systems is a complex and difficult problem, but any autonomous control system that is *health-aware* should be proactive (capable of looking into the future to foresee the consequences of stochastically-occurring failures or off-nominal events) as well as capable of exploiting the interdependencies between agents at the group level. Dynamic programming is shown to provide a natural framework for posing the health management problem. The persistent surveillance example demonstrates how a suitable DP may be formulated to capture the important health-related issues in a problem, and simulation results of the resulting controller demonstrate a large number of distinct, “smart” behaviors, all of which contribute to the overall excellent performance of the controller. Flight experiments show that the DP-based controller can be effectively implemented on a real system while retaining the excellent performance observed in simulation.

The natural emergence of the “smart” behaviors exhibited by the DP controller is appealing, especially considering the complexity and difficulty of manually building heuristics to capture those same behaviors. For example, the Python code for the persistent surveillance problem implemented in this paper has fewer than 200 lines; code for all of the heuristics necessary to implement the same behaviors would likely be far longer and more complex. This difficulty with the heuristic approach only increases as the health management problem to be solved grows in complexity, where the correct heuristics to use may be far from obvious.

VII. ACKNOWLEDGMENTS

The authors would like to thank Jim McGrew, Brandon Luders, Josh Redding, and Spencer Ahrens at MIT for their assistance in this research. The first author is sponsored by the Hertz Foundation and the American Society for Engineering Education. The research has also been supported by the Boeing Company and by AFOSR grant FA9550-04-1-0458.

REFERENCES

- [1] American Institute of Aeronautics and Astronautics), “Worldwide UAV Roundup,” Available at <http://www.aiaa.org/images/PDF/WilsonChart.pdf>, 2007.
- [2] Office of the Secretary of Defense, “Unmanned Aircraft Systems Roadmap, available online <http://www.acq.osd.mil/usd/Roadmap/Final2.pdf>,” OSD, Tech. Rep., 2005.
- [3] H. Paruanak, S. Brueckner, and J. Odell, “Swarming Coordination of Multiple UAV’s for Collaborative Sensing,” in *Proceedings of the 2nd AIAA Unmanned Unlimited Systems, Technologies, and Operations Aerospace Conference*, San Diego, CA, September 2003.
- [4] P. Gaudiano, B. Shargel, E. Bonabeau, and B. Clough, “Control of UAV SWARMS: What the Bugs Can Teach Us,” in *Proceedings of the 2nd AIAA Unmanned Unlimited Systems, Technologies, and Operations Aerospace Conference*, San Diego, CA, September 2003.
- [5] A. Kott, *Advanced Technology Concepts for Command and Control*. Xlibris Corporation, 2004.
- [6] R. M. Murray, “Recent research in cooperative control of multi-vehicle systems,” *ASME Journal of Dynamic Systems, Measurement, and Control* (submitted) special issue on the Analysis and Control of Multi Agent Dynamic Systems, 2007.
- [7] M. J. Hirsch, P. M. Pardalos, R. Murphey, and D. Grundel, Eds., *Advances in Cooperative Control and Optimization. Proceedings of the 7th International Conference on Cooperative Control and Optimization*, ser. Lecture Notes in Control and Information Sciences. Springer, Nov 2007, vol. 369.
- [8] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Belmont, MA: Athena Scientific, 2000.
- [9] J. C. Hartman and J. Ban, “The series-parallel replacement problem,” *Robotics and Computer Integrated Manufacturing*, vol. 18, pp. 215–221, 2002.
- [10] D. Bertsimas and S. S. Patterson, “The Air Traffic Flow Management Problem with Enroute Capacities,” *Operations Research*, vol. 46, no. 3, pp. 406–422, May-June 1998.
- [11] K. Andersson, W. Hall, S. Atkins, and E. Feron, “Optimization-Based Analysis of Collaborative Airport Arrival Planning,” *Transportation Science*, vol. 37, no. 4, pp. 422–433, November 2003.
- [12] J. Rosenberger, E. Johnson, and G. Nemhauser, “Rerouting Aircraft for Airline Recovery,” *Transportation Science*, vol. 37, no. 4, pp. 408–421, November 2003.
- [13] C. Barnhart, F. Lu, and R. Shenoi, “Integrated Airline Scheduling,” in *Operations Research in the Airline Industry*, ser. International Series in Operations Research and Management Science, G. Yu, Ed. Kluwer Academic Publishers, 1998, vol. 9, pp. 384–422.
- [14] L. Clarke, E. Johnson, G. Nemhauser, and Z. Zhu, “The Aircraft Rotation Problem,” *Annals of Operations Research*, vol. 69, pp. 33–46, 1997.
- [15] R. Gopalan and K. Talluri, “The Aircraft Maintenance Routing Problem,” *Operations Research*, vol. 46, no. 2, pp. 260–271, March-April 1998.
- [16] J. Hingle and A. Johnson, “Flight Schedule Planning with Maintenance Considerations,” 2005, submitted to OMEGA, The International Journal of Management Science.
- [17] M. Valenti, B. Bethke, J. How, D. Pucci de Farias, J. Vian, “Embedding Health Management into Mission Tasking for UAV Teams,” in *Proceedings of the American Control Conference*, New York, NY, June 2007.
- [18] M. Valenti, D. Dale, J. How, and J. Vian, “Mission Health Management for 24/7 Persistent Surveillance Operations,” in *AIAA Guidance, Control and Navigation Conference*, Myrtle Beach, SC, August 2007.
- [19] J. How, B. Bethke, A. Frank, D. Dale, and J. Vian, “Real-Time Indoor Autonomous Vehicle Test Environment,” *IEEE Control Systems Magazine*, vol. 28, pp. 51–64, April 2008.