2011 American Control Conference
on O'Farrell Street, San Francisco, CA, USA
June 29 - July 01, 2011

Proactive Planning for Persistent Missions using Composite Model-Reference Adaptive Control and Approximate Dynamic Programming

Joshua Redding, Zac Dydek and Jonathan P. How
Matthew A. Vavrina and John Vian

Abstract— **This paper extends prior work on the persistent mission problem where real-time changes in agent capability are included in the problem formulation. Here, we couple the mission planner with a low-level adaptive controller in real-time to: (1) Provide robustness against actuator degradations and (2) Use parameters internal to the adaptive controller to provide valuable insight into the physical capabilities of the agent. These parameters, in conjunction with sensor health information, form a more complete measure of agent capability, which is used online and in forward planning to enable both reactive and proactive behavior. Flight results are presented for a persistent mission scenario where actuator degradations are induced to demonstrate: (1) The robustness of the composite adaptive controller and its successful integration with the agent-level health-monitoring and mission-level planning systems and (2) The reactive and proactive qualities of the planning system in persistently re-tasking agents under actuator and sensor health degradations.**

I. Introduction

In the context of teams of coordinating agents, many mission scenarios of interest are inherently long-duration and require a high level of agent autonomy due to the expense and logistical complexity of direct human control over the individual agents. Such long-duration missions are practical scenarios that can show well the benefits of agent cooperation. However, they can also accelerate mechanical wear and tear on an agent's hardware platform and increase the likelihood of related failures. It is important that the planning system accounts for the possibility of these failures when constructing a mission plan. Planning problems that coordinate the actions of multiple agents, where each of which is subject to failures are referred to as *multi-agent health management* problems [1,2].

One approach to multi-agent health management problems is *proactive* [3] by constructing a plan based on stochastic models which capture the inherent possibility of failures. Using stochastic models increases the complexity of computing the plan, as it then becomes necessary to optimize *expected* performance, where the expectation is taken over all possible scenarios that might occur. However, since proactive planners prescribe actions that mitigate the consequences of possible future failures, the resulting mission performance can be much better than that achieved by a reactive planner (naturally, depending on the validity of the underlying stochastic models).

This research aims to develop a proactive mission planner for a team of autonomous agents in the presence of sensor and actuator degradations. The problem becomes challenging when stochastic models for sensor and actuator health are included in the formulation as well as stochastic dynamics for fuel consumption. Previous work has shown dynamic programming techniques to be a well-suited solution approach when planning for persistent missions [4]. For example, in [4] the planning problem is formulated as a Markov decision process (MDP) and solved using value iteration. The resulting optimal control policy showed a number of desirable properties, including the ability to proactively recall vehicles to base with an extra, reserve quantity of fuel which resulted in fewer vehicle losses and a higher average mission performance. Similarly, the persistent mission planner is formulated here as an MDP and an approximate dynamic programming technique called Bellman residual elimination [5] is implemented to compute an approximate control policy.

In addition to broadening the on-line metric for agent capability to include actuator health, another important contribution of this research is to couple the MDP mission planner with a low-level adaptive controller in real-time. To accomplish this, the vehicle's baseline controller is retrofitted with a composite model-reference adaptive controller (CMRAC) designed to provide robustness against actuator degradations. Parameters internal to the adaptive controller provide valuable insight into actuator health and therefore into the physical capabilities of the agent [6]. The agent's health monitoring and diagnostic system (HMDS) utilizes these adaptive parameters in conjunction with sensor health information to form a more complete measure of agent capability in real-time. This capability metric is used online and in forward planning to enable both reactive and proactive behaviors against sensor and actuator degradations.

The remainder of the paper is outlined as follows: The multi-agent persistent mission planning problem is described and formulated in Section II, following which the technical details of the HMDS are given in Section III. Results of flight tests performed at Boeing Vehicle Swarm Technology Lab (VSTL) [7] are then provided in Section IV, followed by some concluding remarks.

J. Redding is a Ph.D. candidate in the Aerospace Controls Lab, MIT
Z. Dydek is a Ph.D. from the Active Adaptive Controls Lab, MIT
J. How is a Professor of Aeronautics and Astronautics, MIT
{jredding,zac,jhow}@mit.edu
M. Vavrina is a Research Engineer at Boeing R&T, Seattle, WA
J. Vian is a Technical Fellow at Boeing R&T, Seattle, WA
{matthew.a.vavrina,john.vian}@boeing.com

978-1-4577-0079-8/11/$26.00 ©2011 AACC

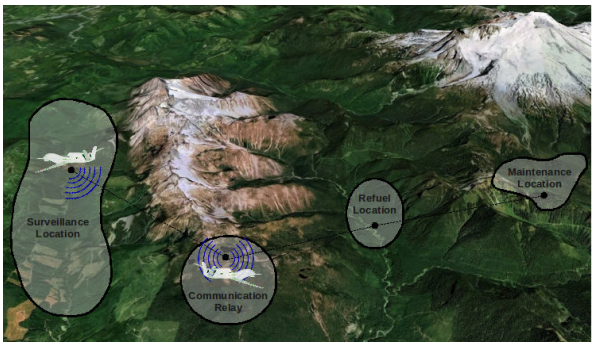

Fig. 1. N agents cooperate to continuously survey a specified region and to track any objects of interest discovered there. This behavior is to be maintained even under sensor and actuator health degradations.

II. Problem Formulation

Figure 1 depicts a possible theater of operation for such a mission with several distinct "locations". Agents coordinate and move between these locations such that a specific number are in the surveillance location at any given time to search for new targets and track those already discovered. The maintenance base location is for agents who have suffered sensor and/or actuator degradations while the refuel base location is more conveniently located so as to minimize swap times for fully operational agents only needing fuel.

Given the qualitative description of the persistent surveillance problem, an MDP can now be formulated. An infinite-horizon, discounted MDP is specified by $(\mathcal{S}, \mathcal{A}, P, g)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P_{ij}(u)$ gives the transition probability from state i to state j under action u, and $g(i,u)$ gives the cost of taking action u in state i. We assume that the MDP model is known and is a function of the agent health metric (detail in Section III). Future costs are discounted by a factor $0 < \alpha < 1$. A policy of the MDP is denoted by $\mu : \mathcal{S} \rightarrow \mathcal{A}$. Given the MDP specification, the problem is to minimize the cost-to-go function J_μ over the set of admissible policies Π:

$$\min_{\mu \in \Pi} J_\mu(i_0) = \min_{\mu \in \Pi} \mathbb{E}\left[\sum_{k=0}^{\infty} \alpha^k g(i_k, \mu(i_k))\right]. \quad (1)$$

For notational convenience, the cost and state transition functions for a fixed policy μ are defined as

$$g_i^\mu \equiv g(i, \mu(i)) \quad (2)$$
$$P_{ij}^\mu \equiv P_{ij}(\mu(i)), \quad (3)$$

respectively. The cost-to-go for a fixed policy μ satisfies the Bellman equation [8]

$$J_\mu(i) = g_i^\mu + \alpha \sum_{j \in \mathcal{S}} P_{ij}^\mu J_\mu(j) \quad \forall i \in \mathcal{S}, \quad (4)$$

which can also be expressed compactly as $J_\mu = T_\mu J_\mu$ where T_μ is the (fixed-policy) dynamic programming operator.

A. State Space $\mathcal{S}$

The state of each UAV i, $i \in \{1 \dots N\}$, is given by four scalar variables describing the vehicle's abstract location, fuel remaining and health status. The location is described by $l_i \in \{L_b, L_0, \dots, L_{s-1}, L_s, L_x\}$, where L_b represents the base location, $L_0 \dots L_{s-1}$ are intermediate locations, L_s is the surveillance location and L_x is a special state denoting that the vehicle has crashed. It is important to note that this representation does not place any restrictions on the physical separation between locations.

Similarly, the fuel state f_i is described by a discrete set of possible fuel quantities, $f_i \in \{0, \Delta f, 2\Delta f, \dots, F_{max} - \Delta f, F_{max}\}$, where Δf is a suitable discrete fuel quantity.

An agent's health status h_i is described by a discrete set of health attributes, $h_i \in \{H_0, H_1, H_2\}$, where H_0, H_1 and H_2 denote the agent as *functional*, *sensor impaired* and *actuator impaired* respectively.

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

$$\mathbf{x} = (l_1, f_1, h_1;\ l_2, f_2, h_2;\ \dots;\ l_N, f_N, h_N;\ r)^T$$

B. Control Space $\mathcal{A}$

The controls u_i available for the i^{th} UAV are drawn from the set $\{+, 0, -\}$. Actions resulting in an agent moving out of bounds are not available. Additionally, if an agent has crashed, *i.e.* $l_i = L_x$, it can take no action ($u_i = \emptyset$). The full control vector $\mathbf{u}$ concatenates the controls for each UAV: $\mathbf{u} = (u_1, \dots, u_N)^T$

C. State Transition Model P

The transition model P encodes agent dynamics, and is partitioned for each individual UAV. Agent location dynamics are described by the following deterministic rules:

- If action $u_i = +$ is taken, agent i moves one unit closer to the surveillance location L_s
- If action $u_i = -$ is taken, agent i moves one unit closer to the base location L_b
- If action $u_i = 0$ is taken, agent i remains in its current location
- If at any time UAV i's fuel level f_i reaches zero, the UAV transitions to the crashed state ($l_i = L_x$)
- If $l_i = L_x$, agent i has crashed and remains in the crashed state forever afterward

Dynamics of the fuel state f_i are:

- If $l_i = L_b$, then f_i increases at the rate $\dot{F}_{refuel}$
- If $l_i = L_x$, then the fuel state remains the same
- Otherwise, agent i is in a flying state and burns fuel at a stochastic rate: f_i decreases by $\dot{F}_{burn}$ with probability p_f and decreases by $2\dot{F}_{burn}$ with probability $(1 - p_f)$.

Dynamics of the health state h_i are:

- If $l_i = L_b$, then h_i is reset to $h_i = H_0$ (agent i's health is restored, i.e. sensors and actuators are fixed)
- If $l_i = L_x$, then h_i remains the same (agent i has crashed)
- Otherwise: agent i's sensor is degraded with probability p_s and actuator fails with probability p_a

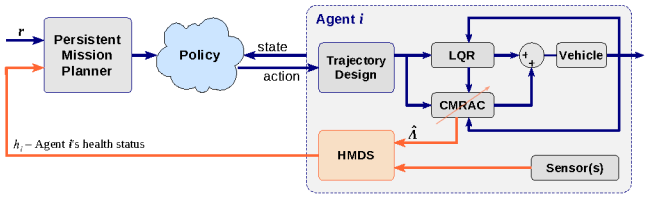

Fig. 2. Parameters $\hat{\Lambda}$ from a composite model-reference adaptive control (CMRAC) scheme are used in the calculation of each agent's health status and factored into future plans by the mission planner.

D. Cost Function g

The cost function $g(\mathbf{x}, \mathbf{u})$ penalizes several undesirable outcomes, including gaps in surveillance coverage (i.e. times when fewer vehicles are in the surveillance area than were requested). Also, a small penalty is incurred for vehicles in the surveillance area with inadequate sensor capability. An additional small cost is associated with each unit of fuel used, which prevents the system from simply launching every UAV on hand - which would certainly result in good surveillance coverage but is inefficient. Also, a small cost is prescribed to vehicles with off-nominal actuator health that are not in a base location. Finally, a high cost is associated with any vehicle crashes. The cost function can be expressed as

$$g(\mathbf{x}, \mathbf{u}) = C_{loc}\delta_n + C_{ds}n_{ds} + C_{da}n_{da} + C_f n_f + +C_c n_c$$

where $\delta_n = \max\{0, (r - n_s(\mathbf{x}))\}$, $n_s(\mathbf{x})$ is the number of UAVs in surveillance area, $n_{ds}(\mathbf{x})$ is the number of UAVs in surveillance area with a degraded sensor, $n_{da}(\mathbf{x})$ is the number of UAVs with degraded actuator health that are not at a base location, $n_f(\mathbf{x})$ is the total number of fuel units burned, $n_c(\mathbf{x})$ is the number of crashed UAVs and C_{loc}, C_{ds}, C_{da}, C_f, and C_c are respectively the relative cost of: a loss of coverage event, a vehicle with a degraded sensor in the surveillance location, a vehicle with a degraded actuator not in a base location, one unit of fuel usage and a vehicle in the crashed state.

E. Approximate Solutions to the MDP

Bellman Residual Elimination (BRE) is a approximate policy iteration technique that is closely related to the class of Bellman residual methods [9]–[11]. Bellman residual methods attempt to perform policy evaluation by minimizing an objective function of the form

$$\left(\sum_{i \in \widetilde{\mathcal{S}}} \left| \widetilde{J}_\mu(i) - T_\mu \widetilde{J}_\mu(i) \right|^2 \right)^{1/2} \tag{5}$$

where $\widetilde{J}_\mu$ is an approximation to the true cost function J_μ and $\widetilde{\mathcal{S}} \subset \mathcal{S}$ is a set of representative sample states. BRE uses a flexible kernel-based cost approximation architecture to construct $\widetilde{J}_\mu$ such that the objective function given by Eq. (5) is identically zero. Details outlining the implementation of the BRE algorithm can be found in Ref [3].

III. Health Monitoring and Diagnostic System

An onboard health monitoring system collects local data and provides a *measure* of agent health. The contents of this measure can vary greatly, depending on the application domain. In this research, the onboard health monitor collects observations regarding sensor and actuator performance and updates internal models accordingly. To support the internal actuator model, the HMDS collects updated parameters indicating the deviation of actual from nominal efficiency, for each actuator. These parameters are updated as a byproduct of running a composite adaptive controller which provides robustness to actuator degradations. These parameters provide key insight into the capability of the mobile agent, and its ability to perform its advertised range of tasks.

Figure 2 shows the connection between the parameters of the adaptive control scheme and actuator health. The key concept being that the adaptation used to keep an agent's ailing vehicle aloft can also give good insights into its now-reduced expected performance. In other words, controller adaptation could be a first line of defense against vehicle component degradation as variations in the adaptive control gains effectively act as vehicle health indicators. Though not directly integrated within the HMDS, the composite adaptive controller is the source of vital information into the HDMS. Therefore, the following sections outline the theory and development of the composite model-reference adaptive controller (CMRAC) for a quadrotor helicopter.

A. Composite Adaptive Controller Design

As recently shown in Ref [6], the linearized quadrotor dynamics can be written as:

$$\dot{x}_p = A_p x_p + B_p \Lambda (u + \theta^T \omega), \tag{6}$$

where $B_p \in \Re^{nxm}$ is constant and *known*, $A_p \in \Re^{nxn}$ is constant and *unknown*, $x_p \in \Re^n_p$, $u \in \Re^m$, $\Lambda \in \Re^{mxm}$ is an unknown diagonal positive definite constant matrix with diagonal elements $\Re \in (0, 1]$, $\theta \in \Re^{nxm}$ is an unknown matched parameter uncertainty, and $\omega = \begin{bmatrix} x_t^T & r^T & 1 \end{bmatrix}^T$ is the regressor vector, where x_t is the extended system state, the dynamics of which are described in Equation (7). The goal is to track a reference command r in the presence of the unknown A_p, Λ, and θ.

We define the system output as $y_p = C_p x_p$, where C_p is the *known* measurement matrix. In our case, let y_p be a vector of measured states consisting of the quadrotor position in cartesian coordinates and heading with respect to the inertial frame. The output tracking error is then given by $e_y = y_p - r$ and augmenting (6) with the integrated output tracking error yields $\dot{e}_{y_I} = e_y$, which leads to the extended open loop dynamics, described by

$$\dot{x}_t = A_t x_t + B_t \Lambda (u + \theta^T \omega) + B_c r. \tag{7}$$

where $x_t = \begin{bmatrix} x_p^T & e_{y_I}^T \end{bmatrix}^T$ is the extended system state vector. The extended open-loop system matrices are given by

$$A_t = \begin{bmatrix} A_p & 0 \\ C_p & 0 \end{bmatrix}, \; B_t = \begin{bmatrix} B_p \\ 0 \end{bmatrix}, \; B_c = \begin{bmatrix} 0 \\ -I \end{bmatrix}, \tag{8}$$

and the extended system output becomes

$$y_t = \begin{bmatrix} C_p & 0 \end{bmatrix} x_t = C_t x_t. \tag{9}$$

1) Reference Model: A nominal controller of the form

$$u_{nom} = K_x x_t, \tag{10}$$

can be designed for the system in Equation (7) in the case where there is no uncertainty, (i.e. $\Lambda = I^{nxn}$). The feedback gains K_x can be selected using LQR or classical design techniques. The reference model used by MRAC is the closed loop system given in Equation (7) with no uncertainty and with the control input given in Equation (10), as shown by

$$\dot{x}_m = A_t x_m + B_t u_{nom} + B_c r = A_m x_m + B_c r. \tag{11}$$

2) Direct Adaptive Controller: Adding to the baseline controller, an adaptive control input is

$$u_{ad} = \hat{K}_x^T x_t + \hat{\theta}_r^T r + \hat{\theta}_d = \hat{\theta}^T \omega, \tag{12}$$

where $\hat{\theta}^T = \begin{bmatrix} \hat{K}_x^T & \hat{\theta}_r^T & \hat{\theta}_d^T \end{bmatrix}$ are time-varying adaptive parameters that will be adjusted in the adaptive law given in Equation (14) below. The overall control input is thus

$$u = u_{ad} + u_{nom} = \hat{\theta}^T \omega + K_x x_t + r. \tag{13}$$

The canonical adaptive law is given by

$$\dot{\hat{\theta}} = -\Gamma \omega e^T P B_t, \tag{14}$$

where Γ is a diagonal positive definite matrix of adaptive gains, $e = x_t - x_m$ is the model tracking error, and P is the unique symmetric positive definite solution of the Lyapunov equation $A_m^T P + P A_m = -Q$, where Q is also symmetric positive definite. This adaptive controller is based on nonlinear stability theory [12]–[17]. The augmented structure of the adaptive controller implies that in the nominal case (i.e. no parameter uncertainty) the overall system is equivalent to the baseline controller. However, when failures or other uncertainties arise, the adaptive controller works to eliminate reference-tracking errors and thereby assists the baseline controller in maintaining both stability and performance.

3) Combined / Composite Adaptive Controller: The so-called *CMRAC conjecture* [18], states that better transient characteristics can be obtained by using prediction errors in addition to tracking errors in the design of adaptive controllers. Thus a Combined (or Composite) MRAC structure was developed by combining aspects of direct and indirect adaptive control. This conjecture has been supported by numerous simulations confirming that indeed CMRAC systems had transient performance better than that of direct MRAC alone [19]–[22]. The CMRAC conjecture, however, remains unproven.

The particular version of CMRAC used in these studies differs from some previous formulations [21,23,24] in that: it is applicable to a generic class of MIMO dynamic systems; online measurements of the system state derivative are not required; and the system is designed to augment a baseline linear controller [25]. To generate the prediction error signal required for indirect adaptation, we first rewrite the dynamics of (7) as

$$\begin{aligned} \dot{x_t} + \lambda_f x_t = \lambda_f x_t + A_m x + B_c r \\ + B\Lambda \left(u + \theta^T \omega \right), \end{aligned} \tag{15}$$

and introduce a stable filter

$$G(s) = \frac{\lambda_f}{s + \lambda_f}, \tag{16}$$

where $\lambda_f > 0$ is the filter inverse constant. The filtered version of x_t is denoted x_{t_f} and is described by the dynamics

$$\dot{x}_{t_f} + \lambda_f x_{t_f} = \lambda_f x_t. \tag{17}$$

Substituting Equation (17) into Equation (15) and letting $z = x_t - x_{t_f}$ we have

$$\dot{z} + \lambda_f z = A_m x_t + B_c r + B\Lambda \left(u + \theta^T \omega \right), \tag{18}$$

and, consequently

$$\begin{aligned} z = z(t_0) e^{-\lambda_f t_0} + \int_{t_0}^{t} e^{-\lambda_f (t-\eta)} \left[A_m x_t(\eta) \right. \\ \left. + B_c r(\eta) + B\Lambda \left(u(\eta) + \theta^T \omega(\eta) \right) \right] d\eta. \end{aligned} \tag{19}$$

Without loss of generality, we can assume that the filter dynamics of Equation (16) and the plant dynamics of Equation (18) have the same initial conditions, that is, $z(t_0) = 0$. Noting that the integral in Equation (19) represents application of the filter in Equation (16), we can rewrite Equation (19) as

$$x_t - x_{t_f} = A_m \frac{x_{t_f}}{\lambda_f} - B_c \frac{r_f}{\lambda_f} + \Lambda \frac{u_f}{\lambda_f} + \Lambda \theta^T \frac{\omega_f}{\lambda_f}, \tag{20}$$

where r_f, u_f, and ω_f are filtered versions of r, u, and ω, respectively. Assuming that B is full rank, we can rearrange Equation (20) as

$$\begin{aligned} \left(B^T B \right)^{-1} B^T \left(\lambda_f (x_t - x_{t_f}) - A_m x_{t_f} - B_c r_f \right) \\ = \Lambda u_f + \Lambda \theta^T \omega_f. \end{aligned} \tag{21}$$

We now denote the left-hand side of Equation (21) as $Y \in \Re^m$ and note that the value of Y can be computed online at every time instant t using the state x_t, the filtered state x_{t_f} and the command input r. The right-hand side of Equation (21) contains the unknown parameters Λ and θ, which can be estimated using the predictor model

$$\hat{Y} = \hat{\Lambda} \left(u_f + \hat{\theta^T} \omega_f \right). \tag{22}$$

The predictor error $e_Y = \hat{Y} - Y$ can thus be written as

$$e_Y = \Lambda \tilde{\theta} \omega_f + \tilde{\Lambda} \left(u_f + \hat{\theta}^T \omega_f \right). \tag{23}$$

Including the indirect adaptation is accomplished by simply adding indirect adaptive terms to the direct adaptive laws given in Equation (14), yielding

$$\begin{aligned} \dot{\hat{\Lambda}}^T = -\Gamma_\Lambda \left(u + \hat{\theta}^T \omega_f \right) e_Y^T, \\ \dot{\hat{\theta}} = \Gamma_\theta \left(\omega e^T P B - \omega_f \gamma_c e_Y^T \right). \end{aligned} \tag{24}$$

It is clear that in the case where $\gamma_c = 0$, the above adaptive

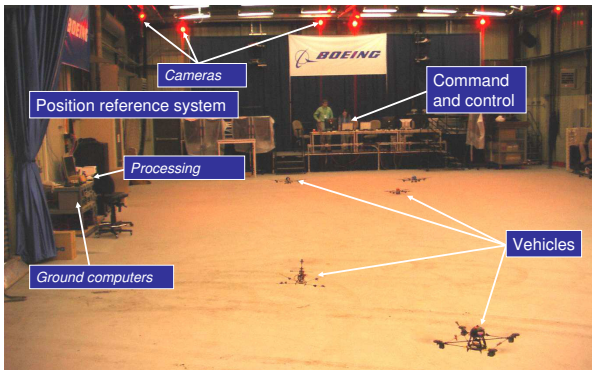

Fig. 3. The Boeing Vehicle Swarm Technology Laboratory (VSTL), a state-of-the-art rapid prototyping indoor flight testing facility [7].

control laws reduce to those of (14). Stability can be shown using the Lyapunov function candidate [25]

$$V = e^T P e + \text{Tr}\left(\tilde{\theta}^T \Gamma_\theta^{-1} \tilde{\theta} \Lambda\right) + \text{Tr}\left(\tilde{\Lambda} \Gamma_\Lambda^{-1} \tilde{\Lambda}^T\right) \quad (25)$$

with derivative

$$\dot{V} = -e^T Q e - 2\gamma_c e_Y^T e_Y \leq 0. \quad (26)$$

The system is therefore globally asymptotically stable by Barbalat's lemma and the tracking error and prediction error asymptotically converge to 0.

It is clear from Equation (24) that, in addition to the parameter estimates $\hat{\theta}$, the CMRAC approach also explicitly generates estimates of $\hat{\Lambda}$. The diagonal elements of $\hat{\Lambda}$ can be viewed as estimates of the current health of the vehicle's actuators, specifically indicating the deviation from nominal efficiency, for each actuator. These are the $\hat{\Lambda}$ parameters passed to the onboard HMDS, as shown in Figure 2.

IV. Flight Results

A. Experimental Setup

Boeing Research and Technology has developed the Vehicle Swarm Technology Laboratory (VSTL), an environment shown in Figure 3 for testing a variety of vehicles in an indoor, controlled environment [7]. VSTL is capable of simultaneously supporting a large number of both air and ground vehicles, thus providing a significant advantage over traditional flight test methods in terms of flight hours logged.

The persistent surveillance mission described in Section II was implemented in the Boeing VSTL with static *search* and dynamic *track* elements. A heterogeneous team of six agents (4 UAVs and 2 UGVs) began at a base location and were tasked to persistently search the surveillance area. As threats are discovered via search, additional agents were called out to provide persistent tracking of the dynamic threat(s).

B. Results

This section presents representative results of flight tests of the multi-agent persistent mission planner in Boeing's Vehicle Swarm Technology Laboratory [7]. Figure 4 shows the team of six agents persistently searching a surveillance region and tracking discovered threats. The figure is divided into two axes. On the top, each agent's location is represented by a colored line as well as with a unique marker shape. On the bottom axis, the health status of UAV_1 is shown in the form of the $\hat{\Lambda}$ value corresponding to the degraded actuator and the output of the onboard HMDS.

Following the blue line (with blue dots) in the top axis of Figure 4, it is seen that UAV_1 experiences an actuator failure shortly after reaching the surveillance location. The failure is externally induced and, as shown on the bottom axis, CMRAC immediately detects a change in performance and the $\hat{\Lambda}$ parameters react accordingly to maintain stability of the vehicle. Also shown on the lower axis, the HMDS uses these controller parameters to quickly estimate the vehicle's health status and inform the planner of UAV_1's degraded health state. After returning to base for repairs, UAV_1 again enters the MDP-governed rotation of vehicles into the surveillance region. This failure-stabilize-estimate-repair cycle is repeated several times to show persistency.

In summary, the results given in Figure 4 show well the significance of successfully integrating low-level adaptation parameters with the mission-level planning system. This is evidenced by fast reaction times at the planner level to failures at the agent level, which are enabled by the collection and proper conditioning of observations at the agent level by the HMDS so concise, relevant information is passed up to the mission level.

V. Conclusions

This paper presented an extension of previous work on the persistent mission problem where real-time changes in agent health were included in the problem formulation. Both sensor and actuator health were included in the metric for agent capability. Actuator health was monitored as a result of augmenting the baseline agent controller with a composite model-reference adaptive controller, which was designed to provide stability and robustness against actuator degradations. Parameters internal to the composite adaptive controller provided valuable insight into the physical capabilities of the agent and were used in conjunction with sensor health information to form a more complete measure of overall agent capability. The combined metric was used online and in forward planning to enable both reactive and proactive behaviors against sensor and actuator degradations. The persistent mission planner was formulated as a Markov decision process and a distributed, approximate solution method was implemented due to the size of the full problem. Flight test results with induced actuator degradations were presented which demonstrated the robustness of the composite adaptive controller and its successful integration with the agent-level health-monitoring and mission-level planning systems. Also demonstrated were the reactive and proactive qualities of the MDP-based planning system to continuously fulfill mission obligations while re-tasking agents under actuator and sensor health degradations.

Acknowledgments

This Research was generously supported by Boeing Research & Technology, Seattle and, in part by AFOSR grant FA9550-09-1-

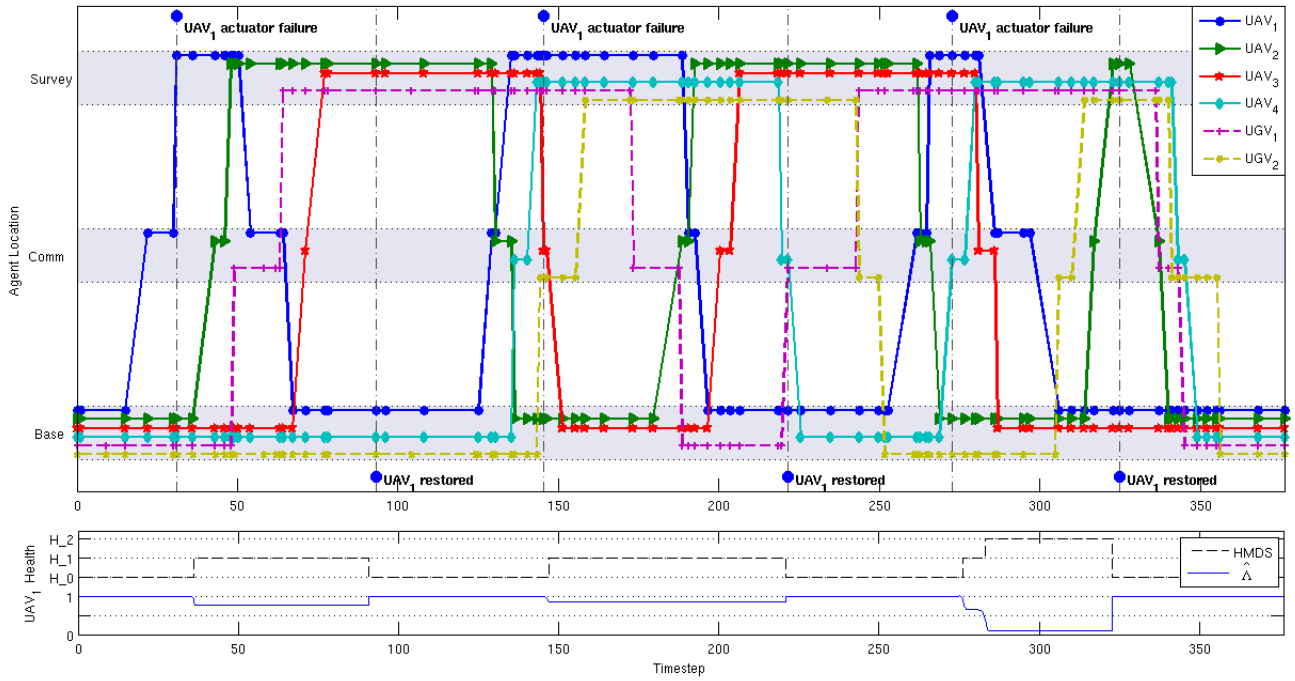

Fig. 4. Results of a flight experiment in the Boeing Vehicle Swarm Technology Lab (VSTL) with six unmanned agents persistently searching a surveillance location and tracking any threats detected there. On the top axis, each agent's location is represented by a colored line as well as with a unique marker shape. On the bottom axis, the health status of UAV_1 is shown by the $\hat{\Lambda}$ value corresponding to the degraded actuator as well as the output of the onboard HMDS, which switches between H_0, H_1, and H_2 accordingly.

0522. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research or the U.S. Government. The authors would also like to acknowledge Dr. Brett Bethke, whose work this research extends.

REFERENCES

[1] M. Valenti, B. Bethke, J. P. How, D. P. de Farias, and J. Vian, "Embedding health management into mission tasking for uav teams," in *American Control Conference (ACC)*, 9-13 July 2007, pp. 5777–5783. [Online]. Available: 10.1109/ACC.2007.4282719

[2] M. Valenti, D. Dale, J. How, and J. Vian, "Mission health management for 24/7 persistent surveillance operations," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Myrtle Beach, SC, August 2007.

[3] B. Bethke, J. P. How, and J. Vian, "Multi-UAV Persistent Surveillance With Communication Constraints and Health Management," in *Guidance Navigation and Control Conference*, August 2009 (AIAA-2009-5654).

[4] B. Bethke, J. How, and J. Vian, "Group health management of UAV teams with applications to persistent surveillance," in *American Control Conference (ACC)*, 2008.

[5] B. Bethke and J. How and A. Ozdaglar, "Approximate Dynamic Programming Using Support Vector Regression," in *Conference on Decision and Control (to appear)*, Cancun, Mexico, 2008.

[6] Z. Dydek, A. Annaswamy, and E. Lavretsky, "Adaptive Control and the NASA X-15-3 Flight Revisited," *Control Systems Magazine, IEEE*, vol. 30, no. 3, pp. 32 –48, Jun. 2010.

[7] E. Saad, J. Vian, G. Clark, and S. Bieniawski, "Vehicle swarm rapid prototyping testbed," in *AIAA Infotech@Aerospace*, Seattle, WA, 2009.

[8] D. Bertsekas, *Dynamic Programming and Optimal Control*. Belmont, MA: Athena Scientific, 2007.

[9] P. Schweitzer and A. Seidman, "Generalized polynomial approximation in Markovian decision processes," *Journal of mathematical analysis and applications*, vol. 110, pp. 568–582, 1985.

[10] L. C. Baird, "Residual algorithms: Reinforcement learning with function approximation." in *ICML*, 1995, pp. 30–37.

[11] R. Munos and C. Szepesvári, "Finite-time bounds for fitted value iteration," *Journal of Machine Learning Research*, vol. 1, pp. 815–857, 2008.

[12] G. Kreisselmeier and K. S. Narendra, "Stable model reference adaptive control in the presence of bounded disturbances," *IEEE Transactions on Automatic Control*, vol. 27, no. 6, pp. 1169–1175, December 1982.

[13] K. S. Narendra and A. M. Annaswamy, *Stable Adaptive Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[14] P. A. Ioannou and J. Sun, *Robust Adaptive Control*. Upper Saddle River, NJ: Prentice-Hall, 1996.

[15] S. Sastry and M. Bodson, *Adaptive control: stability, convergence, and robustness*. Prentice-Hall, Englewood Cliffs, NJ, 1989.

[16] J. J. E. Slotine and J. A. Coetsee, "Adaptive sliding controller synthesis for nonlinear systems," *International Journal of Control*, vol. 43, no. 6, pp. 1631–1651, 1986.

[17] M. Krstic, P. Kokotovic, and I. Kanellakopoulos, *Nonlinear and adaptive control design*. John Wiley & Sons, New York, NY, USA, 1995.

[18] M. Duarte and K. Narendra, "Error models with parameter constraints," *International Journal of Control*, vol. 64, no. 6, pp. 1089–1111, 1996.

[19] J. Slotine and W. Li, "Composite adaptive control of robot manipulators." *Automatica*, vol. 25, no. 4, pp. 509–519, 1989.

[20] M. Duarte-Mermoud, J. Rioseco, and R. Gonzalez, "Control of longitudinal movement of a plane using combined model reference adaptive control," *Aircraft Engineering and Aerospace Technology: An International Journal*, vol. 77, no. 3, pp. 199–213, 2005.

[21] M. Duarte and K. Narendra, "Combined direct and indirect approach to adaptive control," *IEEE Transactions on Automatic Control*, vol. 34, no. 10, pp. 1071–1075, 1989.

[22] E. Lavretsky, "Combined / Composite Model Reference Adaptive Control," in *AIAA Guidance, Navigation, and Control Conference. GNC'09*, 2009.

[23] J. Slotine, W. Li, *et al.*, *Applied nonlinear control*. Prentice-Hall Englewood Cliffs, NJ, 1991.

[24] J. Nakanishi, J. Farrell, and S. Schaal, "Composite adaptive control with locally weighted statistical learning," *Neural Networks*, vol. 18, no. 1, pp. 71–90, 2005.

[25] E. Lavretsky, "Combined / Composite Model Reference Adaptive Control," *IEEE Transactions on Automatic Control*, vol. 54, no. 11, pp. 2692–2697, 2009.