Coordination and Control Experiments on a Multi-vehicle Testbed

Ellis King, Yoshi Kuwata, Mehdi Alighanbari, Luca Bertuccelli, and Jonathan How
Aerospace Controls Laboratory,
Massachusetts Institute of Technology
{etking, kuwata, mehdi_a, lucab, jhow}@mit.edu

Abstract—This paper introduces two unique testbeds that have recently been developed to demonstrate the cooperative control of teams of UAVs. The first testbed uses eight rovers and four blimps operated indoors to emulate a team of heterogeneous vehicles performing a combined reconnaissance and strike mission. The second testbed uses eight small aircraft that are flown autonomously using a commercially available autopilot. This combination of testbeds provides platforms for both advanced research and realistic demonstrations. Numerous trajectory optimization and team coordination algorithms have recently been developed to execute these UAV missions. This paper highlights several of these coordination and control algorithms and presents typical results for representative experiments. These demonstrations of the high-level planning algorithms on scaled vehicles operating in uncertain and dynamic environments represent key steps towards transitioning them to future UAV missions.

I. INTRODUCTION

UAVs offer advantages over conventional manned vehicles in many applications because they can be used in situations otherwise too dangerous for manned vehicles (e.g., eliminating anti-aircraft defenses) and without being weighed down by the systems required by a pilot, UAVs are capable of staying aloft on longer surveillance missions. While the roles and capabilities of UAVs are growing, current UAV control structures were conceived with limited roles in mind for these vehicles. Thus it is necessary to improve on this control structure in order to fully exploit the expanding capabilities of UAVs. This includes developing techniques to optimize the coordination of a fleet of UAVs, which is comprised of the coupled subproblems of determining sub-team composition, allocating resources, and optimizing vehicle trajectories [1]. These are all computationally intensive optimization problems that require good situational awareness to achieve coordinated and cooperative behaviors. Numerous algorithms have recently been developed to achieve this cooperative behavior [2], [3], but a key step towards transitioning these high-level algorithms to future missions is to successfully demonstrate that they can handle the implementation challenges using scaled vehicles operating in realistic environments. Thus two testbeds have been developed at MIT to perform these demonstrations. One uses multiple rovers and blimps operated indoors to emulate a heterogeneous fleet of vehicles performing a search and rescue mission. The second uses eight UAVs that are flown autonomously using a commercially available

autopilot from Cloud Cap Technology. Performing experiments on these testbeds will highlight the fundamental challenges associated with: (i) planning for a large team in real-time; (ii) developing controllers that are robust to uncertainty in situational awareness, and are sufficiently flexible to respond to dynamic changes; and (iii) using communication networks and distributed processing to develop integrated and cooperative plans.

II. COORDINATION ALGORITHMS

Mixed-integer Linear Programming (MILP) has previously been shown to provide a natural framework for posing coordination problems, and approximations such as the *decomposition approach* have proven to provide accurate yet tractable solutions to the overall problem [3], [4], [5]. The decomposition approach simplifies the coupling between the assignment and trajectory design problems by calculating and communicating only the key information that connects them. This is achieved using an approximate cost-to-go calculation to obtain good estimates of the costs associated with feasible paths around "obstacles" (e.g. buildings, no-fly-zones) in the environment. These costs are then used in the assignment problem solved using the *petal* algorithm [5], [6]. Uncertainty in the target classification due to poor or conflicting information enters the problem as uncertainty in the assignment costs. As demonstrated in [7], the assignment process must be robust to these types of uncertainty, and it is also vital to ensure that the reconnaissance and strike tasks are allocated simultaneously to provide the most benefit to the strike part of the missions.

Task Assignment – While the decomposition approach greatly reduces complexity, the problem of task assignment with precedence constraints has been shown to be NP-Hard, and achieving the exact solution for a large team of UAVs is computationally intensive and not suitable for real-time applications. The petal algorithm uses a heuristic method to prune out the solutions that are not likely to be part of the optimal solution, which significantly speeds up the task assignment process. While this approach has been shown to work well for small problems, it is still difficult to run in real-time for problems with larger numbers of vehicles and tasks. To perform reassignment in real-time as required in a dynamic, frequently changing environment, we have
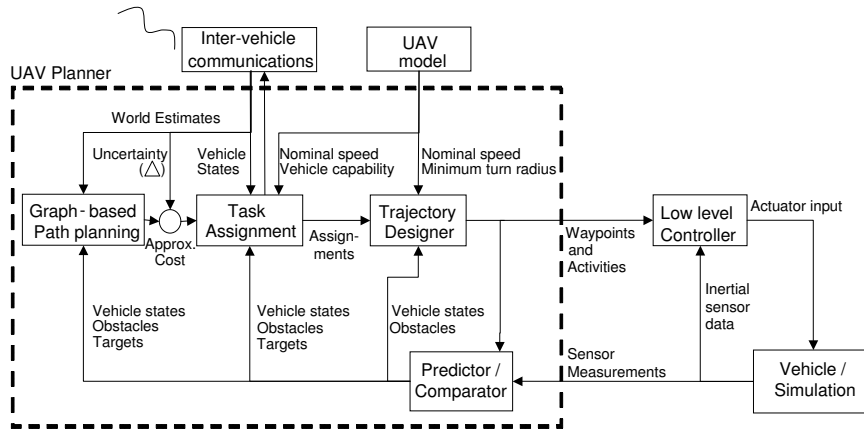
Fig. 1. System algorithm architecture for rover, blimp and UAV testbeds.

extended the petal method to develop a receding horizon task assignment (RHTA) algorithm.

RHTA significantly reduces the computation time by selecting at most m (typically less than 3) tasks for each UAV, then repeating the optimization over several iterations[8]. The process selects tasks to the mission list for each UAV, updates the UAV's position and time, removes the assigned tasks from the task list, and repeats until all the tasks are assigned. Timing and precedence constraints can be imposed using the approach in [5] or by simply requiring that tasks be removed from the list until all precedents have been assigned. The first of these two approaches is more complicated, but should yield a less conservative approach than the second.

Trajectory Optimization – The final step is to compute detailed UAV trajectories around the obstacles, which can be solved using a MILP-based receding horizon planner [9]. This approach has been proven to guarantee the arrival at the target in bounded time, as RH-MILP uses a simple vehicle dynamics model in the near term and an approximate path in the long term. This combination gives a good estimate of the cost-to-go and greatly reduces the computational effort required to design the complete trajectory. Discrepancies in the assumptions made in the two models are handled by ensuring that the planning horizon is sufficiently long [10]. Novel pruning and graph search algorithms have recently been integrated with RH-MILP, and these also have the effect of significantly reducing the computational load. In the following experimental sections, the trajectories shown are designed in real-time using this approach.

Control Architecture – Figure 1 shows the control architecture used for the UAVs, but the setup is very similar for the rovers and blimps [5], [3]. Low-level control and the basic estimation tasks are run onboard, and the planning for the vehicles is done off-board. The *planner* outputs dynamically feasible waypoint lists and actions (i.e. classify, strike, assess) to the vehicles, and monitors the uncertain states of the vehicles and the world map. When significant changes to the situational awareness are detected, the cost map is then updated, the tasks are re-assigned and/or the
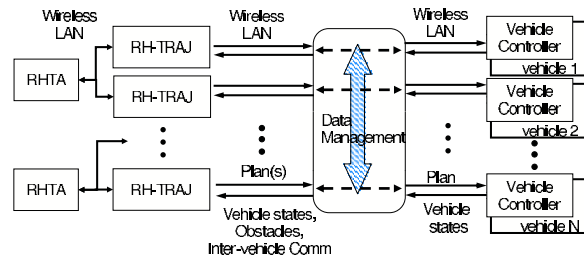

Fig. 2. Control architecture: Decentralized path planning/Distributed task assignment.
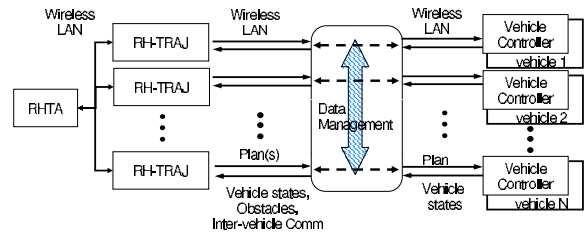

Fig. 3. Control architecture: Decentralized path planning/Centralized task assignment.

trajectories are redesigned.

Note that the system infrastructure was set up to emulate a fully integrated fleet of UAVs – all data passes through a central hub that performs data management between the planning computers and vehicles, effectively simulating communication delays, vehicle sensors and uncertainty in the environment. Using this setup greatly simplifies the testbed, while maintaining nearly all of the functionality of a fully integrated system. For example, as shown in Figures 2 and 3, we can use our testbed to investigate the impact of communication networking issues on the coordination problem by imposing various limitations/constraints on how the planning laptops communicate (using their own wireless or Ethernet links). Future work will demonstrate the effectiveness of various control architectures on the task assignment process, as would be seen in utilizing dynamic sub-teams of various compositions.

The following sections describe the two multi-vehicle testbeds that have been developed to investigate the performance of these coordination and control algorithms.

Fig. 4.   4 of 8 ActivMedia P3-AT Rovers.

Fig. 5.   1 of 4 Blimps.

III. Hardware Testbeds

Rover/Blimp Testbed – The first testbed uses multiple rovers and blimps operated indoors to emulate a heterogeneous fleet of vehicles that could be used to perform Suppression of Enemy Air Defense (SEAD) type missions. The rovers in Figure 4 are ActivMedia's P3-ATs, which are operated with minimum speed and turn rate constraints to emulate the motion of an aircraft. A Sony VAIO mounted on the rover processes sensor data and performs the low-level control, while all high-level planning is done off-board using 2.4 GHz Dell laptops running MATLAB, AMPL, and CPLEX. A direct wireless Ethernet connection provides a fast and reliable network between the laptops, so this is essentially the same as having both laptops onboard. The ArcSecond Constellation 3D-i is used to measure the vehicle position indoors. This sensor uses laser metrology to provide ±4mm position accuracy at 20Hz. The 7ft diameter blimps in Figure 5 were scaled to carry the VAIO and have an identical control architecture. The blimps were designed to perform reconnaissance and classification tasks in conjunction with the rovers that act as strike vehicles. The blimps can also be used to map an uncertain environment for the rovers.

Figures 6–8 show the result for a scenario with 4 rovers and 14 tasks, which introduces enough complexity in the mission that a global optimization would be very difficult for dynamic real-time reassignment. However, RHTA with $m = 2$ is shown to provide good performance despite the complexity of the mission scenario. The mission begins with 4 rovers and initial assignments, but after several seconds, rover 4 is lost. Figure 6 shows the effect of reassignment when the tasks of rover 4 have been distributed among the other rovers. Later in the mission, reassignment must again be performed when tasks 9 and 11 are determined to be located at different positions than previously assumed. Figure 7 shows the final assignments when two new tasks (15 and 16) are discovered and rover 3 is assigned to visit these locations.

Figure 9 shows an experimental result on the rover/blimp heterogenous testbed. The scenario models the environment where the information is partially available, requiring search missions be executed concurrently with the strike. The obstacles impede the rovers, but the blimp is able to fly over them. In this experiment, the blimp is used to search for tasks while the rover executes them. The trajectory on the left shows the path of the blimp performing its search pattern, and the trajectory on the right shows the path of the rover as it navigates to each task. For the purpose of this test, the heading command is sent to the blimp, while the rover follows the waypoints (marked with ◯) generated by the RHTA. Initially, the waypoint C was not known, but the blimp is sent to perform a reconnaissance of the open space to the left, flying over the obstacle in the bottom of the figure. While the rover is en route to execute task B, the blimp discovers the new task (waypoint C) and RHTA reassigns that task to the rover. This initial result demonstrates the successful integration of the heterogenous vehicles in our planning system and future tests will incorporate robustness into the task assignment for a heterogeneous system of rovers and blimps in an uncertain environment.

UAV Testbed – The second testbed is a fleet of 8 UAVs (Figure 10) that are flown autonomously using the Cloud Cap commercial autopilot interfaced directly with the planning and task assignment algorithms. Figure 11 shows the 7.5oz Piccolo autopilot from Cloud Cap Technologies installed in the aircraft. Small aircraft (60-sized trainers) were purposefully chosen to reduce operational complexity while still providing a high degree of flexibility in the missions that can be performed. The large trainer wing and Saito-91 four-stroke engine allow an additional two pounds of payload for sensor or communications upgrades. Twenty minute flights are easily achievable in the current configuration, and further extensions are possible.

The UAV testbed has been operated autonomously on numerous occasions – Figure 13 shows the results of an 22 minute autonomous flight involving two UAVs simultaneously flying the same flight plan. Both vehicles tracked the waypoints in the presence of wind and open loop formation flight was achieved by adjusting the commanded speed until the vehicles were in phase with one another. A 50 meter altitude offset was applied to one of the vehicle trajectories
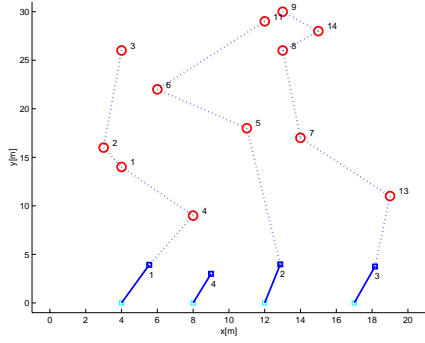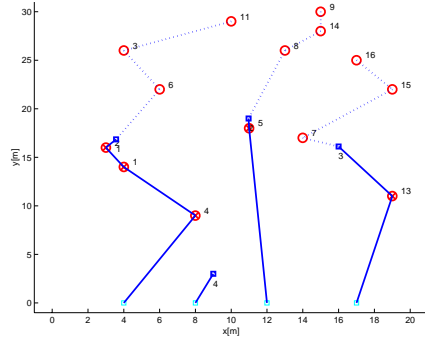
Fig. 6. Re-assignment plan after rover 4 lost.

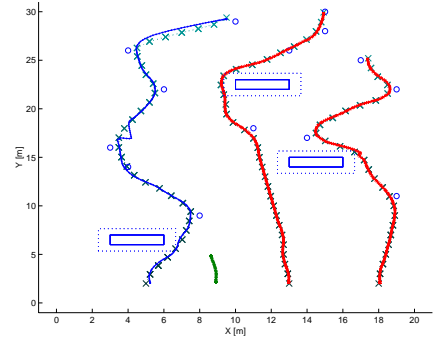Fig. 7. Assignment after wpts 15 & 16 found.
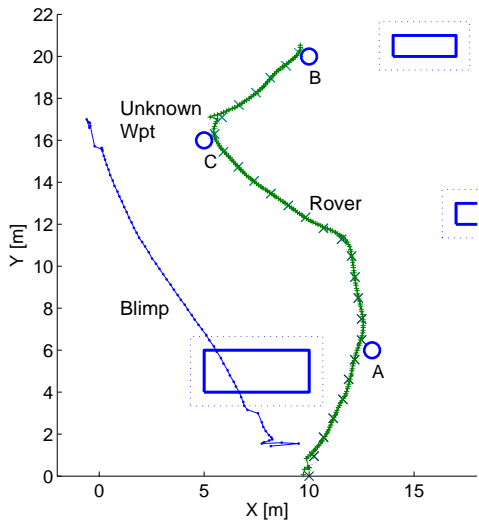

Fig. 8. Experimental rover trajectories.

Fig. 9. Rover and Blimp trajectories recorded during a heterogeneous recon and strike demonstration.

Fig. 10. 6 of the 8 UAVs in the fleet.

Fig. 11. Piccolo autopilot from Cloud Cap Tech.

in Figure 13 to allow for easier viewing.

A wireless video system has been integrated with the UAV testbed to produce high quality images from the airborne vehicles – Figure 12 shows a typical aerial shot from one of the UAVs. This system is used to verify the position of the vehicles and provide user feedback for high level decision making.

Figures 14–15 show results from a mission flown on the UAV testbed using receding horizon control to generate waypoint plans in real time. In this scenario, the goal locations used in the planner (shown as circles) were set in a 400m × 600m box pattern over the extents of the flying field and *timing constraints* [2] were enforced to ensure a clockwise sequencing. Figures 14–15 show the progression of the optimal planned paths and the telemetry data from the vehicle during one circuit. Each "×" segment represents one waypoint plan which is returned from the optimal trajectory designer and is uploaded piecewise to the UAV as the MILP optimization completes. The plans are spaced at 100m or approximately 4 seconds from the next in the sequence. With 2.4GHz Dell laptops running CPLEXv8.0, these computations typically take less than 1 second providing substantial margin in completion times.
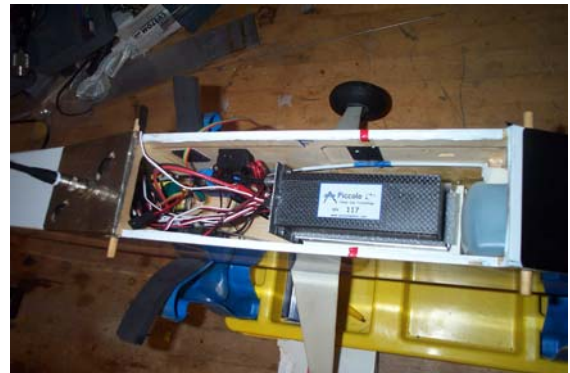
The gray regions in Figures 14–15 represent obstacle locations which were encoded into the scenario using mixed integer constraints [9] to constrain the planned trajectories within a safe operating distance. Although no dynamic changes were made to the environment in this scenario (i.e., pop-up obstacle, goal discovery), the framework of the RH controller also allows for this capability.

This flight test was conducted in the presence of winds approximately 25% of the vehicle airspeed (The PT-60 aircraft are sized for 25m/s nominal airspeed) and in a confined area, requiring minimum turn radius constraints to be enforced. As a result of these conditions, roughly 40m

Fig. 12. Overhead of the local flying field at Crow Island taken with the onboard camera.
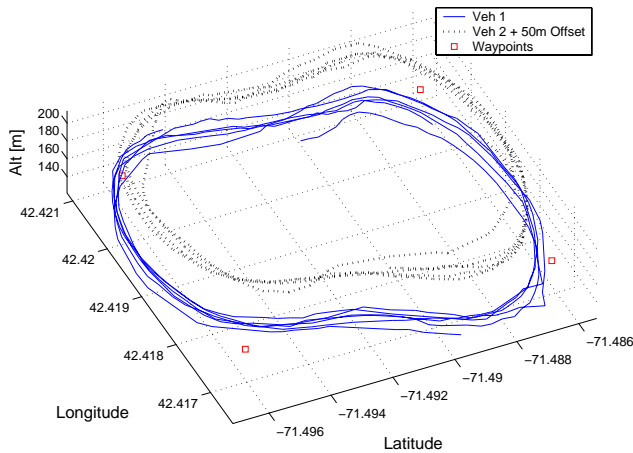

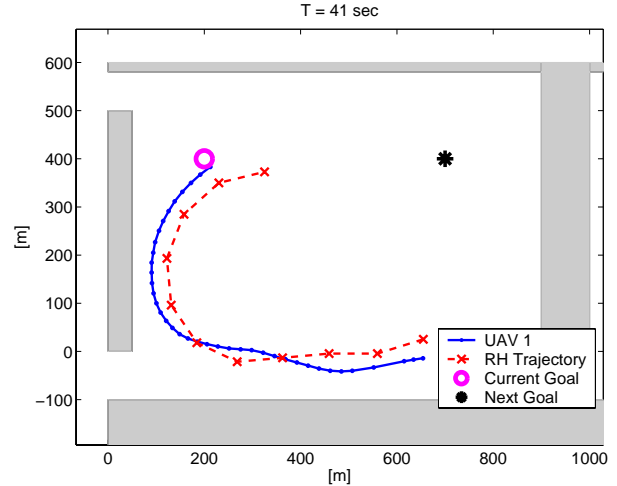Fig. 13. Data from a simultaneous flight of 2 autonomous UAVs. Data offset vertically for clarity.

Fig. 14. Initial demonstration of UAV receding horizon control. Planned paths are dashed lines, vehicle telemetry are solid (start at 41 secs).
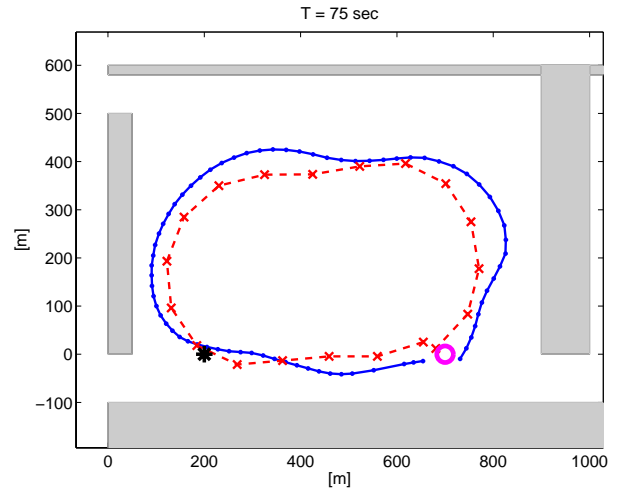

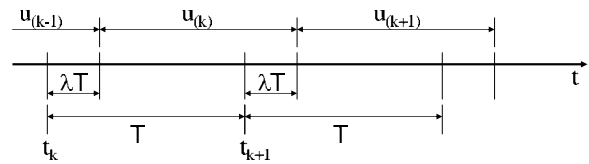Fig. 15. Completion of a circuit. Wind disturbances ∼ 5m/s (W to E).

Fig. 16. Timing for receding horizon control scheme.

overshoot offsets can be seen in the flown trajectories in some instances. Wind disturbances act as a large contributor to the position offsets seen in Figure 15, as such current research investigates the incorporation of along and cross track error feedback into the receding horizon formulation for improved performance. In addition, along track speed control is implemented to ensure meeting local timing constraints and coordination among members of the fleet.

The preceding results demonstrate the capabilities of the system to perform online MILP trajectory optimization and task assignments, but these preliminary results also highlight the need for added compensation for disturbance rejection. Uncertain wind disturbances consisting of both static and turbulent components act on the system and cause the ±40m cross-track offsets seen in the execution of the plan. An estimate of the static wind vector is available from the Cloud Cap avionics, and this can be added to the dynamics model used in the trajectory design. However, the variations of the wind vector and the uncertainty in the wind estimate can still result in significant variations in the actual trajectory flown. To correct for these flight errors, the planner has recently been modified to use current vehicle states in the design of the next trajectory segment.

This results in better feedback of the cross-track errors, which improves the response to disturbances. However, this introduces communication and computation delays, which could lead to an unstable response.

Typical computation times for the trajectory optimizations are approximately 0.5 seconds per iteration, and there is some small communication delay in transmitting the waypoint plans to the vehicle. To avoid planning with old information, we can include an approximate model of the closed-loop vehicle dynamics to propagate the measured vehicle state forward in time [11]. Figure 16 depicts the timing of the receding horizon control scheme. A fixed planning interval, T, is chosen corresponding to time scale

lengths that will make the planner suitably reactive to changes in the environment. Measurements of the cross-track position and heading states are made at time t_k, $X(t_k) = [y_{ct}(t_k) \quad \psi(t_k)]^T$, which triggers the trajectory design algorithm. Using knowledge of the previous plan that was implemented, $u_{(k-1)}$, an estimate of the vehicle state at time $t_k + \lambda\mathsf{T}$ $(0 < \lambda \le 1)$ can be made by propagating the closed-loop dynamics:

$$\hat{X}(t_k + \lambda\mathsf{T}) = A_{cl}X(t_k) + G\bar{w}$$

where A_{cl} is the discretized closed-loop dynamics model, G is the disturbance input matrix for $\bar{w}$, the mean static wind vector. $\hat{X}(t_k + \lambda\mathsf{T})$ is then used as the initial conditions for the start of the plan $u(k)$. The simulations and flights currently use $\lambda = 0.25$ and $\mathsf{T} = 4$s, which corresponds to about 100m waypoint separation.

Figure 17 shows the results of simulating the same RH scenario using this propagation of the vehicle state. The simulation was done with both static wind and turbulence, neither of which were modeled in the planner dynamics, creating difficulties for the low level vehicle controller. Even with these unmodeled inputs, using the prediction step results in dynamically feasible plans for the vehicle with cross-track errors less than ± 5m. We have verified the accuracy of these hardware in the loop simulations, and we expect to see this type of improved performance once flight tests resume this spring.

Because of uncertainty in both the estimate of the static wind vector $\bar{w}$ and the random variations about this estimate $\delta\bar{w}$, the value of the estimated state $\hat{X}(t_k + \lambda\mathsf{T})$ is uncertain, which complicates the trajectory design problem. Assuming an upper bound on the wind disturbance, Ref. [12] presents an algorithm for designing trajectories that are *robustly feasible* for all disturbances in the bounded set. This is achieved by tightening the constraints in the problem, which can be done without increasing the problem size. Note that $\delta\bar{w}$ can be dynamically updated as the confidence in the disturbance estimates improves. We are investigating using these additional robustness constraints within the following algorithm (receding horizon control with propagation and feedback):

1) Plan an N step trajectory subject to dynamical and environmental constraints.
2) Execute the first step of the trajectory, T, and measure the vehicle state. Update the disturbance statistics for the propagation step.
3) Propagate state forward $\lambda\mathsf{T}$ and use the estimated state as the initial condition for the next plan.
4) Return to step 1

IV. CONCLUSIONS

This paper presents hardware demonstrations of the receding horizon task assignment and trajectory design on two new rover/blimp and UAV testbeds. These multi-vehicle testbeds provide unique platforms to evaluate various distributed coordination and control strategies. Future work
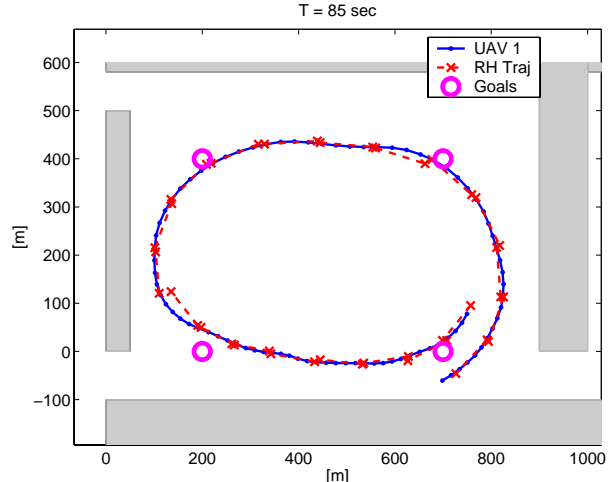

Fig. 17. Predicted performance of receding horizon control with propagation and feedback.

will integrate distributed collision avoidance formulations, task assignment with the formation of dynamic sub-teams, and missions with heterogeneous vehicles (e.g. several rovers and blimps).

REFERENCES

[1] P. Chandler, "Complexity in UAV cooperative control," *American Control Conference*, 2002.
[2] M. Alighanbari, Y. Kuwata and J. How, "Coordination and control of multiple uavs with timing constraints and loitering," *American Control Conference*, 2003.
[3] A. Richards, Y. Kuwata, and J. How, "Hardware demonstration of real-time MILP control," *AIAA Guidance, Navigation, and Control Conference*, 2003.
[4] J. Bellingham, M. Tillerson, A. Richards, and J. How, "Multi-task assignment and path planning for cooperating UAVs," *Conference on Cooperative Control and Optimization,*, 2001.
[5] A. Richards, J. Bellingham, M. Tillerson, and J. How, "Co-ordination and control of multiple UAVs," *AIAA Guidance, Navigation, and Control Conference*, 2002.
[6] G. Laporte and F. Semet, "Classical heuristics for the capacitated VRP," in *The Vehicle Routing Problem*, P. Toth and D. Vigo, Eds., Philadelphia: SIAM, 2002.
[7] L. Bertuccelli, M. Alighanbari, and J. How, "Robust planning for coupled, cooperative UAV missions," *submitted to the Conference on Decision and Control*, 2004.
[8] E. King, Y. Kuwata, M. Alighanbari, and J. How, "Coordination and control experiments for UAV teams," *American Aeronautical Society*, 2004.
[9] J. Bellingham, A. Richards, and J. How, "Receding horizon control of autonomous aerial vehicles," *American Control Conference*, p. 37413746, 2002.
[10] Y. Kuwata and J. How, "Stable trajectory design for highly constrained environments using receding horizon control," *IEEE American Control Conference*, 2004.
[11] R. Franz, M. Milam, and J. Hauser, "Applied receding horizon control of the Caltech ducted fan," *American Control Conference*, pp. 3735–3740, 2002.
[12] A. Richards and J. How, "Decentralized model predictive control of cooperating UAVs," *submitted to the IEEE Conference on Decision and Control*, 2004.