# FLYING ROBOTS: Sensing, Control and Decision Making

H. Jin Kim David H. Shim Shankar Sastry

Department of Electrical Engineering & Computer Sciences
University of California at Berkeley, Berkeley CA 94720
{jin,hcshim,sastry}@eecs.berkeley.edu

*Abstract*—**This paper presents a flight management system (FMS) implemented as on-board intelligence for rotorcraft-based unmanned aerial vehicles (RUAVs). This hierarchical flight management system endows each vehicle with autonomy resembling sense-reason-act processes of intelligent agents in nature. Sensor measurements are processed by flight control computers in order to gradually refine given abstract mission commands into real-time control signals for each vehicle. Intelligence based on probabilistic reasoning is provided by an on-board strategy planner. An on-board trajectory generates a sequence of way-points associated with flight modes, as a result of intelligent reasoning. A tracking control layer is designed using nonlinear model predictive control as well as conventional multi-loop PD control, and integrated with a trajectory generator for logistical action planning. The proposed structure has been implemented on radio-controlled helicopters and validated in a variety of experiments. Results from way-point navigation, a probabilistic pursuit-evasion game and vision-based tracking of a moving target show the promising potential of intelligent flying robots.**

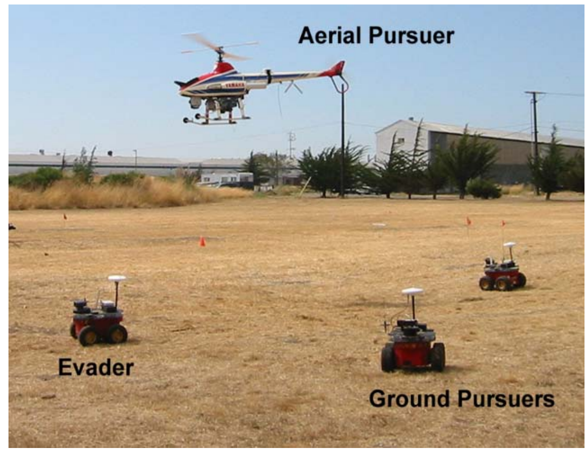


Fig. 1. Berkeley RUAV, Ursa Magna 2, in an autonomous search operation with Unmanned Ground Vehicles

## I. Introduction

Deployment of intelligent robots has been made possible through technological advances. Even though the definitions of "intelligence" and "robot" are still under the subject of much debate, there is little doubt that the world of the future will be filled with intelligent robotic agents employed to autonomously perform tasks, or embedded in systems all around us, extending our capabilities to perceive, reason and act, and substituting human efforts in applications where human participation is dangerous, inefficient and/or impossible. Subscribing to this idea, Rotorcraft-based unmanned aerial vehicles (RUAVs) attract special interests, due to their flight capabilities. The unique lift generation mechanism of rotorcrafts enables rich collection of flight modes such as hover, vertical take off/landing, pirouette, and sideslip. These versatile flight modes, which cannot be achieved by fixed-wing aircrafts, are often desired for high-fidelity detection, location and tracking of targets in a variety of scenarios, such as reconnaissance, aerial surveillance, public safety search and rescue activities.

While the last decade has witnessed remarkable progresses in RUAV research including modeling [1], control theory [2], [3], and small-size electronics, current technology is still far from achieving most real-world solutions. Much of our work as part of the BErkeley AeRobot (BEAR) research project has been directed toward improving the performance of RUAVs to be employed in real-world problems. As a representing problem that addresses many issues in heterogeneous multi-robot systems, we have been studying probabilistic pursuit-evasion games (PEGs), where a team of aerial and ground-based vehicles pursue a team of evading vehicles while concurrently building a map in an unknown environment. In [4], we presented an experimental platform for pursuit-evasion games with unmanned aerial vehicles (UGVs). In [5], we presented algorithms and experimental results on PEGs between our fleet of RUAVs and ground robots in a centralized setting.

In order to employ RUAVs in a wider range of applications, we need to consider decentralized framework for the efficient and reliable operations. As a step towards such a fully decentralized system, we develop individual autonomous agents, aerial and ground-based, and integrate them into a versatile and resilient system. This paper presents the synthesis of a flight management system (FMS) for RUAVs that endows RUAVs with autonomy and accessibility to independently sense, reason, plan and act in coordination with other agents or environments.

The remaining of this paper is as follows: Section II presents overview of flight management system for RUAVs. Section III describes the design of the tracking control system, highlighting the nonlinear model predictive approach. In Section IV, the proposed FMS is applied to three examples: pre-programmed way-point navigation using vehicle control language, a pursuit-evasion game, and vision-based high-speed tracking of a moving target, and Section V concludes the paper.

The authors would like to thank Hoam Chung and Cory Sharp. This research was supported by the ONR grants N00014-97-1-0946 and N00014-00-1-0621, and ARO MURI grant DAAH04-96-1-0341.

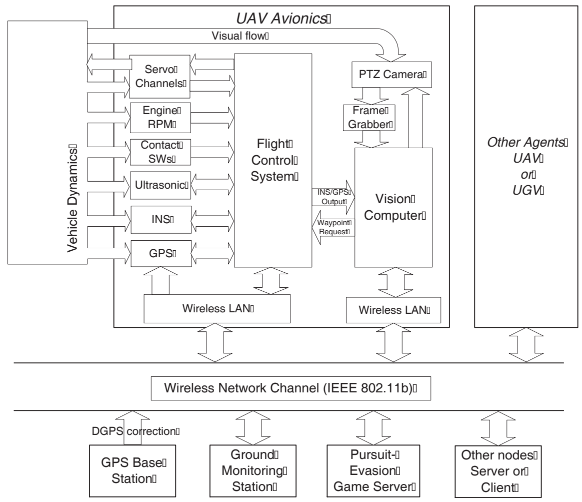


Fig. 2. Flight management system implemented for multi-agent scenarios

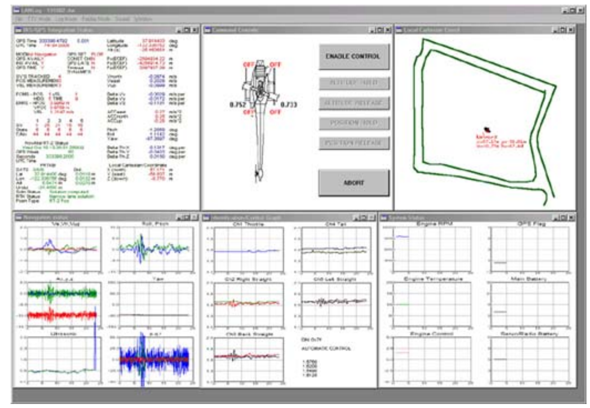

Fig. 3. Graphical interface on ground station of Berkeley RUAV testbed

## II. Flight Management System for Intelligent Unmanned Aerial Vehicles

An "intelligent agent" continuously (1) perceives dynamically changing conditions in its environment, (2) reasons to interpret perceived information, to solve problems and to determine appropriate action, and (3) acts appropriately to affect conditions in its environment. Based on this abstract definition, we presented a hierarchical architecture in [4]. This section describes implementation of each component in our flight management system shown in Fig. 2.

### A. Sensing

Dynamically changing conditions in the environment and the vehicle states are perceived by various on-board sensors. The precise guidance of the host vehicle of much smaller size demands more accurate navigation sensors. GPS-based INS is employed as a central navigation sensor-suite in order to correct the unbounded error of strap-down INS by supplementing a high-accuracy differential GPS. Additional Kalman Filter is used in order to generate position estimates for high-speed position control. Localizing sensors such as ultrasonic sensors and laser range-finders augments the navigation sensor unit for the acquisition of the environment-specific information such as relative distance from the ground surface and for the detection of nearby objects around the host vehicle. Contact switches are installed on the landing gear of the helicopter primarily to assist automatic take-off/landing. A computer vision system [6] is used to detect the target objects or estimate the relative position and attitude with the help of INS/GPS.

### B. Reasoning & Coordination

Data sensed by sensor suites should be properly interpreted by a strategy planner implemented on a flight control computer. When this information is not enough to identify the current state of the world, the world is modeled as a partially observable Markov decision process (POMDP), as described later in Section IV-B. The strategy planner then updates each agent's *belief (information) state*, i.e., probability distribution over the state space of the world, given measurement and action histories, and generates a policy, i.e, a mapping from the agent's belief state to its action set. Search of the optimal policy is computationally intractable in most problems, thus usually sub-optimal policies are implemented [5], or, the class of policies to search through is limited [7]. These algorithms are run on real-time operating systems to satisfy hard real-time constraints.

The strategy planner also manages the configuration of the communication network. The role of communication in the FMS for RUAVs is more critical than in conventional FMSs for manned vehicles, because RUAVs should report the vehicle status and accept external commands typically at a faster rate than human voice communication. Moreover, the support of high quality-of-service (QoS) wireless communication system is desirable in order for multiple RUAVs to function as a tightly coordinated, reconfigurable, distributed networked intelligence.

### C. Action

One of the most essential capabilities of an RUAV is to autonomously guide itself through the requested trajectories or way-points, in an autonomous manner. Each vehicle platform needs a flight controller that generates real-time control signals from the way-points requested by higher-level planners. Such a controller should be able to stabilize and follow the given trajectory in the presence of input saturation, state constraints and strong disturbance, as will be described in III-B. Action-sensing coordination occurs at a very fast rate in order to cope with contingencies, for example, such as detection and avoidance of collisions.

## III. Flight Control and Trajectory Generation

This section describes the configuration of RUAV platforms and the design of control and trajectory generation layer at the vehicle-level of the hierarchy for autonomous flight.

### A. Vehicle Platform

Berkeley RUAVs are built on commercial off-the-shelf (COTS) radio-controlled helicopters of various sizes and payloads. In the experiments reported in this paper, an industrial radio-controlled helicopter, Yamaha R-50, is used. The vehicle platform is equipped with on-board navigation computers and sensors previously shown in Fig. 2. The flight control software, implemented on QNX$^{\text{TM}}$ real-time operation system, manages sensors, vehicle control, and communication. More detailed theoretical and practical issues in building an RUAV are described in [8].

While the autonomy of each vehicle is important, intervention of human intelligence is often necessary due to contingencies or mission characteristics. Open-control architecture allows each strategic planner to accept incoming requests from human operators for mixed initiative planning through human-to-console and console-to-UAV interface. The human-to-console interface, implemented as a graphic-user-interface (GUI) shown in Fig. 3, receives human commands and displays the information downloaded from the UAV. The console-to-RUAV interface sends the commands in a proper data structure to the RUAV controller and receives the RUAV status.

### B. Vehicle Stabilization & Control

Multi-input multi-output (MIMO), nonlinear characteristics and severe disturbance must be accounted for to acquire precise models. A lumped parameter method yields a six degree-of-freedom rigid body model, augmented with the first-order servo-rotor dynamics [1], [8]:

$$\begin{aligned}
\dot{\mathbf{x}}(t) &= f_c(\mathbf{x}(t), \mathbf{u}(t)), \quad \mathbf{x}(0) \text{ given} \qquad (1)\\
\mathbf{x} &= [\mathbf{x}^K, \mathbf{x}^D] \in \mathbb{R}^{n_x}\\
\mathbf{x}^K &= [x^S, y^S, z^S, \phi, \theta, \psi]\\
\mathbf{x}^D &= [\dot{x}^B, \dot{y}^B, p, q, a_{1s}, b_{1s}, \dot{z}^B, r, r_{fb}]\\
\mathbf{u} &= [u_{a1s}, u_{b1s}, u_{\theta_M}, u_{\theta_T}] \in \mathbb{R}^{n_u},
\end{aligned}$$

where $S$ and $B$ denote spatial and body coordinate respectively, and $\phi$, $\theta$, and $\psi$ denote roll, pitch, and yaw, respectively. $p$, $q$, and $r$ denote the angular rates in roll, pitch, and yaw direction in body coordinate system, respectively. The transformation between spatial and body coordinates are given by

$$[\dot{x}^S, \dot{y}^S, \dot{z}^S]^T = \mathbf{R}^{B\to S}[\dot{x}^B, \dot{y}^B, \dot{z}^B]^T,$$

where $\mathbf{R}^{B\to S} \in \mathbf{SO(3)}$ is the rotational matrix of the body axis relative to the spatial axis, represented by $ZYX$ Euler angles $[\phi, \theta, \psi]$. $a_{1s}$ and $b_{1s}$ are longitudinal and lateral flapping angles, and $r_{fb}$ is the feedback gyro system state [1]. $\mathbf{u}$ consists of inputs to the lateral cyclic pitch, longitudinal cyclic pitch, main rotor collective pitch, and tail rotor collective pitch. The unstable RUAV dynamics needs proper stabilization using feedback control, which is performed by the on-board real-time controller. A multi-loop PD (MLPD) controller [9] and a linear robust control system designed with $\mu$-synthesis theory [8] demonstrated stable responses. Especially, hover control by MLPD showed 0.5 m accuracy in the x and y directions, 0.1 m in the altitude, and 3° in the heading. In order to address nonlinear nature and input/state saturation over the flight envelope including agile maneuvers, we design a nonlinear model predictive tracking controller (NMPTC) on Eqn. (2), which is discretized from (1).

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k). \qquad (2)$$

and a cost function for tracking is defined by

$$J \triangleq \phi(\tilde{\mathbf{y}}_N) + \sum_{k=0}^{N-1} L(\mathbf{x}_k, \tilde{\mathbf{y}}_k, \mathbf{u}_k) \qquad (3)$$

with

$$\begin{aligned}
\phi &\triangleq \frac{1}{2}\tilde{\mathbf{y}}_N^T \mathbf{P}_0 \tilde{\mathbf{y}}_N\\
L &\triangleq \frac{1}{2}\tilde{\mathbf{y}}_k^T \mathbf{Q} \tilde{\mathbf{y}}_k + \frac{1}{2}\mathbf{x}_k^T \mathbf{S} \mathbf{x}_k + \frac{1}{2}\mathbf{u}_k^T \mathbf{R} \mathbf{u}_k
\end{aligned}$$

where $\tilde{\mathbf{y}} \triangleq \mathbf{y}_d - \mathbf{y}$, $\mathbf{y} = \mathbf{C}\mathbf{x} \in \mathbb{R}^{n_y}$, $\mathbf{y}_d$ is the desired trajectory, and $\mathbf{S}$ is introduced to bound the state variables that do not directly appear in $\mathbf{y}$. By introducing a sequence of Lagrange multiplier vectors $\{\lambda_k \in \mathbb{R}^{n_x}\}_{k=1}^N$, Eqn. (3) can be written as

$$\begin{aligned}
J = \phi(\tilde{\mathbf{y}}_N) &+ \sum_{k=0}^{N-1} L(\mathbf{x}_k, \tilde{\mathbf{y}}_k, \mathbf{u}_k) \qquad (4)\\
&+ \lambda_{k+1}^T [f(\mathbf{x}_k, \mathbf{u}_k) - \mathbf{x}_{k+1}].
\end{aligned}$$

With the Hamiltonian function

$$H_k = L(\mathbf{x}_k, \tilde{\mathbf{y}}_k, \mathbf{u}_k) + \lambda_{k+1}^T f(\mathbf{x}_k, \mathbf{u}_k), \qquad (5)$$

we have

$$\frac{\partial H_k}{\partial \mathbf{u}_k} = \mathbf{u}_k^T \mathbf{R} + \lambda_{k+1}^T \frac{\partial f_k}{\partial \mathbf{u}_k}. \qquad (6)$$

Since we are interested in finding $\{\mathbf{u}_k\}_{k=1}^N$ that minimizes J, $\{\lambda_k\}_{k=1}^N$ are chosen to simplify $dJ$ as the following:

$$\begin{aligned}
\lambda_N^T &= \frac{\partial \phi}{\partial \mathbf{x}_N} = -\tilde{\mathbf{y}}_N^T \mathbf{P}_0 \mathbf{C} \qquad (7)\\
\lambda_k^T &= \frac{\partial H_k}{\partial \mathbf{x}_k} + \frac{\partial H_k}{\partial \tilde{\mathbf{y}}_k}\frac{\partial \tilde{\mathbf{y}}_k}{\partial \mathbf{x}_k}\\
&= \mathbf{x}_k^T \mathbf{S} + \lambda_{k+1}^T \frac{\partial f_k}{\partial \mathbf{x}_k} - \tilde{\mathbf{y}}^T \mathbf{Q} \mathbf{C}, \qquad (8)
\end{aligned}$$

The control law $\mathbf{u}_k$ is found by gradient-descent method:

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \Delta_k \frac{\partial H_k}{\partial \mathbf{u}_k} \qquad (9)$$

The details on this tracking controller design and on-line optimization using Eqn. (2), (6), (7), and (8), are reported in [10].

`TakeoffTo` ⟨coord⟩ {abs,rel}
: perform autonomous take-off to a certain target point

`Hover` ⟨coord⟩ {abs,rel} {heading=⟨heading⟩ {deg,rad}}
⟨duration⟩ {sec,min}
: hover with a given heading angle for a given time

`FlyTo` ⟨coord⟩ {abs,rel}
{vel ⟨velocity⟩ {mps,kmps,fps,knots,mph}}
{passby,stopover} {autoheading,
heading=⟨heading⟩ {deg,rad}}
: cruise to a certain way-point, stopping over or passing by

`MoveTo` ⟨coord⟩ {abs,rel}
{vel ⟨velocity⟩ {mps,kmps,fps,knots,mph}}
{autoheading, heading=⟨heading⟩ {deg,rad}}
: move to a certain way-point to stop-over with a fixed heading

`BankToTurn` ⟨heading change⟩ {deg,rad}
{{radius} ⟨radius⟩{m,ft}}
{{vel} ⟨velocity⟩ {mps,kmps,fps,knots,mph}}
: perform bank-to-turn during cruise

`Land` : command the vehicle to land

Fig. 4. Vehicle Control Language Syntax

### C. Trajectory Generation

Trajectory generation layer, as a coordinator between the stabilization/tracking layer and the strategic planner, is responsible for refining reference trajectories and triggering the proper control law of the stabilization/tracking layer in order to execute each of these flight modes in a pre-programmed sequence or dynamically upon request. In designing such a way-point navigator, we employ a novel framework, the vehicle control language (VCL). VCL is the standard interface which allows external systems as well as ground operators to request flight sequences using the provided flight command set in Fig. 4. Via rapidly reprogrammable, easily transmitted VCL codes, we obtain the isolation between the strategic planner and the stabilization layer. By abstracting away the details of sensing and control of each agent, we gain the interoperability of a unified framework for high-level planning across heterogeneous platforms. Yet by considering the dynamics of each vehicle in upper-level trajectory planning, the overall system can achieve real-time performance. In the context of VCL, a given flight is decomposed into a sequence of flight modes such as hover, forward flight, bank-to-turn, etc. A set of VCL commands is sent to the VCL execution module residing in the flight computer as a static command file or dynamic command set, over communication channels such as wireless Ethernet or RS-232 serial link. A VCL module consists of the user interface part on the ground station, the language interpreter, and the sequencer on the RUAV FCS.

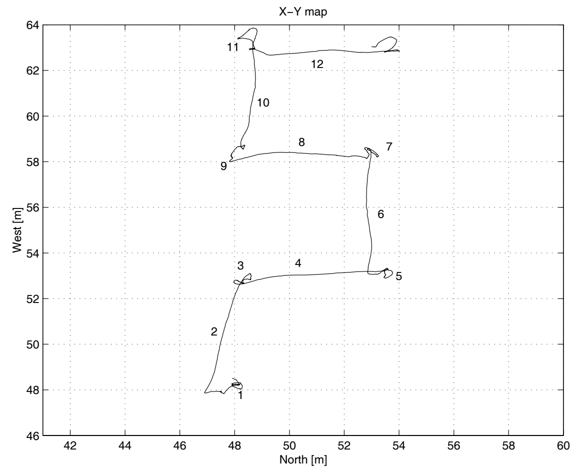


Fig. 5. Flight experiment result for lawn-mowing pattern

## IV. Experiments

In this section, we evaluate the effectiveness of the proposed hierarchical FMS in three types of examples: 1) trajectory tracking/way-point navigation, 2) a pursuit-evasion game (PEG), and 3) high-speed tracking of a moving target assisted by the onboard vision computer.

### A. Way-point Navigation & Trajectory Tracking

This section presents the essential action capability of an RUAV: autonomous navigation. The goal of RUAV is to follow given way-points or reference trajectories, thus, the VCL execution module assumes the highest hierarchy on the guidance of the RUAV.

First, a lawn-mowing pattern is used as a reference trajectory in the flight test shown in Fig. 5. The flight mode, way-point, and other optional parameters are extracted in each line of VCL and then sent to the trajectory coordination layer. Upon the reception of a new VCL command, the on-board VCL execution module activates a suitable control module for the current flight mode associated with the target way-point and other options. The stabilization/tracking layer employing the MLPD controller generates real-time control output at 50Hz for the actuators on the host RUAV.

Although the performance of MLPD controller was satisfactory as shown in Fig. 5, we observe the sideway drift due to the uncompensated lateral force of tail rotor, which is unavoidable in SISO approach. Furthermore, MLPD approach fails to address nonlinearity, input saturation, and state constraints. Motivated by these observations, NMPTC described in III-B is evaluated as an alternative to conventional MLPD controller for a trajectory tracking. As shown in Fig. 6, the benchmark trajectory of the upward spiral with nose-in requires the controller to track (x,y,z) position and heading simultaneously, i.e., $[x_d^S, y_d^S, z_d^S, \psi_d] = [R(t)\cos\frac{2\pi}{10}t, R(t)\sin\frac{2\pi}{10}t, -\frac{2\pi}{10}t, \pi + 2\pi 10t]$, for $0 \le t \le 20sec, R(t) = 5 + t/2$. Since the reference heading rate is given in the opposite direction to the anti-torque of the

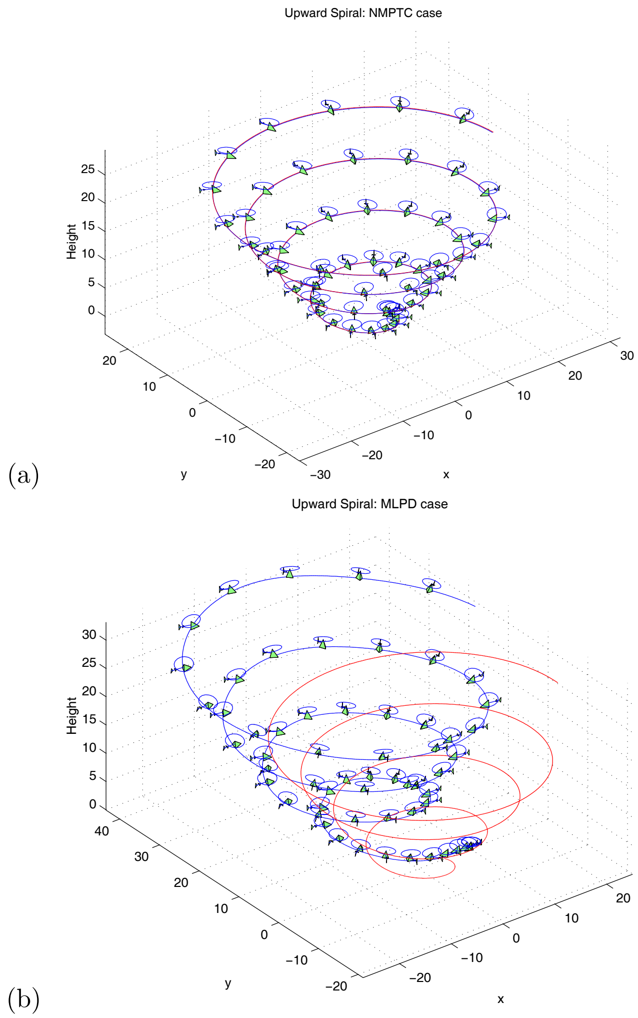


Fig. 6. Upward spiral trajectory tracking, with (a) NMPTC, (b) MLPD controller

main rotor, an extra sideward force is required from the tail rotor. This action generates more sideslip force, which cannot be easily compensated by MLPD controller, due to its incapability to capture the coupled MIMO characteristics of the asymmetric single main/tail rotor configuration. In contrast, NMPTC shows superior tracking performance due to its formulation based on MIMO nonlinear vehicle dynamics. The tremendous computing load, which has been a major drawback of the model predictive control, is dramatically alleviated by the introduction of more powerful processors.

## *B. Pursuit-Evasion Game*

This experiment evaluates the reasoning-action coordination and the performance of dynamic VCL in a pursuit-evasion game (PEG) [5]. The goal of pursuers is to "capture" evaders in a given grid-field. An evader is considered as captured when it is located within a certain range (e.g., 1.5 m) from a pursuer and in the pursuer's visibility region. The initial locations of evaders are unknown a priori. At each time step, the group of pursuers is required to go to the requested way-points and take measurements of their own locations and of any evaders within their visibility regions using the sensor-suites. This measurement is used to build probabilistic maps of the possible locations of evaders and decide the pursuers' next action that minimizes the capture time. From the pursuers' point of view, this PEG is modeled as a POMDP, i.e., a tuple $\langle \mathcal{S}, \mathcal{A}, T, \mathcal{Z}, O, r\rangle$[1]:

- $\mathcal{S}$ is a finite set of states of the world, i.e., the configurations of the pursuers and evaders in the given field;
- $\mathcal{A}$ is a finite set of the pursuers' actions;
- $T : \mathcal{S} \times \mathcal{A} \rightarrow \mathbf{PD}(\mathcal{S})$ is a transition function. $T(s', s, a_t) = \mathrm{P}(\mathbf{s}(t+1) = s' \mid \mathbf{s}(t) = s, \mathbf{a}(t) = a_t)$ is the probability of landing in the state $s' \in S$ under the action $a \in A$ from the state $s \in S$;
- $\mathcal{Z}$ is a finite set of observations the pursuers can experience of their world;
- $O : \mathcal{S} \times \mathcal{A} \rightarrow \mathbf{PD}(\mathcal{Z})$ is the observation function. $O(z_t, s', a_{t-1}) = \mathrm{P}(\mathbf{z}(t) = z_t \mid \mathbf{s}(t) = s', \mathbf{a}(t-1) = a_{t-1})$ is the probability of making observation $z$ given that the pursuer took action $a_t$ and landed in state $s'$;
- $r : \mathcal{S}\times\mathcal{A}\times\mathcal{Z} \rightarrow \mathbb{R}$ is a reward function, e.g., $r(s, a_t, z_t) = 1$ if $s$ corresponds to the evader-captured configuration and 0 otherwise.

The pursuers' belief state, $\eta^t_{(s)} \triangleq \mathrm{P}(\mathbf{s}(t) = s \mid \mathbf{A}_{t-1} = A_{t-1}, \mathbf{Z}_t = Z_t)$ represents the conditional probability that the world is in state $s$, given $\eta^0_{(s)} \triangleq \mathrm{P}(\mathbf{s}(0) = s)$, and the action and observation histories, i.e., $A_{t-1} \triangleq \{a_0, \cdots, a_{t-1}\}$, and $Z_t \triangleq \{z_0, \cdots, z_t\}$. Given that the pursuer observes $z_{t+1}$ after applying $a_t$, the recursive belief state dynamics are obtained by applying Bayes' rule:

$$\begin{aligned}
\eta^{t+1}_{(s')} &= \frac{\mathrm{P}(\mathbf{s}(t+1)=s', \mathbf{A}_t=A_t, \mathbf{Z}_{t+1}=Z_{t+1})}{\sum_{s'\in\mathcal{S}} \mathrm{P}(\mathbf{s}(t+1)=s', \mathbf{A}_t=A_t, \mathbf{Z}_{t+1}=Z_{t+1})} \\
&= \frac{\mathrm{P}(s', A_{t-1}, Z_t, a_t, z_{t+1})}{\sum_{s'\in\mathcal{S}} \mathrm{P}(s', A_{t-1}, Z_t, a_t, z_{t+1})} \\
&= \frac{\mathrm{P}(z_{t+1}|s', a_t) \sum_{s\in\mathcal{S}} \mathrm{P}(s', s, A_{t-1}, Z_t, a_t)}{\sum_{s'\in\mathcal{S}} \mathrm{P}(s', A_{t-1}, Z_t, a_t, z_{t+1})} \\
&= \frac{O(z_{t+1}, s', a_t) \sum_{s\in\mathcal{S}} T(s', s, a_t)\eta^t_{(s)}}{\sum_{s'\in\mathcal{S}} O(z_{t+1}, s', a_t) \sum_{s\in\mathcal{S}} T(s', s, a_t)\eta^t_{(s)}},
\end{aligned}$$

whose denominator can be treated as a normalizing factor, independent of $s'$.

We implemented a variety of computationally-efficient sub-optimal policies [5], including a greedy policy with respect to $\eta^{t+1}(s')$, under which the location in the pursuer's one-step reachability region with the highest probability of containing the evader at the next step is selected as the way-point for the pursuers. These policy computation algorithms are run in real-time using blocking socket of TCP/IP communication and the incoming VCL commands are processed by the on-board VCL execution module as previously described. Fig. 7 shows a PEG of one greedy aerial pursuer vs. one greedy ground evader in a 20m x 20m field. The number of participating agents can be easily changed. The setup of one aerial pursuer is shown so that the load of RUAV is maximized. Along with the trajectories for the pursuer and evader, the evolution of the probabilistic map is shown as the gray-scale background and the square represents the visibility region of RUAV,

[1] Random variables are indicated in bold type according to the usual convention.

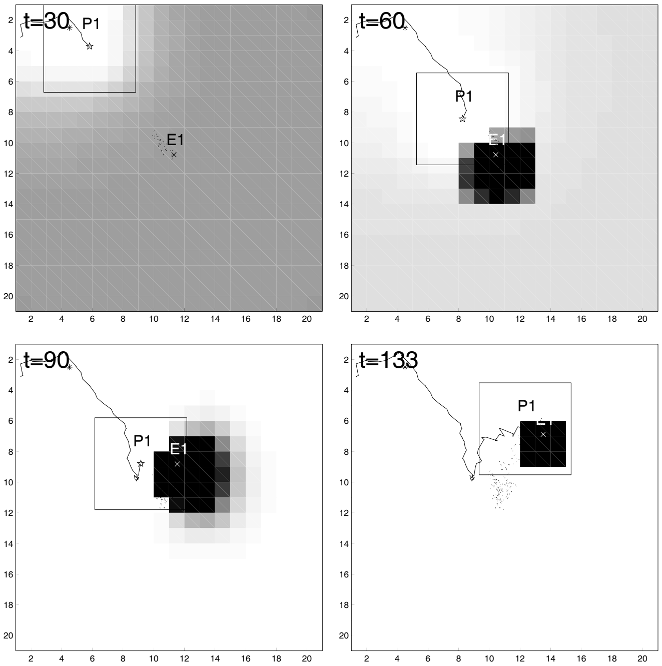


Fig. 7. Snapshots of 1 vs. 1 Pursuit-Evasion Game: Dark color denotes the high probability of containing an evader. (P: Pursuer RUAV, E: Evader UGV)

until the evader is captured in 133 seconds. This experiment shows that the proposed control law and dynamic VCL are well-suited in a hierarchical control structure for the PEG.

### C. High-speed position tracking

This experiment evaluates the sensing-action coordination of an RUAV: the RUAV is required to track a moving UGV, whose relative position is estimated by an on-board vision computer. The vision computer estimates the location of the ground target using a detection algorithm extracting a special marker [6]. A specially-tuned way-point navigator is activated to process the high-rate position-tracking request, i.e., at 3 Hz in this case. In Fig. 8, the trajectories of the RUAV and UGV are shown. The RUAV system shows satisfactory tracking performance with a small error attributed to wind gusts. In the middle of the experiment, it is noticed that the vision computer ceased sending the reference trajectory about 8 seconds. The RUAV FMS demonstrated its fail-safe feature against this adversarial situation by following an *expected* trajectory of targets until a next way-point command is received.

## V. Conclusion

This paper presented a hierarchical flight management system designed for intelligent RUAVs. The experimental results validate the satisfactory performance of the multi-functional flight management system constructed on Berkeley RUAVs in the three examples considered in this paper: way-point navigation, a pursuit-evasion game and tracking of a moving target. Further improvements on tracking performance is obtained by employing the nonlinear model predictive method. Future research effort will be focused on expanding the capability of the flight control system with rich strategy planning logics and increased robustness in order to narrow down the gap between current RUAVs and intelligent flying robots.

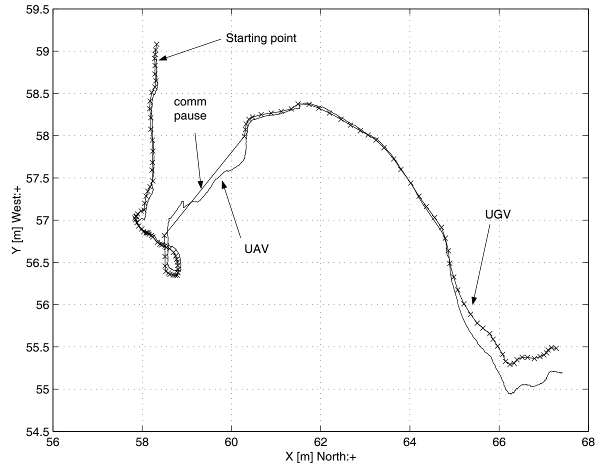


Fig. 8. Tracking of a ground target

## References

[1] B. Mettler, M. B. Tischler, and T. Kanade, "System identification of small-size unmanned helicopter dynamics," in *American Helicopter Society 55th Forum*, Montreal, Quebec, Canada, May 1999.
[2] H. Shim, T. J. Koo, F. Hoffmann, and S. Sastry, "A comprehensive study of control design for an autonomous helicopter," in *Proc. of 37th IEEE Conference on Decision and Control*, Dec. 1998, pp. 3653–3658.
[3] J. E. Corban, A. J. Calise, and J. V. R. Prasad, "Implementation of adaptive nonlinear control for flight test on an unmanned helicopter," in *37th IEEE Conference on Decision and Control*, 1998, pp. 3641–3646.
[4] R. Vidal, S. Rashid, C. Sharp, O. Shakernia, H.J. Kim, and S. Sastry, "Pursuit-evasion games with unmanned ground and aerial vehicles," in *Proc. of IEEE Conference on Robotics and Automation*, 2001, pp. 2948–2955.
[5] H. J. Kim, R. Vidal, H. Shim, and S. Sastry, "A hierarchical approach to probabilistic pursuit-evasion games with unmanned ground and aerial vehicles," To appear in *Proc. of 40th IEEE Conference on Decision and Control*, 2001.
[6] C. S. Sharp, O. Shakernia, and S. S. Sastry, "A vision system for landing an unmanned aerial vehicle," in *IEEE International Conference on Robotics and Automation*, Seoul, Korea, 2001, pp. 1720–1727.
[7] A. Y. Ng and M. Jordan, "PEGASUS: A policy search method for large MDPs and POMDPs," in *Proc. of 17th International Conference on Uncertainty in Artificial Intelligence*, 2000.
[8] D. H. Shim, *Hierarchical Control System Synthesis for Rotorcraft-based Unmanned Aerial Vehicles*, Ph.D. thesis, University of California at Berkeley, 2000.
[9] D. H. Shim, H. J. Kim, and S. Sastry, "Hierarchical control system synthesis for rotorcraft-based unmanned aerial vehicles," in *AIAA Guidance, Navigation and Control Conference*, Denver, CO, August 2000.
[10] H. J. Kim, D. H. Shim, and S. Sastry, "A flight management system for intelligent unmanned aerial vehicles with nonlinear model predictive control," Submitted to *21st American Control Conference*, 2002.