2013 International Conference on Unmanned Aircraft Systems (ICUAS)
May 28-31, 2013, Grand Hyatt Atlanta, Atlanta, GA

Decentralized Control of Unmanned Aerial Vehicles for Multitarget Tracking

Shankarachary Ragi and Edwin K. P. Chong

Abstract— **We design a guidance control method for a fleet of autonomous unmanned aerial vehicles (UAVs) tracking multiple targets in a decentralized setting. Our method is based on the theory of decentralized partially observable Markov decision process (Dec-POMDP). Like partially observable Markov decision processes (POMDPs), it is intractable to solve Dec-POMDPs exactly. So, we extend a POMDP approximation method called *nominal belief-state optimization* (NBO) to solve Dec-POMDP. We incorporate the cost of communication into the objective function of Dec-POMDP, i.e., we explicitly optimize the communication among the UAVs along with the kinematic-control commands for the UAVs. We measure the performance of our guidance method with the following metrics: 1) average target-location error, and 2) average communication cost. The goal to maximize the performance with respect to each of the above metrics conflict with each other, and we show through empirical study how to trade off between these performance metrics using a scalar parameter.**

I. Introduction

There has been a growing interest in the development of guidance methods for UAVs for target tracking. Our work in the past [1], [2] focused on centralized guidance, where there was a notional central controller that collects measurements from the sensors on-board the UAVs, forms tracks on the targets, computes control commands for the UAVs, and sends them back to the UAVs. But that approach did not account for the cost of communication. In general, the communication among the UAVs or between UAVs and a central controller (e.g., ground station) comes at a cost.

The decentralized control of UAVs was studied before in various contexts, e.g., decentralized UAV formation flight [3], [4], decentralized cooperative search in an uncertain environment by UAVs [5], decentralized model predictive control of UAVs [6], and decentralized task assignment for UAVs [7]. In this study, we design a decentralized guidance control method for UAVs for target tracking based on the theory of *decentralized partially observable Markov decision process* (Dec-POMDP). In our problem, the communication among the UAVs is not free, and we explicitly optimize the communication decisions of each UAV (whom to communicate with and when to communicate) along with the kinematic controls. In this method, a UAV collects measurements of targets from the on-board sensors, forms tracks on the targets based on the local observations and the information received from other UAVs, and computes control commands for the local UAV. This kind of communication-aware motion planning for mobile agents has been studied before, e.g., [8]. However, our approach differs from the existing communication-aware planning approaches in that we place our planning method in the context of Dec-POMDP.

Dec-POMDP [9] is a mathematical framework useful for modeling resource control problems where the decision making is decentralized. The Dec-POMDP approach has the following advantages: 1) it offers a general approach to dealing with resource management in a multi-agent scenario with decentralized control, 2) it is a non-myopic approach, which trades off short-term for long-term performance, 3) in comparison to the existing greedy/myopic approaches in the literature, the Dec-POMDP approach results in more effective exploitation of limited resources in a decentralized setting. In addition, we explicitly optimize the communication between the UAVs (also called agents in this paper). In general, solving a Dec-POMDP exactly is intractable. We can extend centralized POMDP (*partially observable Markov decision process*) approximation methods

The authors are with the Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523-1373, USA. Emails: {shankar.ragi, edwin.chong}@colostate.edu

978-1-4799-0817-2/13/$31.00 ©2013 IEEE

to solve the Dec-POMDP. The goal is to design a decentralized UAV guidance method that is implementable in real-time. Therefore, we need an approximation method with computational requirements that are not prohibitive. In this study, we choose an approximation method called *nominal belief-state optimization* (NBO), which we used in the past [1], [2] to solve a centralized POMDP. The NBO method is relatively less expensive (in time-consumption) than other approximation methods.

The Dec-POMDP evolves in discrete time-steps, where the length of each time-step is T seconds. We use k as the discrete-time index. For the purpose of this study, we assume that the decision epochs at each agent are synchronized.

II. Problem Specification

Targets. The targets are ground-based vehicles and are assumed to be moving on a 2-D plane.

Unmanned Aerial Vehicles. For the purpose of this study, we assume that the UAVs fly at a constant altitude, i.e., the UAVs move on a 2-D plane. The kinematics of each UAV is controlled by forward acceleration and bank angle. Each UAV is equipped with an on-board sensor that generates the position measurements of the targets. For simplicity, we assume that the locations and velocities of every UAV in the system are available at each UAV (e.g., by communicating GPS coordinates).

Measurement Error. The sensors on-board each UAV generate measurements of the target locations. The measurement of a target location at a sensor is corrupted by a spatially varying random error that depends on the relative location of the target with respect to the location of the sensor/UAV.

Communication. The UAVs have communication capabilities, i.e., each UAV can transmit and receive information to/from other UAVs. The next section provides a detailed description of the communication between the agents.

Tracker. Each UAV is equipped with a tracker (tracking algorithm) that maintains tracks on each target and updates them via Kalman filter equations given the local observations and the information received from other UAVs.

Objective. Our goal is to control each UAV, in a decentralized setting, such that a cost metric, which includes the mean-squared error between the tracks and the targets and the cost of communication, is minimized (discussed later).

III. Communication Between Agents

In this study, we assume that there is an underlying communication network over which the agents can communicate information to each other. We will not concern ourselves with the details of this underlying network. Instead, for each pair of agents, we will treat the end-to-end path through the network between them as an abstract *communication link* over which the two agents can exchange information. To account for the cost of communicating information over the network, we assign a numerical value to each communication link (which we will call the *cost* of the link). For simplicity, the value of the cost of a link is assumed to be proportional to the distance between the two associated agents. More precisely, if d_k^{ij} is the distance between the ith and the jth UAVs at time k, then the cost of their link is αd_k^{ij}, where α is a given proportionality constant. We further assume that communication delays are sufficiently small relative to the time duration between decision epochs that any information communicated at discrete-time k is received in time for decision-making at discrete-time $k+1$. The communication among the agents is restricted in the following ways:

- At any decision epoch, an agent can communicate with at most one other agent. The rationale for this restriction is that allowing for multiple agent destinations per decision epoch results in prohibitive computational requirements when making optimal communication decisions over a long time horizon (planning over a long horizon is a characteristic of Dec-POMDP formulation).
- If an agent i decides to communicate with agent j at time k, then the agent i can choose and send to j one of the L locally generated target observations in the past L time-steps (including the current-step). We set the value of L to a sufficiently small value so that the computational requirement is not prohibitive.

IV. Problem Formulation

In this section, we cast the UAV guidance problem into the framework of a *decentralized partially*

observable Markov decision process (Dec-POMDP for short). To cast the UAV guidance problem into the Dec-POMDP framework, we need to define the following key components in terms of our guidance problem as follows.

A. Dec-POMDP Ingredients

Agents. There are N agents (or UAVs) in the system. Let $\mathcal{I} = \{1, \ldots, N\}$ represent the set of agents.

States. We account for three subsystems in specifying the state: the UAV(s), the target(s), and the tracker (includes the state of the tracking algorithm at each agent). More precisely, the state at time k is given by $x_k = (s_k, \chi_k, T_k)$, where s_k represents the UAV state, χ_k represents the target state, and T_k represents the joint-track state. The UAV state s_k includes the locations and velocities of each UAV. The target state χ_k includes the location and velocity of each target. The joint-track state is given by $T_k = (T_k^1, \ldots, T_k^N)$, where $T_k^i = (\xi_k^i, P_k^i)$ is the state of the tracking algorithm at agent i, where ξ_k^i is the posterior mean vector and P_k^i is the posterior covariance matrix.

Joint Actions. The joint-action is a tuple, the components of which are actions corresponding to individual agents. Let $u_k = (u_k^1, \ldots, u_k^N)$ represent the joint-action, where u_k^i is the action-vector at agent i. The local action vector u_k^i includes the kinematic controls and the communication decisions for the ith UAV at time k. The kinematic controls of a UAV includes its forward acceleration and bank angle. The communication decision at agent i at time k can be represented by (g_k^i, l_k^i), where $g_k^i \in \mathcal{I} \cup \{0\} - \{i\}$ is the ID of the agent whom agent i will communicate with at time k ($g_k^i = 0$ means that agent i will not communicate with any other agent at time k), and $l_k^i \in \{k - L, \ldots, k\}$ is the chosen time-step from the past L time-steps (starting from k) such that the agent i will send to agent g_k^i its local target observation generated at time-step l_k^i. The local action at agent i, $i = 1, \ldots, N$, can be represented by $u_k^i = (a_k^i, g_k^i, l_k^i)$, where a_k^i includes the kinematic controls (forward acceleration and bank angle) for agent/UAV i, and (g_k^i, l_k^i) represents its communication decision at time k.

State Transition Law. The state transition law specifies the next-state distribution given the current state and joint-action, i.e., $x_{k+1} \sim p_k(\cdot|x_k, u_k)$, where p_k is a conditional probability distribution. Since we have several sub-systems to describe the state, we define the state-transition law separately for each sub-system. The state of UAV i, $i = 1, \ldots, N$, evolves according to the kinematic equations given the actions/control commands, i.e., $s_{k+1}^i = \psi(s_k^i, a_k^i)$ (the function ψ is defined later), where a_k^i represents the local kinematic controls (forward acceleration and bank angle). The target state evolves according to $\chi_{k+1} = \mathbf{F}\chi_k + e_k$, where e_k represents the process noise, where the distribution of e_k is given by $e_k \sim \mathcal{N}(0, \mathbf{Q})$. We use constant velocity model [10], [11] to represent the dynamics of a target (see [10] or [11] for the definition of $\mathbf{F}$ and $\mathbf{Q}$). Finally, the track state at each agent evolves according to the Kalman filter update equations given the local observations and the observations received from other agents.

Observations and Observation Law. Each joint-observation is a tuple, the components of which are observations made by individual agents. Let $z_k = (z_k^1, \ldots, z_k^N)$ be the joint-observation vector at time k, where z_k^i represents the observation at agent i. The observation law specifies the distribution of joint-observations given the current state and joint-action, i.e., $z_k \sim q_k(\cdot|x_k, u_k)$, where q_k is a conditional probability distribution. The observations at agent i, represented by z_k^i, includes the observation of UAV state, i.e., the current location and velocity of each UAV, the local track state T_k^i, and the target state. We assume that the UAV and the local track states are fully observable at an agent, and the target state is not fully observable; instead, the agent generates a random measurement of the target state that is a function of the locations of the targets and the agent. An agent i, $i = 1, \ldots, N$, upon receiving information about the target-state (i.e., target observations) from other agents, fuses the locally generated observations with the observations received from other agents, and updates the local track state.

Cost Function. A cost function $C(x_k, u_k)$ specifies the cost (real number) of being in a given state x_k and performing a joint-action u_k. The cost function includes the mean-squared tracking error and the cost of communication (details are discussed later).

The Dec-POMDP starts at a random initial state x_0 (whose distribution is given), and at any typical time-step k, the state x_k transitions to x_{k+1} given the joint-action vector u_k. The joint-action u_k performed at the current state x_k incurs a global cost $C(x_k, u_k)$. As a Dec-POMDP evolves over time as a dynamical process, the agents may not know the underlying state exactly, but each agent generates observations of the underlying state, providing the agent with clues of the actual underlying states. Given the Dec-POMDP formulation, the goal is to find joint-actions over a horizon H such that the expected cumulative cost, over a time horizon H, is minimized.

An agent may not know the action taken and the observation generated at another agent. An agent may decide to communicate with another agent, and these decisions to communicate are embedded into the joint-action vector u_k. The communication among the agents incurs a cost (as discussed in Section III), which is embedded in the global cost function $C(x_k, u_k)$. The local observations allow each agent to infer, with some uncertainty, what states actually occurred. This uncertainty is represented by the *local belief-state*, which is the *a posteriori* distribution of the underlying state given the history of local observations and local actions made by that agent, including the information gathered from other agents. Just as in centralized POMDPs, in the decentralized case the local belief-state will be used as "feedback" information that is needed for controlling the system. In other words, we seek an optimal joint-policy that depends only on the local belief-states.

B. Objective and Optimal Policy

The problem is to minimize the cumulative cost over horizon H, given by $\mathrm{E}\left[\sum_{k=0}^{H-1} C(x_k, u_k)\right]$. In the centralized case (if it was a POMDP problem), this objective function can be written in terms of "global" belief-states:

$$J(b_0) = \mathrm{E}\left[\sum_{k=0}^{H-1} c(b_k, u_k) \middle| b_0\right],$$

where $c(b,u) = \int C(x,u)b(x)\,dx$, b_k is the "global" belief-state, i.e., the posterior distribution at time k, and $\mathrm{E}[\cdot|b_0]$ represents conditional expectation given the initial belief-state b_0 at time $k = 0$. The goal is to pick the joint-actions over time so that the objective function is minimized. In general, the actions chosen for agents at each time should be allowed to depend on the entire history of observations and actions up to that time. However, if an optimal choice of such a sequence of actions exists, then there is an optimal choice of actions that depends only on "belief-state feedback." Hence, if we ignore the decentralized nature of the problem, what we seek is an optimal *policy*, which maps the belief-state at each time to the joint-action tuple at that time. The optimal policy is characterized by ***Bellman's principle*** [12], according to which the optimal action at time k is

$$\pi^*(b_0) = \arg\min_u \left\{c(b_0, u) + \mathrm{E}\left[J^*(b_1)|b_0, u\right]\right\},$$

where b_0 is the initial belief-state, b_1 is the random next belief-state, and $\mathrm{E}[J^*(b_1)|b_0, u]$ is the expected future cost of action u, which is also called the *expected cost-to-go* (ECTG). We assume a long horizon, which makes the dependence on the horizon of the ECTG negligible, and the optimal policy stationary.

Let us define the *Q-value* of taking action u at belief-state b as follows:

$$Q(b, u) = c(b, u) + \mathrm{E}\left[J^*(b')|b, u\right], \qquad (1)$$

where b' is the random next belief-state. Therefore, the optimal policy is given by

$$\pi^*(b_0) = \arg\min_u Q(b_0, u).$$

In the decentralized case, we do not have access to the "global" belief-state. Instead, every agent maintains a *local* belief-state, which may vary from agent to agent (because the observation histories differ from agent to agent). Our approach is for each agent to compute its own local action as follows. The agent i computes, at time k,

$$\pi^i(b_k^i) = \arg\min_u Q(b_k^i, u), \qquad (2)$$

where b_k^i is the local belief-state at agent i at time k. In other words, an agent i computes a joint-action by taking into account only its local belief-state. After the computation of the joint-action, agent i implements its local component. Our approach maintains the *lookahead* (non-myopic) property that is a common theme among POMDP

solution approaches, allowing us to account for the future impact of actions in our decision making.

In practice, it is intractable to compute the Q-value. Therefore, the literature on POMDP methods has focused on approximation methods [13]. We extend one such method called *nominal belief-state optimization* (NBO) to solve our Dec-POMDP approximately. This method was introduced in [1], [2] to solve a centralized UAV guidance problem, which was posed as a POMDP. The following subsection extends the NBO method to solve the current Dec-POMDP problem.

V. NBO Approximation Method for Dec-POMDP

The computation time of the joint-communication decisions at an agent i is exponential in the number of UAVs N and the length of the horizon H, which is prohibitive. For this reason, in the NBO method, we adopt a heuristic approach as follows. With regard to the communication decisions, we let each agent optimize only its "local" communication decisions over the time horizon H. More precisely, we let an agent i optimize only the decisions on whom to communicate with (g_k^i) and what to communicate (l_k^i), along with the (global) joint-kinematic controls over the time horizon H. We let each agent optimize the joint-kinematic controls (along with local communication decisions) to induce coordination among the agents. After the computation of joint-kinematic controls, an agent implements its local component at each time-step. Therefore, we use the following objective function at agent i:

$$J(b_0^i) = \mathrm{E}\left[\sum_{k=0}^{H-1} c(b_k^i, a_k, g_k^i, l_k^i)\middle| b_0\right],$$

where b_k^i is the local belief-state at time k, a_k is the joint-action corresponding to the kinematic controls, (g_k^i, l_k^i) is the local communication decision, and $c(\cdot)$ is the local cost function that depends on the local belief-state, joint-kinematic controls, and local communication decisions.

The target motion model is given by

$$\chi_{k+1} = \mathbf{F}\chi_k + e_k,\ e_k \sim \mathcal{N}(0, \mathbf{Q}). \quad (3)$$

We use the constant velocity model to represent the kinematics of a target (see [10] or [11] for the definition of $\mathbf{F}$ and $\mathbf{Q}$). The observation corresponding to the target-state, at an agent i, $i = 1, \ldots, N$, is given by:

$$z_k^{i,\chi} = \mathbf{H}\chi_k + w_k^i,\ w_k^i \sim \mathcal{N}\left(0, \mathbf{R}\left(\chi_k, s_k^i\right)\right), \quad (4)$$

where $\mathbf{H}$ is the observation model and $\mathbf{R}(\cdot)$ is the measurement covariance matrix. In this paper, we assume that the agents generate noisy observations of the 2-D target positions.

Since we assumed Gaussian distributions, the local target belief state can be expressed (or approximated) as $b_k^{i,\chi}(\chi) = \mathcal{N}(\chi - \xi_k^i, \mathbf{P}_k^i)$, where ξ_k^i and $\mathbf{P}_k^i$ evolve according to the Kalman filter (or information filter) equations given the local observations and the observations received from other agents. The agent i, when it decides to communicate with agent j, sends its local target-observation generated at time $k - a$ ($a \in \{0, \ldots, L\}$) to agent j. At agent i, the local track state variables ξ_k^i and $\mathbf{P}_k^i$ evolve according to the Kalman filter equations given the locally generated target-observations and the target-observations (time-stamped) received from other agents.

In the NBO method, the objective function at agent i is approximated as follows:

$$J(b_0^i) \approx \sum_{k=0}^{H-1} c(\hat{b}_k^i, a_k, g_k^i, l_k^i), \quad (5)$$

where $\hat{b}_1^i, \hat{b}_2^i, \ldots, \hat{b}_{H-1}^i$ is a *nominal* local belief-state sequence. The nominal (local) target belief-state sequence can be identified with the nominal local tracks $(\hat{\xi}_k^i, \hat{\mathbf{P}}_k^i)$, which are obtained from the Kalman filter equations with exactly zero-noise (*nominal* noise) sequence as follows:

$$\begin{aligned} \hat{b}_k^{i,\chi}(\chi) &= \mathcal{N}\left(\chi - \hat{\xi}_k^i, \hat{\mathbf{P}}_k^i\right), \\ \hat{\xi}_{k+1}^i &= \mathbf{F}\hat{\xi}_k^i, \end{aligned} \quad (6)$$

and the evolution of $\hat{\mathbf{P}}_k^i$ is described in Algorithm 1, where s_{k+1}^i evolves according to the UAV kinematic equations (defined in the next subsection) given the control commands.

In the NBO method, each agent computes the cost of a communication link, which is proportional to the distance between the corresponding UAVs, by evolving the locations of other

Algorithm 1 Information Filter for NBO

if $g_k^i \neq 0$ **then** ▷ (if i decides to communicate with g_k^i)
  if $l_k^i = k$ **then** ▷ (if i decides to send current observation to g_k^i)
    $\hat{\mathbf{P}}_{k+1|k}^i = \mathbf{F}\hat{\mathbf{P}}_k^i\mathbf{F}^{\mathrm{T}} + \mathbf{Q}$
    $V \leftarrow (\hat{\mathbf{P}}_{k+1|k}^i)^{-1}$
    $V \leftarrow \left[V + \mathbf{H}^{\mathrm{T}}\left[\mathbf{R}\left(\hat{\xi}_{k+1}^i, s_{k+1}^{g_k^i}\right)\right]^{-1}\mathbf{H}\right]$
  else ▷ (if i decides to send an observation from the recent past to g_k^i)
    Rollback the information filter at i to time-step l_k^i and re-run the information filter algorithm with the assumption that the observation from agent g_k^i was available at time-step l_k^i, and update $\hat{\mathbf{P}}_k^i$, and then do the following
    $\hat{\mathbf{P}}_{k+1|k}^i = \mathbf{F}\hat{\mathbf{P}}_k^i\mathbf{F}^{\mathrm{T}} + \mathbf{Q}$
    $V \leftarrow (\hat{\mathbf{P}}_{k+1|k}^i)^{-1}$
  end if
end if
▷ (updating with the locally generated observation)
$V \leftarrow \left[V + \mathbf{H}^{\mathrm{T}}\left[\mathbf{R}\left(\hat{\xi}_{k+1}^i, s_{k+1}^i\right)\right]^{-1}\mathbf{H}\right]$
$\hat{\mathbf{P}}_{k+1}^i = V^{-1}$

UAVs according to the locally computed joint-kinematic controls. Given the communication decision (g_k^i, l_k^i) at agent i, we compute $\hat{\mathbf{P}}_{\mathbf{k}}^{\mathbf{i}}$ by assuming that the agent i receives an observation from agent g_k^i generated at time l_k^i. This idea is captured in the equations presented in Algorithm 1.

The NBO cost function, which includes the mean-squared error between the tracks and the targets and the cost of communication, can be written as

$$c(\hat{b}_k^i, a_k, g_k^i, l_k^i) = \mathrm{Tr}\,\hat{\mathbf{P}}_{k+1}^i + \beta\alpha\hat{d}_k^{i,g_k^i}, \qquad (7)$$

where β is a scaling factor, α is a given proportionality constant, and $\hat{d}_k^{i,g_k^i}$ is the distance between agents i and g_k^i computed at agent i (obtained by evolving the locations of other UAVs with the locally computed joint-kinematic controls). Therefore, the cumulative cost function at agent i is given by (with truncated horizon [1], [2])

$$J_H(b_0^i) = \sum_{k=0}^{H-1}\left(\mathrm{Tr}\,\hat{\mathbf{P}}_{k+1}^i + \beta\alpha\hat{d}_k^{i,g_k^i}\right). \qquad (8)$$

Here, we adopt an approach called "receding horizon control," according to which we optimize the action sequence for H time steps at the current time-step and implement only the action corresponding to the current time-step and again optimize the action sequence for H time-steps in the next time-step.

VI. UAV Kinematics

This subsection defines the UAV kinematic model, which was represented by the function ψ in subsection IV-A. The state of the ith UAV at time k is given by $s_k^i = (p_k^i, q_k^i, V_k^i, \theta_k^i)$, where (p_k^i, q_k^i) represents the position coordinates, V_k^i represents the speed, and θ_k^i represents the heading angle. The kinematic control action for UAV i is given by $a_k^i = (f_k^i, \phi_k^i)$, where f_k^i is the forward acceleration and ϕ_k^i is the bank angle of the UAV. The kinematic equations of the UAV motion [2] are as follows:

1) speed update:

$$V_{k+1}^i = \left[V_k^i + f_k^i T\right]_{V_{\min}}^{V_{\max}},$$

where $[v]_{V_{\min}}^{V_{\max}} = \max\{V_{\min}, \min(V_{\max}, v)\}$,

2) heading angle update:

$$\theta_{k+1}^i = \theta_k^i + (gT\tan(\phi_k^i)/V_k^i),$$

3) location update:

$$p_{k+1}^i = p_k^i + V_k^i T\cos(\theta_k^i),$$
$$q_{k+1}^i = q_k^i + V_k^i T\sin(\theta_k^i),$$

where $V_{\min}$ and $V_{\max}$ are the minimum and the maximum limits on the speed of the UAVs, g is the acceleration due to gravity, and T is the length of the time-step.

VII. Simulation Results

We implement our approach in MATLAB, where we use the command *fmincon* (an optimization tool in MATLAB) to solve the optimization problem (8). The length of the time horizon H is set to six time-steps for all simulations in this paper. The target measurement error, i.e., w_k in (4) is distributed according to the normal distribution $\mathcal{N}(0, \mathbf{R}_k(\chi_k, s_k^i))$, where $\mathbf{R}_k$ reflects 10% range uncertainty and 0.01π radian angular uncertainty. For the purpose of simulations, we set the value of α in (8) equal to 0.01. In the simulations, the trajectory of a target is represented by a sequence of small circles and the trajectory of a UAV is represented by a curve joining the arrows that point toward the heading direction of the UAV.

We define the following performance metrics: 1) *average target-location error* and 2) *average communication cost*. The *average target-location error* is computed as follows. At every time-step, we compute the errors (squared distance) between the actual target location and the estimated target location from each UAV and we find the average of these errors (from each UAV's target location estimate). We summate these average errors over the simulation run-time; the mean of these average errors (from each run) is called the *average target-location error*, which is computed at the end of the simulation. The *average communication cost* is computed as follows. At every time-step, based on who (UAV) is communicating with whom (UAV), we summate the costs of these communications by computing the distance between the corresponding pairs of UAVs, and we call the output of this summation CommCost-per-step. At every step of the simulation run-time, we compute the CommCost-per-step, and the mean of these CommCost-per-steps (from each run) is called the *average communication cost*, which is computed at the end of the simulation.

We simulate a scenario with two UAVs and two targets as shown in Figures 1. In Figure 1, the two targets start at the bottom, and as time progresses, the right target moves toward the north-east and the left target moves toward the north-west. In this scenario, the value of β in (8) is set equal to 1. It is evident from Figure 1 that the UAVs coordinate with each other, with the aid of communication, to track the targets that are moving away from each other. This figure shows one UAV following right target and the other UAV following the left target, which demonstrates the coordination among the UAVs in maximizing the coverage of targets. To provide insight into the computational complexity, we compute the average time it takes to compute the control commands for each UAV in MATLAB. The average computational time is 5.2 sec with $H = 6$ on a lab computer (Intel Core i7-860 Quad-Core Processor with 8MB Cache and 2.80 GHz speed). This computation time can be greatly reduced on a better processor and by further optimizing the code (e.g., by performing parallel computations). In summary, our approach induces coordination among the UAVs even with restricted communication (as in Section III) and with relatively low computational time. Figures 2 and 3 depict the simulation of the above scenario with $\beta = 50$ and $\beta = 100$ in (8) respectively.

Next, we conduct an empirical study to evaluate the affect of β on the performance with respect to *average target-location error* and *average communication cost*. We simulate the above scenario (with two UAVs and two targets) for 50 Mote-Carlo runs for $\beta = 1$, $\beta = 50$, and $\beta = 100$ in (8). For each β and in each Mote-Carlo run, we compute the *average target-location error* and *average communication cost*. Figure 4 shows the plots of the cumulative frequency of *average target-location error*s for each value of β. This plot demonstrates that the performance with respect to *average target-location error* degrades as the value of β increases. Figure 5 shows the plot of cumulative frequency of *average communication cost*s for each value of β. This plot demonstrates that the performance with respect to *average communication cost* improves as the value of β increases. Therefore, we can use β as a tuning parameter to trade off between the performances with respect to *average target-location error* and *average communication cost*, which is evident from Figures 4 and 5.

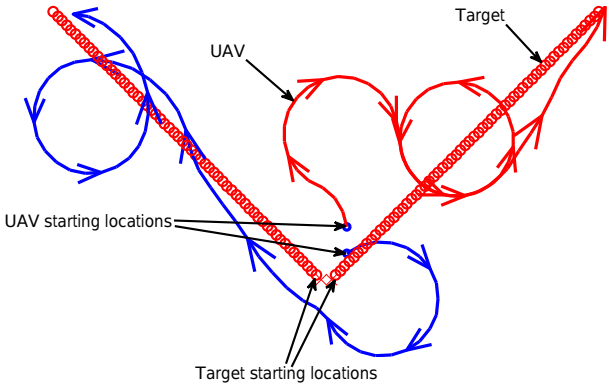

Fig. 1. Two UAVs tracking two targets; $\beta = 1$

VIII. Concluding Remarks

In this study, we designed a decentralized guidance control method for UAVs tracking multiple targets based on the theory of *decentralized partially observable Markov decision process* (Dec-POMDP). We extended a POMDP approximation

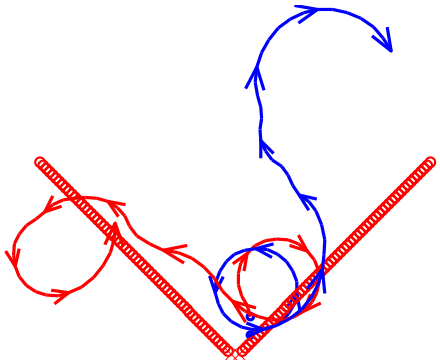

Fig. 2. Two UAVs tracking two targets; $\beta = 50$

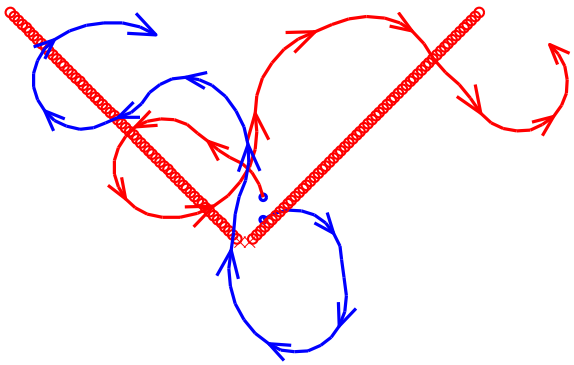

Fig. 3. Two UAVs tracking two targets; $\beta = 100$

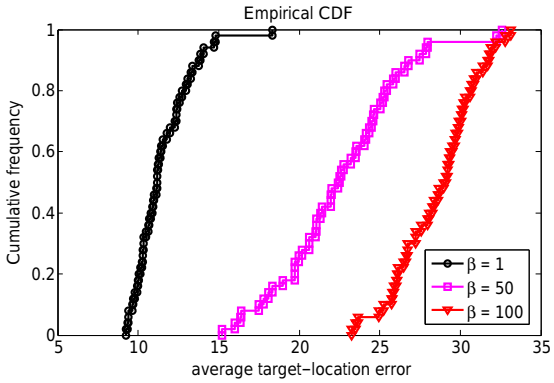

Fig. 4. Performance with respect to *average target-location error* for various values of β

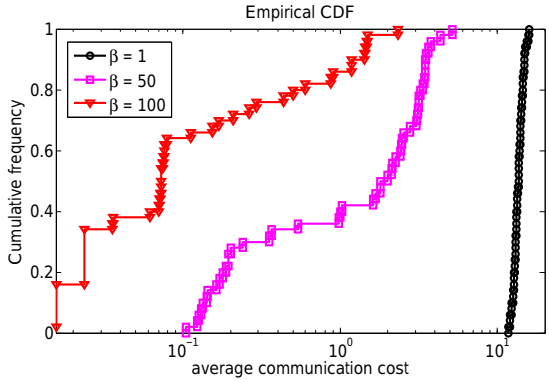

Fig. 5. Performance with respect to *average communication cost* for various values of β

method called *nominal belief-state optimization* (NBO) to solve our decentralized guidance control problem, which we posed as a Dec-POMDP. Although the communication between the UAVs was restricted, the NBO method achieved coordination among the UAVs, as evident from the simulation results in Section VII. The simulation results also demonstrate that the parameter β can be used as a tuning parameter to trade off between the performances with respect to the *average target-location error* and the *average communication cost*.

REFERENCES

[1] S. A. Miller, Z. A. Harris, and E. K. P. Chong, "A POMDP framework for coordinated guidance of autonomous UAVs for multitarget tracking," *EURASIP J. Adv. Signal Process.*, vol. 2009, 2009.

[2] S. Ragi and E. K. P. Chong, "Dynamic UAV path planning for multitargte tracking," in *Proc. 2012 American Control Conference*, Montreal, QC, Canada, Jun. 2012, pp. 3845–3850.

[3] H. Min, F. Sun, and F. Niu, "Decentralized UAV formation tracking flight control using gyroscopic force," in *Proc. International Conference on Computational Intelligence for Measurement Systems and Applications (CISMA 2009)*, Hong Kong, May 2009, pp. 91–96.

[4] H. Rezaee and F. Abdollahi, "A synchronization strategy for three dimensional decentralized formation control of unmanned aircrafts," in *Proc. 37th Annual Conference of the IEEE Industrial Electronics Society*, Melbourne, Australia, Nov. 2011, pp. 462–467.

[5] Y. Yang, A. A. Minai, and M. M. Polycarpou, "Decentralized cooperative search by networked UAVs in an uncertain environment," in *Proc. 2004 American Control Conference*, Boston, MA, Jun. 2004, pp. 5558–5563.

[6] A. Richards and J. How, "Decentralized model predictive control of cooperating UAVs," in *Proc. 43rd IEEE Conference on Decision and Control*, Atlantis, Paradise Island, Bahamas, Dec. 2004, pp. 4286–4291.

[7] M. Alighanbari and J. P. How, "Decentralized task assignment for unmanned aerial vehicles," in *Proc. 44th IEEE Conference on Decision and Control, and European Control Conference 2005*, Seville, Spain, Dec. 2005, pp. 5668–5673.

[8] A. Ghaffarkhah and Y. Mostofi, "Communication-aware motion planning in mobile networks," *IEEE Trans. Autom. Control*, vol. 56, pp. 2478–2485, 2011.
[9] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein, "The complexity of decentralized control of markov decision processes," *Math. Oper. Res.*, vol. 27, pp. 819–840, 2002.
[10] S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*. Boston, MA: Artech House, 1999.
[11] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. New York: Wiley-Interscience, 2001.
[12] R. Bellman, *Dynamic Programming*. Princeton, NJ: Princeton Univ. Press, 1957.
[13] E. K. P. Chong, C. Kreucher, and A. O. H. III, "Partially observable markov decision process approximations for adaptive sensing," *Disc. Event Dyn. Sys.*, vol. 19, no. 3, pp. 377–422, 2009.