J Intell Robot Syst (2014) 73:811–822
DOI 10.1007/s10846-013-9904-9

Decentralized Guidance Control of UAVs with Explicit Optimization of Communication

Shankarachary Ragi · Edwin K. P. Chong

Received: 24 August 2013 / Accepted: 12 September 2013 / Published online: 27 September 2013

Abstract We design a decentralized guidance control method for autonomous unmanned aerial vehicles (UAVs) tracking multiple targets. We formulate this guidance control problem as a decentralized partially observable Markov decision process (Dec-POMDP). As in the case of partially observable Markov decision process (POMDP), it is intractable to solve a Dec-POMDP exactly. So, we extend a POMDP approximation method called *nominal belief-state optimization* (NBO) to solve our control problem posed as a Dec-POMDP. We incorporate the cost of communication into the objective function of the Dec-POMDP, i.e., we explicitly optimize the communication among the UAVs at the network level along with the kinematic control commands for the UAVs. We measure the performance of our 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. The NBO method induces coordination among the UAVs even though the system is decentralized. To demonstrate the effectiveness of this coordination, we compare our Dec-POMDP approach with a greedy approach (a noncooperative approach), where the UAVs do not communicate with each other and each UAV optimizes only its local kinematic controls. Furthermore, we compare the performance of our approach of optimizing the communication between the UAVs with a fixed communication scheme—where only the UAV kinematic controls are optimized with an underlying fixed (non-optimized) communication scheme.

Keywords Decentralized partially observable Markov decision process · UAV guidance control · Decentralized control · Target tracking

S. Ragi (✉) · E. K. P. Chong
Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523-1373, USA
e-mail: shankar.ragi@colostate.edu

E. K. P. Chong
e-mail: edwin.chong@colostate.edu

1 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] was 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) involves a nonzero 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). Each UAV has a fixed communication range, i.e., two UAVs can directly (with no relays) communicate with each other only if the distance between them is less than a constant value. The communication network among the UAVs is dynamic in the sense that a communication link between two UAVs forms when they are within each other's communication range, and breaks otherwise. At each time instance, any two UAVs can communicate with each other if there exists a path between them through the network. The communication among the UAVs involves a nonzero cost, and we explicitly optimize the communication decisions (at the network level) of each UAV (with whom to communicate and when to communicate) along with the kinematic controls. In this approach, 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 the joint kinematic control commands and the communication decisions. 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) at the network level. In general, solving a Dec-POMDP exactly is intractable. Instead, we extend a centralized POMDP (*partially observable Markov decision process*) approximation method to solve the Dec-POMDP. We seek a solution to our guidance control problem that is implementable in real-time (a typical requirement). 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 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.

2 System and Problem Description

Targets The targets are ground-based vehicles and are assumed to be moving in 2-D (our framework easily extends to 3-D, with a concomitant additional computational cost).

Unmanned Aerial Vehicles For the purpose of this study, we assume that the UAVs fly at a constant altitude, i.e., the UAVs move in 2-D (again, our framework easily extends to 3-D). 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 Each UAV has communication capabilities, i.e., each UAV can transmit and receive information to/from other UAVs, subject to certain constraints. 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).

3 Communication Between Agents

In this study, the UAVs communicate with each other over a network. Each UAV has a fixed communication range of d_c, i.e., each UAV can transmit or receive information to or from another UAV only if the distance between them is less than d_c. We define the notion of a communication link between a pair of UAVs as follows. At each time step, there exists a communication link between a pair of UAVs if the distance between them is less than d_c, and there exists no link otherwise. A pair of UAVs, not within the communication range of each other, can still exchange information by relaying the information through the network. The network is dynamic, i.e., the links between the UAVs form and break with time. At each time step, a UAV can send or receive information to or from another UAV if there exists a *path* (a sequence of links connecting UAVs) between them through the network. We 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 $k+1$. To account for the cost of communicating information over the network, we assign a numerical value to each communication link, and we refer to this value as the *cost* of the link. For simplicity, the value of the cost of the link between a pair of agents, if it exists, 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 such that there exists a link between them (i.e., $d_k^{ij} < d_c$), then the cost of the link is αd_k^{ij}, where α is a given proportionality constant. The cost of a path through the network is the sum of the costs of the links that form the path. At any time step, a pair of UAVs cannot communicate with each other if there exists no path between them through the network.

The communication among the agents is restricted in the following ways:

– At each 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 the 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.

4 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.

4.1 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 the following three sub-systems 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 $T_k^i = (\xi_k^i, \mathbf{P}_k^i)$ is the state of the tracking algorithm at agent i, where ξ_k^i is the posterior mean vector and $\mathbf{P}_k^i$ 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 $g_k^i \in \mathcal{I} \cup \{0\} \setminus \{i\}$ is the ID of the agent with whom agent i will communicate 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 for $i = 1, \ldots, N$ 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 probability density 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 density. 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 for $i = 1, \ldots, N$ evolves according to the kinematic equations given the actions/control commands, i.e., $s_{k+1}^i = \psi(s_k^i, a_k^i)$ (see below for an explicit definition of ψ), where a_k^i represents the local kinematic controls (forward acceleration and bank angle). The state of the ith UAV at time k is given by $s_k^i = \left(p_k^i, q_k^i, V_k^i, \theta_k^i\right)$, 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:

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

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

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

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

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

The target state evolves according to

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

We use the constant velocity model to represent the kinematics of a target (see [10] or [11] for the definition of $\mathbf{F}$ and $\mathbf{Q}$). 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 probability density 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 density. The observations at agent i, represented by z_k^i, includes the observation of the 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, at each time step the agent has access only to a random measurement of the target state that is a function of the locations of the targets and the agent. Specifically, the observation corresponding to the target state, at agent i for $i = 1, \ldots, N$, is given by

$$z_k^{i,\chi} = \mathbf{H}\chi_k + w_k^i,\ w_k^i \sim \mathcal{N}\left(0, \mathbf{R}\left(\chi_k, s_k^i\right)\right), \qquad (2)$$

where $\mathbf{H}$ is the observation model and $\mathbf{R}(\cdot)$ is the measurement covariance matrix that depends on the locations of the targets and the agent. We assume that the agents generate noisy observations of the 2-D target positions. An agent i for $i = 1, \ldots, N$ 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 probability density 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 3), 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* density 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.

4.2 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 were a POMDP problem), this objective function can be written in terms of "global" belief-states as follows:

$$J(b_0) = \mathrm{E}\left[\sum_{k=0}^{H-1} c(b_k, u_k)\middle| b_0\right], \qquad (3)$$

where $c(b, u) = \int C(x, u)b(x)\,dx$, b_k is the "global" belief-state, i.e., the posterior density at time k, and $\mathrm{E}\left[\cdot|b_0\right]$ 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, ignoring for the time being 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:

$$Q(b, u) = c(b, u) + \mathrm{E}\left[J^*(b')|b, u\right], \tag{4}$$

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,

$$\pi^i(b^i_k) = \arg\min_u Q(b^i_k, u), \tag{5}$$

where b^i_k 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 exactly. 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. In our description of the NBO method, as an approximation method for the function Q in Eq. 5, we will approximate the function J in Eq. 3 instead. This approximation to J is then optimized to approximate J^*, which is a part of function Q as indicated in Eq. 4.

5 NBO Approximation Method for Dec-POMDP

The computation complexity 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 its own communication decisions g^i_k (with whom to communicate) and l^i_k (what to communicate), 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:

$$J(b^i_0) = \mathrm{E}\left[\sum_{k=0}^{H-1} c(b^i_k, a_k, g^i_k, l^i_k)\middle| b_0\right],$$

where b^i_k is the local belief-state at time k, a_k is the joint action corresponding to the kinematic controls, (g^i_k, l^i_k) 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.

We assume that b_0 is Gaussian. Therefore, by Eqs. 1 and 2, the local target belief state can be expressed (or approximated) as $b^{i,\chi}_k(\chi) = \mathcal{N}\left(\chi - \xi^i_k, \mathbf{P}^i_k\right)$, where ξ^i_k and $\mathbf{P}^i_k$ 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 $k - a$ ($a \in \{0, \dots, L\}$) 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:

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

where $\hat{b}_1^i, \hat{b}_2^i, \dots, \hat{b}_{H-1}^i$ 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:

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

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)
$V \leftarrow \left[V + \mathbf{H}^{\mathrm{T}}\left[\mathbf{R}\left(\hat{\xi}_{k+1}^i, s_{k+1}^i\right)\right]^{-1}\mathbf{H}\right]$
$\hat{\mathbf{P}}_{k+1}^i = V^{-1}$
```

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 earlier in Subsection 4.1) given the control commands.

In the NBO method, each agent i computes the path costs originating from i over the time horizon H. These path costs depend on the locations of the neighboring agents over the time horizon H. Each agent implements its control actions independently (i.e., there is no central controller), which means that an agent has no access to the future locations of its neighboring agents. Therefore, each agent computes the path costs over the horizon H by evolving (only for predictive-lookahead purposes) the locations of other agents according to the locally computed joint kinematic controls.

Given the communication decision (g_k^i, l_k^i) at agent i, we compute $\hat{\mathbf{P}}_k^i$ by assuming that the agent i receives an observation from agent g_k^i generated at time l_k^i. The rationale for doing this is as follows. When an agent i sends an observation from time step l_k^i to an agent j, only agent j perceives the benefit of this action as the measurement fusion happens at agent j. The agent i, however, does not directly perceive the benefit from this measurement fusion at j because our system is decentralized. Therefore, the agent i computes the benefit (from measurement fusion) of sending the observation from time l_k^i to agent j approximately by assuming that agent j sends the observation generated at time l_k^i to agent i and that the measurement fusion happens at agent 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

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

where β is a scaling factor, α is a given proportionality constant, and $\hat{D}_k^{i,g_k^i}$ is the length of the shortest path between agents i and g_k^i (obtained from Dijkstra's algorithm, where the length of each path is the path cost defined earlier) if there exists at least one path between i and g_k^i or $\hat{D}_k^{i,g_k^i}$ is a (relatively) large and constant value otherwise (i.e., when there exists no path). The function $\hat{D}_k^{i,g_k^i}$ is evaluated at agent i 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])

$$J_H(b_0^i) = \sum_{k=0}^{H-1} \left(\mathrm{Tr}\, \hat{\mathbf{P}}_{k+1}^i + \beta\alpha \hat{D}_k^{i,g_k^i} \right). \tag{6}$$

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.

6 Simulation Results

We implement our approach in MATLAB, where we use the command *fmincon* (an optimization tool in MATLAB) to minimize the objective function in Eq. 6. The length of the time horizon H is set to six time steps. The target measurement error, i.e., w_k^i in Eq. 2 is distributed according to the normal distribution $\mathcal{N}\left(0, \mathbf{R}_k\left(\chi_k, s_k^i\right)\right)$, where $\mathbf{R}_k$ reflects 10% range standard deviation and 0.01π radian angular standard deviation. For the purpose of simulations, we set the value of α in Eq. 6 to be 0.01. In Figs. 1, 2, 3, 6, and 7, 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 squared distance (squared error) 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 runtime; the mean of these average errors (from each Monte Carlo run) is called the *average target-location error*. The *average communication cost* is computed as follows.

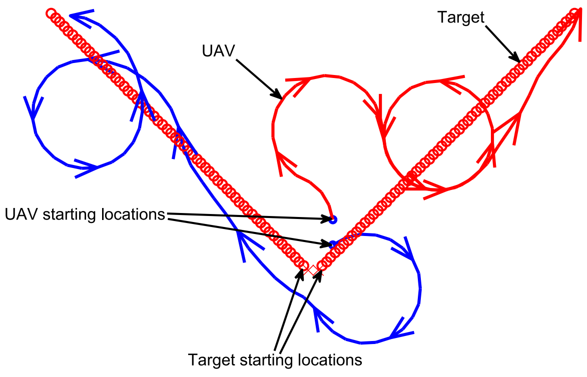

Fig. 1 Two UAVs tracking two targets; $\beta = 1$

At every time step, based on who is communicating with whom, we summate (over each pair of communicating UAVs) the costs of these communications, and we call the output of this summation CommCost-per-step. At every step of the simulation runtime, we compute the CommCost-per-step, and the mean of these CommCost-per-steps (from each Monte Carlo run) is called the *average communication cost*.

We simulate a scenario with two UAVs and two targets as shown in Fig. 1. In Fig. 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 Eq. 6 was set to be 1. It is evident from Fig. 1 that the UAVs coordinate with each other, with the aid of communication, to

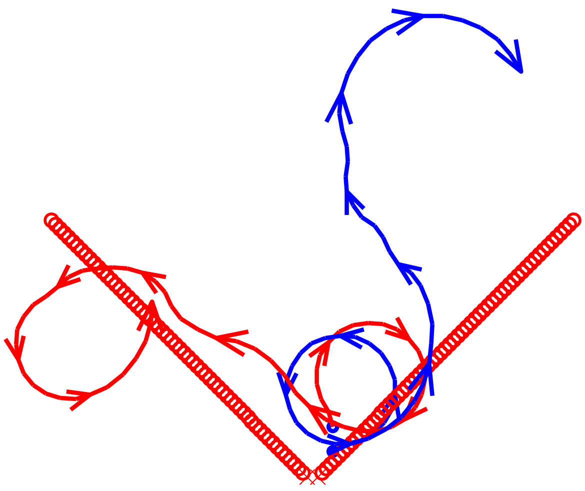

Fig. 2 Two UAVs tracking two targets; $\beta = 50$

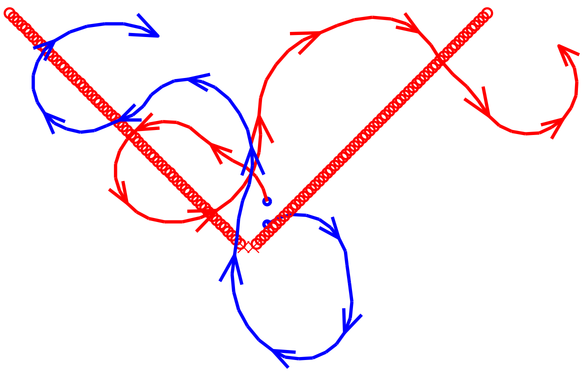

Fig. 3 Two UAVs tracking two targets; $\beta = 100$

track the targets that are moving away from each other (of course, this motion is not known to the UAVs beforehand). This figure shows one UAV following the right target and the other UAV following the left target, which demonstrates the coordination among the UAVs in maximizing the coverage of targets. In summary, our approach induces coordination among the UAVs even with restricted communication (as in Section 3) and with relatively low computational time. Figures 2 and 3 depict the simulation of the above scenario with $\beta = 50$ and $\beta = 100$ in Eq. 6 respectively.

To provide insight into the computational complexity, we evaluate the average time it takes to compute the control commands for each UAV in MATLAB. The average computational time is 5.2 s 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).

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 Eq. 6. 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, as expected. 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, again as expected. 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 Figs. 4 and 5.

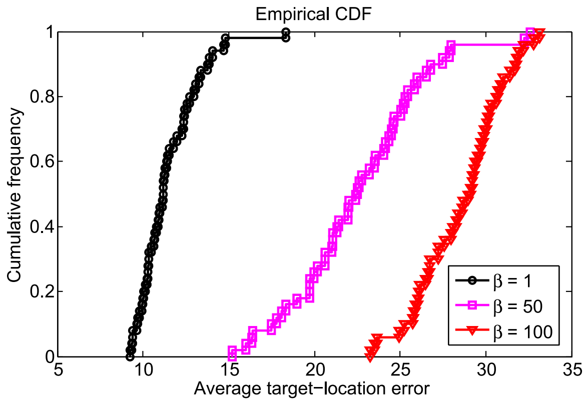

Fig. 4 Performance with respect to *average target-location error* for various values of β

6.1 Dec-POMDP Approach vs. Greedy Approach

In this subsection, we compare the performance of our Dec-POMDP approach with a greedy approach (defined as follows). In the greedy approach, the UAVs do not communicate with each

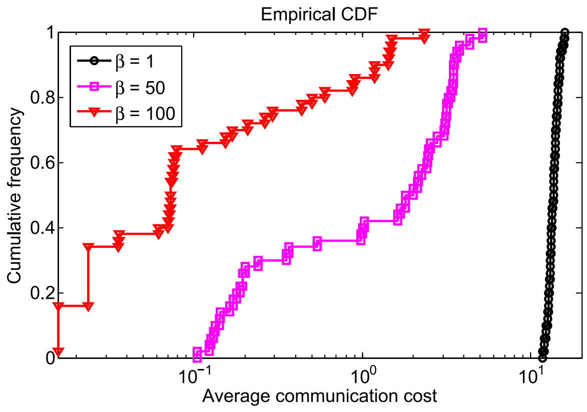

Fig. 5 Performance with respect to *average communication cost* for various values of β

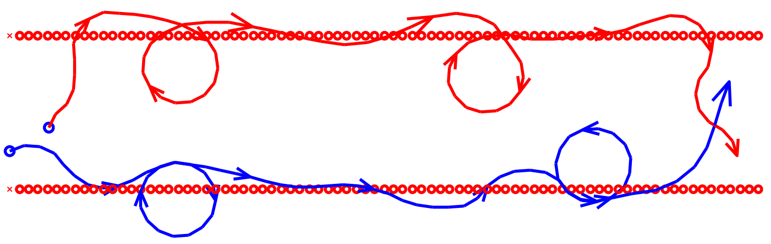
Fig. 6 Two UAVs tracking two targets via Dec-POMDP approach

other, and each UAV optimizes only its own kinematic controls over the time horizon ($H = 6$). Clearly, our Dec-POMDP approach induces cooperation among the UAVs by letting each UAV optimize the joint kinematic controls (along with local communication decisions), and the greedy approach is non-cooperative in the sense that each UAV behaves in a non-cooperative manner by optimizing only its own kinematic controls. We implement these two approaches for a scenario with two UAVs and two targets. Figures 6 and 7 depict the behavior of the UAVs in the Dec-POMDP and greedy approaches respectively. We run this simulation for 200 Monte Carlo runs, and compare the performances (with respect to *average target-location error*) of these two approaches, as depicted in Fig. 8. Figure 8 demonstrates that our Dec-POMDP approach significantly outperforms the greedy approach. It is unsurprising that the Dec-POMDP approach outperforms the greedy approach. The purpose of Fig. 8 is to show the quantitative difference in performance between these approaches. Specifically, the Dec-POMDP approach results in average target location-error values that are approximately three times smaller compared to that of the greedy approach.

6.2 Optimized Communication Scheme vs. Fixed Communication Scheme

In our Dec-POMDP approach, we explicitly optimize the communication among the UAVs along with the joint kinematic controls. We now compare the performance of this approach with an approach where the communication scheme is fixed, i.e., not optimized. We call this new scheme the "fixed communication scheme." An overview of this scheme is given below.

Fixed Communication Scheme In this scheme, we do not optimize the communication decisions, i.e., we only optimize the joint kinematic controls at each UAV. Each UAV, at each time step, sends the observations generated in the current time step to its closest neighbor among all the neighbors in its communication range. Since communication decisions cannot be controlled anymore, the action space of the Dec-POMDP includes only the kinematic controls of the UAVs. Moreover, the cost function that is used does not penalize the communication (i.e., $\beta = 0$ in Eq. 6). However, we incorporate the benefits (from measurement fusion) of this fixed communication scheme while computing trace objective in the NBO cost function in Eq. 6 (with $\beta = 0$).

We implement these two approaches for a scenario with three UAVs and two targets. Figure 9 depicts the performance comparison of the fixed communication scheme and the optimized communication scheme with respect to both the performance measures—*average target-location error* and *average communication cost*. It is evident from Fig. 9 that the performance of the fixed communication scheme is slightly better than the optimized communication scheme with respect to

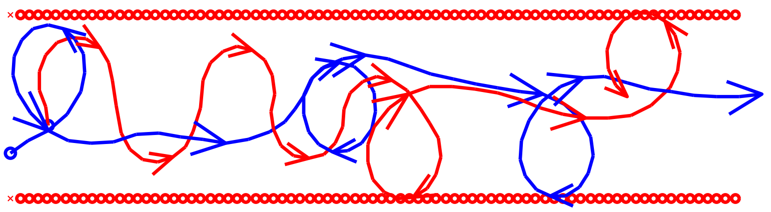
Fig. 7 Two UAVs tracking two targets via Greedy approach

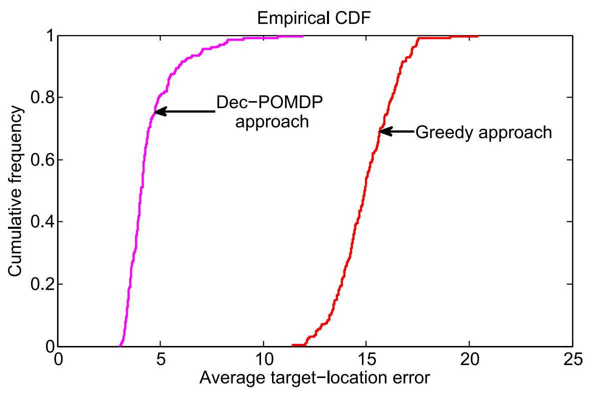

Fig. 8 Dec-POMDP approach vs. greedy approach

average target-location error, which is to be expected because the fixed communication scheme does not use the communication cost in its cost function and is focused on minimizing only location error. However, the optimized communication scheme significantly outperforms the fixed communication scheme in terms of *average communication cost*.

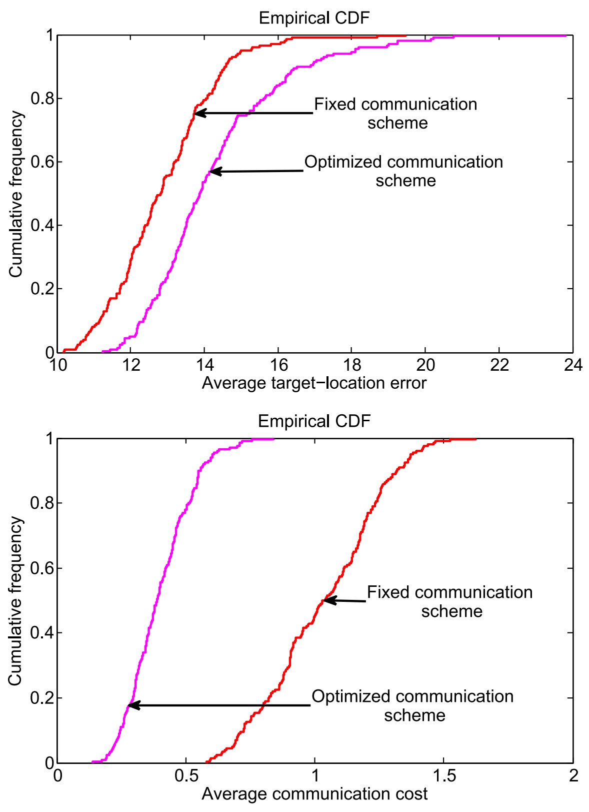

Fig. 9 Fixed communication scheme vs. optimized communication scheme

7 Concluding Remarks and Future Work

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 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 6. The 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*. Specifically, when we increased the value of β (which is the weight on the cost of communication) the performance with respect to *average target-location error* (or tracking error) degraded while the performance with respect to *average communication cost* improved. In other words, the UAVs communicate less often at the expense of degraded tracking performance when we increase the value of β.

We then compared the performance of our Dec-POMDP approach with a greedy approach. In the Dec-POMDP approach, each UAV optimizes joint actions and implements its local component. In contrast, in the greedy approach, each UAV only optimizes its own kinematic controls. We showed quantitatively how much our Dec-POMDP approach outperforms (with respect to *average target-location error*) the greedy approach.

In our Dec-POMDP approach, we optimized the communication decisions along with the UAV kinematic controls by including the communication decisions in the Dec-POMDP action space. To demonstrate the effectiveness of this approach, we compared the performance of this optimized communication scheme with a different scheme called the fixed communication scheme. In this fixed communication scheme, each UAV communicates with its closest neighbor among all the neighbors in its communication range. Our results demonstrated that the performance of the fixed communication scheme is slightly better than the optimized communication scheme with respect to the *average target-location error*, which was expected because the fixed communication scheme does not use the communication cost in its cost function and is focused on minimizing only location error. However, the optimized communication scheme significantly outperformed the fixed communication scheme in terms of *average communication cost*.

In our study, we did not incorporate communication delays, i.e., we assumed 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 $k+1$. It would be interesting to see how communication delays affect the performance of these decentralized UAV guidance algorithms.

References

1. Miller, S.A., Harris, Z.A., Chong, E.K.P.: A POMDP framework for coordinated guidance of autonomous UAVs for multitarget tracking. EURASIP J. Adv. Signal Process. **2009**, 724597 (2009)
2. Ragi, S., Chong, E.K.P.: Dynamic UAV path planning for multitargte tracking. In: Proc. 2012 American Control Conference, pp. 3845–3850. Montreal, QC (2012)
3. Min, H., Sun, F., Niu, F.: Decentralized UAV formation tracking flight control using gyroscopic force. In: Proc. International Conference on Computational Intelligence for Measurement Systems and Applications (CISMA 2009), pp. 91–96. Hong Kong (2009)
4. Rezaee, H., Abdollahi, F.: A synchronization strategy for three dimensional decentralized formation control of unmanned aircrafts. In: Proc. 37th Annual Conference of the IEEE Industrial Electronics Society, pp. 462–467. Melbourne, Australia (2011)
5. Yang, Y., Minai, A.A., Polycarpou, M.M.: Decentralized cooperative search by networked UAVs in an uncertain environment. In: Proc. 2004 American Control Conference, pp. 5558–5563. Boston, MA (2004)
6. Richards, A., How, J.: Decentralized model predictive control of cooperating UAVs. In: Proc. 43rd IEEE Conference on Decision and Control, pp. 4286–4291. Atlantis, Paradise Island (2004)
7. Alighanbari, M., How, J.P.: Decentralized task assignment for unmanned aerial vehicles. In: Proc. 44th IEEE Conference on Decision and Control, and European Control Conference 2005, pp. 5668–5673. Seville, Spain (2005)
8. Ghaffarkhah, A., Mostofi, Y.: Communication-aware motion planning in mobile networks. IEEE Trans. Autom. Control **56**, 2478–2485 (2011)
9. Bernstein, D.S., Givan, R., Immerman, N., Zilberstein, S.: The complexity of decentralized control of markov decision processes. Math. Oper. Res. **27**, 819–840 (2002)
10. Bar-Shalom, Y., Li, X.R., Kirubarajan, T.: Estimation with Applications to Tracking and Navigation. Wiley-Interscience, New York (2001)
11. Blackman, S., Popoli, R.: Design and Analysis of Modern Tracking Systems. Artech House, Boston (1999)
12. Bellman, R.: Dynamic Programming. Princeton University Press, Princeton (1957)
13. Chong, E.K.P., Kreucher, C., III, A.O.H.: Partially observable markov decision process approximations for adaptive sensing. Disc. Event Dyn. Sys. **19**(3), 377–422 (2009)