

# Solving Sensor Network Coverage Problems by Distributed Asynchronous Actor-Critic Methods \*

Paris Pennesi, <sup>†</sup> Ioannis Ch. Paschalidis <sup>‡</sup>

**Abstract**—Multi-robots systems exploiting sensor network capabilities can be successfully employed to cope with several tasks, including coverage, surveillance, target tracking, and foraging, in partially known environments subject to dynamical changes. In this paper we define a *reward collection problem*, where both the positions and the values of the rewards change with time. We propose a distributed actor-critic method to solve the problem, establish its convergence, and demonstrate its adaptation capabilities. Our analysis leverages ideas from actor-critic methods and consensus algorithms.

**Index Terms**—Actor-critic methods, consensus algorithms, dynamic programming, sensor networks.

## I. INTRODUCTION

THE role of communication constraints in distributed control and the emergence of collective behaviors in multi-agent systems acting by local rules has been the subject of several papers in recent years. In particular, the availability of Wireless Sensor NETWORKS (WSNETs) offers the opportunity to approach practical problems in a different way, sensing the environment and collecting data dynamically. This opportunity, however, presents the challenge to develop novel distributed control algorithms to exploit the information provided by the sensor network architecture.

Multi-robots systems exploiting sensor network capabilities can be successfully employed to cope with several tasks, notably, coverage, surveillance, target tracking, and foraging, in partially known environments subject to dynamical changes. In this paper we abstract and generalize those problems defining a *reward collection problem*, where both the positions and the values of the rewards change with time. In this problem a robot swarm explores an unknown and changing environment looking for the target points and their relative rewards. Any robot has limited sensing capabilities and is able to communicate locally with other robots. Problems of this type, but adapted to coverage control have also been studied in [1–5] (see also references therein).

To solve the reward collection problem we propose a *distributed multi-agent actor-critic algorithm*. The algorithm extends the algorithm of [6] into a distributed multi-agent setting. Our algorithm allows any agent to have its own control policy which can be updated using information provided by

its neighbors (e.g., the ones into a certain communication range). This updating follows a consensus-like algorithm; such algorithms and their analysis go back to [7], and have more recently been considered in [8, 9]. We show that under suitable conditions the robots reach consensus and converge to the optimal control policy.

The rest of the paper is organized as follows. Sec. II states the reward collection problem. Sec. III introduces a class of parametric control policies. Sec. IV describes the distributed actor-critic method. Sec. IV-A establishes its convergence. Sec. V reports numerical results for the reward collection method. Conclusions are in Sec. VI.

**Notational Conventions:** Throughout the paper all vectors are assumed to be column vectors. We use lower case boldface letters to denote vectors and for economy of space we write  $\mathbf{x} = (x_1, \dots, x_R)$  for the column vector  $\mathbf{x}$ .  $\mathbf{x}'$  denotes the transpose of  $\mathbf{x}$ ,  $\|\mathbf{x}\|$  the Euclidean norm of  $\mathbf{x}$ ,  $\mathbf{0}$  the vector of all zeroes,  $\mathbf{e}$  the vector of all ones, and  $\mathbf{e}_i$  the  $i$ th unit vector. We use upper case boldface letters to denote matrices and write  $\mathbf{I}$  for the identity matrix and  $\mathbf{0}$  for the matrix of all zeroes. The notation  $\mathbf{A} > \mathbf{B}$  represents an element-wise comparison.

## II. PROBLEM FORMULATION

We consider a 2-dimensional *mission space*,  $\mathcal{S} \subset \mathbb{R}^2$ , in which there is a set of  $M$  *target points* indexed by  $i = 1, \dots, M$  whose positions are indicated, at time  $k$ , by  $\mathbf{m}_k^i \in \mathcal{S}$ ,  $i = 1, \dots, M$ . These positions may change over time; we assume that  $\{\mathbf{m}_k^i; k = 1, 2, \dots\}$  is a stationary stochastic process for each  $i$ . To each *target point*  $i$ ,  $i = 1, \dots, M$ , we associate a *reward*  $R_k^i \in \mathbb{R}_+$ .

The mission space is to be explored by  $N$  agents indexed by  $j = 1, \dots, N$ , whose positions at time  $k$  are indicated by  $\mathbf{x}_k^j \in \mathcal{S}$ . To each agent  $j$  we associate a *capacity*  $C_k^j \in \mathbb{R}_+$ . When an agent “visits” a target point it collects a reward which depends on the available reward at the target point and the capacity of the agent. Every visit has also the effect of depleting a part of the agent’s capacity. We assume that agents start their exploration of the mission space from the origin  $\mathbf{0} \in \mathcal{S}$ . Every agent  $j$  navigates in  $\mathcal{S}$  and, from time to time, returns to the origin which has the effect of replenishing the agent’s capacity to its initial value  $C_0^j$ .

In particular, the dynamics of the sequences  $\{R_k^i, i = 1, \dots, M\}$  and  $\{C_k^j, j = 1, \dots, N\}$  are described by the following equations. For all  $i = 1, \dots, M$

$$R_{k+1}^i = \begin{cases} \max(R_k^i - C_k^j, 0), & \text{if } \exists j \in 1, \dots, N \\ & \text{such that } \mathbf{m}_k^i = \mathbf{x}_k^j, \\ R_k^i + w_k & \text{otherwise,} \end{cases} \quad (1)$$

\* Research partially supported by the NSF under grants DMI-0330171, DMI-0300359, CNS-0435312, ECS-0426453, EFRI-0735974, and by the DOE under grant DE-FG52-06NA27490.

<sup>†</sup> Dipartimento di Ingegneria Informatica, Gestionale e dell’Automazione, Università Politecnica delle Marche, Via Brecce Bianche, Ancona 60100, ITALY, e-mail: p.pennesi@diiga.univpm.it.

<sup>‡</sup> Corresponding author. Center for Information & Systems Eng., Dept. of Manufacturing Eng., and Dept. of Electrical and Computer Eng., Boston University, 15 St. Mary’s St., Brookline, MA 02446, e-mail: yannis@bu.edu, url: http://ionia.bu.edu/.

where  $\{w_k\}$  is a sequence of i.i.d. random variables, and for all  $j = 1, \dots, N$

$$C_{k+1}^j = \begin{cases} \max(C_k^j - R_k^i, 0), & \text{if } \exists i \in 1, \dots, M \\ & \text{such that } \mathbf{m}_k^i = \mathbf{x}_k^j, \\ C_0^j & \text{if } \mathbf{x}_k^j = \mathbf{0}, \\ C_k^j + g_k & \text{otherwise,} \end{cases} \quad (2)$$

where  $\{g_k\}$  is also a sequence of i.i.d. random variables. The sequence of rewards,  $\Phi_k^j$ , collected by each agent  $j$  over time is characterized as

$$\Phi_k^j = \begin{cases} \min(R_k^i, C_k^j), & \text{if } \exists i \in 1, \dots, M \\ & \text{such that } \mathbf{m}_k^i = \mathbf{x}_k^j, \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

The scenario we are considering is completed by the sensor model of the agents. The agents can *sense* the reward located at the *target points* depending on the “intensity” of the target and their distance from it. We assume that agents can identify (or know in advance) the number of targets present in the mission space and can pick up a “signal” from each target. Specifically, for all  $i = 1, \dots, M$

$$s_k^i(\mathbf{y}) = \frac{R_k^i}{2\pi \det(\Sigma^i)^{1/2}} e^{-(\mathbf{y} - \mathbf{m}_k^i)' (\Sigma^i)^{-1} (\mathbf{y} - \mathbf{m}_k^i) / 2} \quad (4)$$

is the signal associated with the *target point*  $i$  and measured, at time  $k$ , at the position  $\mathbf{y} \in \mathcal{S}$  of the *mission space*. In (4)  $\Sigma^i$  is a positive definite weight matrix associated with the *target point*  $i$ ,  $\det(\Sigma^i)$  denotes its determinant, and prime denotes transpose.

Given this setup we are interested in a policy that guides the agents in the mission space so that we maximize the long-term average total reward collected given by

$$\lim_{k \rightarrow \infty} \frac{1}{k} \sum_{\tau=1}^k \sum_{j=1}^N \Phi_\tau^j. \quad (5)$$

### III. A PARAMETRIC CLASS OF POLICIES

To solve the problem introduced in Section II we will first discretize the mission space  $\mathcal{S}$  by superimposing a 2-dimensional grid. We assume that the positions of the target points  $\mathbf{m}_k^i$ , for all  $k$  and  $i$ , are always on this grid, as well as, all agents move on the grid. The position of the  $j$ th agent,  $j = 1, \dots, N$ , at time  $k$ , is represented by the two-dimensional vector  $\mathbf{x}_k^j$ .

We are interested in a policy for each agent that is based on its current position, the signals it measures from all target points, and, potentially, any information it receives from other agents. At time  $k$  the  $j$ th agent can choose a control action  $\mathbf{u}_k^j \in \mathcal{U} = \{(1, 0), (-1, 0), (0, 0), (0, 1), (0, -1)\}$  to move from the position  $\mathbf{x}_k^j$  to the position  $\mathbf{x}_{k+1}^j = \mathbf{x}_k^j + \mathbf{u}_k^j$ . We adopt the convention that if  $\mathbf{x}_k^j$  is on the border of the grid then the control set  $\mathcal{U}$  contains only the feasible control actions.

To derive an optimal policy, each agent can resort to Markov decision theory and use dynamic programming techniques. Clearly, this is not practical for large instances of the problem. Instead, we focus on a parametrized class of policies. We consider the following class of *Randomized*

*Stationary Policies (RSP)* where each agent  $j$  at time  $k$  and position  $\mathbf{x}_k^j$  selects control  $\mathbf{u} \in \mathcal{U}$  with probability

$$\mu_{\theta^j}(\mathbf{u} | \mathbf{x}_k^j) = \frac{\exp(\xi_{\theta^j}(\mathbf{u}, \mathbf{x}_k^j))}{\sum_{\mathbf{v} \in \mathcal{U}} \exp(\xi_{\theta^j}(\mathbf{v}, \mathbf{x}_k^j))}, \quad (6)$$

where

$$\xi_{\theta^j}(\mathbf{u}, \mathbf{x}) = \sum_{i=1}^M \theta_i^j C_k^j s_k^i(\mathbf{x} + \mathbf{u}) + \theta_0^j e^{-\|\mathbf{x} + \mathbf{u}\|}, \quad (7)$$

and where  $\theta^j = (\theta_0^j, \dots, \theta_M^j)$ . The vector  $\theta^j$  is a parameter vector maintained by agent  $j$  which affects agent's  $j$  policy. Notice that the structure of the policy favors control actions that lead to targets emitting stronger signals. When the agent's capacity or the available rewards are low then we favor control actions that tend to bring the agent closer to the origin.

We are interested in finding the best policy within this class, namely, selecting for each agent a parameter vector  $\theta^j$  so that the objective (5) gets maximized. To that end, we will leverage the actor-critic algorithm of [6] which can obtain a local maximum of the policy's performance with respect to  $\theta^j$ . However, the algorithm of [6] is *centralized* and can be run by one agent to optimize its own  $\theta$ . We seek to develop a *distributed multi-agent* version of the algorithm that allows multiple agents to “collaborate” and reach a consensus in a  $\theta$  value that can be used by all of them. As we will see, this has advantages as exploration of the mission space is done by many agents in parallel and the information is fused in finding an optimal  $\theta$ .

### IV. THE DISTRIBUTED ACTOR-CRITIC METHOD

In this section we develop a distributed multi-agent version of the actor-critic algorithm. We start by introducing a general Markov decision problem and reviewing some results from the single-agent version of the actor-critic algorithm presented in [6].

Consider a Markov decision process with finite state space  $\mathcal{X}$ , and finite action space  $\mathcal{U}$ . Let  $c : \mathcal{X} \times \mathcal{U} \mapsto \mathbb{R}$  be a given reward function. Let  $\{\mu_\theta, \theta \in \mathbb{R}^n\}$  be a set of *randomized stationary policies (RSPs)*, parameterized in  $\theta$ . In particular,  $\mu_\theta(\mathbf{u} | \mathbf{x})$  denotes the probability of taking the action  $\mathbf{u}$  given the state  $\mathbf{x}$ , under the *RSP*  $\theta$ . The sequence of states  $\{\mathbf{x}_k\}$  and the sequence of state-control pairs  $\{(\mathbf{x}_k, \mathbf{u}_k)\}$  of the Markov decision process, generated by an *RSP*  $\theta$ , form Markov chains for every  $\theta$ .

For our purposes, we make the same assumptions as in [6] for the family of allowed *RSPs*. In particular, the following assumption is in effect.

#### Assumption A

- For every  $\mathbf{x} \in \mathcal{X}$ ,  $\mathbf{u} \in \mathcal{U}$ , and  $\theta \in \mathbb{R}^n$ , we have  $\mu_\theta(\mathbf{u} | \mathbf{x}) > 0$ .
- For every  $(\mathbf{x}, \mathbf{u}) \in \mathcal{X} \times \mathcal{U}$ , the mapping  $\theta \mapsto \mu_\theta(\mathbf{u} | \mathbf{x})$  is twice differentiable. Furthermore, the function  $\nabla_\theta \ln \mu_\theta(\mathbf{u} | \mathbf{x})$  is bounded and has a bounded first derivative for any fixed  $\mathbf{x}$  and  $\mathbf{u}$ .
- For every  $\theta \in \mathbb{R}^n$  the Markov chains  $\{\mathbf{x}_k\}$  and  $\{(\mathbf{x}_k, \mathbf{u}_k)\}$  are irreducible and aperiodic, with

stationary probabilities  $\pi_{\theta}(\mathbf{x})$  and  $\eta_{\theta}(\mathbf{x}, \mathbf{u}) = \pi_{\theta}(\mathbf{x})\mu_{\theta}(\mathbf{u}|\mathbf{x})$ , respectively.

- (d) There is a positive integer  $I$ , state  $\mathbf{x}^* \in \mathcal{X}$ , and  $\epsilon_0 > 0$  such that for all  $\theta_1, \dots, \theta_I$  it follows  $\sum_{k=1}^I (\mathbf{P}[\theta_1] \cdots \mathbf{P}[\theta_I])_{\mathbf{x}\mathbf{x}^*} \geq \epsilon_0$  for all  $\mathbf{x} \in \mathcal{X}$ , where  $\mathbf{P}[\theta]$  denotes the transition probability for the Markov chain  $\{\mathbf{x}_k\}$  under the RSP  $\theta$ .

This is identical to Assumption 2.1 in [6]; we note that for the setting and policy introduced in Sec. III parts (a), (b) are automatically satisfied and in part (d) we can take  $\mathbf{x}^* = \mathbf{0}$ . This assumption is satisfied if part (c) of the assumption holds, and the policy probabilities are all bounded away from zero uniformly in  $\theta$ .

We are interested in finding the parameter vector  $\theta$  that maximizes the average reward function  $\bar{\alpha} : \mathbb{R}^n \mapsto \mathbb{R}$ , given by:

$$\bar{\alpha}(\theta) = \sum_{\mathbf{x} \in \mathcal{X}, \mathbf{u} \in \mathcal{U}} c(\mathbf{x}, \mathbf{u}) \eta_{\theta}(\mathbf{x}, \mathbf{u}). \quad (8)$$

For each  $\theta \in \mathbb{R}^n$  let us now define a differential reward function  $V_{\theta} : \mathcal{X} \mapsto \mathbb{R}$ , as solution of the following Poisson equation:

$$\bar{\alpha}(\theta) + V_{\theta}(\mathbf{x}) = \sum_{\mathbf{u} \in \mathcal{U}} \mu_{\theta}(\mathbf{u}|\mathbf{x}) \left[ c(\mathbf{x}, \mathbf{u}) + \sum_{\mathbf{y} \in \mathcal{X}} p(\mathbf{y}|\mathbf{x}, \mathbf{u}) V_{\theta}(\mathbf{y}) \right], \quad (9)$$

where  $p(\mathbf{y}|\mathbf{x}, \mathbf{u})$  is the probability that the next state is  $\mathbf{y}$  given that the current state is  $\mathbf{x}$  and action  $\mathbf{u}$  is taken. We also need to define the  $Q$ -value function  $Q_{\theta} : \mathcal{X} \times \mathcal{U} \mapsto \mathbb{R}$  by:

$$Q_{\theta}(\mathbf{x}, \mathbf{u}) = c(\mathbf{x}, \mathbf{u}) - \bar{\alpha}(\theta) + \sum_{\mathbf{y} \in \mathcal{X}} p(\mathbf{y}|\mathbf{x}, \mathbf{u}) V_{\theta}(\mathbf{y}). \quad (10)$$

The following result is from [10].

**Theorem IV.1 (Average Reward Gradient)** *We have*

$$\nabla \bar{\alpha}(\theta) = \sum_{\mathbf{x} \in \mathcal{X}, \mathbf{u} \in \mathcal{U}} \eta_{\theta}(\mathbf{x}, \mathbf{u}) Q_{\theta}(\mathbf{x}, \mathbf{u}) \psi_{\theta}(\mathbf{x}, \mathbf{u}), \quad (11)$$

where

$$\psi_{\theta}(\mathbf{x}, \mathbf{u}) = \nabla_{\theta} \ln \mu_{\theta}(\mathbf{u}|\mathbf{x}). \quad (12)$$

We write  $\psi_{\theta}(\mathbf{x}, \mathbf{u}) = (\psi_{\theta,1}(\mathbf{x}, \mathbf{u}), \dots, \psi_{\theta,1}(\mathbf{x}, \mathbf{u}))$ .

The actor-critic algorithm ([6]) works with a parametrization of the  $Q$ -function in terms of a vector  $\mathbf{r} = (r_1, \dots, r_m) \in \mathbb{R}^m$ :

$$Q_{\theta}^{\mathbf{r}}(\mathbf{x}, \mathbf{u}) = \sum_{l=1}^m r_m \phi_{\theta,l}(\mathbf{x}, \mathbf{u}).$$

For reasons explained in [6], a typical choice for the features  $\phi_{\theta,l}(\mathbf{x}, \mathbf{u})$  is to set  $m = n + 1$ ,  $\phi_{\theta,l}(\mathbf{x}, \mathbf{u}) = \psi_{\theta,l}(\mathbf{x}, \mathbf{u})$  for  $l = 1, \dots, n$ , and fix  $\phi_{\theta,n+1}(\mathbf{x}, \mathbf{u})$  to the constant function that is everywhere equal to one except at  $\mathbf{x} = \mathbf{0}$  where  $\phi_{\theta,n+1}(\mathbf{0}, \mathbf{u}) = 0$  for all  $\mathbf{u}$ . The algorithm interchanges estimation of the parameter  $\mathbf{r}$  based on observations from a sample path of the Markov process (critic) with gradient-based updates on  $\theta$  using an estimate of  $\nabla \bar{\alpha}(\theta)$  from the same sample path (actor).

Next we develop our *distributed multi-agent* version of the actor-critic algorithm. We have  $N$  agents; the  $j$ th agent samples the Markov chains, under an RSP  $\theta^j$ . The  $j$ th agent performs the critic phase of the algorithm on the basis of its own observations. For the actor phase, each agent  $j$  uses its own estimate of  $\nabla \bar{\alpha}(\theta^j)$  and any information it receives from other agents.

Now, we are ready to state the iteration equations of the distributed actor-critic algorithm. We present two versions of the algorithm, the  $TD(1)$  and the  $TD(\lambda)$ , which are different only in the critic part. The critic part of the algorithm for each agent  $j$  is the same as in [6]. Specifically, consider agent  $j$  and let  $\alpha_k^j \in \mathbb{R}$  the average reward estimate at time  $k$ ,  $\mathbf{r}_k^j \in \mathbb{R}^m$  the estimate of parameter vector  $\mathbf{r}$  at time  $k$ , and  $\mathbf{Z}_k^j \in \mathbb{R}^m$  the estimate of Sutton's eligibility trace at time  $k$ . Let also  $\theta_k^j \in \mathbb{R}^n$  be the parameter of the actor for the  $j$ th agent at time  $k$ . The updates take place at states-action pairs visited by a single sample path (potentially generated by a simulation) of the Markov process. Let  $(\mathbf{X}_k^j, \mathbf{U}_k^j)$  be the state-action pair sampled at time  $k$  from the  $j$ th agent. The critic update equations for the  $j$ th agent are as follows:

$$\alpha_{k+1}^j = \alpha_k^j + \gamma_k^j (c(\mathbf{X}_{k+1}^j, \mathbf{U}_{k+1}^j) - \alpha_k^j), \quad (13)$$

$$\mathbf{r}_{k+1}^j = \mathbf{r}_k^j + \gamma_k^j d_k^j \mathbf{Z}_k^j, \quad (14)$$

$$d_k^j = c(\mathbf{X}_k^j, \mathbf{U}_k^j) - \alpha_k^j + \mathbf{r}_k^{j'} \phi_{\theta_k^j}(\mathbf{X}_{k+1}^j, \mathbf{U}_{k+1}^j) - \mathbf{r}_k^{j'} \phi_{\theta_k^j}(\mathbf{X}_k^j, \mathbf{U}_k^j), \quad (15)$$

where in the  $TD(1)$  case

$$\mathbf{Z}_{k+1}^j = \begin{cases} \mathbf{Z}_k^j + \phi_{\theta_k^j}(\mathbf{X}_{k+1}^j, \mathbf{U}_{k+1}^j), & \text{if } \mathbf{X}_{k+1}^j \neq \mathbf{x}^* \\ \phi_{\theta_k^j}(\mathbf{X}_{k+1}^j, \mathbf{U}_{k+1}^j) & \text{otherwise,} \end{cases} \quad (16)$$

and in the  $TD(\lambda)$  case, for  $0 < \lambda < 1$ ,

$$\mathbf{Z}_{k+1}^j = \lambda \mathbf{Z}_k^j + \phi_{\theta_k^j}(\mathbf{X}_{k+1}^j, \mathbf{U}_{k+1}^j). \quad (17)$$

In the above  $\gamma_k^j$  is a positive step-size parameter, and  $\mathbf{x}^*$  is a special state that the Markov process visits infinitely often. (For the application we considered in Section III we can take  $\mathbf{x}^* = (0, 0)$ . We want to remark that the special state  $\mathbf{x}^*$  plays a role only in the  $TD(1)$  case and it is not necessary to return at  $\mathbf{x}^*$  in the  $TD(\lambda)$  version of the algorithm.)

To present the actor update equations we introduce some notation. Agents communicate with each other and exchange their parameter vectors  $\theta_k^j$ . Naturally, some of these messages may not be received by agents who happen to be outside the communication range of the transmitting agent. We denote by  $T^{ij}$  the set of times that agent  $i$  receives a message from agent  $j$ . We will assume that  $T^{ij}$  is either empty or infinite, that is, either  $j$  never sends messages to  $i$  or, if it does, it gets close to  $i$  often enough for its message to be received by  $i$ . Suppose  $i$  receives a message sent from  $j$  at time  $k$ . Then, we let  $t^{ij}(k)$  the time this message was sent, namely,  $i$  receives  $\theta_{t^{ij}(k)}^j$ . The actor update iteration is as follows:

$$\theta_{k+1}^i = \mathbf{A}_k^{ii} \theta_k^i + \sum_{j \neq i} \mathbf{A}_k^{ij} \theta_{t^{ij}(k)}^j + \beta_k^i \Gamma(\mathbf{r}_k^i) \cdot \mathbf{r}_k^{i'} \phi_{\theta_k^i}(\mathbf{X}_{k+1}^i, \mathbf{U}_{k+1}^i) \psi_{\theta_k^i}(\mathbf{X}_{k+1}^i, \mathbf{U}_{k+1}^i), \quad (18)$$

where  $\Gamma(\cdot)$  is a scalar that controls the step-size  $\beta_k^i$  on the basis of the value of  $\mathbf{r}_k^i$ . Furthermore,  $\mathbf{A}_k^{ij} = \text{diag}(a_{k,1}^{ij}, \dots, a_{k,n}^{ij})$ ,  $\mathbf{A}_k^{ij} \geq \mathbf{0}$ ,  $\sum_{j=1}^N a_{k,l}^{ij} = 1$ , for all  $i, l, k$ , and  $\mathbf{A}_k^{ij} = \mathbf{0}$  if  $k \notin T^{ij}$  for all  $i \neq j$ . Notice that we set  $\mathbf{A}_k^{ij} = \mathbf{0}$  if agent  $i$  does not receive the actor parameter from agent  $j$ ; otherwise it receives such information and combines it with its own actor parameter as in (18). In the next section we prove the convergence of the algorithm to a stationary point of the average reward function.

#### A. Convergence

To show the convergence of the distributed actor-critic algorithm we rely on the proof of the convergence of the centralized actor-critic algorithm in [6] and the work on distributed stochastic gradient methods in [11].

Before we proceed with the main result we introduce some additional assumptions. The following assumption on information exchange between agents will be crucial in establishing that the agents reach consensus on their  $\theta^i$ 's. To state the assumption, we introduce the directed graph  $\mathcal{G} = (\mathcal{N}, \mathcal{E})$  with nodes  $\mathcal{N} = \{1, \dots, N\}$  corresponding to the set of agents. An edge  $(j, i) \in \mathcal{E}$  if and only if  $T^{ij}$  is infinite. The graph  $\mathcal{G}$  represents the communication pattern between agents.

#### Assumption B

- There is a directed path in  $\mathcal{G}$  from every node to every other node.
- there exists a positive constant  $\gamma$  such that
  - $\mathbf{A}_k^{ii} \geq \gamma \mathbf{I}$ ,  $\forall i, k$ ;
  - $\mathbf{A}_k^{ij} \geq \gamma \mathbf{I}$ ,  $\forall i, j$  and  $k \in T^{ij}$ .
- The time between consecutive transmissions of  $\theta_k^j$  from agent  $j$  to agent  $i$  is bounded by some  $B \geq 0$  for all  $(j, i) \in \mathcal{E}$ .
- Communication delays are bounded by some  $B_0 \geq 0$ .

We will also adopt Assumption 3.3 on stepsizes from [6], which we repeat for completeness.

#### Assumption C

For all agents  $i$  the stepsizes  $\gamma_k^i, \beta_k^i$  and the function  $\Gamma(\cdot)$  appearing in (18) satisfy

- $\gamma_k^i, \beta_k^i$  are deterministic, nonincreasing, and satisfy

$$\begin{aligned} \sum_k \beta_k^i &= \infty, & \sum_k (\beta_k^i)^2 &< \infty, \\ \sum_k \gamma_k^i &= \infty, & \sum_k (\gamma_k^i)^2 &< \infty, \\ \sum_k (\beta_k^i / \gamma_k^i)^d &< \infty, \end{aligned}$$

for some  $d > 0$ .

- For some positive constants  $C_1 < C_2$ :

$$\begin{aligned} \|\mathbf{r}\| \Gamma(\mathbf{r}) &\in [C_1, C_2], & \forall \mathbf{r} \in \mathbb{R}^m, \\ \|\Gamma(\mathbf{r}) - \Gamma(\mathbf{s})\| &\leq \frac{C_2 \|\mathbf{r} - \mathbf{s}\|}{1 + \|\mathbf{r}\| + \|\mathbf{s}\|}, & \forall \mathbf{r}, \mathbf{s} \in \mathbb{R}^m. \end{aligned}$$

**Theorem IV.2 (Distributed Actor-Critic)** Under Assumptions A–C the sequences  $\{\theta_k^i\}$  generated by each agent  $i$  according to the distributed actor-critic algorithm satisfy:

- in the TD(1) case

$$\liminf_{k \rightarrow \infty} \|\nabla \bar{\alpha}(\theta_k^i)\| = 0, \quad \forall i \text{ w.p.1};$$

- in the TD( $\lambda$ ) case, for each  $\epsilon > 0$ , there exists  $\lambda$  such that

$$\liminf_{k \rightarrow \infty} \|\nabla \bar{\alpha}(\theta_k^i)\| < \epsilon, \quad \forall i, \text{ w.p.1}.$$

*Proof:* Consider agent  $i$  and as in [6, Sec. 6] define

$$\begin{aligned} \mathbf{H}_{\theta^i}(\mathbf{x}, \mathbf{u}) &= \psi_{\theta^i}(\mathbf{x}, \mathbf{u}) \phi'_{\theta^i}(\mathbf{x}, \mathbf{u}), \\ \bar{\mathbf{H}}(\theta^i) &= \sum_{\mathbf{x}, \mathbf{u}} \eta_{\theta^i}(\mathbf{x}, \mathbf{u}) \psi_{\theta^i}(\mathbf{x}, \mathbf{u}) \phi'_{\theta^i}(\mathbf{x}, \mathbf{u}). \end{aligned}$$

Let  $\bar{\mathbf{r}}^i(\theta^i)$  be the limit of the critic parameter  $\mathbf{r}_k^i$  if the policy parameter  $\theta^i$  was held fixed. The critic part of the algorithm is identical with the single agent version in [6], hence this limit exists. The actor update can be written as follows:

$$\begin{aligned} \theta_{k+1}^i &= \mathbf{A}_k^{ii} \theta_k^i + \sum_{j \neq i} \mathbf{A}_k^{ij} \theta_{t^{ij}(k)}^j \\ &\quad + \beta_k^i \mathbf{H}_{\theta_k^i}(\mathbf{X}_{k+1}^i, \mathbf{U}_{k+1}^i) (\mathbf{r}_k^i \Gamma(\mathbf{r}_k^i)) \\ &= \mathbf{A}_k^{ii} \theta_k^i + \sum_{j \neq i} \mathbf{A}_k^{ij} \theta_{t^{ij}(k)}^j \\ &\quad + \beta_k^i \bar{\mathbf{H}}(\theta_k^i) (\bar{\mathbf{r}}^i(\theta_k^i) \Gamma(\bar{\mathbf{r}}^i(\theta_k^i))) \\ &\quad + \beta_k^i (\mathbf{H}_{\theta_k^i}(\mathbf{X}_{k+1}^i, \mathbf{U}_{k+1}^i) - \bar{\mathbf{H}}(\theta_k^i)) (\mathbf{r}_k^i \Gamma(\mathbf{r}_k^i)) \\ &\quad + \beta_k^i \bar{\mathbf{H}}(\theta_k^i) (\mathbf{r}_k^i \Gamma(\mathbf{r}_k^i) - \bar{\mathbf{r}}^i(\theta_k^i) \Gamma(\bar{\mathbf{r}}^i(\theta_k^i))). \end{aligned} \quad (19)$$

Setting

$$\mathbf{f}^i(\theta) = \bar{\mathbf{H}}(\theta) \bar{\mathbf{r}}^i(\theta), \quad (20)$$

$$\mathbf{e}_k^{i,(1)} = (\mathbf{H}_{\theta_k^i}(\mathbf{X}_{k+1}^i, \mathbf{U}_{k+1}^i) - \bar{\mathbf{H}}(\theta_k^i)) \mathbf{r}_k^i \Gamma(\mathbf{r}_k^i), \quad (21)$$

$$\mathbf{e}_k^{i,(2)} = \bar{\mathbf{H}}(\theta_k^i) (\mathbf{r}_k^i \Gamma(\mathbf{r}_k^i) - \bar{\mathbf{r}}^i(\theta_k^i) \Gamma(\bar{\mathbf{r}}^i(\theta_k^i))), \quad (22)$$

$$\mathbf{e}_k^{i,(3)} = \mathbf{A}_k^{ii} \theta_k^i + \sum_{j \neq i} \mathbf{A}_k^{ij} \theta_{t^{ij}(k)}^j - \theta_k^i, \quad (23)$$

the actor update can be written as

$$\begin{aligned} \theta_{k+1}^i &= \theta_k^i + \mathbf{e}_k^{i,(3)} + \beta_k^i \mathbf{e}_k^{i,(1)} + \beta_k^i \mathbf{e}_k^{i,(2)} \\ &\quad + \beta_k^i \bar{\mathbf{H}}(\theta_k^i) (\bar{\mathbf{r}}^i(\theta_k^i) \Gamma(\bar{\mathbf{r}}^i(\theta_k^i))). \end{aligned} \quad (24)$$

Now, the update equation above and the error terms  $\mathbf{e}_k^{i,(1)}$ , and  $\mathbf{e}_k^{i,(2)}$  can be handled exactly as in [6, Section 6]. To establish the convergence of the actor what remains to be shown is that  $\lim_k \mathbf{e}_k^{i,(3)} = \mathbf{0}$  w.p.1.

To that end, we will use the analysis in [11]. Notice that our actor update (18) has the same form as Eq. (2.1) in [11], i.e., the form of a consensus algorithm perturbed by the gradient which can be viewed as a “noise” term. In Eq. (2.1) of [11], we will take the stepsize  $\gamma^i(n)$  to be equal to 1 and the gradient  $s^i(n)$  to be equal to  $\beta_k^i \mathbf{H}_{\theta_k^i}(\mathbf{X}_{k+1}^i, \mathbf{U}_{k+1}^i) (\mathbf{r}_k^i \Gamma(\mathbf{r}_k^i))$ . Let  $\mathbf{y}_k$  be defined as in Eqs. (2.13) and (2.14) of [11]. That is, in our setting  $\mathbf{y}_k$  is the value of  $\theta$  that all agents would agree to if at time  $k$  they switched to a new actor update iteration that has only the first two terms of (18). Define, also,

$$b_k = \sum_{i=1}^N \beta_k^i \|\mathbf{H}_{\theta_k^i}(\mathbf{X}_{k+1}^i, \mathbf{U}_{k+1}^i) (\mathbf{r}_k^i \Gamma(\mathbf{r}_k^i))\|. \quad (25)$$

Following the same reasoning as in [11] (and using Assumption B) one can establish an inequality equivalent to (A.2) in [11], namely, there exists a  $d \in [0, 1)$  such that

$$\|\mathbf{y}_k - \boldsymbol{\theta}_k^i\| \leq A \sum_{n=1}^k d^{k-n} b_n, \quad (26)$$

where  $A$  is some constant. Let now

$$\tilde{\mathbf{H}}_k = (\beta_k^1 \|\mathbf{H}_{\boldsymbol{\theta}_k^1}(\mathbf{X}_{k+1}^1, \mathbf{U}_{k+1}^1)(\mathbf{r}_k^1 \Gamma(\mathbf{r}_k^1))\|, \dots, \beta_k^N \|\mathbf{H}_{\boldsymbol{\theta}_k^N}(\mathbf{X}_{k+1}^N, \mathbf{U}_{k+1}^N)(\mathbf{r}_k^N \Gamma(\mathbf{r}_k^N))\|).$$

We have

$$\begin{aligned} \mathbf{E}[\sum_k (b_k)^2] &= \mathbf{E}[\sum_k (e^i \tilde{\mathbf{H}}_k)^2] \\ &\leq \|e\|^2 \sum_i \sum_k (\beta_k^i)^2 \mathbf{E}[\|\mathbf{H}_{\boldsymbol{\theta}_k^i}(\mathbf{X}_{k+1}^i, \mathbf{U}_{k+1}^i)(\mathbf{r}_k^i \Gamma(\mathbf{r}_k^i))\|^2] \\ &\leq NC \sum_i \sum_k (\beta_k^i)^2 \mathbf{E}[\|\mathbf{H}_{\boldsymbol{\theta}_k^i}(\mathbf{X}_{k+1}^i, \mathbf{U}_{k+1}^i)\|^2] \\ &\leq NC' \sum_i \sum_k (\beta_k^i)^2 < \infty. \end{aligned}$$

The 2nd inequality above holds for some  $C > 0$  and is due to Assumption C. For some constant  $C' > 0$ , the 3rd inequality above follows from [6, Lemma 4.3] which states that  $\mathbf{E}[\|\mathbf{H}_{\boldsymbol{\theta}_k^i}(\mathbf{X}_{k+1}^i, \mathbf{U}_{k+1}^i)\|^2]$  is bounded. The finiteness of  $\mathbf{E}[\sum_k (b_k)^2]$  follows from Assumption C(a). We conclude that  $b_k$  converges to zero almost surely. The convergence of  $b_k$  establishes that  $\mathbf{y}_k - \boldsymbol{\theta}_k^i$  converges to zero for all  $i$  almost surely which in turn implies the convergence of  $e_k^{i,(3)}$  to zero. This completes the proof. ■

## V. NUMERICAL RESULTS

In this section we present some numerical results of the application of the distributed actor-critic control scheme described in Sec. III and in Sec. IV to the reward collection problem stated in Sec. II.

Given the previous setting, the algorithm runs as follows:

- 1) *Initialization phase*: for each agent  $i$ ,  $i = 1, \dots, N$  assign, randomly, an initial position  $\mathbf{x}_0^i$  and the initial values for the parameters  $\boldsymbol{\theta}_0^i, \mathbf{r}_0^i, \mathbf{Z}_0^i$ .
- 2) Each agent  $i$  chooses the control actions  $\mathbf{u}_k^i$  according to the *RSP*  $\boldsymbol{\theta}_k^i$ .
- 3) Each agent  $i$  receives the value of the parameters  $\boldsymbol{\theta}_k^j$ ,  $j \in \Omega_k^i = \{j = 1, \dots, R \mid \|\mathbf{x}_k^i - \mathbf{x}_k^j\| \leq \delta_k\}$  from all other agents within a  $\delta_k$  range.
- 4) Each agent updates its own parameters according to the distributed actor-critic algorithm (cf. (14)–(18)) and using the information received from other agents.
- 5) iterate from 2).

Note that the parameter  $\delta_k$  is a behavioral parameter that can speed up the convergence process, in simulation, and can be interpreted as the actual maximum communication radius. It will be clear later, that the parameter  $\delta_k$  is related to the number of agents, and the right combination of number of agents and  $\delta_k$  leads to the “best” performance.

We consider a  $20 \times 20$  grid, with two *target points* as the *mission space*,  $\mathcal{S}$ . The *target points* are in positions (10, 1) and (10, 5), and their initial rewards are 2000 and 1000, respectively.

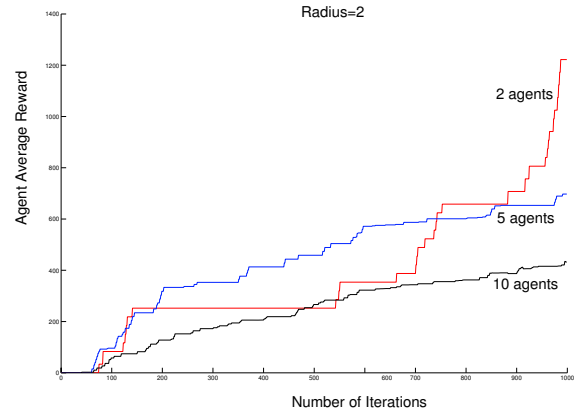


Fig. 1. Total Average Reward for three different number of agents and communication radius equal to 2.

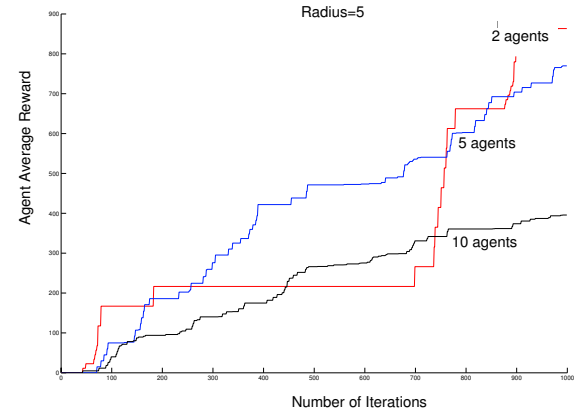


Fig. 2. Total Average Reward for three different number of agents and communication radius equal to 5.

We take as performance index the *total average reward* which measures the average reward collected by each agent, up to time  $k$ . More formally, the *total average reward* at time  $k$  is given by

$$\bar{\Phi}_k = \frac{1}{N} \sum_{\tau=1}^k \sum_{j=1}^N \Phi_{\tau}^j \quad (27)$$

where  $\Phi_{\tau}^j$  is defined in (3).

Next, we run the simulation with different number of agents and different values of the communication radius (the results are shown in Figs. 1, 2 and 3). The simulation confirms our intuition that by increasing the number of agents we achieve a faster collection of the rewards in the *mission space*  $\mathcal{S}$ , if a large communication radius is adopted. Comparing Fig. 1, Fig. 2 and Fig. 3 it is possible to understand the influence of the communication radius  $\delta$ . Increasing  $\delta$  the agents share their knowledge about the environment more often and this leads to a faster convergence towards an *optimal* choice of the parameter  $\boldsymbol{\theta}$ .

It is also important to note that, when a few agents are employed, using a large communication radius does not improve the performance of the algorithm. Our understanding is that with a large radius and few agents, the system

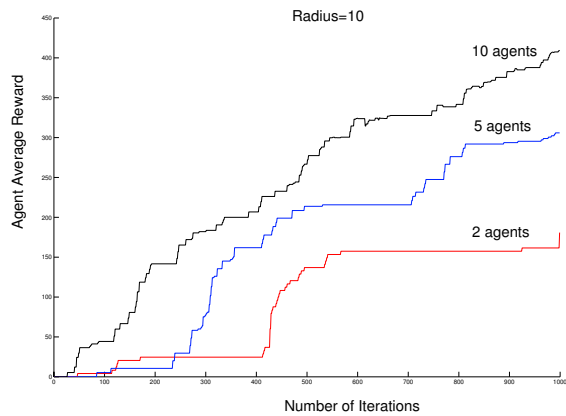


Fig. 3. Total Average Reward for three different number of agents and communication radius equal to 10.

averages the knowledge (behavior) of the agents even if their relative distance is large and they cover (sample) just a little portion of the map. However, increasing the value of the communication radius,  $\delta$ , leads to faster convergence of the algorithm towards the consensus point. This finding is illustrated in Table I.

TABLE I

FOR EACH VALUE OF THE PARAMETER  $\delta$ , THE AVERAGE AND THE VARIANCE COLUMNS SHOW, RESPECTIVELY, THE AVERAGE NUMBER OF ITERATIONS TO CONVERGE AT  $\theta^*$  AND THE CORRESPONDING VARIANCE.

$\delta$	Two Agents		Five Agents		Ten Agents	
	Average	Variance	Average	Variance	Average	Variance
1	287	58	261	55	254	54
5	143	36	138	34	126	30
10	118	25	110	24	98	21

We think that the main advantage of the algorithm is its adaptive capability at a dynamically changing environment. To highlight this feature of the algorithm we run a set of simulations where after a certain amount of time the *mission space*  $\mathcal{S}$  changes, in one time step, to the *mission space*  $\mathcal{S}'$ , where the *target points* are in positions (18, 15) and (3, 14), respectively, maintaining the same rewards of the *mission space*  $\mathcal{S}$ .

The distributed actor-critic control scheme reacts with the change of the *mission space* adapting its control parameters and, following a period of learning, the result is a new increase in the total average reward (see Fig. 4).

## VI. CONCLUSIONS

We defined a reward collection problem in an unknown and changing environment, and we approached this problem using a multi-robot system controlled by a distributed actor-critic method. We showed that the distributed algorithm guarantees consensus and convergence to an optimal parametric control policy. We performed a numerical test to investigate the relation between the number of agents and the communication radius. We observed that a large number of agents is useful when the communication radius is large for them to be able to coordinate and learn from each other, otherwise, if

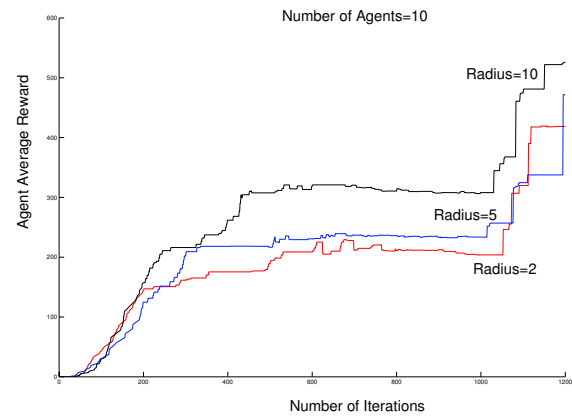


Fig. 4. Total Average Reward for three different number of agents and communication radius equal to 2. After 600 time steps the mission space switch from the mission space  $\mathcal{S}$  to the mission space  $\mathcal{S}'$ .

the number of agents is small and the communication radius is large the system averages the knowledge (behavior) of the agents even if their relative distance is large and they cover (sample) a small portion of the map.

## REFERENCES

- [1] W. Li and C. G. Cassandras, "Distributed Cooperative Coverage Control of Sensor Networks," in *Proc. of 44th IEEE Conf. on Decision and Control*, 2005, pp. 2542 – 2547.
- [2] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 2, pp. 243–255, 2004.
- [3] V. de Silva and R. Ghrist, "Coordinate-free coverage in sensor networks with controlled boundaries via homology," *International Journal of Robotics Research*, vol. 25, no. 12, pp. 1205–1222, 2006.
- [4] K. Lerman, C. Jones, A. Galstyan, and M. Matari, "Analysis of dynamic task allocation in multi-robot systems," *International Journal of Robotics Research*, vol. 25, no. 3, pp. 225–241, 2006.
- [5] B. Gerkey and M. Matari, "A formal framework for the study of task allocation in multi-robot systems," *International Journal of Robotics Research*, vol. 23, no. 9, pp. 939–954, 2004.
- [6] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," *SIAM Journal on Control and Optimization*, vol. 42, no. 4, pp. 1143–1166, 2003.
- [7] J. N. Tsitsiklis, "Problems in decentralized decision making and computation," Ph.D. dissertation, Department of EECS, MIT, 1984.
- [8] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Transactions on Automatic Control*, vol. 48, no. 6, pp. 2953–2958, 2003.
- [9] L. Moreau, "Stability of multiagent systems with time-dependent communication links," *IEEE Transactions on Automatic Control*, vol. 50, no. 2, pp. 169–182, 2005.
- [10] P. Marbach and J. Tsitsiklis, "Simulation-based optimization of Markov reward processes," *IEEE Trans. Automat. Contr.*, vol. 46, no. 2, pp. 191–209, 2001.
- [11] J. Tsitsiklis, D. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Trans. Automat. Contr.*, vol. 31, no. 9, pp. 803–812, 1986.