2018 European Control Conference (ECC)
June 12-15, 2018. Limassol, Cyprus

Cloud-supported effective coverage of 3D structures

Antonio Adaldo, Dimos V. Dimarogonas, and Karl H. Johansson

Abstract— **In this paper, we present a distributed algorithm for cloud-supported effective coverage of 3D structures with a network of sensing agents. The structure to inspect is abstracted into a set of landmarks, where each landmark represents a point or small area of interest, and incorporates information about position and orientation. The agents navigate the environment following the proposed control algorithm until all landmarks have reached a satisfactory level of coverage. The agents do not communicate with each other directly, but exchange data through a shared cloud repository which is accessed asynchronously and intermittently. We show formally that, under the proposed control architecture, the networked agents complete the coverage mission in finite time. The results are corroborated by simulations in ROS, and experimental evaluation is in progress.**

I. Introduction

This paper addresses a problem of the inspection of a 3D structure with a team of autonomous sensing agents. Instead of communicating with each other, the agents upload information on a cloud repository to keep track of the progress of the inspection. Our problem falls within the scope of coverage problems for networks of mobile sensing agents. These problems have attracted a notable volume of research in the past few decades, because they constitute a flexible model for numerous applications, such as deployment, inspection and surveillance with networked robots [1].

Coverage problems can be divided in two large categories. Static coverage problems [2]–[6] are about finding a good placement for the sensing agents, and possibly controlling the agents to reach their target placements. Dynamic coverage problems [7]–[9] are about controlling the motion of the agents to survey an environment continuously, until it has been searched sufficiently well. Awareness coverage control [10], [11] can be considered a form of dynamic coverage where the agents are also required to learn a density function which characterizes the importance of each point in the environment. The problem addressed in this work falls within the class of dynamic coverage problems.

In classical works on dynamic coverage problems [7], the sensors have circular sensing patterns, meaning that their sensing power is maximal at their own position, and decays with the distance from the sensor. This model is only appropriate to describe omnidirectional sensors, such as temperature sensors or circular laser scans. In more recent works [8], the sensors have anisotropic sensing patters, which allows to model a larger variety of sensing devices, such as monocular cameras or cone-shaped laser scans. However, these models are still deficient in describing certain settings, such as the inspection of structure, the surveillance of a building, or the reconstruction of a 3D surface with depth sensors, because they do not capture the morphology of the structure to inspect. Conversely, in this paper we consider a network of sensing agents with generic, anisotropic and heterogeneous sensing patterns, and we also take into account the 3D geometry of the object to inspect. Moreover, classical works assume that each sensing agent is continuously aware of the motion of all other agents, or that the agents can communicate continuously. More recently, intermittent inter-agent communication has been considered [9]. In this work, inter-agent communication is completely replaced by communication with a shared information repository, or *cloud*. Connections with the cloud are event-triggered and intermittent, and the amount of data exchanged upon each connection is limited. Cloud-supported solutions are becoming increasingly popular for various types of multi-agent problems [12]–[14], thanks to the recent explosion of cloud-based services and technologies. Note that the cloud is not a centralized omniscient computer. In this work, the cloud is barely a shared information repository which receives asynchronous and partial information about the progress of the inspection. All calculations need to be performed locally and online by the individual agents. Finally, most existing work on effective coverage address the problem of inspecting a planar, continuous area. Conversely, we consider the inspection of a 3D structure abstracted into a finite set of landmarks, where each landmark carries information about the local curvature of the surface. This setup is not only more general with respect to considering a planar continuous environment, but also computationally more tractable, and allows the proposed control algorithm to be implemented on small embedded processors.

The algorithm is formally shown to complete the inspection in finite time, and it is demonstrated by simulating a team of sensing agents in the ROS framework [15], paving the way to experimental evaluation. Each agent is simulated as a different ROS node, effectively reproducing the distributed nature of the algorithm.

II. Preliminaries

The null vector in $\mathbb{R}^n$, with $n \in \mathbb{N}$, is denoted by 0_n. A vector in $\mathbb{R}^n$ is also intended as the corresponding column vector in $\mathbb{R}^{n\times 1}$. The cross product between two vectors $u, v \in \mathbb{R}^n$ is denoted by $u \times v$. The skew operator is denoted by $\mathrm{S}(\cdot)$ (i.e., $\mathrm{S}(u)v = u \times v$). The identity matrix in $\mathbb{R}^{n\times n}$ is denoted by I_n. The transpose of a vector or matrix is denoted by $(\cdot)^\intercal$. The Euclidean norm of a vector is denoted by $\|\cdot\|$. The set $\mathcal{S}^2 = \{u \in \mathbb{R}^3 : \ \|u\| = 1\}$ is called the *unit sphere*; a vector $u \in \mathcal{S}^2$ represents a direction in $\mathbb{R}^3$ and its

The authors are with the Department of Automatic Control, School of Electrical Engineering, KTH Royal Institute of Technology, Osquladas väg 10, 10044, Stockholm. Emails: {adaldo,dimos,kallej}@kth.se

This work has received funding from the European Unions Horizon 2020 Research and Innovation Programme under the Grant Agreement No.644128, AEROWORKS.

kinematics can be described as

$$\dot{u} = -\,\mathrm{S}(u)\omega, \tag{1}$$

where ω is called the *angular velocity* of the vector. Since $\mathrm{S}(\omega)u = \omega \times u$ is orthogonal to u, a vector $u \in \mathcal{S}^2$ which evolves according to (1) remains forever in $\mathcal{S}^2$. Moreover, since $\mathrm{S}(u)u = 0$, the kinematics (1) is invariant with respect to variations of ω along the direction of u; in other words, $\mathrm{S}(\omega + \alpha u)u = \mathrm{S}(\omega)u$ for all $\alpha \in \mathbb{R}$. The gradient of a function $f(x)$ with respect to a variable $x \in \mathbb{R}^3$ is denoted by $\partial f(x)/\partial x \in \mathbb{R}^3$. Gradients are also considered column vectors. Set closure is denoted by $\mathrm{Cl}(\cdot)$.

A *hybrid automaton* [16] is a tuple $\mathcal{H} = (Q, X, I, F, D, E, G, R)$, where: $Q = \{q^1, q^2, \ldots\}$ is a set of discrete states; $X \subset \mathbb{R}^n$ is a continuous state space; $I \subseteq Q \times X$ is a set of possible initial states; $F : Q \times X \to X$ is a set of vector fields, with $f(q, x)$ being the dynamics of x under state q; $D : Q \to 2^X$ is a set of domains, with $D(q)$ being the domain under state q; $E :\subseteq Q \times Q$ is a set of edges, with $(q^1, q^2) \in E$ signifying a possible transition from state q^1 to state q^2; $G : E \to 2^X$ is a set of guards, meaning that $x \in G_{1,2}$ triggers a transition from q^1 to q^2; $R : E \times X \to 2^X$ is a set of reset maps, meaning that, upon a transition from q^1 to q^2, the continuous state x of the system is reset to a value in $R_{1,2}(x)$. A hybrid automaton can be represented as a graph, where each node represents a discrete state and each edge represents a transition. Each node is labeled with the corresponding vector field, while each edge is labeled with the corresponding guard and reset map. If the reset map for an edge (q^1, q^2) is not specified, it is implied that $R_{1,2}(x) = x$ for all $x \in G_{1,2}$. Initial discrete states are labeled with a "start" flag. Each controller introduced in this paper is given as a hybrid automaton, and it is presented in its graph form. The set of the initial states is omitted whenever it is clear from the context.

III. Problem Statement

In this paper, we consider a set of N sensing agents indexed as $1, \ldots, N$, and we denote $\mathcal{N} = \{1, \ldots, N\}$. The ith agent is characterized by its position $p_i(t) \in \mathbb{R}^3$ and its orientation $n_i(t) \in \mathcal{S}^2$, and it is denoted as $A_i(t) = (p_i(t), n_i(t))$. A sensing agent is an abstraction of a mobile sensor, such as a camera or a laser scan. The orientation of the agent defines the direction that the agent is looking at. For example, if the sensor is a camera, the orientation of the agent corresponds to the direction that the camera is pointing to. The kinematics of the agents is given by

$$\dot{p}_i(t) = v_i(t), \tag{2a}$$

$$\dot{n}_i(t) = -\,\mathrm{S}(n_i(t))\omega_i(t) \tag{2b}$$

for all $i \in \mathcal{N}$. Here v_i and ω_i are control inputs; v_i is called the linear velocity of the agent, while ω_i is called the angular velocity of the agent.

The agents are required to inspect a surface which is abstracted into a set of M landmarks indexed as $1, \ldots, M$, where each landmark corresponds to a point on the surface, and we denote $\mathcal{M} = \{1, \ldots, M\}$. Like a sensing agent, the jth landmark is defined by its position $q_j \in \mathbb{R}^3$ and its orientation $m_j \in \mathcal{S}^2$—where this time the orientation m_j corresponds to the outward normal to the surface evaluated at q_j—and it is denoted as $L_j = (q_j, m_j)$. However, the positions and orientations of the landmarks are constant.

The perception of a generic landmark L attained by a generic agent $A = (p, n)$ is a function of the position and orientation of the landmark with respect to the agent. We denote this function as $\mathrm{per}(A, L)$, and we let it take values in $[0, 1]$, where $\mathrm{per}(A, L) = 0$ means that the landmark L is not visible at all by agent A, while $\mathrm{per}(A, L) = 1$ denotes the best possible perception. We require the further three properties for the perception function:

- P1 $\mathrm{per}(A, L)$ is continuously differentiable with respect to both p and n;
- P2 there exists $R > 0$ such that, if $\|p - q\| > R$, then $\mathrm{per}(A, L) = 0$, $\partial\,\mathrm{per}(A, L)/\partial p = 0_3$ and $\mathrm{S}(n)(\partial\,\mathrm{per}(A, L)/\partial n) = 0_3$;
- P3 for each $L \in \mathbb{R}^3 \times \mathcal{S}^2$, there exists $A_L^* \in \mathbb{R}^3 \times \mathcal{S}^2$ such that $\mathrm{per}(A_L^*, L) = 1$, $\partial\,\mathrm{per}(A_L^*, L)/\partial p = 0_3$ and $\mathrm{S}(n)(\partial\,\mathrm{per}(A_L^*, L)/\partial n) = 0_3$.

Property P1 is a technical assumption needed to prove our main result. Property P2 entails that a landmark cannot be perceived by a sensor if it is too fat. Property P3 entails that for any pose of a landmark there exists a sensor pose that yields the best possible perception, and such sensor pose constitutes a local optimum of the perception function.

For each landmark L_j, we define the instantaneous coverage $\gamma_j(t)$ as the sum of the perceptions of that landmark attained by the N sensors:

$$\gamma_j(t) = \sum_{i=1}^{N} \mathrm{per}(A_i(t), L_j). \tag{3}$$

Moreover, we define the *cumulated coverage* $\Gamma_j(t)$ as the integral of the instantaneous coverage:

$$\Gamma_j(t) = \int_0^t \gamma_j(\tau)\,\mathrm{d}\tau\,. \tag{4}$$

Note that, since $\gamma_j(t) \geq 0$, $\Gamma_j(t)$ is a nondecreasing function of the time.

The control objective is that the cumulated coverage of each landmark reaches a desired value C^*. This objective can be formalized as follows:

Definition 1: The inspection is completed successfully when $\Gamma_j(t) \geq C^*$ for all $j \in \mathcal{M}$.

Correspondingly, we introduce the *landmark coverage errors*

$$e_j(t) = \max\{0, C^* - \Gamma_j(t)\}, \tag{5}$$

and the *total coverage error*

$$E(t) = \sum_{j=1}^{M} e_j(t), \tag{6}$$

so that the control objective corresponds to driving all the landmark coverage errors to zero, or equivalently, driving the total coverage error to zero. Note that the coverage errors $e_j(t)$ are by definition nonnegative and nonincreasing.

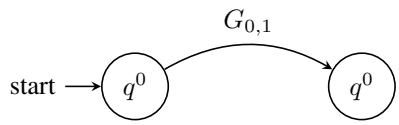

Fig. 1: Pose controller for a single agent. The vector fields under See (7).

IV. Hybrid control of an agent's kinematics to reach a desired position and orientation

Before we delve into the proposed control strategy, we shall show that the agent kinematics (2) can be controlled to reach any desired position and orientation $A^* = (p^*, n^*)$ asymptotically. To this aim, consider a generic agent $A_i(t) = (p_i(t), n_i(t))$ with kinematics (2) and under the hybrid controller defined in Figure 1, with:

$$v_i(t) = p^* - p_i(t), \tag{7a}$$
$$q^0 : \ \omega_i(t) = z_i, \tag{7b}$$
$$q^1 : \ \omega_i(t) = \mathrm{S}(n_i(t))n^*, \tag{7c}$$
$$G_{0,1} : n_i(t)^\intercal n^* \geq -\varsigma, \tag{7d}$$

and where $\varsigma \in (0,1)$ is a control parameter, and $z_i \in \mathcal{S}^2$ is any unit vector orthogonal to $n_i(0)$.

Lemma 1: Consider a generic agent $A_i(t) = (p_i(t), n_i(t))$ with kinematics (2). Under controller (7), the agent reaches (p^*, n^*) asymptotically.

Proof: Since the kinematics of $p_i(t)$ and $n_i(t)$ are decoupled, they can be analyzed separately. Since $v_i(t) = p^* - p_i(t)$ regardless of the discrete state q_i, $p_i(t)$ converges trivially to p^*. As for $n_i(t)$, consider the candidate Lyapunov function $V(t) = 1 - n_i(t)^\intercal n^*$, and distinguish the two cases (i) $n_i(0)^\intercal n^* \geq -\varsigma$ and (ii) $n_i(0)^\intercal n^* < -\varsigma$. In Case (i), the controller switches immediately to state q^1, and we have

$$\begin{aligned}\dot{V}(t) &= -\,\dot{n}_i(t)^\intercal n^* = -(-\,\mathrm{S}(n_i(t))\omega_i(t))^\intercal n^* \\ &= -\,(-\,\mathrm{S}(n_i(t))\,\mathrm{S}(n_i(t))n^*)^\intercal n^* \\ &= -\,\|\mathrm{S}(n_i(t))n^*\|^2 \leq 0.\end{aligned} \tag{8}$$

Note that, from (8), we have that $n_i(t)^\intercal n^*$ is nondecreasing and that $\dot{V}(t) = 0$ if and only if $n_i(t) = \pm n^*$. However, since $n_i(0)^\intercal n^* \geq -\varsigma$, we conclude that $\dot{V}(t) = 0$ if and only if $n_i(t) = n^*$. Hence, $V(t)$ is a Lyapunov function and $n_i(t)$ converges to n^* asymptotically. In Case (ii), the controller is initially in state q^0, and $n_i(t)$ rotates around z_i with constant angular speed. After $n_i(t)$ has described a rotation of at most $2\arccos(\varsigma)$, we have $n_i(t)^\intercal n^* \geq -\varsigma$, which causes the controller to switch to state q^1, and we can reason as in Case (i). ■

An important consequence of Lemma 1 is that, if the initial conditions $A_i(0)$ and the target A^* are chosen out of a compact set $\mathcal{I}$, then the agent reaches any arbitrarily small neighborhood of $A^* = (p^*, n^*)$ in finite time. (Note that, since $\mathcal{S}^2$ is compact, the condition that $A_i(0)$ and A^* should lie in a compact set reduces to $p_i(0)$ and p^* being in a compact set.) In particular, if A^* is such that $\mathrm{per}(A^*, L) = 1$ for some landmark L, we have the following corollary.

Corollary 1: Let a sensing agent $A_i(t)$ with kinematics (2) be controlled by Controller 1, with $A^* = A^*_L$ is such that $\mathrm{per}(A^*, L) = 1$. Then, for any $\epsilon' < 1$, there exists a finite time $T_{\mathcal{I},\epsilon'}$ such that $\mathrm{per}(A_i(t), L) \geq \epsilon'$ for all $t \geq T_{\mathcal{I},\epsilon'}$.

V. Effective coverage control for one agent

In this section, we describe a controller that lets a single sensing agent attain the effective coverage of a set of landmarks. This controller will serve as a stepping stone to introduce our multi-agent cloud-supported control scheme in Section VI.

First, note that when the coverage mission is performed by a single agent, the coverage of each landmark only depends on the motion of that agent. Therefore, the agent can keep track of the coverage error associated to each landmark with only local information. Denoting the position and orientation of the agent as $A_i(t) = (p_i(t), n_i(t))$, the derivative of the total coverage error is

$$\dot{E}(t) = -\sum_{\substack{j=1,\ldots,M \\ e_j(t)>0}} \mathrm{per}(A_i(t), L_j) \leq 0. \tag{9}$$

Since the control objective is to drive $E(t)$ to zero, we can control $A_i(t)$ according to a gradient descent of $\dot{E}(t)$, so that the decay of $E(t)$ is as fast as possible.

The proposed single-agent controller is a hybrid controller that switches between a gradient descent of $\dot{E}(t)$ and controller (7), the latter being used as a backup when the gradient of $E(t)$ is too small. The gradient-descent controller can be written as follows:

$$v_i(t) = -\,\frac{\partial \dot{E}(t)}{\partial p_i}, \tag{10a}$$
$$\omega_i(t) = \mathrm{S}(n_i(t))\frac{\partial \dot{E}(t)}{\partial n_i}. \tag{10b}$$

Writing $\ddot{E}(t) = (\partial \dot{E}/\partial p_i)^\intercal \dot{p}_i(t) + (\partial \dot{E}/\partial n_i)^\intercal \dot{n}_i(t)$, and using (2), we can see that under Controller (10) we have

$$\ddot{E}(t) = -\|v_i(t)\|^2 - \|\omega_i(t)\|^2 \leq 0. \tag{11}$$

However, $\ddot{E}(t)$ is not defined for t such that, for some $j \in \{1, \ldots, M\}$, $e_j(t) = 0$ and $e_j(\tau) > 0$ in a left neighborhood of t. In these time instants, instead, $\dot{E}(t)$ istantaneously increases by $\mathrm{per}(A_i(t), L_j)$, because L_j has been completely inspected and stops contributing to the decay in the total coverage error.

The controller starts in the gradient-descent mode (10). The condition to switch to (7) is

$$\dot{E}(t) > -\epsilon \cap E(t) > 0, \tag{12}$$

where $\epsilon \in (0,1)$ is a constant threshold. Condition (12) means that the inspection is not complete, but the decay of $E(t)$ is close to zero. When switching to (7), one landmark L_ι is selected among those with a positive coverage error, and the reference pose is set as $A^*_{L_\iota}$. Note that, when this transition happens, at least one landmark with positive coverage error must exist, since $E(t) > 0$. The condition to switch back to (10) is

$$e_\iota(t) = 0 \cup \dot{E}(t) \leq -\epsilon', \tag{13}$$

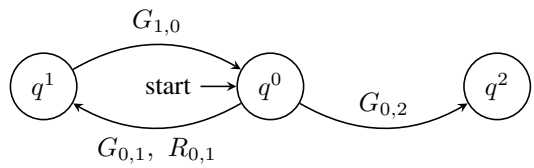

Fig. 2: Effective-coverage controller for a single agent. States: q^0 runs (2) and (10); q^1 runs (2) and (7) with $A^* = A^*_{L_\iota}$; q^2 terminates. Guards: $G_{0,1}$ is (12); $G_{1,0}$ is (13); $G_{0,2}$ is $E(t) = 0$. Reset maps: $R_{0,1}$ is $\iota :\in \{1, \dots, M : \ e_\iota(t) > 0\}$.

where $\epsilon' \in (\epsilon, 1)$. Condition (13) means that either L_ι has been completely inspected ($e_\iota(t) = 0$) or the decay of the coverage error is faster ($\dot{E}(t) \leq -\epsilon'$). The inspection terminates when the condition $E(t) = 0$ is detected.

The proposed switching controller is given in Figure 2 in the form of a hybrid automaton, and it is referred to as Controller 2 in the rest of the paper.

Now we can prove that a sensing agent with kinematics (2) controlled by Controller 2 completes the inspection in finite time.

Theorem 1: Consider an agent $A_i(t) = (p_i(t), n_i(t))$ with kinematics (2), and let it be controlled by Controller 2. Let $\mathcal{I} \subset \mathbb{R}^3 \times \mathcal{S}^2$ be a compact set that encloses the initial pose $A_i(0)$ of the sensor and all the landmarks $L_1, \dots, L_M$. Then, there exists $T > 0$ such that $E(t) = 0$ for all $t \geq T$.

Proof: First note that, when Controller 2 is in state q^0, we have $\dot{E}(t) \leq -\epsilon$. Since $E(t) \in [0, C^*]$ by definition, Controller 2 can be in state q^0 for a time not larger than C^*/ϵ. However, under state q^0, $\dot{E}(t)$ is nonincreasing. Therefore, the only way to trigger a transition to state q^1 is that the coverage error $e_j(t)$ of some landmark L_j reaches zero, causing $\dot{E}(t)$ to istantaneously increase by $\mathrm{per}(A_i(t), L_j)$. However, from Corollary 1, we know that, once Controller 2 is in state q^1, after a time not larger than $T_{\mathcal{I},\epsilon'}$, we will have $\mathrm{per}(A_i(t), L_\iota) \geq \epsilon'$, which implies $\dot{E}(t) \leq -\epsilon'$. Therefore, Controller 2 can only remain in state q^1 of $T_{\mathcal{I},\epsilon'}$ for each landmark. Since there are M landmarks, Controller 2 can be in state q^1 for at most a time of $MT_{\mathcal{I},\epsilon'}$ before the inspection is complete. Hence, the theorem is proved with $T = C^*/\epsilon + MT_{\mathcal{I},\epsilon'}$. ■

Remark 1: A sensible choice of ϵ and ϵ' is needed to avoid a slow inspection and frequent switching between states q^0 and q^1 in Controller 2. For example, a larger ϵ' reduces the switching frequency, but may slow down the inspection, because the agent spends more time focusing on a single landmark rather than on the gloabal coverage error. ■

VI. Cloud-supported effective coverage control for multiple agents

In this section, we consider the case that the inspection is performed by a team of N agents aided by a shared information repository (cloud). In this scenario, each individual agent does not know the coverage error $e_j(t)$ associated to each landmark, or the total coverage error $E(t)$. Therefore, for each agent i and each landmark j, we introduce the *estimated coverage error* $\hat{e}^i_j(t)$, which is initialized as $\hat{e}^i(0) = C^*$ and evolves as

$$\dot{\hat{e}}^i_j(t) = -\,\mathrm{per}(A_i(t), L_j). \tag{14}$$

Similarly, we let $\hat{E}^i(t) = \sum_{j=1}^M \hat{e}^i_j(t)$. Each agent can intermittently communicate with the cloud to upload and download information about the progress of the inspection. The cloud substitutes inter-agent communication, and allows the agents to gather their contributions to the inspection. For each landmark, the cloud maintains an estimate $\hat{e}^{\mathrm{cloud}}_j(t)$ of the coverage error associated to that landmark, which is initialized as $\hat{e}^{\mathrm{cloud}}_j(0) = C^*$. Hence, the amount of information contained in the cloud scales linearly with the number M of landmarks, and does not grow over time. We denote as $t_{i,k}$ the time when agent i connects with the cloud for the kth time. (Conventionally, $t_{i,0} = 0$ for all agents.) Cloud accesses are considered as instantaneous events. This assumption is mild because the time scale of wireless communication is arguably orders of magnitude faster than the time scale of the physical motion of the agents. Only one agent at a time can access the cloud: if it happens that $t_{i,k} = t_{j,h}$ for some agents i, j and some integers k, h, the two accesses happen one after the other in any order. Between two consecutive connections to the cloud, an agent needs to keep track only of its own contribution to the coverage. This contribution can be captured simply with a nonnegative scalar c_{ijk} defined as

$$c_{ijk} = \int_{t_{i,k-1}}^{t_{i,k}} \mathrm{per}(A_i(t), L_j)\,\mathrm{d}t\,. \tag{15}$$

When agent i connects to the cloud at time $t_{i,k}$, the estimated coverage errors are updated as

$$\hat{e}^{\mathrm{cloud}}_j(t^+_{i,k}) = \hat{e}^i_j(t^+_{i,k}) = \max\{0, \hat{e}^{\mathrm{cloud}}_j(t^-_{i,k}) - c_{ijk}\}. \tag{16}$$

This update means that the contribution c_{ijk} is incorporated in the coverage error estimated by the cloud and coverage error estimated by the agent is immediately updated to match the one estimated by the cloud. In this way, the contribution given to the coverage by each agent is collected in the cloud, and can be accessed later by other agents. Comparing (5) with (16), we can see that estimates of the coverage errors have the following remarkable properties:

$$\hat{e}^i_j(t^+_{i,k}) \leq \hat{e}^i_j(t^-_{i,k}); \tag{17a}$$

$$e_j(t) \leq \min\{\hat{e}^{\mathrm{cloud}}_j(t), \hat{e}^i_j(t)\}. \tag{17b}$$

Property (17a) means that a connection with the cloud can only cause the local estimates of the coverage errors to decrease. Property (17b) means that the estimates of the coverage errors are always overestimates; in particular, $\hat{E}^i(t) = 0$ implies $E(t) = 0$. This gives a natural stopping condition for the agents: when $\hat{E}^i(t) = 0$, agent i knows that the inspection is complete. Between two consecutive cloud accesses, each agent is controlled on the base of its estimated coverage errors. Namely, (10) is replaced with

$$v_i(t) = -\,\frac{\partial \dot{\hat{E}}^i(t)}{\partial p_i}, \tag{18a}$$

$$\omega_i(t) = \mathrm{S}(n_i(t))\frac{\partial \dot{\hat{E}}^i(t)}{\partial n_i}. \tag{18b}$$

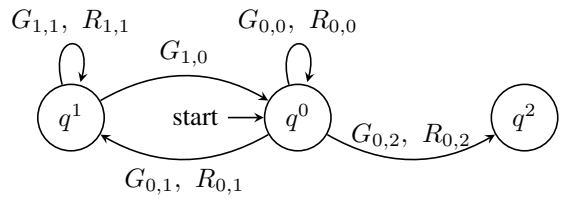

Fig. 3: Cloud-supported effective-coverage controller for a multi-agent team. States: q^0 runs (2) and (18); q^1 runs (2) and (7); q^2 terminates. Guards: $G_{0,1}$ is (12); $G_{1,0}$ is (13); $G_{0,2}$ is $\hat{E}^i(t) = 0$; $G_{0,0}$ and $G_{1,1}$ are (19). Updates: $R_{0,1}$ is $\iota \in \{1, \dots, M : \ e_\iota(t) > 0\}$; $R_{0,0}$, $R_{1,1}$ and $R_{0,2}$ are (16).

The cloud accesses are triggered according to the following recursive rule:

$$t_{i,k+1} = \inf\{t > t_{i,k} : \ C_{i,k}(t) \geq \varsigma(M_i(t) + 1) \text{ or } \hat{E}^i(t) = 0\}, \quad (19)$$

where

$$C_{i,k}(t) = \sum_{j=1}^{M} \int_{t_{i,k}}^{t} \mathrm{per}(A_i(\tau), L_j) \mathrm{d}\tau, \quad (20)$$

$$M_i(t) = |\{j \in \{1, \dots, M\} : \ \hat{e}_j(t) > 0\}|, \quad (21)$$

and where ς is a positive constant. This rule has the intuitive meaning that a cloud access is triggered when the agent has accumulated enough coverage contribution to share with the other agents ($C_{i,k}(t) \geq \varsigma(M_i(t)+1)$) or when the inspection is complete ($\hat{E}^i(t) = 0$). The value of ς represents a tradeoff between how often the agents access the cloud and how promptly they upload their contributions on the cloud. The proposed controller is formalized as a hybrid automaton in Figure 3 and is referred to as Controller 3. Note that the looping transitions in q^0 and q^1 represent the connections to the cloud. Upon these connections, agent i shares its contributions c_{ijk}, so that the estimated coverage errors $\hat{e}^i_j$ and $\hat{e}^{\text{cloud}}_j$ can be update according to (16). In the following Theorem 2 we prove formally that a team of sensing agents controlled by Controller 3 completes the inspection in finite time.

Theorem 2: Consider a team of mobile sensing agents with poses $A_i(t) = (p_i(t), n_i(t))$, with $i \in \{1, \dots, N\}$ and let the agents be controlled by Controller 3. Let $\mathcal{I} \subset \mathbb{R}^3 \times \mathcal{S}^2$ be a compact set that encloses the initial poses $A_i(0)$ of all the agents and all the landmarks $L_1, \dots, L_M$. Then, there exists $T > 0$ such that $E(t) = 0$ for all $t \geq T$.

Proof: First note that, under the proposed controller, the estimated errors $\hat{E}^i(t)$ are nonincreasing. In fact, the motion of the agents imposes $\dot{\hat{E}}^i(t) \leq 0$, while, from (19) we can see that the cloud accesses impose $\hat{e}^i_j(t^+_{i,k}) \leq \hat{e}^i_j(t^-_{i,k})$, which summing over the landmarks yields $\hat{E}^i(t^+_{i,k}) \leq \hat{E}^i(t^-_{i,k})$. Similarly to the proof of Theorem 1, note that when Controller 3 is in state q^0, we have $\dot{\hat{E}}^i(t) \leq -\epsilon$. Since $\hat{E}^i(t) \in [0, C^*]$ by definition, and since $\hat{E}^i(t)$ has been shown to be nonincreasing, the controller can only remain in q^0 for a time of at most C^*/ϵ. However, since, under (18), $\dot{\hat{E}}^i(t)$ is nonincreasing, the only way for Controller 3 to transit to state q^1 is that the estimated error $\hat{e}^i_j(t)$ of a landmark reaches zero, causing $\dot{\hat{E}}^i(t)$ to drop by $\mathrm{per}(A_i(t), L_j)$. Since there are M landmarks, the controller can only transit to state q^1 for at most M times. Once the controller is in state q^1, by Corollary 1, it will take at most a time $T_{\mathcal{I},\epsilon'}$ to have $\dot{\hat{E}}^i(t) \leq -\epsilon'$ and transit back to q^0. Hence, the controller can only remain in q^1 for at most a time of $MT_{\mathcal{I},\epsilon'}$. We must conclude that $\hat{E}^i(t)$ reaches zero in at most $MT_{\mathcal{I},\epsilon'} + C^*/\epsilon$. Since $E(t) \leq \hat{E}^i(t)$ by design, this result also implies that $E(t)$ reaches zero in at most $MT_{\mathcal{I},\epsilon'} + C^*/\epsilon$. ■

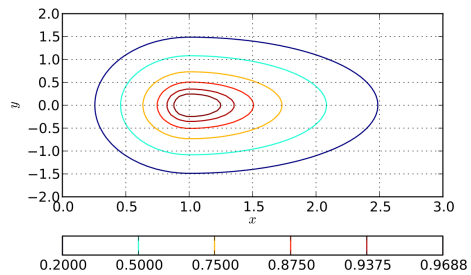

Fig. 4: Contour plot of $f(p, n, q)$ for $p = 0_3$, $n = (1, 0, 0)$ and $q \in [x, y, 0]$ with $x \in [0, 3]$, $y \in [-2, 2]$.

VII. Simulation

In this section, we present a simulation of the proposed cloud-supported distributed controller. We consider a network of $N = 4$ agents and a set of $M = 100$ landmarks sampled from a suface with the shape of an extruded sinusoid. The desired coverage for all landmarks is set to $C^* = 100$. The sensing pattern of the agents is chosen as follows:

$$\mathrm{per}(A_i, L_j) = f(p_i, n_i, q_j) \lfloor n_i m_j \rfloor, \quad (22)$$

where

$$\lfloor x \rfloor = \max\{x, 0\}, \quad (23)$$

$$g(p, n, q) = \left\lfloor 1 - \frac{\|q - p - rn\|^2}{R^2} \right\rfloor, \quad (24)$$

$$h(p, n, q) = \left\lfloor 1 - \frac{((q - p - rn)n)^2}{r^2} - \frac{\|q - p - rn\|^2 - ((q - p - rn)n)^2}{R^2} \right\rfloor, \quad (25)$$

$$f(p, n, q) = \begin{cases} g(p, n, q) & (q - p - rn)n > 0, \\ h(p, n, q) & (q - p - rn)n < 0, \end{cases} \quad (26)$$

and where $0 < r < R$. A contour plot of $f(p, n, q)$ is given in Figure 4. One can verify that (22) satisfies the properties of a perception function as listed in Section III.

Figure 6 shows the total coverage error estimated by the cloud and the trajectories followed by the agents. Figure 5 shows the number of cloud accesses for each agent over time and details which controller is active for each agent. Finally, Figure 7 shows four snapshots of the configuration of the

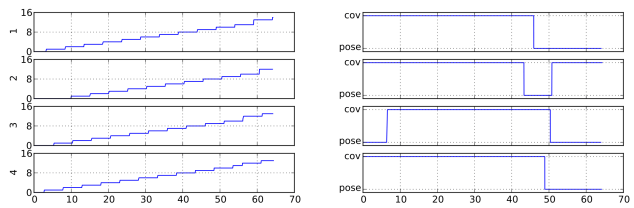

Fig. 5: (Left) Number of cloud accesses for each agent during the simulation. (Right) Discrete state in Controller 3 for each agent: `pose` for q^1, corresponding to (7); `cov` for q^0, corresponding to (18).

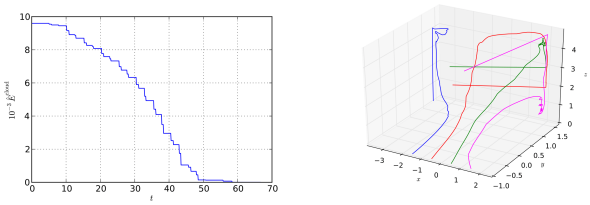

Fig. 6: (Left) Coverage error estimated by the cloud. (Right) Trajectories followed by the agents.

agents and the landmarks during the simulation. In these snapshots, each agent is represented as an arrow located in p_i and pointing in the direction of n_i, while each landmark is represented as a dot located in q_j. The orientation of the landmarks is not represented to avoid cluttering the pictures. The color of the landmarks varies from red to blue to represent the value of $\hat{e}_j^{\text{cloud}}$ from C^* to zero. From Figure 7, we can see how the agents adjust their orientation to follow the local curvature of surface.

VIII. Conclusions and Future Developments

We have proposed a distributed control algorithm for the inspection of a 3D surface with a team of mobile sensing agents with generic, heterogeneous sensing patterns. The controller is based on intermittent connections of the agents with a cloud repository, which eliminates the need for inter-agent communication. We have shown that the controller guarantees that the inspection is completed in finite time, and we have validated it by simulation. We are currently working on incorporating a collision avoidance scheme in the proposed controller and on its experimental evaluation on a network of aerial robotic sensors.

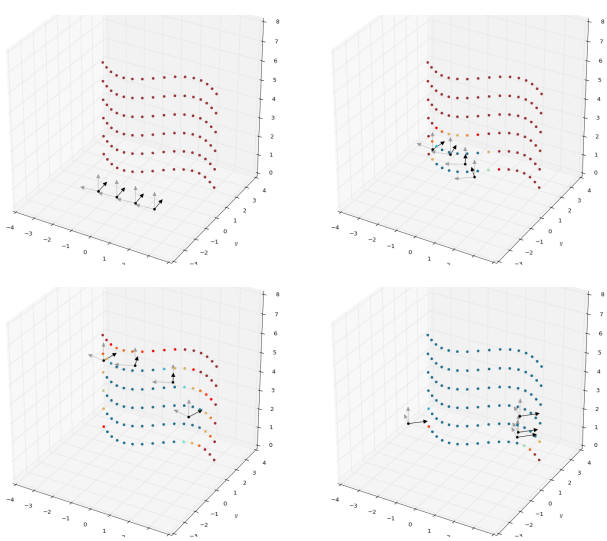

Fig. 7: Position and orientations of the agents during the simulation. Times from left to right: 0, 20, 40, 60.

References

[1] C. G. Cassandras and W. Li, "Sensor networks and cooperative control," *European Journal of Control*, vol. 11, no. 4-5, pp. 436–463, 2005.

[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] J. W. Durham, R. Carli, P. Frasca, and F. Bullo, "Discrete Partitioning and Coverage Control for Gossiping Robots," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 364–378, 2012.

[4] Y. Stergiopoulos, M. Thanou, and A. Tzes, "Distributed Collaborative Coverage-Control Schemes for Non-Convex Domains," *IEEE Transactions on Automatic Control*, vol. 60, no. 9, pp. 2422–2427, 2015.

[5] Y. Stergiopoulos and A. Tzes, "Spatially distributed area coverage optimisation in mobile robotic networks with arbitrary convex anisotropic patterns," *Automatica*, vol. 49, no. 1, pp. 232–237, 2013.

[6] A. Adaldo, D. V. Dimarogonas, and K. H. Johansson, "Hybrid coverage and inspection control for anisotropic mobile sensor teams," in *IFAC World Congress*, 2017.

[7] I. I. Hussein and D. M. Stipanovic, "Effective Coverage Control for Mobile Sensor Networks With Guaranteed Collision Avoidance," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 4, pp. 642–657, 2007.

[8] D. Panagou, D. M. Stipanovic, and P. G. Voulgaris, "Distributed dynamic coverage and avoidance control under anisotropic sensing," *IEEE Transactions on Control of Network Systems*, 2016.

[9] Y. Wang and I. I. Hussein, "Awareness coverage control over large-scale domains with intermittent communications," *IEEE Transactions on Automatic Control*, vol. 55, no. 8, pp. 1850–1859, 2010.

[10] C. Song, G. Feng, Y. Fan, and Y. Wang, "Decentralized adaptive awareness coverage control for multi-agent networks," *Automatica*, vol. 47, pp. 2749–2756, 2011.

[11] C. Song, L. Liu, G. Feng, Y. Wang, and Q. Gao, "Persistent awareness coverage control for mobile sensor networks," *Automatica*, vol. 49, pp. 1867–1873, 2013.

[12] A. Adaldo, D. Liuzza, D. V. Dimarogonas, and K. H. Johansson, "Cloud-supported formation control of second-order multi-agent systems," *IEEE Transactions on Control of Network Systems*, 2017.

[13] J. W. Durham, R. Carli, P. Frasca, and F. Bullo, "Dynamic partitioning and coverage control with asynchronous one-to-base-station communication," *IEEE Transactions on Control of Network Systems*, vol. 3, no. 1, pp. 5589–5594, 2016.

[14] M. T. Hale and M. Egerstedty, "Differentially private cloud-based multi-agent optimization with constraints," in *Proceedings of the American Control Conference*, Chicago, IL, USA, 2015.

[15] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Mg, "ROS: an open-source Robot Operating System," in *IEEE International Conference on Robotics and Automation*, 2009.

[16] J. Lygeros, K. Johansson, S. Simic, and S. Sastry, "Dynamical properties of hybrid automata," *IEEE Transactions on Automatic Control*, vol. 48, no. 1, pp. 2–17, 2003.