Distributed online assignment of charging stations in persistent coverage control tasks based on LP relaxation and ADMM

Zhiyuan Lu\textsuperscript{a}, Shunya Yamashita\textsuperscript{a}, Junya Yamauchi\textsuperscript{b} and Takeshi Hatanaka\textsuperscript{a}

\textsuperscript{a}Tokyo Institute of Technology, Tokyo, Japan; \textsuperscript{b}The University of Tokyo, Tokyo, Japan

\section*{ABSTRACT}
This paper investigates distributed online assignment of charging stations for a drone network in a persistent coverage control task. To ensure persistency not only in motion but also in energy, drones need to go back to charging stations before running out of their batteries. Coverage control schemes with energy persistency were presented in the literature based on so-called control barrier functions. These methodologies, however, assume a fixed correspondence between a drone and a charging station, but always returning to a preassigned station is not necessarily an efficient decision, namely the constraint may hinder the monitoring behaviour of the drones. Dynamically reassigning charging stations to drones is thus expected to enhance the coverage performance. To this end, we formulate an online assignment problem of charging stations with parameters determined by the control barrier function values in real time, and exactly relax the formulated optimization problem to a linear programming problem. We then propose a distributed solution to the problem based on ADMM and the overall partially distributed control architecture including persistent coverage control and online assignment of charging stations. The control system is finally demonstrated through Monte Carlo simulation.

\section*{1. Introduction}
Drones are widely used for a variety of monitoring tasks due to their small size and ease of deployment [1–3]. Deploying multiple drones is expected to improve the work efficiency in monitoring vast areas. For this purpose, coverage control is one of the promising technologies which deploys mobile robots over a mission space optimally under a distributed control law [4–6]. Coverage control has also been applied to multiple drones to use their on-board camera for monitoring the environment [7–9]. While coverage control drives robots to a static optimal solution, such algorithms may not fully take advantages of mobility, and not suitable when dynamically changing environments need to be monitored. Motivated by this issue, persistent coverage control was presented in [10], where the robots persistently revisit points on the mission space.

Considering the differential characteristics of drones from ordinary robots, we have additional requirements. To ensure persistency in mission, we need to make sure that they keep safe battery levels. To certify the energy persistency, constraint-based controllers were presented in [11,12] based on the notion of control barrier function (CBF) [13]. In particular, these papers arrange charging stations on the mission space and guarantee that the drone always has enough power to return to the charging stations. However, the additional constraint may limit the distance between the drone and the preassigned charging station, which may prevent drones from fully exploiting their potential exploitation capabilities. Dynamically reassigning the charging stations to drones is expected to mitigate the strong constraints on the drone motion and enhance the coverage performance.

In this paper, we design an algorithm to dynamically assign charging stations for the scenario of persistent coverage control. We first formulate an optimal assignment problem of charging stations to drones as an integer programming problem. We then relax the problem into a linear programming (LP) problem exactly and numerically demonstrate the computational advantages of the relaxed problem. The exact LP relaxation brings another benefit, namely the optimal assignment is computed in a distributed fashion by an existing distributed optimization technique. In this paper, we employ the alternating direction method of multipliers (ADMM), a well-known distributed optimization algorithm [14–17], and present a novel partially distributed control architecture combining persistent coverage control and optimal assignment of charging stations. We finally conduct Monte Carlo simulations to verify that the present algorithm improves the monitoring efficiency.
The contributions of this paper are summarized as below.

- We present the concept of dynamically reassigning charging stations for persistent coverage control for the first time.
- We propose a novel cost function based on a CBF to avoid a risk that reassignment prohibits drones from returning a charging station.
- We reduce the assignment problem of charging stations into distributed convex optimization through LP relaxation.
- We present a distributed control structure which combines the persistent coverage control and ADMM-based assignment algorithm of charging stations, and demonstrate its effectiveness through simulations.

The contents of this paper are in part presented in our conference version [18]. The incremental contribution of this paper is to add a Monte Carlo simulation to better present the effectiveness of the proposed algorithm.

2. Preliminary: CBF-based coverage control

Let us consider \( n \) drones with 1Ds \( I = \{1, \ldots, n\} \) flying over the mission space \( \mathcal{Q} \subset \mathbb{R}^2 \). As shown in Figure 1, drones are required to monitor the mission space \( \mathcal{Q} \) by cameras mounted on the drones. A drone potentially can move in the three-dimensional space but, for simplicity, we assume that they are locally controlled so that their flight altitudes are common and fixed. Accordingly, the position of drone \( i \in I \) is represented by the 2-D coordinates \( p_i \in \mathbb{R}^2 \) other than the coordinate corresponding to the height. The dynamics of drone \( i \in I \) is then modelled as

\[
\dot{p}_i(t) = u_i(t),
\]

where \( u_i(t) \in \mathbb{R}^2 \) is the control input to be designed later.

We assume that the camera of each drone \( i \) can observe the range within the circle with radius \( R \) centred at \( p_i \) in the mission space as illustrated in Figure 1. Then the sensing range of the on-board camera of drone \( i \) can be formulated as

\[
\mathcal{B}_i(p_i) = \{q \in \mathcal{Q} \mid \|q - p_i\| \leq R\}.
\]

To manage the area overlapping of other drones, we introduce so-called Voronoi partition defined as

\[
\mathcal{V}_i(p) = \{q \in \mathcal{Q} \mid \|q - p_i\| \leq \|q - p_j\| \forall j \in I \setminus \{i\}\},
\]

where \( p \) is a collection of \( p_i \) (\( i \in I \)). The sensing region of drone \( i \) is then assumed to be characterized by the set

\[
\mathcal{S}_i(p) := \mathcal{B}_i(p_i) \cap \mathcal{V}_i(p).
\]

In this paper, we address the monitoring task in the paradigm of coverage control [4]. To this end, we define the objective function to be maximized by the drones as

\[
J(p, t) := -\sum_{i=1}^{n} \int_{\mathcal{S}_i(p)} \|q - p_i\|^2 \phi(q, t) \, dq + r \int_{\mathcal{Q} \setminus \cup_{i=1}^{n} \mathcal{S}_i} \phi(q, t) \, dq \quad (r \leq -R^2)
\]

following [4]. The function \( \phi : \mathcal{Q} \times \mathbb{R}_\geq \rightarrow \mathbb{R} \), called density function, represents relative importance over the mission space. The authors in [4,7] presented a gradient ascent algorithm for a time invariant density function \( \phi \) to render \( J \) maximized but the strategy leads drones to a stationary configuration. Instead, we consider the persistent coverage control that requires drones to patrol the mission space while revisiting every point in the mission space infinitely often. The papers [10] revealed that the persistently covering motion is achieved by combining the gradient ascent algorithm with the density update

\[
\frac{d\phi(q, t)}{dt} = \begin{cases} -\delta \phi(q, t), & \text{if } q \in \cup_{i=1}^{n} \mathcal{S}_i \\ \delta (1 - \phi(q, t)), & \text{otherwise}. \end{cases} \quad (\delta, \delta > 0) \]

The paper [12] further presents a constraint-based controller with the notion of CBF in order that drones meet additional requirements, namely performance guarantee and battery charging. The performance guarantee is reflected by the following constraint with a performance specification \( \gamma \).

\[
h_{\text{perf}}(p, t) = J(p, t) - \gamma \geq 0.
\]

To describe the specification on the battery charging, Dan et al. [12] take the battery model

\[
\dot{E}_i = -K_{ch} (K_{ch} \geq 0),
\]
where $E_i$ is the battery level of drone $i$. A charging station for drone $i$ is then assumed to lie at $q_i$ on $Q$, and drone $i$ can start charging only if it lies inside of the charging radius $d_{ch}$. Namely, the battery level increases only when charging, and otherwise it decreases at the rate of $K_{ch}$. Let us then define the function

$$h_{i,ch}(p_i, E_i) = E_i - E_{\min}\frac{K_{ch}}{K_{ch}}\left(\|p_i - q_i\| - d_{ch}\right),$$

(8)

where $E_{\min}$ denotes the minimal battery level that drones should maintain. We select a positive real value for $K_{ch}$ to let $K_{ch}/K_{ch}(\|p_i - \hat{p}_i\| - d_{ch})$ larger than the energy required by traveling the distance $\|p_i - q_i\|$. The specification of the battery charging is then reflected by the constraint

$$h_{i,ch}(p_i, E_i) \geq 0.$$  

(9)

To enforce (6) and (9), [12] presents the constraint-based controller $u_i = K_i(p, t)$:

$$K_i(p, t) = \arg\min_{(u_i, \lambda_i, \nu_i) \in \mathbb{R} \times \mathbb{R}} \left\|u_i^*\right\|^2 + \|\lambda_i\|^2 + \|\nu_i\|^2

\text{s.t.} \quad \left(\frac{\partial J(p, t)}{\partial p_i}\right)^T u_i^* + \xi(p, t) \geq \epsilon_i \lambda_i,

- \left(\frac{K_{ch}}{K_{ch}}\frac{1}{\|p_i - \hat{p}_i\|} - K_{ch}\right) u_i^* - K_{ch}

+ \beta \left(h_{i,ch}(p_i, E_i)\right) \geq \epsilon_i \nu_i,$$

(10)

with

$$\xi_i(p, t) := \int_{S_i} \left(\hat{d} - k\right) \left\|q - p_i\right\|^2 \phi(q, t) \, dq

+ r \left\{\int_{Q} \left(\hat{d} + (k - \hat{d})\phi(q, t)\right) \, dq

- \int_{S_i} \left(\hat{d} + (k - \hat{d})\phi(q, t)\right) \, dq\right\}

- \frac{k}{n} \gamma \quad (k > 0),$$

(11)

where $\lambda_i$ and $\nu_i$ are the slack variables to allow constraint violations due to the conflicts among specifications. The non-negative $\epsilon_i$ and $\epsilon_i$ ($\epsilon_i \gg \epsilon_i$) are weights that adjusted to let the guarantee of battery level have a higher priority than the coverage performance guarantee. The function $\beta$ is a locally Lipschitz extended class $\mathcal{K}$ function.

**Remark 2.1:** In this paper, we take a circular sensing range model in (2), but the above solution can be easily extended to another typical sensing range model, namely rectangular model, by using the gradient of $f$ presented in [7] and accepting additional computation complexity.

### 3. Optimal assignment of charging stations

While (10) enforces drones to comply with the constraint (9), it may prohibit drones from leaving their preassigned charging stations further than their battery capacities. This kind of assignment may be unreasonable because drones need to land at their assigned charging stations even if other charging stations are much closer. To improve the monitoring efficiency, we update the assignment of charging stations in real time, based on the current state of the drones.

Let us now consider charging stations $J = \{1, \ldots, m\}$ placed on the mission space $Q$. We assume that only one drone can be charged at a station at the same time similarly to [11,12]. Differently from the previous section, $m$ may differ from the number of drones $n$. In the sequel, the position of each charging station $j \in J$ is denoted by $s_j \in Q$. We assume that if a drone enters the disk centred at $s_j$ with radius $d_{ch}$, it can start charging. Define a variable $\delta_{ij}$ as

$$\delta_{ij} = \begin{cases} 
1 & \text{if station } j \text{ is assigned to drone } i, \\
0 & \text{if station } j \text{ is not assigned to drone } i.
\end{cases}$$

(12)

Regardless of $n$ and $m$, a drone needs to be assigned to at most one charging station. However, the other conditions for $\delta_{ij}$ to satisfy vary depending on $n$ and $m$. For example, when $m < n$, a charging station may have to accept multiple drones to ensure feasibility. Although some variations would be conceivable, we take the following conditions that $\delta_{ij}$ has to satisfy:

1. $m > n$, there are more charging stations than drones. In this case, it is reasonable to assume that each station accepts at most one drone and that no drone may be assigned to some charging stations. The specification is described as

$$\sum_{j=1}^{m} \delta_{ij} = 1 \quad (i = 1, \ldots, n),$$

$$\sum_{i=1}^{n} \delta_{ij} = 1 \quad (j = 1, \ldots, m).$$

(13)

2. $m = n$, there is a one-to-one correspondence between drones and charging stations. In this case, the variables need to satisfy

$$\sum_{j=1}^{m} \delta_{ij} = 1 \quad (i = 1, \ldots, n),$$

$$\sum_{i=1}^{n} \delta_{ij} = 1 \quad (j = 1, \ldots, m).$$

(14)

3. $m < n$, the number of drones is larger than charging stations, where multiple drones may be
assigned to a charging station but too much congestion at a station needs to be possibly avoided. The specification can be described as
\[ \sum_{j=1}^{m} \delta_{ij} = 1 \ (i = 1, \ldots, n), \]
\[ \sum_{j=1}^{n} \delta_{ij} \geq 1 \ (j = 1, \ldots, m). \] (15)

Remark that all of the above formulations implicitly assume that only one drone can be charged at the same time at a station based on [11, 12]. However, this restriction can be trivially eliminated. Let us denote, by \( \bar{n} \), the maximal number of drones that can be charged simultaneously at station \( j \), and define \( n = \sum_{j=1}^{m} \bar{n}_j \). Then, we replace the right-hand sides of the bottom constraints in (13), (14), and (15) by \( \bar{n} \), and they are applied to the cases of \( n \leq \bar{n} \), \( n = \bar{n} \), and \( n > \bar{n} \), respectively. All of the subsequent results hold even under this modification.

To determine the optimal assignment of the charging stations, we formulate the optimization problem:
\[
\min_{\delta \in \{0, 1\}^{nm}} \sum_{i=1}^{n} \sum_{j=1}^{m} \theta_{ij} \delta_{ij} \quad \text{subject to:} \ (13), \ (14), \text{ or } (15).
\] (16)

In the problem, we must choose the parameter \( \theta = (\theta_{ij})_{i \in I, j \in J} \) appropriately so that it reflects the actual requirement. In this paper, we employ the function in (8) to design a penalty against assigning drone \( i \) to station \( j \). To this end, we define
\[
h_{ij, ch} = E_i - E_{\min} - \frac{K_{ch}}{k_{ch}} \left( \| p_i - s_j \| - d_{ch} \right),
\] (17)
where \( h_{ij, ch} \) gets low, possibly negative, when \( E_i \) is low and \( s_j \) is far from \( p_i \). Similarly to (9), when \( h_{ij, ch} \) is larger than 0, the battery level for drone \( i \) to visit charging station \( j \) is safe. Accordingly, drone \( i \) should be assigned to station \( j \) with a large \( h_{ij, ch} \). Based on this insight, \( \theta_{ij} \) in (16) should negatively correlated to \( h_{ij, ch} \). A trivial option is thus \( \theta_{ij} = 1 / h_{ij, ch} \), but it does not penalize a pair such that \( h_{ij, ch} < 0 \).

We thus employ a function such that
\[
\theta_{ij} = \begin{cases} 
\frac{1}{h_{ij, ch} + \epsilon}, & \text{if } h_{ij, ch} \geq 0, \\
1 - \frac{1}{h_{ij, ch} \epsilon'}, & \text{if } h_{ij, ch} < 0,
\end{cases}
\] (18)
where \( \epsilon \) and \( \epsilon' \) are very small positive values and \( \epsilon > \epsilon' \) so that \( h_{ij, ch} \mapsto \theta_{ij} \) is a monotonically decreasing function, as shown in Figure 2.

The problems (16) with (13), (14), and (15) can be reformulated as
\[
\min_{\delta \in \{0, 1\}^{nm}} \theta^T \delta \quad \text{subject to } F\delta = 1_n, \ G\delta \leq 1_m, \ (19a)
\]
\[
\min_{\delta \in \{0, 1\}^{nm}} \theta^T \delta \quad \text{subject to } F\delta = 1_n, \ G\delta \geq 1_m, \ (19b)
\]
\[
\min_{\delta \in \{0, 1\}^{nm}} \theta^T \delta \quad \text{subject to } F\delta = 1_n, \ G\delta \\ (19c)
\]

respectively, where
\[
\theta = \left[ \theta_{11} \cdots \theta_{1m} | \cdots | \theta_{n1} \cdots \theta_{nm} \right]^T,
\]
\[
\delta = \left[ \delta_{11} \cdots \delta_{1m} | \cdots | \delta_{n1} \cdots \delta_{nm} \right]^T,
\]
\[
F = \begin{bmatrix} 1^T_m & O \\ O & 1^T_m \end{bmatrix}, \ G = \begin{bmatrix} I_m & \cdots & I_m \end{bmatrix}.
\]

\( I_m \) is the \( m \)-by-\( m \) identity matrix and \( 1_n \ (1_m) \) is the \( n \) (\( m \)) dimensional all-one vector.

4. Solution to assignment problem

4.1. LP relaxation

In this section, we present solutions to the aforementioned optimization problems in (19). This kind of integer programming is well known to be NP-complete. In the case with large number of drones, the computational time can be a bottleneck for the optimal assignment of the charging station.

Now, it is trivial to see that \( [F^T \ G^T]^T \) is totally unimodular using Theorem 19.3 in [19], and \( [1^T_m \ 1^T_m]^T \) is an all-integer vector. In this case, all of the integer problems in (19) are known to be exactly relaxed to the linear programming problems:
\[
\min_{x \in \{0, 1\}^{nm}} \theta^T x \quad \text{subject to } Fx = 1_n, \ Gx \leq 1_m, \ (20a)
\]
\[
\min_{x \in \{0, 1\}^{nm}} \theta^T x \quad \text{subject to } Fx = 1_n, \ Gx = 1_m, \ (20b)
\]
\[
\min_{x \in \{0, 1\}^{nm}} \theta^T x \quad \text{subject to } Fx = 1_n, \ Gx \geq 1_m, \ (20c)
\]
where the binary constraint $\delta \in \{0, 1\}^{nm}$ is relaxed to $x \in [0, 1]^{nm}$. All of these problems are further reformulated in a unified form:

$$\min_{X \in \mathcal{X}} \Theta^T X \quad \text{subject to } AX = b,$$  \hspace{1cm} (21)

where $b := [1_1^T, 1_m^T]^T$. The definitions of $\Theta$, $\mathcal{X}$, and $A$ vary depending on the cases (13)–(15) as below.

(13): $\Theta = \begin{bmatrix} \theta \\ 0_m \end{bmatrix}$,

$\mathcal{X} = \left\{ X = \begin{bmatrix} x \\ y \end{bmatrix} \in \mathbb{R}^{(n+1)m} | x \in [0, 1]^{nm}, \ y \geq 0_m \right\}$,

$A = \begin{bmatrix} F \\ G \end{bmatrix}$,

(14): $\Theta = \theta$,

$\mathcal{X} = \left\{ X = x \in \mathbb{R}^{nm} | x \in [0, 1]^{nm} \right\}$,

$A = \begin{bmatrix} F \\ G \end{bmatrix}$,

(15): $\Theta = \begin{bmatrix} \theta \\ 0_m \end{bmatrix}$,

$\mathcal{X} = \left\{ X = \begin{bmatrix} x \\ y \end{bmatrix} \in \mathbb{R}^{(n+1)m} | x \in [0, 1]^{nm}, \ y \geq 0_m \right\}$,

$A = \begin{bmatrix} F \\ G \end{bmatrix}$,

where $0_m$ is the $m$-dimensional all-zero vector. It is thus sufficient to solve the unified problem (21). In the sequel, the dimension of $X$ is denoted by $N$.

We confirm improvement of the computational efficiency of the above relaxation by a simulation under a simpler scenario. To highlight only the computational time, we set the parameters used in (17) and (18) as $E_i = 0.6 \forall i, E_{\text{min}} = 0.15, K_{\text{ch}} = 0.03, k_{\text{ch}} = 0.1, d_{\text{ch}} = 0.125, \varepsilon = 0.001$, and $\varepsilon' = 0.0001$. Positions of drones and charging stations are randomly generated on a two-dimensional mission space of $[-1, 1] \times [-1, 1]$. Figure 3 shows how the assignment is optimized with 10 drones and 10 charging stations. The relaxed optimization problem (16) gives exactly the same assignment as the original integer program Equation (21).

We next compare the computation time between (16) and (21) under different numbers of drones and charging stations. Table 1 shows the average computational time of 1000 operations. We use the method of traverse to solve the integer optimization problem and linprog function in MATLAB to solve the linear programming problem. It can be observed that the linear programming is advantageous when $n = m$ is equal to or larger than 7.

| n  | 1   | 2   | 3   | 4   | 5   |
|----|-----|-----|-----|-----|-----|
| IP | 0.002 | 0.015 | 0.044 | 0.136 | 0.579 |
| LP | 4.709 | 5.04 | 5.10 | 5.131 | 5.173 |
| n  | 6   | 7   | 8   | 9   | 10  |
| IP | 3.308 | 22.266 | 178.593 | 1631.163 | 16439.127 |
| LP | 5.616 | 5.925 | 6.029 | 6.152 | 6.548 |

**4.2. Distributed solution based on ADMM**

For a large scale of problem, parallelization and distributed computing can greatly improve the efficiency so that we can take advantages of CPU’s multiple cores and multiple machines. Another remarkable benefit of the above LP relaxation is that its distributed solution is obtained by distributed optimization techniques. Based on this idea, we employ a promising distributed optimization algorithm called alternating direction method of multipliers (ADMM) [14].

The problem described in (21) can be reformulated into the following ADMM form:

$$\min_{X \in \mathcal{X}, Z \in \mathbb{R}^N} \Theta^T X + h(Z) \quad \text{subject to } X - Z = 0,$$  \hspace{1cm} (22)

where $Z \in \mathbb{R}^N$ is a decision variable, and $h : \mathbb{R}^N \rightarrow \{0, \infty\}$ is defined as

$$h(Z) := \begin{cases} 0, & \text{if } AZ = b, \\ \infty, & \text{otherwise.} \end{cases}$$  \hspace{1cm} (23)

**Figure 3.** Position of drones (red dots), charging stations (green dots), and their assignment (segments). The left figure shows result with integer programming problem and the right figure shows result with linear programming problem.
The augmented Lagrangian is formulated as
\[
L(X, \mu, Z) := \Theta^T X + h(Z) + \mu^T (X - Z) + \frac{\rho}{2} \|X - Z\|^2, \quad (\rho > 0),
\]
where \(\mu \in \mathbb{R}^N\) is the Lagrange multiplier associated with the equality constraint, \(X = Z\). The ADMM algorithm is then formulated as follows [14]:
\[
\begin{align*}
X[k+1] &= \arg \min_{X \in \mathbb{R}^n} L(X, \mu[k], Z[k]), \quad (25a) \\
Z[k+1] &= \Pi_C((1-\rho)Z[k] + \rho X[k+1] + \mu[k]), \quad (25b) \\
\mu[k+1] &= \mu[k] + X[k+1] - Z[k+1], \quad (25c)
\end{align*}
\]
where \(\Pi_C : \mathbb{R}^N \rightarrow \mathbb{R}^N\) denotes the projection to set \(C := \{X \mid AX = b\}\), and for \(Y \in \mathbb{R}^N\), it is defined as
\[\Pi_C(Y) := (I_N - A^\dagger A)Y + A^\dagger b,\]
where \(A^\dagger\) is a pseudo-inverse matrix of \(A\). Then, \(Z[k]\) in (25b) satisfies \(Z[k] \in C\) for all steps.

Let us now show that (25) is implemented in a partially distributed fashion. We first consider the cases of \(n > m\) and \(n < m\). The variables \(x, Z, \mu, \) and the parameter \(\theta\) are decomposed as
\[
x = [x_1^T \cdots x_m^T]^T, \quad x_i := [x_{i1} \cdots x_{im}]^T \in \mathbb{R}^m,
\]
\[
Z = [Z_x^T Z_y^T]^T, \quad Z_x = [Z_{x1}^T \cdots Z_{xm}^T]^T (Z_xi \in \mathbb{R}^m),
\]
\[
\mu = [\mu_x^T \mu_y^T]^T, \quad \mu_x = [\mu_{x1}^T \cdots \mu_{xm}^T]^T (\mu_xi \in \mathbb{R}^m),
\]
\[
\theta = [\theta_x^T \cdots \theta_n^T]^T, \quad \theta_i := [\theta_{i1} \cdots \theta_{im}]^T \in \mathbb{R}^m,
\]
where \(x_{ij}\) corresponds to \(\delta_{ij}\). The augmented Lagrangian is then decomposed as
\[
L(X, \mu, Z) = \sum_{i=1}^n L_i(x_i, \mu_x, Z_x) + h(Z)
\]
\[+ \mu_y^T (y - Z_y) + \frac{\rho}{2} \|y - Z_y\|^2, \quad L_i(x_i, \mu_x, Z_x) := \theta_i^T x_i + \mu_{xi}^T (x_i - Z_xi) + \frac{\rho}{2} \|x_i - Z_xi\|^2.
\]

Using the above local variables, parameters and local functions \(L_i\), the algorithm (25) is rewritten as
\[
\begin{align*}
x_i[k+1] &= \arg \min_{x_i \in \mathbb{R}^m} L_i(x_i, \mu_x, Z_x), \quad (26a) \\
y[k+1] &= \arg \min_{y \geq 0} \left(\mu_y^T (y - Z_y) + \frac{\rho}{2} \|y - Z_y\|^2\right), \quad (26b) \\
Z[k+1] &= \Pi_C((1-\rho)Z[k] + \rho X[k+1] + \mu[k]), \quad (26c) \\
\mu_x[k+1] &= \mu_x[k] + x[k+1] - Z[k+1], \quad (26d) \\
\mu_y[k+1] &= \mu_y[k] + y[k+1] - Z_y[k+1]. \quad (26e)
\end{align*}
\]

The update laws (26a) and (26d) are then locally executed by drone \(i\), while (26b), (26c), and (26e) are left to a central computer. The important feature of the present partially distributed architecture is that only the local update law (26a) relies on the time-varying \(\theta_i\) associated with the physical world. The same architecture is trivially obtained in the case of \(n = m\) also. Namely, in this case, the iterations (26a) and (26d) are run by drone \(i\) and (26c) is by the central computer.

The overall partially distributed control architecture is illustrated in Figure 4. The bottom block in the central computer (left) updates the density \(\phi(q, t)\) based on \(\{p_i\}_{i\in Z}\), which is needed since the density depends on the history of the coverage states similarly to [10,12]. In the right box, the algorithm is divided into two blocks. The top one assigns a charging station, which corresponds to (26a) and (26d), and the bottom computes the control input \(u_i\) based on (10). They exchange

![Figure 4](image_url)

**Figure 4.** Overall control architecture with persistent coverage control and ADMM-based distributed assignment of charging stations in the case of \(n > m\) and \(m > n\).
information of $h_{ij,ch}$ ($j \in J$) and the assignment variable $\delta_{ij}^{*}$ ($j \in J$). The top block in the right corresponds to (26b), (26c), and (26e), and it updates $Z, y$, and $\mu_y$ based on $X$ and $\mu_x$ collected from the drones to manage the constraint $AZ = b$.

**Remark 4.1:** It is undesirable that a drone is assigned to a charging station where another drone is charging, in the case of $n \leq m$. Such an event is avoided by just setting $E_i$ to $E_{\text{min}}$ during the charging in the algorithm of drone $i$ since drone $i$ can never visit other station from the state $E_i = E_{\text{min}}$ and at most one drone can be assigned to each station due to the bottom constraint in (13) or (14).

## 5. Simulation

In this section, we verify the effectiveness of the present distributed online assignment of charging stations.

In the simulation the mission space is set to a two-dimensional square plane of $[-2, 2] \times [-2, 2]$. The number of drones and charging stations is set to $n = m = 6$. Charging stations are evenly located at $(-1, 1), (1, -1), (-1, -1), (0, 1), (1, 1), $ and $(0, -1)$. Parameters of the simulation are described in Table 2, where we set the performance certificate parameter to $\gamma = 0$. In the simulation, each drone is charged with the speed $C_{ch}$ and stops charging if the energy reached the upper bound $\bar{E}$. The initial positions of the drones are randomly generated over the mission space as illustrated in Figure 5(a). The initial battery level of drones are set as $E_1(0) = 1.0$, $E_2(0) = 0.9$, $E_3(0) = 0.8$, $E_4(0) = 0.7$, $E_5(0) = 0.6$, and $E_6(0) = 0.5$, respectively. In the simulation, we always start iterations of ADMM from $\mu[0] = 0$ and $Z[0] = 0$, and stop the iterations if the number of elements of $X$ that are greater than 0.9 is equal to $n = m = 6$, all other elements are smaller than 0.1, and $\|X - Z\|_1 < 0.1$, or the number of iterations reaches an upper bound set to 500.

Figure 5 shows the snapshots of the simulation with the present distributed solution after 0, 1, 19, 35, 36, and 119 times of iteration, where the drones are illustrated by blue dots and charging stations by the green circles. The colour of mission space represents the values of the time-varying density function $\phi(q, t)$. The yellow regions indicate high densities, while the blue low densities. The sensing range of each drone is illustrated by the red circle. We see from these figures that the drones spread out over the mission space. It can be also observed, from (a) to (b) and from (d) to (e), that

![Figure 5. Snapshots of the simulation: (a) initial state, (b) 1st iteration, (c) 19th iteration, (d) 35th iteration, (e) 36th iteration, (f) 119th iteration.](image-url)
Figure 6. Assigned charging stations for a long duration.

Figure 7. Time responses of the battery levels of drones. The assignment of charging stations is updated successfully. In (f), the drone in the upper centre has stopped at the charging station to charge its battery. Figure 6 shows the assignments of the charging stations for a longer time interval, where horizontal lines represent changes in the charging stations assigned to a drone, with different colours representing different charging stations. At initial state, the plot for drone 1 shows grey, which implies the drone 1 is assigned to the charging station 6. We see that the present algorithm continues to reassign the stations on an ongoing basis. Figure 7 shows time responses of the battery levels, where the dashed line indicates the minimum battery level $E_{\text{min}}$. It is confirmed that the required level is maintained for all drones. It is concluded from the above results that the present algorithm works as expected. Remark that the speed of charging is much faster than the real. Nonetheless, we take this model since it is desired for drones to repeat battery charges many times within a limited time interval to validate the proposed algorithm. High fidelity simulation is left as future work.

We next discuss the performance of the present method, where we conduct comparative studies with the case without any online assignment of stations and the centralized solution to (21). Now, their performances will vary depending on the initial conditions. To get a general conclusion, we performed Monte Carlo

Figure 8. Average results of Monte Carlo experiments: (a) comparison of the summation of distances between drones and their charging stations and (b) comparison of the coverage performance.
simulations, where we run the above two algorithms and the present distributed solution 100 times for randomly generated initial positions of the drones with the same initial battery levels $E_1(0), \ldots, E_6(0)$ as above.

Figure 8 shows the time responses of $\sum_{i=1}^n \| p_i - q_i^* \|$ and the average of the values of the objective function $J$ obtained from the 100 simulations, where $q_i^*$ is the position of the charging station assigned to drone $i$ at each time. It can be observed from Figure 8(a) that the online assignment of the stations reduces the total sum of the distances for the drone-station pairs, which implies that the online assignment enhances the coverage performance. Figure 8(b) directly demonstrates the advantage of the online assignment in the coverage performance, where we see that the cases with the online assignment outperform that with fixed stations. In both figures, we also do not find any significant difference between centralized optimization and distributed optimization. In view of the limited information managed by the central computer, we conclude that the present distributed solution is superior to the centralized one.

6. Conclusion

In this paper, we addressed online assignment of charging stations for drones in the scenario of persistent coverage control task. We started formulating the assignment problem as a linear integer programming. Then, we designed the cost function based on the CBF for battery charging. The integer programming problem was then relaxed to a linear programming owing to the total unimodularity of the constraint matrix. We furthermore reformulated the relaxed problem as a distributed optimization form and adopted ADMM algorithm to solve it in a distributed manner. Finally, the effectiveness of distributed online assignment of charging stations was verified through Monte Carlo simulations.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported in part by Japan Society for the Promotion of Science (JSPS) KAKENHI under Grant 21K04104.

Notes on contributors

Zhiyuan Lu received B.S. degree from Southwest Jiaotong University, China in 2017 and M.S. degree from Tokyo Institute of Technology, Japan in 2021. He is currently a Ph.D. student at Tokyo Institute of Technology. His research interests include multi-robot systems.

Shunya Yamashita received the B.Eng., M.Eng. and Ph.D. degrees from Tokyo Institute of Technology in 2017, 2019 and 2022, respectively. Since April 2022, he is a Postdoctoral Fellow of the Japan Society for the Promotion of Science. He received the Takeda Prize (2020) and Outstanding Paper Award (2020) from The Society of Instrumental and Control Engineers. His research interests include distributed optimization, population game and passivity-based control.

Junya Yamauchi received the Ph.D. degree from Tokyo Institute of Technology in 2018. He was an assistant professor at Tokyo Institute of Technology from 2018 to 2020 and is currently an assistant professor at the University of Tokyo from 2020. His research interests include autonomy in multi-robot systems.

Takeshi Hatanaka received the Ph.D. degree in applied mathematics and physics from Kyoto University in 2007. He then held faculty positions at Tokyo Institute of Technology and Osaka University. Since April 2020, he is an associate professor at Tokyo Institute of Technology. He is the coauthor of “Passivity-Based Control and Estimation in Networked Robotics” (Springer, 2015), and the coeditor of “Economically-enabled Energy Management” (Springer Nature, 2020). His research interests include cyber-physical & human systems and networked robotics. He received the Kimura Award (2017), Pioneer Award (2014), Outstanding Book Award (2016), Control Division Conference Award (2018), Takeda Prize (2020), and Outstanding Paper Awards (2009, 2015, 2020, 2021) all from The Society of Instrumental and Control Engineers (SICE). He is serving/served as an AE for IEEE TSCCT, Advanced Robotics, and SICE JCMSI, and is a member of the Conference Editorial Board of IEEE CSS. He is a senior member of IEEE.

References

[1] Zhou G, Li C, Cheng P. Unmanned aerial vehicle (UAV) real-time video registration for forest fire monitoring. Proc. 2005 IEEE International Geoscience and Remote Sensing Symposium; 2005. p. 1803–1806.
[2] Rutkaukas M, Asenov M, Ramamoorthy S, et al. Autonomous multi-species environmental gas sensing using drone-based fourier-transform infrared spectroscopy. Opt Express. 2019;27(7):9578–9587.
[3] Chung S, Paranjape AA, Dames P, et al. A survey on aerial swarm robotics. IEEE Trans Robot. 2018;34(4):837–855.
[4] Cortés J, Martínez S, Bullo F. Spatially-distributed coverage optimization and control with limited-range interactions. ESAIM: Control, Optim Ca. EDP Sciences. 2005;11(4):691–719.
[5] Hayashi N, Segawa K, Takai S. 2D voronoi coverage control with Gaussian density functions by line integration. SICE J Control Meas Syst Integr. 2017;10(2):110–116.
[6] Takeuchi N, Naruse M, Sekiguchi K, et al. Autonomous patrol and invader detection by coverage controlled quadcopters. Proc. 2019 12th Asian Control Conference; 2019. p. 1119–1124.
[7] Schwager M, Julian BJ, Angermann M, et al. Eyes in the sky: decentralized control for the deployment of robotic camera networks. Proc IEEE. 2011;99(9):1541–1561.
[8] Funada R, Santos M, Yamauchi J, et al. Visual coverage control for teams of quadcopters via control barrier functions. Proc. 2019 International Conference on Robotics and Automation; 2019. p. 3010–3016.
[9] Bentz W, Hoang T, Bayasgalan E, et al. Complete 3-D dynamic coverage in energy-constrained multi-UAV sensor networks. Auton Robots. 2018;42(4):825–851.
[10] Hübel N, Hirche S, Gurious A, et al. Coverage control with information decay in dynamic environments. IFAC Proc. 2008;41(2):4180–4185.
[11] Egerstedt M, Pauli JN, Notomista G, et al. Robot ecology: constraint-based control design for long duration autonomy. Annu Rev Control. 2018;46:1–7.
[12] Dan H, Hatanaka T, Yamauchi J, et al. Persistent Object Search and Surveillance Control with Safety Certificates for Drone Networks Based on Control Barrier Functions. Front Robot AI. [cited 2021 Oct 8], p. 1–13.
[13] Ames AD, Xu X, Grizzle JW, et al. Control barrier function based quadratic programs for safety critical systems. IEEE Trans Automat Contr. 2017;62(8):3861–3876.
[14] Boyd S, Parikh N, Chu E, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. Found Trends Mach Learn. 2010 Jan;3(1):1–122.
[15] Nguyen DH, Azuma S, Sugie T. Novel control approaches for demand response with real-time pricing using parallel and distributed consensus-based ADMM. IEEE Trans Ind Electro. 2019;66(10):7935–7945.
[16] Hidaka N, Miyamoto T, Sugihara H. An ADMM approach to distributed optimization in distribution system considering fairness among consumers. Proc. 2019 IEEE International Conference on Systems, Man and Cybernetics; 2019; p. 2254–2258.
[17] Namba T, Funabiki S, Takaba K. Stochastic distributed model predictive control of microgrid with uncertain PV power prediction. SICE J Control Measure Syst Integr. 2021;14(1):39–50.
[18] Zhiyuan L, Yamashita S, Hatanaka T. Persistent environmental monitoring with distributed online assignment of charging stations. Proc. SICE International Symposium on Control Systems 2022; Japan: [cited 2022 Mar 8–10]. online.
[19] Schrijver A. Theory of linear and integer programming. Chichester: John Wiley & Sons; 1998.