Desperate Coverage Problem in Mission-driven Camera Sensor Networks

Yi Hong†, Junghyun Kim‡, Donghyun Kim‡, Deying Li†§, Alade O. Tokuta‡

† Key Laboratory of Data Engineering and Knowledge Engineering (Renmin University of China), MOE, School of Information, Renmin University of China, Beijing 100872, China
‡ Department of Mathematics and Physics, North Carolina Central University, Durham 27707, USA
§ Corresponding author, Email: deyingli@ruc.edu.cn

Abstract. Recently, camera sensor network is attracting huge amount of attention due to the growing popularity of multimedia applications. This paper investigates a new scheduling problem in camera sensor network whose goal is to cover a set of targets as efficiently as possible during a given mission period. In particular, we consider a desperate situation in which we may not have enough camera sensors to cover all of the targets of interest during the mission. The goal of our problem of interest is to schedule the sensors such that (a) the number of the most important targets which are fully covered during the mission period is maximized, and (b) the overall target-temporal coverage which is defined as the gross sum of the weight of each target multiplied by the time period when the target is covered is maximized. We formally introduce the desperate coverage problem in mission-driven camera sensor networks (DCP-MCSN) and propose a new heuristic algorithm for this NP-hard problem. To evaluate the performance of the proposed algorithm, we compare it with a theoretical upper bound. Our simulation result shows that our algorithm performs very close to the upper bound as well as outperforms the existing alternative.

Keywords: Camera sensor network, target-temporal coverage, coverage problem, mission-driven.

1 Introduction

Thanks to the growing popularity of multimedia applications of wireless sensor networks, camera sensor networks, which are wireless networks of camera sensors, are getting more attentions very recently. A camera sensor node is a kind of directional sensor node whose sensing range resembles a sector of a disk. In Fig. 1, the sector with a solid line is the area covered by a directional sensor $s_i$. Therefore, $s_i$ is covering a target $u$. The beamwidth $\theta$ is dependent on the sensor’s physical characteristics. $d_{(i,j)}$ is a vector from $s_i$ to the center of the surface of the $j$th sector. However, the design and implementation of camera sensor networks involve additional challenges on the top of the complexity of directional sensor network due to several key features of camera sensor networks.
For instance, a picture of the occiput of a person is hardly useful to verify the identity of the person, which implies that a camera sensor node may not be able to cover an object within its sensing sector depending on the facing direction of the object. As shown in Fig. 2, a target $u$ is covered by two camera sensors $s_1$ and $s_2$. The inner angle between the viewing orientation of $s_1$ and $u$’s facing orientation, $\theta_1$, is smaller than that between the viewing orientation of $s_2$ and $u$’s facing orientation, $\theta_2$. As a result, $s_1$ can capture a more recognizable image of $u$ than $s_2$.

Most wireless sensor nodes are battery operated and it is very difficult or expensive to replace or recharge the battery. As a result, energy efficiency has been a major issue of wireless sensor networks. One popular approach to extend the lifetime of wireless sensor network is to exploit the redundancy of coverage in wireless sensor network. That is, in many application scenarios, sensor nodes are randomly deployed and thus it is highly likely that a target of interest can be monitored by more than one node at the same time. Then, we can make a sleep-wakeup schedule of the nodes which can cover the same target and activate them one by one while the rest of them fall asleep. In the literature, the problem of computing a sleep-wakeup schedule of sensor nodes such that the total time period satisfies a certain coverage quality requirement is referred as a “coverage problem”. One popular coverage problem is the full-coverage problem whose goal is to maximize the lifetime of a sensor network while seamlessly covering an area or targets of interest [7,8].

Most coverage problems in wireless sensor networks have focused on the lifetime maximization issue. In those problems, the lifetime of the networks is an optimization goal, not a constraint. However, this is not always the case in reality. For instance, consider an application of wireless sensor networks in which we have to cover an area of interest or a set of targets during “a given mission period” (which is a constraint), but we do not have enough sensors to fully cover the whole area or the targets regardless from the scheduling algorithm we use. Certainly, the existing coverage algorithms that we discussed so far cannot deal
Fig. 2. The sensing model of the camera sensor

with this situation. Recently, Liu and Cao introduced one way to handle this challenging issue. Given a required surveillance mission period, they suggested to maximize the spatial-temporal coverage of the area or targets of interest, which is defined as the gross sum of the total time that each distinct area is covered multiplied by the size of the area (in case of area coverage) [9] or the gross sum of the weight of each target covered multiplied by the total time duration that the target is covered (in case of target coverage) [10]. Very recently, Hong et al. extended this idea and investigated a spatial-temporal coverage problem in camera sensor networks [1].

In this paper, we introduce another way to deal with the question of how to provide the best quality coverage over a set of targets of interest during a given mission period with insufficient number of camera sensor nodes. This study is motivated by our observation that when we solely aim to maximize the coverage quality (the total target-temporal coverage) like [1, 10], it is possible that some targets with the highest priority could be de-prioritized by a number of insignificant targets, and as a result, relatively ignored in the course of the schedule building process. The problem of monitoring a warehouse with two entrances during a certain time period is a good example. Clearly, the entrances are the most important targets since the face of any trespasser can be easily detected by a camera observing the entrance. Meanwhile, a shelf inside the warehouse without any item is clearly a less-important target. To deal with this situation, we suggest to provide full-coverage to as many higher priority targets as possible during the mission period while the spatial-temporal coverage over the rest of targets can be maximized. We would like to emphasize that we are not trying to refute the significance of the target-temporal-maximization-only approach used in [1,10], but we attempt to propose a new complement for those approaches to deal with this desperate situation. The main contributions of this paper can be summarized as follows.

(a) We introduce a new scheduling problem in camera sensor networks, namely desperate coverage problem in mission-driven camera sensor networks (DCP-MCSN). More formal description of the problem is given in Section 4.
We propose a new heuristic algorithm for our new NP-hard problem, called desperate coverage algorithm in mission-driven camera sensor networks (DCA-MCSN). In detail, given a set of camera sensors, a set of targets, and a mission period, DCA-MCSN performs a binary search (see Fig. 3) to find a constant \( k \) such that the most \( k \) important targets are fully covered during the mission period. During the search process, \( k \) is assumed to be a some constant (e.g. initially \( k \leftarrow \lceil \frac{m}{2} \rceil \)) and Network-Scheduler (Algorithm 3) is executed. Network-Scheduler internally calls two sub-procedures Sector-Decider (Algorithm 1) and Sensor-Scheduler (Algorithm 2) to determine the sensing sector of each camera sensor and the activation schedule of each camera sensor. Dependent on Network-Scheduler, the \( k \) value is increased or decreased and the whole process is repeated until we finalize \( k \) and obtain a schedule which allows the camera sensors to fully cover the most \( k \) important targets and the target-temporal coverage is maximized.

We compare the average performance of DCA-MCSN with the theoretical upper bound in simulation as well as with the only existing competitor in [1]. Our simulation result shows that our algorithm outperforms the target-temporal-maximization-only algorithm in camera sensor network in [1] in terms of the number of the most important targets which are fully-covered during the whole mission period.

The rest of this paper is organized as follows. Section 2 present related work. Some preliminaries are introduced in Section 3. We introduce the formal definition of DCP-MCSN and our algorithm for this problem in Section 4. Our simulation result is presented in Section 5. Finally, we conclude this paper in Section 6.
2 Related Work

Frequently, the problem of constructing a sleep-wakeup schedule of a wireless sensor network with the goal of maximizing the continuous monitoring time to meet a given coverage quality requirement is referred as a coverage problem [8–11]. Most types of wireless sensor networks thoroughly investigated in the literature consider sensor nodes with omnidirectional sensing range. Consequently, the area covered by those sensor nodes resembles a disk with the node in the center of it. In contrast, the sensing area covered by a directional sensor node more looks like a sector of a disk [2]. In [12], Cai et al. has investigated the multiple directional cover set problem in directional sensor networks, whose objective is to compute a sleep-wakeup schedule of a given set of sensor nodes such that the time for the sensor network to fully cover a given set of targets is maximized. In the meantime, Ai et al. [13] studied a new coverage problem in directional sensor networks whose goal is given a set of sensor nodes with adjustable camera orientations, to determine the direction (orientation) of each sensor so that the number of covered targets is maximized (primary objective) while the number of sensor nodes used is minimized (secondary objective). In additions, the other aspects of directional sensor networks are also investigated such as vulnerability [14], fault-tolerance [16], and so on [15,19].

Briefly speaking camera sensor network is a kind of directional sensor network and thus they share several properties. However, due to several unique features of camera sensors, especially the effective-sensing surveillance model in Definition 1, the scheduling algorithms for directional sensor networks are not directly applicable to camera sensor networks. Most importantly, a camera sensor node can cover a target only if the face of the target can be seen by the sensor. In [5], Liu et al. introduced the directional $k$-coverage problem in camera sensor networks whose objective is minimizing the number of camera sensors to cover an area of interest such that any spot in the area can be monitored by at least $k$ different cameras. In [19], Wang and Cao assumed that a camera sensor network version of the full coverage problem, called the full-view coverage problem. In this problem, a target is full-view covered by a camera sensor network if we can capture the face of a target which is within the sensing range of the camera sensor network independent of the face direction of the target. Previously, Ma et al. extended the result in [19] and investigated the full-view barrier coverage problem in camera sensor networks.

So far, most coverage problems in wireless sensor networks have concerned about the lifetime maximization issue. Therefore, they are not applicable to a desperate situation in which we are required to cover an area or a set of targets of interest during a given mission period. One possible solution of this challenging issue is introduced by Liu and Cao [9,10], who suggested to maximize the spatial-temporal coverage of the area or targets of interest. In our previous work, we extended this idea and investigated a spatial-temporal coverage problem in camera sensor networks [1]. However, Liu and Cao’s approach comes with a non-neglectible shortcoming - a target with the highest priority could be de-prioritized by a number of insignificant targets, and as a result, is ignored.
by a scheduling algorithm. In this paper, we address this issue and attempt to propose an alternative solution of the problem as a complement of Liu and Cao’s approach.

3 Preliminaries

In this section, we review two important preliminaries of our paper, the effective-sensing model in camera sensor networks [5], and the Identifiability Test [1], which is a procedure to check, given a set of targets and a set of camera sensors, if the target is effectively-covered by the camera sensors. As we mentioned earlier, this paper assumes that a target is recognized or effectively-sensed (covered) by a camera sensor if the facial view of the target is observed by a camera sensor. the face direction of each target can be estimated by an existing head pose estimating strategy such as the one in [6]. More formal definition of the effective-sensing model in camera sensor networks is as follow.

Definition 1 (Effective-sensing Model). Consider a target $t_k$ located at $(x_k, y_k)$ and a sensor $s_i$ located at $(x_i, y_i)$. $t_k$ is effectively-sensed (covered) by $s_i$ if $t_k$ is within the sensing area of $s_i$ and the internal angle between two vectors $f_k$ and $v_{(k,i)}$ is no greater than $\phi$, i.e. $\alpha(f_k, v_{(k,i)}) \leq \phi$ (see Fig. 4), where vector $f_k$ is the facing orientation of $t_k$, $v_{(k,i)} = (x_i - x_k, y_i - y_k)$, and $\phi \in [0, \frac{\pi}{2})$ is a predefined parameter called the max viewing angle.

In this paper, we consider a camera sensor network of $n$ camera sensors and $m$ targets of interest. Each camera sensor is with uniform beamwidth $\theta$ and with $q \geq 1$ available sectors, i.e. $q = \lceil \frac{2\pi}{\theta} \rceil$. We denote the face direction of a target $t_k$ by a vector $f_k$. By the effective-sensing model in Definition 1, we can define the conditions to be satisfied for a target $t_k$ to be effectively-covered by a camera sensor $s_i$ with its $j_{th}$ sector as below.

Definition 2 (Identifiability Test). A target $t_k$ is effectively-covered by the $j_{th}$ sensing sector of a camera sensor $s_i$ only if it passes all of the the following three sub-tests.

Sub-test 1: check whether $t_k$ is in the sensing range of $s_i$, i.e. check if $\|v_{(k,i)}\| \leq R$ is true, where $R$ is the maximum sensing range of $s_i$.

Sub-test 2: check whether $t_k$ is within the $j_{th}$ sensing sector of $s_i$, i.e. check if
Table 1.

<table>
<thead>
<tr>
<th>Terms, Symbols</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S$</td>
<td>a set of homogenous camera sensor nodes ${s_1, ..., s_n}$</td>
</tr>
<tr>
<td>$T$</td>
<td>a set of targets ${t_1, ..., t_m}$</td>
</tr>
<tr>
<td>$W$</td>
<td>the set of the weights of the targets in $T$ ${w_1, \cdots, w_m}$</td>
</tr>
<tr>
<td>$\theta$</td>
<td>the uniform field-of-vision of the camera sensors</td>
</tr>
<tr>
<td>$R_s = 1$</td>
<td>the normalized sensing range of the camera sensors</td>
</tr>
<tr>
<td>$R_t = 1$</td>
<td>the transmission range of the camera sensors with omnidirectional transmission radios</td>
</tr>
<tr>
<td>$L_i$</td>
<td>the battery lifetime of $s_i$</td>
</tr>
<tr>
<td>$f_{tk}$</td>
<td>the face direction vector of $t_k$</td>
</tr>
<tr>
<td>$B, r$</td>
<td>the length of a given mission and the length of each cycle (we assume $\frac{B}{r}$ is an integer for simplicity)</td>
</tr>
<tr>
<td>$q$</td>
<td>$2\pi/\theta$, the number of available orientations for each sensor</td>
</tr>
<tr>
<td>$d_{(i,j)}, S_{(i,j)}$</td>
<td>the vector of $s_i$’s $j$th available orientation and the set of targets covered by $s_i$ using its $j$th sector</td>
</tr>
<tr>
<td>$F$</td>
<td>a collection of the subsets of ${S_{(i,j)}</td>
</tr>
</tbody>
</table>

$d_{(i,j)} \cdot v_{(k,i)} \geq \|v_{(k,i)}\| \cdot \cos \frac{\theta}{2}$ is true, where $d_{(i,j)}$ is a vector from $s_i$ to the center of the surface of the $j$th sector.

**Sub-test 3:** check whether $t_k$ is effectively-covered by the $j$th sensing sector of $s_i$, i.e. check if $f_{tk}^T \cdot v_{(k,i)} \geq \|v_{(k,i)}\| \cdot \cos \phi$ is true.

From now on, $S_{(i,j)}$ will be used to represent (a) the set of targets effectively-covered by the $j$th sensing sector of $s_i$, and (b) the $j$th sensing sector of $s_i$, depending on the context. Note that in the first case, all $S_{(i,j)}$s can be determined within a polynomial time.

## 4 Desperate Coverage Problem in Mission-driven Camera Sensor Networks (DCP-MCSN) and Its Solution

Now, we introduce the formal definition of DCP-MCSN in Section 4.1 and our solution for this problem, namely desperate coverage algorithm in mission-driven camera sensor networks (DCA-MCSN) in Section 4.2. Table 1 summarizes the terms, notations, and their semantics used in this paper.

### 4.1 Formal Definition of DCP-MCSN

**Definition 3 (DCP-MCSN).** DCP-MCSN is to find a subcollection (schedule) $Z \subset F$ such that (a) for each sensor node $s_i$, at most one sector $S_{(i,j)}$ is used to form the subsets in $Z$, (b) the number of the first $k$ pivotal targets which are fully covered during $B$ is maximized (primary objective), and (c) the overall target-temporal coverage is maximized (secondary objective).
**Theorem 1.** DCP-MCSN is NP-hard.

**Proof.** A simpler form of DCP-MCSN without the second requirement regarding maximizing the number of the first $k$ pivotal targets is already proven to be NP-hard in [1]. Therefore, this problem in NP-hard.

### 4.2 Desperate Coverage Algorithm in Mission-driven Camera Sensor Networks (DCA-MCSN)

In this section, we introduce our algorithm for DCP-MCSN, namely DCA-MCSN. Given a set of camera sensors, a set of targets, and a mission period, DCA-MCSN performs a binary search (see Fig. 3). In detail, initially the number of the most important targets which can be seamlessly covered by the camera sensors is set to $k \leftarrow \lceil \frac{m}{2} \rceil$, and execute Network-Scheduler (Algorithm 3), which internally invokes two sub-procedures Sector-Decider (Algorithm 1) and Sensor-Scheduler (Algorithm 2). At the end, if Network-Scheduler returns a schedule of camera sensors such that the most $k$ important targets are fully covered during the mission period, we set $k \leftarrow \lfloor (m + \frac{m}{2})/2 \rfloor$ and execute Network-Scheduler. Otherwise, we set $k \leftarrow \lfloor (\frac{m}{2} - 1)/2 \rfloor$ and execute Network-Scheduler. At the end the search will be terminated, and we will find the maximum $k$ that can be achieved by our algorithm as well as its corresponding schedule. We would like to emphasize that similar to our scheduling strategy in [1], the mission period $B$ is divided into multiple cycles with a uniform length $r$, and Network-Scheduler is invoked to construct a schedule for a short term with $r$ unit time length. As a result, Network-Scheduler is invoked for $\frac{B}{r}$ times during the total mission period $B$. In the followings, we introduce the description of Sector-Decider, Sensor-Scheduler, and Network-Scheduler.

**Sector-Decider** Algorithm 1 is the formal description of Sector-Decider. The input of this algorithm is the set of camera sensors $S$, the set of targets $T$, and the set of the most important $k$ targets (those with the heaviest weights) in $T$. The goal of this greedy algorithm is to determine one active sector from all available ones for each camera sensor. In Lines 1-3, for each sector of each camera sensor, the set $S(i,j)$ of targets which can be effectively covered by a sensor $s_i$ with its $j$ sector is computed.

In detail, in the preliminary phase (Lines 5-7), for each target $a_k$ among the first $k$ pivotal targets, the accumulated time recorder $C_k$ that this target is effectively covered by some camera sensor is set to 0. Then, the loop spanning over Lines 8-24 is repeated to determine the set of sectors $Z$ such that one sector (represented by those targets, e.g. $S(i,j)$, which are covered by the sector) is selected from each sensor node. This loop largely consists of two parts:

(a) The first part (Lines 9-15) is for the case that there exists at least one target among the most important $k$ ones which may not be fully covered during the whole period $r$. And this part is used to determine an active sector for
Algorithm 1 Sector-Decider $(S, T, \{a_{m_1}, \ldots, a_{m_k}\})$

1: for each $(i, j)$ pair, $1 \leq i \leq n, 1 \leq j \leq q$ do
2:  Run Identifiability Test and compute $S_{(i,j)}$, the set of targets effectively covered by $s_i$ with its $j$th sector.
3: end for
4: Set $Z \leftarrow \emptyset, UC_k \leftarrow \{a_{m_1}, \ldots, a_{m_k}\}, UC' \leftarrow T \setminus UC_k$, and $UW \leftarrow S$.
5: for each target $a_k' \in UC_k$ do
6:  $C_k' \leftarrow 0$.
7: end for
8: loop
9:  $(i, j) \leftarrow \arg \max_{i,j} |S_{(i,j)} \cap UC_k|$.
10:  if $S_{(i,j)} \cap UC_k \neq \emptyset$ then
11:    $Z \leftarrow Z \cup \{S_{(i,j)}\}, UW \leftarrow UW \setminus \{s_i\}, UC' \leftarrow UC' \setminus S_{(i,j)}$.
12:  for each $a_k \in S_{(i,j)} \cap UC_k$ do.
13:    $C_k' \leftarrow C_k' + l_i = C_k' + L_i / B_r$.
14:  end for
15:  $UC_k \leftarrow UC_k \setminus \{a_k' \mid C_k' \geq r, \text{ where } k' \in \{m_1, \ldots, m_k\}\}$.
16: else
17:  $(i', j') \leftarrow \arg \max_{i',j' \leq q} |S_{(i',j')} \cap UC'|$.
18:  if $S_{(i',j')} \cap UC' = \emptyset$ then
19:    break; /* quit the loop */
20:  end else
21:  $Z \leftarrow Z \cup \{S_{(i',j')}\}, UW \leftarrow UW \setminus \{s_i'\}, UC' \leftarrow UC' \setminus S_{(i',j')}$.  
22: end if
23: end if
24: end loop
25: Return $Z$.

an undecided sensor node such that there are more uncovered targets among the most important $k$ targets which will be covered. Once we found such a sector, e.g. $S_{(i,j)}$, and add this to $Z$. Line 13 is used to update the total time that the most important $k$ targets can be covered by the sectors in $Z$ so far.

(b) The second part (Lines 17-22) is for the case that all the most important $k$ targets could be fully covered during the whole period $r$. This part is used to add more sectors to $Z$ such that the total target-temporal coverage can be maximized.

Sensor-Scheduler Algorithm 2 is the formal description of SENSOR-SCHEDULER. The inputs of this algorithm are

(a) a single sector $c_i = S_{(i,j)}$ of a sensor node $s_i$ for some $j$ selected by SECTOR-DECIDER,

(b) $N_i$ which is the set of sectors, e.g. $S_{(i,j)} s_i$, selected by SECTOR-DECIDER such that there exist some targets which can be covered by both the selected sector of $s_i$ and another sector in $N_i$,
Algorithm 2 Sensor-Scheduler \((c_i, N_i, Z', Z, T_k, Sch)\)

1: \(Sch_i \leftarrow \emptyset\).

2: if \(\exists \alpha_k \in c_i \cap T_k\) such that \((t_{x'} - red_{x'}) < r\), where
   
   1. \(t_{x'} = \sum_{c_i \in A(k') \setminus Z} l_f\),
   2. \(red_{x'} = \sum_{c_x, c_y \in A(k') \setminus Z', \ \ A \neq y} \max(\min(c_x, f, c_y, f) - \max(c_x, b, c_y, b), 0)\), and
   3. \(A(k')\) is the set of selected sectors by SECTOR-DECIDER effectively covering \(\alpha_k \in T\).

   then

3: \(Sch_i \leftarrow Sch_i \cup \{\max_{c_i, f' \in N_i \cap A(k') \setminus Z} \{c_i, f'\} | \forall \alpha_k' \in c_i \cap T_k\}\).

4: else

5: \(Sch_i \leftarrow Sch_i \cup \{c_i, b, c_i, f, c_i, b - l_i, c_i, f - l_i | \forall c_i' \in N_i \setminus Z'\} \cup \{0, r\}\).

6: end if

7: Find the \(c_i, b_0\) satisfying \(\Delta TC_i = \min_{c_i, b \in Sch_i} red_i\), where \(red_i = \sum_{c_j \in N_i \setminus Z'} (\max(\min(c_i, f, c_j, f) - \max(c_i, b, c_j, b), 0) \cdot \sum_{a_x \in c_i, c_y} w_{a_y})\). Note that if \(c_i, f\) is larger than the length of the round \(r\), we should rewrite the original working period \([c_i, b, c_i, f]\) as \([c_i, b, r]\) and \([0, c_i, f - r]\).

8: Return \(\langle c_i, b_0, c_i, f_0 \leftarrow c_i, b_0 + L_i/(\frac{a_i}{r})\rangle\).

(c) a sub-collection \(Z'\) of \(Z\),
(d) \(Z\) itself,
(e) the set \(T_k\) of the most important \(k\) targets, and
(f) \(Sch\), which is the schedule of cameras which are determined so far.

The goal of this algorithm is to determine the schedule of sector \(c_i\) within each short term with \(r\) unit time. The most important part is to make sure that the most important \(k\) targets are getting coverage during the whole period \(r\). For this purpose, Line 2 checks if there exists a target in \(T_k\) which has not received sufficient coverage based on the schedule \(Sch\) (which is still under consideration), but still can be covered by \(c_i\). If so, we add the latest finish time of sensors in \(Sch\) which covers such a target to \(Sch_i\) as the beginning time of \(c_i\) (Line 3). If no such a target exist, we just try to find a point to start to use \(c_i\) so that the target-temporal coverage is maximized. At the end, the activation period of \(c_i\), which is represented by two moments \(c_i, b_0\) and \(c_i, f_0\) are returned. Note that we utilize a new metric, namely the target-temporal coverage redundancy of a camera sensor node \(red_i\) in Line 7 of Algorithm 2. This metric is used to evaluate how much the node \(c_i\) is wasted by monitoring targets which are already monitored by the other nodes. By rescheduling the current node such that the overall target-temporal coverage redundancy can be decreased, we can improve the quality of the current schedule.

Network-Scheduler Based on the above two sub-procedures SECTOR-DECIDER and SENSOR-SCHEDULER, NETWORK-SCHEDULER’s formal description is shown in Algorithm 3. The inputs of this algorithm is the mission period \(B\), the set of camera sensors \(S\), the set of targets \(T\), the weight of the targets \(W\), and the
Algorithm 3 Network-Scheduler \((B, S, T, W, T_k \{a_{m_1}, \cdots, a_{m_k}\})\)

1: \(Z \leftarrow \text{SECTOR-DECIDER} (S, T, T_k). \quad TC \leftarrow 0.\)
2: \ifD for each \(x \in \{i|S_{i,j} \in Z\} \ifD \text{do}\)
3: \(c_x \leftarrow \emptyset, Z' \leftarrow \emptyset, T_k \leftarrow \{a_{m_1}, \cdots, a_{m_k}\}, \text{and } N_i \leftarrow \emptyset.\)
4: \text{end for}\)
5: \ifD for each \(S_{i,j} \in Z \ifD \text{do}\)
6: \(c_i \leftarrow S_{i,j}), Z' \leftarrow Z' \cup \{c_i\}, \text{and } N_i \leftarrow \{c_x|c_x \in Z \setminus c_i \text{ and } c_i \cap c_x \neq \emptyset\}.\)
7: \text{end for}\)
8: \ifD for each \(s_i \in S \ifD \text{do}\)
9: \(Sch \leftarrow \emptyset \text{ and } TC_i \leftarrow 0.\)
10: \text{end for}\)
11: \ifD while \(Z' \neq \emptyset \ifD \text{do}\)
12: \(\text{Construct a maximal independent set } M(Z') \text{ of } Z'.\)
13: \ifD for each \(c_i \in M(Z') \ifD \text{do}\)
14: \(\langle c_i, b, c_i, f \rangle \leftarrow \text{SENSOR-SCHEDULER} (c_i, N_i, Z', T_k, Sch).\)
15: \text{end for}\)
16: \(Z' \leftarrow Z' \setminus M(Z'), \quad \text{Sch} \leftarrow \text{Sch} \cup \{(c_i, b, c_i, f)|\forall c_i \in M(Z')\}.\)
17: \text{end while}\)
18: \ifD for each \(x \in \{a_{m_1}, \cdots, a_{m_k}\} \ifD \text{do}\)
19: \ifD if \(a_x \) is not fully covered during \(r \) time slots based on \(Sch \) then\)
20: \(\text{Return } \langle \text{FALSE, } 0, 0, -1, \rangle.\)
21: \ifD end if\)
22: \text{end for}\)
23: \(TC \leftarrow \sum_{k \in \{1, \cdots, m\}, \{m_1, \cdots, m_k\}} w_k \cdot (t_k - \text{red}_k) + r \cdot \sum_{k \in \{m_1, \cdots, m_k\}} w_k, \text{ where}\)
24: 1. \(t_k = \sum_{i \in A(k)} t_i, \text{ and}\)
25: 2. \(\text{red}_k = \sum_{i, j \in A(k) \land i \neq j} \max\{\min\{c_i, f, c_j, f\} - \max\{c_i, b, c_j, b\}, 0\}.\)
26: \text{Return } \langle \text{TRUE, } Z, \text{Sch}, TC \rangle.\)

set of the most important \(k \) targets, \(\{a_{m_1}, \cdots, a_{m_k}\}. \) The outcome of this algorithm consists of four tuples. The first one is either true or false. It is true if the schedule generated by this algorithm is providing a full-coverage of the most important \(k \) targets during the while mission period \(B. \) Otherwise, it returns false. The second item returned is \(Z \) which includes one sector for each camera sensor to achieve the target-temporal coverage \(TC, \) which is the fourth item of the output of this algorithm. The third one is \(Sch, \) which is the schedule of each camera sensor.

This algorithm is based on the assumption that all camera sensor nodes are already scheduled, i.e. the activation period of each camera sensor node is already determined randomly, and we try to refine the schedule of each node (one by one) based on the current schedule of the other nodes such that the overall target-temporal coverage can be maximized. Note that the pre-assignment of the schedule of all nodes can be done via assign each node’s beginning time with 0. Based on the pre-assignment of the schedule, we can improve the quality of it by iteratively identifying a node which can make the overall target-temporal coverage be increased and rescheduling this node.
In detail, in Line 1, \textsc{Sector-Decider} is called to determine the active sector for each sensor node. Lines 2-10 is to initialize the variables. Lines 11-17 are used to determine the schedule of camera sensors. In particular, Line 12 utilizes maximal independent set of $Z'$ so that the camera sensors whose activation period is determined do not overlap. In this way, we can further maximize the target-temporal coverage. Finally, Lines 18-24 are checking of the determined schedule is satisfiable.

\textbf{A Big Picture} Now, we give a simple example to illustrate how the overall algorithm works. The camera sensor network is composed of 5 camera sensors, each of which has 8 available orientations, and 5 targets, each of which has its own facing direction (as shown by the shorter arrow of each target in Fig. 5). Among these 5 targets, there are 2 pivotal targets $a_1$ and $a_2$, i.e. $T_k = \{a_1, a_2\}$ and $w_1 = 4, w_2 = 3, w_3 = 2, w_4 = 1, w_5 = 1$. In the scheduling of each sensor, the length of each round $r = 10$ and the sensors’ battery lifetime per round $l_1, l_2, l_3, l_4, l_5$ are 5, 6, 4, 4, 6, respectively.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{fig5.png}
\caption{An instance of the overall algorithm}
\end{figure}

In Fig. 5, each sensor has been assigned the working direction, i.e. Lines 1-7 of Algorithm 3 have been accomplished for this network, and we obtain each sensor’s neighbor set: $N_1 = \{S_2, S_4\}$, $N_2 = \{S_1, S_3\}$, $N_3 = \{S_2\}$, $N_4 = \{S_1\}$, $N_5 = \{\emptyset\}$. Then we begin to generate a sleep-wakeup schedule for each sensor (Lines 8-17 of Algorithm 3). We first go into the while loop (Lines 11-17 of Algorithm 3):

(a) In the first loop, $Z' = \{S_1, S_2, S_3, S_4, S_5\}$ and an MIS of $Z'$ is $\{S_1, S_3, S_5\}$, which is obtained based on each sensor’s neighbor set. For each sensor in $\{S_1, S_3, S_5\}$, we run Algorithm 2 to get its starting time point:

(i) For $S_1$, $S_1 \cap T_k = \{a_1\}$ and $t_1 - rcd_1 = 0 < r$, so $Sch_1 = \{S_2.f\} = \{0\}$. It is easy to find that $S_1.b = 0$ satisfying $\Delta TC_1 = 0$, thus we can return $S_1.b = 0$. 

...
(ii) The output of Algorithm 2 on $S_3$ is similar to that of $S_1$ and $S_3.b = 0$. 
(iii) For $S_5$, since $S_5 \cap T_k = \emptyset$ and $N_5 = \{\emptyset\}$, $SCH_5 = \{0, 10\}$. 
We can easily get that $S_5.b = 0$ satisfying $\Delta TC_5 = 0$, thus we can return $S_5.b = 0$. 
(b) In the second loop, $Z'$ is updated into $\{S_2, S_4\}$ and the MIS of the current $Z'$ is $\{S_2, S_4\}$. For each sensor in $\{S_2, S_4\}$, we run Algorithm 2 to get its starting time point:

(i) For $S_2$, $S_2 \cap T_k = \{a_1, a_2\}$ and $t_1 - red_1 = 5 < r$, $t_2 - red_2 = 0 < r$, so $SCH_2 = \{S_1, f, S_3, f\} = \{5, 4\}$. It is easy to find that $S_2.b = 4$ satisfying $\Delta TC_2 = 4 (= \min\{7, 4\})$, thus we can return $S_2.b = 4$.

(ii) For $S_4$, since $S_4 \cap T_k = \emptyset$, $SCH_4 = \{S_1, b, S_1, f, 0, 10\} = \{0, 5, 10\}$.
We can easily get that $S_4.b = 5$ satisfying $\Delta TC_4 = 0$, thus we can return $S_4.b = 5$.

After the second loop, $Z'$ is updated into $\emptyset$, thus the while loop finishes. Finally we obtain that the scheduling are $S_1.b = 0$, $S_2.b = 4$, $S_3.b = 0$, $S_4.b = 5$, $S_5.b = 0$ and $TC = 94(= 4 \times (5 + 6 - 1) + 3 \times (4 + 6) + 2 \times (5 + 4) + 1 \times 6 + 1 \times 0)$. 

5 Simulation Results and Analysis

In this section, we present our results and discuss about the performance of our algorithm for DCA-MCSN. Specifically, we randomly deploy $n$ camera sensor nodes and $m$ targets within a $100 \times 100$ 2-D virtual space, where $n$ is set to 150, 200, \ldots, 400. It is expected that the number $n$ of available camera sensor nodes, whose battery lifetime is significantly lower than the mission lifetime $L$, is greater than the number of $m$ of targets, and thus we set $m$ as 20, 30, \ldots, 70. We set maximum sensing radius $R$ of each camera sensor to be 50. We assume the weight of each target is a random number within the range of $(0, 10)$ We assume the mission lifetime per a unit $r$ is 10 unit time. Lastly, we randomly assign the face direction vector of each target, and the maximum viewing angle is $\frac{\pi}{4}$. In this problem, each camera sensor node has $q$ sensing sectors and we will use one of them, i.e. the beamwidth $\theta$ of each sector is $\frac{2\pi}{q}$.

In the simulation, we study the average characteristic behavior of the proposed algorithm, and evaluate its average performance with regard to the target-temporal coverage $TC$ and the number of the most important targets fully covered by sensors given. Especially, we will check how this algorithm works with two important parameters: the uniform battery lifetime, $bl$, of each sensor and the beamwidth $\theta$ of each camera sensor. In particular, we will check the following six cases: Case (a) $bl = 4, \theta = \frac{\pi}{7}$, Case (b) $bl = 4, \theta = \frac{\pi}{7}$, Case (c) $bl = 4, \theta = \frac{\pi}{7}$, Case (d) $bl = 6, \theta = \frac{\pi}{7}$, Case (e) $bl = 6, \theta = \frac{\pi}{7}$, Case (f) $bl = 6, \theta = \frac{\pi}{7}$. We also use the sum of the weights of all targets, $\sum_{1 \leq k \leq m} w_k$, multiplied by the mission time duration per a unit $r$ as the theoretical upper bound, which is noted by $UB$. Clearly $UB$ can be larger than actual achievable optimum.
Fig. 6. Target-temporal coverage achieved by DCA-MCSN versus the number of sensors.

Fig. 7. Target-temporal coverage achieved by DCA-MCSN versus the number of targets.
5.1 Simulation Results for DCA-MCSN

In Fig. 6, we compare the performance of DCA-MCSN against $UB$ under the 6 parameter settings (Case (a) - (f)) while the number of targets to be covered is fixed to 50. By comparing the results with $bl = 4$ (Cases (a), (b), and (c)) against $bl = 6$ (Cases (d), (e), and (f)), we can learn that if the beamwidth $\theta$ of each sensor’s sector is fixed, the performance of the algorithm becomes close to $UB$ with larger longer battery lifetime $bl$. This is because with larger $bl$ value, we have more number of sensor nodes to fully cover more number of targets during the mission period. Next, we study how the total $TC$ is affected by the beamwidth $\theta$ of each sensor’s sector while we fix $bl$ to 4 and 6, respectively. By comparing Cases (a), (b), and (c) in Fig. 6 as well as Cases (d), (e), and (f) in Fig. 6, we can learn that $TC$ increases as $\theta$ grows. This is because the number of targets covered by each sensor increases as $\theta$ is getting larger. Overall, from Fig. 6, we can observe that the initial battery lifetime $bl$ of each sensor has more significant influence than the beamwidth $\theta$ on the performance of the algorithm.

Next, we study Fig. 7 in which we fix the number of sensor nodes deployed to 250 and varies the number of targets to be covered from 20 to 70. This result clearly shows that the performance of our algorithm is close to the theoretical upper bound independent of the other parameters including $bl$, $\theta$, and the number of targets $m$ once we obtain enough coverage over the area.

5.2 Simulation Results for DCA-MCSN vs. ACA

![Image of Fig. 8](image)

Fig. 8. The number of the first $k$ important targets covered by sensors by DCA-MCSN versus ACA.

In this subsection, we compare the performance of DCA-MCSN against ACA (TEC-NC-Adjuster) in [1] (which showed outstanding performance for solving...
the target-temporal effective-sensing coverage with adjustable cameras problem, TEC-AC, in the simulations) under the 3 parameter settings (Case (d) - (f)) while the number of targets to be covered is fixed to 50. We study how many important targets are fully covered by the two strategies (see Fig. 8). From this result, we can observe that for both of the strategies, the number of the most important target which are fully covered during the mission period is affected by $\theta$, i.e. $k$ is steadily increasing along with the growth of $\theta$. Furthermore, DCA-MCSN works better than ACA under all parameter settings, which implies that DCA-MCSN is more efficient than ACA for the mission-driven application considering the coverage quality of the targets with higher priority.

6 Conclusion

In this paper, we investigate a new coverage problem in camera sensor networks. In this problem, we study how to schedule a given set of camera sensor nodes to cover a set of given targets with weight related to their importance and fixed face directions such that the number of the first $k$ important targets effectively covered during a given mission period can be maximized as well as the overall target-temporal coverage is maximized (as the secondary optimization goal). The main contribution of this paper is the possibility of using insufficient number of camera sensors to best support a coverage mission in a way that never considered before. Since we does not always have enough number of sensor nodes to complete a coverage mission, our paper is of great applicability in many real application scenarios. As a future work, we plan to study the applicability of this model in the hybrid sensor network of both a number of cheap static sensor nodes and a few expensive mobile sensor nodes, which we believe also have a wide range of applications in many real life scenarios.

Acknowledgement

This research was jointly supported by National Natural Science Foundation of China under grant 91124001, the Fundamental Research Funds for the Central Universities, and the Research Funds of Renmin University of China 10XNJ032.

References


