Coverage Quality in Visual Sensor Networks

Aissan Dalvandi*

Department of Electrical & Computer, Islamic Azad University, Qazvin Branch, Qazvin, Iran

Received 4 November 2010; revised 2 April 2011; accepted 25 April 2011

Abstract

Coverage quality of targets is one of the most significant criteria for some applications such as surveillance and environmental monitoring. Cost is also an important factor for the coverage problem in visual sensor networks. Therefore, the present study aims to investigate a novel coverage problem by considering both cost and coverage quality. To accomplish this purpose, firstly a criterion for the coverage quality of visual sensors is defined with regard to the attributes of lens of their camera. Then, considering cost and quality objective functions, Max-Quality Min-Cost Selection problem (MQMCS) is addressed and formulated as a bi-objective programming. Finally, two centralized and distributed algorithms that with a high probability can find a cover set with the maximum coverage quality and the minimum number of sensors are proposed.

Keywords: Visual Sensor Network; Coverage Quality; Cost; Bi-objective Programming; Objective Function.

1. Introduction

In recent years, visual sensor networks have emerged as promising platforms for many applications like environmental monitoring [1], [2] and battlefield surveillance [3]. Coverage is one of the fundamental functionalities of sensor networks. Practically, visual sensors in most of the environments such as [4], [5], [6], [7] and [8] have been distributed densely. Therefore, we should select a set of them in order to cover interested targets.

In a WSN, a sensor covers a target if the target is in the sensing range of the sensor. There are three coverage models depending on how targets are defined:

1) Targets form a contiguous region and the objective is to select a subset of sensors to cover the region [9]. Typical solutions involve geometry properties based on the positions of sensor nodes.

2) Targets form a contiguous region and the objective is to select a subset of sensors to cover the rest of sensors [10]. This model assumes the network is sufficiently dense so that point coverage can simulate area coverage. Typical solutions involve constructing dominating sets or connected dominating sets [11] based on traditional graph theory.

3) Targets are discrete points and the objective is to select a subset of sensors to cover all of the targets. Typical solutions [12] use the traditional set coverage or bipartite graph models.

In this paper, we focus on the third coverage problem.

In the coverage problem, cost that can be defined as a function of the number of selected sensors is a major issue. Therefore, we should select the minimum number of visual sensors that can cover targets. Moreover, seeing targets with high quality can be one of the goals of monitoring applications.

A visual sensor has a non-uniform sensing region. That is to say, because of the attributes of lens of its camera, it can cover a target with different qualities in different orientations. Thus, we define a criterion for the coverage quality of visual sensors. Then, we define Max-Quality Min-Cost Selection problem (MQMCS) that finds a cover set with the maximum coverage quality and the minimum cost. We also formulate this problem as a bi-objective linear programming and solve it with the weighted-sum method. Since the number of selected sensors does not affect the coverage quality, we could solve problem for the exact weight and find an efficient cover set. By considering a higher weight for the cost function and a smaller weight for the quality function, we select the minimum number of sensors that cover all targets with the maximum quality. Afterwards, since finding a directional cover set is NP-complete [13], we propose two centralized and distributed

* Corresponding author. E-mail: aissan.dalvandi@qiau.ac.ir
algorithms in order to have a cover set with the maximum coverage quality and the minimum cost.

For solving MQMCS problem, the following assumptions and scenario are adopted in this paper. Some targets with known locations are deployed in a two-dimensional Euclidean plane. Visual sensors are distributed densely in the defined area. We use two algorithms (MQMCS-C and MQMCS-d) for finding a set of distributed sensors as a cover set. This cover set will be able to cover the interested targets with the maximum coverage quality and the minimum cost. Then, we evaluate these algorithms by three criteria (success rate, coverage percentage and coverage quality) and show that these algorithms with a high probability are able to find efficient cover set. It is important to note that different sensing regions of each directional sensor do not overlap. However, in this paper we do not put restrictions on the overlaps between shapes of directions of different sensors. The selected direction of one sensor is named working direction. Furthermore, if the target is placed in the working direction of the sensor, it will be covered by the sensor.

The rest of the paper is organized as follows: Section 2 briefly surveys the related literature. In Section 3, a criterion for the coverage quality of visual sensors is presented. In Section 4, Mix-Quality Min-Cost Selection problem is defined and in Section 5 it is formulated as a bi-objective linear programming. In Section 6, two algorithms (MQMCS-c and MQMCS-d) are suggested that select sensors with the maximum quality and the minimum cost as a cover set, and then they are evaluated. Finally, the conclusion is provided in Section 7.

Table 1
Summary of some studies on the coverage problem (TC: Target Coverage, AC: Area Coverage, C: Centralized, D: Distributed, NL: Network Lifetime)

<table>
<thead>
<tr>
<th>Paper</th>
<th>Field</th>
<th>Method</th>
<th>Dimension</th>
<th>Algorithm</th>
<th>Primary objective</th>
<th>Secondary objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>[14]</td>
<td>AC</td>
<td>D</td>
<td>2D</td>
<td>DGreedy</td>
<td>Max coverage</td>
<td>-</td>
</tr>
<tr>
<td>[15]</td>
<td>AC</td>
<td>D</td>
<td>2D</td>
<td>-</td>
<td>full coverage</td>
<td>Prolonging NL</td>
</tr>
<tr>
<td>[16]</td>
<td>AC</td>
<td>D</td>
<td>2D</td>
<td>EFCEA</td>
<td>Enhancing AC</td>
<td>Max NL</td>
</tr>
<tr>
<td>[17]</td>
<td>AC</td>
<td>D</td>
<td>2D</td>
<td>E-SURE</td>
<td>Prolonging NL</td>
<td>-</td>
</tr>
<tr>
<td>[18]</td>
<td>AC</td>
<td>D</td>
<td>2D</td>
<td>Self-orienting</td>
<td>Max coverage</td>
<td>-</td>
</tr>
<tr>
<td>[19]</td>
<td>AC</td>
<td>C</td>
<td>2D</td>
<td>Adaptive deployment</td>
<td>Min total cost</td>
<td>satisfying coverage requirement</td>
</tr>
<tr>
<td>[20]</td>
<td>AC</td>
<td>C</td>
<td>2D</td>
<td>Coverage enhancing</td>
<td>Max coverage</td>
<td>-</td>
</tr>
<tr>
<td>[21]</td>
<td>AC/TC</td>
<td>C</td>
<td>2D</td>
<td>Greedy</td>
<td>Guaranteeing k-coverage</td>
<td>Min sensors</td>
</tr>
<tr>
<td>[22]</td>
<td>TC</td>
<td>C</td>
<td>2D</td>
<td>Model direction partition</td>
<td>Prolonging NL</td>
<td>-</td>
</tr>
<tr>
<td>[23]</td>
<td>TC</td>
<td>C</td>
<td>2D</td>
<td>ILPSNCS</td>
<td>Max coverage / Min sensors</td>
<td>Prolonging NL</td>
</tr>
<tr>
<td>[24]</td>
<td>TC</td>
<td>C</td>
<td>2D</td>
<td>CGA, SNCS</td>
<td>Max coverage / Min sensors</td>
<td>Prolonging NL</td>
</tr>
<tr>
<td>[25]</td>
<td>TC</td>
<td>D</td>
<td>2D</td>
<td>DGA, SNCS</td>
<td>Max coverage / Min sensors</td>
<td>Prolonging NL</td>
</tr>
<tr>
<td>[26]</td>
<td>TC</td>
<td>C</td>
<td>2D</td>
<td>DCS-GA, WT_Greedy</td>
<td>Full coverage</td>
<td>Prolonging NL</td>
</tr>
<tr>
<td>[27]</td>
<td>TC</td>
<td>D</td>
<td>2D</td>
<td>DCS-GA, WT_Dist</td>
<td>Full coverage</td>
<td>Prolonging NL</td>
</tr>
<tr>
<td>[28]</td>
<td>TC</td>
<td>C</td>
<td>2D</td>
<td>WCGA</td>
<td>Max coverage</td>
<td>-</td>
</tr>
<tr>
<td>[29]</td>
<td>TC</td>
<td>D</td>
<td>3D</td>
<td>EDO</td>
<td>Covering critical targets</td>
<td>Max coverage / NL</td>
</tr>
<tr>
<td>[30]</td>
<td>TC</td>
<td>D</td>
<td>2D</td>
<td>NSS</td>
<td>Max NL</td>
<td>-</td>
</tr>
<tr>
<td>[31]</td>
<td>TC</td>
<td>D</td>
<td>2D</td>
<td>ILP</td>
<td>Min total cost</td>
<td>Max coverage / NL</td>
</tr>
<tr>
<td>[32]</td>
<td>TC</td>
<td>C</td>
<td>2D</td>
<td>ILP</td>
<td>Prolonging NL</td>
<td>-</td>
</tr>
<tr>
<td>[33]</td>
<td>TC</td>
<td>C</td>
<td>2D</td>
<td>CBDA</td>
<td>Prolonging NL</td>
<td>-</td>
</tr>
<tr>
<td>[34]</td>
<td>TC</td>
<td>C</td>
<td>3D</td>
<td>VFA-ACE</td>
<td>Improving coverage</td>
<td>-</td>
</tr>
<tr>
<td>[35]</td>
<td>TC</td>
<td>D</td>
<td>3D</td>
<td>Simulated annealing</td>
<td>Improving coverage</td>
<td>-</td>
</tr>
<tr>
<td>[36]</td>
<td>TC</td>
<td>C</td>
<td>2D</td>
<td>Direction partition</td>
<td>Full coverage / Min sensors</td>
<td>-</td>
</tr>
<tr>
<td>[37]</td>
<td>TC</td>
<td>C</td>
<td>2D</td>
<td>ILP</td>
<td>Min sensors</td>
<td>-</td>
</tr>
<tr>
<td>[38]</td>
<td>TC</td>
<td>C</td>
<td>2D</td>
<td>ILP</td>
<td>Min total cost</td>
<td>-</td>
</tr>
<tr>
<td>[39]</td>
<td>TC</td>
<td>C</td>
<td>2D</td>
<td>Greedy algorithm</td>
<td>connected network / Min sensors</td>
<td>-</td>
</tr>
<tr>
<td>[40]</td>
<td>TC</td>
<td>C</td>
<td>2D</td>
<td>Strip-based algorithm</td>
<td>connected network / Min sensor</td>
<td>-</td>
</tr>
</tbody>
</table>
2. Related work

In contrast to omni-directional sensors that have an omni-angle of sensing range, directional sensors have diverse sensing regions with each being defined by a sector of the sensing disk centred at the sensor in a certain direction with a sensing radius. The sensing sector of a directional sensor is characterized by the following parameters:

1) \((x_i, y_i)\): the Cartesian coordinates that denote the physical location of the sensor in a two-dimensional plane,

2) \(\varphi_i^f\): the maximum angle of sensing that can be achieved by the sensor. It is also called the Field of View (FOV),

3) \(r_i\): the maximum sensing range of the sensor beyond which a control point cannot be monitored, and

4) \(\hat{d}_{ij}\): the unit vector that cuts the sensing sector into half.

These parameters define the direction of sensors. It is important to note that in this paper we assume that the sensor nodes have been equipped with a device that enables them to switch or rotate in different directions in order to meet the sensing coverage requirements. Therefore, we don’t distinguish between the terms sensor and node. The coverage problem in directional sensors has been attracting more attention recently. Coverage in general answers the questions about the surveillance that can be provided by a particular sensor network. Thus, we need a coverage that meets goals of the problem such as increasing the lifetime, covering all targets or enhancing coverage quality. In this regard, [32] discussed area coverage problems, and provided a directional sensor model in which each sensor has a fixed direction and analyzes the possibility of full area coverage. But [34] assumed that each sensor is allowed to work in several directions, and therefore proposed a directional sensor model, similar to that suggested in [32], in order to find a minimal set of directions that can cover the maximal number of targets.

In [14] there were several non-disjoint cover sets and a special work time for each of them to maximize the network lifetime. Therefore, three heuristic algorithms based on Linear Programming were proposed and evaluated. Most of the current studies, described in Table 1, develop algorithms for the 2D environment. There are a few algorithms proposed for the 3D environment [29]. Adjusting sensor parameters may fill coverage holes or help to cover more target points [32]. Nevertheless, increasing the sensing radius and/or field of view has a cost in terms of energy depletion and budget. In this paper, we focus on maximizing the coverage quality and minimizing the cost in order to cover all targets not mentioned in the studies conducted into the coverage problem summarized in Table 1. These studies provided centralized or distributed algorithms for the target coverage or area coverage problem, and pursued different objectives such as maximizing coverage, minimizing total cost or prolonging network lifetime.

3. The coverage quality of visual sensors

In this section we define a criterion for the coverage quality of visual sensors. Because of the properties of the lens of their camera, one sensing range of visual sensors becomes non-uniform. That is to say, a visual sensor senses one target in diverse dimensions with different coverage qualities. Based on the attributes of lens, we know that targets that are closer to the center of the field of view of the visual sensor will be seen with a higher quality. Therefore, the coverage quality of the visual sensor for one target can be defined by the angle between the unit vector of the visual sensor and the vector of orientation from the sensor to the target (Figure 1). This angle changes continuously from zero to \(\text{FOV}/2\). When the angle equals zero, the target is in the center of the field of view, and thus it is seen with the highest quality.

![Sensor Coverage](image)

**Fig. 1.** Sensor \(i\) in direction \(j\) covers target \(k\). \(\theta\) is the angle between unit vector \(j\) and the vector, connecting sensor \(i\) with target \(k\).

Moreover, the nearer to \(\text{FOV}/2\) the angle is, the farther from the center of the field of view the target is placed. As a result, the visual sensor covers the target with a lower quality. According to the points illustrated we can define the coverage quality of sensors as follows:

\[
q_{ikj} = \begin{cases} 
1 + \frac{\theta}{\text{FOV}/2} & \text{if } s_i \text{ covers target } k \text{ in direction } j \\
0 & \text{otherwise} 
\end{cases}
\]  

(1)

It is obvious that for targets placed in the frontier points \(\theta = \text{FOV}/2\), \(q_{ikj} = 0^+\). In addition, when targets are placed on vector of orientation \(j\), \(\theta = 0\) and consequently \(q_{ikj} = 1^+\). It is important to note that the coverage quality of sensors for the targets placed out of the sensing range is equal to zero. By using this criterion, we describe two different methods for the coverage quality of target \(k\) in the following sections.
4. Max-Quality Min-Cost Selection problem

In this section, after presenting notations, we describe Max-Quality Min-Cost Selection problem (MQMCS).

4.1. Notations and Assumptions

We adopt the following notations and definitions throughout the paper:

- M: the number of targets.
- N: the number of sensors.
- W: the number of directions per sensor.
- A: the set of target A = \{a_1, a_2, ..., a_M\}.
- S: the set of sensors. S = \{s_1, s_2, ..., s_N\}.
- \(d_{ij}\): the jth direction of the i\(^{th}\) sensor, 1 < \(i \leq N\), 1 < \(j \leq W\). We define \(D = \{d_{ij} | 1 \leq i \leq N, 1 \leq j \leq W\}\).
- \(C\): a subset of D consisting of the directions of the selected sensors. \(C = \{d_{ij} | a_k is covered by d_{ij}, \forall a_k \in A\}\).

4.2. Problem definitions

**Definition 1.** Cover Set: Given a collection D of subsets of a finite set A and a partition S of D, a cover set for A is a subset \(C \subseteq D\) such that every element in A belongs to at least one member of C and every two elements in C cannot belong to the same member of S.

**Definition 2.** Directional Cover Set Problem (DCS): Given a collection D of subsets of a finite set A and a partition S of D, find a cover set for A.

**Definition 3.** Max-Quality Min-Cost Selection problem (MQMCS): Problem of finding a directional cover set that has maximum coverage quality and minimum cost.

According to Definition 1, because a sensor can be selected at one of its directions, |C| is the number of sensors that have been selected by the DCS.

The DCS is to be NP-complete by reduction from the 3-CNF-SAT problem [13]. As the MQMCS is a kind of DCS problem too, it is NP-complete.

5. The Optimization Formulation of MQMCS Problem

In this section, a bi-objective mixed integer programming formulation is developed to find an optimal subset of visual sensors and their directions in order to minimize the total cost and maximize the coverage quality while covering all interested targets. We first introduce the notations used in the formulation. Then, the BMIP model is described.

5.1. Notations

The notation is composed of sets, decision variables, and parameters.

5.1.1. Sets
- \(S\), Sensors
- \(T\), Targets
- \(D\), Directions (or orientations) for a sensor

5.1.2. Variable
- \(d_{ij}\), A 0-1 variable such that \(d_{ij} = 1\) if and only if a sensor \(i \in S\) has orientation \(j \in D\)
- \(c_k\), A 0-1 variable such that \(c_k = 1\) if and only if target \(k \in T\) is covered by at least one sensor

5.1.3. Parameter
- \(C\), cost of a sensor
- \(G\), coverage matrix \([g_{i,k,j}]\) where \(g_{i,k,j} = 1\) if sensor \(i \in S\) covers target \(k \in T\) in direction \(j \in D\)
- \(Q\), quality matrix \([q_{i,k,j}]\) where \(q_{i,k,j}\) is the coverage quality of sensor \(i \in S\) for target \(k \in T\) in direction \(j \in D\) (calculated based on the criterion in section 3)

5.2. The Formulation of MQMCS problem

In this section, we present the objective functions and essential constraints in terms of the above notation in order to formulate the multi-objective MQMCS problem.

We assume that only one type of sensor with different directions is available. For each sensor, the FOV parameters \(r\) and \(\alpha\) are given. We assume that the positions of all \(N\) sensors and \(M\) targets are given and fixed. Similarly each pose of sensor is able to select among \(W\) directions for each sensor.

5.2.1. The Cost Objective function

One of the goals of MQMCS problem is to minimize the total cost of the selected sensors. Since C is the cost of each sensor and \(d_{ij}\) is the binary variable that shows sensor \(i\) is selected in direction \(j\), we can formulate the cost objective function according to the following equation:

\[
\min \sum_{i} \sum_{j} C * d_{ij}
\]  

(2)

5.2.2. The Quality Objective function

Another goal of MQMCS problem is to maximize the coverage quality. Therefore, we should calculate the total coverage quality that can be presented by summing the coverage quality of targets. As a result, first we should calculate the coverage quality of each target.

In target coverage problems, the number of visual sensors covering one target does not affect the coverage
quality of that target. In other words, seeing one target from different directions by diverse sensors does not provide additional information. In fact, the coverage quality of the target is equal to the coverage quality of the visual sensor that covers it with the best view. Consequently, if some visual sensors cover one target, the coverage quality of this target will be equal to the maximum coverage quality among all sensors. As a result, the coverage quality of target \(k\) will be calculated by equation 3. In this equation, \(Q_k\) is the coverage quality of target \(k\) and \(q_{i,k,j}\) is calculated according to equation 1.

\[
Q_k = \max_{i,j} q_{i,k,j} \tag{3}
\]

According to the points demonstrated, we can formulate the quality objective function as follows (equation 4):

\[
\max \sum_k \max_{i,j} q_{i,k,j} \cdot d_{ij} \tag{4}
\]

The term \(\max_{i,j} q_{i,k,j} \cdot d_{ij}\) in the quality objective function is non-linear because it involves the maximum function. To avoid the complexity of such mixed integer non-linear programming (MINLP) models, the above model is linearized by defining a new variable and reformulating the objective function as follows:

\[
Z_k = \max_{i,j} q_{i,k,j} \cdot d_{ij} \tag{5}
\]

Therefore, we replace \(\max_{i,j} q_{i,k,j} \cdot d_{ij}\) with the new variable defined as \(Z_k\), and introduce the following constraints:

\[
Z_k \geq q_{i,k,j} \cdot d_{ij} \tag{6}
\]

\[
\prod_{i,j} (Z_k - q_{i,k,j} \cdot d_{ij}) = 0 \tag{7}
\]

As equation 7 is a non-linear constraint. It should be linearized. We can rewrite it as follows:

\[
(Z_k - q_{1,k,1} \cdot d_{11}) = 0 \tag{8}
\]

\[
\quad \text{Or}
\]

\[
(Z_k - q_{1,k,2} \cdot d_{12}) = 0 \tag{9}
\]

\[
\quad \text{Or}
\]

\[
\quad \text{Or}
\]

\[
\quad \text{Or}
\]

\[
(Z_k - q_{1,k,w} \cdot d_{1w}) = 0 \tag{10}
\]

\[
\quad \text{Or}
\]

\[
(Z_k - q_{2,k,1} \cdot d_{21}) = 0 \tag{11}
\]

\[
\quad \text{Or}
\]

\[
\quad \text{Or}
\]

\[
\quad \text{Or}
\]

\[
(Z_k - q_{2,k,w} \cdot d_{2w}) = 0 \tag{12}
\]

Moreover, each \((Z_k - q_{i,k,j} \cdot d_{ij}) = 0\) in the illustrated constraints can be replaced with two equations 15 and 16:

\[
(Z_k - q_{i,k,j} \cdot d_{ij}) \geq 0 \tag{15}
\]

\[
(Z_k - q_{i,k,j} \cdot d_{ij}) \leq 0 \tag{16}
\]

At the end, by defining some new binary variables, we have:

\[
(Z_k - q_{1,k,1} \cdot d_{11}) \leq P \cdot y_1 \tag{17}
\]

\[
(Z_k - q_{1,k,1} \cdot d_{11}) \geq -P \cdot y_1 \tag{18}
\]

\[
\quad \text{Or}
\]

\[
\quad \text{Or}
\]

\[
(Z_k - q_{N,k,w} \cdot d_{NW}) \leq P \cdot y_{W \cdot N} \tag{19}
\]

\[
(Z_k - q_{N,k,w} \cdot d_{NW}) \geq -P \cdot y_{W \cdot N} \tag{20}
\]

\[
y_1 + y_2 + \cdots + y_{W \cdot N} \leq (W \cdot N) - 1 \tag{21}
\]

It is important to note that \(P\) is a very large number such as \(10^{14}\). As a result, all the points mentioned help us formulate the multi-objective MQMCS problem for the max method according to Figure 5.

5.2.3. Constraints

In this subsection, we present the constraints that define MQMCS problem. We need to express the variables defining coverage in terms of the other defined variables just mentioned as follows. Since \(c_k = 1\), if and only if at least one sensor covers target \(k\), we introduce the following two inequalities:

\[
c_k \cdot (\sum_{i,j} d_{ij} \cdot g_{ik,j} - 1) \geq 0 \tag{22}
\]

\[
(c_k - 1) \cdot (\sum_{i,j} d_{ij} \cdot g_{ik,j}) \geq 0 \tag{23}
\]
The first two constraints (22) and (24) involve products of binary variables, thus they are nonlinear. To linearize the inequalities, we introduce a new binary variable for each nonlinear term as well as two additional constraints [35]. Therefore, we replace each \( c_k \cdot d_{ij} \) term with a binary variable \( v_{ik,j} \), and describe equations 22 and 23 as same as equations 24 and 25:

\[
\sum_{i} \sum_{j} v_{ik,j} \cdot g_{ik,j} - \sum_{i} d_{ij} \cdot g_{ik,j} \geq 0 \\
\sum_{i} \sum_{j} v_{ik,j} \cdot g_{ik,j} - c_k \geq 0
\] (28)

To introduce variable \( v_{ik,j} \), we should use the following constraints:

\[
c_k + d_{ij} \geq 2 \cdot v_{ik,j} \\
 c_k + d_{ij} - 1 \leq v_{ik,j}
\] (29)

Subject to:

\[
c_k + d_{ij} \geq 2 \cdot v_{ik,j} ; 1 \leq j \leq W, 1 \leq k \leq M, 1 \leq i \leq N
\]

\[
c_k + d_{ij} - 1 \leq v_{ik,j} ; 1 \leq j \leq W, 1 \leq k \leq M, 1 \leq i \leq N
\]

\[
\sum_{j} v_{ik,j} \cdot g_{ik,j} - \sum_{i} d_{ij} \cdot g_{ik,j} \geq 0 ; 1 \leq k \leq M
\]

\[
\sum_{j} v_{ik,j} \cdot g_{ik,j} - c_k \geq 0 ; 1 \leq k \leq M
\]

\[
\sum_{i} c_k = M
\]

\[
\sum_{i} d_{ij} \leq 1 ; 1 \leq i \leq N
\]

\[
Z_k \geq (\sum_{i} c_k \cdot d_{ij}) ; 1 \leq i \leq W, 1 \leq k \leq M, 1 \leq l \leq N
\]

\[
Z_k - (\sum_{i} c_k \cdot d_{ij}) \cdot P \cdot y_{h} , 1 \leq k \leq M, 1 \leq l \leq N, 1 \leq j \leq W, 1 \leq h \leq W \cdot N
\]

\[
Z_k - (\sum_{i} c_k \cdot d_{ij}) \geq -P \cdot y_{h} , 1 \leq k \leq M, 1 \leq l \leq N, 1 \leq j \leq W, 1 \leq h \leq W \cdot N
\]

\[
y_z + y_{z} + \cdots + y_{z,N} \leq (W \cdot N) - 1
\]

\[
d_{ij}, v_{ik,j}, c_k, y_{h} \in \{0,1\} , 1 \leq k \leq M, 1 \leq l \leq N, 1 \leq j \leq W, 1 \leq h \leq W \cdot N
\]

\[
P = 10^{14}
\]

Fig. 2. The Formulation of MQMCS problem

To ensure that exactly one pose is assigned to each sensor, we also use the following constraint (equation 28) for each sensor \( i \).

Further, to guarantee that all targets are covered, the following constraint is needed as well:

\[
\sum_{k} c_k = M
\] (29)

By using the provided objective functions and constraints, our sensor deployment problem can now be formulated as a BMIP model. The result is shown in Figure 2.

We use the weighted-sum method to solve the BMIP model. First, we convert the minimum to the maximum for the cost objective function as follows [36]:

\[
\min \Sigma_i \Sigma_j C \cdot d_{ij} = \max \Sigma_i \Sigma_j - C \cdot d_{ij}
\] (30)

Thus, we have this objective function:

\[
\max w \frac{\Sigma_i \Sigma_j - C \cdot d_{ij}}{f_c} + (1 - w) \frac{\Sigma_k z_k}{f_q}
\] (31)

Now, two objective functions are mutually maximized, \( f_c \) and \( f_q \) are the normalization factors for the cost and quality objective functions, respectively, and \( w \) is the weighting factor which shows the relative importance of two objective functions. We also add \( w \in [0 \ 1] \) to the previous constraints [37].

We should solve this problem by considering the weighted objective function for different weights and then drawing the Pareto front diagram.

Figure 3 shows the Pareto front solutions obtained by AIMMS 11.0. In this scenario 100 sensors and 10 targets are deployed uniformly in a region of \( R \times R \), where \( R = 200 \). The X axis indicates the total cost calculated by \( \Sigma_i \Sigma_j C \cdot d_{ij} \), and the Y axis shows the quality per target that is the average of the coverage quality of targets and calculated by \( \Sigma_k z_k \frac{M}{M} \).

Fig. 3. Relationship between the cost and quality objective functions for 100 sensors and 10 targets
As shown in Figure 3, there is one point with maximum quality and minimum cost. This point presents a low weight for the quality objective function and a high weight for the cost objective function.

Below, two diagrams (Figure 4 and Figure 5) are presented that describe the relationship between \( w \) and each objective function in order to show this point. All in all, according to the three figures, by considering an exact amount for \( w \), we can gain an efficient cover set that has the minimum cost and the maximum quality. Therefore, if we select by solving weighted objective functions for this weight, we will have a cover set with the minimum cost with each of its members covering at least one target with the maximum quality.

We solve the presented weighted objective function by considering \( w = 0.9 \) in order to find a cover set. In this scenario 200 sensors and 10 targets are deployed uniformly in a region of \( R \times R \), where \( R = 200 \). Figure 6 shows the cover set that is gained by AIMMS 11.0. As you see, 9 sensors are selected in order to cover 10 targets.

![Figure 4](image4.png)

**Fig. 4.** Relationship between the cost objective function and \( w \) for 100 sensors and 10 targets

![Figure 5](image5.png)

**Fig. 5.** Relationship between the quality objective function and \( w \) for 100 sensors and 10 targets

6. The Solution to MQMCS

In this section, first we present a centralized target-based algorithm named MQMCS-c that has a high possibility to find a cover set with minimum cost (minimum number) and maximum coverage quality. Then we propose a distributed target-based algorithm named MQMCS-d that has more scalability in comparison with MQMCS-c. Finally, we evaluate these algorithms.

6.1. The MQMCS-c algorithm

In this subsection, we propose a target-based algorithm named MQMCS-c. This algorithm has two stages: main stage and cropping stage.

6.1.1. Main stage

In the main stage, we have a visual sensor network with a set \( A \) of \( M \) targets, a set \( S \) of \( N \) sensors and a set \( D \) of directions. Because the algorithm firstly selects one target and then find the sensor that can cover this target, we name it a target-based algorithm.

The algorithm firstly prioritizes targets by the following strategy: The number of sensors that can cover each target is defined as \( p_i \). Now, for two targets \( i \) and \( j \), if \( p_i < p_j \), target \( i \) will has a higher priority in comparison with target \( j \).

Besides, we classify the sensors by using the two definitions below. If a sensor has more than one direction that can cover at least one member of set \( A \), we say that these directions conflict with each other and they are named conflicting directions. Otherwise, if the sensor only has one direction that covers at least one of the targets, it is named a non-conflicting direction. For example, if the directions \( d_{i,j} \) and \( d_{i,j'} \) of sensor \( s_i \) with each covering at least one of the members of set \( A \) exist, they will be in conflict with each other. We classify non-conflicting and conflicting directions as two separate sets and name them non-conflict and conflict.
sets, respectively. Furthermore, we calculate the coverage quality of the sensors for all the targets.

MQMCS-c uses the following strategy in order to present a cover set that has the maximum quality and the minimum cost (minimum sensors):

We use \( C \) to denote a selected set of directions for covering \( A \). First, it picks up the target with the highest priority. Then if the non-conflict set is not empty, we search it to find the direction of the sensor that can cover the selected target with the highest quality. It is important to note that the coverage quality of the sensor for each target is calculated according to the criterion in section 3. This policy leads us to select the sensor that is able to monitor the target with the best quality. We remove the targets that are covered by the selected non-conflict direction from \( A \). Then, we select another target with the highest priority in \( A \), and do all the steps just mentioned.

It should be pointed out that if the non-conflict set was empty or we cannot find any non-conflict direction that covers the selected target, we will search among the conflict directions for finding the sensor that can cover the selected target. By repeating these steps, a set of directions \( C \) is gained. When \( A \) is empty, the algorithm succeeds to find a cover set that is named \( C \). When \( A \) is not empty and we cannot find any sensor in neither the non-conflict nor the conflict set for its members, the algorithm fails to find a cover set. Therefore, it returns \( C = \emptyset \).

The cover set provided by the MQMCS-c algorithm at the end of the main stage is able to cover all the targets of \( A \) but as it may have some redundancy in some situations, the cropping stage is presented to reduce this redundancy.

6.1.2. Cropping stage

Goals of an application form its policy to find a cover set. Thus, our policy has significant effects on the algorithm efficiency. It is also obvious that algorithms with less redundancy will be more efficient, but in some cases if we first consider the redundancy, we may not achieve our main goals. Therefore, in such cases finding results without paying attention to the redundancy and then deleting redundant results is the best way.

In this paper, our main goal is to find a cover set with a high coverage quality and minimum number of sensors (minimum cost). Subsequently, selecting sensors that cover targets with a good field of view (the illustrated criterion in section 3) leads us to finding a cover set with a reasonable coverage quality. We must also select minimal number of directions. Since in each iteration of the main stage we select one target and then try to cover it, a sensor selected in one iteration may cover targets that were covered in the previous iterations. In other words, we may select one direction for the covering targets of which some are covered by the direction selected in the previous iterations. Selecting this new direction changes previously selected directions from essential directions to redundant directions. Therefore, we present the cropping stage that removes surplus directions. After this stage, each member of \( C \) will be useful and covers at least one target of \( A \) that other selected directions cannot cover.

In each iteration of the cropping stage, we select a direction of \( C \) that covers most number of the targets and add it to \( C \). Then we delete all targets that are covered by this direction from \( A \) and repeats until \( A = \emptyset \). Finally \( C \) is a cover set with the least redundancy. The two stages of the MQMCS-c algorithm are illustrated below.

MQMCS-c algorithm:

Main stage:

1. Get \( S = \{ s_i | i = 1, 2, 3, ..., N \} \) as an input variable.
2. Get \( A = \{ s_i | i = 1, 2, 3, ..., M \} \)
3. Get \( D = \{ d_{ij} | 1 \leq j \leq W, 1 \leq i \leq N \} \)
4. Calculate non_conflict and conflict sets.
5. For each \( s_i \) in \( A \):
   \[ p_k = \{ s_i | a_k \in D_{ij}, 1 \leq j \leq W \} \]
6. End for;
7. For each \( a_k \) in \( A \):
8. For each \( d_{ij} \) in \( D \):
9. Calculate \( q_{ikj} \)
10. End for;
11. End for;
12. A are sorted decreasingly according to \( T_k = 1/p_k \)
13. \( C = \emptyset \)
14. While \( A \neq \emptyset \)
15. Pick up one member of \( A \) as a \( a_k \) that has the biggest \( T_k \)
16. If non_conflict \( \neq \emptyset \)
17. \[ U = \{ d_{ij} | d_{ij} \in \text{non_conflict} \land a_k \in D_{ij} \} \]
18. If \( U \neq \emptyset \)
19. Pick \( d_{ij} \) in \( U \) that \( q_{ikj} \) is max in \( U \)
20. \( A = A - D_{ij} \)
21. \( C = C \cup \{ d_{ij} \} \)
22. non_conflict = non_conflict - \( \{ s_i | d_{ij} \in s_i \} \)
23. End if;
24. End if;
25. If \( (\text{non_conflict} = \emptyset \lor U = \emptyset ) \)
26. \[ U = \{ d_{ij} | d_{ij} \in \text{conflict} \land a_k \in D_{ij} \} \]
27. If \( U \neq \emptyset \)
28. Pick \( d_{ij} \) in \( U \) that \( q_{ikj} \) is max in \( U \)
29. \( A = A - D_{ij} \)
30. \( C = C \cup \{ d_{ij} \} \)
31. conflict = conflict - \( \{ s_i | d_{ij} \in s_i \} \)
32. End if;
33. End if;
34. End while;
35. If \( A = \emptyset \)
36. \( C = \emptyset \)
37. End if;
38. Return \( C \)
6. The MQMCS-d algorithm

In this subsection, we propose a distributed algorithm called MQMCS-d to find a cover set when centralized algorithms are inapplicable. In this algorithm, a sensor only communicates with its neighbours in its communication rang. There are two stages in this algorithm: the communicating stage and the decision stage.

Because in this algorithm each sensor prioritizes targets that can cover, we name it a target-based algorithm.

6.2.1. Communicating stage

In the communicating stage, each sensor scans its targets that its directions can cover and assigns three numbers \( p_{1k} \), \( p_{2k} \) and \( p_{3k} \) as priorities to each target locally. A target can be covered by the directions of a sensor and its neighbours fewer times. Initially, each sensor is in the active state. First, each sensor scans the environment to detect the targets, denoted as \( D_{ij} \), that can be covered by each of its directions. Then it calculates its coverage quality for each \( a_k \in D_{ij} \), denoted as \( q_{ik} \). Sensor \( s_i \) maintains \( D_{ij} \) and \( q_{ik} \), for \( j = 1 \ldots W \). Then \( s_i \) broadcasts a message indicating \( q_{ik} \) as its coverage quality in each of its directions for the targets that can be covered by it to its neighbours, denoted as \( N_i \). After waiting for a period to receive the broadcasted messages of its neighbours, \( s_i \) assigns priorities \( p_{1k}, p_{2k} \) and \( p_{3k} \) to each target \( a_k \) in \( \bigcup_{j=1}^{W} D_{ij} \).

\[
P_{1k} = 1 + \left[ q_{ik} > 0, \forall a_k \in \bigcup_{j=1}^{W} D_{ij}, d_{ij} \in s_i, s_{ij} \in N_i \right]
\]

\[
P_{2k} = \max \left( q_{ik} \mid a_k \in \bigcup_{j=1}^{W} D_{ij}, d_{ij} \in s_i, s_{ij} \in N_i \cup s_i \right)
\]

\[
P_{3k} = \{ s_{ij} \mid s_{ij} \in N_i, s_{ij} \text{ cover } a_k \text{ with max quality} \} \quad (34)
\]

The denominator of \( p_{1k} \) indicates how many times a target \( a_k \) in \( \bigcup_{j=1}^{W} D_{ij} \) can be covered by the directions of \( s_i \) and its neighbours. \( p_{2k} \) shows the coverage quality of the sensor that has the maximum value among the sensors that can cover \( a_k \) and \( p_{3k} \) maintains the id of this sensor.

After each sensor assigns the priorities to all targets that its directions can cover, it moves to the decision stage.

6.2.2. Decision stage

In this stage, a sensor probes the states of its neighbours and decides about its work direction. First, each sensor \( s_i \) initializes a timer \( T_p \) as a value uniformly distributed in \([0, \delta_p]\) and goes to sleep. When the timer \( T_p \) decreases up to zero, \( s_i \) wakes up and marks itself as the PREWORK. Note that the sensor in the PREWORK state does not respond to its neighbours. Then \( s_i \) broadcasts a probing message and waits for a period for its neighbours’ replies. On receiving the message, any active neighbour \( s_{ij} \) which is not in the PREWORK state responds to \( s_i \) with a message which contains its \( q_{ik} \). At last, \( s_i \) makes a decision based on its neighbours’ replies. Among the uncovered targets, it picks up \( a_k \) with the highest priority \( p_{1k} \); then if \( s_i \) can cover \( a_k \) with \( q_{ik} = p_{2k} \), it will erase the PREWORK mark, and works in the direction that covers this target; otherwise, it checks whether \( p_{3k} \) belongs to one of its active neighbours or not: if it belongs, \( s_i \) will erase the PREWORK mark, and works in the direction that covers this target; otherwise, it will select \( a_k \) having \( p_{1k} \) with the highest value after the previously selected target. These steps will be repeated until one of the directions of the sensor is selected. At the end, if no direction of the sensor is selected, it can simply go to sleep.

QMCS-d algorithm:

Communicating stage:

1. Each sensor \( s_i \) detects the targets that can be covered by each of its directions \( D_{ij} \), for \( j = 1 \ldots W \).
2. \( s_i \) calculates \( q_{ik} \) for each \( a_k \in \bigcup_{j=1}^{W} D_{ij} \) for \( j = 1 \ldots W \).
3. \( s_i \) broadcasts a message including \( q_{ik} \) for each \( a_k \in \bigcup_{j=1}^{W} D_{ij} \) for \( j = 1 \ldots W \) to its neighbors \( N_i \).
4. \( s_i \) waits for a period for receiving the broadcasted messages of its neighbors.
5. For each \( a_k \in \bigcup_{j=1}^{W} D_{ij} \), \( s_i \) assigns priority \( p_{1k} \).
6. \( s_i \) assigns priority \( p_{1k} \).
7. \( s_i \) assigns priority \( p_{2k} \).
8. \( s_i \) assigns priority \( p_{3k} \).
9. End for;
10. \( s_i \) initializes a timer \( T_p \) and goes to sleep.
11. If \( T_p > 0 \)
12. Wait
13. Else
14. Go to the decision phase.
15. End if;

Decision stage:
16. \( s_i \) wakes up and marks itself as the PREWORK
17. \( s_i \) broadcasts a probing message and waits for a period for replies.
18. For each \( s_j \in N_i \)
19. If \( s_j \) is active but not in the PREWORK state
20. \( s_i \) responds to \( s_j \) and indicates \( q_{ikj} \)
21. End if;
22. End for;
23. For \( \forall a_k \in \bigcup_{j=1}^{W} D_{ij} \)
24. If \( a_k \) is uncovered
25. \( U = U \cup \{ a_k \} \)
26. End if;
27. End for;
28. \( s_i \) picks up \( a_k \) with the highest \( p_{ik} \) in \( U \)
29. \( U = U - \{ a_k \} \)
30. For \( j = 1 \ldots W \)
31. If \( p_{2k} = q_{ikj} \)
32. \( s_i \) erases its PREWORK mark
33. \( s_i \) Works in the \( d_{ij} \) direction
34. Else if \( 3 s_j = p_{2k}, s_j \in N_i \), \( s_i \) is active
35. \( s_i \) erases its PREWORK mark
36. \( s_i \) Works in the \( d_{ij} \) direction
37. End if;
38. End for;
39. If \( s_i \) is the PREWORK and \( U \neq \emptyset \)
40. Go 13
41. End if;
42. If \( s_i \) is the PREWORK and \( U = \emptyset \)
43. \( s_i \) goes to sleep;
44. End if;

6.3. Evaluation

We assess the performance of the MQMCS-\( c \) and MQMCS-\( d \) algorithms through simulations running on a computer with a 3 GHz CPU and 1 GB of memory. \( N \) sensors with sensing radius \( r \) and \( M \) targets are deployed uniformly in a region of \( R \times R \), where \( R = 400 \). Each sensor has \( W \) directions.

Each algorithm runs 1000 times through the random placement of sensors and targets. For the QCS-\( d \) algorithm, we assume that the communication radius is twice the size of the sensing radius. We evaluate the algorithms using the three criteria of success rate, coverage percentage and coverage quality.

6.3.1. Success Rate

Figure 7 shows the success rate of the MQMCS-\( c \) and MQMCS-\( d \) algorithms. The success rate is the ratio of the number of samples where a cover set is successfully found by each algorithm to the total number of samples. We consider two scenarios: \( M = 40, r = 100, W = 3 \) and \( M = 40, r = 100, W = 8 \). According to the figure, the MQMCS-\( c \) algorithm has a higher success rate than the MQMCS-\( d \) algorithm in both scenarios. In addition, it shows that increasing the number of sensors increases the success rate while increasing the number of directions per sensor decreases it. Figure 8 illustrates the relationship between the success rate of the two algorithms and \( r \), the radius of sensor for \( M = 40, N = 50, W = 3 \) and \( M = 40, N = 50, W = 8 \). As you see, increasing the \( r \) increases the success rate, but the success rate drops when \( W \) increases.

Fig. 7. Success rate vs. number of sensors \( N \) with \( r=100, M=40, W=3 \) and \( r=100, M=40, W=8 \).

Fig. 8. Success rate vs. sensing radius of sensor \( r \) with \( N=50, M=40, W=3 \) and \( N=50, M=40, W=8 \).

6.3.2. Coverage Percentage

Figure 9 shows the coverage percentage of the MQMCS-\( c \) and the MQMCS-\( d \) algorithms. The coverage percentage is the ratio of the number of covered targets to the total number of targets \( M \). We consider two previous scenarios. We can see from this figure that the coverage percentages of both algorithms increase quickly when \( N \) increases from 10 to 25 and somehow slowly after 25. The coverage percentage of the MQMCS-\( d \) algorithm is slightly smaller.
than that of the MQMCS-c algorithm. The figure also shows that the coverage percentages of the two algorithms drop when $W$ grows.

Figure 10 shows the relationship between the coverage percentage and $r$ for the two scenarios mentioned above. From this figure we can see that the MQMCS-c algorithm can have a relatively higher coverage percentage even when $W = 8$.

6.3.3. Coverage quality

Figure 11 indicates the coverage quality of the MQMCS-c and the MQMCS-d algorithms. The coverage quality is the average of coverage quality per each covered target. For our two scenarios, the figure reveals that when more sensors are developed, the coverage quality of both algorithms will be higher.

The coverage percentage drops when $W$ decreases as well. Figure 12 shows the relationship between this criterion and $r$. From this figure we can see that in the two scenarios for different values of $r$, MQMCS-c has a higher coverage quality than MQMCS-d. What’s more, the coverage quality of both algorithms quickly increases when $r$ increases from 15 to 85. Then it somehow slowly drops after 85. It is important to note that up to $r=85$, by increasing the $r$, the number of selected sensors increases to swell the coverage percentage (Figure 13). As a result, the coverage quality also increases. After $r=85$, the coverage percentage slowly increases while the number of the selected sensors decreases, because by increasing the $r$, one sensor can cover more targets but with a lower quality. Thus, fewer sensors to cover all targets are needed. Consequently, after $r=85$, the number of both selected sensors and coverages drop.
Fig. 13. Number of selected sensors vs. sensing radius of sensor (r) with N=50, M=40, W=3 and W=8.

7. Conclusion

In this paper, we studied the problem of finding a cover set in order to minimize cost and maximize quality (max-Quality Min-Cost Selection problem). To do so, we defined a criterion for the coverage quality of visual sensors and then formulated the MQMCS problem as a bi-objective mixed integer programming. A centralized algorithm named MQMCS-c and a distributed algorithm named MQMCS-d were proposed for presenting a cover set with the maximum quality and the minimum cost. It is concluded that the MQMCS-c algorithm has a higher possibility to find a cover set, and has a greater coverage percentage and coverage quality than the MQMCS-d algorithm.

References


