A Graph Theoretic Approach for Maximizing Target Coverage using Minimum Directional Sensors in Randomly Deployed Wireless Sensor Networks

***Abstract*—The coverage maximization by activating minimum number of sensors is one of the most fundamental problems in wireless sensor networks. The use of directional sensors adds new dimension to it. It has great applications in security, tracking, monitoring and surveillance systems. As the problem is known to be NP-hard, most of the previous works attempt different greedy heuristics for achieving a near optimal solution. Unlike other traditional research works, we tackle the problem in a graph theoretic approach and provide novel solutions using different polynomial order graph algorithms. Use of graph models significantly reduces the run time complexity and also increases the optimality of the solution.**

I. Introduction

Coverage problem is one of the fundamental problems in the context of wireless sensor networks. A number of studies have been conducted to achieve maximum target coverage by activating the least amount of sensors. More elaborate studies are available in the literature for the so-called *isotropic wireless sensors* which have a circular or disk shaped uniform sensing regions. Consequently, different algorithms have been developed for coverage maximization using such sensors. But a limited number of works are available for *directional sensors* having tunable orientations. A directional sensor is able to sense in a particular direction at a time by fixing its orientation. Directional sensor networks have many applications such as surveillance, monitoring and tracking of objects, sensing and environmental monitoring, healthcare etc., to name a few. A very common example of directional sensors is 'cameras' which has become an essential device in the security domain.

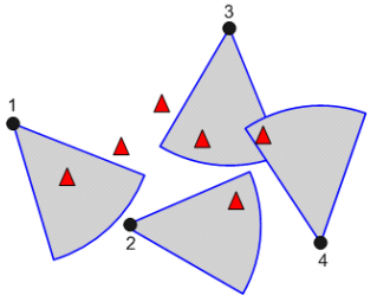


Figure 1: A random deployment scenario.

The solution to coverage problem focuses on two aspects; maximization of the coverage area or maximization of the target (i.e., discrete objects) coverage. Both approaches require use of minimal sensors. The process of selecting a sensor usually follows two steps; (i) selecting a sensor and, (ii) fixing its orientation. The process is proven to be an NP-complete problem. Therefore, to get a near optimal solution in polynomial time, we provide several heuristics in this work. Most of the existing works on directional sensor networks are implemented using Greedy approach. Although a greedy approach can provide near optimal solution in most of the scenarios, there are many cases where it fails. Another issue with the greedy approach is that the process of selecting a sensor and fixing its orientation is merged together. Splitting these two steps can provide more flexibility in developing many heuristics as we demonstrate in this paper.

In this paper, we took an unorthodox approach (unlike existing works) in the problem formulation and providing necessary solutions. We model the problem using graph theoretic approach so that one can have the opportunities to experiment with already-known graph algorithms having polynomial time complexity. We develop a weighted multi graph from the configurations of the sensors and targets that we term as a 'conflict graph'. Using the information available from the conflict graph, we can easily develop different heuristics for selecting a sensor. Then we fix the orientation of the chosen sensor greedily. We have considered a random deployment environment since deterministic deployment is impractical. In inclement scenario, random deployment (see fig. 1) is more suitable. Here, a fixed number of targets and sensors are randomly deployed. The locations of all the targets are known to each sensor. Thus, each sensor knows which targets are covered by it in each orientation. Using this information we can develop a conflict graph where each sensor acts as a vertex and a weighted edge is added between two vertices if there is at least one target in common that is covered by the both sensors for some orientation of each of them. The weight of each edge represents the number of common targets covered by the two sensors at their particular orientations. We then implement different heuristics using this conflict graph to solve the target coverage problem.

In the literature, the discrete target coverage problem was viewed from two different perspectives:–*sensor-oriented* and *target-oriented*. All the existing works are based on either one of them. But the proposed heuristic combines the merits of both the approaches to attain better performance. There is another advantage of using conflict graphs. In greedy

approach, each selection affects the problem model in such a way so that it requires sorting of the remaining sensor-orientation pairs based on the number of targets covered by them. But conflict graph approach localizes this effect of sensor selection within the neighboring sensors only. Thus, the conflict graph approach has an upper hand over greedy approach with respect to time complexity. Besides, most visual sensor networks (VSN) are over-provisioned systems where there exist enough sensors to meet the coverage requirement. Greedy approach performs well for under provisioned systems where there do not exist enough sensors to cover all the targets, but performs poorly in an over-provisioned system. The proposed conflict graph approach provides a way around to this drawback of the greedy approach.

II. Background and Problem Formulation

In this section at first we introduce the VSN with relevant parameters. Then we formalize the coverage problem that we are trying to solve in this thesis.

A. Visual sensor network description and parameters

The sensing region of a directional sensor (camera) can be characterized by its Field of View (FoV) defined as follows:

Field of View (FoV): The FoV of a camera is the extent of the observable/sensing region that can be captured at any given direction. Some cameras come with fixed-FoV and for some, FoVs are adjustable. The smart cameras used in current VSNs are known as Pan-Tilt-Zoom (PTZ) cameras where FoV can be self-adjusted in three dimensions:

i) horizontal movement in pan
ii) vertical movement or tilt
iii) change in depth-of-field by changing zoom

In this thesis, we limit ourselves to pan-only cameras, i.e., we assume that a camera can move only in horizontal direction and its FoV is only described by its pan.

The pan of a camera is formally defined using the following two parameters:

- R_s: Maximum coverage range of the camera beyond which a target cannot be detected with acceptable accuracy in a binary detection test.
- θ: The maximum sensing/coverage angle of a camera in a certain direction. This angle is also known as Angle of View (AoV).

Thus, when a camera is oriented towards a particular direction, it can cover a circular sector (called a pan) defined by R_s and θ. We assume that every camera possesses a specific number of non-overlapping pans, of which, only one can be selected in a particular deployment. For example: a camera with FoV defined by $\theta = \frac{\pi}{4}$ can pick any one of eight disjoint orientations. Fig. 2 depicts these parameters of camera coverage. Here, two cameras c_1 and c_2 have eight pans each and can be oriented towards any of these eight pans. We assume that cameras are homogeneous in terms of parameters. Position of a target and a sensor are expressed through Cartesian coordinates (x, y) in a two-dimensional plane. $\vec{d_{ik}}$ is a unit vector which cuts each pan (i.e., the sensing sector) into half representing the orientation of camera c_i towards pan p_k. $\vec{v_{ij}}$ is a vector in the direction from camera c_i to target t_j.

Target in Sector (TIS) Test: With TIS test (Ai and Abouzeid, 2006) one can verify whether a target t_j is coverable by a given sensor c_i. To conduct this test, at first we calculate the angle ϕ_{ij} between camera orientation $\vec{d_{ik}}$ of pan p_k and the target vector $\vec{v_{ij}}$.

$$\phi_{ij} = cos^{-1}\left(\frac{\vec{d_{ij}}.\vec{v_{ij}}}{\left|\vec{d_{ij}}\right| |\vec{v_{ij}}|}\right) \tag{1}$$

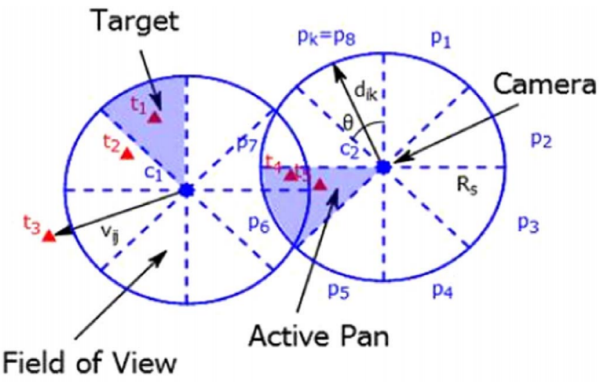


Figure 2: Coverage parameters of a directional sensor

A target is coverable by a camera's FoV if the span of its FoV contains the target and the target is located within the sensing range of the camera. Geometrically, $\vec{d_{ik}}$ divides the pan p_k into two equal halves and if a target is located in either of them, it is coverable by that camera on the pan p_k. Thus, the angle ϕ_{ij} needs to be less than or equal to half of the AoV, i.e., $\frac{\theta}{2}$. The other condition requires that the target has to be inside the maximum sensing range of the camera, i.e., $|\vec{v_{ij}}| \leq R_s$.

Conducting TIS tests over every pan p_k of camera c_i and every target t_j, we can build a binary coverage matrix $A^M_{N\times Q}$ of the network comprising of M targets and N cameras with Q pans where an entry in the matrix can be calculated as:

$$a^j_{ik} = \begin{cases} 1 & \text{if target } t_j \text{ is covered by camera } c_i \text{ at pan } p_k \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

B. Problem formulation

In this section we provide a formal description of the problem. Suppose we have a set of M randomly deployed

targets $\tau = \{t_1, t_2, ..., t_M\}$ and a set of N homogeneous directional cameras $C = \{c_1, c_2, ..., c_N\}$, each of which can be (self) orientated towards one of the Q possible pans $\rho = \{p_1, p_2, ..., p_Q\}$. Let us define a tuple $\langle c_i, p_k \rangle$ which is a camera-pan pair denoting a camera c_i oriented towards pan p_k for any $1 \leq i \leq N$ and $1 \leq k \leq Q$. Define a function $F(\langle c_i, p_k \rangle)$ which returns the set of targets covered by the camera-pan pair $\langle c_i, p_k \rangle$. Suppose δ is the set of tuples that can cover at least one target, i.e.,

$$\delta = \{\langle c_i, p_k \rangle : |F(\langle c_i, p_k \rangle)| > 0\} \quad i \epsilon [1, N]\,; k \epsilon [1, Q]$$

The goal is to find a subset $\delta^{'} \subseteq \delta$ such that $\left|\delta^{'}\right|$ is the minimum among all cardinalities of possible subsets of δ, and the quantity $\left|\cup_{\langle c_i, p_k \rangle \epsilon \delta'} F(\langle c_i, p_k \rangle)\right|$ gets maximized.
In other words, we seek to find orientations of minimum number of cameras to cover maximum number of targets which is also traditionally known as *Maximum Coverage with Minimum Sensors* (MCMS) problem.

III. Related Work

Extensive works have been done over the past years in the area of wireless sensor networks for isotropic sensors. Only a few works are done with directional sensors. Generally, directional sensors are considered as visual sensors since they can sense only along a particular direction at any time. The primary goals of the research works in the area of VSN are design issues related to optimal placement and orientation of the visual sensors, energy efficient scheduling techniques to increase network lifetime, cost-effective transmission system for obtained sensor data, and fault tolerant network design to attain secured connectivity of the network. Yap and Yen (2014) [11] have worked on these design issues. Liu et al. (2016) [6] have done a comprehensive survey on different design aspects of VSN. There has been some works regarding optimal placement and orientation of sensors to achieve some predefined goals. Chakrabarty et al. (2002) [2] investigated on optimal placement problem considering the deployment area as a grid of coordinates. They designed an Integer Linear Programming (ILP) model for solving small scale instances of the problem and for solving large scale instances, followed a Divide and Conquer approach. H*ö*rster and Lienhart (2006) [5] devised a rank based greedy heuristic for placing the sensors where rank of sensor was calculated based on its cost and achievable target coverage. Here, we only focus on the problem of selection and orientation of the randomly deployed sensors. In a randomly deployed environment, it is required to keep the number of active sensors at a time to a minimum due to limitation on energy consumption. So, the most important question in this context is how to select the sensors in an appropriate manner? The solution to this problem can be viewed from two directions: - (i) sensor-oriented approach, and (ii) target-oriented approach. There has been several works focusing on these two approaches. All of them aimed at maximizing target coverage with minimum number of sensors. Ai and Abouzeid (2006) [1] proposed an ILP formulation that gave an exact optimal solution to this MCMS problem. But it is not suitable for large-scale scenario due to its high computational complexity and low computational capabilities of the deployed sensors. As a solution to this issue they proposed a Centralized Greedy Algorithm (CGA) and a Distributed Greedy Algorithm (DGA). DGA was introduced to achieve scalability in place of CGA. But CGA had some shortcomings in case of resolving tie between sensors. To come round the shortcomings of CGA, Munishwar and Abu-Ghazaleh (2013) [10] proposed another greedy heuristic called Centralized Force-directed Algorithm (CFA). The modification they made was that if a sensor covered some targets in only one pan, that sensor is selected first. So, they created a way to assign priority to each sensor pans. Instead of activating a sensor having maximum number of uncovered targets in some pans, they decided to activate the sensor having targets only in a single pan first and cover those targets. To find out such sensors, they introduced a concept, force. The force of a sensor-pan pair is the ratio of coverable targets in that pan of the sensor to the total number of uncovered targets available in all pans of that sensor. The higher the value of force, the higher is the priority. Munishwar et al. (2011) [9] also suggested a distributed algorithm to achieve coverage maximization. Here, they overlooked the idea of minimizing active sensors. Initially, all sensors assign a unique priority value by themselves. Then each sensor detects the total number of targets it can cover in each direction and orients itself to the direction of maximum target coverage. This orientation information is exchanged among the sensors located within twice of the sensing range. If there is more than one sensor covering a particular region, the orientation of the highest priority sensor prevails. For assigning priority they proposed two approaches: - (i) an area based approach, and (ii) a target based approach. In area based approach higher priority is assigned to the sensor with lower degrees of overlap in coverage with other sensors. In target based approach, a sensor's priority is higher if it has lower number of useful pans, in other words, it has coverable targets in fewer pans than other sensors. All research works discussed till now are based on sensor oriented approach. There has been works on target oriented approach too. H. Zannat et al. (2016) [12] proposed three different heuristics for target oriented approach. The proposed algorithms are Greedy Target Oriented Heuristic (GTOH), Pure Target Oriented Heuristic (PTOH), and Hybrid Target Oriented Heuristic (HTOH). This approach differs from sensor oriented approach in a very vital aspect. In sensor oriented approach, priority is assigned to sensor and all targets are treated equally. But here, targets are assigned some priority and in each iteration, higher priority targets are selected and a sensor-pan pair is assigned to it. In GTOH, targets coverable by a single sensor (lonely targets) is given the highest priority. However, targets coverable by more than one sensors are assigned equal priority without any consideration of number of sensors covering them. PTOH differs from GTOH in the weight assignment of targets and the sensor selection criteria at each iteration. HTOH combines

the ideas of both GTOH and PTOH. The weight assignment of targets and rank calculation of sensor-pan pairs are similar to PTOH. But the rule for selecting sensor-pan pairs follows GTOH. There has been a few works (Fusco and Gupta, 2009 [4]; Lu and Yu, 2014 [7]; Costa et al., 2014 [3]) focusing on k-coverage problem in directional sensor networks. Costa et al. (2014) [3] proposed an algorithm to achieve a high level of monitoring redundancy for some critical targets by concurrently viewing them using more than one visual sensor. The optimal solution to k-coverage problem has been proved to be NP-hard by Fusco and Gupta (2009) [4]. They provided a centralized greedy solution to the problem assuming that each sensor has overlapping pans instead of discrete pans. They also proposed approximation algorithms to some related problems on directional sensors: (i) orient all the given sensors in order to maximize coverage, (ii) place and orient a minimum number of sensors in order to cover the given area, and (iii) place and orient the given number of sensors to maximize the area covered. Malek et al. (2016) [8] identified a novel issue called coverage balancing in k-coverage problem and provided centralized solutions to jointly solve k-coverage and coverage balancing. We limit ourselves only to single coverage of each target.

IV. Methodology

In this section we discuss the proposed algorithm which is based on conflict graphs. The section begins with the definition of conflict graphs.

A. Conflict Graph

Using graph theoretic approach at first we build a *conflict graph*. The nodes of this graph are the deployed sensors. The edges between two sensors are created on the basis of the targets in conflict. The targets which are covered by more than one sensors are called conflicted targets. For each pair of orientations between two sensors, there will be an edge if there are one or more conflicted targets. That means each edge actually represents the conflict or coverage overlap between a pair of orientations of two sensors. Thus the graph becomes a weighted multi-graph as there could be more than one edge between two nodes indicating conflicts in multiple orientation pairs between them. The weight of the edges are the total number of conflicted targets involved in an orientation pair. So, we name the graph a *conflict graph*. Let us explain the conflict graph using an example. In figure 3 there are three sensors and fifteen targets that are randomly deployed. Sensor, s_1 has six and one target in range along pans, p_1 and p_5 respectively. Sensor, s_2 has seven and three targets in range along pans, p_3 and p_7 respectively. Lastly, sensor, s_3 has two, one and five targets in range along pans, p_1, p_4 and p_5 respectively. In *conflict graph* of the given scenario of figure 3 there are exactly three nodes that represent three sensors. We observe that there are four overlapping pair of pans where there exist conflicted targets. We add an edge with weight 5 between nodes, s_1 and s_2 since there are five conflicted targets in pan pair, p_1 and p_3 of those sensors respectively. Similarly, we have an

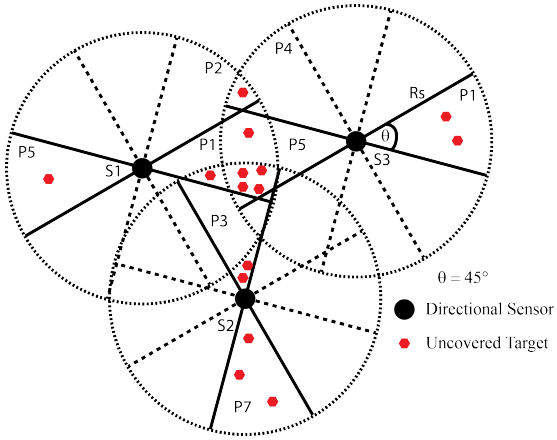


Figure 3: An example scenario

edge with weight 4 between s_2 and s_3 in pans, p_3 and p_5 of them respectively. There are conflicted targets in two pairs of pans between nodes, s_1 and s_3. Therefore, we add two edges with weights 5 and 1 between them in pan-pairs, $\langle p_1, p_5 \rangle$ and $\langle p_2, p_4 \rangle$ respectively (see figure 4). Here, the sum of weights of edges connected to node, s_1 is 11. It means that there are total eleven conflicted targets with other sensors in different pan-pairs of the associated sensor and the other. Similarly, there are total 9 and 10 conflicts associated with sensors, s_2 and s_3 respectively. Besides edges repsenting conflicts, there

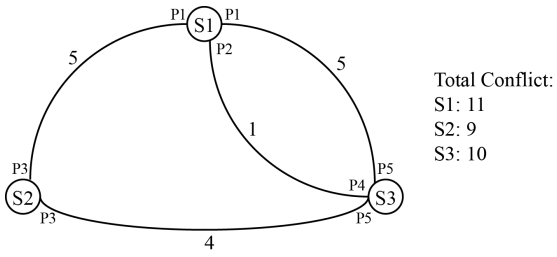


Figure 4: Conflict graph (Weighted multi-graph)

will be another set of self-edges in all sensors in the graph. We name them *shadow* edges which will be created for each orientations of a sensor. The weight of these shadow edges will be different from the other edges. The weight here will represent the total number of non-conflicted or lonely targets in each orientation of a sensor. In target-oriented method, the lonely targets are looked into at first whereas in our approach without looking at the targets we can keep track of the lonely targets. In that sense, we can incorporate the merits of target-oriented method in our approach. Here, we add shadow edges to the *conflict graph* of figure 4 and get the graph as of figure 5. From figure 3, we can see that there is a lonely target in pan, p_5 of sensor, s_1. So, we add an edge with weight 1 between node, s_1 and the the shadow node of itself. Similarly, there is an edge with weight 2 between shadow node of s_3 and itself as there are two lonely targets in pan, p_1 of sensor, s_3. Finally, we add two edges with weights 2 and 3 for pans, p_3 and p_7

of sensor, s_2 for the lonely targets falling in those two pans respectively.

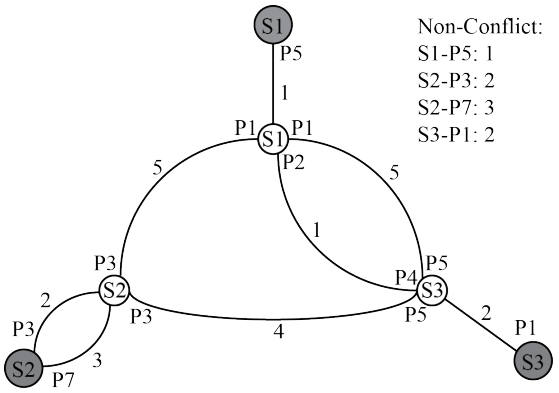


Figure 5: Conflict graph with shadow edges

B. Heuristics

After modeling the graph, we get the information of total conflicts of all the sensors and also the total number of non-conflicted or lonely targets. Unlike traditional methods, we are spitting the selection of sensor and its orientation in two steps. At first we will select the sensor from the modeled graph in one step and next, we will select the orientation from that selected sensor. For selecting the sensor, we can use the heuristic of total maximum or minimum number of conflicts. We can also select that sensor which covers the maximum lonely targets (5). If we select the sensors with the total maximum conflict, we are actually resolving the conflicts among the conflicted sensors which will allow the remaining sensors involved in the conflict to focus more on the lonely targets. Again if the total number of targets covered by a sensor is much more than the number of conflicts, then the sensor is covering mostly the lonely targets. If we select those sensors, this will be more likely target-oriented. After selection of the sensor, for selecting the orientation of that sensor we can greedily take that orientation which covers the most targets. Then the selection process of orientation will be in sensor-oriented greedy approach. When we have the conflict graph, we can apply more different heuristics to find out better solutions for different scenarios.

Total Max Conflict Heuristics (TMxCH). In Algorithm 1 we show how to implement TMxCH from a given scenario. The basic assumption is that both sensors and targets are static. At first, a *conflict graph* is developed from the given co-ordinates of sensors and targets using the coverage parameters of the sensors namely, Range (R_s) and AoV (θ) with the help of **TIS** test. At each iteration, firstly, an unused sensor is selected that is causing the maximum conflict with other sensors. Then, the orientation of that selected sensor in which it covers maximum number of uncovered targets is selected. At the end of an iteration, the *conflict graph* is updated (see IV-C).

Total Min Conflict Heuristics (TMnCH). In Algorithm 2 we similarly form the *conflict graph*. The only change from the TMxCH is that while selecting the sensor, we choose the one causing minimum number of conflicts with other sensors. Finally, the *conflict graph* is updated at the end of each iteration in the same way as well. **TMxCH with Shadow Edge (TMxCHSE)**. In Algorithm 3 from the given *conflict graph with shadow edges* we check for any valid shadow edge. If there is a shadow edge, it means that there are targets which are covered by only one sensor in a particular orientation. We call these targets *non-conflicting or lonely* targets. First, these targets are covered. When all the *lonely* targets are covered, the algorithm follows TMxCH.

Algorithm 1 Total Max Conflict Heuristics (TMxCH)

Require: C = set of unused cameras; T = set of uncovered targets; P = set of discrete pans
Ensure: Z = set of sensor-orientation pairs given by TMxCH
1: model the conflict graph, $G = (V, E)$ where V = {set of nodes (sensors)}; E = {set of weighted edges}
2: **while** $C \neq \phi$ **or** $T \neq \phi$ **do**
3: **for** each $c_i \in C$ **do**
4: sum_i = sum of conflicts with all $p_k \in P$ of $c_j \in C$ for $\forall j \neq i$
5: **end for**
6: $c_i = max(sum_i \ \forall \ c_i \in C)$ {select the sensor with maximum conflicts}
7: **for** each $p_k \in P \ of \ c$ **do**
8: $p_k = max(|t_k|)$ {find the orientation covering maximum targets}
9: **end for**
10: τ = set of targets covered by $\langle c_i, p_k \rangle$ sensor-orientation pair
11: $C = C \backslash \{c_i\}; \ T = T \backslash \{\tau\}$
12: $Z = Z \ \cup \ \{\langle c_i, p_k \rangle\}$
13: $V^{'} = V \backslash \{c_i\}$
14: $E^{'} = E \backslash$ {set of edges connected to c_i}
15: update weight of the edges of the neighboring nodes of c_i
16: $G = (V^{'}, E^{'})$
17: **if** $t_{ik} = \phi \ \forall \ p_k \in P \ of \ c_i \in C$ **then**
18: $break$
19: **end if**
20: **end while**

C. Updating conflict graph

It is already known that which sensor is selected and which targets are covered at this iteration. The selected sensor is removed from the set of unused sensors and also the node representing that sensor in the *conflict graph* is removed from the graph. The tricky part comes afterwards. This removal of a sensor (node) from the current graph has an impact on the new *conflict graph* which will be passed on to the next iteration. In all the existing literatures, for removing the targets which are covered at any iteration, it was required to traverse through all the unused sensors since there was no efficient data structures which could tell that the sensors which had the

Algorithm 2 Total Min Conflict Heuristics (TMnCH)

Require: C = set of unused cameras; T = set of uncovered targets; P = set of discrete pans
Ensure: Z = set of sensor-orientation pairs given by TMnCH
1: model the conflict graph, $G = (V, E)$ where V = {set of nodes (sensors)}; E = {set of weighted edges}
2: **while** $C \neq \phi$ **or** $T \neq \phi$ **do**
3: **for** each $c_i \in C$ **do**
4: sum_i = sum of conflicts with all $p_k \in P$ of $c_j \in C$ for $\forall j \neq i$
5: **end for**
6: $c_i = min(sum_i \ \forall \ c_i \in C)$ {select the sensor with minimum conflicts}
7: **for** each $p_k \in P \ of \ c$ **do**
8: $p_k = max(|t_k|)$ {find the orientation covering maximum targets}
9: **end for**
10: τ = set of targets covered by $\langle c_i, p_k \rangle$ sensor-orientation pair
11: $C = C \backslash \{c_i\}; \ T = T \backslash \{\tau\}$
12: $Z = Z \ \cup \ \{\langle c_i, p_k \rangle\}$
13: $V^{'} = V \backslash \{c_i\}$
14: $E^{'} = E \backslash$ {set of edges connected to c_i}
15: update weight of the edges of the neighboring nodes of c_i
16: $G = (V^{'}, E^{'})$
17: **if** $t_{ik} = \phi \ \forall \ p_k \in P \ of \ c_i \in C$ **then**
18: $break$
19: **end if**
20: **end while**

same targets in their different pans. So, the complexity of this removal process was of $O(n)$. But, we have the *conflict graph*. It holds the information of overlapping coverage for a target between two sensors with the help of an edge between them. So, when a sensor is getting selected, to update the graph for the next iteration, we only need to check the neighboring nodes (sensors). If there is no edge between two sensors, it means that they have no overlapping targets in any of their pans. That is why removing any target under the coverage of any one of them will have no impact to the edges associated with the other one. We only update the weight of the edges of the neighboring nodes (sensors), when we remove a node (sensor) from the graph. This reduces the time complexity to $O(\Delta)$ (Δ means the average number of neighbors of a node in the *conflict graph*).

V. Performance Evaluation

In this section we compare the performance of different *conflict graph heuristics* (CHs) with centralized greedy algorithm (CGA) of Ai and Abouzeid (2006) [1] through various simulations.

Algorithm 3 TMxCH with Shadow Edge (TMxCHSE)

Require: C = set of unused cameras; T = set of uncovered targets; P = set of discrete pans
Ensure: Z = set of sensor-orientation pairs given by TMxCHSE
1: model the conflict graph, $G = (V, E, S)$ where V = {set of nodes (sensors)}; E = {set of weighted edges}; S = {set of shadow edges}
2: **while** $C \neq \phi$ **or** $T \neq \phi$ **do**
3: **if** $S \neq \phi$ **then**
4: $\langle c_i, p_k \rangle$ = the sensor-orientation pair covering most non-conflicting targets
5: **else**
6: **for** each $c_i \in C$ **do**
7: sum_i = sum of conflicts with all $p_k \in P$ of $c_j \in C$ for $\forall j \neq i$
8: **end for**
9: $c_i = max(sum_i \ \forall \ c_i \in C)$ {select the sensor with maximum conflicts}
10: **for** each $p_k \in P \ of \ c$ **do**
11: $p_k = max(|t_k|)$ {find the orientation covering maximum targets}
12: **end for**
13: **end if**
14: τ = set of targets covered by $\langle c_i, p_k \rangle$ sensor-orientation pair
15: $C = C \backslash \{c_i\}; \ T = T \backslash \{\tau\}$
16: $Z = Z \ \cup \ \{\langle c_i, p_k \rangle\}$
17: $V^{'} = V \backslash \{c_i\}$
18: $E^{'} = E \backslash$ {set of edges connected to c_i}
19: $S^{'} = S \backslash$ {set of shadow edges connected to c_i}
20: update weight of the edges of the neighboring nodes of c_i
21: $G = (V^{'}, E^{'}, S^{'})$
22: **if** $t_{ik} = \phi \ \forall \ p_k \in P \ of \ c_i \in C$ **then**
23: $break$
24: **end if**
25: **end while**

Table I: Acronyms used in the plots

Acronym	*Actual term*	*Reference to work*
CGA	Centralized Greedy Algorithm	Ai and Abouzeid [1]
TMnCH	Total Minimum Conflict Heuristics	This paper
TMxCH	Total Maximum Conflict Heuristics	This paper
TMxCHSE	TMxCH with Shadow Edge	This paper

A. Simulation Environment

We consider a fixed area of 1000×1000 square unit, where N cameras and M targets were randomly deployed. Position of cameras and targets are defined as 2-D points (x, y) in Cartesian co-ordinate system. The sensing range, R_s is assumed to be 100 unit, and the Angle Of View (AoV) is set to $\theta = \frac{\pi}{4}$. With an AoV of $\frac{\pi}{4}$, the number of discrete pans, Q becomes 8 ($\because \frac{2\pi}{\frac{\pi}{4}} = 8$).

Scenario generation. We generate two types of static scenar-

ios: (i) sensor-only scenarios, and (ii) target-only scenarios. In order to generate sensor-only scenarios, we vary the number of sensors over a fixed deployment area of 1000×1000 square unit. Number of sensors ranges from N=25 to N=400 in different scenarios with an increment of 25 sensors from one scenario to the other. As we increase the number of sensors, the larger set includes the sensors from the previous scenario. This ensures a consistent evaluation of the impact of the enlarged population of cameras by retaining all the "features" of the previous environment and simply making it better. In order to generate target-only scenarios, 25–225 targets were randomly spread over a 1000×1000 square region with an increase of 25 targets at each step. Similar strategy of including targets from the smaller sets to the larger sets was adopted while generating the target-only scenarios. Finally, we combine one sensor-only scenario (for a certain N) with one target-only scenario (for a certain M) to create a visual sensor network (VSN). Then the proposed heuristics along with the existing algorithm are simulated over these VSNs. For each VSN size (i.e., for a certain N and M), we have generated 5 instances. Performance measures are reported as an average of these 5 random VSNs (unless explicitly stated otherwise).

We have adopted two different approaches to capture the performance during simulation: (i) by varying the number of sensors N while keeping the number of targets M fixed, and (ii) by varying the number of targets M while keeping the number of sensors N fixed.

B. Performance Metrics

We analyze the performance of the different VSNs using three performance criteria: *coverage ratio* (CR), *ratio of active sensors* (RAS), and *target coverage per sensor* (TCPS). These performance metrics are defined as follows:

1) Coverage Ratio (CR): CR is the fraction of targets covered by the selected sensors after running an algorithm, i.e.:

$$CR = \frac{Number\ of\ targets\ covered}{Total\ number\ of\ targets} \tag{3}$$

We assume that the Optimal Solution covers all the targets, the metric CR indicates the proximity of performance of a heuristic with respect to the optimal solution. The CR of the Optimal Solution is always 1. For any heuristic, the higher value of CR indicates that the performance of that heuristic is better.

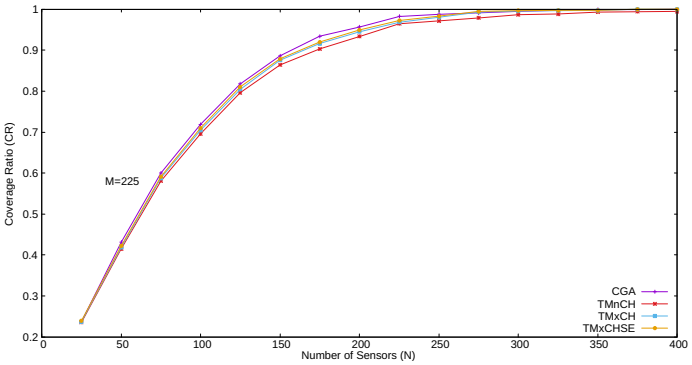
2) Ratio of Active Sensors (RAS): RAS is the fraction of sensors selected by an algorithm:

$$RAS = \frac{Number\ of\ sensors\ activated}{Total\ number\ of\ sensors} \tag{4}$$

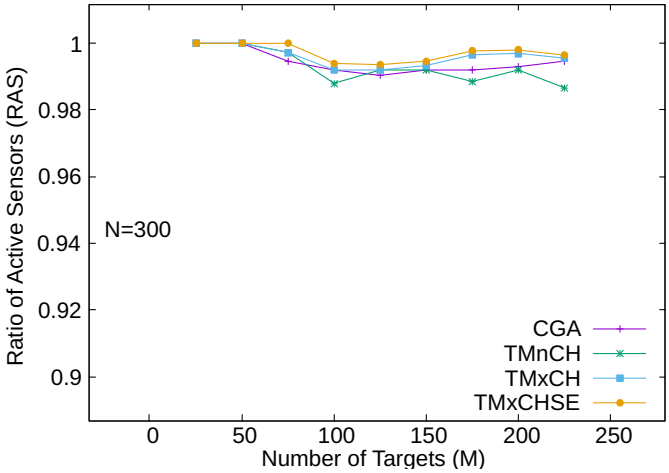
While CR is a measure of target coverage, RAS is a measure of sensor usage. The *smaller* the ratio, the better is the performance.

3) Target Coverage Per Sensor (TCPS): TCPS is the ratio of the number of covered targets and the number of sensors activated, i.e.:

$$TCPS = \frac{Number\ of\ targets\ covered}{Number\ of\ sensors\ activated} \tag{5}$$



(a)



(b)

Figure 6: Coverage performance comparison of CGA, TMnCH, TMxCH and TMxCHSE heuristics

It is to mention that TCPS provides average number of targets covered per sensor. This is an important metric since a particular algorithm may achieve higher RAS value activating a large number of sensors or lower CR value by not covering all the targets. Thus, the better performance of one metric may not reflect the poorer performance of the other metric. Considering this fact, TCPS is introduced which can combine the effect of both sensor usage and target coverage (i.e., considers CR and RAS simultaneously).

C. Results

In this section we provide the performance comparison graphs for each of the proposed *conflict graph heuristics* (CHs) along with centralized greedy algorithm (CGA) of Ai and Abouzeid (2006) [1]. Table I summarizes all the acronyms used in the plots.

1) Target Coverage: At first, we measure the coverage ratio by keeping the number of targets fixed at M=225 and varying the number of sensors N from 25 to 400. The result is shown in figure 6a. All the algorithms converge to the optimal coverage when number of sensors becomes high enough to cover all 225 targets, i.e., the system moves into an over-provisioned one. This happens when the number of cameras

crosses 300. Next we plot CR in figure 6b by keeping the number of sensors fixed at N=300 and varying the number of targets M from 25 to 225. As the number of sensors is lot more than the number of targets, clearly the system is over-provisioned and all approaches have enough sensor to cover all the targets and thereby nearly converge to the optimal coverage.

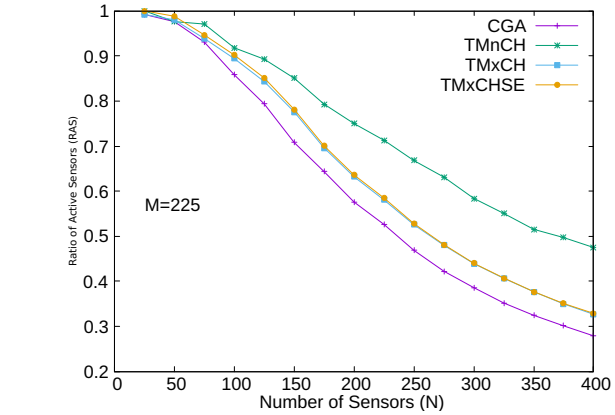
In figure 6a, all the algorithms perform very close to each other. However, CGA has better CR values when the number of sensors are less w.r.t. the number of targets that is, the system is under-provisioned. But gradually, TMxCHSE and TMxCH out-perform CGA when the system becomes over-provisioned. TMnCH has the worst performance among all. It works like a pure target-oriented algorithm. It would perform better when the ratio of lonely targets is very high in the VSN. TMxCH is the opposite of TMnCH. It selects the sensor involved in the most conflicts at first. That is, it tries to resolve conflicting targets first and conflicting targets are just the opposite of lonely targets. TMxCHSE combines the benefits of both TMxCH and TMnCH. While selecting a sensor, it takes both the number of conflicting and non-conflicting targets into account and chooses the best option between the two.

In figure 6b, all the heuristics give almost optimal coverage. Here, the number of sensors is fixed at 300 and the number of targets are varied. TMxCHSE performs the best here as well. TMxCH, TMnCH and CGA follow it respectively. It is to observe that with the increase of the number of targets, the system becomes less over-provisioned and CGA starts to perform slightly better than TMnCH. But still, TMxCHSE and TMxCH remains above CGA in CR values.

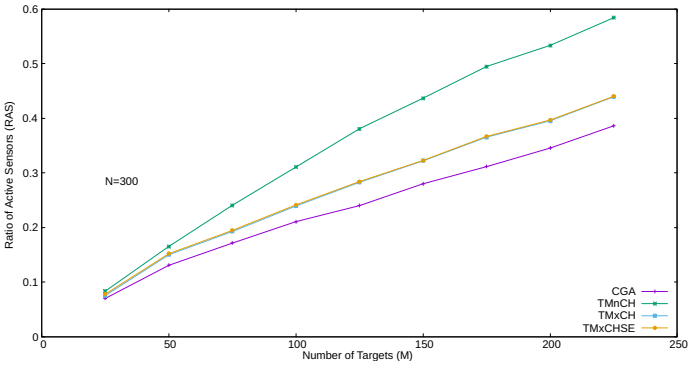
2) Sensor Usage: The ratio of active sensor (RAS) values are shown in figure 7 where we vary both the number of sensors and the number of targets. In figure 7a the number of targets M is kept fixed at 225 and the number of sensors N varies from 25 to 400. As we increase N the system approaches to an over-provisioned one. In terms of sensor usage, CGA of Ai and Abouzeid (2006) [1] performs close to optimal as it greedily activates sensors but does not cover all of the targets. Next better performance is given by TMxCH. TMxCHSE performs almost similar to TMxCH, but it activates a little bit more sensors than TMxCH. We can allow this since TMxCHSE gives better coverage than TMxCH. TMnCH uses more sensors than TMxCHSE. Actually, TMnCH works in a target-oriented approach where it selects the sensor with least conflict first. In other words, TMnCH selects the sensors covering more lonely targets at first. That is why it activates more sensors than other heuristics to achieve the best coverage.

Figure 7b shows the RAS values when we keep the number of sensor N fixed at 300 and vary the number of targets from 25 to 225. When we increase the number of targets while keeping the number of sensors fixed the system becomes less over-provisioned and the ratio of active sensors linearly increases. The performance comparison of different heuristics shows similar behavior as of figure 7a.

3) Target Coverage Per Sensor: Finally, the *conflict graph heuristics* (TMxCHSE and TMxCH) come out as the best



(a)



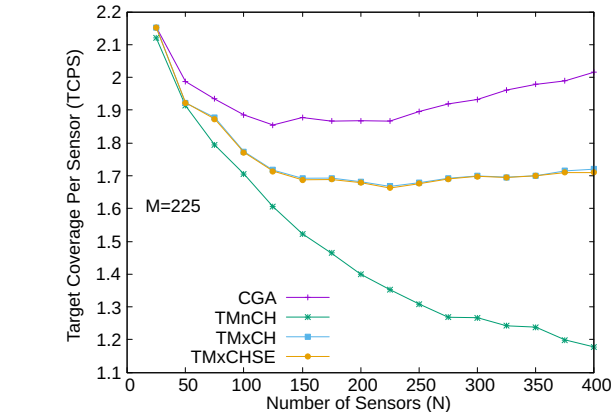
(b)

Figure 7: Sensor usage comparison of CGA, TMnCH, TMxCH and TMxCHSE heuristics

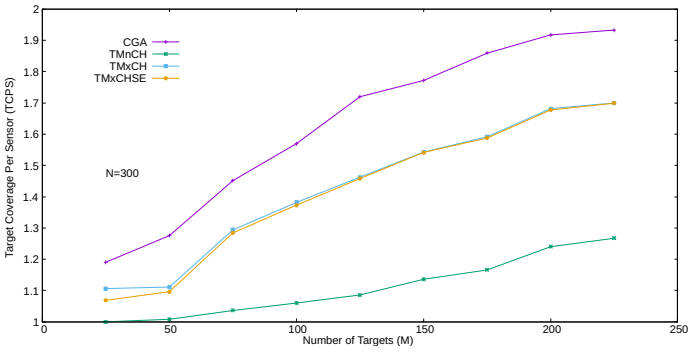
one's when we look at figure 8a and 8b. A question may arise since CGA shows higher values in TCPS plots. Though it covers more targets per sensor, it loses to both TMxCHSE and TMxCH in ensuring higher coverage to an over-provisioned VSN. From both of the figures, it is clearly evident that for different number of sensors, TCPS of TMxCHSE is convergent to the optimal as it provides higher coverage using fewer sensors. Between TMxCH and TMxCHSE, TMxCH activates less sensors as its RAS value is better as shown in figure 7a but it has poorer coverage w.r.t. TMxCHSE as shown in figure 6a. When we scale the coverage measure to camera usage measure, i.e., consider *coverage/camera usage* as the metric of interest (i.e., the TCPS), performance of TMnCH is not as good as TMxCHSE and TMxCH.

VI. Conclusion

We have presented a graph theoretic approach to coverage problem of Visual Sensor Networks (VSNs) that provides near-optimal performance to achieve maximum target coverage using minimum number of sensors. We have formulate the problem using a *conflict graph* and describe our proposed heuristics along with their implementation. We have proposed three heuristics in this paper each of which tries to tackle the problem in different way. TMxCH can be interpreted as a sensor-oriented approach. TMnCH is quite the opposite of



(a)



(b)

Figure 8: Target coverage per sensor (TCPS) comparison of CGA, TMnCH, TMxCH and TMxCHSE heuristics

TMxCH. It attacks the problem in a target-oriented way by giving higher priority to the sensors involved in least conflicts. Lastly, TMxCHSE introduces the concept of shadow edge which enables us to look into the problem in both sensor-oriented and target-oriented ways at the same time. Then we have presented the performance evaluation of our heuristics varying the number of targets while keeping the number of sensors fixed and vice versa. We have simulated performance of CHs and CGA and taking Coverage Ratio (CR), Ratio of Active Sensors (RAS), and Target Coverage Per Sensor (TCPS) as metrics. The results show that CHs cover more targets achieving higher CR, and activating less number of sensors having lower values of RAS. It is also mentionable that CHs produce higher TCPS ratio in over-provisioned systems which supports our claim that *conflict graph heuristics* is a better solution for such systems.

There are yet some open problems to answer. For this study, we limit ourselves to *pan-only* sensors i.e., we allowed only horizontal movements of the sensors. Orientations in other dimensions (like tilt and zoom) are also possible. We only consider uniform distribution of the sensors and targets. It will be interesting to see how the proposed heuristics perform under other kinds of distribution systems. Our future work plan includes applying this graph theoretic approach to other variants of this problem; such as k-coverage [4], balanced k-coverage [8], heterogeneous k-coverage, etc.

REFERENCES

[1] AI, J., AND ABOUZEID, A. Coverage by directional sensors in randomly deployed wireless sensor networks. *Journal of Combinatorial Optimization 11*, 1 (2006), 21–41.

[2] CHAKRABARTY, K., IYENGAR, S. S., QI, H., AND CHO, E. Grid coverage for surveillance and target location in distributed sensor networks. *IEEE Transactions on Computers 51*, 12 (2002), 1448–1453.

[3] COSTA, D. G., SILVA, I., GUEDES, L. A., PORTUGAL, P., AND VASQUES, F. Enhancing redundancy in wireless visual sensor networks for target coverage. In *Proceedings of the 20th Brazilian Symposium on Multimedia and the Web* (2014), ACM, pp. 31–38.

[4] FUSCO, G., AND GUPTA, H. Selection and orientation of directional sensors for coverage maximization. In *Proceedings of the 6th Annual IEEE Communications Society Conference on, IEEE Sensor, Mesh and Ad Hoc Communications and Networks* (2009), IEEE, pp. 1–9.

[5] HÖRSTER, E., AND LIENHART, R. On the optimal placement of multiple visual sensors. In *Proceedings of the 4th ACM International Workshop on Video Surveillance and Sensor Networks* (2006), ACM, pp. 111–120.

[6] LIU, J., SRIDHARAN, S., AND FOOKES, C. Recent advances in camera planning for large area surveillance: a comprehensive review. *ACM Computing Surveys 49*, 1 (2016), 6:1–6:37.

[7] LU, S., AND YU, S. A fuzzy k-coverage approach for rfid network planning using plant growth simulation algorithm. *Journal of Network & Computer Applications 39* (2014), 280–291.

[8] MALEK, S. M. B., SADIK, M. M., AND RAHMAN, A. K. M. On balanced k-coverage in visual sensor networks. *Journal of Network and Computer Applications 72* (2016), 72–86.

[9] MUNISHWAR, V. P., AND ABU-GHAZALEH, N. B. Target-oriented coverage maximization in visual sensor networks. In *Proceedings of the 9th ACM International Symposium on Mobility Management and Wireless Access* (2011), ACM, pp. 175–178.

[10] MUNISHWAR, V. P., AND ABU-GHAZALEH, N. B. Coverage algorithms for visual sensor networks. *ACM Transactions on Sensor Networks 9*, 4 (2013), 45.

[11] YAP, F. G., AND YEN, H. H. A survey on sensor coverage and visual data capturing/processing/transmission in wireless visual sensor networks. *Sensors 14*, 2 (2014), 3506–3527.

[12] ZANNAT, H., AKTER, T., TASNIM, M., AND RAHMAN, A. The coverage problem in visual sensor networks: A target oriented approach. *Journal of Network & Computer Applications* (2016).