This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the ICC 2008 proceedings.

Self-Orienting Wireless Multimedia Sensor Networks for Maximizing Multimedia Coverage

Nurcan Tezcan Wenye Wang
Department of Electrical and Computer Engineering
North Carolina State University
Email: {ntezcan,wwang}@eos.ncsu.edu

***Abstract*— The performance of a wireless multimedia sensor network (WMSN) is tightly coupled with the pose of individual multimedia sensors. In particular, orientation of an individual multimedia sensor (direction of its sensing unit) is of great importance for the sensor network applications in order to capture the entire image of the field. In this paper, we study the problem of *self-orientation* in a wireless multimedia sensor network, that is finding the most beneficial pose of multimedia sensors to maximize multimedia coverage with occlusion-free viewpoints. We first propose a *distributed algorithm* to detect a node's multimedia coverage and then determine its orientation, while minimizing the effect of occlusions and total overlapping regions in the sensing field. Our approach enables multimedia sensor nodes to compute their directional coverage, provisioning self-configurable sensor orientations in an efficient way. Simulations show that using distributed messaging and self-orientation having occlusion-free viewpoints significantly increase the multimedia coverage.**

I. Introduction

As more sophisticated sensing electronics are manufactured cheaper everyday, the nature of the information to be hauled by wireless sensor networks (WSNs) change. We are now able to capture audio-visual information from the environment using low-cost, low-resolution cameras embedded to the sensor nodes. The need for using such multimedia sensors is usually driven by the necessity of providing comprehensive information pertaining to a specific region of interest. From the perspective of sensor networking, considerable works are present for omnidirectional coverage problem [2], [3], which aim to cover a plane by arranging circles on the plane. A common limitation of these existing protocols [2], [3] is that the collected information on phenomena (e.g., temperature, concentration of a substance, light intensity, pressure, humidity, etc.) are assumed to come from a *omni-directional* sensing. However, multimedia sensors, (i.e., low-resolution cameras, microphones, etc.) have the unique feature of capturing direction-sensitive multimedia content. Especially, video sensors can only capture useful images when there is line of sight (LOS) between the event and itself [1]. Hence, coverage models developed for traditional wireless sensor networks are not sufficient for deployment planning of a multimedia sensor network.

Finding the most favorable orientation for multimedia sensors for maximizing multimedia coverage is an important and challenging problem. First, WMSNs are composed of large number of interconnected low-cost sensor nodes having battery-operated, low energy consumption multimedia sensors, e.g., smart camera, low-resolution imaging sensors [1]. Second, multimedia coverage is highly occluded by any obstacle in the environment (e.g., trees, buildings, lakes, etc.). In such WMSNs having large number of nodes, inherent disadvantages due to physical obstacles can be turned into a multi-modality advantage, with the flexibility to adjust orientations of the multimedia sensors attached to the wireless nodes.

There have been several works on vision planning which take the object geometry information as an input from a database, as well as mods of the camera and the lens to determine camera poses and settings [8]. Therefore, orientation of multimedia sensors can be performed on site once the multimedia sensors have been deployed. However, such methods need accurate field information database before deployment and are mostly applied to a small number of multimedia devices. Due to external effects or application-specific queries in WMSNs, multimedia nodes may need to change/re-orient their pose over time. In WMSNs, nodes may fail due to battery outage or external effects which should be handled by a dynamic update of the poses which can be performed via local information exchange among sensors.

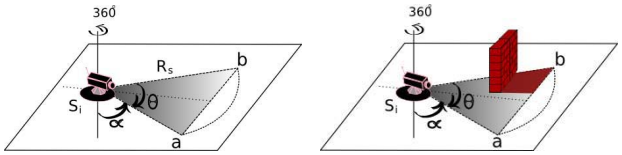
In this paper, we present a *distributed algorithm* that finds the most beneficial orientations for the sensors used in a WMSN. We specifically consider (i) minimizing the effects of occlusion in the environment and (ii) improving the cumulative quality of the information sensed from the region of interest. Using the proposed algorithm, each node discovers its neighbors and examine possible overlapping sensing regions as well as the obstacles in the environment. Nodes use only the local information and communication overhead is incurred only between neighboring nodes. Each sensor node then determines the most beneficial orientation for its multimedia sensor so that the entire image of a field can be constructed using low-resolution snapshots from multiple sensors. Our approach enables multimedia sensor nodes to monitor their coverage performance, provisioning self-configurable sensor orientations.

The remainder of the paper is organized as follows. We summarize the challenges on multimedia converge and define multimedia coverage problem in Section II, and propose a new distributed algorithm for multimedia coverage calculation in Section III. Performance evaluation is discussed in Section IV, and Section V concludes the paper.

II. Multimedia Coverage and Self-Orientation

As audio-visual sensors take their places on wireless nodes, omnidirectional sensing range assumption loses ground significantly since a typical audio or video sensor has a sectoral perception and effected by surrounding obstacles heavily. Multimedia sensors, such as cameras, are powerful multi-dimensional

[0]This work is supported in part by National Science Foundation under Award ECCS-0524519.



(a) (b)

Fig. 1. Illustration of two dimensional field of view (FoV) of a multimedia sensor node, where α is the vertical angle to the boundary edge of FoV , Θ is the FOV vertex angle, and R_s is the maximum multimedia sensing range.

sensors that can capture a directional view, usually called *Field of View (FoV)*. The most commonly used low-resolution camera module is equipped with a lens providing a 45^o FoV [8]. In this work, we assume sensors nodes have a *fixed* lenses providing field of view with angle Θ, and they can only pan to adjust their FoV. We use the term "*camera sensors*" for simplicity to represent wireless multimedia sensors including video and audio sensors having directional view. We also assume that each node is equipped to learn its location information via any lightweight localization technique for wireless sensor networks [4].

A sensor is called *self-orienting*, if it is capable of adjusting its pose at the point of deployment (low-cost multimedia sensors [6] that are capable of panning). In this context, term *field of view* refers to the directional view of a multimedia sensor and assumed to be an isosceles triangle (two-dimensional approximation). A field of view of a sensor s_i is denoted by $\Lambda_i^{\Theta_i}$, where the parameter Θ_i is the vertex angle of the isosceles triangle.

We defined visible FoV, denoted by $v\Lambda_i^{\Theta_i}$, as a FoV of a sensor node s_i which is visible to the sensor itself, i.e., has not been blocked by any obstruction within FoV, $\forall$ obs_j in $\mathcal{A}$, if $obs_j \cap \Lambda_i^{\Theta_i} = \emptyset$, then $\Lambda_i^{\Theta_i} \Rightarrow v\Lambda_i^{\Theta_i}$, where obs_j is an obstacle in the sensing field. The contrary of vFoV is the *occluded FoV* such that, $\forall$ obs_j in $\mathcal{A}$, if $obs_j \cap \Lambda_i^{\Theta_i} \neq \emptyset$, then $\Lambda_i^{\Theta_i} \Rightarrow o\Lambda_i^{\Theta_i}$. The visible FoV is referred to as *overlapping FoV* if it intersects with any of the neighboring sensor's visible FoV, $\forall s_j \in \mathcal{C}_i$, if there exists $v\Lambda_i^{\Theta_i} \cap v\Lambda_j^{\Theta_j} \neq \emptyset$, then $\Lambda_i^{\Theta_i} \Rightarrow x\Lambda_i^{\Theta_i}$.

The FoV disk associated with a sensor defines the set of all possible FoVs. For simplicity, we assume that the orientation of all sensors can be anywhere in between $[0^0, 360^0]$; thus, FoV disk is a circular disk having a radius of R_s, i.e., the maximum distance to capture with a given resolution.

III. A Distributed Solution to Multimedia Sensors Self-Orientation

In this section, we will explain the details of the self-orientation algorithm for a multimedia sensor network. Our algorithm start exchanging messages between neighbors to collect this neighborhood information. All sensors broadcast a `HELLO_MSG` indicating their unique sensor IDs and their location coordinates. We assume that stationary sensors having identical FoV ranges are located in the sensing field. Initial messaging ensures that every sensor is aware of its neighbors and their locations. The rest of algorithm has two major phases: (i) distributed FoV detection and (ii) self-orientation algorithm. Next, we walk through each phase in detail.

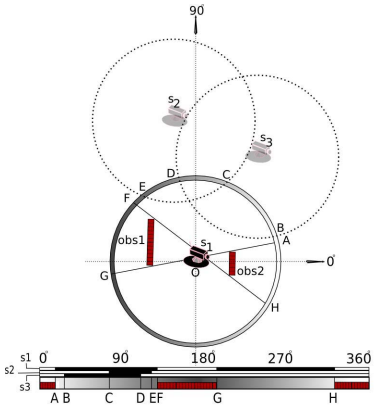


Fig. 2. An example showing the perimeter test for sensor s_1.

A. Distributed FoV Detection

Distributed FoV detection uses *three* consecutive tests to detect sensor's maximum visible FoVs. The first test, namely *perimeter test*, checks the existence of a visible FoV within $[0^0, 360^0]$. If a sensor fails to find a visible FoV during the perimeter-test, it moves to the second test called *neighbor-distance test* which examines the distance with FoV neighbors. Finally, *obstacle-distance test*, is performed if the sensor fails from the neighbor-distance test which compares the occluded FoVs to find the largest visible FoV. Here, we explain these three tests in detail as follows:

1) Perimeter Test: In perimeter-test, each sensor scans its *FoV disk perimeter* to determine whether a visible FoV (which can not be captured by any other FoV neighbor) exists in its FoV disk. The reason is that FoV disk perimeter can effectively show occlusions and possible overlapping regions. The intersection points of any tangent touching an existing obstacle on the perimeter can be used to determine the size of occlusion. For example in Fig. 2, FoV disk of sensor s_1 is illustrated. There are two obstacles inside its FoV disk which are close enough to s_1 that may result in occlusion. The intersections of the tangents on the perimeter are shown with points F and G for the first obstacle (obs1); H and A for the second obstacle (obs2). Therefore, a sensor s_i can determine that if there exists a $\Lambda_i^{\Theta_i}$ where $\Lambda_i^{\Theta_i} \cap \Lambda_i^{\angle FOG} = 0$ or $\Lambda_i^{\Theta_i} \cap \Lambda_i^{\angle HOA} = 0$ then $\Lambda_i^{\Theta_i}$ is a *visible* FoV and we refer arcs $\widehat{FG}$ (counter clock-wise) and $\widehat{HA}$ as *occluded arcs* on the FoV disk of s_1.

```
perimeter_test () {
  for each {obs_k in FoV disk}
    compute the occluded arc of obs_k
  for each {s_j ∈ C_i}
    compute the overlapped arc of s_j
  scan perimeter (0^0 ,360^0) for any arc P_m P_n
    if {(P_m P_n) > Θ} and {visible(P_m P_n ) == TRUE } and
      {overlapped ( P_m P_n ) == FALSE }
       return PASS;
     else {return FAIL;}
```

Fig. 3. Pseudo code of perimeter test.

Perimeter-test not only finds the visible FoV but also helps

to determine non-overlapping FoVs in a FoV disk. In this step, sensors do not know the orientations of their FoV neighbors. However, they can determine possible overlapping FoVs inside their FoV disks. Similar to occluded arcs, each sensor finds possible *overlapping arcs* on its perimeter using the location information received from its neighbors. To do this, the intersection points of the arcs are determined and the perimeter is scanned as illustrated in Fig. 2. For example, sensor s_1 has an overlapping arc $\widehat{BD}$ and $\widehat{CE}$.

By examining each FoV neighbor and obstacles, a sensor decides whether occluded and overlapped arcs enclose its perimeter from 0^0 to 360^0 [5]. If there is a $v\Lambda_i^{\Theta_i}$ with $\Theta_i \geq \Theta$ such that $x\Lambda_i^{\Theta_i}$ does not exist, we refer that "perimeter-test" is passed. This means that the sensor has a visible FoV which has not been captured by any other sensor in any orientation. Since our goal is to maximize the visible FoV in the total sensing region, sensors which pass the perimeter-test will adjust their pose. On the other hand, sensors that do not pass the perimeter-test continue the FoV detection with the neighbor-distance test, which will be explained in the following subsection.

neighbor_distance_test () {
  scan perimeter (0^0 ,360^0) for any arc $\widehat{P_mP_n}$
    if {occluded($\widehat{P_mP_n}$) == FALSE } *and* { ($\widehat{P_mP_n}$) $> \Theta$}
      if {$x\Lambda_i^{\widehat{P_mP_n}}$ exists with FoV neighbor s_j}
        find the distance $d(i,j)$
      return PASS;
      store s_j with max $d(i,j)$;
    else { **return** FAIL; }}

Fig. 4. Pseudo code of neighbor-distance test.

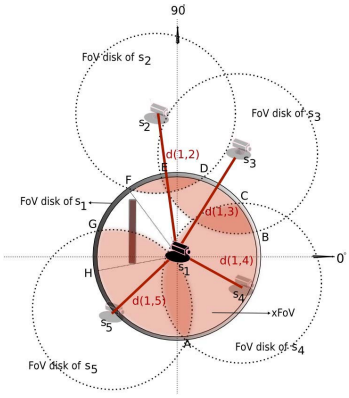


Fig. 5. An example showing the neighbor-distance test for sensor s_1.

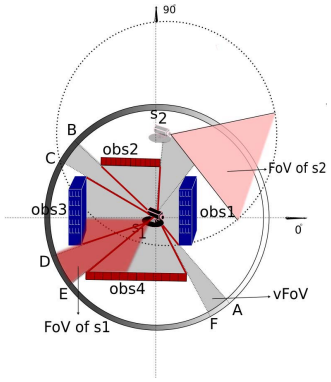


Fig. 6. An example showing the obstacle-distance test condition for sensor s_1.

2) Neighbor-Distance Test: Passing the parameter test implies that a sensor has visible FoV, which can not be covered by its neighbors in any orientation (non-overlapped in any case). In neighbor-distance test, however, we examine whether a sensor has visible FoV which might be overlapped. If a sensor has a $v\Lambda_i^{\Theta_i}$ with an angle $\Theta_i \geq \Theta$ in its perimeter, it is assumed to pass the neighbor-distance test, otherwise it moves to obstacle-distance test. Sensors that pass the neighbor-distance test then find the largest visible FoV based on neighbor's distances.

Even though the final orientations of the neighbors are not known, FoV neighbors might have overlapping FoVs. In this case, sensors need to find the smallest overlapping FoV by scanning visible arcs and calculating the distances between each neighbor. A closer neighbor implies a larger overlapping FoV. In Fig. 5, FoV disk of sensor s_1 and its neighbors are shown. Since perimeter of s_1 is enclosed by an occluded arc $\widehat{FH}$ and overlapping arcs $\widehat{FA}$, $\widehat{BC}$, $\widehat{DE}$, and $\widehat{GA}$, sensor s_1 fails the perimeter-test. However, it passes the neighbor-distance test, since arc $\widehat{HF}$ is visible which is greater than Θ, the FoV angle of the camera sensors which is assumed to be fixed. Among the neighbors s_2, s_3, s_4 and s_5, sensor s_2 has the largest distance to s_1, denoted by $d(1,2)$, indicating smallest possible overlapping FoV, shown as dark shaded areas inside the FoV disk.

3) Obstacle-Distance Test: Finally in obstacle-distance test, sensors with no vFoV are examined. Fig. 6 shows an example sensor s_1 surrounded by four obstacles. Since there is no *visible* arc in the perimeter greater than Θ, the final orientation of sensor s_1 will not have a visible FoV. However, by finding the distances between the obstacles and the sensor node, occluded FoV can be minimized by keeping the visible FoV maximized. Similar to neighbor-test, a closer obstacle means a larger occluded FoV. In such conditions, a sensor scans the perimeter in order to find the most beneficial arc Θ, to maximize the visible FoV.

Note that the perimeter of FoV disk may not be fully-occluded or fully-overlapped. For example, In Fig. 6, arc $\widehat{FA}$ and $\widehat{DE}$ are visible and non-overlapped arcs, but smaller than Θ. In such cases, these small segments can be included to the FoV. In Fig. 6, the FoV of sensor s_1 is shown in shaded region which includes the arc $\widehat{CD}$ and occluded regions with larger distance from obstacles.

Note that, in our algorithm, multimedia sensors can update their neighbor list and orientations periodically by taking the advantage of local information exchange. Thus, all tests are performed using up-to-date FoV neighbors and their orientation decisions.

B. Distributed FoV Detection-Based Heuristic Algorithm for Self-Orientation

Under the 360^0 pan-capability assumption, multimedia sensors will determine their pose for self-orienting by using their local information. The dimensions and the locations of the obstacles are assumed to be known by sensors before self-orientation. We do not consider the multimedia sensors as

obstacles with respect to the other multimedia sensors due to their small size. Using the tests presented in Section III-A, we propose a heuristic algorithm as follows:

STEP 1: Sensors send HELLO_MSG that indicates the location of the sensors. For self-orientation, sensors must build a list of FoV neighbors that are close enough to have an overlapping FoV. A received HELLO_MSG is then used to update the neighbor lists. Note that, we assume that the maximum sensing range, R_s, is equal or smaller than the transmission range of the multimedia sensors.

STEP 2: After exchanging HELLO_MSG, each sensor has an up-to-date FoV neighbor list with their locations and priori-known obstacle locations. Next step is performing the perimeter test. As we explained in Section III-A.1, perimeter test checks if a sensor s_i has a visible FoV, $v\Lambda_i^{\alpha_i}$, which can not be captured by any other FoV neighbor in a FoV disk. Thus, when perimeter test is passed, the sensor s_i can self-orient to $v\Lambda_i^{\alpha_i}$ and finalize the self-orienting algorithm. On the other hand, sensors failing the perimeter test will continue the algorithm with the neighbor-distance test.

In particular, perimeter test shows the existence of *at least one* $vFoV$ that can not be observed by others in any orientation. However, there may be more than than one visible FoVs that result in passing perimeter test. Then sensors change their pose to the most beneficial $vFoV$. Here, the term *beneficial* corresponds to having *smallest panning angle* to a self-orienting multimedia sensor. Therefore, a sensor selects a $v\Lambda_i^{\alpha_i}$ with a vertical angle of α_i to the boundary such that $|\alpha_i - \alpha_0|$ is the smallest among all possible $vFoVs$, where α_0 is the current vertical angle. After changing the pose, a sensor should advertise its decision to all its neighbors with a POSE_ADV_MSG and finalize the self-orienting procedure. Then, sensors that have failed in the perimeter test update their neighbor list based on the POSE_ADV_MSGs they received. If a sensor receives a POSE_ADV_MSG from a FoV neighbor, it updates its neighbor list by adding the pose of its neighbor for the next steps.

STEP 3: In step 3, sensors invoke neighbor-distance test to find a occlusion-free FoV. By passing the neighbor distance test, a sensor determines the existence of a visible FoV in the FoV disk. From the visible FoVs, it selects the pose toward the FoV neighbor s_d with maximum distance using candidate pose selection procedure and sends its candidate pose by a CANDIDATE_ACK_MSG to the neighbor s_d. This message indicates the candidate pose of the sensor to its neighbors. Since sensor nodes perform the self-orienting simultaneously, sensors then receive CANDIDATE_ACK_MSG from their neighbors who have passed the neighbor-distance test, thus replying with a ACK_POSE_MSG if no $xFoV$ occurs. Whenever a sensor receives ACK_POSE_MSG, it indicates that the sensor can select this pose safely and finalize the self-orienting procedure. Otherwise, a sensor should repeat the step 5 with the second minimum distance neighbor.

STEP 4: Finally, sensors that failed from perimeter and neighbor distance test perform the last test, obstacle-distance test. Since they have failed from the previous tests, no visible FoV exists in their FoV disk. Sensors will select an occluded FoV with maximum coverage; that is, the pose toward the obstacle with maximum distance similar to the neighbor-distance test. If there is a visible FoV with an angle smaller than Θ, the final pose will be selected from the small vFoV including occluded FoV to maximize the visible region that the sensor will capture.

At the end of this algorithm, each sensor selects its pose and self-orients to maximize total visible FoVs on the sensing field. Next, we show our simulation results for different scenarios.

IV. Performance Evaluation

We have used Ns-2 simulator [7] for the performance evaluation of our algorithms. Simulations have been performed for randomly placed sensor nodes in a rectangular two-dimensional terrain. All sensor nodes have been configured with an *FoV* vertex angle $\Theta = 60^0$, and an R_s of $30m$. A sensing field spanning an area of 250 x $250m^2$ has been used on which the number of sensors were varied to study the system performance from sparse to dense deployments. In the basic scenario, 50 static multimedia sensor nodes are deployed with self-orientation capabilities.

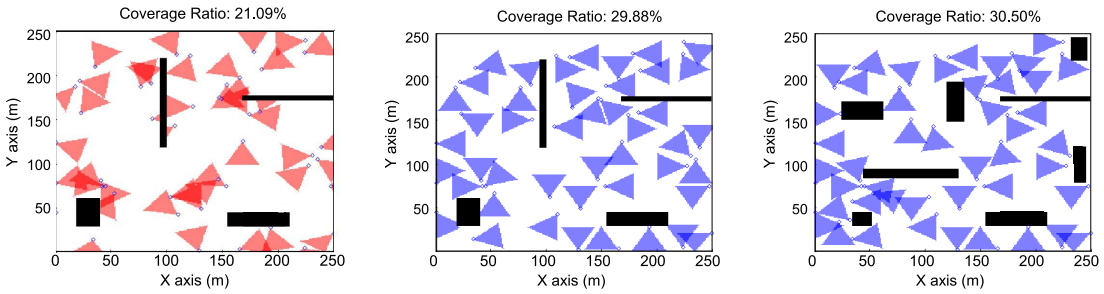
In our simulations, we consider total coverage and messaging overhead as the two key metrics to evaluate the performance of our *self-orienting algorithms*. We assume that global access to obstacle locations on a calibrated coordinate system is available for the sensors. Total visible *FoV* is calculated in a bitmap fashion using 62500 bins (i.e. $1mX1m$ bins for each point) on the $250m$ x $250m$ field. Bins that fall into a sensor's triangular *FoV* are tested for *line of sight (LOS)* view (i.e., line segment from the bin corresponding to the *FoV* point to the camera sensor should not intersect with any obstacle on the field).

The effect of self-orientation on coverage: In Fig. 7, a sensing field with several obstacles (represented by black rectangular areas) and 50 multimedia sensors with $30m$ range is shown. Each multimedia sensor is illustrated with a "small diamond" and its vFoV is shown with a dark shaded area. The FoVs for the network in Fig. 7 (a) are randomly determined, whereas in Fig. 7 (b) and (c) using the proposed self-orienting algorithm.

In Fig. 7 (a), an experiment outcome with random orientation is illustrated, resulting 21.09% overall coverage of the field. Although sensors had the capability to exchange information regarding their neighbors and obstacles, due to the lack of proper coordination, several sensors went overlapping. Mostly occluded FoVs are serious waste of resources. However, in Fig. 7 (b), sensors were programmed to determine their FoV disk, scan their coverage neighbors, obstacles and communicate with their neighbors to decide on the optimal pose. We observed that by using our approach in a 50 node network, a coverage

Nodes	Obstacles	Multimedia coverage gain %
50	4	8.92%
50	8	6.06%
100	4	16.39%
100	8	12.08%
150	4	9.3%
150	8	7.8%

TABLE I
MULTIMEDIA COVERAGE RATIOS.



(a) Random orientation, N=50, 4 random obstacles.

(b) Self-orientation algorithm, N=50, 4 random obstacles with same deployments as (a).

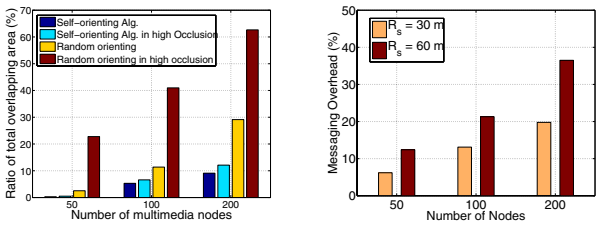
(c) Self-orientation algorithm, N=50, 8 random obstacles.

Fig. 7. Multimedia coverage.

ratio of 29.88% could be achieved, which is very close to the maximum possible coverage with 50 sensors of $30m$ range on this field. Similarly, in Fig. 7 (c), similar sensors deployed on to the field having 8 obstacles. We observed an average of 30.50% coverage ratio that slightly better than the scenario having 4 obstacles. Note that, in our experiments we do not target to reach full coverage but increase the total covered area restricted by given number of nodes that have limited multimedia ranges (corresponds to low-resolution). A set of resultant coverage gain (%) of self-orienting algorithms are also given in Table I for different scenarios. Here, coverage gain is defined as the increase (in %) when self-orintation algorithm is used compared to random orintation in the same deployment. The results are the average of five iterations of each test.

The effect of self-orientation on overlapping area: Self-orienting algorithm not only determines occlusion-free viewpoints for sensors but also avoids overlapping FoVs using neighbor-distance test, as explained in Section III-A.2. For example, in Fig. 8, coverage ratio gains up to 41% were obtained by using self-orientation. Preventing overlapping FoVs contributed 12% of the the total increase in coverage. In Fig. 8 (a), we show the ratio of overlapping FoV when self-orientation algorithm is used. We observe that increase in the number of nodes causes dramatic increase in the total overlapping area. Self-orientation results in at most 9% overlapping area, whereas random orientation results in overlapping areas up to 29%.

The overhead of self-orientation algorithm: For the first test we present, multimedia sensors with a $30m$ range on a field of 250 x $250m$ are used. In Fig. 8 (b), we show the ratio of total number of messages used by the self-orienting algorithm to the total number of control messages, including routing. As we explained in Section III-B, our algorithm uses $O(n)$ messages which is 6% of all control messages on average when $N = 50$. The ratio increases only up to 35% of total control traffic when $N = 200$ and $R_s = 60m$, indicating a very dense network with high degree of connectivity.



(a) Overlapping FoV ratio.

(b) Messaging overhead.

Fig. 8. Performance of self-orientation algorithm.

V. Conclusion

In this paper, we proposed a self-orienting algorithm for multimedia wireless sensor networks in order to *maximum field occlusion* and to attain occlusion-free coverage. We find that (i) the proposed algorithm uses local information; that is, communication overhead is incurred only between neighboring nodes with a complexity of $O(N)$, (ii) the proposed algorithm is a fully distributed, which can operate after initial deployment and update the orientation of multimedia sensors on the fly, (iii) the proposed algorithm can support prioritized or accurate observation that require more than multiple inputs from more than one sensor node, and (iv) coverage is increased even sparse networks by using self-orientation instead of random orientations, for arbitrary obstacles in the sensor field.

References

[1] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury. A Survey on Wireless Multimedia Sensor Networks. *Computer Networks*, 2007.

[2] M. Cardei, M. Thai, and W. Wu. Energy-efficient Target Coverage in Wireless Sensor Networks. In *Proc. of IEEE Infocom*, Miami, Forida, USA, March 2005.

[3] H. Gupta, S. R. Das, and Q. Gu. Connected Sensor Cover: Self-Organization of Sensor Networks for Efficient Query Execution. In *Proc. of ACM Mobihoc*, Annapolis, Maryland, USA, June 2003.

[4] T. He, C. Huang, B. M. Blum, and J. A. Abdelzaher. Range-free Localization Schemes for Large Scale Sensor Networks. In *In Proc. of ACM Mobicom*, pages 81–95, San Diego, California, USA, September 2003.

[5] M.-F. Huang and Y.-C. Tseng. The Coverage Problem in a Wireless Sensor Network. In *Proc. of ACM WSNA*, San Diego, CA, USA, September 2003.

[6] M. Nicolescu and G. Medioni. Electronic Pan-Tilt-Zoom: A Solution for Intelligent Room Systems.

[7] Ns-2. http://www.isi.edu/nsnam/ns. 2004.

[8] K. A. Tarabanis, R. Y. Tsai, and A. Kaul. Computing Occlusion-free Viewpoints. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2:273–292, March 1996.