

A Robust Clustering Algorithm for Target Tracking in Wireless Acoustic Sensor Networks

Raghu Kisore Neeliseti*, Alvin Lim*, Pratima Agrawal[†] and Qing Yang*

*Department of Computer Science and Software Engineering

Auburn University, Auburn, Alabama 36849

Email: [neelira,lim,yanqin]@auburn.edu

[†]Department of Electrical and Computer Engineering

Auburn University

Auburn, Alabama 36849

Email: agrawpr@auburn.edu

Abstract—The advancement of MEMS technologies has made it possible to produce tiny wireless sensor devices. These tiny sensors hold the promise of revolutionizing sensing in a wide range of application domains because of their flexibility and low cost. One such application is target localization and tracking using acoustic signal of the target. The capabilities of these tiny devices are limited by their battery power, storage capacity, computational power and communication bandwidth. These limited capabilities make the decisions made by each sensor error prone. Hence most target detection and tracking algorithms require the sensors to work in groups in order to improve the reliability of target tracking algorithms. This makes it necessary for deployed sensors to discover and group together so that their coverage can be maximized. In addition, with the advent of video sensor networks it has become possible to record a video of the target once it is detected and later be relayed to an external agent. In this paper, we propose a clustering algorithm that tries to produce the maximum number of possible clusters given a certain type of deployment. The proposed clustering algorithm is distributed in nature and has the ability to reconfigure in the event of node failure. The algorithm is highly localized and hence does not need flooding across the entire network. Since the algorithm allows for more clusters to track the same region the system reliability is greatly improved.

I. INTRODUCTION

The increasing capabilities and declining cost of computing and communication devices, has led to an increase in the number of applications of wireless sensor networks. One such application is battlefield surveillance. Surveillance involves both detection and tracking of intruders. Tracking based on the strength of acoustic signal received by a set of sensors is a common technique. This requires deployed sensors to work in groups. Though each sensor is capable of detecting the presence of a target, its results are error prone and could result in false alarms. Hence to increase the accuracy of the detection algorithm it becomes necessary to fuse the measurements of a group of sensors. This makes it necessary for the deployed sensors to work in small clusters so that the overall reliability of the surveillance system can be improved.

Wireless sensors can be used to detect various features such as thermal signatures, ferro-magnetic content or acoustic signal. The absence or presence of a target phenomenon can be inferred by aggregating the measured values from a small

group of sensors deployed. In this paper we assume that each sensor is equipped with a microphone and hence can record the acoustic signal.

Detection requires that the system discriminate between a target's absence and presence. Successful detection requires a node to correctly estimate a target's presence while avoiding false detections in which no target is present. This can be done by using triangulation based on acoustic measurements made by at least three sensors. On the other hand target tracking is more complicated since it involves maintaining the target's position as it moves over time in a region covered by the sensor network's field of view. Therefore the tracking algorithm should be able to identify the orientation of the target's path and its velocity in addition to location of the target's position. One such algorithm to track the target is the CPA (Closest Point of Approach) algorithm [1]. In this algorithm a group of four sensors make CPA measurements, and then, based on the measurements, the trajectory of the target can deduced with reasonable accuracy. The capability of the application can be further enhanced if we assume that each sensor is equipped with a camera. This is possible with the advent of low cost cameras that are capable of providing resolution in the order of mega pixel. The CPA measurements made by each of the four sensors will be reported to a node (actually one of the four sensors) where the target's trajectory can be computed. Once the parameters of the target's trajectory are computed, the camera associated with the sensor can be programmed to pan in the target's direction. However, this requires real time data from each of the four individual sensors, i.e. the CPA measurements generated by the sensors must be delivered to a node in a timely fashion. Out-dated reports are of little use. These timing constraints call for a clustering algorithm.

The overall system architecture consists of two self-contained components: the acoustic target tracking subsystem which deals with the detection and processing of acoustic signals and the communication subsystem which is responsible for exchanging sensor data and high quality tracking results. One way to address the limited computational and battery power of wireless sensor devices is to organize the sensors

into clusters. Sensors in each cluster coordinate in sensing and communication to perform the sensing task. To deal with the inaccuracy in measurement and unreliability typical of low-end devices in remote or hostile environments, we suggest a clustering algorithm that organizes the sensors into redundant clusters so as to obtain more robust results.

In general, target classification and tracking algorithms rely on information provided by a cluster of sensors. In case of target classification each sensor is equipped with different modalities, such as magnetic, radar, thermal, acoustic, chemical, electric, seismic and optical. Hence the target classification draws its results from observations made by a cluster of modalities. This emphasizes the need for a clustering algorithm that can exploit the redundancy in the sensor deployment and reduce the latency in the exchange of raw data and the amount of raw data that needs to be exchanged.

The proposed clustering algorithm is distributed in nature and the number of clusters to be formed can be easily controlled. Further, since the cluster head chooses its member nodes from its one hop neighbors, the raw data has to travel only one hop. Finally, the target tracking results of each cluster head can be progressively fused with those of its neighboring clusters.

The remainder of the paper is outlined as follows. Section II reviews related work; Section III defines the problem, while Section IV provides details of the proposed clustering algorithm, followed by performance evaluation in Section V. Finally, Section VI offers some concluding remarks.

II. RELATED WORK

The system that we are presently referring to is an intrusion detection system which is essentially a surveillance situation of practical importance and is well-suited to wireless sensor networks. The intrusion detection system is designed as a dense, distributed, wireless network of multi-modal, resource-poor sensors combined into loosely coherent sensors that perform in situ detection and estimation. There are several issues of interest in designing such distributed intrusion detection systems. The first and foremost is the sensor deployment algorithms. These algorithms aim at maximizing the field of coverage of a given set of sensors. One metric to identify the effectiveness of a deployment strategy is by measuring the worst and best case coverage paths. In [2] the authors optimize deployment of heterogeneous sensors through Linear Programming. In [3] the authors propose three approximation algorithms for a variation of the SET K-COVER problem, where the objective is to partition the sensors into covers such that the number of covers that include an area, summed over all areas, is maximized. In [4] the authors analyze the minimum number of nodes needed for random deployment so as to meet a desired value for least path of exposure metric. They assume Gaussian distribution for the random deployment strategy. In [5] the authors propose algorithms to provide k-coverage in a mostly sleeping network. The aim of the algorithm is to save energy and at the same time provide certain desired degree of coverage of the protected region at all times. However, all these

algorithms analyze the degree of coverage from the perspective of target detection but not target tracking.

One method of judging the effectiveness of a particular sensor deployment algorithm is by measuring the worst and best case coverage. [6], [7] provide algorithms to measure the worst and best case coverage based on Voronoi diagrams. In [8] the authors analyze worst case coverage (also called the breach path) in case of directional field-of-view sensor networks.

Line in the sand [9] system is a prototype model that can detect and classify up to three different target types. In [9], the authors discuss various issues in developing such systems, largely emphasizing data fusion algorithms. VigilNet [10] is a real time large-scale sensor network system that can track, detect and classify the targets in a timely and energy-efficient manner. In [10], the authors perform mathematical analysis of various delays and accuracy of the system. Both the above systems rely on mutual co-operation of group of clusters. They both assume the availability of a clustering algorithm. In [11], a target detection algorithm localizes a sound source using triangulation based on the acoustic measurements made by a group of three sensors. Once again the existence of clustering algorithm is assumed.

[12] evaluates three different architectures for fusing data collected by the sensors. The three schemes analyzed are a centralized scheme, a progressive scheme and a distributed scheme. A centralized source number estimation scheme is a processing structure in which all sensors send their raw data to a central processing unit where source number estimation is performed. A progressive source number estimation scheme is a processing structure that a group of sensors update the source number estimation result sequentially based on each sensor's local observation and the partial estimation result from its previous sensors in the sequence. So, the information transmitted through the network is the estimation result or partial decision. Finally, a distributed or cluster based source number estimation scheme is a structure including two levels of processing: source number estimation within each cluster and decision fusion between different clusters. The authors conclude that the cluster-based distributed approach using the progressive intra-cluster estimation has the best performance in the sense that it can provide much higher detection probability than the centralized approaches, while at the same time occupying the least amount of network bandwidth and consuming the least amount of energy. [13] Introduces Markov chain Monte Carlo data association algorithm to track an unknown number of targets. The algorithm once again relies on the existence of a clustering algorithm.

The clustering algorithm presented in this paper has the ideal features pointed out in [12]. The algorithm is distributed in nature and allows for intra cluster data aggregation. The intra cluster data aggregation is made possible by the overlapping nature of the clusters. This also leads to redundancy and increases the success rate of target detection.

III. PROBLEM DEFINITION

Data fusion algorithms rely on clustering algorithms so that the raw data collected by individual sensors can be combined in an efficient way. One such data fusion algorithm useful for target tracking application is based on closest point of approach or CPA algorithm. The CPA algorithm [1] estimates the target motion parameters based on CPA measurements made by at least four sensors. The target motion parameters being speed, direction, bearing of the target's path and the precise location of the target at a certain instance of time.

The purpose of clustering algorithm is to fuse the CPA measurements made by individual sensors at minimum cost (in terms of energy) and at the same time provide best coverage possible for a given deployment. One way to achieve data fusion is to deliver the CPA measurements to a centralized location. However transmission of raw data to the centralized location would mean transmitting 4 raw data packets instead of one fused value. This is not economical in terms of energy. Instead, clustering allows for local data fusion. Since each clusterhead is only one hop away from its member nodes less energy is consumed in transmission of raw data and also the probability of losing raw data is reduced.

A cluster once formed can track a target only with certain accuracy. The primary sources of error are false alarms at each sensor, loss of raw data and sensor failures. Hence it is necessary to provide redundancy. Redundancy can be achieved by allowing more than one cluster to track the same region. In the proposed algorithm we allow for redundancy by allowing two clusters to share a predetermined number of nodes. By increasing the number of shared nodes more clusters are formed in the same region and hence increases the system reliability. However Figure 5 shows that failure rate can be considerably reduced by increasing the number of clusters tracking a region from 1 to 3. However this increased reliability comes at an extra cost in terms of energy. In the proposed algorithm reliability can be controlled by adjusting the number of nodes two clusters can share. This also leads to a series of overlapping clusters and allows for data fusion between inter clusters.

A. Closest point of approach

Each sensor monitors the acoustic signal from the target with the help of a microphone, i.e. it monitors the signal energy for a given time window. The sensor confirms the presence of a target (called event) once the signal strength exceeds a certain threshold. The threshold is dynamically updated based on background noise statistics to reduce false alarm rate. Once a node detects an event (i.e. the presence of a moving vehicle), it stores a time series segment corresponding to the event. Figure 1 shows the time series segment corresponding to the interval in which the energy first exceeds the threshold (start of event) and eventually drops below the threshold (end of event) after reaching a peak value. The time at which the acoustic signal peaks is called the closest point of approach (CPA).

The CPA measurements made by each of the four sensor are reported to a centralized location (which is just any of the

four sensors) where the individual measurements are fused together by the CPA algorithm to determine the target motion parameters, such as precise location at a certain instant of time, velocity and orientation. Once these parameters have been calculated, the camera attached to the sensor can be programmed to record a video of the target and the recorded video can be sent to an external actor by using a data centric routing protocol, such as Directed Diffusion. However, the algorithm has pitfalls. There are certain configurations in which the algorithm fails to estimate the motion parameters. Figure 2 shows all the possible target trajectories with respect to the way sensors are deployed.

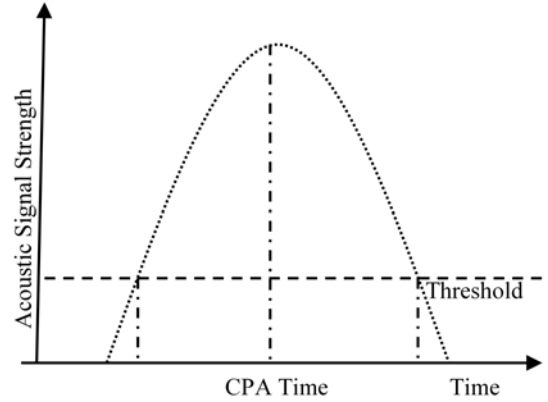


Fig. 1. Event detection by thresholding the energy of the acoustic signal detected by the microphone. The horizontal line represents the threshold. The maximum reading corresponds to CPA time.

The CPA algorithm can estimate the target parameters only when there is uneven number of sensors deployed on either side of the target's trajectory. At the minimum, to detect the target motion parameters we need CPA measurements from four sensors with 3 on one side and one on the other side of the target trajectory. Also the algorithm requires the three sensors that are on one side of the trajectory to be non collinear. In all other cases the solution is ambiguous [1].



Fig. 2. Classification of target trajectories according to the way of sensor field decomposition.

The clustering algorithm described in this paper avoids the issue of even deployment of sensors by grouping five sensors together into one cluster. One of the five sensors is chosen as the cluster head. All the member nodes send their raw data to the cluster head, which then makes use of the CPA algorithm to estimate the target motion parameters. Figure 3 shows a cluster formed between five nodes and the dotted lines

represent the possible paths a target can take. Each cluster will have a cluster head to which all the other sensors can send in their results. The cluster head then runs the CPA algorithm to identify the trajectory's parameters. The cluster head needs CPA measurements from only 4 sensors, but does not know which 4 measurements will lead to a solution and hence tries out combinations. It then ignores the invalid combinations and averages the valid solutions.

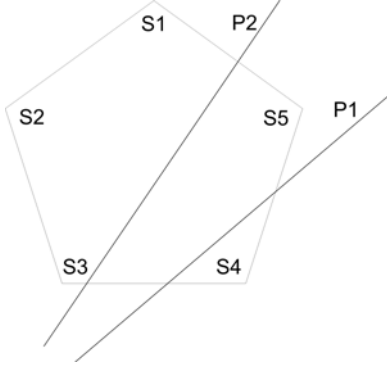


Fig. 3. A cluster formed with five sensors.

Each cluster has a certain failure rate in detecting the target. The failures arise because of the inefficiency of the individual sensors in detecting CPA time. The errors could be because of the ambient noise or because of the failure to identify the right threshold. Further, in some cases the raw data packets sent by some of the sensors to the cluster head might be lost. Hence it becomes necessary to have redundant clusters so that the tracking efficiency can be improved.

Assuming that each sensor fails to detect the event with a probability p , then the probability of failure for a cluster to track an event can be obtained by evaluating the following two cases.

Case 1: Assume that the target takes the path P1 i.e. 1 sensor on one side of the target trajectory and all the other sensors on the other side. Let P1 represent the probability of failure.

$$P = 1 - \text{probability of success} = 1 - ({}^4C_3 * (1-p)^4 * p + (1-p)^5) \quad (1)$$

Case 2: Assume that the target takes the path P2 i.e. 2 sensors on one side of the trajectory and the rest on the other side. Let P2 represent the probability of failure.

$$P = 1 - \text{probability of success} = 1 - ({}^2C_1 * (1-p)^4 * p + (1-p)^5) \quad (2)$$

Since there are only 5 possible scenarios in which Case 1 can happen and 10 possible scenarios in Case 2 can happen, the probability with which a cluster can fail to detect an event is $0.33*(1) + 0.67*(2)$.

Figure 4 represents the individual failure rates of each case and the overall failure rate with changing failure rates of individual sensors. A sensor fails to detect because of various factors, such as ambient noise and issues associated with the fine tuning of thresholds for the acoustic detector. The total

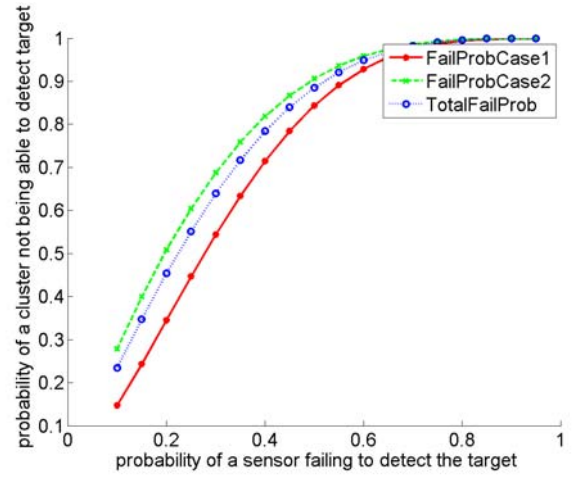


Fig. 4. Failure probability of a cluster in detecting the target given individual sensor failure probability.

success rate can be improved by having more than one cluster monitor a certain region. This makes it necessary to have overlapping clusters. If we have six sensors, then we can have six clusters such that any two clusters differ by at least one sensor. Incorporating new nodes and forming new clusters with different set of sensors can give a different perspective to evaluate the tracking parameters. Figure 5 shows how the total failure to track a target decreases as more clusters track the target. It can be seen that the failure probability can be reduced greatly by increasing the number of clusters tracking a target from 1 to 3. Any further increase in the number of clusters does lead to a significant improvement.

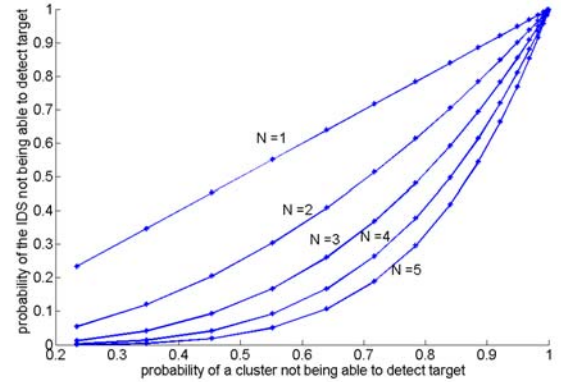


Fig. 5. Improvement in the target detection as more number of clusters monitors a region.

In the algorithm presented in the next section, the amount of redundancy can be controlled by controlling the number of sensors two clusters can have in common.

B. Protocol Overview

The clustering algorithm forms as many polygons as possible and prevents polygons with exactly the same set of sensors

state. Assume that each of the nodes have exchanged the FORM_POLY_INFORM messages. By the nature of the node positions, node 9 is aware of the cluster being formed by 10; 10 is aware of the clusters being formed by both 10 and 11; and 11 is aware of the cluster being formed by 10. Now there is a tie between nodes 9, 10 and 11. Since the algorithm allows for nodes with lower ID to form clusters, only node 9 would be able to form a polygon. This because 10 would loose the race to 9 and 11 would loose the race to 10. We could better monitor the field if 9 and 11 can form clusters as their respective set of member nodes differ by 2 nodes. Hence, in such situations, 10 allows 11 to form a cluster by sending a STATUS_ACCEPTED message.

Before the STATUS_ACCEPTED message reaches node 11, assume that node 11 looses to node 8 (since 8 has lower ID, it has precedence over 11 in case of, tie.). In this case, we would end up with just two clusters formed by nodes 9 and 8. This might lead to a region between nodes 9 and 8 not being monitored. In order to avoid such situations, node 11 sends out a STATUS_ACCEPTED message to node 10, and node 10 goes ahead to form a cluster. Node 11 will then move to STATUS_ACCEPTED_SENT state.

- **ACK_STATUS_ACCEPTED:** While in STATUS_ACCEPTED_SENT state, if a node receives a STATUS_ACCEPTED message from a node with a lower ID, it replies with a STATUS_ACCEPTED message and waits for (ack_timer period) the lower ID to acknowledge the message with ACK_STATUS_ACCEPTED message. On the other hand, if it receives an STATUS_ACCEPTED message from a higher node ID, it moves to CHEAD state and the lower ID acknowledges with a ACK_STATUS_ACCEPTED message. This final step of the algorithm assures that nodes with a lower ID gets to form a polygon even in the presence of wireless losses, and, at the same time, assures that no void regions (regions that are not monitored by any cluster) are formed.

IV. PERFORMANCE EVALUATION

The clustering algorithm was implemented in ns-2.27 [14]. The total number of protocol packets generated in order to maintain the clusters for a period of 100 seconds was measured. Four different deployment topologies were evaluated. The first was grid deployment in which 100 nodes were laid in 10 * 10 matrix and separated by 40m. The second topology was a random deployment consisting of 150 sensors uniformly deployed over a 670*670 m field. The next two sensor deployments were based on pentagonal tiling.

Pentagonal tilings were analyzed since it is possible to have non-overlapping clusters of five sensors and hence assure that each part of the field can be monitored by at least one cluster. In an ideal case where each cluster can monitor the region that it encloses with a probability of 1. By having one of the five nodes of each pentagon to be a cluster head and the other

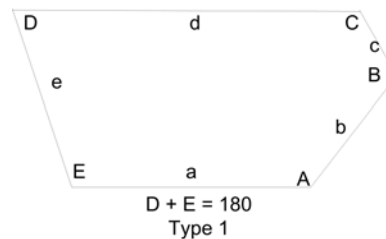


Fig. 8. Pentagonal tessellations.

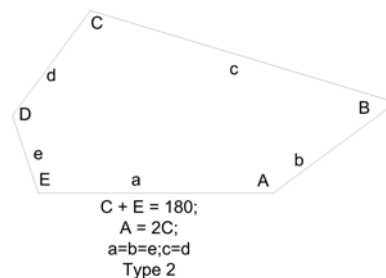


Fig. 9. Pentagonal tessellations.

nodes as its member nodes, the entire field can be monitored with a minimum number of clusters. In all, 14 different types of convex pentagons [15] can tile a plane but we evaluated only two of them. Figure 8 and 9 shows the geometrical properties of the two.

In tessellation of Type I, 280 sensors were deployed in a rectangular field of 500 * 1000m. In case of Type II, a total of 300 nodes were deployed in a rectangular field of 500*1000m.

A node that intends to form a cluster can adopt different strategies in identifying its member nodes. Two different strategies were evaluated. In the first strategy the node intending to form a cluster chooses the nodes that are closest to it, while in the second version the algorithm chose the farthest nodes to form a cluster. By choosing the farthest nodes the region tracked by each cluster is maximized, but increases the protocol overhead to maintain them as they tend to be less stable. On the other hand by choosing closest nodes as member nodes leads to more stable clusters but reduces the area monitored by each cluster. In each scenario the algorithm was analyzed by allowing the cluster heads to form clusters with varying number of identical nodes (the same nodes appear in two different clusters). The number of identical nodes was chosen to be 2, 3 and 4.

Choosing the farthest nodes increases the area tracked by a single cluster but also increases the protocol overhead when compared to the minimum coverage approach. This is because the member nodes of a cluster and the cluster head use HELLO messages (for the simulations the hello timer was set to 4 sec) to verify that all the members of the cluster are alive. A node that receives a HELLO message replies with a HELLO_REPLY message, both of which are broadcast messages. All the member nodes that receive the broadcast

message update their hello timer for the node that just sent the HELLO_REPLY. However, when the cluster head chooses the farthest nodes as its member nodes there is a higher probability that not all the member nodes are within one another's communication range and this leads to increased hello messages.

The policy of allowing clusters to have more and more shared nodes leads to an increase in the number of clusters and increase in redundancy. As the clusters are allowed to have more shared nodes, it can be seen that the protocol overhead is reduced. This is because when $N=2$ (N being the number of identical nodes), two clusters that are in close vicinity have a greater chance of having more than two nodes in shared (i.e. identical). Hence the sensors spend more time in resolving ties and this leads to increased protocol overhead. For the same reason the clusters face lesser conflicts with neighbours when $N=3$ and $N=4$. In case of the maximum coverage approach, the overhead is more or less uniform for $N=2, 3, 4$ as each cluster is formed with farthest nodes there is a lesser chance of conflict. This leads to an increase in the number of clusters formed. Also, more clusters would mean more HELLO packets to make sure that the member nodes are alive. Furthermore, in the maximum coverage approach since the member nodes of a cluster are geographically more distant from each other; the ability of a cluster to track a target is reduced.

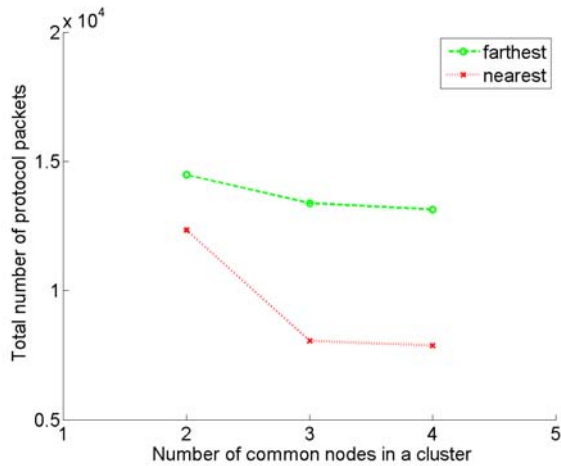


Fig. 10. Protocol overhead in Grid deployment.

From Figures 10, 11, and 12 it can be seen that both the number of identical nodes two different clusters is allowed to have and the approach taken to choose the member nodes while forming a new cluster affect the protocol overhead.

The probability of conflicts between clusters is also dependant on the deployment strategy. In random and tessellation based deployment strategy it can be seen that for the case $N=2$ the minimum coverage approach spends even more time in resolving conflicts whereas the protocol overhead for the maximum coverage approach is constant for all the deployment patterns. From Table I, it can be seen that more clusters

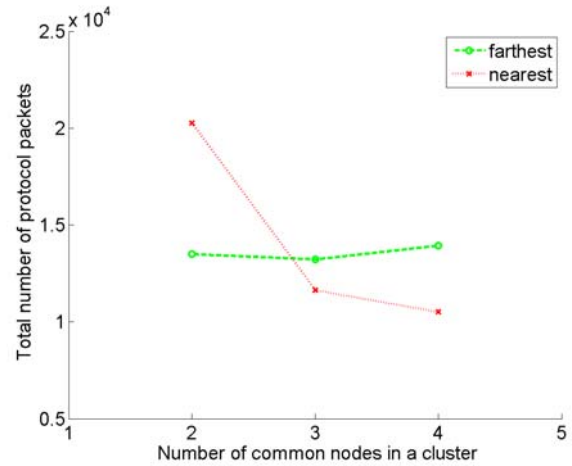


Fig. 11. Protocol overhead in Random deployment.

TABLE I
COMPARES THE TOTAL NUMBER OF CLUSTERS FORMED IN CLOSEST APPROACH AND THE FARTEST APPROACH. READ AS (MINIMUM COVERAGE VS MAXIMUM COVERAGE)

	N = 2	N = 3	N = 4
Grid	50 Vs 83	72 Vs 93	92 Vs 99
Random	48 Vs 98	77 Vs 135	112 Vs 139
Type I Tessellation	90 Vs 202	192 Vs 272	260 Vs 280
Type II Tessellation	74 Vs 238	135 Vs 286	155 Vs 288

can be formed by choosing the maximum coverage approach. However this leads to increase in number of cluster heads and hence increase in the number HELLO message exchanged to make sure that the cluster is active.

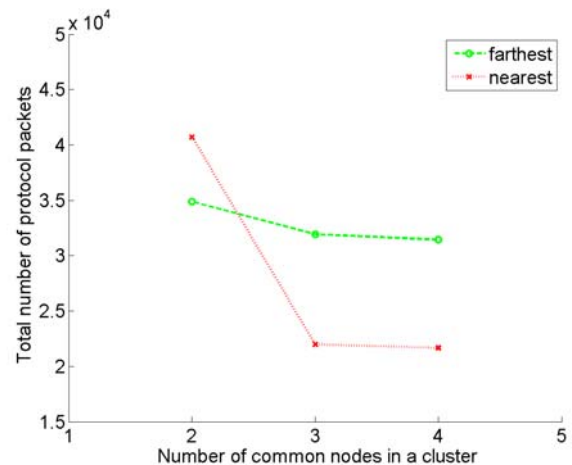


Fig. 12. Protocol overhead in case of sensor deployment using Type 1 pentagonal tessellation.

Whenever a node dies the cluster heads of all the clusters that the dead node belongs to are dissolved and a race for

forming new clusters begins. Also since a node, can be a cluster head and at the same time be a member node of another cluster, the dissolution of one cluster could lead to a ripple effect. A larger ripple means a larger number of clusters have to be reorganized and hence more protocol overhead. The extent of the ripple is dependent on the geographical location of the node and its relationship to other nodes.

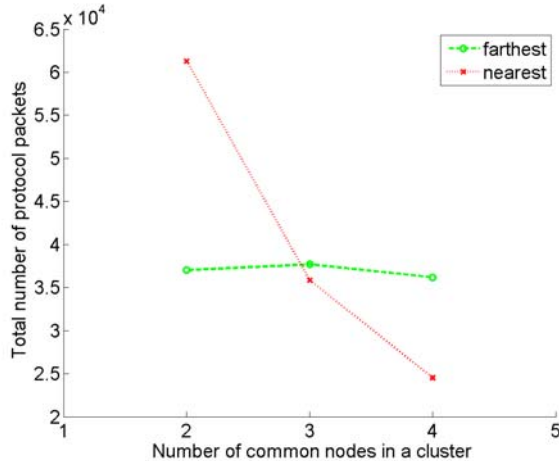


Fig. 13. Protocol overhead in case of sensor deployment using Type 2 pentagonal tessellation.

So in each of the deployment scenarios we kill a node that is in the center of the field. In Figures 14-17 it can be seen that the ripple is contained to a smaller region, and hence, the increase in protocol overhead is small. In the proposed clustering algorithm, a cluster head once in CHEAD state continues to remain so until one of its member node dies. This limits the spread of ripple once a node dies. We analyze the protocol overhead when the clusters are formed using minimum coverage approach. The additional protocol packets generated to re-organize the clusters in the event of node failure is maximum for grid deployment and insignificant in all other deployment strategies.

Table 1 shows that by adopting the maximum coverage approach (each cluster head forms a cluster by choosing geographically farthest nodes from its one hop neighbors) in the clustering algorithm we can form as many as twice the number of clusters with closest approach (each cluster head forms a cluster by choosing geographically closest nodes from its one hop neighbors) even when $N=2$. This justifies the large protocol overhead observed in case of maximum coverage approach.

V. CONCLUSION

In this paper we propose a distributed clustering algorithm to track intruders. The localized nature of the algorithm ensures that reconfiguration does not significantly increase the protocol overhead. The algorithm builds a series of overlapping clusters which allow for more than one cluster to track a region. This redundancy improves the overall system

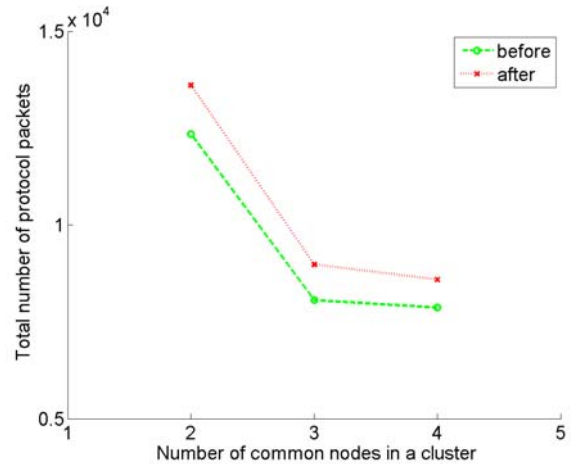


Fig. 14. Protocol overhead in case of Node failure in grid deployment (minimum coverage approach).

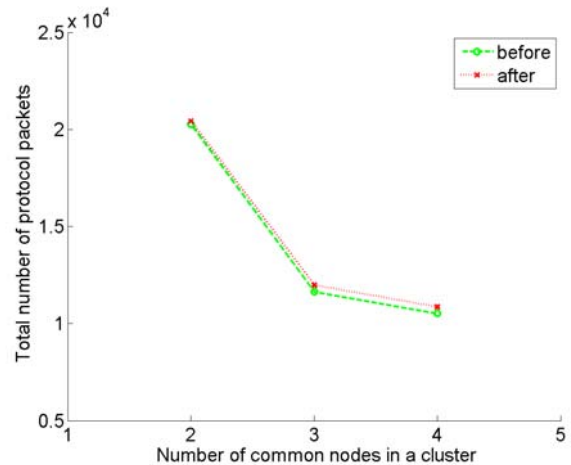


Fig. 15. Protocol overhead in case of Node failure in Random deployment (minimum coverage approach).

reliability. The overlapping clusters also allow for tracking of curvilinear targets. We analyzed the factors that affect the protocol overhead. The protocol overhead depends on the sensor deployment strategy, number of clusters and the approach adopted by the cluster head in choosing its member nodes. Maximum coverage approach forms larger numbers of clusters than the minimum coverage approach and also leads to proportionate increase in the protocol overhead. Node failures lead to re-organization of clusters and hence increase protocol overhead. In the worst case there is a 11% increase in protocol overhead to reconfigure the clusters. This happens when the sensors are deployed in a grid. In the future we would like to analyze the efficiency of clustering algorithms by measuring the average probability of tracking.

REFERENCES

- [1] F. Dommermuth, "The estimation of target motion parameters from cpa time measurements in a field of acoustic sensors." in *J. Acoust. Soc.*

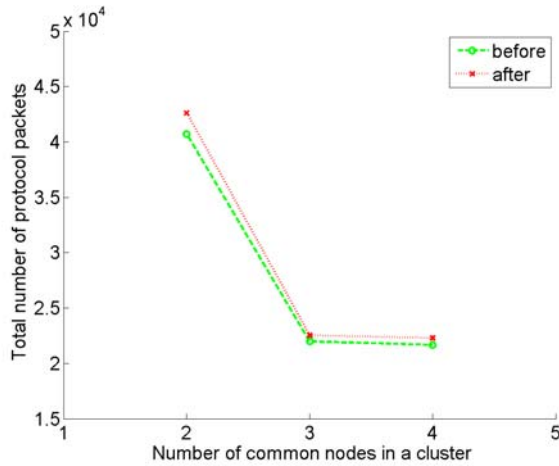


Fig. 16. Protocol overhead in case of Node failure when sensors are deployed using Type 1 pentagonal tessellation (minimum coverage approach).

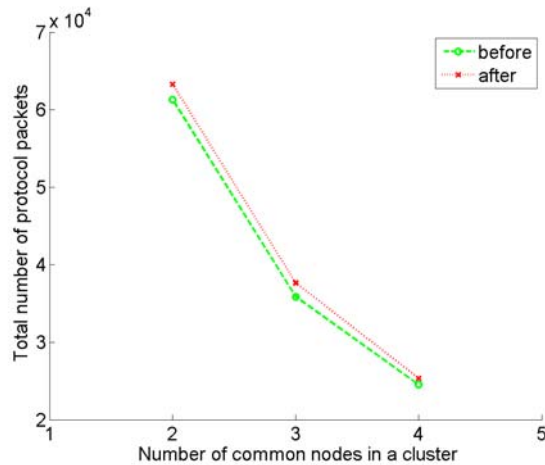


Fig. 17. Protocol overhead in case of Node failure when sensors are deployed using Type 2 pentagonal tessellation (minimum coverage approach).

[9] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda, Y. Choi, T. Herman, S. Kulkarni, U. Arumugam, M. Nesterenko, A. Vora, and M. Miyashita, "A line in the sand: A wireless sensor network for target detection, classification, and tracking." vol. 46, no. 5. *Computer Networks*, 2004, pp. 605–634.

[10] H. Tian, A. Pascal, Y. Ting, L. Liqian, Z. Gang, S. Radu, C. Qing, A. Stankovic, and T. Abdelzaher, "Achieving real-time target tracking using wireless sensor networks." in *Embedded Computing System, 2007. ACM Transactions on*, 2007.

[11] W. Qixin, C. Wei-Peng, Z. Rong, L. Kihwal, and S. Lui, "Acoustic target tracking using tiny wireless sensor devices." in *Information Processing in Sensor Networks, 2003. Proceedings. 2nd International Workshop on*, 2003.

[12] W. Xiaoling and Q. Hairong, "Mobile agent based progressive multiple target detection in sensor networks." in *Acoustics, Speech, and Signal Processing, 2004. Proceedings. IEEE International Conference on*, 2004.

[13] S. Oh, P. Chen, M. Manzo, and S. Sastry, "Instrumenting wireless sensor networks for real-time surveillance." in *Robotics and Automation, 2006. Proceedings. the International Conference on*, 2006.

[14] "Network simulator 2 (ns2)," University of California at Berkeley, 1997. [Online]. Available: <http://www.isi.edu/nsnam/ns/>

[15] B. Grunbaum and G. Sheperd, *Tilings and Patterns*. New York, U.S.A: W.H Freeman And company, 1986.

Am., vol. 83, no. 5, 1988, pp. 1476–1480.

[2] X. Xu and S. Sahni, "Approximation algorithms for sensor deployment," in *Innovations and Real-Time Applications of Distributed Sensor Network, 2006. Proceedings. 2nd International Symposium on*, 2006.

[3] Z. Abrams, A. Goel, and S. Plotkin, "Set k-cover algorithms for energy efficient monitoring in wireless sensor networks," in *IEEE IPSN*, 2004.

[4] T. Clouqueur, V. Phipatanasuphorn, S. K., and P. Ramanathan, "Sensor deployment strategy for detection of targets traversing a region," in *Mobile Networks and Applications*, vol. 28, no. 8, 2003, pp. 453–461.

[5] S. Kumar, T. Lai, and B. J., "On k-coverage in a mostly sleeping sensor network," in *Mobile Computing and Networking, 2004. Proceedings. Tenth Annual International Conference on. ACM*, 2004.

[6] G. Veltri, Q. Huang, G. Qu, and M. Potkonjak, "Minimal and maximal exposure path algorithms for wireless embedded sensor networks," in *SensSys*, 2003.

[7] S. Megerian, F. Koushanfar, M. Potkonjak, and M. Srivastava, "Worst- and best-case coverage in sensor networks." in *Mobile Computing, 2004. IEEE Transactions on*, vol. 3, no. 4. IEEE, 2004.

[8] J. Adriaens, S. Megerian, and M. Potkonjak, "Optimal worst-case coverage of directional field-of-view sensor networks." in *Sensor, Mesh and Ad Hoc Communications and Networks, 2006. The third annual IEEE Communications Society Conference on*. IEEE, 2006, pp. 336–345.