TARGET LOCALIZATION IN WIRELESS SENSOR NETWORKS USING ERROR CORRECTING CODES IN THE PRESENCE OF BYZANTINES

Aditya Vempaty[⋆] *Yunghsiang S. Han*[†] *Pramod K. Varshney*[⋆]

[⋆] Department of EECS, Syracuse University, NY, 13244 USA
[†]Department of EE, National Taiwan University of Science and Technology, Taiwan, ROC
email:{avempaty,varshney}@syr.edu, yshan@mail.ntust.edu.tw

ABSTRACT

We consider the problem of target localization using quantized data in Wireless Sensor Networks in the presence of Byzantines (malicious sensors). Since the effect of Byzantines can be treated as errors in the transmitted data, we propose the use of error correcting codes for the task of target localization. We design coding based iterative schemes for target localization where, at every iteration, the Fusion Center performs an M-ary hypothesis test and decides the Region of Interest for the next iteration. Simulation results show that our proposed schemes provide a better performance as compared to the traditional Maximum Likelihood Estimation and are also computationally much more efficient.

Index Terms— Target Localization, Byzantines, Error Correcting Codes, Wireless Sensor Networks

1. INTRODUCTION

Wireless sensor networks (WSNs) have been extensively employed to monitor a region of interest (ROI) for reliable detection/estimation/tracking of events [1–4]. In this paper, we focus on target localization in WSNs using quantized data, where, due to power and bandwidth constraints, each sensor sends quantized data to a Fusion Center (FC). The FC combines these local sensors' data to estimate the target location. We consider target localization in the presence of Byzantine attacks [5–8] (also referred to as Data Falsification Attacks). Byzantine attacks are internal security attacks which involve malicious sensors within the network which send false information to the FC to disrupt the global inference process. In our previous work [8], we have considered target localization in WSNs in the presence of Byzantines and showed that the FC becomes 'blind' to the local sensor's data when the fraction of Byzantines is greater than 50%. We also proposed mitigation techniques for the network to make the Byzantines 'ineffective' in their attack strategy.

Wang et al. in [9] proposed a coding based distributed classification fusion approach which is tolerant to faults in the network. Since Byzantines can be treated as faults in the network, motivated by [9], in this paper, we propose a novel coding based target localization approach to tolerate Byzantines in the network. The proposed schemes provide a coarse estimate in a much more computationally efficient manner as compared to the Maximum Likelihood Estimation (MLE) and also has a higher Byzantine fault tolerance capability.

This work was supported in part by AFOSR under Grant FA9550-10-1-0263, National Science Council of Taiwan under Grants NSC 99-2221-E-011-158 -MY3 and NSC 101-2221-E-011-069 -MY3.

2. SYSTEM MODEL

Let N sensors be deployed in a WSN as shown in Fig. 1 to estimate the location of a target at $\theta = [x_t, y_t]$ where x_t and y_t denote the coordinates of the target in a 2-D Cartesian plane. We assume that the signal radiated from this target follows an isotropic power attenuation model [2]. The signal amplitude a_i received at the i^{th} sensor is given by

$$a_i^2 = P_0 \left(\frac{d_0}{d_i}\right)^n \tag{1}$$

where P_0 is the power measured at the reference distance d_0, $d_i \neq 0$ is the distance between the target and the i^{th} sensor whose location is represented by $L_i = [x_i, y_i]$ for $i = 1, 2 \cdots, N$ and n is the path loss exponent. In this paper, without loss of generality, we assume $d_0 = 1$ and $n = 2$. The signal amplitude measured at each sensor is corrupted by additive white Gaussian noise (AWGN):

$$s_i = a_i + n_i \tag{2}$$

where s_i is the corrupted signal at the i^{th} sensor and the noise n_i follows $\mathcal{N}(0, \sigma^2)$.

Due to energy and bandwidth constraints, the local sensors quantize their observations using threshold quantizers and send binary quantized data to the FC:

$$D_i = \begin{cases} 0 & s_i < \eta_i \\ 1 & s_i > \eta_i \end{cases} \tag{3}$$

where D_i is the quantized data at the i^{th} sensor and η_i is the threshold used by the i^{th} sensor for quantization. The FC fuses the data received from the local sensors and estimates the target location. Traditional target localization uses MLE [2]:

$$\hat{\theta} = \arg\max_{\theta} p(\mathbf{D}|\theta) \tag{4}$$

where $\mathbf{D} = [D_1, D_2, \cdots, D_N]$ is the vector of quantized observations received at the FC.

We assume the presence of $B = \alpha N$ number of Byzantines in the network. Byzantines are local sensors which send false information to the FC to deteriorate the network's performance. In this paper, we assume that the Byzantines attack the network independently [8] where the Byzantines flip their data with probability '1' before sending it to the FC.[1] In other words, the data sent by the i^{th}

[1]It has been shown in [8] that the optimal independent attack strategy for the Byzantines is to flip their data with probability '1'.

sensor is given by:

$$u_i = \begin{cases} D_i & \text{if } i^{th} \text{ sensor is honest} \\ \bar{D}_i & \text{if } i^{th} \text{ sensor is Byzantine} \end{cases} \quad (5)$$

The channels between the local sensors and the FC are assumed to be ideal and the FC estimates the target location using the received data $\mathbf{u} = [u_1, u_2, \cdots, u_N]$. For such a system, it has been shown in [8] that the FC becomes 'blind' to the network's information for $\alpha \geq 0.5$. Therefore, for the remainder of the paper, we analyze the system when $\alpha < 0.5$.

3. TARGET LOCALIZATION USING ERROR CORRECTING CODES

In this section, we propose Localization using Error Correcting Codes which is tolerant to Byzantine attacks. Our algorithm is iterative in which at every iteration, the ROI is split into M regions and an M-ary hypothesis test is performed at the FC. The FC, through feedback, declares this region as the ROI for the next iteration. The M-ary hypothesis test solves a classification problem where each sensor sends a binary quantized value based on a code matrix C. The code matrix is of size $M \times N$ with elements $c_{ji} \in \{0, 1\}$, $j = 0, 1, \cdots, M-1$ and $i = 1, \cdots, N$ where each row represents a possible hypothesis and each column i represents i^{th} sensor's binary decision rule. After receiving the binary decisions $\mathbf{u}$ from local sensors, the FC performs minimum Hamming distance based fusion and decides on the hypothesis H_j for which the Hamming distance between row of C corresponding to H_j for $j = 0, \cdots, M-1$ and the received vector $\mathbf{u}$ is minimized. In this way, the search space for target location is reduced at every iteration and we stop the search after a pre-determined stopping criterion. We also limit ourselves to a regular grid network and, therefore, the optimal splitting of the ROI at every iteration is to split it into equal sized regions as shown in Fig. 1.

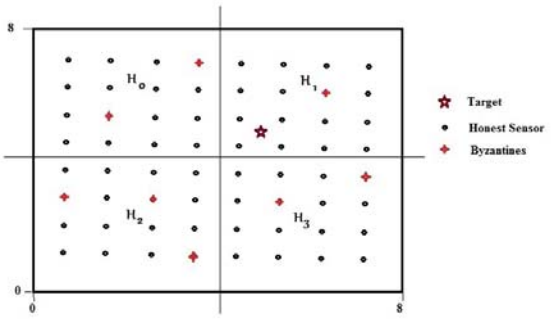

Fig. 1. Equal region splitting of the ROI for the M-hypothesis test

3.1. Basic Coding Based Scheme

In this subsection, we present the basic coding based scheme for target localization. Since there are N sensors in a regular grid as shown in Fig. 1, the number of sensors in the new ROI after every iteration is reduced by a factor of M. After k iterations, the number of sensors in the ROI are $\frac{N}{M^k}$ and, therefore, the code matrix at the $(k+1)^{th}$ iteration would be of size $M \times \frac{N}{M^k}$.[2] Since the code matrix should always have more columns than rows, $k^{stop} < \log_M N$ where k^{stop} is the number of iterations after which the scheme terminates. After k^{stop} iterations, there are only $\frac{N}{M^{k^{stop}}}$ sensors present in the ROI and a coarse estimate $\hat{\theta} = [\hat{\theta}_x, \hat{\theta}_y]$ of the target's location can be obtained by taking an average of locations of the $\frac{N}{M^{k^{stop}}}$ sensors present in the ROI:

$$\hat{\theta}_x = \frac{M^{k^{stop}}}{N} \sum_{i \in ROI_{k^{stop}}} x_i \quad \text{and} \quad \hat{\theta}_y = \frac{M^{k^{stop}}}{N} \sum_{i \in ROI_{k^{stop}}} y_i \quad (6)$$

where $ROI_{k^{stop}}$ is the ROI at the last step.

Since the scheme is iterative, the code matrix needs to be designed at every iteration. Observing the structure of our problem, we can design the code matrix in a simple and efficient way as described below. As pointed out before, the size of the code matrix C^k at the $(k+1)^{th}$ iteration is $M \times \frac{N}{M^k}$. Each row of this code matrix C^k represents a possible hypothesis described by a region in the ROI. Let R_j^k denote the region represented by the hypothesis H_j for $j = 0, 1, \cdots, M-1$ and let S_j^k represent the set of sensors that lie in the region R_j^k. Also, for every sensor i, there is a unique corresponding region in which the sensor lies which is represented as $r^k(i)$. It is easy to see that $S_j^k = \{i \in ROI_k | r^k(i) = j\}$. The code matrix is designed in such a way that for the j^{th} row, only those sensors that are in R_j^k have '1' as their element in the code matrix. In other words, the elements of the code matrix are given by

$$c^k_{(j+1)i} = \begin{cases} 1 & \text{if } i \in \mathcal{S}_j^k \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

for $j = 0, 1, \cdots, M-1$ and $i \in ROI_k$.

The above construction can also be viewed as each sensor i using a threshold η_i^k for quantization (as described in (3)). Let each region R_j^k correspond to a location θ_j^k for $j = 0, 1, \cdots, M-1$, which in our case is the center of the region R_j^k. Each sensor i decides on a '1' if and only if the target lies in the region $R^k_{r^k(i)}$. Every sensor i, therefore, performs a binary hypothesis test described as follows:

$$\begin{aligned} H_1 : &\quad \theta \in R^k_{r(i)} \\ H_0 : &\quad \theta \notin R^k_{r(i)} \end{aligned} \quad (8)$$

If $d_{i,j}$ represents the Euclidean distance between the i^{th} sensor and θ_j^k for $i = 1, 2, \cdots, N$ and $j = 0, 1, \cdots, M-1$, then $r^k(i) = \arg\min_l d_{i,l}$. Therefore, the condition $\theta^k \in R^k_{r^k(i)}$ implies a threshold η_i^k given by

$$\eta_i^k = \frac{\sqrt{P_0}}{d_{i,r^k(i)}} \quad (9)$$

This ensures that if the signal amplitude at the i^{th} sensor is above the threshold η_i^k, then θ^k lies in region $R^k_{r^k(i)}$ leading to minimum distance decoding.

3.2. Exclusion Method for Decoding with Weighted Average

Although the scheme proposed in Section 3.1 has a very good Byzantine fault tolerance capability as shown later in Section 5, the performance can be improved by using an exclusion method for decoding where the best two regions are kept for next iteration and using weighted average to estimate the target location at the final

[2] We assume that N is divisible by M^k for $k = 0, 1, \ldots, \log_M N - 1$.

step. This scheme builds on the basic coding scheme proposed in Section 3.1 with the following improvements:

- Since after every iteration two regions are kept, the code matrix after the k^{th} iteration is of size $M \times \frac{2^k N}{M^k}$ and the number of iterations needed to stop the localization task needs to satisfy $k^{stop} < \log_{M/2} N$.
- At the final step, instead of taking an average of the sensor locations of the sensors present in the ROI at the final step, we take a weighted average of the sensor locations where the weights are the 1-bit decisions sent by these sensors. Since, a decision $u_i = 1$ would imply that the target is closer to the sensor i, a weighted average ensures that the average is taken only over the sensors for which the target is reported to be close.

Therefore, the target location estimate is given by

$$\hat{\theta}_x = \frac{\sum_{i \in ROI_{k^{stop}}} u_i x_i}{\sum_{i \in ROI_{k^{stop}}} u_i} \quad \text{and} \quad \hat{\theta}_y = \frac{\sum_{i \in ROI_{k^{stop}}} u_i y_i}{\sum_{i \in ROI_{k^{stop}}} u_i} \tag{10}$$

The exclusion method results in a better performance compared to the basic coding scheme since it keeps the best two regions after every iteration. This observation is also evident in the numerical results presented in Section 5.

4. PERFORMANCE ANALYSIS

4.1. Byzantine Fault Tolerance Capability

For the basic coding scheme described in Section 3.1, each column in C^k contains only one '1' and every row of C^k contains exactly $\frac{N}{M^{k+1}}$ '1's. Therefore, the minimum Hamming distance of C^k is $\frac{2N}{M^{k+1}}$ and, at the k^{th} iteration, it can tolerate a total of at most $\frac{N}{M^{k+1}} - 1$ faults due to the presence of Byzantines in the network.

However, when the exclusion method based scheme described in Section 3.2 is used, since the two best regions are considered after every iteration, the fault tolerance performance improves and we can tolerate a total of at most $2\frac{2^k N}{M^{k+1}} - 1$ faults. This improvement in the fault tolerance capability can be observed in the simulation results presented in Section 5.

4.2. Probability of Detection of Target Region

Another metric to analyze the performance of the proposed scheme is the probability of detection of the target region. It is an important metric when the final goal of the target localization task is to find the approximate region or neighborhood where the target lies rather than the true location itself. Since the final ROI could be one of the $N/M^{k^{stop}}$ regions, a metric of interest is the probability of 'zooming' into the correct region. In other words, it is the probability that the true location and the estimated location lie in the same region. In the remainder of this section, we derive the detection probability of target region.

The final region of the estimated target location is the same as the true target location, if and only if we 'zoom' into the correct region at every iteration of the proposed scheme. If P_d^k denotes the detection probability at the k^{th} iteration step, the overall detection probability is given by

$$P_D = P(\cap_k \text{correct detection at step k}) = \prod_k P_d^k \tag{11}$$

In this paper, we derive the detection probability for the basic coding scheme. The detection probability for the exclusion method can be found similarly and is omitted for the sake of brevity.

Let us consider the k^{th} iteration and define the received vector at the FC as $\mathbf{u}^k = [u_1^k, u_2^k, \cdots, u_{N_k}^k]$ where N_k are the number of local sensors reporting their data to FC at k^{th} iteration. Let $\mathcal{D}_j^k$ be the decision region of j^{th} hypothesis defined as follows:

$$\mathcal{D}_j^k = \{\mathbf{u}^k | d_H(\mathbf{u}^k, \mathbf{c}_j^k) \leq d_H(\mathbf{u}^k, \mathbf{c}_i^k) \text{ for } 1 \leq i \leq M\}$$

where $d_H(\cdot,\cdot)$ is the Hamming distance between two vectors, and $\mathbf{c}_i^k$ is the codeword corresponding to hypothesis i in code matrix C^k. Then define the reward $r_{\mathbf{u}^k}^{j,k}$ associated with the hypothesis j as

$$r_{\mathbf{u}^k}^{j,k} = \begin{cases} \frac{1}{q_{\mathbf{u}^k}} & \text{when } \mathbf{u}^k \in \mathcal{D}_j^k \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

where $q_{\mathbf{u}^k}$ is the number of decision regions to whom $\mathbf{u}^k$ belongs to. Note that $q_{\mathbf{u}^k}$ can be greater than one when there is a tie at the FC. Since the tie-breaking rule is to choose one of them randomly, the reward is given by (12). According to (12), the detection probability at the k^{th} iteration is given by

$$\begin{aligned} P_d^k &= \sum_{j=1}^{M} P(H_j^k) \sum_{\mathbf{u}^k \in \{0,1\}^{N_k}} P(\mathbf{u}^k | H_j^k) r_{\mathbf{u}^k}^{j,k} \\ &= \frac{1}{M} \sum_{j=1}^{M} \sum_{u^k \in \mathcal{D}_j^k} \left(\prod_{i=1}^{N_k} P(u_i^k | H_j^k) \right) \frac{1}{q_{\mathbf{u}^k}} \end{aligned} \tag{13}$$

where $P(u_i^k | H_j^k)$ denotes the probability that the sensor i sends the bit $u_i^k \in \{0,1\}$, $i = 1, 2, \cdots, N_k$, when the true target is in the region R_j^k corresponding to H_j^k at iteration k.

By assuming that the probability that a sensor is Byzantine is p_α, we get

$$P(u_i^k = 1 | H_j^k) = \textstyle\int_{\theta \in R_j^k} P(u_i^k = 1 | \theta) d\theta \tag{14}$$

$$= \textstyle\int_{\theta \in R_j^k} [(1 - p_\alpha) Q\left(\frac{\eta_i^k - a_i}{\sigma}\right) + p_\alpha \left(1 - Q\left(\frac{\eta_i^k - a_i}{\sigma}\right)\right)] d\theta \tag{15}$$

where η_i^k is the threshold used by the i^{th} sensor at k^{th} iteration, σ^2 is the noise variance, a_i is the amplitude received at the i^{th} sensor given by (1) when the target is at θ and $Q(x)$ is the complementary cumulative distribution function of standard Gaussian and is given by

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{(-t^2/2)} dt \tag{16}$$

Since (14) is complicated, it can be approximated using θ_j^k which is the center of the region R_j^k. Eq. (14) now simplifies to

$$P(u_i = 1 | H_j^k) \approx (1 - p_\alpha) Q\left(\frac{\eta_i^k - a_{ij}^k}{\sigma}\right) + p_\alpha \left(1 - Q\left(\frac{\eta_i^k - a_{ij}^k}{\sigma}\right)\right) \tag{17}$$

where a_{ij}^k is the signal amplitude received at the i^{th} sensor when the target is at θ_j^k.

Using (11), the probability of detection of target region can be found as the product of detection probabilities at every iteration k. As pointed out before, these expressions can be extended to the scheme where exclusion method is used for decoding.

5. SIMULATION RESULTS & DISCUSSION

In this section, we present the simulation results to evaluate the performance of the proposed schemes in the presence of Byzantine faults. We analyze the performance using two performance metrics: mean square error (MSE) of the estimated location and probability of detection (P_D) of the target region. We use a network of $N = 512$ sensors deployed in a regular 8 m $\times$8 m grid as shown in Fig. 1. Let α denote the fraction of Byzantines in the network that are randomly distributed over the network. The received signal amplitude at the local sensors is corrupted by AWGN noise with noise standard deviation $\sigma = 3$. The power at the reference distance is $P_0 = 200$. At every iteration k, the ROI is split into $M = 4$ equal regions as shown in Fig. 1. We stop the iterations for the basic coding scheme after $k^{stop} = 2$ iterations. The number of sensors in the ROI at the final step are, therefore, 32. In order to have a fair comparison, we stop the exclusion method after $k^{stop} = 4$ iterations, so that there are again 32 sensors in the ROI at the final step.

Fig. 2 shows the performance of the proposed schemes in terms of the MSE of the estimated target location when compared with the traditional maximum likelihood estimation described by (4). The MSE has been found by performing 1×10^3 Monte Carlo runs with the true target location randomly chosen in the 8 m$\times$8 m grid.

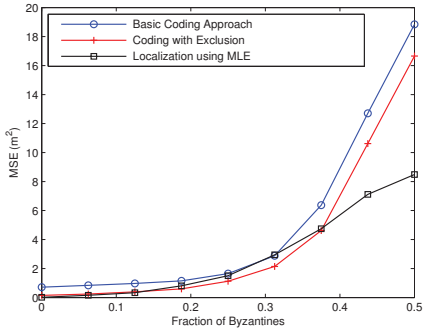

Fig. 2. MSE comparison of the three localization schemes

As can be seen from Fig. 2, the performance of the exclusion method based coding scheme is better than the basic coding scheme and outperforms the traditional MLE based scheme when $\alpha \leq 0.375$. When $\alpha > 0.375$ the traditional MLE based scheme has the best performance. Although the traditional MLE approach performs better than our basic coding approach, it is at a cost of higher computation. It is important to note that the proposed schemes provide a coarse estimate as against the traditional MLE based scheme which optimizes over the entire ROI. The traditional scheme is computationally much more expensive than the proposed coding based schemes. In the simulations performed, the proposed schemes are around 150 times faster than the conventional scheme when the global optimization toolbox in MATLAB was used for the conventional scheme. The computation time is very important in a scenario when the target is moving and a coarse location estimate is needed in a timely manner.

Fig. 3 shows the performance of the proposed schemes in terms of the detection probability of the target region. The detection probability has been found by performing 1×10^4 Monte Carlo runs with the true target randomly chosen in the ROI. Fig. 3 shows the reduction in the detection probability with increase in α when more sensors are Byzantines sending false information to the FC.

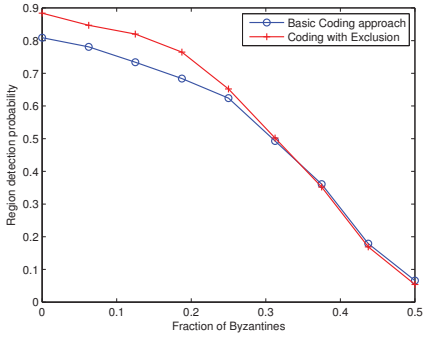

Fig. 3. Probability of detection of target region as a function of α

In order to analyze the effect of the number of sensors on the performance, we perform simulations by changing the number of sensors and keeping the number of iterations the same as before. Figs. 4 and 5 show the effect of number of sensors on MSE and detection probability of the target region respectively when the exclusion method based coding scheme is used. As can be seen from both the figures (Figs. 4 and 5), the fault-tolerance capability of the proposed scheme increases with increase in the number of sensors.

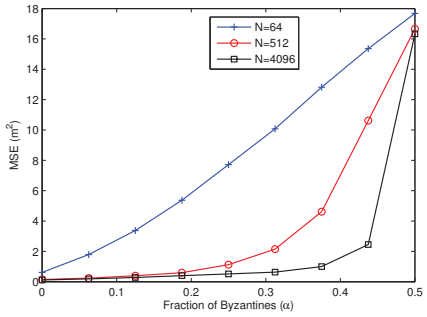

Fig. 4. MSE of the target location estimate with varying N

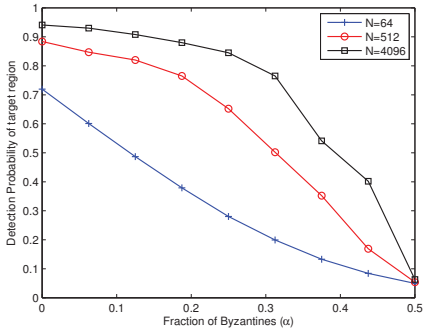

Fig. 5. Probability of detection of target region with varying N

In the future, we plan to extend this work to the case of target tracking when the target's location is changing with time and the sensor network's aim is to track the target's motion. The proposed schemes provide an insight on M-ary search trees and show that the idea of coding based schemes can also be used for other applications involving 'search' such as rumor source localization in social networks.

6. REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Commun. Mag.*, vol. 40, no. 8, pp. 102–114, Aug. 2002.

[2] R. Niu and P. K. Varshney, "Target location estimation in sensor networks with quantized data," *IEEE Trans. Signal Process.*, vol. 54, no. 12, pp. 4519–4528, Dec. 2006.

[3] X. Wang, M. Fu, and H. Zhang, "Target tracking in wireless sensor networks based on the combination of KF and MLE using distance measurements," *IEEE Trans. Mobile Comp.*,, vol. 11, no. 4, pp. 567 –576, April 2012.

[4] V. Veeravalli and P. K. Varshney, "Distributed inference in wireless sensor networks," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 370, no. 1958, pp. 100–117, Jan 2012.

[5] S. Marano, V. Matta, and L. Tong, "Distributed detection in the presence of Byzantine attacks," *IEEE Trans. Signal Process.*, vol. 57, no. 1, pp. 16 –29, Jan. 2009.

[6] A. Rawat, P. Anand, H. Chen, and P. K. Varshney, "Collaborative spectrum sensing in the presence of Byzantine attacks in cognitive radio networks," *IEEE Trans. Signal Process.*, vol. 59, pp. 774–786, Feb. 2011.

[7] A. Vempaty, K. Agrawal, H. Chen, and P. K. Varshney, "Adaptive learning of Byzantines' behavior in cooperative spectrum sensing," in *Proc. IEEE Wireless Comm. and Networking Conf. (WCNC)*, Mar. 2011.

[8] A. Vempaty, O. Ozdemir, K. Agrawal, H. Chen, and P. K. Varshney, "Localization in Wireless Sensor Networks: Byzantines and Mitigation Techniques," *IEEE Trans. Signal Process.*, vol. 61, no. 6, pp. 1495–1508, 15 March 2013.

[9] T.-Y. Wang, Y. S. Han, P. K. Varshney, and P.-N. Chen, "Distributed fault-tolerant classification in wireless sensor networks," *IEEE J Sel. Area Comm.*, vol. 23, no. 4, pp. 724 – 734, April 2005.