Collaborative Localization in Visual Sensor Networks

MAHMUT KARAKAYA, Oak Ridge National Laboratory
HAIRONG QI, University of Tennessee

Collaboration in visual sensor networks is essential not only to compensate for the limitations of each sensor node but also to tolerate inaccurate information generated by faulty sensors. This paper focuses on the design of a collaborative target localization algorithm that is resilient to sensor faults. We first develop a distributed solution to fault-tolerant target localization based on a so-called certainty map. To tolerate potential sensor faults, a voting mechanism is adopted and a threshold value needs to be specified which is the key to the realization of the distributed solution. Analytical study is conducted to derive the lower and upper bounds for the threshold such that the probability of faulty sensors negatively impacts the localization performance is less than a small value. Second, we focus on the detection and correction of one type of sensor faults, error in camera orientation. We construct a generative image model in each camera based on the detected target location to estimate camera’s orientation, detect inaccuracies in camera orientations and correct them before they cascade. Based on results obtained from both simulation and real experiments, we show that the proposed method is effective in localization accuracy as well as fault detection and correction performance.

Categories and Subject Descriptors: I.4 [Image Processing and Computer Vision]: General
General Terms: Scene Analysis, Sensor fusion

ACM Reference Format:
DOI = 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

1. INTRODUCTION
Although vision serves as the most powerful of human senses, it has not been adequately exploited in wireless sensor networks. In the past few decades, many multicamera systems have been developed for different applications ranging from security monitoring to surveillance. In these applications, expensive and high-resolution cameras with pan-tilt-zoom capability are usually deployed into large buildings (e.g., malls and airports) and open areas (e.g., parking lots and public parks) to capture the events in a controlled sensing field [Micheloni et al. 2010]. In general, the position and orientation of cameras are predetermined and well-ordered to optimize the placement of cameras. However, in a hostile and dangerous environment (e.g., battlefield), it is not possible or feasible to deploy the cameras with accurate position and orientation. Additionally, in case of a large amount of sensors deployment, it is impractical to manipulate camera locations and orientations after deployment in order to reach a de-
sired visual coverage. Therefore, camera nodes might be randomly deployed into the sensing field from a moving platform (e.g., airplane or vehicle) in order to monitor the environment [Hynes et al. 2004].

Due to their high cost and significant power consumption, random deployment of camera nodes in a sensing field was not practical in the past. However, recent advances in imaging, networking, and circuit design technologies have made it possible to produce significantly small-size, low-power, and affordable visual sensor platforms with on-board processing and wireless communication capabilities [Rinner and Wolf 2008]. These technological improvements dramatically change the concept of cameras from being a black box that can only capture videos or images to being an intelligent device that not only takes pictures but also analyzes and reports the events in the scene.

Although the development of visual sensor platforms is still at its early stage, there have already been quite a few platforms reported, including, for example, Stanford’s MeshEye system [Hengstler et al. 2007], UC Berkeley’s CITRIC camera node [Chen et al. 2008], the WiCa (wireless camera) node from NXP research and Philips research [Abbo and Kleihorst 2002], UCLA’s Cyclops [Rahimi et al. 2004], and CMU’s DSPcam [Kandhalu et al. 2009]. Among these platforms, the most successful one is probably CMU’s CMUCam [Rowe et al. 2007]. Its fourth generation (CMUCam4 [Goode et al. 2012]) already costs below $100 as of today as compared to its third generation, sold for $300 in 2010. In addition, the recent rapid development of smartphones provides another affordable yet robust and powerful platform with more than required capabilities that a visual sensor platform needs (i.e., a camera, a processor, a wireless card). We can predict that in the near future the cost for visual sensor platforms will be around $50 or even less in mass production where hundreds of such platforms can be easily deployed to form a visual sensor network (VSN).

VSNs have generated a new emerging interdisciplinary research field and have attracted researchers from many diverse disciplines including computer vision, image processing and wireless sensor networks. Based on their potential capabilities, many researchers refer to the visual sensor network as the fundamental of the next generation of smart surveillance systems [Aghajan and Cavallaro 2009]. Visual sensor networks are facilitated in many different multi-camera applications in diverse environments ranging from surveillance to entertainment.

Although many potential applications have been made possible using these powerful visual sensor platforms, the sensing nature of visual sensors also presents unique challenges to conventional scalar sensor network (SSN) solutions [Qian and Qi 2008]. First of all, the huge data volume of visual sensors generally requires high network bandwidth for data transmission. Secondly, the existence of “visual occlusions” among crowded targets cannot be avoided due to the directional sensing characteristic of cameras. Thirdly, visual information obtained by each sensor node is neither sufficient nor accurate because of the limited field of views and limited computational capacity of visual sensors. Finally, although visual sensor nodes are roughly calibrated (including both external parameters, i.e., position and orientation, and internal parameters, i.e., image center, focal length, and distortion coefficients) at initial sensor placement, the calibration of some nodes may not be accurate or external parameters may change during the network’s lifetime because of external effects such as wind, seismic events, and precipitation [Clouqueur et al. 2004]. These sensor errors may cause the generation of “faulty” information that would affect the accuracy of decision making. Therefore, the design of a practical solution for various VSN applications generally requires simple but effective local processing to reduce data transmission as well as collaboration among sensor nodes to not only compensate for the limitations of each sensor node but also improve the accuracy and robustness of the sensor network.
In this paper, we focus on the target localization problem. Taking advantage of the distributed sensing nature and collaborative in-network processing capability of VSNs, we will tackle challenging issues in this classic computer vision problem, including localizing targets in a crowded environment with the existence of visual occlusion, partial or inaccurate information, and presence of a number of faulty nodes in the network. Assume that each visual sensor node is calibrated after initial deployment using algorithms like [Devarajan and Radke 2007], i.e., assume initially most sensors still provide correct information, we first design and develop a distributed solution for fault-tolerant target localization where each node performs simple but effective image processing algorithms locally and integration is done among selected number of “most informative” nodes to arrive at the final result. We conduct analytical study to derive the lower and upper bounds for the voting threshold value needed during integration to guarantee that there is adequate amount of redundant information to tolerate the fault in visual sensors.

Under the context of target localization, since visual sensors are directional sensing devices, small amount of external force would cause more deviation in their orientation than in their position and relatively small deviation in camera orientation would introduce more significant fault to the network compared to the deviation in sensor position. Therefore, for fault detection and correction, we focus on errors in camera orientation. Based on the locations of detected targets, we then construct a generative image model in each sensor node that estimates the camera orientation, detect inaccuracies in camera orientations and correct them before they cascade.

The main contributions of this paper are two-fold:

1. We conduct analytical study on the upper and lower bounds of the voting threshold value used during integration such that a tradeoff can be achieved between energy conservation and redundancy. The bounds are derived based on both the visual coverage estimation and the probability estimation of the existence of faulty nodes. This automatic threshold selection scheme greatly facilitates practical implementation of distributed solutions to collaborative target localization with fault tolerance.

2. In many applications, it may not be possible or feasible to periodically calibrate the position and orientation of sensor nodes after the initial deployment. By constructing a generative image model, our proposed algorithm is able to not only detect the faulty node but also further correct them before the faults cascade. Thus, the proposed target localization algorithm is more reliable and robust and energy-efficient.

The paper is organized as follows: Section 2 describes the background and related works in the fault tolerance study in sensor networks. Section 3 describes the target localization method. Collaborative processing with global certainty map is presented in Section 4. In Section 5, distributed fault-tolerant collaborative localization algorithm with an analytic study on the voting threshold selection is presented. Section 6 describes the distributed fault detection and correction algorithm in visual sensor networks. Experimental results are presented in Section 7. Finally, we conclude the paper in Section 8.

2. BACKGROUND AND RELATED WORKS
This section presents an overview of the works in literature related to fault tolerance methods in sensor networks. Sources of faults in sensor networks can be classified into two main categories: sensing-based faults and networking-based faults [Paradis and Han 2007]. First, since sensor nodes are usually deployed into a hostile or harsh environment with limited energy sources, they may generate faulty information or
may totally fail during data collection because of external interference, power shortage, or failure of hardware. Second, networking-based faults are mainly generated by either permanent or temporary failures during the communication between nodes because of the low communication bandwidth, link traffic congestion, or node movement to out of communication range, etc.

To address these issues, there exist many works related to fault tolerance in scalar sensor networks (SSNs) for robust surveillance and monitoring applications where the sensing devices are normally 1-D omnidirectional (e.g., temperature and acoustic sensor). Most research focuses on networking-based fault tolerance and various efficient methods have been developed to recover the faulty information by transferring the local sensor data in each node through different routes in the network. [Bredin et al. 2010] proposed the multi-path routing technique to establish k-connectivity between every pair of nodes in the network by effective sensor deployment and selection of the sleep/wakeup schedule of sensor nodes. [Pu et al. 2009] extended the k-connectivity algorithm in order to improve connectivity through placing additional sensor nodes and reach k-connected or partially k-connected sensor network. To ensure a reliable communication topology in multi-tier heterogeneous sensor networks, [Abedi et al. 2011] presented a Bayesian Network model by computing the failure probability of relay nodes, providing at least two disjoint paths to the base station, and selecting the most optimal path with the lowest probability of failure.

These existing networking-based fault tolerance methods focus on finding alternative paths for data transmission with minimum energy consumption in order to provide reliable and real-time communication among sensor nodes. However, even if the communication link works in real time without any failure, the received data from an individual sensor node might still be faulty due to the failure in the sensing level of the node (e.g., inaccurate sensor calibration or hardware failure). In [Clouqueur et al. 2004], the fault tolerance in target detection algorithm was based on taking the “mean” of local decisions obtained by each scalar sensor with extreme values dropped and results filtered by certain threshold. In [Ding et al. 2007], an exploratory work was introduced toward fault-tolerant target localization in SSNs by utilizing the “median” to filter out extreme values and combining estimations of target locations from multiple epochs/iterations. [Tuan et al. 2010] proposed fault tolerance by using the quartile method that not only determines the correct data range but also sifts the correct sensors by data discreteness. For more detailed survey, readers may refer to [Yu et al. 2007] where the existing fault management approaches in wireless sensor networks were classified into three categories as fault detection, diagnosis, and recovery. Although all these works provide detailed investigation of the 1-D scalar sensing models, none of which consider the directional sensor deployment into the sensing field.

Compared with the 1-D scalar sensors, visual sensors have a limited and directional sensing capability, also referred to as the field of view (FOV). Even if the existing fault tolerance methods in SSNs may address the networking-based faults in VSNs, they cannot be directly applied to visual sensor networks to tolerate the sensing-based faults. Therefore, in this paper, we focus on the tolerance for sensor related faults and assume to have reliable communications between sensor nodes without any network failure. In the remainder of the paper, we refer to the fault as sensing-based fault.

Fault tolerance in VSNs is more challenging than in SSNs because of the unique features of cameras, including the existence of visual occlusion and the directional sensing characteristics with limited field of view. Due to its directional sensing nature, visual sensor nodes are very sensitive to any external interference or inaccuracy in their initial calibration (especially, camera orientation) that causes the generation of faulty information. Therefore, camera calibration is an essential prerequisite and demanding task in VSNs. Accurate and periodic camera calibration has been considered...
one of the effective approaches for “fault” detection, correction and tolerance in VSNs. One of the widely used techniques to calibrate the cameras is to deploy additional markers to the environment, such as a bright red LED or a set of special patterns like checker boards, and to estimate the camera parameters by using the epipolar geometry between cameras. In [Poelman and Kanade 1997] and [Devarajan and Radke 2007], the extracted feature points from the captured images were used to estimate the camera parameters by utilizing matrix factorization and iterative belief propagation. However, these methods either require the deployment of additional tools or the computational complexity of the feature extraction algorithm is too high to be deployed in real applications. Therefore, they may not be viable for periodic camera calibration.

In our previous study [Karakaya and Qi 2010], a centralized fault tolerance, detection and correction algorithm was proposed for target localization in VSNs with the existence of visual occlusion, partial information and a number of faulty nodes. The shortcomings of this method are the energy consumption of nodes, communication load, and system scalability in large-scale sensor networks because all available information from every sensor node is required to sink into the base station. On the other hand, in this paper, the distributed fault tolerant target localization algorithm is realized by the automatic selection of the voting threshold value, thus achieving a tradeoff between energy conservation and required redundant information for fault tolerance based on the visual coverage estimation and the probability estimation of the existence of faulty nodes in VSNs.

3. TARGET LOCALIZATION WITH LOCAL CERTAINTY MAP

Among all the major processes taking place in a VSN, i.e., sensing, processing, and communication, communication consumes most of the energy with energy spent on sensing and processing being negligible. Therefore, many adaptive video communication methods including [Gualdi et al. 2008] and [Micheloni et al. 2008] have been proposed to save the communication cost during the transmission of the video images to central processing unit. However, due to the sheer amount of data generated at each camera node, local processing is still needed to provide only the necessary information with a much smaller data volume than raw video sequences. We need to keep in mind that local processing cannot be computationally expensive; otherwise, it will consume as much energy as communication. In this section, we discuss a simple but effective target localization algorithm conducted locally at each sensor node.

In traditional target localization algorithms, the intersections of the back-projected 2D cones of the targets are calculated to localize all the individual targets. If the cones from different sensor nodes intersect at the same point, it can be considered there is at least one target in that intersection. However, in crowded environments, many “empty” intersections that are not actually occupied by any target are created due to either the visual occlusion (e.g., intersection D) or ghost positioning (e.g., intersection
C), as shown in Fig. 1. Therefore, the target existence information at the intersections is not certain. To remove the uncertainty and to detect the real locations of the targets have been a very challenging problem in visual sensor networks.

In this section, we briefly describe an inverse approach to traditional target localization problem proposed by the authors in [Karakaya and Qi 2011]. Instead of resolving the uncertainty about the target existence, we identify and study the non-occupied areas in the visual cones. Since it is certain that there is no target in the corresponding region (assuming all sensor node inputs are trustworthy for now with relaxations discussed in the following sections), we refer to the map generated from this process as the certainty map. In a certainty map, the environment is divided into uniform grids where each grid represents that the corresponding ground space is certain about target non-existence (labeled with white) or uncertain about target existence (labeled with black). If an area within the FOV of a sensor node is not occupied or occluded by any object, it is declared as a non-occupied area. The occupied areas are the ones where it is possible that there exist targets. The uncertainty is due to either occlusion or outside of the FOV of the camera.

Fig. 2 illustrates the steps in constructing the certainty map at a sensor node. After background subtraction, each object/segment extracted sweeps a cone in the 3D space as shown in Fig. 2a. To find the 2D visual cones of the object, these 3D cones are projected onto a plane parallel to the ground as shown in Fig. 2b. The non-occupied (white) areas in the 2D visual cones are thus determined to construct the local certainty map as shown in Fig. 2c.

In this paper, it is assumed that after the initial deployment, each sensor node, \( s_i \), estimates, once, its coordinates in the 2D global coordinate system as \( (x_{s_i}, y_{s_i}) \) and its orientation as \( \theta_{s_i} \) by using a camera calibration algorithm, like the one in [Devarajan and Radke 2007]. Let \( v_{s_i} \) denote the vector that describes the non-occupied areas within the FOV of the sensor node, \( s_i \). The vector is composed of a series of \( (\varphi_{i,j}, \psi_{i,j}) \) pairs, where \( \varphi_{i,j} \) and \( \psi_{i,j} \) record the starting and ending angles, respectively, of the \( j^{th} \) non-occupied area in the corresponding planar projection of the 3D visual cones onto the 2D ground space, \( j = 1, \ldots, K_i \), where \( K_i \) is the number of non-occupied areas detected in sensor node \( s_i \). Instead of transmitting the foreground image or binary local certainty map, we can then describe the certainty map through a much condensed vector representation in order to reduce the amount of transmitted data volume from each sensor node,

\[
v_{s_i} = [x_{s_i}, y_{s_i}, \varphi_{1,1}, \psi_{1,1}, \ldots, \varphi_{1,K_i}, \psi_{1,K_i}] \tag{1}
\]
Note that $v_s$, can be further compressed using existing coding and lossless compression techniques to decrease the data size to save the communication cost without introducing much computational overhead for each sensor node. The conversion between $v_s$ and the certainty map can be done through a mapping function, $f(v_s)$, whose value at coordinate $(x, y)$ is,

$$f_{x,y}(v_s) = \begin{cases} 
1, & \text{if } \varphi_{i,j} \leq \arctan \frac{x-x_s}{y-y_s} \leq \psi_{i,j} \\
0, & \text{otherwise.}
\end{cases} \quad (2)$$

Let $S = \{s_1, s_2, \ldots, s_N\}$ denote the set of sensor nodes in the network, we have

$$U_S = \bigcup_{i=1}^{N} f(v_{s_i}) \quad (3)$$

where $U_S$ denotes the union (or binary OR operation) formed by all the local certainty maps in $S$. Targets are located in the complement of $U_S$, i.e., the unresolved regions.

If the information obtained from each sensor node is accurate and trustworthy, the size of the total uncertain region in the global certainty map would monotonically decrease and the convergence of the global certainty map is guaranteed. The idea is that the uncertain regions will be shrinking as local certainty maps are fused. If the non-existence of target for certain region is declared by one sensor node, the corresponding region is globally announced as non-occupied and cleared from the certainty map. If the entire surveillance area is covered by sensor nodes, then the only uncertain region left would be the location of targets.

4. COLLABORATIVE TARGET LOCALIZATION WITH GLOBAL CERTAINTY MAP

Both centralized and distributed integration can be adopted in order to fuse the local certainty maps generated at each node. In the centralized approach, each camera node sends its local certainty map, i.e., vector $v_{s_i}$, to a processing center for information fusion. In the distributed approach, a global certainty map is constructed that is propagated through the network to integrate with local certainty maps.

In order to save energy, minimum number of camera nodes need to be involved and the route (or itinerary) of certainty map migration plays a key role in arriving at this minimum subset of nodes. In [Karakaya and Qi 2011], the authors have proposed a dynamic itinerary selection method where the route of propagation is determined by the content of the map such that it is guaranteed the next integration would be conducted by the node that clears the most uncertain region leading to the shortest path.

In the dynamic itinerary selection method, a global certainty map is constructed and initiated to be all black, i.e., not certain about target non-existence at any grid points within the surveillance area. Then each iteration sensor nodes compute their local certainty maps, calculate the size of area that can be cleared from the current global certainty map, and broadcast it through the network. The sensor node which has the largest clarification area (or winner) gets the priority over others to integrate its local certainty map with the global certainty map. This winner then broadcasts the updated global certainty map to the network to allow other sensor nodes to recalculate the amount of additional clearance on the certainty map. This procedure repeats until the winner with the largest clarification amount is less than a predefined threshold. That is, none of the remaining sensor nodes would have adequate additional clearance area on the certainty map as the size of the clearance area monotonically decreases and that the integration process can be stopped immediately.

The advantage of using dynamic itinerary in distributed integration is that less number of nodes might be involved in the integration process and that the cleared
areas from the certainty map monotonically decreases between iterations. However, dynamic itinerary also introduces overhead by having to communicate both the size of cleared area and the global certainty map in several iterations, while in centralized processing only local certainty map is communicated in one iteration. The key is how many sensors are involved in the integration process. Both analytical study and experiments with simulated and real data have been conducted in [Karakaya and Qi 2011] to evaluate the performance of these two integration schemes from the perspectives of the amount of data transmitted and total energy consumption. It has been shown that the dynamic itinerary-based distributed integration is effective in localization accuracy as well as energy and bandwidth efficiency. Note that energy consumption and bandwidth usage might be further decreased by defining clusters within the network and communicating between visual sensor nodes through multicast transmission instead of broadcast.

5. DISTRIBUTED FAULT-TOLERANT TARGET LOCALIZATION

Until now, we have assumed that the information obtained from each visual sensor node is accurate and trustworthy. However, in real world applications, this is seldom true. Sensor faults due to initial calibration error or external effects frequently occur that would deviate from the initial calibration results. In this section, we first model the error incurred in camera orientations and present the voting algorithm for fault tolerant certainty map integration. Then, for the distributed integration, the threshold value of the voting algorithm is analyzed based on a tradeoff between energy conservation and redundancy in order to tolerate the fault in sensors while localizing the targets in the sensing field.

5.1. Fault Model in Visual Sensors

Each sensor node, deployed into a sensing field, has six degrees of freedom including pan, tilt, roll as orientation parameters and \(x, y, z\) as position parameters. In order to simplify the sensor deployment, we assume that each camera is tightly attached to a sensor platform which is parallel to the ground and each sensor node is deployed in a flat surface at a fixed height without any vertical orientations (i.e., tilt and roll). Relaxation to these assumptions will be discussed at the end of Sec. 6. Therefore, the horizontal orientation (i.e., pan) and the position (i.e., \(x, y\)) of each sensor node in the sensing field become sensor deployment variables. In the rest of the paper, we refer to the orientation as horizontal orientation or pan and use them interchangeably.

When visual sensor nodes are deployed into a sensing field, the external parameters of cameras, i.e., position and orientation, can be determined by using different calibration methods. However, initial calibration of some sensor nodes might be inaccurate or their positions and orientations may change after deployment due to node movements or external effects. The error in sensor orientation and position negatively impacts the accuracy of target localization.

For a visual sensor node, it is more prone to have error in sensor orientation than in sensor position because same amount of external force applied to the visual sensor node changes the sensor orientation more dramatically than the sensor position. Moreover, due to the directional sensing nature of cameras, relatively small deviation in camera orientation causes more significant fault to target localization than same amount of deviation in the sensor position. Therefore, in this paper, we focus on the error in the horizontal orientation of cameras.

The error in horizontal orientation of the faulty nodes can be modeled by a combination of two types of errors. First of all, the Gaussian noise models the initial calibration inaccuracy. Secondly, the Byzantine fault [Clouqueur et al. 2004] models the error generated by external effects where orientation becomes arbitrary and can be
Fig. 3: (a) Image captured by a camera and (b) its local certainty map. Data fusion by using (c) Binary certainty map and (d) Gray-scale certainty map. From (e) to (i) final version of the certainty map using voting with the threshold 1, 2, 3, 4, and 5, respectively.

any value in \([0^\circ, 360^\circ]\). The Byzantine fault originates from the Byzantine Generals’ problem (Lamport et al. 1982), an agreement problem to arrive at a unanimous decision in the presence of the traitors whether to attack enemy or to withdraw. Therefore, the camera orientation can be expressed as,

\[
\theta_{s_i} = \theta^*_{s_i} + N_{s_i}(0, \sigma) + \delta_{s_i}
\]  

(4)

where \(\theta_{s_i}\) and \(\theta^*_{s_i}\) are the actual (or measured) and true (or calibrated) orientations of the \(i^{th}\) sensor node, \(s_i\), respectively. \(N_{s_i}(0, \sigma)\) is the Gaussian noise with zero-mean and standard deviation \(\sigma\), and \(\delta_{s_i}\) denotes the Byzantine fault in orientation.

5.2. Voting

Single sensor node, which gives inaccurate information about the location of the targets, negatively impacts the performance and sometimes causes failure of the localization algorithm. To obtain more accurate and robust results, certain degree of redundancy is necessary to tolerate the inaccurate information and failure of some nodes.

Voting is one of the most commonly used multiple sensor fusion techniques to integrate individual sensor results (Klein 1993). In our previous paper (Karakaya and Qi 2009), we proposed to utilize the voting approach for target counting to tolerate potential inaccurate information from sensor nodes where each sensor node has equal importance to contribute to the voting result. In the voting approach, if a sensor node declares the target non-existence at a specific location of the certainty map, the certainty value of that location is increased. Therefore, instead of a binary certainty map with 1 indicating 100% certainty of non-existence of targets, as shown in Fig. 3c, the voting approach generates a gray-scale certainty map, shown in Fig. 3d with non-zero regions indicating certain degree of confidence of the non-existence of targets. The higher the number of votes, the more certain it is.
A threshold value needs to be specified in the end to convert the gray-scale certainty map to binary for decision making purpose. To choose 1 as the threshold value means that there is no tolerance for failure of any sensor node. If one of the sensor nodes claims the non-existence of any object at any location in the certainty map, the algorithm believes it and clears the corresponding region from the certainty map as in the case of the binary certainty map. To be more robust to sensor failures, the threshold value can be chosen greater than one. For example, if the threshold value is selected as two, to clear a specific area from the certainty map, at least two sensors must declare the clearness of that region. Higher threshold value requires more sensor nodes to reach a consensus. Fig. 3(e-i) shows the final versions of gray-scale certainty map using the voting approach with different threshold values 1, 2, 3, 4, and 5, respectively. The size of the uncertain areas is very small for small threshold values. When increasing the voting threshold value it also increases the size of uncertain areas in the final certainty map and begins to introduce the non-occupied areas as part of the object, as shown in Fig. 3h and i.

In order to conserve energy in the energy starving sensor network, not only each visual sensor node transmits a very limited amount of data but that a limited number of sensor nodes should be involved in the decision making procedure. However, in the sensing field where faulty nodes are likely to exist, we cannot trust a single sensor which might be a faulty node and declare the non-existence of a target within a region. Therefore, to tolerate the fault in the sensor network, adequate amount of redundant information is required in the voting algorithm. However, the amount of required redundant information is unknown. It has to be decided accurately before the certainty map integration in order to reach a trustworthy final certainty map while saving energy by involving a limited number of sensor nodes.

5.3. Analytic Study of the Voting Threshold

Since visual sensor nodes in the sensing field might be faulty, distributed integration requires redundant information to tolerate the fault. In order to find a tradeoff between energy conservation and required redundant information, we first estimate the probability that a specific grid point of the sensing field is covered by exactly \( w \) many faulty sensor nodes, \( P_f(w) \). We assume that the infinitesimal visual sensor nodes with uniform FOV and sensing radius are randomly deployed within a very large two-dimensional sensing field, \( R \). Since each region in the sensing field has equal importance based on the probability of target existence, all sensor nodes are uniformly and independently distributed into the sensing field. Based on this deployment strategy, the locations of visual sensor nodes can be modeled by a two-dimensional stationary Poisson point process with sensor density \( \lambda_s \) [Wang et al. 2010]. It is also assumed that orientations of visual sensors are uniformly distributed over \([0^\circ, 360^\circ)\). Let \( \rho \) and \( \theta \) denote, respectively, the sensing radius and angle of view of a sensor node.

A sensor node covers a specific grid point \((x, y)\) in \( R \), of the sensing field if the node is located in a circular area \( A \) with radius \( \rho \) centered at the corresponding grid point and is oriented towards the center of the circle which is illustrated in Fig. 4. In the rest of the paper, the circular area \( A \) is referred to as the “detectability area”. Therefore, the probability that exactly \( w \) many faulty sensor nodes cover a specific grid point is

\[
P_f(w) = \sum_{j=w}^{\infty} \sum_{i=w}^{j} \mathcal{P}(j; \lambda_s \times A) C_i^j (p)^i (1-p)^{j-i} C_w^i (p_f)^w (1-p_f)^{i-w}
\]

where \( \mathcal{P}(j; \lambda_s \times A) \) denotes the probability that a detectability area \( A \) contains exactly \( j \) sensor nodes from a Poisson point process with sensor density \( \lambda_s \), i.e., \( \mathcal{P}(j; \lambda_s \times A) = e^{-\lambda_s \times A} \frac{(\lambda_s \times A)^j}{j!} \) where \( A = \pi \rho^2 \). Also, \( p \) denotes the probability of the sensor node...
facing towards the center of detectability area, \(A\), i.e., \(p = \theta/(2\pi)\), \(p_f\) denotes the probability of the sensor node being faulty, and \(C_j^i\) denotes the number of combinations of \(i\)-node subset from a \(j\)-node set. Eq. 5 can be further derived as,

\[
P_f(w) = \sum_{j=w}^{\infty} \mathcal{P}(j; \lambda_s \times A) p^wp_f^j (1-p)^{j-w} C_i^j C_w^i \left( \frac{p(1-p_f)}{1-p} \right)^{i-w} \\
= \sum_{j=w}^{\infty} \mathcal{P}(j; \lambda_s \times A) p^wp_f^j (1-p)^{j-w} \sum_{z=0}^{i-w} C_j^i C_w^{i-z} \left( \frac{p(1-p_f)}{1-p} \right)^{i-w} \\
\overset{(a)}{=} \sum_{j=w}^{\infty} \mathcal{P}(j; \lambda_s \times A) p^wp_f^j (1-p)^{j-w} C_j^i \sum_{z=0}^{i-w} \left( \frac{p(1-p_f)}{1-p} \right)^{i-w} \\
\overset{(b)}{=} \sum_{j=w}^{\infty} e^{-\lambda_s \times A} (\lambda_s \times A)^j \frac{1}{j!} p^wp_f^j (1-p)^{j-w} \sum_{z=0}^{i-w} \frac{1}{w!(j-w)!} \left( \frac{1-pp_f}{1-p} \right)^{i-w} \\
= \frac{1}{w!} e^{-\lambda_s \times A} (\lambda_s \times A pp_f)^w \sum_{j=w}^{\infty} \frac{(\lambda_s \times A)(1-pp_f)^j}{(j-w)!} \\
= \frac{1}{w!} e^{-\lambda_s \times A} (\lambda_s \times A pp_f)^w \sum_{n=0}^{\infty} \frac{(\lambda_s \times A)(1-pp_f)^n}{(n)!} \\
\overset{(c)}{=} \frac{1}{w!} e^{-\lambda_s \times A} (\lambda_s \times A pp_f)^w e^{(\lambda_s \times A)(1-pp_f)} \\
\overset{\text{where}}{=} \frac{1}{w!} (\lambda_s \times A pp_f)^w e^{-\lambda_s \times A pp_f} \\
= \mathcal{P}(w; \lambda_s \times A \times p \times p_f) \\
= \mathcal{P}(w; \lambda_f \times A \times p) \\
\tag{6}
\]

where (a) follows the combination properties, \(C_j^i C_w^{i-z} = C_j^i C_w^{i-z}\), (b) follows the binomial coefficient property, \((x+y)^n = \sum_{z=0}^{\infty} C^n_x z^{n-z} y^z\) where \(z = i-w\), and (c) follows the property of power series, \(\sum_{n=0}^{\infty} \frac{z^n}{n!} = e^z\). \(\lambda_f\) denotes the density of faulty sensor nodes i.e., \(\lambda_f = \lambda_s \times p_f = \frac{N_s}{\pi} \times p_f = \frac{N_f}{\pi}\) where \(N_s\) and \(N_f\) denote the total number of deployed sensor nodes and expected number of faulty nodes in the sensing field, respectively.

From the derivation result in Eq. 6, we observe that the probability that exactly \(w\)many faulty sensor nodes cover a specific grid point, \(P_f(w)\), follows the Poisson point process with density \(\lambda_s \times p \times p_f\) in the detectability area \(A\). In order to accurately tolerate the faulty nodes in a sensor network, the redundant information at a specific grid point of the sensing field should be more than the number of faulty nodes. The amount of the redundant information is controlled by the selection of the voting threshold. Therefore, the selected voting threshold should ensure the probability of fault that a specific grid point in the sensing field is covered by at least \(W\)-many faulty sensor nodes.
Fig. 4: Illustration of the sensor deployment in a detectability area, A.

is smaller than a fault-tolerance value $\varepsilon_1$. This probability of fault follows,

$$ P_f(w \geq W) < \varepsilon_1 $$

$$ \sum_{w=W}^{\infty} P_f(w) < \varepsilon_1 $$

$$ \sum_{w=W}^{\infty} P(w; \lambda_f \times A \times p) < \varepsilon_1 $$

$$ 1 - \sum_{w=0}^{W-1} P(w; \lambda_f \times A \times p) < \varepsilon_1 $$

$$ 1 - F_{\mathbb{P}}(W - 1; \lambda_f \times A \times p) < \varepsilon_1 $$

(7)

where $F_{\mathbb{P}}(W - 1; \lambda_f \times A \times p)$ is the cumulative probability distribution (cdf) of Poisson distribution with parameter $\lambda_f \times A \times p$. Thus, Eq. 7 introduces the lower bound for the voting threshold selection.

In addition, the selected voting threshold should also ensure the visual K-coverage probability that each grid point is covered by at least K sensor nodes is higher than a certain probability in order to accurately tolerate the faulty sensor nodes in the sensing field. In other words, the probability that each point is covered by less than K sensor nodes is smaller than a coverage-tolerance value $\varepsilon_2$. Therefore, the K-coverage constraint for the voting threshold selection is

$$ P(k \geq K) > 1 - \varepsilon_2 $$

$$ \sum_{k=K}^{\infty} P(k) > 1 - \varepsilon_2 $$

$$ 1 - \sum_{k=0}^{K-1} P(k) > 1 - \varepsilon_2 $$

$$ \sum_{k=0}^{K-1} P(k) < \varepsilon_2 $$

$$ F_{\mathbb{P}}(K - 1) < \varepsilon_2 $$

(8)
where $F_P(K - 1)$ is the cumulative probability distribution (cdf) of visual coverage probability, $P(k)$ which is discussed in [Karakaya and Qi 2012] in detail. Thus, Eq. 8 introduces the upper bound for the voting threshold selection.

The optimization problem of voting threshold selection which ensures the fault-tolerant target localization can be expressed as,

$$V_{thr} = \arg \min_W \left| 1 - F_P(W - 1; \lambda_f \times A \times p) - \varepsilon_1 \right| \quad (9)$$

where $V_{thr}$ is the selected voting threshold. Therefore, the solution to the optimization problem is that the voting threshold is the smallest positive $V_{thr}$ value of Eq. 9. However, there is no explicit solution to Eq. 9. $V_{thr}$ can be found by using the exhaustive search method.

In this paper, it is assumed that homogeneous visual sensor nodes with the same sensing radius, $\rho$ and angle of view, $\theta$ are deployed into the sensing field for collaborative target localization. In order to make the scenario more realistic where homogeneous visual sensors are likely to be deployed, we can relax these assumptions by considering the heterogeneous sensor deployment to the sensing field. In the heterogeneous visual sensor deployment, we deploy different types of visual sensor nodes into the sensing field with different sensor density, $\lambda_s$, sensing radius, $\rho$, and angle of view, $\theta$. For instance, if two types of sensor nodes (Type I and Type II) are deployed into the sensing field with sensor density, $\lambda_{s1}$ and $\lambda_{s2}$, sensing radius, $\rho_1$ and $\rho_2$, and angle of view, $\theta_1$ and $\theta_2$, a target can be covered by $k$ many sensor nodes with any combinations of Type I and Type II sensor nodes. In addition, $w$ out of $k$ many sensor nodes can be faulty with any combinations of Type I and Type II sensor nodes. Therefore, in the calculation of the K-coverage probability for the voting threshold selection, we have to consider the different detection probability of each type of sensor node in their different size of detectability area, $A$ and calculate the K-coverage probability based on their sensor related parameters (i.e., $\lambda_s$, $\rho$, and $\theta$). The complete analysis of coverage estimation in heterogeneous visual sensor networks was presented in [Karakaya and Qi 2012]. The probability that exactly $w$-many faulty sensor nodes cover a specific grid point $P_f(w)$ can be calculated with the same way as the K-coverage probability.

6. DISTRIBUTED FAULT DETECTION AND CORRECTION THROUGH GENERATIVE IMAGE MODEL

In Sec. 5, we presented the voting algorithm with automatic threshold selection to tolerate camera errors upon initial calibration. Therefore, targets in the sensing field are reliably localized by a collaborative effort of distributed camera nodes. However, if the initial errors do not get to be corrected in a timely fashion, together with potential Byzantine faults, errors can cascade that would eventually affect the accuracy of the localization result. In Sec. 5.1, faults in camera orientations are modeled in two types, namely, Gaussian noise and Byzantine fault. Gaussian noise models the inaccuracy of camera orientation due to environmental noise or calibration error. The Byzantine fault model assumes that camera orientation might be an arbitrary value so provided information from that sensor node is arbitrary as well.

To detect faulty nodes with inaccuracies in camera orientation estimation, we take into account both the actual image captured by each node and the final certainty map generated from the collaborative processing. We first utilize a so-called “generative image model” to estimate the ideal foreground image of each camera based on the target locations extracted from the final certainty map and the camera’s orientation and position. We then propose the fault detection and correction model to identify the faulty nodes and correct them from orientation inaccuracies.

6.1. Generative Image Model

The generative image model was proposed in [Fleuret et al. 2008] to generate the ideal background subtracted images if targets and camera locations are known. Let $R_{s_i}$ denote the actual 2D foreground image and $E_{s_i}$ denote the synthetic 2D image generated by the sensor node, $s_i$, based on the final certainty map, where each target is represented as a cylindrical object. The planer projection of the 3D visual cones preserves the most useful information of moving targets along a plane which is perpendicular to the projection plane in target localization applications [Yang et al. 2003]. Therefore, instead of using 2D images ($E_{s_i}$ and $R_{s_i}$) to estimate the orientation of each sensor node, we use 1D scanline images ($e_{s_i}$ and $r_{s_i}$), that is generated by summing the rows of the foreground image, to measure the distance between a synthetic foreground image, $e_{s_i}$, and the actual foreground image, $r_{s_i}$. The synthetic foreground images, $e_{s_i} (\theta)$, are generated based on all possible camera orientations in $[0^\circ, 360^\circ]$ with a selected step-size such that $(360/\text{step-size})$-many candidate synthetic images are generated.

For each sensor node, $s_i$, we calculate the normalized pseudo-distance, $\Psi$, to measure the distance between the two 1D scanline images, $r_{s_i}$ and $e_{s_i} (\theta)$ [Fleuret et al. 2008],

$$\Psi(r_{s_i}, e_{s_i} (\theta)) = \frac{|r_{s_i} \odot (1 - e_{s_i} (\theta)) + (1 - r_{s_i}) \odot e_{s_i} (\theta)|}{|r_{s_i}|}$$

(10)

where $\odot$ denotes the element-wise product between two images and $|r_{s_i}|$ is the sum of its pixel values for any gray-scale image $r_{s_i}$.

6.2. Fault Detection and Correction Model

The expected orientation of each camera, $\theta_{s_i}^e$, is the one which minimizes the pseudo-distance,

$$\theta_{s_i}^e = \arg \min_\theta \Psi(r_{s_i}, e_{s_i} (\theta))$$

(11)

where the fault in camera orientation is detected if there is a difference between the expected camera orientation, $\theta_{s_i}^e$, and the actual (or measured) camera orientation, $\theta_{s_i}$. We then update the actual camera orientation,

$$\theta_{s_i} = \theta_{s_i}^e.$$ 

(12)
Collaborative Localization in Visual Sensor Networks 39:15

Fig. 5 illustrates the calculation of the pseudo-distance between the actual and the synthetic foreground images. The simple scenario for fault-tolerant target localization is shown in Fig. 5a with two targets. Fig. 5b shows the actual and synthetic foreground images, \((R_s, E_s)\) and their 1-D scanline images, \(r_s\) and \(e_s\), respectively. The pseudo-distances of real and synthetic 1-D scanline images for different \(\theta\) values are shown at the bottom of Fig. 5b. The expected orientation is the orientation value, \(\theta\) which shows the minimum pseudo-distance.

In this paper, we focus on the error in camera orientation and ignore the error in sensor localization. In order to make the scenario more realistic where error in vertical camera orientation (i.e., tilt and roll) is likely to occur, we can relax the initial deployment assumptions made in Sec. 5.1 by considering the error in both the vertical and horizontal orientations of the camera. Error in horizontal orientation causes the shift in the certainty map and error in vertical orientation causes the shrinking or enlarging of the certainty map, as shown in Fig. 6. The proposed schemes can be generalized to handle inaccuracies in both vertical and horizontal camera orientations as well as camera position by using the generative image model where the ideal foreground images are generated not only for all possible horizontal camera orientations but also for possible vertical orientations and sensor locations.

7. EXPERIMENTS AND RESULTS

In this section, we evaluate the performance of the fault detection, correction, and tolerance algorithm for collaborative target localization in distributed visual sensor networks using both simulated and real experiments with different amount of faults added to camera orientations. The main purpose of the simulation experiments is to thoroughly evaluate the performance of the proposed fault-tolerant collaborative localization algorithm under different settings and to observe the performance trend which would otherwise be difficult to set up in real experiments.
7.1. Experiments using Simulation

After each deployment of targets and cameras, the simulation software generates the corresponding foreground image of targets in the FOV of each sensor node. Each sensor node then computes the 2D visual cones of the non-occupied areas using the planar projection and generates local certainty map as described in Sec. 3. By using the distributed integration, the certainty map is progressively clarified where the voting threshold is optimally selected. Targets are located at the remaining uncertain regions in the map. In order to detect and correct the faulty nodes, target locations in the final certainty map are broadcasted to each node. Then, each node first generates its synthetic image by using the generative image model and compares its actual image with the synthetic image. If the pseudo-distance between the actual image and synthetic image is not smaller than a pre-defined threshold, the sensor node determines itself as faulty and estimates its camera orientation by using the generative image model and then updates its orientation.

7.1.1. Experimental Setup, Minimum Sensor Density, and Optimal Threshold Selection. In our simulation, round-shaped targets of uniform size are deployed on a 2D sensing field, and infinitely small-size sensor nodes with uniform FOV and focal length are randomly deployed into the sensing field and directed horizontally facing the sensing field. The locations of each sensor node and target are randomly generated assuming there is no overlap between the targets and sensors. The orientation of each sensor node is a floating point number randomly generated in $[0^\circ, 360^\circ]$. In all the simulations, we assume each node is accurately calibrated and synchronized with each other after initial deployment. Also, each node is able to find its location by using a positioning system, such as GPS.

Following is the setup of some typical parameters: The 2D sensing field is 50m x 50m large. The radius of each target is 0.5m. The uniform sensing range of sensor nodes is 20m in length and 45$^\circ$ in angle. Each node is in the communication range of other nodes and is able to communicate with each other. Fig. 7 illustrates a random deployment example of 50 targets, represented as discs, and 500 cameras, represented as points.

When choosing parameters for sensor network deployment, one of the most important factors is the minimum sensor density to ensure the visual coverage probability that each point is covered by at least K sensor nodes is higher than a certain percentage. In other words, the probability that each point is covered by less than K sensor nodes is smaller than a coverage-tolerance value, $\varepsilon_2$, as incorporated in Eq. 8 to satisfy
Collaborative Localization in Visual Sensor Networks

the selected voting threshold values, $V_{thr}$. Therefore, in order to select the required threshold, minimum number of sensor nodes are supposed to be deployed into the sensing field to ensure the K-coverage constraint. Here, we first calculate the minimum number of sensors need to be deployed into the sensing field. Please refer to [Karakaya and Qi 2012] for more detailed discussion.

Fig. 8 shows the visual coverage probability for different K-coverage requirement values corresponding to different numbers of sensor nodes, $N_s$. We observe that the visual coverage probability decreases as K increases because of the more demanding coverage requirement. In addition, the visual coverage probability increases as $N_s$ increases due to denser visual sensor nodes deployed. We also observe that to satisfy the selected voting threshold values, $V_{thr}$, as 1, 2, 3, 4, or 5, that is, to have K equal to 1, 2, 3, 4, or 5, we need to deploy at least 160, 240, 320, 380, and 460 sensor nodes in the sensing field, respectively, if the coverage-tolerance value is 0.02. Therefore, we deploy 500 cameras into the sensing field in order to make sure there are enough sensors in the sensing field to localize 50 targets.

Then, the threshold value should be determined for distributed integration by using the automatic threshold selection method described in Section 5.3. Fig. 9 illustrates the probability that a specific grid point in the sensing field is covered by at least W-many faulty sensor nodes, $1 - F_P(W - 1; \lambda_f \times A \times p_f)$ and the probability that each grid point is covered by less than K sensor nodes, $F_P(K-1)$, that ensures the visual K-coverage probability.

We observe that to decrease the probability that a specific grid point in the sensing field is covered by at least W-many faulty sensor nodes, $1 - F_P(W - 1; \lambda_f \times A \times p_f)$, a lower voting threshold value is required. In addition, to increase the probability that a sensor node is faulty, $p_f$, increases the probability that a specific grid point in the sensing field is covered by at least W-many faulty sensor nodes and requires a higher voting threshold value. However, the voting threshold value cannot be selected too high otherwise it is not ensured to have accurate K-coverage in the sensor network for fault tolerance. In order to select the threshold value automatically based on the solution to Eq. 9, we set the tolerance values as $\varepsilon_1 = 0.02$ and $\varepsilon_2 = 0.02$ (horizontal line in Fig. 9b ), the threshold value for the minimum pseudo-distance to 0.7 and select the probability that sensor node is faulty, $p_f$ from 0.01 to 0.05.
7.1.2 Effect of Probability of Error in Sensor Node. We conduct five simulation experiments to study the effect of the probability that a sensor node is faulty, \( p_f \), on the performance of the fault detection and fault correction algorithm. In each set of the experiments, different amount of Byzantine faults are generated by changing \( p_f \) from 0.01 to 0.05 and added to the orientations of the sensor nodes randomly for ten times.

In the first experiment, we set \( p_f = 0.01 \). Since 500 sensor nodes are deployed into the sensing field, there are 5 faulty sensor nodes on average, i.e., \( N_f = 5 \). The lower bound of the voting threshold is 3 that is determined by the probability that a specific grid point is covered by at least \( W \)-many faulty sensor nodes, \( 1 - P_f(W - 1; \lambda_f \times A \times p) \) and the probability that each grid point is covered by less than \( K \) sensor nodes, \( F(K-1) \), for automatic threshold selection and (b) zoom in of the rectangle area in (a). The horizontal line is the selected tolerance values, \( \varepsilon_1 = 0.02 \) and \( \varepsilon_2 = 0.02 \).

Table I shows the confusion matrix of the fault detection results and some related metrics (i.e., accuracy, sensitivity, specificity, and precision) as well as the fault correction results. The confusion matrix consists of four result cells that report true positive (i.e., faulty nodes detected as faulty), false positive (i.e., non-faulty nodes detected as faulty), true negative (i.e., non-faulty nodes detected as non-faulty), and false negative (i.e., faulty nodes detected as non-faulty). Accuracy is the ratio of the true results (both true positives and true negatives) to the total number of all results. Precision measures the proportion of the true positives against all the positive results (both true positives and false positives). Sensitivity and specificity are defined, respectively, as the proportion of true positives which are correctly detected faulty nodes as faulty and the proportion of true negatives which are correctly detected non-faulty nodes as non-faulty.

We observe that the fault detection algorithm shows a high accuracy, sensitivity, specificity, and precision. Also, 78% of the faulty nodes are corrected by the fault correction algorithm and 18% of the faulty nodes are classified as the symmetric sensor.
Table I: Confusion matrix of fault detection results, related calculations of fault detection algorithm and fault correction results where $p_f = 0.01$ and $V_{thr} = 3$.

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
<th>Accuracy</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faulty</td>
<td>49</td>
<td>0.9990</td>
<td>0.9800</td>
<td>0.9996</td>
<td>0.9412</td>
</tr>
<tr>
<td>Not Faulty</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Not Faulty</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4946</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Total Number of Corrected Faulty Nodes**: 39
- **Total Number of Symmetric Faulty Nodes**: 9
- **Total Number of Faulty Nodes**: 50
- **Total Number of Falsified Non-Faulty Node**: 2
- **Total Number of Corrected Non-Faulty Node**: 1
- **Total Number of Non-Faulty Node Detected Faulty**: 3

A symmetric sensor node is defined if the fault correction algorithm gives several possible orientations for the corresponding sensor node. This might happen if there is no target in the field of view of the sensor node at that specific time frame and the actually captured image of the sensor is empty. Symmetric sensor nodes can be detected as faulty but they might not be corrected because there are more than one possible orientation angle that shows the minimum pseudo-distance. Whenever a symmetric node covers a target, the symmetry in their field of view will be eliminated and its orientation will be corrected.

In addition, we observe that the fault detection algorithm misclassifies three non-faulty sensor nodes as faulty nodes because the residual areas around the targets might cause the localized target to slightly shift due to the digitization of sensing area as grids. One out of three misclassified sensor nodes is reoriented to its actual orientation. However, the orientation of two misclassified sensor nodes is falsified which makes the falsification ratio as 0.00040.

In the second experiment, we set $p_f = 0.02$. Since 500 sensor nodes are deployed into the sensing field, there are 10 faulty sensor nodes on average, i.e., $N_f = 10$. The voting threshold value is automatically selected as 4, i.e., $V_{thr} = 4$ that is the smallest value for the optimization problem of automatic voting threshold in Eq. 9 as shown in Fig. 9 where lower and upper bound of the voting threshold is 4 and 5, respectively. Table II shows the confusion matrix and related metrics as well as the fault correction results. We observe that the fault detection algorithm shows a high accuracy, sensitivity, specificity, and precision. Also, 71% of the faulty nodes are corrected by the fault correction algorithm and 24% of the faulty nodes are classified as symmetric sensor nodes. The overall falsification ratio is 0.00320.

In the third experiment, we set $p_f = 0.03$. Since 500 sensor nodes are deployed into the sensing field, there are fifteen faulty sensor nodes on average, i.e., $N_f = 15$. The voting threshold value is automatically selected as 4, i.e., $V_{thr} = 4$ that is the smallest value for the optimization problem of automatic voting threshold in Eq. 9 as shown in Fig. 9 where lower and upper bound of the voting threshold is 4 and 5, respectively. Table III shows the confusion matrix and related calculations of fault detection algorithm and fault correction results. We observe that the fault detection algorithm shows a high accuracy, sensitivity, specificity, and precision. Also, 81.3% of the faulty nodes are corrected by the fault correction algorithm and 14% of the faulty nodes are classified as symmetric sensor nodes. The overall falsification ratio is 0.00340.

In the fourth experiment, we set $p_f = 0.04$. Since 500 sensor nodes are deployed into the sensing field, there are twenty faulty sensor nodes on average, i.e., $N_f = 20$. The
Table II: Confusion matrix of fault detection results, related calculations of fault detection algorithm and fault correction results where $p_f = 0.02$ and $V_{thr} = 4$.

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
<th>Accuracy</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faulty</td>
<td>99</td>
<td>0.9958</td>
<td>0.9900</td>
<td>0.9961</td>
<td>0.8376</td>
</tr>
<tr>
<td>Not Faulty</td>
<td>19</td>
<td>0.4881</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>71</th>
<th>Faulty Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Number of Corrected Faulty Nodes</td>
<td></td>
<td>Corrected</td>
</tr>
<tr>
<td>Total Number of Symmetric Faulty Nodes</td>
<td>24</td>
<td>Symmetric</td>
</tr>
<tr>
<td>Total Number of Faulty Nodes</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>Total Number of Falsified Non-Faulty Node</td>
<td>16</td>
<td>Total</td>
</tr>
<tr>
<td>Total Number of Corrected Non-Faulty Node</td>
<td>3</td>
<td>Non-Faulty Nodes</td>
</tr>
<tr>
<td>Total Number of Non-Faulty Node Detected Faulty</td>
<td>19</td>
<td>Falsified</td>
</tr>
</tbody>
</table>

Table III: Confusion matrix of fault detection results, related calculations of fault detection algorithm and fault correction results where $p_f = 0.03$ and $V_{thr} = 4$.

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
<th>Accuracy</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faulty</td>
<td>148</td>
<td>0.9946</td>
<td>0.9867</td>
<td>0.9948</td>
<td>0.8555</td>
</tr>
<tr>
<td>Not Faulty</td>
<td>25</td>
<td>4825</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>122</th>
<th>Faulty Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Number of Corrected Faulty Nodes</td>
<td>122</td>
<td>Corrected</td>
</tr>
<tr>
<td>Total Number of Symmetric Faulty Nodes</td>
<td>21</td>
<td>Symmetric</td>
</tr>
<tr>
<td>Total Number of Faulty Nodes</td>
<td>150</td>
<td></td>
</tr>
<tr>
<td>Total Number of Falsified Non-Faulty Node</td>
<td>17</td>
<td>Total</td>
</tr>
<tr>
<td>Total Number of Corrected Non-Faulty Node</td>
<td>8</td>
<td>Non-Faulty Nodes</td>
</tr>
<tr>
<td>Total Number of Non-Faulty Node Detected Faulty</td>
<td>25</td>
<td>Falsified</td>
</tr>
</tbody>
</table>

voting threshold value is automatically selected as 5, i.e., $V_{thr} = 5$ that is the smallest value for the optimization problem of automatic voting threshold in Eq. 9 as shown in Fig. 9 where lower and upper bound of the voting threshold is 5 and 5, respectively. Table IV shows the confusion matrix and related calculations of the fault detection algorithm as well as the fault correction results. We observe that the fault detection algorithm shows a high accuracy, sensitivity, specificity, and precision. Also, 72.5% of the faulty nodes are corrected by the fault correction algorithm and 21% of the faulty nodes are classified as symmetric sensor nodes. The overall falsification ratio is 0.00720.

In the last experiment, we set $p_f = 0.05$. Since 500 sensor nodes are deployed into the sensing field, there are twenty-five faulty sensor nodes on average, i.e., $N_f = 25$. There is no solution for the optimization problem of automatic threshold selection in Eq. 9 because lower bound of the voting threshold is 6 and upper bound of the voting threshold is 5. In order to select the voting threshold, we relax the K-coverage constraint by changing $\varepsilon_2$ from 0.02 to 0.05 which makes the upper bound of the voting threshold is 7. Therefore, voting threshold value is selected as 6, i.e., $V_{thr} = 6$. Table V shows the confusion matrix and related calculations of the fault detection algorithm as well as the fault correction results. We observe that the fault detection algorithm shows a high accuracy, sensitivity, and specificity rates. However precision rate decreased to 64.5%. Also, 75.2% of the faulty nodes are corrected by the fault correction algorithm and 16.8% of the faulty nodes are classified as symmetric sensor nodes. The overall falsification ratio increase to 0.02320.
Table IV: Confusion matrix of fault detection results, related calculations of fault detection algorithm and fault correction results where $p_f = 0.04$ and $V_{thr} = 5$.

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
<th>Accuracy</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faulty</td>
<td>Faulty</td>
<td>0.9904</td>
<td>0.9850</td>
<td>0.9906</td>
<td>0.8140</td>
</tr>
<tr>
<td>Faulty</td>
<td>Not Faulty</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Not Faulty</td>
<td>197</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Not Faulty</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Not Faulty</td>
<td>45</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Not Faulty</td>
<td>4755</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Total Number of Corrected Faulty Nodes | 145 |
| Total Number of Symmetric Faulty Nodes | 42  |
| Total Number of Faulty Nodes          | 200 |
| Total Number of Falsified Non-Faulty Node | 36  |
| Total Number of Corrected Non-Faulty Node | 9   |
| Total Number of Non-Faulty Node Detected Faulty | 45  |

Table V: Confusion matrix of fault detection results, related calculations of fault detection algorithm and fault correction results where $p_f = 0.05$ and $V_{thr} = 6$.

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
<th>Accuracy</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Faulty</td>
<td>Faulty</td>
<td>0.9716</td>
<td>0.9560</td>
<td>0.9724</td>
<td>0.6459</td>
</tr>
<tr>
<td>Faulty</td>
<td>Not Faulty</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Not Faulty</td>
<td>239</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Not Faulty</td>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Not Faulty</td>
<td>131</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Not Faulty</td>
<td>4619</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Total Number of Corrected Faulty Nodes | 188 |
| Total Number of Symmetric Faulty Nodes | 42  |
| Total Number of Faulty Nodes          | 250 |
| Total Number of Falsified Non-Faulty Node | 116 |
| Total Number of Corrected Non-Faulty Node | 15  |
| Total Number of Not Faulty Node Detected Faulty | 131 |

As an overall comment on these five experiments, we observe that to increase the probability that a sensor is faulty, $p_f$, decreases the accuracy, sensitivity, specificity and precision rates of the fault detection algorithm. Especially, the decrement on the precision rate is obvious. In addition, the performance of the fault correction decreases as $p_f$ increases. The falsification ratio is increased at higher $p_f$ values. The main reason of the decrement on the precision rate and increment on the falsification ratio is the inadequate visual coverage probability. The initial setup of the sensor network allows the voting threshold to be selected at most five which satisfies the coverage-tolerance value $\varepsilon_2$. In order to select the required voting threshold for the fault tolerance algorithm with higher $p_f$, we can relax the $K$-coverage constraint by increasing $\varepsilon_2$ from, for example, 0.02 to 0.05 which means that on average there is 5% chance that a grid point in the sensing field is covered by less than the selected threshold value. Therefore, it might be possible that some non-occupied regions cannot be removed and appear as targets. Thus, the fault detection algorithm detects non-faulty nodes as faulty.

7.2. Experiments using Real Data
Besides simulation, we also conduct a set of real experiments, shown in Fig. 10a, where an 8 by 13 square feet area is surrounded by 42 mobile sensor platforms (MSPs) with onboard processing, wireless communication and imaging capabilities. Four of the MSPs are located at the corners of the experimental area and oriented toward the center of the area at 3 feet in height. The rest of the MSPs are located 1 foot apart from
Fig. 10: (a) Experimental setup with 8 people and 42 cameras, (b) Images captured by 5th, 9th, 22nd, 34th cameras, (c) Corresponding non-occupied 2D visual cones of the images in Fig. 10b and (d) the final certainty map.

each other and oriented in perpendicular angle with the sides of the room at 3 feet in height. In this experimental setup, each square foot area is discretized into 100 grid locations to construct the certainty map, corresponding to a regular grid with a 9 cm resolution. Noted that although 42 MSPs are deployed to form the VSN, not all of them are used to localize the eight targets. For example, if there is no faulty node, the eight targets can be localized by using only 13 sensors. The effect of the number of sensors deployed on the localization performance has been thoroughly evaluated through simulation (See Sec. 7.1). In this experiment, our main purpose is to show the certainty map-based algorithm works in a real-world setup. By deploying a visual sensor network with denser sensor nodes than needed, the factor related to sensing would not be an issue, so that we can focus on fault tolerant collaborative localization performance.

We first evaluate the performance of the proposed algorithm on its capability in localizing targets in a crowded scene, e.g., to identify eight targets within an 8 × 13 square feet area as shown in Fig. 10a. Images are captured by each MSP, as shown in Fig. 10b. Fig. 10c illustrates the local certainty maps generated at the 5th, 9th, 22nd, and 34th MSPs. After integration of local certainty maps, targets are localized in the final certainty map as shown in Fig. 10d.

To study the effect of voting threshold on target localization accuracy, we add Byzantine faults to orientations of 6 sensor nodes to evaluate the localization accuracy of four people, whose true locations are shown in Fig. 11a. In Fig. 11(b-f), the final version of the certainty maps are presented using different threshold values as 1, 2, 3, 4, and 5, respectively. In Fig. 11b, the threshold value is 1 so there is no tolerance for any sensor failure. If one of the MSPs claims non-existence of the target, it is 100% accepted. As a result, the algorithm failed to localize one of the four targets in the scene. However, this missing target can be identified if using a higher voting threshold value, as shown in Fig. 11(c-e), where the threshold is set to 2, 3 and 4, respectively, indicating that to clear the specific area from the certainty map at least 2, 3 or 4 of the MSPs must agree that region should be cleared. Nevertheless, the voting threshold value cannot be selected too high (V_{thr} = 5) as some of the non-occupied areas would then start to be mislabeled as target, as shown in Fig. 11f.

Fig. 11(g-l) shows the robustness of the proposed target localization algorithm against the Byzantine fault. We set the voting threshold value to 4 and add Byzantine fault to different numbers of sensor nodes, as 4, 8, 10, 12, 14, and 18 nodes, respectively. We observe that the proposed method is able to tolerate faulty nodes inputs before its total number reaches 10 (25% of the total number of deployed sensor nodes).
8. CONCLUSION AND DISCUSSION

In this paper, we presented a distributed collaborative target localization algorithm that can reliably detect the position of crowded targets with the existence of a number of faulty sensor nodes, detect the faulty nodes and correct the error in their orientation. We conducted analytical study on the optimal solution to the voting threshold that helps arrive at a tradeoff between energy consumption and degree of redundancy. Camera orientations were estimated using a generative image model in each camera to detect inaccuracy in camera orientations and make the correction. From both simulation and real experimental results, we showed that the proposed collaborative target localization method is effectiveness in providing high localization accuracy as well as satisfactory fault detection and correction performance.

In this paper, it is assumed that visual sensor nodes are randomly deployed into the sensing field for collaborative target localization. Even if sensor nodes are manually deployed into the sensing field or location and orientation of sensor nodes are changed after deployment, the proposed fault detection, correction and tolerance method is still applicable. However, the actual visual coverage will be different than the visual coverage estimation for random deployment. Generally speaking, the visual coverage in the manual deployment scenario is supposed to be better than that of the random deployment. In case of manual sensor deployment, it might be safe to assume that the visual coverage is known. Then, we can automatically select the voting threshold by using the known visual coverage. If the visual coverage in the manual deployment is unknown, we can still use the probability theory as a lower bound for the coverage estimation.
REFERENCES


Collaborative Localization in Visual Sensor Networks


Received March 2012; revised September 2012; accepted December 2012