Target Detection and Counting using a Progressive Certainty Map in Distributed Visual Sensor Networks

Mahmut Karakaya and Hairong Qi
EECS Dept., University of Tennessee
Knoxville, TN 37996-2100
Email: mkarakay@utk.edu, hqi@utk.edu

***Abstract*—Visual sensor networks (VSNs) merge computer vision, image processing and wireless sensor network disciplines to solve problems in multi-camera applications by providing valuable information through distributed sensing and collaborative in-network processing. Collaboration in sensor networks is necessary not only to compensate for the processing, sensing, energy, and bandwidth limitations of each sensor node but also to improve the accuracy and robustness of the sensor network. Collaborative processing in VSNs is more challenging than in conventional scalar sensor networks (SSNs) because of two unique features of cameras, including the extremely higher data rate compared to that of scalar sensors and the directional sensing characteristics with limited field of view. In this paper, we study a challenging computer vision problem, target detection and counting in VSN environment. Traditionally, the problem is solved by counting the number of intersections of the back-projected 2D cones of each target. However, the existence of visual occlusion among targets would generate many false alarms. In this work, instead of resolving the uncertainty about target existence at the intersections, we identify and study the non-occupied areas in the cone and generate the so-called certainty map of non-existence of targets. This way, after fusing inputs from a set of sensor nodes, the unresolved regions on the certainty map would be the location of target. This paper focuses on the design of a light-weight, energy-efficient, and robust solution where not only each camera node transmits a very limited amount of data but that a limited number of camera nodes is used. We propose a dynamic itinerary for certainty map integration where the entire map is progressively clarified from sensor to sensor. When the confidence of the certainty map is satisfied, a geometric counting algorithm is applied to find the estimated number of targets. In the conducted experiments using real data, the results of the proposed distributed and progressive method shows effectiveness in detection accuracy and energy and bandwidth efficiency.**

I. Introduction

Recent advances in CMOS technology make it possible to produce significantly small size and low cost visual sensor platforms with imaging, on-board processing and communication capabilities [1]. A number of such platforms can form a so-called visual sensor network (VSN) that covers a larger surveillance area and is capable of solving computer vision problems through distributed sensing and collaborative in-network processing.

Although potentially powerful, VSNs also present unique challenges that could hinder their practical deployment. For example, VSNs generally require high network bandwidth for data transmission, which cannot be adequately addressed because of the energy constraint and the usage of the low-bandwidth wireless communication. Therefore, *energy-efficient*, *light-weight* solutions need to be pursued [2].

Among all the major processes taken place in a VSN, i.e., sensing, processing, and communication, communication consumes most of the energy. Usually, the energy consumed on sensing and processing can be neglected compared to that on communication. However, when processing images, some computationally expensive algorithms can easily consume as much energy as communication. Therefore, we need to follow two basic guidelines when solving computer vision problems in a distributed environment. First, the sensor data, i.e., the image, should be pre-processed locally to reduce the amount of transmitted data volume [3]. Second, develop simple but effective local processing algorithms to reduce the computational cost.

In this paper, we study a challenging computer vision problem, target detection and counting, using visual sensor networks. The target detection and counting problem in VSNs faces two major challenges [4]. First of all, because of the crowded targets, "visual occlusions" among targets cannot be avoided. Secondly, since visual sensors have limited field of views, and limited computational capacity, visual information obtained by each sensor node is neither sufficient nor accurate. To estimate the number of targets in a crowded environment with the existence of visual occlusion and partial or inaccurate information is the major challenge of this work. In this paper, we present a distributed and progressive solution to estimate the number of crowded targets by performing simple but effective image processing algorithms on each sensor node, transmitting very limited amount of processed data among only a limited number of nodes, and using an efficient data fusion algorithm to obtain the final result.

In traditional target localization algorithms, the intersections of the back-projected 2D cones of the targets are calculated to localize all the individual targets. 2D cones correspond to the possible occupancy information in the visual hulls, also referred to as the *existence information* in this paper. If the cones from different sensors intersect at the same point, it can be considered there is at least one target in that intersection. However, in crowded environments, many

978-1-4244-4620-9/09/$25.00 ©2009 IEEE

"empty" intersections that are not actually occupied by any targets are created because of occlusion, as shown in Fig. 1. Therefore, the existence information in the corresponding intersections is not certain. To remove the uncertainty about the target existence at the intersections and to detect the real locations of the targets among possible intersections have been very challenging problem in computer vision.

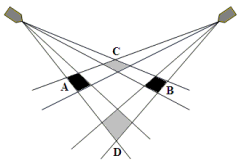

Fig. 1. Four intersections are created from back projection, where intersections A and B are locations of two real targets but empty intersections C and D are also generated due to occlusion.

To solve this problem, we present a technique to localize crowded targets by a collaborative effort from a group of sensor nodes. Instead of resolving the uncertainty about the target existence, we identify and study the non-occluded areas in the visual hull, also referred to as the *non-existence information* in this paper. Since it is certain that there is no target in the corresponding region, we refer to the map generated from this process as the *certainty map*. This essential difference in the proposed method shows its effectiveness in target localization and counting.

In our system, each camera extracts objects of interest from the background, computes the visual hull of objects, determines the non-occupied areas by targets and combines them with other sensor nodes to jointly access the situation in a more accurate way. Certainty map is used as the information exchange unit between sensor nodes when fusing the non-occupied regions to result into a globally consistent belief of non-occupancy, as shown in Fig. 2. The final certainty map, which keeps the real occupied regions, is used to estimate the number and locations of targets. This system is both computational and bandwidth efficient.

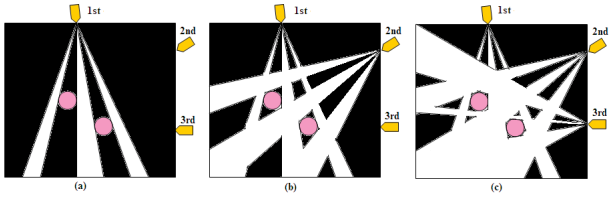

Fig. 2. (a) The certainty map generated by the 1st sensor node (b) Fusion of certainty maps generated by the 1st and the 2nd sensor nodes (c) Fusion of certainty maps generated by the 1st, 2nd, and 3rd sensor nodes.

The paper is organized as follows: Section II describes the background and related works in the visual sensor network. Section III introduces our proposed distributed and progressive target detection and counting method. In Section IV, collaborative processing and itinerary selection techniques are described. The system architecture and experiment results are presented in Section V. Finally, we conclude in Section VI.

II. BACKGROUND AND RELATED WORKS

In the literature, there exists many works related to the larger-scale surveillance and monitoring of activities. Applications like target detection [4], tracking [5] and counting [6] based on wireless networks have been previously investigated.

Single camera approaches [7]–[9] are easy to deploy but they could not be applied efficiently in complex environments such as crowded and occluded target scenarios because they could not provide 3D information by using a single camera view. Therefore, several multi-camera systems are researched and proposed to detect and track multiple objects and compute their accurate 3D locations in a complex environment. In [5], and [10], stereo techniques are used to collect the information from different cameras to localize people. However, these methods are computationally complex as they use color, shape, or texture based methods [11]–[13] to segment, detect, and track multiple targets in a scene by matching each pair of views between camera nodes, and matching is computationally expensive in crowded environments because of the large number of possibilities.

In recent multi-camera systems, occupancy map is popularly used technique to localize objects by using foreground images captured by overhead cameras [14] or using visual hull procedure and motion information of the objects [6]. However, these methods either require a central processing center or the computational complexity is too high to be deployed in VSNs. They use all the available information from every camera to achieve specific tasks like detecting the activities of objects and tracking them without any sensor and itinerary selection. Therefore, they may be useful for a small local group of sensors but not applicable in a large distributed VSN.

III. TARGET COUNTING AND CERTAINTY MAP

In the following sections, we first present the centralized version of the proposed method to estimate the number of crowded targets within the sensing region of a VSN. In the proposed method, each sensor node captures a snapshot of the scene, processes it locally to compute the local certainty map and sends the result to the processing center to collaboratively reach a decision about the number of targets in the scene. This method can be carried out in three main steps: local processing, construction and fusion of the local certainty maps, and target counting.

A. Local Processing

Because of the sheer amount of data generated at each camera node, local processing is needed to provide the necessary information with a much smaller data volume. We also need to keep in mind that local processing cannot be computationally expensive; otherwise, it will consume as much energy as communication. Because of its algorithmic simplicity, the background subtraction algorithm is adopted for object segmentation. The background subtraction algorithm detects foreground objects by subtracting the background from each captured image [15]. By using the background subtracted image, the planer projection of visual hull, the possible regions

occupied by certain object [16], can be generated to derive the non-occupied areas in the field of view (FOV) of each camera node, as shown in Fig. 3. The non-occupied areas are certain areas in which there is no object. On the other hand, there is an uncertainty about the object existence in occupied areas which can appear to be the real object or made by occlusion.

B. Certainty Map

In a certainty map, the environment is divided into uniformly sampled grids where each grid point represents that the corresponding ground space in the surveillance area is *certain* about target non-existence (labeled with one or white) or *uncertain* about target existence (labeled with zero or black). Since certainty map is a binary image consisting of zeros and ones, it can be further compressed by using special coding techniques to decrease the data size to save the communication cost.

Fig. 3 illustrates the steps in constructing the certainty map at a sensor node. For the corresponding scenario in Fig. 3, we obtain the foreground image in Fig. 3(c) by subtracting the background image in Fig. 3(b) from the original image in Fig. 3(a). Each object sweeps a cone in 3D space as shown in Fig. 3(d). To find visual hull of the object, these 3D cones are projected onto a plane parallel to the ground as seen in Fig. 3(e). The non-occupied (clear) areas in the visual hull are thus determined to construct the local certainty map as shown in Fig. 3(f).

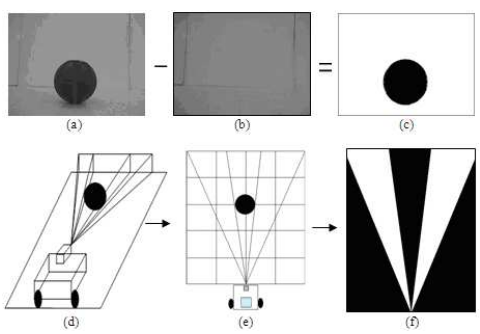

Fig. 3. Illustration of local processing and construction of the certainty map at a visual sensor node. (a) The original captured image by sensor node, (b) The background image, (c) The foreground image, (d) Visual hulls of objects in 3D (e) Projection of 3D cones onto a plane parallel to the ground, and (f) Constructed local certainty map.

In recent techniques [6], [17], after receiving the visual hulls from all sensor nodes, the central processor computes the occupancy map with a standard visual hull procedure which localizes the objects at the intersection of the occluded visual hulls. Although this paper shares the same visual hull idea, it differs from previous works in that we identify the *non-occupied areas* where the non-existence of target is certain and progressively combine these non-occupied areas to localize the objects in a distributed fashion.

In this paper, it is assumed that each sensor node, s_i knows its coordinates in the 2D global coordinate system as (x_i, y_i). Let v_{s_i} denote such non-occupied areas in FOV of the sensor node, s_i. Let $\varphi_{i,j}$ denote the starting angle and $\psi_{i,j}$ the ending angle of the jth non-occupied area in corresponding planer projection of visual hull onto the 2D ground space, $j = 1, \ldots, k$. Therefore, we can describe the total non-occupied areas in a sensor node v_{s_i} as $(x_i, y_i, \varphi_{i,1}, \psi_{i,1}, \ldots, \varphi_{i,k}, \psi_{i,k})$. Let $S = \{s_1, s_2, \ldots, s_n\}$ denote the set of sensor nodes in the network and $f(v_{s_i})$ denote the function which generates the local certainty map by using the non-occluded areas in each sensor node. We then have

$$U(S) = f(v_{s_1}) \cup f(v_{s_2}) \cup \ldots \cup f(v_{s_n})$$

where *U(S)* denotes the union formed by all the local certainty maps in *S*. Targets are located in the complement of *U(S)*, the remaining regions from union of the local certainty maps.

By using the visual hull, non-occupied areas in the FOV of each camera node are cleared from the certainty map, where white pixels indicate the cleared areas, because there is 100% confidence about no target in that region. On the other hand, black pixels in the certainty map indicate that there are still uncertainties regarding target existence which can be one of the three possibilities. First, the region is occupied by target; second it is outside of the FOV of the camera; and third, occlusion.

The size of the total uncertain region in global certainty map monotonically decreases, thus guarantees the convergence of the global certainty map. The idea is that the uncertainty 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.

C. Target Counting Algorithm

The final certainty map consists of sets of small regions with potential locations of targets, referred to as the phantoms. Phantoms are the remaining areas in the certainty map that could not be cleared by any sensor nodes. In literature, there are different approaches to count the objects in each phantom. In [6], the area of each phantom is divided by object size regardless of object shape. However, because of the occlusion of objects in crowded environments, there will be some residual areas in the phantom that cannot be clarified by any cameras and make the size of the phantoms bigger than the object size. Therefore, it is necessary to consider the shape of the object and residual areas when to compute the number of objects in each phantom. If there is no occlusion and there is infinite number of cameras, the size of the phantom converges to the actual size of the object. However, in crowded targets, it is not possible to prevent occlusion and the residual areas in the phantom. In Fig. 4, the possible residual areas are shown for cylindrical objects. The planer projections of the cylindrical objects are discs in the ground space. The smallest residual areas occurs when the objects touch each other assuming there is no overlapping between the objects.

In Fig. 4(a), the residual area around the object converges to zero if there is infinite number of cameras. In Fig. 4(b) and 4(c), the smallest residual areas around the objects, shown for two and three objects cases, are the areas between the objects.

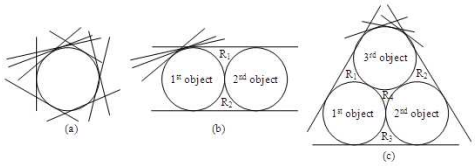

Fig. 4. Three different scenarios of residual areas for cylindrical objects.

These residual areas, R_i, cannot be prevented so they have to be taken into consideration in the calculation of minimum size of the total area for different number of object cases in a phantom. They are calculated by adding the corresponding minimum residual areas to total area of objects, such as two and three object cases shown in Fig. 4. The number of objects can be found by comparison between the area of each phantom and the pre-calculated minimum size of the total area for different number of object cases.

IV. COLLABORATIVE PROCESSING AND ITINERARY SELECTION

In order to fuse or integrate the certainty map, two mechanisms can be adopted, centralized and progressive. In the centralized approach, each camera node sends its local certainty map to a processing center for information fusion. In the progressive approach, the certainty map is propagated through the network. At each camera node, the certainty map is refined by integrating with local certainty map to add more certainty to the previous map. In other words, the local information helps clarify uncertain areas as this fusion process prolongs.

If there is no energy or bandwidth limitation, the information from all the available cameras can be used to compute the best possible certainty map since the residual areas in the certainty map monotonically decreases as the number of cameras increases. However, the energy resource is very limited in VSNs and increasing the number of used camera nodes decreases the network lifetime. In the next section, we will compare the performance between centralized and progressive integration of the certainty map through experiments. Here, we first tackle a critical problem in realizing the progressive integration, i.e., how to determine the itinerary that the certainty map propagates. This problem is two-fold: the itinerary of propagation and when to stop the integration.

A. Itinerary Selection

The visual data provided by physically close camera nodes might be highly correlated. If removal of a sensor node from the itinerary does not shrink the uncertainty region of the certainty map too much, the information provided by that camera node is redundant. In order to save energy, transmission of the certainty map between these sensor nodes must be avoided. To find the minimum subset of camera nodes to construct the accurate certainty map is the main goal of itinerary selection.

In literature, different approaches for sensor selection have been proposed for different scenarios. In [18], clustering the parallel and perpendicular cameras to the vectors connecting the objects is a good approach for two-object case. However, how to find the vectors connecting the objects without prior knowledge of objects location or how to update the vectors for moving objects is another problem.

In this paper, we study three different itinerary selection criteria as fix, random and dynamic itinerary to fuse the certainty map.

1) Fix Itinerary: To overcome the problems identified above, the cameras can be clustered by using their orientation angles regardless of the target location. The best choice for the next camera is to select the camera which has the best complimentary field of view of the previous camera. The 90 degree difference in the camera orientation with the previous camera gives the best complimentary FOV. Therefore, to determine the fix itinerary, cameras which have the 90 degree difference in the camera orientation with others are clustered into small subsets.

2) Random Itinerary: Instead of using the fix itinerary, we also implement the itinerary in which cameras are randomly selected to integrate the certainty map.

3) Dynamic Itinerary: Neither fix nor random itinerary determines the route of certainty map migration based on the *content* of the map. Therefore, it is not guaranteed that the next integration would clear the most uncertainty region leading to the shortest route.

To overcome this problem, we propose a dynamic itinerary selection method, where in each iteration sensor nodes compute their local certainty maps, calculate the size of area that can be cleared from the current certainty map and broadcast it through the network. The sensor node which has the largest clarification area gets the priority over the others to integrate its local certainty map with current certainty map and broadcasts the updated current certainty map to the network to allow other sensor nodes to recalculate the amount of the additional clearance on the current certainty map. This procedure repeats until confidence test is satisfied.

The method is described in Algorithm 1, where *S* denotes the set of sensor nodes and $|f(v_{s_i})|$ denotes the total size of area in local certainty map that can be cleared from the current certainty map by sensor node s_i. δ is the threshold to determine that if the sensor node holds adequate additional clarification information to remain in set *S*. If the clarification information, $|f(v_{s_i})|$, is below the threshold δ, the sensor node, s_i, removes itself from the set *S*.

B. Confidence Test

No matter which itinerary approach is adopted, a common question each faces is when to stop the integration process. Because of energy constraints, we have to stop transferring the certainty map in the VSN when certain criteria can be satisfied.

Algorithm 1 : Dynamic Itinerary

Initialize:
Certainty map, $CM = \emptyset$, is fully occupied.
$S = \{s_1, s_2, \ldots, s_n\}$ {set of sensor nodes}
repeat
  for each s_i in S **do**
    Compute clarification amount, $|f(v_{s_i})|$, by using the current *CM*
    if $|f(v_{s_i})| < \delta$ **then**
      $S = S - s_i$ {remove redundant sensor}
    else
      Broadcast $|f(v_{s_i})|$ to the network to determine the highest one.
    end if
  end for
  if $s_j = max(|f(v_S)|)$ **then**
    $S = S - s_j$ {remove the winning sensor}
    Update *CM* by s_j, the jth sensor node.
    Broadcast updated *CM* to the remaining nodes within S.
  end if
until $S = \emptyset$

In the fix or random itinerary, the certainty map is transmitted to the next sensor node without prior knowledge about how much it can clarify from the certainty map. If the sensor node does not contribute to certainty more than a defined threshold, it raises a stopping flag. If the number of consecutive stopping flags reaches a certain percentage of the total number of sensor nodes, transmission is stopped and the final certainty map is declared. This method has an apparent drawback. If the certainty map travels among sensor nodes with redundant information at some portion of the itinerary, the procedure will be stopped even though other sensors can still contribute to the map.

In dynamic itinerary, it is not necessary to count the number of flags because in each iteration, each sensor node propagates its clearance amount to the network and the additional clearance amount is monotonically decreasing. If none of the sensor nodes has adequate additional clearance area on the certainty map, we can stop transmitting the certainty map immediately.

C. Voting

The other critical question about the proposed algorithm is the accuracy and robustness. Single camera node, which gives inaccurate information about the location of the targets, negatively impacts the performance and sometimes causes failure of the target detection algorithm. To obtain more accurate and robust results, certain degree of redundancy is necessary to tolerate the inaccurate information and failure of some camera nodes.

Voting is one of the most commonly used multiple sensor fusion techniques to combine individual sensor results [19]. In simple voting strategy, each sensor node has equal importance to contribute to the voting result. In this paper, we propose to use the voting approach to tolerate potential inaccurate information from sensor nodes.

In the voting approach, if a sensor node believes that there is no object in the specific location of the certainty map, it increments the certainty value of that specific location. Therefore, instead of a binary certainty map with 1 indicating 100% certainty of non-existence of targets, as shown in Fig. 5(a), the voting approach generates a gray-scale certainty map, shown in Fig. 5(b) with non-zero regions indicating certain degree of belief of the non-existence of targets, and the higher the number of votes, the more certain it is.

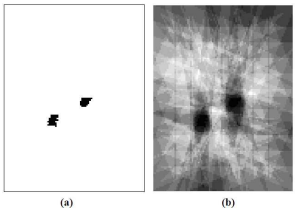

Fig. 5. (a) Binary certainty map and (b) Gray-scale certainty map.

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 for 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 certainty map, at least two sensors must declare the clearness of that region. Higher threshold value requires more sensor nodes to reach a consensus. In next section, we have designed an experiment to show how the threshold value affects the performance of the target counting algorithm.

V. EXPERIMENTAL RESULTS

In our experimental setup shown in Fig. 6, two static objects are located in a 9 by 12 feet square area surrounded by 38 mobile sensor platforms (MSPs) with onboard processing, wireless communication and imaging capabilities. The objects are 1 foot in height and cameras are located at 6 inches in height. Four of the MSPs are located at the corners of the experimental area and oriented toward the center of the area. The rest of the MSPs are located 1 foot apart and oriented to the room with perpendicular angle with the sides of the area. In this experimental setup, each foot square area is discretized into 100 grid locations to construct the certainty map, corresponding to a regular grid with a 9 cm resolution.

Images are captured by each MSP with different field of views as shown in Fig. 7(top). Background subtraction is first performed to obtain the foreground objects shown in Fig. 7(middle). The visual hulls of the non-occupied areas

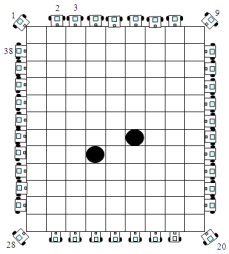

Fig. 6. Experimental setup with 2 objects and 38 cameras.

are computed by using the planer projection, as shown in Fig. 7(bottom).

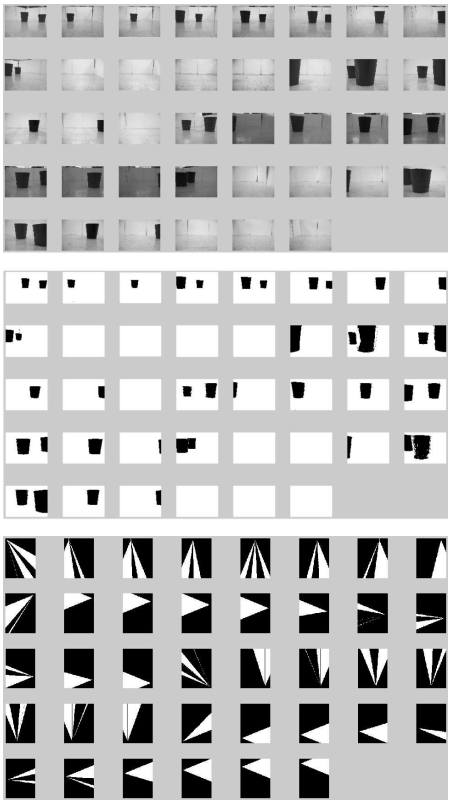

Fig. 7. (top) Images captured by cameras 1 to 38. (middle) Foreground images from cameras 1 to 38. (bottom) Non-occluded visual hulls of each camera.

We conduct two sets of experiments to study the effect of choosing different itinerary schemes and that of different voting threshold on the performance of the detection. The following values are used for a couple of parameters introduced in the algorithm:

- δ, the threshold to determine if a sensor node holds adequate additional clarification information, is set to be 10% of the object size. When belowing this threshold, a stopping flag is raised.
- The number of consecutive stopping flags is set to be 10% of the total number of sensors in the network, i.e., 4. If the number of consecutive flags is 4, then the itinerary is stopped and a final certainty map is generated.

A. Effect of Different Itineraries

Three different itineraries, fix, random and dynamic, are tested to show the effect of the itinerary selection on the performance of the algorithm.

Fig. 8(top) shows the intermediate certainty maps that is progressively improved from node to node following a fixed itinerary, discussed in Sec. IV-A1. After transmitting the certainty map to the 19th MSP, the additional clearance from the certainty map is less than the threshold, δ, therefore, the corresponding MSP raises a stopping flag to be carried to others about its inadequate contribution to the certainty map. When there have been four consecutive flags raised for robustness purpose, the itinerary would stop in order to save energy. In the fix itinerary, the itinerary stopped at the 22nd MSP.

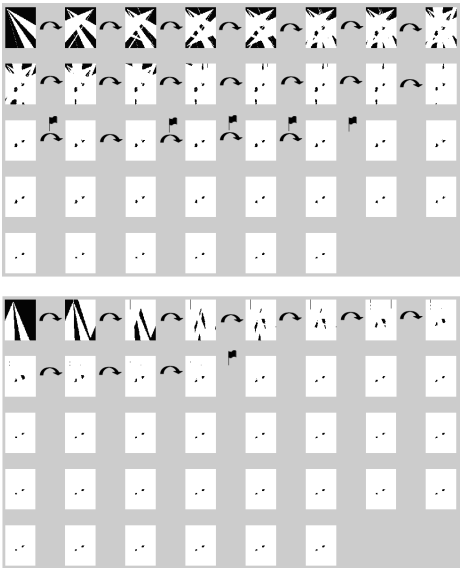

Fig. 8. (top) The clarification of the certainty map using the fix itinerary. (bottom) The clarification of the certainty map by using the dynamic itinerary.

For dynamic itinerary, since the amount of cleared areas monotonically decreases from sensor to sensor, we do not have to wait until getting four consecutive flags to stop the integration. The itinerary is stopped as soon as the size of the additional clearance region is less than the threshold, δ. In Fig. 8(bottom), the progress of the certainty map in dynamic itinerary is shown. The itinerary is stopped at the 12th sensor node. We observe that the dynamic itinerary scheme generates the best route to clarify the certainty map by using the least amount of sensor nodes.

The comparison of fix itinerary, average result of the 1000 different random itineraries and dynamic itinerary is shown in Fig. 9 with the number of uncertain pixels at each iteration (or sensor stop) as a performance metric. We observe that the dynamic itinerary shows the best performance by clarifying the certainty map at the fastest rate. In addition, random selection of the itinerary clarifies the certainty map faster than fix itinerary and gives better result in general.

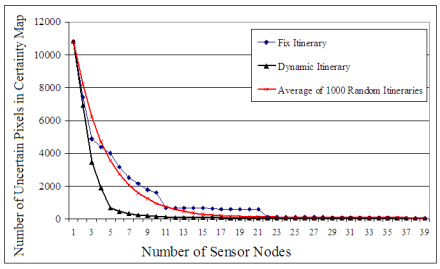

Fig. 9. Total number of uncertain pixels in certainty map for different itinerary selection.

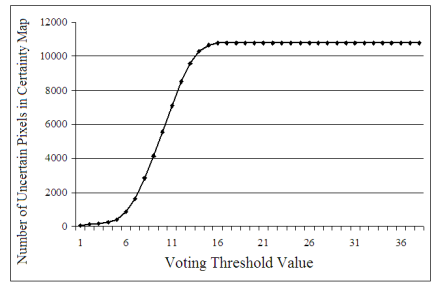

Fig. 11. Total area in final certainty map for different voting threshold values.

B. Effect of the Voting Threshold

Voting is adopted to improve the accuracy and robustness of the target counting algorithm. In Fig. 10, the final version of the certainty map is shown by using different threshold values as 1, 2, 3, 4, 5, and 6, respectively. In Fig. 10(a), the threshold value is 1 so there is no tolerance for any sensor failure. If one of the MSP claims a pixel as non-existence of the object, it is 100% accepted. In Fig. 10(b), the threshold value is 2 and to clear the specific area from certainty map at least two MSPs must agree that region should be cleared.

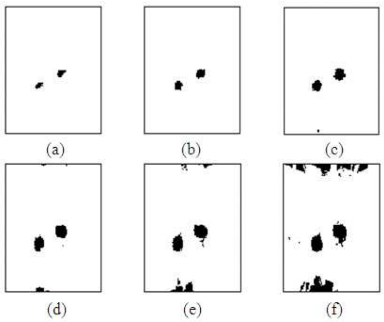

Fig. 10. Final version of the certainty map using voting with different threshold values.

The total numbers of dark pixels in the final certainty map, which correspond to the uncertain areas of target existence, for different voting threshold values are shown in Fig. 11. If the selected threshold is small, the uncertain areas are very few. When increasing the voting threshold value it also increases the total numbers of uncertain areas in the certainty map and begins to introduce the non-occupied areas as object as shown in Fig. 10(d-f).

As shown in Fig. 11, the total number of uncertain pixels exponentially increases if the voting threshold value increases. It reaches maximum at threshold value 15 because there is no pixel that can be clarified by 15 cameras at the same time. The density of cameras is not enough to clarify the non-occupied areas for that threshold value. Therefore, the selection of the threshold value depends on the required tolarance as well as the sensor density.

In most visual camera applications, the cameras are assumed to be oriented accurately in the scene. There is no tolerance to inaccurately mounted cameras which have different orientations than assumed because of external effects or internal hardware problems or inaccurate installation. By using voting mechanism this kind of inaccuracy is automatically tolerated.

In the following experiment, we add Gaussian random noise with zero mean and different standard deviation values to all the camera orientations. In Fig. 12, the total number of uncertain pixels in the final version of the certainty map is shown for different standard deviation and for different voting threshold values.

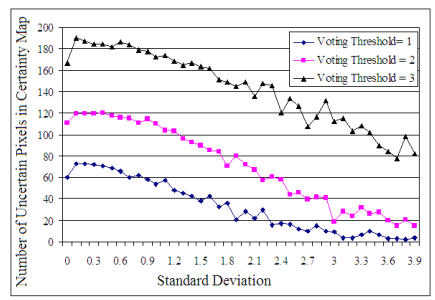

Fig. 12. Total area in final certainty map for Gaussian noise with zero mean and different standard deviation values added to orientation of the visual sensors for different voting threshold values.

To add more Gaussian noise by increasing the standard deviation amount reduces the system performance. Higher voting threshold values can be selected to avoid the performance drop. However, we cannot select the voting threshold too high because it depends on the density of the sensor nodes as shown in Figs. 10 and 11. We observe in Fig. 12 that the performance of the system with voting threshold three is better than voting threshold one and two because the inaccuracy in the camera orientation is tolerated by using the voting to fuse the certainty maps generated by different sensor nodes.

C. Performance Comparison through Analytical Study

The advantage of using dynamic itinerary in distributed processing 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. To provide an analytical comparison between different schemes in reaching a final certainty map, we compare the number of bits transmitted within the network which is a good indicator of energy consumption and bandwidth usage.

Assume there are K sensors in the network, among which k_d sensors are involved in the target counting process if using dynamic itinerary, and k_o sensors involved if using fixed or random itinerary. Assume there are n targets within the surveillance area of the K sensors, the worst case scenario for data transmission is that there are $n+1$ non-occupied regions in local certainty map which can be expressed as a data vector, $v_{s_i} = (x_i, y_i, \varphi_{i,1}, \psi_{i,1}, \ldots, , \varphi_{i,n+1}, \psi_{i,n+1})$. The size of the data vector is 32(4+2n) bits per transmission, assuming each value is represented as 32-bit floating point. In dynamic itinerary, the total size of area in local certainty map that can be cleared from the current certainty map by sensor node s_i, $|f(v_{s_i})|$, is also required to propagate to the network with a size of 16*t* bits where *t* is the number of sensor nodes remaining in *S* for each iteration and we assume $|f(v_{s_i})|$ can be represented as a short integer which is 16 bits. *t* monotonically decreases when redundant and winner sensor nodes are removed from *S* in each iteration. The total amount of data transmitted in dynamic itinerary is less than other itineraries if the following equation holds:

$$32(4+2n)k_o > 32(4+2n)k_d + 16t$$

$$t \leq \ K * k_d.$$

Thus,

$$\frac{k_o}{k_d} > 1 + \frac{K}{8+4n}$$

As shown in Fig. 13, when the ratio of the number of sensor nodes involved in reaching the final certainty map in dynamic itineraries and other itineraries are greater than certain number, dynamic itineraries require less amount of data transmission indicating less energy consumption and less bandwidth occupation than fix and random itinerary used in distributed processing.

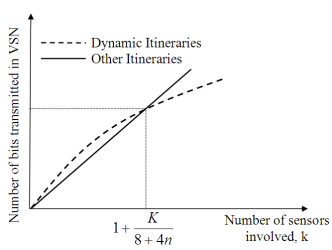

Fig. 13. Comparison of the number of bits transmitted in VSN by using dynamic itinerary and other itineraries.

VI. Conclusion

In this paper, we presented an algorithm that can reliably detect the position of crowded targets and count them in distributed wireless VSNs under certain energy and bandwidth constraints. To achieve our goal, we designed a light-weight, energy-efficient, and robust solution where not only each camera node transmits a very limited amount of data but that a limited number of camera nodes is used to estimate the number of targets. We identified and studied the non-occupied areas in the back-projected 2D cones, generated the certainty map of non-existence of targets and defined a dynamic itinerary for certainty map integration where the entire map is progressively clarified from sensor to sensor until the confidence of the certainty map is satisfied. When the confidence of the certainty map is satisfied, a geometric counting algorithm finds the estimated number of targets. In the conducted experiments using real data, the results of the proposed distributed and progressive method showed effectiveness in detection accuracy and energy and bandwidth efficiency.

References

[1] S. Lim and A. El-Gamal, "Integrating image capture and processing-beyond single chip digital camera," *In Proceedings of the SPIE Electronic Imaging*, vol. 4306, Jan. 2001.

[2] K. Obraczka, R. Manduchi, and J. Garcia-Luna-Aveces, "Managing the information flow in visual sensor networks," *The 5th International Symposium on Wireless Personal Multimedia Communications.*, vol. 3, pp. 1177–1181, Oct. 2002.

[3] J. M. Kahn, R. H. Katz, and K. S. J. Pister, "Next century challenges: Mobile networking for smart dust," *Human Motion, 2000. Proceedings. Workshop on*, pp. 271–278, 1999.

[4] C. Qian and H. Qi, "A distributed solution to detect targets in crowds using visual sensor networks," *Second ACM/IEEE International Conference on Distributed Smart Cameras.*, pp. 1–10, Sept. 2008.

[5] J. Krumm, S. Harris, B. Meyers, B. Brumitt, M. Hale, and S. Shafer, "Multi-camera multi-person tracking for easyliving," *Third IEEE International Workshop on Visual Surveillance.*, pp. 3–10, 2000.

[6] D. Yang, H. Gonzalez-Banos, and L. Guibas, "Counting people in crowds with a real-time network of simple image sensors," *Ninth IEEE International Conference on Computer Vision, Proceedings.*, vol. 1, pp. 122–129, Oct. 2003.

[7] I. Haritaoglu, D. Harwood, and L. Davis, "W4: Who? when? where? what? a real time system for detecting and tracking people," *Third IEEE International Conference on Automatic Face and Gesture Recognition. Proceedings.*, pp. 222–227, Apr 1998.

[8] M. Han, W. Xu, H. Tao, and Y. Gong, "An algorithm for multiple object trajectory tracking," *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 1, pp. 864–871, June-2 July 2004.

[9] M. Isard and J. MacCormick, "Bramble: a bayesian multiple-blob tracker," *Eighth IEEE International Conference on Computer Vision, ICCV 2001. Proceedings.*, vol. 2, pp. 34–41, 2001.

[10] T. Sogo, H. Ishiguro, and M. Trivedi, "Real-time target localization and tracking by n-ocular stereo," *Omnidirectional Vision, 2000. Proceedings. IEEE Workshop on*, pp. 153–160, 2000.

[11] R. Collins, "A space-sweep approach to true multi-image matching," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Proceedings CVPR '96.*, pp. 358–363, Jun 1996.

[12] A. Mittal and L. Davis, "M2tracker: A multi-view approach to segmenting and tracking people in a cluttered scene," *Int'l J. Computer Vision*, vol. 51, no. 3, pp. 189–203, 2003.

[13] I. Haritaoglu, D. Harwood, and L. Davis, "Hydra: multiple people detection and tracking using silhouettes," *Image Analysis and Processing. Proceedings. International Conference on*, pp. 280–285, 1999.

[14] E. Halvorson and R. Parr, "Counting objects with a combination of horizontal overhead sensors," *Technical Report, Department of Computer Science Duke University*, 2007.

[15] R. Qian and M. Sezan, "Video background replacement without a blue screen," *International Conference on Image Processing. Proceedings, ICIP 99.*, vol. 4, pp. 143–146, 1999.

[16] W. Matusik, C. Buehler, and L. McMillan, "Polyhedral visual hulls for real-time rendering," *Eurographics Workshop on Rendering*, 2001.

[17] D. Beymer, "Person counting using stereo," *Human Motion, 2000. Proceedings. Workshop on*, pp. 127–133, 2000.

[18] D. Yang, J. Shin, A. Ercan, and L. Guibas, "Sensor tasking for occupancy reasoning in a network of cameras," *In Proceedings of Workshop Broadband Advance Sensor Network*, Oct 2004.

[19] L. Klein, "A boolean algebra approach to multiple sensor voting fusion," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 29, no. 2, pp. 317–327, Apr 1993.