Collaborative Self-Localization Techniques for Wireless Image Sensor Networks

Huang Lee and Hamid Aghajan
Wireless Sensor Networks Lab
Department of Electrical Engineering
Stanford University, Stanford, CA 94305
Email: huanglee@stanford.edu and aghajan@stanford.edu

Abstract—We introduce a novel localization technique that can jointly estimate the locations of a moving target and the sensor nodes in a wireless image sensor network. The proposed method is based on in-node image processing and can be implemented in a decentralized or clustered fashion. In our approach, two image sensors are used to define a relative coordinate system. In order to synchronize the observations, the node defined as the origin broadcasts packets that trigger image capture at other nodes. In the decentralized version of the technique, each one of the two reference nodes broadcasts its image plane position of the moving target at a few time instances. Each of the other nodes in the network that can detect the target in its image plane upon receiving a number of triggering broadcasts, calculates its own relative coordinates and orientation as well as the coordinates of the observed target. In the clustered version of the proposed technique, observations gathered by the nodes within a neighborhood cluster are sent to a cluster-head, which can be the reference node at the origin. The cluster-head combines the data and calculates the coordinates of the target and all the nodes that contributed observations. Experimental results are provided to verify the performance of both versions of the proposed algorithm.

I. INTRODUCTION

Wireless sensor networks are an emerging technology for monitoring the physical world [1] [2]. In a sensor network application, large numbers of tiny sensor nodes may be deployed and collaborate to gather data from the environment. Each node is equipped with a sensing modality, such as an image sensor, and has the capability to communicate over wireless channels. Such wireless sensor networks find applications in smart environments, surveillance, environmental monitoring, wildlife observation and tracking, and others.

Most applications in sensor networks rely on the knowledge of sensor positions. However, manual location entry results in high deployment cost and is unrealistic in large networks. Node localization therefore is a fundamental problem in sensor networks [3] [4] [5] [6]. Recently, research on image sensor networks has received large interest; however, only limited study of the localization problem has been reported for these networks [7] [8]. In image sensor networks, each node may be equipped with a low-resolution camera because of the complexity and cost limitations. Furthermore, calibration in multi-camera systems is impractical in large networks [9]. Hence, a localization algorithm that can utilize low-resolution images and requires low computational power and minor sensor calibration is very much desired.

We assume that there is a moving target in the network and want to jointly estimate the locations and orientations of sensor nodes and the position of the moving target. In the proposed approach, we select two image sensors as reference nodes to define a relative coordinate system. In order to synchronize the observations of all image sensors, a low complexity synchronization protocol is implemented in the network. The synchronized sensor nodes acquire images simultaneously and locate the moving target in their image planes. Based on the requirements, these sensor nodes can exchange information in different ways. In the decentralized version of the proposed technique, the two reference nodes broadcast their observations so that each of the other sensors can estimate its location and orientation. Each node only uses its own observations and the information sent by the reference nodes. In the cluster-based version, all observed data are sent to the reference node at the origin (cluster-head), which then calculates the coordinates of the target and locations and orientations of all the other sensor nodes.

II. SYSTEM MODEL AND ALGORITHMS

We consider a two dimensional localization problem as shown in Fig. 1. Two reference nodes are used to define the origin and the unit length. Our goal is to find the positions of the other sensors and the moving target related to the reference nodes, and the orientations of all sensor nodes. We assume that both the reference nodes and the sensor nodes requiring localization can observe the moving target simultaneously. This can be achieved by having the reference node at the origin broadcast a synchronization signal. This synchronization concept is similar to [10], and is feasible for slowly moving targets.

After constructing the relative coordinate system, we define the sensor orientation $\theta_k$ for sensor node $k$ as shown in Fig. 1. Each sensor node can use the pinhole model to estimate the angular offset $\phi$ to the target relative to its orientation. The pinhole camera model is shown in Fig. 2, and the relationship between the angular bearing to the target and camera’s physical parameters can be found to be

$$\phi = \tan^{-1} \left( \frac{2d}{D} \tan \left( \frac{\Psi}{2} \right) \right)$$  \hspace{1cm} (1)
where $D$ is the horizontal resolution in pixels, $d$ is the distance from the center of the image plane in pixels, and $Ψ$ is the angular span of the field of view of the image sensor.

An observation is made at node $k$ whenever this node and the two reference nodes acquire frames at the same time instance. The observed angle of node $k$ for the $n$th observation is denoted by $φ_k^n$. For each observation, we have an unknown distance between each sensor and the target, which is denoted by $λ_k^n$ for node $k$ and the $n$th observation. From each observation, two independent equations can be obtained by writing algebraic equations relating the positions of the nodes, the target, and each of the two reference nodes. So for $N$ observations we have

$$le^{jδ} = λ_0^n e^{jθ_0} e^{jφ_0^n} - λ_0^n e^{jθ_0} e^{jφ_0^n} - 1, n = 0, \ldots, N - 1$$

$$le^{jδ} = λ_1^n e^{jθ_1} e^{jφ_1^n} - λ_2^n e^{jθ_2} e^{jφ_1^n} + 1, n = 0, \ldots, N - 1$$

where $le^{jδ}$ is the coordinate of node 2 in polar coordinates and is unknown.

For $N$ observations, we have a total of $3N + 3 + 2 (λ_0^n, λ_1^n, λ_2^n, θ_0, θ_1, θ_2, l, δ)$ unknown parameters and $4N$ equations which come from the real and imaginary parts of (2) and (3). Hence, we need at least $N \geq 5$ observations to solve for all unknown parameters. To solve for these nonlinear equations, we can apply the Gauss-Newton method [11] [12]. Application of iterative algorithms for nonlinear equations in wireless sensor network problems has been reported in [13] [14].

**A. Gauss-Newton Algorithm**

Before applying the Gauss-Newton algorithm, we re-arrange equations (2) and (3) to cancel the variables in $le^{jδ}$ and obtain $2N - 1$ new equations

$$0 = λ_0^n e^{jθ_1} e^{jφ_1^n} - λ_0^n e^{jθ_1} e^{jφ_0^n} - 1, n = 0, \ldots, N - 1$$

$$0 = λ_1^n e^{jθ_1} e^{jφ_1^n} - λ_0^n e^{jθ_2} e^{jφ_0^n} - 1, n = 0, \ldots, N - 1$$

$$0 = λ_2^n e^{jθ_2} e^{jφ_1^n} + 1, n = 0, \ldots, N - 2$$

Now, the number of unknown parameters is $3N + 3 (λ_0^n, λ_1^n, λ_2^n, θ_0, θ_1, θ_2)$. To apply the Gauss-Newton method, we assign $r_1, r_2, \ldots, r_{4N - 2}$ to the real and imaginary parts of (4) and (5):

$$r_{4n+1} = λ_0^n \cos (θ_0 + φ_0^n) - λ_0^n \cos (θ_1 + φ_1^n) - 1, n = 0, \ldots, N - 1$$

$$r_{4n+2} = λ_0^n \sin (θ_0 + φ_0^n) - λ_0^n \sin (θ_1 + φ_1^n), n = 0, \ldots, N - 1$$

$$r_{4n+3} = λ_1^n \cos (θ_1 + φ_1^n) - λ_0^n \cos (θ_0 + φ_0^n) - \lambda_0^n \cos (θ_2 + φ_0^n) + \lambda_2^n \cos (θ_0 + φ_0^n) + 1, n = 0, \ldots, N - 2$$

$$r_{4n+4} = λ_1^n \sin (θ_1 + φ_1^n) - λ_0^n \sin (θ_0 + φ_0^n) - \lambda_0^n \sin (θ_2 + φ_0^n) + \lambda_2^n \sin (θ_0 + φ_0^n), n = 0, \ldots, N - 2$$

and define the vector $\mathbf{x} = [r_1, r_2, \ldots, r_{4N - 2}]^T$ where the superscript $(\cdot)^T$ denotes the transpose. We also define a $(3N + 3) \times 1$ vector to specify the unknown parameters

$$\mathbf{x} = [θ_0, θ_1, θ_2, λ_0^n, \ldots, λ_2^n, \ldots, λ_2^{N - 1}]^T.$$ (11)

The Jacobian matrix used in the Gauss-Newton algorithm is given by

$$\mathbf{J} = \begin{bmatrix}
\frac{∂r_1}{∂x_1} & \frac{∂r_1}{∂x_2} & \cdots & \frac{∂r_1}{∂x_{3N + 3}} \\
\frac{∂r_2}{∂x_1} & \frac{∂r_2}{∂x_2} & \cdots & \frac{∂r_2}{∂x_{3N + 3}} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{∂r_{4N - 2}}{∂x_1} & \frac{∂r_{4N - 2}}{∂x_2} & \cdots & \frac{∂r_{4N - 2}}{∂x_{3N + 3}}
\end{bmatrix} \ (4N - 2) \times (3N + 3)$$

where $x_i$ is the $i$th element in vector $\mathbf{x}$. Finally, the update rule for the Gauss-Newton method is given by

$$\mathbf{x}_{i+1} = \mathbf{x}_i - (\mathbf{J}^T (\mathbf{x}_i) \mathbf{J} (\mathbf{x}_i))^{-1} \mathbf{J}^T (\mathbf{x}_i) \mathbf{r} (\mathbf{x}_i)$$

where $i$ indicates the $i$th iteration. The complexity of computing the inverse matrix is around $O (N^3)$, and an inversion operation is required in each iteration. If the computation power is limited, steepest descent methods [12] can be applied to reduce the complexity, and (13) can be simplified to

$$\mathbf{x}_{i+1} = \mathbf{x}_i - α^i \mathbf{J}^T (\mathbf{x}_i) \mathbf{r} (\mathbf{x}_i)$$

where $α^i$ is a positive step size.
**B. Special Case and Linearized Algorithm**

When the orientations \((\theta_0, \theta_1)\) of the reference nodes are known, observations made at the two reference nodes are sufficient to locate the target by triangulation. Therefore, the algorithm can be decomposed into two stages. In the first stage, we use the observations from the reference nodes to obtain the target coordinates. Given \(\phi_0^0, \phi_1^0\) from image planes at the reference nodes, the distances between the target and the reference nodes at the \(n\)th observation are estimated by

\[
\hat{\lambda}_0^n = \frac{\sin (\phi_0^0 + \theta_1)}{\sin (\phi_1^0 - \phi_0^0 + \theta_1 - \theta_0)}
\]

\[
\hat{\lambda}_1^n = \frac{\sin (\phi_1^0 - \phi_0^0 + \theta_1 - \theta_0)}{\sin (\phi_0^0 - \phi_0^0 + \theta_1 - \theta_0)}
\]

The target coordinates are then estimated by

\[
\hat{\lambda}_0^n e^{j(\phi_0^0 + \theta_0)} = \frac{\sin (\phi_0^0 + \theta_1)}{\sin (\phi_1^0 - \phi_0^0 + \theta_1 - \theta_0)} e^{j(\phi_0^0 + \theta_0)}. \tag{17}
\]

After obtaining the target coordinates, we can linearize the equations. Assume that the estimated target positions at time instances \(n\) and \(m\) are given by \(p_n\) and \(p_m\). We have

\[
e^{j\theta_2} (p_n - p_m) - \hat{\lambda}_2^n e^{j\phi_2^n} + \hat{\lambda}_2^m e^{j\phi_2^m} = 0. \tag{19}
\]

The unknown variables in the equation are \(\theta_2\), \(\lambda_2^n\), and \(\lambda_2^m\). Once we obtain these values, we can find the sensor coordinates \(e^{j\theta_2}\). So, the whole algorithm is linearized in the special case.

On the other hand, we can still apply the Gauss-Newton method to solve this special case. The procedure of constructing \(\pi, \tau\), and Jacobian matrix \(J\) is similar to the general case, but now we have a smaller matrix to invert, and the complexity is still \(O(N^3)\) times the number of iterations.

**C. Decentralized Scheme and Cluster-based Scheme**

The proposed algorithm can be implemented as a decentralized or cluster-based scheme. In the decentralized version, each sensor node estimates its own location and orientation and the target’s position by exploiting the relationship between its observations and the broadcast information. The algorithm flowchart for this case is shown in Fig. 3 (a), and the network data flow is shown in Fig. 4 (a). In the cluster-based version, each sensor node sends its observations to the reference node defining the origin (cluster-head) as shown in Fig. 4 (b), and the cluster-head solves for the coordinates of all the nodes and the target. The flowchart of the cluster-based scheme is shown in Fig. 3 (b).

In the cluster-based scheme, there are observed information contributed by several nodes. Therefore, we can study the relationship between the number of nodes \(K\) in a cluster and the number of observations \(N\) required to solve the equations in the general and special cases. In the general case where the orientations of the reference nodes are unknown, given \(N\) observations, we have \(KN + K (\lambda_0^0, \lambda_1^1, \ldots, \lambda_{K-1}^K, \theta_0, \theta_1, \ldots, \theta_{K-1})\) unknown parameters. For the number of equations, we showed earlier that there are \(2N - 1\) independent equations when there are three sensor nodes in the network. It can be easily verified that we can obtain \(N - 1\) additional linearly independent equations when a new sensor node joins the cluster. Hence, the number of total equations is \(2N - 1 + (K - 3) (N - 1)\) and is \(2KN - 2N - 2K + 4\) when taking the real and imaginary parts of all equations. Therefore, \(K\) and \(N\) should satisfy the following inequality

\[
KN - 2N - 3K + 4 \geq 0. \tag{20}
\]

The decentralized scheme can be considered a three-node cluster. So, we need five observations. In addition, if we increase the number of nodes in a cluster to four, only four observations will be required, which is smaller than the requirements for the decentralized scheme.

The other two cases which are worth studying are when \(N\) or \(K\) approaches infinity, respectively:

\[
K > \left[ \lim_{N \to \infty} \frac{2N - 4}{N - 3} \right] = 2 \tag{21}
\]
Equation (21) shows that no matter how many observations we make, we need at least three nodes in a cluster. On the other hand, according to (22), the number of required observations is always larger than three. Hence, we can conclude that the minimum number of required observations $N$ is four. In other words, increasing the number of nodes in a four-node cluster cannot further decrease the number of required observations.

In the special case where the orientations of the reference nodes are known, the number of unknowns becomes $KN + K − 2 (λ_0, λ_1, \ldots, λ_{K−1}, θ_2, θ_3, \ldots, θ_{K−1})$, and the number of equations is still $2KN − 2N − 2K + 4$. Therefore, $K$ and $N$ should satisfy the inequality

$$KN − 2N − 3K + 6 = (N − 3)(K − 2) \geq 0.$$  

(23)

So, we can solve for the unknowns only whenever $N \geq 3$ and $K \geq 2$.

D. Reference Node Hand-off

In a large network, the moving target may travel through different clusters where the nodes in the same cluster can observe the target at the same time. Each cluster has its own reference nodes; therefore, each cluster may define its own relative coordinate system. If the reference nodes are different from other sensor nodes and are equipped with global positioning systems, the relative coordinate can be transferred to absolute coordinates; hence, the coordinates of different clusters will be readily consistent. However, if we assume that none of the sensor nodes in the network is equipped with a global positioning system, each cluster will only know about its own coordinate system. In order to unify the coordinates among all clusters, the adjacent clusters must have at least a common member node. For example, cluster 1 shown in Fig. 5 contains nodes 0, 1, 2, and 3, and cluster 2 contains nodes 3, 4, and 5. The target moves from cluster 1 to cluster 2. Since node 3 knows its “relative” coordinates and orientations in both clusters, the coordinates of these two clusters can be unified.

III. Experiments

We now set to verify the performance of the proposed localization algorithm via experiments in an indoor environment. A set of Agilent Technologies ADCM 1670 image sensor modules (Fig. 6 (b)) are deployed for the experiments and the localization algorithms are programmed in MATLAB running on laptop computers. All nodes communicate with each other over wireless channels where IEEE 802.11b is adopted as the underlying protocol. For the experiments we use API libraries and MATLAB functions developed at our lab [15] for controlling the image sensors and performing packet transmission over wireless channels. We apply the background subtraction method introduced in [16] to detect the moving target (Fig. 6 (a)). In order to reduce the noise in the subtracted frames, the image frames grabbed by the sensor nodes are processed by a median filter [17]. While the target travels in the network, the reference node at the origin broadcasts synchronization signals regularly. The sensor nodes can therefore acquire images of the target simultaneously. We assume that the orientations of the reference nodes are known and both are perpendicular to the unit line as shown in Fig. 6 (c).

In the decentralized scheme, we deploy five sensors in the network. The reference nodes broadcast the positions of the target on their image plane. After obtaining enough observations, both the nonlinear (Gauss-Newton) and linear localization algorithms are run in each sensor node. The results
are shown in Figs. 7 and 8, respectively.

For the cluster-based scheme, four sensor nodes are deployed in the network. After acquiring the images, the sensor nodes other than the reference node at the origin (cluster-head) send their observations to the cluster-head. The cluster-head then estimates the desired coordinates and orientations for all cluster members. Only the nonlinear method is presented here (Fig. 9) since the linear method gives the same performance under the clustered and decentralized schemes.

IV. CONCLUSIONS

This work addressed the localization problem in a novel way, in that visual observations made from a moving target are used to derive equations that can be solved to obtain both the node and target coordinates simultaneously. The proposed algorithm can be implemented in decentralized or cluster-based fashions. In the general case, the model leads to non-linear equations, whereas if the orientations of the two reference nodes are known, the algorithm can be decomposed into two stages and the model becomes linear. The proposed algorithm was implemented for a network of five image sensors, each interfaced with MATLAB, which performed the functions of image processing, synchronization among the nodes, data transfer through wireless channels, and the localization algorithm.

REFERENCES