The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Distributive Target Tracking in Sensor Networks with a Markov Random Field Model

Lufeng Shi and Jindong Tan

***Abstract*— Tracking in sensor networks has shown great potentials in many real world surveillance and emergency system. Due to the distributive nature and unpredictable topology structure of the randomly distributed sensor network, a good tracking algorithm must be able to aggregate large amounts of data from various unknown sources. In this paper, a distributive tracking algorithm is developed using a Markov random field (MRF) model to solve this problem. The Markov random field (MRF) utilizes probability distribution and conditional independency to identify the most relevant data from the less important data. The algorithm converts the randomly distributed network into a regularly distributed topology structure using cliques. This makes tracking in the randomly distributed network topology simple and more predictable. Simulation demonstrate that the algorithm performs well for various sensor field setting, and for various target sizes.**

I. INTRODUCTION

Tracking has a wide range of civilian and military applications. It can be used in traffic control, surveillance, emergency response systems, search and rescue, supply chain management, and battle field awareness systems. Using sensor networks for tracking has become a popular choice in recent years. The sensor network is a distributive system which contains large amounts of small and inexpensive sensor nodes. The large amount of sensors provides enough redundancy to ensure the system is robust, and the small size allows the sensors to work under various space-constrained places, which provides direct line of sight measurements. However, the individual sensors have limited computation capabilities, so collaboration between sensors is needed in order to make inferences in the network. The sensor nodes also have limited communication capabilities, hence excessive collaboration is also infeasible. The key in distributed tracking algorithms is the in-network processing, where the data is aggregated while it is propagated. In this case, each sensor handles only part of the computation and transmits only the aggregated result.

There are many types of interested targets in sensor networks. These interested targets may be humans, animals, robots, vehicles, or even an area of events, such as spreading fire. These types of targets can usually be classified into two types, the small targets, and the large targets. The physical size of the small targets is relatively small compared to the average distances between sensors in the networks. Therefore they are usually modeled as a point, and their locations are represented by cartesian coordinates. When the target moves, the trajectory of the target is usually represented by a collection of these coordinates. Human, animals, robots, and most vehicles can be considered as small targets. The area of events are usually considered as large target. The large targets are too large to be represented by single coordinates, and detection of large targets usually involves many sensors. The large targets are usually identified by the sensors that can detect them, or they can be identified by the sensors that are close to the boundary.

The sensor network itself has a distributive topology which strongly resembles a graph, hence the sensor network can be modeled as a graphical model. The graphical model represents sensor nodes as random variables, and the links between the sensors are modeled as correlations. Making inferences in the sensor network can be treated as a stochastic process on a graph. The sensor networks can be further classified as a special type of graph, the Markov random field. In a Markov random field, the sensor nodes obey the Markov property where the nodes are only correlated to their immediate neighbors, hence the Markov property can be used to isolate the target location to a small set of sensors rather than the entire sensor field.

In this paper, a graphical model-based tracking algorithm is developed, and the sensor network is assumed to be a Markov random field. The algorithm can track both small and large target types. It identifies a small chained-form network along the target trajectory of small targets or the target boundary of large targets. The chained-form network is a sub-graph of the whole network, and it contains all the information needed to precisely locate the target. The randomly distributed sensor field is first converted into a grid-shaped structure using triangle cliques, and then a search algorithm is used to find the sensors that are along the trajectory of the target. By using a statistical graph approach, each sensor only represent a probability rather than a decision. Hence sensor failure and miss detection is handled automatically, since a few incorrect probabilities will not affect the joint probability distribution in the long run. With a Markov random field, the conditional independency allows each sensor to aggregate the current data without worrying about future data from other sources.

II. RELATED WORKS

Distributive tracking algorithms of small target in sensor networks are usually following two paths. The first path is

This work is supported in part by the National Science Foundation under Grant ECS #0528967 and CSR #0720781, Army Research Laboratory under contract ARL W911NF-06-2-0029 and CERDEC W15P7T-06-P228, Air Force Research Laboratory under the DURIP project FA-9550-07-1-0500, and TARDEC.

Lufeng Shi and Jindong Tan are with the Department of Electrical Computer Engineering, Michigan Technological University, Houghton, MI 49931, USA, {lushi,jitan}@mtu.edu

978-1-4244-3804-4/09/$25.00 ©2009 IEEE

the multi-sensor data fusion approach, and the second path is the graph approach.

The data fusion approaches are usually developed under the Bayesian framework, where a posterior distribution is computed by manipulating the Bayes rule and the joint density distribution of the sensory data. This method usually has two steps: first, predict the new state using past data, and second, update the prediction using new data. The Multiple Hypothesis Tracking (MHT) and Joint Probabilistic Data Association (JPDA) are the typical algorithms based on the Bayesian framework [1] and [2]. They exhaustively enumerate possible target actions and the associated probability, and then decide the best series of actions based on the probability. These approaches suffer from hypothesis explosion, which induces high communication and computation cost in the network. Many new variations of the data fusion approaches have been developed to simplify the computation in recent years[3], [4], [5].

One of the most popular variations is the Kalman filtering. Kalman filters introduced a pre-computable term, the Kalman gain, to reduce the computation load when new data arrives. Kalman filters also set a number of new constraints. The most notable constraints are the Gaussian noise and linear dynamic system assumptions. Many of the recent works are focused on relaxing these constraints [6], [7], [8].

In the graph-based approach, the topology of the graph is used to represent actual locations, and the tracking process is to categorize the detecting sensors based on their location information. In [9], for example, the rooms and the hallways are modeled as nodes in the graph, and the target is assumed to transit from node to node. Another graph-based tracking method is shown in [10]. Other graph-based methods study the overlapping of the sensing region with the aid of the known network topology [11], [12], [13]. All of these graph based algorithms depend heavily on knowledge of the network topology. The topology has to be regularly distributed and the node locations have to be known.

Large targets are usually modeled by their event boundaries. Some algorithms are developed for event boundary detection in the sensor networks. The algorithm proposed in [14] and [15] used a threshold-based system for each sensor to make detection decisions. The algorithm developed in [16] uses the k-nearest neighbor to group the detection sensors into clusters, and the cluster would represent an area of event. The outlier sensors are determined to be false alarms. A more advanced method is offered in [17], where the sensor field is recursively divided into sub-regions, until each sub-region contains only null detection sensors, or detection sensors. The boundary can be easily identified once these sub-regions are constructed.

These algorithms described above either heavily rely on probabilistic data association and ignore the topology of the network, or only exploit the topology and fail to collaborate sensor data statistically. In contrast, a statistical graphical model-based algorithm is developed in this paper. This algorithm not only utilizes the probabilistic framework for data association, but also studies the network topology to identify the independency between data in order to reduce the computation cost of the data association. Studying the network topology also makes the data routing easier during the data association. The algorithm also handles large targets and small targets at the same time, which is important in some fire and rescue situations. The large targets and small targets are usually considered separately in previous studies.

III. Problem Formulation

A. Graph representation of sensor field

Consider a typical tracking problem in a sensor field. Assume in this sensor field S with K sensors, each sensor is capable of detecting the presence of the targets,

$$m_k = \begin{cases} 1+n & \text{if target present } (H_1); \\ n & \text{if target absent } (H_0). \end{cases} \tag{1}$$

where m_k is a random variable that describes the detection status of the sensor k, $k \in \{1, 2, \ldots, K\}$, and n is zero mean noise. The distribution of the m_k is dependent on the noise distribution,

$$\begin{aligned} P(m_k|H_0) &= P_n(m_k) \\ P(m_k|H_1) &= P_n(m_k - 1), \end{aligned} \tag{2}$$

where $P_n(m_k)$ is the noise distribution.

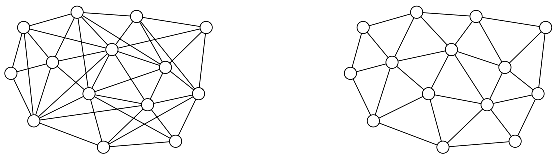
Given the sensing model of the individual sensor, the sensor field can be modeled using a graph. In the graph $G = (V, E)$, V represent the K sensors in the network, and E represent the measurement correlations between sensors. Each m_k corresponds to a measurement in V_k. According to the detection model of individual sensors, any two sensors that are within each other's detection range are correlated, and this makes the statistical information in the network redundant and difficult to analyze, as shown in Fig. 1(a). To reduce the statistical redundancy, the sensor field is assumed to be a Markov random field, where edges (correlations) only exist between immediate neighbors, as shown in Fig. 1(b). Even when two sensors are in range of each other, if they are separated by other sensors, they are conditionally independent to each other given the middle sensors. The conditional independency can be expressed mathematically as

$$\prod_{k=1}^{K} p(m_k|m_1, m_2, \ldots, m_K \setminus m_k) = \prod_{k=1}^{K} p(m_k|NB(m_k)), \tag{3}$$

where $NB(m_k)$ represent the immediate neighbors of k. Each connected neighborhood can be called a clique. The cliques are independent of each other, and hence the joint probability density distribution can be computed by multiplication. Let $M = \{m_1, m_2, \ldots, m_K\}$, the joint distribution of the sensor field is modeled using the Gibbs distribution,

$$p(\mathbf{M}) = \frac{1}{Z} \exp\left(\sum_{c \in C} (\psi_c(m_k))\right), k \in c \tag{4}$$

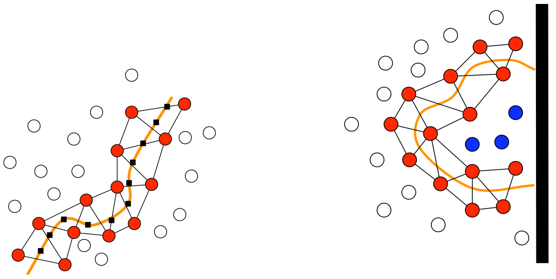
where $c \in C$ are the cliques in the field, and $\psi_c(m_k)$ represent the clique potential. Z is a normalization constant. With



(a) Network topology with all possible edges shown

(b) Network topology after assuming Markov property

Fig. 1. graph representation and Markov random field representation of the same sensor field.



(a) Representation of a small target in sensor networks

(b) Representation of a large target in sensor network

Fig. 2. Similar representation of various sized targets in the sensor network.

this network model, when computing the joint probability distribution, the probability distribution of each clique can be pre-computed locally, and hence it is distributive in-network processing.

B. *Definition of the targets*

In this paper, the targets in the sensor network can be classified into two categories, the small targets and the large targets. The small target can be modeled as a point mass. A fixed small target is a single cartisian coordinate (x_t, y_t). This allows us to do numerical manipulations such as measure the distance and perform multilateration. Once the target starts moving, it produces a continuous trajectory line. The usual way of modeling this trajectory line is to capture a static snapshot of the coordinate at each time frame, and the trajectory line can be represented by a vector of coordinates θ. Assume the sample is taken over T time period,

$$\theta = \{(x_t^1, y_t^1), (x_t^2, y_t^2), \dots, (x_t^T, y_t^T)\}. \tag{5}$$

As shown in Fig. 2(a), black squares are the snapshot coordinate at each time frame, the orange line is the target trajectory, and the circles are the sensors. The connected red circles are the sensors that are most relevant to the target trajectory; they will form a chained-form network which is used to estimate the location of the target and provide a routing path for data aggregation. The challenge is to identify these sensors from a randomly distributed network as shown in Fig. 1(b).

A large target is an area of event, where a single Cartesian coordinate cannot be used to represent this type of target, hence we cannot obtain a collection of coordinates over time to represent the large targets. The boundary of a large target is a continuous line, which exhibits a similar characteristic as the trajectory of the small target, thus a chained-form network can be identified as shown in Fig. 2(b) to represent the current location of the target. The main objective for tracking both large targets and small targets is to identify the chained-form network as shown in Fig. 2.

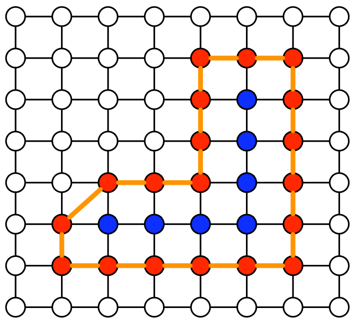


Fig. 3. Boundary lines in regularly distributed network

IV. Tracking in a Graph with Ising Model

The Ising model is a Markov random field with a grid topology. With this regular distributed network, it is much simpler to identify the chained-form network. Fig. 3 shows a possible target trajectory (boundary) in Ising model, where the circles are the sensors. The hollow circles represent the null detection sensors, and the colored circles represent sensors that are detecting the target. The goal is the identify the boundary of the colored sensors from the rest of the network.

In Fig. 3, the boundary sensors are the separators between the hollow circles and the colored circles. Hence to find the boundary sensors, the basic strategy is to search through the sensor field line by line, and identify the change point. This is simple since hollow sensors and colored sensors are in different detection statuses H_0 and H_1 as described in equation (1). The detailed algorithm can be formulated as following.

Assume a line of l sensors with sensor measurements m,

$$m_1^l = (m_1, m_2, \dots, m_l). \tag{6}$$

Let δ be the change point between two detection statuses, and further assume that H_0 comes before H_1 in the line of sensors,

$$\begin{aligned} m_1^\delta &\quad \text{in state} \quad H_0 \\ m_\delta^l &\quad \text{in state} \quad H_1 \end{aligned} \tag{7}$$

According to the Markov property, the probability of having δ as change point can be expanded as

$$P(\delta \in \text{boundary}|m_1^l, H_0, H_1) = CP(m_1^\delta|H_0)P(m_\delta^l|H_1), \tag{8}$$

where C is a normalization constant.

With equation (8), we can search through the all possible δ to find the most probable boundary location

$$\delta = \arg\max_{\delta} P(\delta \in \text{boundary}|m_1^l, H_0, H_1). \tag{9}$$

Equation (8) is only valid if the segment m_1^δ is in state to H_0, and segment m_δ^l is in state H_1. This is because the term $P(m_1^\delta|H_0)$ paires m_1^δ with state H_0. In real applications this is not always the case. Sensors m_1^δ may be in either state H_0 or H_1, hence $P(m_1^\delta)$ should be used in place of $P(m_1^\delta|H_0)$, and it can be computed using the prior,

$$P(m_1^\delta) = \int P(m_1^\delta|H)P(H)dH \tag{10}$$

where $P(H)$ is the joint prior of H_0 and H_1.

The assumption for Ising model is essential. Without the grid shape, the search will encounter unexpected branches and the correlation will vary from sensor pairs to sensor pairs. This will cause additional difficulty in performing the search. However, in typical sensor network deployment, networks are usually not grids. To make this algorithm applicable to all kinds of sensor networks, an equivalent topology structure for the sensor network must be derived which must satisfy the grid assumption. In this paper, a clique based topology representation is used to convert the irregular topology of the sensor network to a grid-like structure, then the algorithm is applied to this grid-like structure.

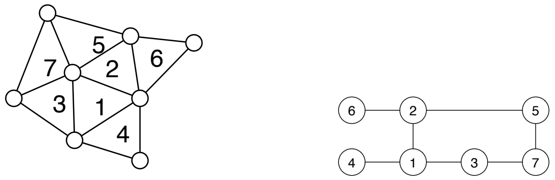
V. TRACKING IN RANDOMLY DISTRIBUTED SENSOR NETWORK

Typical sensor networks are randomly distributed, and the Ising model assumption cannot be achieved. However, by exploit the property of cliques and conditional independence, the randomly distributed network can be converted into a grid-like structure, which is very similar to the Ising model. Therefore, the method used for the Ising model can be extended to a randomly distributed network.

A. Grid-like topology construction

In an Ising model, each node has exactly four connected neighbors (except borders and corners), and the distance (correlation) between each neighbor is exactly the same. In order to construct a topology structure similar to the Ising model, we have to satisfy these two conditions: 1) have a fixed number of connected neighbors, 2) have the same distance (correlation) between connected neighbors.

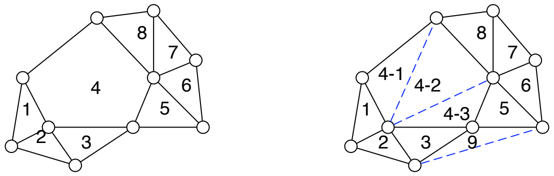
These two conditions can be easily satisfied by adopting triangular cliques. A clique is a cluster of sensors which are fully connected within the cluster. When three node triangle cliques are used, each clique would have exactly three adjacent triangles. The adjacent cliques are correlated by the two shared nodes. Hence, the clique structure can be used to simulate the behavior of a grid structure. Fig. 4 shows how a randomly distributed sensor topology can be converted into a grid-like structure. The network is first triangulated, each triangle is a natural clique, then a single "super node" is used to represent a whole clique in the grid-like structure.



(a) Random deployed sensor field

(b) Grid-like structured clique topology

Fig. 4. Converting the randomly distributed sensor network into a grid-like clique structure



(a) Triangulated sensor field with coverage holes

(b) Virtual triangles constructed to get across void areas

Fig. 5. Converting the randomly distributed sensor network into a grid-like clique structure

B. Void area in grid-like topology

The most popular triangulation algorithm is the Delaunay triangulation. However, Delaunay triangulation cannot be applied to the sensor network directly. This is because the algorithm does not have a bound on edge length; it may result in long edges that far exceeding the actual sensing and communication range of the sensors. At the same time, Delaunay triangulation is very difficult to achieve in a distributive fashion, hence it is not suitable for distributive sensor network. Therefore, a sub-graph of the Delaunay triangulation such as Relative Neighborhood Graph (RNG), or Gaberial Graph (GG) can be obtained. However, in RNG or GG, polygons may exist, where three-sensor cliques cannot be constructed as shown in Fig. 5(a). Area 4 in the figure is a void that cannot be modeled as a triangle clique.

Double-sensor cliques are introduced to solve this problem. For example, in Fig. 5(a), the five edges on the pentagon are modeled as five double-sensor cliques. Two of these double-sensor cliques can join together to form a virtual triangle if they share one sensor. A virtual link is added between the two non-sharing sensors in the group to finish the triangle. Fig. 5(b) is a demonstration of this situation. The dashed lines are virtual lines added to divide the polygon region into virtual triangles. Triangle $4-2$ is a special case, where it is formed by only one double-sensor clique. So, $4-2$ is treated as a triangle when constructing the clique structure, but when computing the joint distribution, it is just a single double-sensor clique. The construction of the virtual triangles obeys the following five rules as described in Algorithm 1.

C. The tracking algorithm in a grid-like structure

After the grid-like structure is constructed, the search algorithm for Ising model can be applied to identify the

Algorithm 1 Constructing virtual triangles with double-sensor cliques

1) An edge is a double-sensor clique if at least one of its adjacent triangles are missing.
2) In order to form a virtual triangle, the two real edges must both be double-sensor cliques.
3) Virtual links can be treated as double-sensor cliques when constructing virtual triangles,
4) Virtual links are not included when computing the clique potential and other statistic related quantities.
5) Virtual triangles can be formed if the added virtual link does not cross any of the existing links (virtual or real).

trajectory (boundary) of the target. However, since we don't have a true grid structure, and each "node" in our topology actually represents three nodes in the original structure, the detection models need to be modified slightly. The new detection model can be expressed as

$$\begin{aligned} P(H_0) &= \prod_{k=1}^{3} P(m_k = n) = \prod_{k=1}^{3} P_n(m_k) \qquad (11) \\ P(H_1) &= \prod_{k=1}^{3} P(m_k = 1 + n) = \prod_{k=1}^{3} P_n(m_k - 1), \end{aligned}$$

With $P(H_0)$ and $P(H_1)$ computed, the $P(C_1^\delta)$ and $P(C_\delta^l)$ can be computed. Note that l is the total number of the cliques in the line, δ is the change point, and C is the cliques. Since computation of $P(C_1^\delta)$ anc $P(C_\delta^l)$ are symmetric, we only focus on the computation for $P(C_1^\delta)$.

Given the detection result, the readings of each clique is independent to each other, hence,

$$P(C_1^\delta | H) = \prod_i P(C_i | H). \qquad (12)$$

We can then expand equation (10) as,

$$\begin{aligned} P(C_1^\delta) &= \int P(C_1^\delta | H) P(H) dH \qquad (13) \\ &= \int P(C_\delta | D) P(C_1^{\delta-1} | H) P(T) dH \\ &= P(C_1^{\delta-1}) \int P(C_\delta | H) P(H | C_1^{\delta-1}) dH. \end{aligned}$$

This separates the terms containing the current clique C_δ and all previous cliques $C_1^{\delta-1}$, hence it can be carried out in a distributive fashion using message passing algorithm. Each clique need only to compute their local $P(C_1^\delta)$ and pass it down the line; the next clique in line will treat the received value as $P(C_1^{\delta-1})$. With the detection pattern defined, the equation 8 can be rewritten as

$$\begin{aligned} P(\delta &\in \text{boundary} | C_1^l, H_0, H_1) \qquad (14) \\ &= K P(C_1^\delta) P(C_\delta^l) \\ &= K P(C_1^{\delta-1}) \int P(C_\delta | H) P(H | C_1^{\delta-1}) dH \\ &\times P(C_\delta^{n-1}) \int P(C_\delta | H) P(H | C_\delta^{n-1}) dH \end{aligned}$$

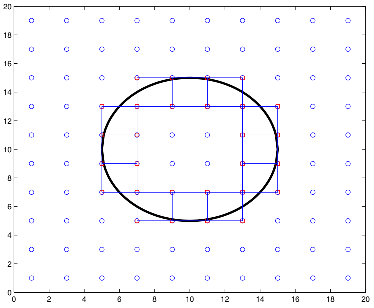


Fig. 6. Large target in a grid sensor field using Ising model

This equation works if there is only one change point δ in the line of cliques, however, there may be several change points in each line.

Let us introduce a new variable μ to represent the number of change points in the path, and tweak the equation a little to incorporate this variable,

$$\begin{aligned} P &\quad (\delta \in \text{boundary}, \mu | C_1^l, H_0, H_1) \qquad (15) \\ = P &\quad (\mu) P(\delta \in \text{boundary} | \mu, C_1^l, H_0, H_1), \\ = P &\quad (\mu) K P(C_1^{\delta_1}) \prod_{i=1}^{\mu-1} P(C_{\delta_i}^{\delta_{i+1}}) P(C_{\delta_\mu}^l) \end{aligned}$$

$P(\mu)$ is a priori distribution; it can be obtained empirically. For instance, if a large target is known to be a circular shape, a Poisson distribution with expected value of 2 should be a good assumption, because a straight line of sensors would have two intersections with the circle.

VI. SIMULATIONS

For the purpose of demonstration, assume 100 sensors are deployed in a 20 by 20 square region. Each sensor has a sensing radius of 5. The algorithm is carried out on this square region to detect a small target, a large target, and multiple large targets.

The search is first conducted on a sensor field that is modeled by an Ising model (grid). Fig. 8 shows the results. Since it is a grid network, no triangle clique is constructed. Circles are sensors, the red ones are the sensors that detecting the large target. The lines are the links in the chained-form network representing the target boundary.

For the randomly deployed sensor field, a Gabriel Graph is constructed on top of the sensor field, then virtual lines are constructed to break the polygons into triangle cliques. The triangle cliques are represented using "super nodes", and the "super nodes" topology is arranged into a grid-like structure.

Fig. 9 shows the detection of the large target in the randomly deployed sensor field. The red triangles are real triangle cliques, and the blue lines are virtual lines that connect the double-sensor cliques into virtual triangles. Only the cliques that are representing the boundary of the target are shown.

Since the search algorithm is carried out on a Markov random field, the detection of the target is independent

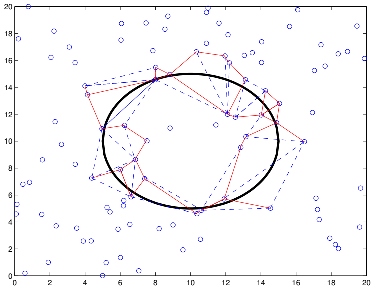


Fig. 7. Large target in a randomly deployed sensor field

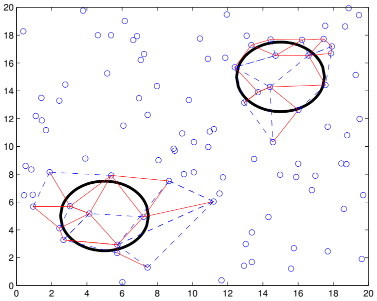


Fig. 8. Detection of two large target in the sensor network

given the immediate neighboring sensors. Hence, if we have multiple large targets, they can be tracked as well, using the same search algorithm. Fig. 8 shows the detection of two large targets in the same sensor field.

Finally, small target trajectory is searched by the algorithm, and located on the sensor field with a chained-form cliques. Fig. 9 shows the tracking result for the small target. Once the chained-form network is constructed, multilateration can be used in each triangle cliques to compute the exact position of the target at each time instance. The formulation of the multilateration can be found in [18].

VII. Conclusion

A distributive algorithm is developed to solve target tracking problems in sensor networks. The algorithm inherits the statistical framework for tracking, and at the same time exploits the topology structure for both simplicity and generality. Both the trajectory of the small targets and the event boundary of the large targets can be found using this algorithm. The distributive algorithm allows large scale implementation. Simulations demonstrate the functionality of the algorithm against small targets, large targets, and multiple targets.

References

[1] S. S. Blackman, “Multiple hypothesis tracking for multiple target tracking,” *IEEE Aerospace and Electronic Systems Magazine*, vol. 19, no. 1, pp. 5–18, 2004.

[2] J. Vermaak, S. Godsill, and P. Perez, “Monte carlo filtering for multi target tracking and data association,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 1, pp. 309–332, 2005.

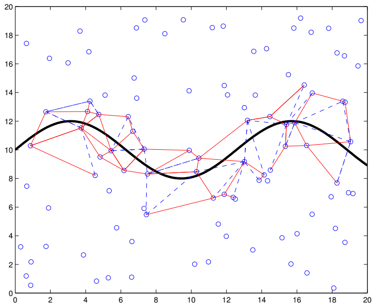


Fig. 9. Small target trajectory in the sensor field

[3] D. Smith and S. Singh, “Approaches to multisensor data fusion in target tracking: A survey,” *IEEE Transactions on Knowledge and Data Engineering*, Jan 2006.

[4] E. F. Nakamura, A. A. F. Loureiro, and A. C. Frery, “Information fusion for wireless sensor networks: Methods, models, and classifications,” *ACM Computing Surveys*, vol. 39, no. 3, p. 9, 2007.

[5] E. F. Nakamura and A. A. F. Loureiro, “Information fusion in wireless sensor networks,” in *Proceedings of the international conference on Management of data*, (New York, NY, USA), pp. 1365–1372, ACM, 2008.

[6] D. Atherton, J. Bather, and A. Briggs, “Data fusion for several kalman filters tracking a single target,” *IEEE Proceedings Radar, Sonar and Navigation*, vol. 152, pp. 372–376, October 2005.

[7] H. Y. Cheng and J. N. Hwang, “Multiple-target tracking for crossroad traffic utilizing modified probabilistic data association,” in *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 1, pp. I–921–I–924, April 2007.

[8] E. Chan, G. Baciu, and S. Mak, “Using wi-fi signal strength to localize in wireless sensor networks,” in *International Conference on Communications and Mobile Computing*, vol. 1, pp. 538–542, Jan. 2009.

[9] C. S. Jensen, H. Lu, and B. Yang, “Graph model based indoor tracking,” in *Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*, pp. 122–131, May 2009.

[10] S. Oh and S. Sastry;, “Tracking on a graph,” *Fourth International Symposium on Information Processing in Sensor Networks*, pp. 195 – 202, Mar 2005.

[11] J. Singh, U. Madhow, R. Kumar, S. Suri, and R. Cagley, “Tracking multiple targets using binary proximity sensors,” *Proceedings of the 6th international conference on Information processing in sensor networks*, pp. 529–538, 2007.

[12] X. Liu, G. Zhao, and X. Ma, “Target localization and tracking in noisy binary sensor networks with known spatial topology,” *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 2, pp. II–1029–II–1032, 2007.

[13] L. Hong, J. Rushing, S. J. Graves, S. Tanner, and E. Criswell, “Real time target tracking with binary sensor networks and parallel computing,” *IEEE International Conference on Granular Computing*, pp. 112–117, 2006.

[14] K. Ren, K. Zeng, and W. Lou, “Fault-tolerant event boundary detection in wireless sensor networks,” *Global Telecommunications Conference,*, pp. 1 – 5, Jan 2006.

[15] K. Ren, K. Zeng, and W. Lou, “Secure and fault-tolerant event boundary detection in wireless sensor networks,” *IEEE Transactions on Wireless Communications*, vol. 7, pp. 354 – 363, Jan 2008.

[16] B. Krishnamachari and S. Iyengar, “Distributed bayesian algorithms for fault-tolerant event region detection in wireless sensor networks,” *IEEE Transaction on Computers*, vol. 53, pp. 241–250, Jan 2004.

[17] R. Nowak and U. Mitra, “Boundary estimation in sensor networks: Theory and methods,” *Lecture Notes in Computer Science*, vol. 2634, pp. 80–95, Jan 2003.

[18] A. Savvides, H. Park, and M. Srivastava, “The n-hop multilateration primitive for node localization problems,” *Mobile Networks and Applications*, Jan 2003.