A Data Fusion Algorithm for Large Heterogeneous Sensor Networks

Hong Lin, John Rushing, Sara Graves, and Evans Criswell
Information Technology and Systems Center
University of Alabama in Huntsville, Huntsville, Alabama 3589, USA
linh@itsc.uah.edu

Abstract

A distributed search based data fusion algorithm is presented for target detections in large heterogeneous sensor networks. A score function is introduced as the objection function during the optimal search. The network state is determined when the score is the highest. A close to optimal solution can be obtained before the arrival of the next sensor data thus enabling real time target tracking. The algorithm is evaluated with a series of real-time simulations on networks of variable sensor compositions with a commodity Linux cluster.

1. Introduction

With the advancing microsensor, computer [1-2], and wireless technology, networks of densely distributed primitive sensors are becoming increasingly attractive in both military and civilian applications. These networks can continuously monitor a much larger geographic area and track objects of interest with a much lower cost than that of using the traditional sensor networks, such as radars. Since the communication between sensors and data processing units is via wireless links, the sensors can be distributed to monitor regions once unreachable to human. Many challenges in data fusion, data communication, and energy efficiency arise with the wide applications of wireless sensor networks [3].

Dynamic target detection and tracking with sensor networks is the process of network state estimation of more than one object over a region of interest for a period of time [4]. The temporal dynamic network state is determined through fusing information from multiple sensors in the networks. In heterogeneous sensor networks, the data fusion algorithm needs to integrate the disparate sensor data from different types of sensors. In conventional sensor networks where each sensor is capable of detecting and locating the targets, the sensor data fusion is a process of associating the estimates from each sensor to build target tracks with different data association models [4-8]. The sensor data fusion in large wireless primitive sensor networks is very different from that in the conventional sensor network since each primitive sensor is not capable of locating one or more targets. A particle filtering algorithm for target tracking using binary sensor networks has been reported in [9]. An optimal search based fusion algorithm with simulated annealing approach also has been developed for target tracking in large binary sensor networks [10]. Shrivastava, et al [11] explored the fundamental performance limits of tracking a target in a two-dimensional field of binary sensors. Deans, et al [12] used the adaptation of bundle adjustment for SLAM problem using a bearing-only sensor. For large bearing-only sensor networks, a heuristic fusion algorithm with triangulation and clustering approach has been studied in [13]. Gorski, et al [14] introduced a high level error reduction algorithm which tries to reduce the errors on the fusion results, not during the sensor fusion for a large wireless sensor network in general. According to our knowledge, there is no report on a fusion algorithm which is capable of fusing the disparate sensor data from both primitive sensors and position sensors in a large heterogeneous sensor network.

In this paper, a simple search based fusion algorithm is introduced for networks composed of large numbers of low-fidelity primitive sensors and small numbers of high fidelity position sensors. The algorithm is also parallelized for distributing computing, so the right decision can be made within the limited time frame in real time target tracking applications.

2. Sensor Model and Sensor Networks

Three types of sensors are considered in the sensor data fusion algorithm, binary sensors, bearing-only sensor, and position sensors, and each sensor is aware of its location. This can be easily achieved with the GPS technology. All the sensors are assumed to be omnidirectional, capable of detecting multiple targets in all directions. Each type of sensors has its own false

0-7695-2981-X/07 $25.00 © 2007 IEEE
DOI 10.1109/WASA.2007.23

positive error rate (*fp*) and false negative error rate (*fn*). These errors encapsulate failures of various types including communication loss, sensor malfunction, false alarm in a cluttered environment, etc. We will not consider the possibility that the closer target might block the other one on the same direction. We argue that this is highly unlikely in case of real world when the targets are moving in a three-dimensional space.

2.1. Binary and bearing-only sensors

A binary sensor can only give an *on* or *off* signal to indicate the presence or absence of targets within its detection range. A bearing-only sensor, such as a wireless surveillance camera, can report the angular direction of a target relative to a directional reference in case of detection. The measurement of a bearing-only sensor to a target can be represented by a line segment starting from the sensor position and ending at the point determined by the detection angle and the sensor detection range.

Besides the false errors, the bearing sensors also have errors in the reported angles due to signal noise. Gaussian white noise with zero mean is commonly used to model such errors [15]. The detection and measuring information of the bearing-only sensors are shown in Fig. 1.

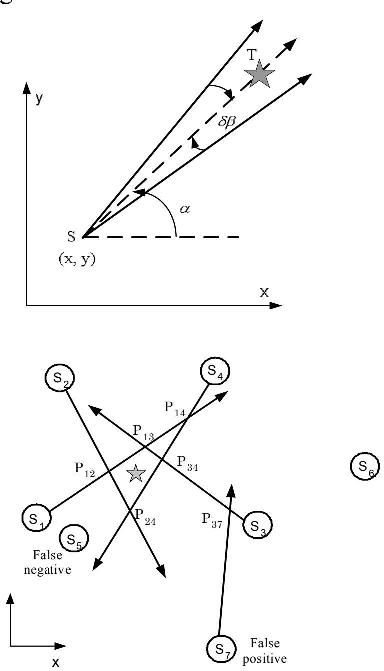


Figure 1. Target detection with one (a) and more (b) bearing-only sensors.

In Fig. 1a, target T is detected by sensor S. The angular measurement is the true bearing α corrupted by the noise $\delta\beta$, i.e. $\alpha+\delta\beta$. The noise can be simulated and generated with a predefined bearing variance. Fig. 1b illustrates the detections of seven bearing-only sensors (S_1 to S_7) around target T. Sensors S_1 to S_5 are capable of detecting target T; S_6 and S_7 are too far away to detect the target. S_5 has no measurement due to the sensor's false negative error and S_7 gives a random angle due to the false positive error. The five line segments result in six intersections. All these intersections are evaluated with the score function defined by Equation (2) in Section 3 during the optimal search.

2.2. Position sensors

The position sensors, such as radars, are very sophisticated systems with very low error rates, and modeling such a system is beyond the scope of this research. What the data fusion algorithm needs from the position sensors is a set of reported target positions. The position variance can be determined through the range, azimuth (bearing), and elevation variances in case of three-dimensional network model. These variances increase with the reduction of signal to noise ration (SNR). As defined in [16], the radar signal attenuation model can be simplified as $SNR = C_0 / d^4$, where d is the distance between the target and the sensor. In two-dimensional space, the range variance can be defined as $\sigma_r^2 = C_1 / SNR$ and azimuth variance as $\sigma_\theta^2 = C_2 / SNR$. The constants are set as the follows: C_0 = 1.5x10^9 mile^4, C_1 = 5 x 10^{-3} mile^2, and C_2 = 1.2 degree squared. The position error increases significantly as the target is farther away from the sensor.

2.3. Heterogeneous sensor networks

We are interested in networks composed of a small number of the reliable position sensors, and a large number of the short-range primitive sensors to cover a large area, such as an 800 x 800 mile region. The binary and bearing-only sensors will be distributed randomly and uniformly over the region. With the manageable number of position sensors, it is reasonable to arrange the sensors evenly over the region to fully take the advantage of position sensors.

3. Sensor Fusion Algorithm

The challenge of data fusion for the heterogeneous sensor networks is how to efficiently merge the different information originating from different types of sensors in a consistent manner. In this paper, the measurements from the binary and the bearing-only sensors are converted to a set of potential target positions. Three transforms will be performed on these potential positions and those from the position sensors. Each transform will be evaluated with a score function. The different error rates of the sensors will be considered in the score function.

3.1. Potential target locations–the hypotheses

The information from a binary sensor is its location, and a 1 or 0 signal indicating if there is target within its sensing range. Hence if a binary sensor gives a 1 signal, the best estimate for the target location we can have is the position of the binary sensor itself.

For the bearing-only sensors, each sensor will give a set of line vectors starting from the sensor position toward the targets for targets within its sensing range. Two or more bearing-sensors are needed to locate a target. The detection lines from two reachable bearing-only sensors will intersect if they both detect the target. The best estimate for the target location is the centroid of the cluster of the intersections. Because of the existence of other type of sensors in the heterogeneous sensor networks, the best estimate from the bearing-only sensors might not be the optimal solution for the whole network. Hence it is not necessary to solve for the centroid of the cluster of intersections.

The sum of the locations of the binary sensors, the intersections of all the bearing-only sensor measurements (line segments), and the set of reported positions from position sensors gives the set of potential target locations. A score function, which is introduced in the next section, will be utilized to evaluate all the potential locations and their transforms to determine the network state.

3.2. The score function

In the Bayesian inference data fusion, the likelihoods of a set of hypotheses are computed with the information at hand [17], and the hypothesis is true if the likelihood is larger than a threshold. With the assumption that the error rates are independent across the sensors and over time, the likelihood for hypothesis h that there is a target at location p can be evaluated with the following equation,

$$L(p) = \frac{\prod_{i=1}^{k_1}(1-fp_i)\prod_{j=1}^{k_2}(fn_j)}{\prod_{i=1}^{k_1}(fp_i)\prod_{j=1}^{k_2}(1-fn_j)}. \qquad (1)$$

In Equation (1), f_{pi} is the false positive error rate of sensor i, and f_{nj} is the false negative error rate of the sensor j, k_1 is the number of sensors which report the detection, k_2 is the number of sensors that report no detection within the detection range. The sum of k_1 and k_2 is the number of sensors of all types which can see location p. In Equation (1), the numerator gives the probability of h being true, and the denominator is the probability of h being false. When the numerator is larger than the denominator, the likelihood of hypothesis h being true is higher.

Since multiplication of fractions is computationally expensive and prone to floating point errors such as underflow and overflow, a convenient transformation into the logarithmic world is applied to Equation (1). We call ln(L(p)) the score function $\Lambda(p)$ for location p. With the position sensor model described in Section 2.3, the position variation increases when the sensor is far away from the target. The sensor might not report the detection when the position error is larger than a predefined threshold and give a negative contribution to the score. Therefore, contribution from sensor closer to the target should be weighted more than those further away from p. Based on the position sensor model described in Section 2.3, d^2 will be used to weight the contribution of the position sensors in computing Equation (1), where d is the distance from a sensor to p. The final score equations is defined in Equation (2).

$$\begin{aligned}\Lambda(p) = &\sum_{i=1}^{k_1}\left(\ln(1-fp_i)-\ln(fp_i)\right)/d_{ip}^2 + \\ &\sum_{j=1}^{k_2}\left(\ln(fn_j)-\ln(1-fn_j)\right)/d_{jp}^2\end{aligned} \qquad (2)$$

3.3. The optimal search algorithm

The objective of the optimization search is to find the subset of the original hypotheses which has the highest score value based on the readings of all the sensors over the network. During the search process, three types of transforms are applied randomly: *add*, *move*, and *delete*.

The *delete* transform computes the scores of every location from the current target set, and those whose scores are negatives based on the current readings of all the sensors will be removed.

The *add* transform selects a target randomly from the potential target set at the current time and compute its score with Equation (2). The operation will be accepted when the score is positive; rejected otherwise. The *move* transform selects a target location randomly from the current target set and perform a set of movements of small distance. The *move* transform is necessary to locate the target more precisely. During the search, *add* or *move* transform will be selected randomly. This process is repeated for as many iterations as possible given the time constraint for real time target tracking.

4. Efficiency and Scalability Improvement

Efficiency and scalability are two great challenges in data fusion for large scale sensor networks, especially for in real time application. We present two approaches in improving the fusion efficiency and scalability: the parallel computing and the *update window* scheme.

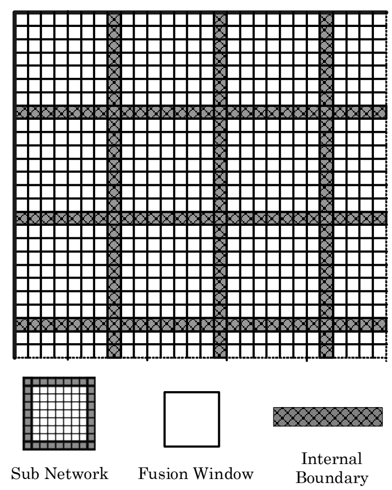
4.1. Fusion algorithm parallelization

For data fusion in networks with a very large number of sensors, the computational time of the fusion algorithm is strongly dependent on the number of sensors. We adapt the spatial decomposition approach [18]. The region of interest will be divided into consecutive small regions. The fusion time for each sub network can be reduced close to n-fold depending on the parallelization overhead due to the n-fold reduction of the number of sensors on a n-node cluster. Each slave processor works as a data processing unit collecting and processing data only from the sensors in the corresponding sub region. A master process controls and synchronizes the network target detecting process. At the end of the data fusion process, all the slave processes send their local network state estimate to the master process. The master process collects the observations from each sub region and performs the data association with Nearest Neighbor Data Association [4] (NNDA). In the networks considered here, and in most of the sensor network applications with primitive sensors, the number of targets is far less than the number of sensors. Since only the observed target information is sent from the slave processes to the master, the communication over head is relatively small.

In dynamic target detection, it is possible that a target moves across an internal boundary between two sub networks. Also, targets close to a boundary of one region can also be detected by some sensors in its neighboring sub regions. Therefore the divided networks must overlap.

4.2. Update windows

For short-ranging primitive sensors, the effects of a particular transform to the score function in Equation (2) are local to the area where the transform occurs. There is no need to check every binary and bearing-only sensor over the entire network but only those close by. Hence, a sub network can be further divided into small grids spatially. We call these grids *update windows*. The score function with respect to the binary and bearing-only sensors can be computed within each *update window* and its neighboring *update windows* in case the target is within the detection range of the sensors in the neighbors. The sensors of the update windows can be set during the network initialization. The size of the update windows can be small as long as it is larger than the sensor detection range. A 10 mile by 10 mile window size is assumed with the three-mile sensing range assumption for the binary and the bearing only sensors. Hence, the computing time for evaluating a transform with Equation (2) is further reduced dramatically as the number of primitive sensors on each update window could be very small. Considering the small number of position sensors and their much longer detection range, the *update window* scheme will not be applied to the position sensors. The relationships of the sensor network, sub networks, the overlapped internal boundaries, and the *update windows* are shown in Fig. 2. The overlapped region for the internal boundaries is set to the size of one update window.



Figuew 2. Sensor network, overlapped sub networks, and fusion windows used in the distributed fusion algorithm.

5. Evaluations

5.1. Simulation data

This research considers the possibility of accurately detecting dynamic targets over a very wide area using very large numbers of primitive sensors and a very small number of position sensors in the two dimensional space only. The input target tracks for the simulations are generated using Modern Air Power, a theatre level air combat simulation used by the USAF Squadron Officers College (SOC) for training. There are about 100 dynamic targets of aircraft over an 800 mile × 800 mile area for a period of 300 seconds.

The simulations are performed on a Linux cluster composed of 25 computing nodes; each node has two SMP CPUs of 2.6 GHz. With one processor for the master process, the cluster can have up to 49 slave processes running in parallel for data fusion. The entire region is divided into 7 × 7 sub networks. Each sub network is assigned to one slave processor. The lam-oscar (Open Source Cluster Application Resources) package installed on the Linux cluster provides the standard message passing interface (MPI) for communications between the master and the slave processes. A series of simulations are performed to evaluate the efficiency and accuracy of the distributed fusion algorithm.

The performance of the distributed fusion algorithm will be evaluated using the same set of metrics as in [10], probability of target detection (POD), false alarm rate (FAR), and average detection position error (AVD).

5.2. Experimental simulations

Table 1. Performance metrics of networks of variable sensor compositions and 5 second time intervals

Number of Sensors			Performance		
Binary (k)	Bearing-Only (k)	Position	POD (%)	FAR (%)	AVD [mile]
1,000	*0*	*0*	*99.2*	*0.44*	*0.018*
0	*200*	*0*	*99.7*	*0.00*	*0.004*
0	*0*	*196*	*99.8*	*0.76*	*0.153*
500	*5*	*25*	*98.9*	*2.69*	*0.215*
500	*0*	*0*	*98.5*	*7.27*	*0.305*
0	*50*	*0*	*60.5*	*0.00*	*0.007*
500	*50*	*0*	*98.9*	*3.06*	*0.187*

5.2.1 Sensor network performance. To evaluate the performance of the fusion algorithm, a series of simulations on networks of different configurations are implemented for real time target detection. The performance metrics of these simulations are summarized in Table 1. A 5s time interval is used in all the simulations. Real-time target detection can be achieved with a performance of higher than 99% POD, less than 1.0% FAR for networks having sensors up to one million. The network with mixed three types of sensors is far out performed the corresponding homogeneous sensor networks.

5.2.2 Efficiency. To further investigate the efficiency of the distributed fusion algorithm, a series of simulations are performed using time intervals of 5, 4, 3, 2, 1 and 0.5 seconds respectively on a heterogeneous network of 500,000 binary sensors, 50,000 bearing-only sensors and 25 position sensors. The results are summarized in Table 2. The network performance starts to decay with one-second time interval, i.e. the optimal solution could not be obtained due to the lack of enough fusion time.

Table 2. Performance of network of 500k binary sensors, 50k bearing-only sensors, and 25 position sensors with variable simulation time intervals.

Time Interval [second]	POD (%)	FAR (%)	AVD [mile]
5	*98.89*	*2.69*	*0.215*
4	*98.95*	*2.8*	*0.208*
3	*98.53*	*2.67*	*0.212*
2	*98.53*	*2.66*	*0.214*
1	*96.52*	*2.14*	*0.215*
0.5	*87.57*	*1.99*	*0.222*

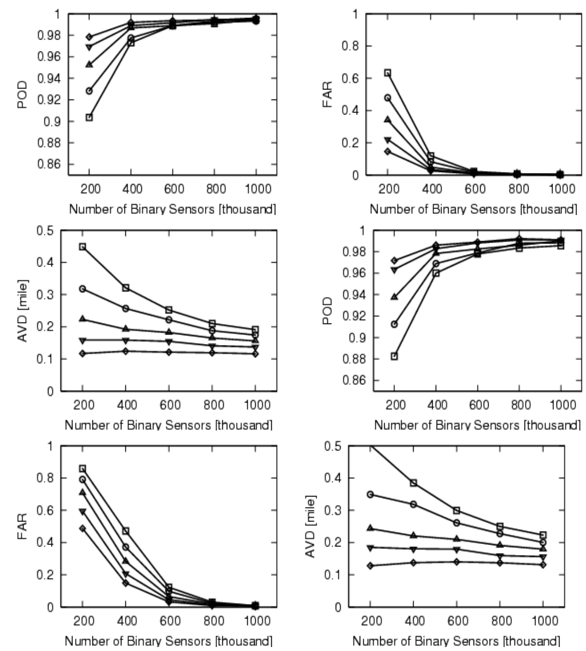


Figure 3. Simulation with networks of 25 position sensors, variable binary and bearing-only sensors; 5s time interval, *fp* and *fn* for binary and bearing-only sensor are set to (a) 10%, (b) 15%.

5.2.3 Higher sensor error rate. Simulations with higher error rate primitive sensors are implemented to see how the error rates affect the network performance. Fig. 3 compares the simulation results on sensor networks with 25 evenly distributed position sensors and variable primitive sensors of error rates 10% (a) and 15% (b). With the same network composition, the network performance is worse with sensors of higher error rates. Better network performance can be expected when extra binary or/and bearing-only sensors are added to the network. The simulation results also indicate that there is a saturation point of the network density beyond which the network performance cannot be further improved by adding more sensors.

6. Conclusions

A distributed data fusion algorithm is presented and evaluated for real time target tracking with large heterogeneous sensor networks. The algorithm fuses the disparate sensor data from the low fidelity primitive sensors as well as the high fidelity position sensors to obtain the best network state estimate accurately and efficiently with significant sensor error tolerance.

7. Acknowledgements

This work has been supported by DoD through Information Technology and Systems Center, University of Alabama in Huntsville under GS-23F-0062P.

8. References

[1] J. W. Gardner, V. K. Varadan, and O. O. Awadelkarim, *Microsensor, MEMS and Smart Devices*. New York: Wiley, 2001.

[2] L. Doherty, B. A. Warneke, B.E. Boser, and K. S. J. Pister, "Energy and performance considerations for smart dust," *International J. of Parallel and Distributed Systems and Networks*, Vol. 4, No. 3, 2001, pp. 121-133.

[3] C-Y Chong and S. P. Kumar, "Sensor networks: Evolution, opportunities, and challenges," *Proceedings of the IEEE*, Volume 91, No. 8, Aug. 2003, pp. 1247-1256.

[4] L. A. Klein, *Sensor and Data Fusion: A Tool for Information Assessment and Decision Making*, SPIE Press, 2004

[5] E. Waltz and J. Llinas, *Multisensor Data Fusion*, Artech House, 1990.

[6] S. D. O'Neil and L. Y. Pao. "Multisensor fusion algorithms for tracking," *Proc. 1993 American Control Conference*, San Francisco, CA, Jun. 1993, pp.859-863.

[7] P. K. Varshney, "Multisensor data fusion," *Electronics & Communication Engineering Journal*, Dec. 1997.

[8] M. E. Liggins II, C-Y Chong, I. Kadar, M. G. Alford, V. Vannicola, and S. Thomopoulos, "Distributed fusion architectures and algorithms for target tracking," *Proceedings of the IEEE*, Vol. 85, No. 1, Jan. 1997.

[9] J. Aslam, Z. Butler, F. Constantin, V. Grespi, G. Cybenko, D. Rus, "Tracking a moving object with a binary sensor network," *Conference on Embedded Networked Sensor Systems*, Los Angeles, CA, 2003.

[10] H. Lin, 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*, Atlanta, GA, May 10-12, 2006.

[11] N. Shrivastava, R. Mudumbai, U. Madhow, and S. Suri, "Target tracking with binary proximity sensors: Fundamental limits, minimal descriptions, and algorithms," *SenSys'06*, November 1-3, 2006, Boulder, Colorado, USA.

[12] M. Deans and M. Hebert. "Experimental comparison of techniques for localization and mapping using a bearing--only sensor," *7th International Symposium on Experimental Robotics,* Honolulu, HI, December 2000.

[13] H. Lin, *Data Fusion in Bearing-only Sensor Networks for Real-Time Target Tracking*, A Master Thesis, University of Alabama in Huntsville, 2006.

[14] J. Gorski, L. Wilson, I.H. Elhajj, and J. Tan, "Data fusion and error reduction algorithms for sensor networks," *International Conference on Intelligent Robots and Systems*, Aug. 2-6, 2005, pp. 610-615.

[15] Y. Bar Shalom and T. E. Fortmann. *Tracking and Data Association*. Academic-Press, Boston, 1988.

[16] R. Niu and P. K. Varshney, "Distributed detection and fusion in a large wireless sensor network of random size," Journal on Wireless Communications and Networking, Vol. 5, No. 4, pp. 462-472, 2005.

[17] Y. Zhang and Q. Ji, "Active and dynamic information fusion for multisensor systems with dynamic Bayesian networks," *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics*, Vol. 36, No. 2, April 2006.

[18] Charles Pedersen, "Architecture and performance of a parallel data-fusion multi-target tracking code for real-time applications," *Software Tech News, Vol. 4, No. 1, High Performance Computing*. Available: http://www.softwaretechnews.com/stn4-1/datafusion.html