EKF-based Adaptive Sensor Scheduling for Target Tracking

Yang Liu
Center for Control and Optimization
South China University of Technology
Guangzhou, 510460, China
liuyang185177@gmail.com

Zhendong Sun
Center for Control and Optimization
South China University of Technology
Guangzhou, 510460, China
zdsun@scut.edu.cn

Abstract

For Wireless Sensor Networks (WSN), target tracking is a canonical problem that collaborates signal and information processing to dynamically manage sensor resources and efficiently process distributed sensor measurements. This paper proposes an adaptive sensor scheduling strategy that jointly sets up distribute dynamic clustering, selects the tasking sensor, and determines the sampling interval. The approach utilizes Least-Square (LSQ) in initializing, Extended Kalman Filter (EKF) in tracking accuracy estimation, and adaptive sampling in velocity prediction. Simulation results demonstrate significant improvement in tracking accuracy compared to the non-adaptive approaches.

Keywords: WSN, target tracking, adaptive sensor scheduling, EKF.

1 Introduction

Wireless Sensor Network (WSN) is an emerging technology for monitoring the physical world with a densely deployed network of sensor nodes [1]. The main advantages of WSN include its low cost, rapid deployment, self organization, and fault tolerance. It is widely applicable to environment monitoring, industrial sensing and diagnostics, military surveillance, and navigation and control of mobile robots, to list a few.

Due to limited communication, computation, and sensing capabilities, WSN relies on collaborative signal and information processing $(CSIP)$ to dynamically manage/schedule sensor resources and efficiently process distributed sensor measurements [2]. In moving target tracking literature, the tasking sensors were scheduled based on the uniform sampling interval, ignoring the changing of the target dynamics and the estimation accuracy. In [5], Xiao, Wu, and Xie proposed an adaptive sensor scheduling scheme for target tracking in WSN by jointly selecting the next tasking sensor and determining the sampling interval based on predicted tracking accuracy and tracking cost.

This paper proposes an adaptive sensor scheduling strategy by jointly setting up distribute dynamic clustering, selecting the tasking sensor, and determining the sampling interval according to the predicted tracking moving velocity. The approach employs Least-Square (LSQ) used in initializing, an Extended Kalman Filter based on estimation technique to predict the tracking accuracy, and adaptive sampling in velocity prediction. The main contribution of this work is the developed of a distribute dynamic clustering algorithm and an adaptive sampling interval algorithm.

The paper is organized as follows: The distribute dynamic clustering algorithm is briefed in Section 2. The proposed tracking algorithm and system model are presented in Section 3, and an adaptive sensor scheduling is proposed in Section 4. Simulation results are given in Section 5. Finally, we conclude in Section 6.

2 Distribute dynamic clustering algorithm

To solve the problem that energy in sensor limited and to prolong the lifetime of sensor network, we use distribute tracking algorithm to make sensor dynamic cluster [3]. It assumes that all sensors have same communication radius and synchronization. The distribute dynamic clustering algorithm can be described as follows:

$\mathbf{a}$. Initialization. When the target reaches the monitored field, the sensors detected the target make up cluster, select the sensor with nearest distance from target to a cluster head (CH). CH becomes the central of signal and information processing, and receives the tasking sensor measure data.

$\mathbf{b}$. CH sends a message to trigger the second nearest sensor i, and commands it to measure the target distance to get an initial X and Y position. The tasking sensor returns a measure message (t, p, d), where t is the current time, p is the known position of tasking sensor, and d is the distance between the target and tasking sensor.

$\mathbf{c}$. CH receives the new measure distances, performs data fusion, EFK, and predicts the position of the target.

978-0-7695-3494-7/08 $25.00 © 2008 IEEE
DOI 10.1109/ISISE.2008.286

d. Turn to step a. CH verifies if it is necessary to set up a new cluster. If yes, it sets up a new cluster and selects new CH, then the old CH sends the historical data and state estimation to new CH. If not, CH sends a sample message to trigger $(i+1)th$ nearest sensor, until all sensor in cluster have sample (then return to the second one).

f. When the target is moving, it returns upwards step, until the target departs from the monitored field.

3 Tracking Algorithm

This section discusses the different components of our tracking algorithm and how they fit together [4]. We start by formally defining the tracking problem. If a tasking sensor obtains a distance estimate, it sends to CH. CH gets a triple: (t, p, d). Over time, the CH obtains a sequence of such triples: $(t_1, p_1, d_1), \cdots\cdots, (t_n, p_n, d_n)$, where the subscripts increase by one for each distance samples.

The three procedures of the Kalman filter tracking algorithm are shown as follow: a least-squares minimization (LSQ), an extended kalman filter (EKF), and an outlier rejection.

3.1 Least Squares Minimization

If the mobile device is static, a standard way to solve the problem of estimating $\hat{\theta}_n$ is to minimize the sum of the squares of the error terms corresponding to each distance sample. This method, called least-squares minimization (LSQ), estimates $\hat{\theta}_n$ by minimizing

$$\sum_{i=1}^{n}(\|\hat{\theta}_n - p_i\| - d_i)^2$$

Here, $\|\hat{\theta}_n - p_i\|$ is the Euclidean distance between the estimated coordinate of the mobile device and the tasking sensor at position p_i.

To produce a good $\hat{\theta}_n$, we make the simultaneous assumption and assume that the distances were collected when the target was at the same point. LSQ is useful in initializing and resetting the Kalman filter, a capability we use when the system first turns on or when our filter gets into a bad state.

3.2 Extended Kalman Filter

Here, we consider the problem of tracking a single target and use Extended Kalman Filter (EKF). The following takes the PV (Position-Velocity) model with 4 sensors as an example to illustrate our tracking algorithms and the target motion is nonlinear [7].
Let state

$$X = [x, y, \dot{x}, \dot{y}]^T, Z = [d_1, d_2, d_3, d_4]^T,$$

where

$$d_i = \sqrt{(x - x_i)^2 + (y - y_i)^2},$$

(x_i, y_i) is the location of tasking sensor i. The state equation is given as follows:

$$X_{k+1} = F(\Delta t) X_k + W_k \tag{1}$$

where

$$\mathbf{F}(\Delta t_k) = \begin{pmatrix} 1 & \Delta t_k & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t_k \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

and W_k is system process white noise.
The output equation can be written as

$$Z_k = Z_k + \hat{Z}_K = H X_k + \hat{Z}_K \tag{2}$$

where $\hat{Z}_K$ is measurable white noises. The output equations can be linearized at position k as follows:

$$\mathbf{Z}_k = \begin{pmatrix} \frac{\partial d_1}{\partial x} & 0 & \frac{\partial d_1}{\partial y} & 0 \\ \frac{\partial d_2}{\partial x} & 0 & \frac{\partial d_2}{\partial y} & 0 \\ \frac{\partial d_3}{\partial x} & 0 & \frac{\partial d_3}{\partial y} & 0 \\ \frac{\partial d_4}{\partial x} & 0 & \frac{\partial d_4}{\partial y} & 0 \end{pmatrix} \tag{3}$$

where

$$\frac{\partial d_i}{\partial x} = \frac{x - x_i}{\sqrt{(x - x_i)^2 + (y - y_i)^2}} \tag{4}$$

$$\frac{\partial d_i}{\partial y} = \frac{y - y_i}{\sqrt{(x - x_i)^2 + (y - y_i)^2}} \tag{5}$$

for i = 1, 2, 3, 4.
The following steps describe equations that need to be evaluated on-line for EKF.

```
Algorithm $EKF$ Tracking:
1: Project the state ahead: $\hat{X}_k^- = F(\Delta t_k)\hat{X}_{k-1}$, and calculate $Z_k^-$ based on $\hat{X}_k^-$
2: Project the error covariance ahead:
       $P_k^- = F(\Delta t_k) P_{k-1} F(\Delta t_k)^T + W Q_{k-1} W^T$
3: Compute the Kalman gain:
       $K_k = P_k^- H_k^T (H_k P_k^- H_k^T + V_k R_k V_k^T)^{-1}$
4: Update estimation with measurements:
       $\hat{X}_k = \hat{X}_{k-1}^- + K_k (Z_k - Z_k^-)$
5: Update the error covariance:
       $P_k = (I - K_k H_k) P_k^-$
6: Repeat and go to Step 1.
```

In the algorithm, W and V are the Jacobian matrices of partial derivatives of state and output functions with respect to the process noise and measurement noise, respectively. P, R and Q are the covariance matrices of the error in the state estimate, measurement noise, and process noise, respectively.

3.3 Outlier Rejection

The third module in our tracking algorithm implements outlier rejection. Since the EKF provides an estimate of the current position based on its state estimate, CH can compute an estimate of the value of any distance measurement from tasking sensor to moving target. The difference between this estimate and the actual measurement defines a residual r ($r^2 = (d(i) - d)^2$) of each measurement. If $r^2 > \gamma$ (an empirically-selected parameter), then we say that the measurement is an $outlier$.

The residual is computed based on the EKF's state. If this fraction significantly exceeds the fraction of outliers we expect to receive, we then declare that the EKF is in a bad state. At that stage, we look to our least squares model for possible recourse. That is, we compute the residual of the LSQ output with respect to the most recent measurement. If this residual r_{lsq} is less than the EKF's residual r_{lsq}, then we reset the filter with the least squares estimate as our position. If the least squares residual is higher, then we do nothing, and we expect this testing to continue in future time-steps.

4 Adaptive sensor scheduling

In existing work for target tracking in WSN, usually the tasking sensors are scheduled based on the uniform sampling interval, ignoring the changing of the target dynamics and obtained estimation accuracy. Here, we propose an adaptive sensor scheduling strategy by jointly setting up distribute dynamic clustering , selecting the tasking sensor, and determining the sampling interval according to the predicted target moving velocity.

Suppose the current time step is k, CH has state estimation X_{k-1} and estimation covariance matrix P_{k-1} of the time step $k-1$. It first updates the state estimation by incorporating its new measurement Z_K (from tasking sensor i) using the EKF algorithm described in Section 3. Then it uses the sensor scheduling algorithm to select the next tasking sensor i and the next sampling interval Δt_k such that the sensor i can undertake the sensing task at the time $t_{k+1} = t_k + \Delta t_k$. We suppose Δt_k should be in the range $[T_{min},\ T_{max}]$, where T_{min} and T_{max} are the minimal and maximal sampling intervals.

If sensor i is selected with the sampling interval Δt_k, defining the sampling interval based on the predicted target moving velocity. The predicted velocity V of time step k is based on state estimation $X_K = [x, y, \dot{x}, \dot{y}]$, defined as:

$$V^2 = V_x^2 + V_y^2 = \begin{bmatrix} 0 & 0 & 1 & 1 \end{bmatrix} X_k \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}^T \tag{6}$$

Give a predefined threshold V_0 and sampling interval Δt_k defined as:

$$\Delta t_k = \begin{cases} T_{max}, & \text{if } V/V_0 \le 1, \\ T_{max} \times [V_0/V], & \text{if } 1 < V/V_0 < 5, \\ T_{min}, & \text{if } V/V_0 \ge 5. \end{cases} \tag{7}$$

The proposed adaptive sensor scheduling algorithm is described as follows:

a. If the predicted velocity is less than the threshold velocity, it means that the target moving at a low velocity, and should be keeping maximal sampling interval T_{max}.

b. If the predicted velocity exceeds the threshold velocity, then it means that the target moving at a quick velocity. If we adjust the sampling interval according to the predicted tracking moving velocity, it can achieve satisfactory tracking accuracy.

c. If the predicted velocity exceeds five times threshold velocity, then the target moved quickly, it should be obtained a bad tracking accuracy, so we use the minimal sampling interval T_{min}. At the same time, Outlier Rejection which described in Section 3.3 would be used.

5 Simulation results

In this paper, we use the Position-Velocity model explained in Section 3. Simulations are done to validate and characterize the performance of the proposed adaptive sensor scheduling algorithm by $Matlab7.0$. The monitored field is $250m \times 250m$ and covered by 3×3 mesh sensors. For the sensor, we suppose communication radius of sensor $r = 150m$, $T_{min} = 0.1s$ and $T_{max} = 0.5s$.

For the target, we also assume motion trajectory as follow:

$$X(t) = 6t + 20;\ Y(t) = 75\sin(0.3t) + 120.$$

It travels from $(20, 120)$ to $(200, 150)$. The predefined threshold is $V_0 = 25m/s$. We compare the performance of this proposed adaptive sensor scheduling with the non-adaptive sensor scheduling scheme in [4].

Fig.2 shows that LSQ is in initializing, and trajectories departure actual path. After 4s, estimated target trajectories by using EKF convergent to the actual path. Compared with the non-adaptive, estimated target trajectory which is used the adaptive sensor scheduling scheme is closer to the actual path. Fig.3 shows tracking accuracy achieved by different sensor scheduling schemes, tracking accuracy is almost lowed by $10m$. Compared with the non-adaptive, the adaptive sensor scheduling scheme schedules sensors with the highest frequency and can achieve the better tracking accuracy quickly, so it can achieve significant improvement tracking accuracy.

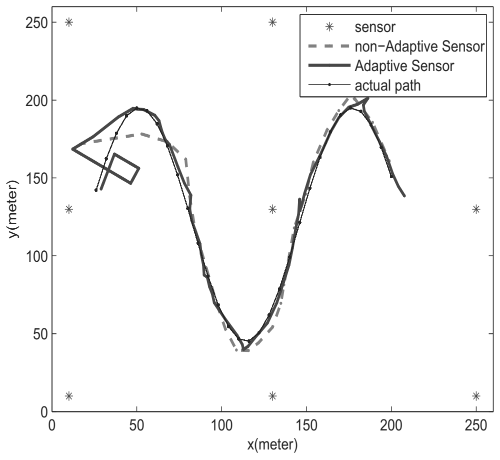

Figure 1. Estimated target trajectory using different sensor scheduling schemes

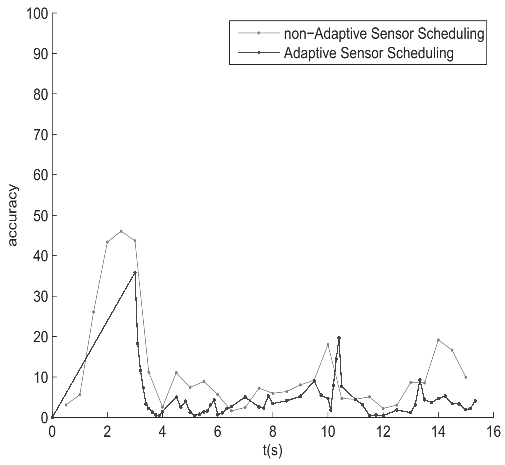

Figure 2. Tracking accuracy achieved by different sensor scheduling schemes

6 Conclusions

This paper presented an adaptive sensor scheduling strategy by jointly setting up distribute dynamic clustering, selecting the tasking sensor, and determining the sampling interval according to the predicted tracking moving velocity. The approach employed Least-Square (LSQ) used in initializing , Extended Kalman Filter (EKF) based on estimation technique to predict the tracking accuracy, and adaptive sampling in velocity prediction. Simulation results demonstrated that, compared to the non-adaptive approach, the proposed approach can achieve significant improvement tracking accuracy.

References

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless sensor networks: A survey,” Computer Networks, vol. 38, no. 4, pp. 393-422, 2002.

[2] F. Zhao, J. Liu, L. Guibas, and J. Reich, “Collaborative signal and information processing: An information directed approach,” Proceedings of the IEEE, vol. 91, no. 8, pp. 1199-1209, 2003.

[3] W. Chen, J. Hou, and L. Sha, “Dynamic clustering for acoustic target tracking in wireless sensor networks,” In Proceedings of the 11th IEEE International Conference on Network Protocols (ICNP03), pp. 258-271, Atlanta, Georgia, USA, Nov 2003.

[4] A. Smith, H. Balakrishnan, M. Goraczko, and N. Priyantha, “Tracking moving devices with the cricket location system,” In 2nd International Conference on Mobile Systems, Applications and Services (Mobisys), pp. 190-202, Boston, MA, Jun 2004.

[5] W. Xiao, J. K. Wu, and L. Xie, “Adaptive sensor scheduling for target tracking in wireless sensor network,” Proceeding of SPIE, vol. 5910, pp. 59100B-1-9, 2005.

[6] F. Xue, Z. Xue, and X. R. Zhang, “Decentralized information particle filter for passive tracking in sensor networks,” International Conference on Communications and Networking, pp. 406-408, Beijing, China, Oct 2006.

[7] G. Welch, G. Bishop, “An introduction to the kalman filter,” UNC-Chapel Hill, TR95-041, Apr 2004.

[8] Y. F. Zhu, A. Shareef, “Comparisons of three kalman filter tracking algorithms in sensor network,” International Workshop on Networking, Architecture, and Storages(IWNAS’06), Shenyang, China, Aug 2006.

[9] I. R. Petersen, A. V. Savkin, “Robust kalman filtering for signals and systems with large uncertainties,” Springer Verlag, 1999.