Distributed Interfering Sensor Scheduling Scheme for Target Tracking

Fan Zhang*, Jiming Chen*†, Hongbin Li*, Youxian Sun* and Xuemin (Sherman) Shen†
*State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou, 310027 China
Email: {zhangfan, hbli, yxsun}@iipc.zju.edu.cn
† Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada
Email: {jmchen, xshen}@bbcr.uwaterloo.ca

***Abstract*—For target tracking applications, active ultrasonic sensors may suffer from inter-sensor-interference when these highly dense deployed sensors are not scheduled. In this paper, we propose a dynamic distributed sensor scheduling (DSS) scheme, where the tasking sensor is elected spontaneously from the sensors with pending sensing tasks in a distributed way via random competition by using Carrier Sense Multiple Access (CSMA)-like fashion, and releases the channel immediately when ranging task is done. Both simulation results and testbed experiment demonstrate the efficiency of DSS scheme in terms of system scalability and tracking performance.**

I. Introduction

Wireless sensor networks (WSNs) have been considered as a promising technique for area surveillance applications [1] [2] [3], and target localization/tracking is essential for these applications. Many approaches have been proposed for target tracking within WSNs [3] [4] [5] [6] [12]. According to target behavior, most of the previous works can be categorized into two classes: cooperative [3] [4] [5] [6] and non-cooperative [7] [12]. A cooperative target is part of the network and emits certain forms of physical signals that reveal its presence or report its own identification. In the non-cooperative scenario, however, there exists no information exchange between the target and the network infrastructure. Therefore, sensor nodes need to detect and identify the target *actively* by emitting energy and measuring the feedback. Our previous work [7], a tracking system aiming at non-cooperative targets, utilized passive infrared sensors for target detection and the active ultrasonic sensors for ranging.

In non-cooperative tracking systems using ultrasonic sensors, there is severe Inter-Sensor Interference (ISI) when nearby active sensors work simultaneously at high frequency. Such interference results in erroneous sensor readings and leads to unacceptable estimation results. For the active ultrasonic sensors, there are two types of ISI, direct ISI and indirect ISI. Direct ISI occurs when the sound wave propagates directly from a transmitting ultrasonic sensor to another receiving ultrasonic sensor. Indirect ISI happens when the sound wave propagates from a transmitter to another receiver via the reflection or diffusion by other objects (including the targets). Fig. 1 shows an example of ISI, where Sensor 3's ranging process periodically collides with that of Sensor 1. Sensor 3 always is interfered by sensor 1 and can not get accurate measurement signal. An occasional ranging task

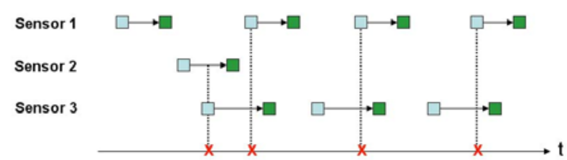


Fig. 1. Inter-Sensor Interference in Ultrasonic Ranging

initiated by sensor 2 is interfered by the signal from Sensor 3. Erroneous ranging measurements are even more harmful than missing the reflected signal, since the target estimator accepts the measurement with high confidence. Therefore, sensor scheduling is needed to ensure that, during any time, only one sensor in an ISI region can work to detect the target.

Sensor scheduling, also referred as sensor selection or sensor management, concerns turning on the right sensors at the right time to achieve desirable performance with minimal energy consumption. Some previous works have addressed the sensor scheduling problem for different tracking systems in WSNs [8] [9] [10] [11]. However, most of them mainly study the tradeoff between tracking performance and energy consumption. In [8], this problem is formulated as a partially observable Markov decision process, and Monte Carlo method is developed using a combination of particle filters for belief-state estimation and sampling based Q-value approximation for lookahead. Adaptive sensor activation [9] selects the next tasking sensor and its associated sampling interval based on the prediction of tracking accuracy and energy cost. Priority list sensor scheduling [10] facilitates efficient distributed estimation in sensor networks, even in the presence of unreliable communication, by prioritizing the sensor nodes according to local sensor schedules based on the predicted estimation error. However, all the methods mentioned above can not be applied directly to the ISI problem for active sensors.

In [12], two sensor scheduling schemes for active sensors are proposed. In the periodic sensor scheduling (PSS) scheme, each ultrasonic sensor detects the target in turn, within the predefined time slots assigned to it. A critical drawback of this sensor scheduling scheme is the existence of empty measurement, where a scheduled sensor may not be in the vicinity of the target. As a result, the system expects lower tracking accuracy and wastes of energy. Whereas in the

CHINACOM 2010, August 25-27, Beijing, China

DOI 10.4108/chinacom.2010.49

TABLE I
NOTATION DEFINITIONS

Symbol	Definition
$\lvert \cdot \rvert$	$\lvert \cdot \rvert$ represents the cardinality of a set.
V	set of the sensor nodes, i.e., V={1,2,...,$\lvert V \rvert$}.
$[0, T]$	duration that the target in the monitored area.
$x(t)$	position of the target at time t, and $t \in [0, T]$.
x_i	position of sensor node i, and $i \in V$.
T_d	die-out time of the ultrasonic wave in a ranging operation.
R	sensing range of the ultrasonic sensors.
$f(t)$	function that record the process of the scheduling during $[0, T]$, the detailed definition is given in Equation 1 and $f : [0, T] \rightarrow V \cup \{0\}$.
N	total number of nonzero element in $f(t)$, means the number of the selection.
n_i	the ith nonzero element in $f(t)$, means the ith tasking node. So $i = 1, 2, ..., N$, $n_i \in V$.
t_i	time of the ith tasking node selection, i.e, $f(t_i) = n_i$.

adaptive sensor scheduling scheme, the next tasking ultrasonic sensor is selected adaptively according to the state prediction of the target. In this scheme, each node needs to know the positions of its neighbors, and the sensor selection process is very computation intensive, since the current tasking node takes the complex calculation for node selection. The considerable computation time may cause delay to the next step sensing, deteriorate tracking accuracy and even lead to target loss. Meanwhile, due to the distributed nature of WSNs, the previously mentioned scheduling schemes are less applicable since scheduling has to be performed in a distributed way to ensure scalability.

In this paper, we introduce a distributed sensor scheduling (DSS) scheme for tracking system with ultrasonic sensors based on the classic CSMA (carrier sense multiple access). The tasking sensor node is elected spontaneously from the sensor candidates in a distributed way via random competition. As soon as the ranging task is done, the channel is released immediately for other pending sensing tasks. Therefore, the main feature of the DSS scheme is its effectiveness for large scale networks and robustness to dynamic topology changes.

The remainder of the paper is organized as follows: Section II presents the DSS scheme. Sections III and IV evaluate the scheme with extensive simulation and test-bed experiment. Finally, conclusions and future work are given in Section V.

II. Distributed Sensor Scheduling Scheme

Typically, each sensor node in WSNs has short sensing range and the on-board processor is limited both in memory and processing speed. In our tracking system, each sensor node is built with one Passive InfraRed (PIR) sensor, multiple ultrasonic ranging sensors and one processing / communication board. The sensor nodes only take charge of detecting / sensing the target and transmitting the sensing results to the central computer. An Extended Kalman Filter (EKF) is applied to the central computer to estimate the position of the target. In this paper, single target tracking is considered and we only concern how to sense the target efficiently via the sensor nodes negotiation.

We first define some notations in Table I. The definition of function f is:

$$f(t) = \begin{cases} i, \text{ select node } i \text{ as the tasking node at time } t \\ \\ 0, \text{otherwise} \end{cases} \quad (1)$$

In order to avoid ISI between ultrasonic sensors, only one sensor node is tasked to actuate its ultrasonic sensors for range measurement each time. And the time difference between two successive measurement epoches should be larger than T_d. Simulation results in [7] reveal that the tracking performance can benefit from higher sampling frequency. With view to the existence of empty detection when the scheduled sensor is not in vicinity of the target, we define another function $g(t)$ to represent the effective scheduling:

$$g(t) = \begin{cases} 1, \ f(t) \neq 0 \text{ and } \|x_{f(t)} - x(t)\| < R \\ \\ 0, \text{otherwise} \end{cases} \quad (2)$$

Therefore, the ISI problem could be converted into an optimization form:

$$\begin{aligned} \text{maximum} \quad & \frac{1}{T}\int_0^T g(t)dt \\ \text{subject to} \quad & t_i - t_{i-1} \geq T_d \\ & i = 2, 3, ..., N \end{aligned} \quad (3)$$

In fact, by considering the ranging operation of a sensor node as the occupation of a shared channel, the ISI problem among active ultrasonic sensors in WSNs can be converted to the problem of multiple access in a shared channel. Hence the schemes for MAC can be used to solve the ISI problem.

A common MAC paradigm is TDMA (time-division multiple access) which schedules transmission times of all nodes to occur at different times. The PSS scheme mentioned above is kind of this fashion. However, TDMA has many disadvantages when it is applied to WSNs. First, it often requires a centralized node to a collision-free schedule. Furthermore, developing an efficient schedule with a high degree of channel reuse is very difficult. Second, TDMA needs clock synchronization. Although clock synchronization is an essential feature of many sensor applications, tight synchronization incurs high energy overhead because it requires frequent message exchanges. Third, sensor networks may undergo frequent topology changes because of time-varying channel conditions, physical environmental changes, battery outage and node failures. Handling dynamic topology changes is expensive, possibly requiring a global change.

Another classic MAC protocol is CSMA, which is a probabilistic protocol. In a nutshell, CSMA verifies the absence of other traffic before transmitting in a shared channel. It is popular because of its simplicity, flexibility and robustness. It does not require clock synchronization and global topology information, and dynamic node joining and leaving are handled gracefully without extra operations. Because of these features, sensor nodes negotiate with each other using CSMA-like fashion is our proposed scheme.

The main idea of DSS scheme is that when a node has a pending ultrasonic sensing request, it checks if there is already an occupation announced by other node in recent σ millisecond. If no occupation is recorded, the node broadcasts its own message to announce occupation in the upcoming σ millisecond and then conducts target range sensing. Otherwise, the node waits a bounded random time for next round occupation. Obviously, the minimal σ should be equal to T_d . The detailed procedure is summarized below:

- a) In initial state, all the nodes are in sleep mode; when a mobile target enters the monitored area, sensor nodes close to the target will be activated by the PIR sensor. As the target moves along the trace in the area, some activated sensor nodes may go back to sleep mode because the target moves out of its sensing region. Also, there are some newly activated sensor nodes;
- b) Once a sensor node is activated, it will calculate a random $T_{backoff}$, and then start its *delay timer* with interval $T_{backoff}$. Let
$$T_{backoff} \in [T_{min}, T_{max}] \text{ and } T_{min} = T_d$$
- c) As soon as the node with the smallest $T_{backoff}$ triggers its *delay timer*, it will broadcast a DETECT message to the other activated nodes and become the tasking node to sense the target. Any node that overhears the DETECT message will immediately give up its declaration as the tasking sensor node by restarting the *delay timer* with a new random $T_{backoff}$. The $timer_{delay}$ in the tasking sensor node is also restarted with new random $T_{backoff}$.

Without loss of generality, it is assumed there is no transmission delay between sensor nodes. Therefore, the DETECT message broadcasted by the selected node can be used as a locally synchronization signal to make all the activated nodes start their *delay timer* at the same time. It is easy to see that the computation burden is distributed among the activated sensor nodes and the sensor selection is totally distributed. Another advantage of the DSS scheme is that each node does not need to know the positions of its neighboring nodes and thus conserving the limited memory for other processing.

In theory, it is almost impossible that more than one sensor nodes broadcast DETECT message at the same time because the $T_{backoff}$ is randomly selected. In practice, however, the $T_{backoff}$ can only be selected from a finite discrete set due to quantization constraint. For one sensor node, we have

$$T_{backoff} \in \{T_0, T_1, \ldots, T_{M-1}, T_M\}$$

$$T_0 = T_{min}, T_M = T_{max}$$

$$T_i - T_{i-1} = \Delta t, i = 1, 2, \ldots, M$$

where Δt is a constant value denoting the resolution of the *delay timer* which depends on the sensor nodes and it equals to $1ms$ in our system. In order to avoid the situation that more than one nodes sense the target at the same time, the node with the smallest ID will be chosen as the tasking node among the nodes whose *delay timer* are triggered simultaneously, and Algorithm 1 is the pseudo-code for the DSS scheme. The parameter M denotes the size of candidate set for $T_{backoff}$. Larger M indicates that sensor nodes are less possible to collide with each other, but the expected backoff time may be larger and result in sensing latency.

Algorithm 1 Distributed Sensor Scheduling Algorithm

Input: the trace of the target $x(t)$
Output: the sensor scheduling results $f(t)$
Initialization: $f(t) = 0, t \in [0, T]$;
//det_send means broadcast the DETECT message.
//$det\_recv(j)$ means receive the DETECT message from node j.
for target moves in the monitored area **do**
  if target moves into the sensing range of node i **then**
    sensor i is activated from the sleep mode;
  end if
  for each activated node i **do**
    generate a random $T_{backoff}$;
    start the *delay timer* with $T_{backoff}$;
    wait for det_send or $det\_recv(j)$;
    if not det_send and $det\_recv(j)$ at time t **then**
      stop the *delay timer*;
      generate a new random $T_{backoff}$;
      start the *delay timer* with $T_{backoff}$;
      set $f(t) = j$;
    end if
    if det_send and not $det\_recv(j)$ at time t **then**
      sensing the target;
      generate a new random $T_{backoff}$;
      start the *delay timer* with $T_{backoff}$;
      set $f(t) = i$;
    end if
    if det_send and $det\_recv(j)$ at time t **then**
      if $i > j$ **then**
        stop the *delay timer*;
        generate a new random $T_{backoff}$;
        start the *delay timer* with $T_{backoff}$;
        set $f(t) = j$;
      else
        sensing the target;
        generate a new random $T_{backoff}$;
        start the *delay timer* with $T_{backoff}$;
        set $f(t) = i$;
      end if
    end if
  end for
  if target moves out of the sensing range of node i **then**
    sensor i goes back to sleep mode;
  end if
end for

III. Simulation Results

We evaluate the proposed DSS scheme with the simulation. We compare the tracking performance of the DSS scheme with the PSS scheme.

In the simulation, we model the monitored area as a grid map. A movement trace of the mobile target is generated with the Random WayPoint mobility model (RWP) [13]. An estimation error at one point in the trace is defined as the geographic offset between the estimated position and corresponding true position

$$e(t) = \|\mathbf{x}(t) - \widetilde{\mathbf{x}}(t)\| \tag{4}$$

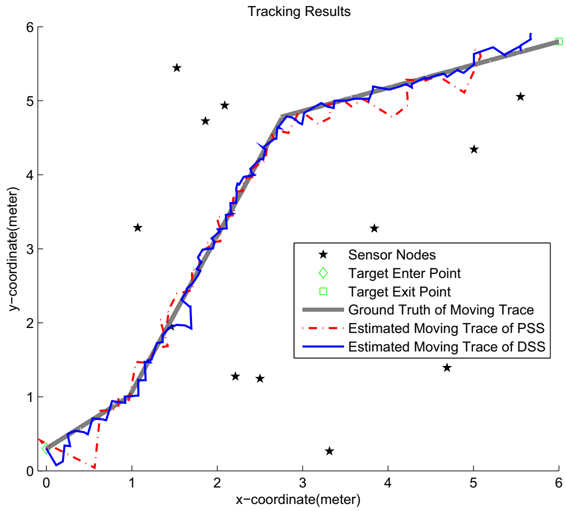


Fig. 2. Simulated Trajectories of PSS and DSS Schemes

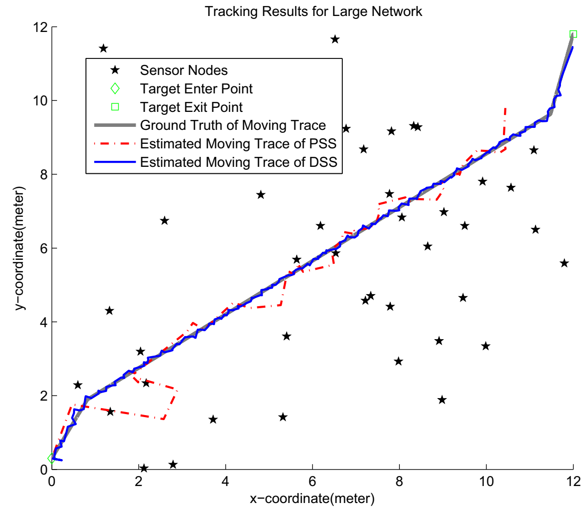


Fig. 3. Simulated Trajectories for Larger Network

where $\mathbf{x}(t)$ is the ground truth position vector and $\widetilde{\mathbf{x}}(t)$ is the estimated position.

The mean tracking error is defined as the averaged error of all the points in the trace

$$\overline{e} = \frac{1}{(t_K - t_0)} \Sigma_{i=1}^{K} (t_i - t_{i-1}) e(t_i) \tag{5}$$

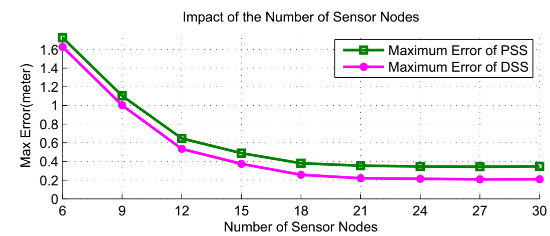
where t_0 and t_K are the start time and end time, respectively and K denotes the total numbers of estimations.

The presented results are averaged over 50 independent simulation runs for high confidence. The following table illustrates the default simulation setup:

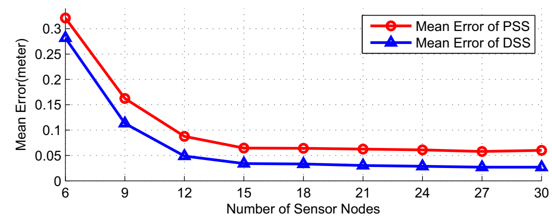
Parameter	Description
Field Area	$6 \times 6m^2$
Sensing Noise Model	$\mathbb{N}(0, 0.0025)$
Number of Sensor Nodes	12, uniformly deployed
Target Velocity	$\mathbb{U}(0.3, 0.7)m/s$
PIR range and angle	$0 \sim 3m$, $\pm\pi$
Ultrasonic range and angle	$0 \sim 3m$, $\pm\pi$
T_d	$30ms$
M	15

For the sake of clarity, we assume that the sensing angle of PIR sensor and ultrasonic sensor are both $\pm\pi$, i.e., that the sensing model is a circle.

We compare the PSS scheme with the proposed DSS scheme. Fig. 2 shows the target moving traces estimated by PSS and DSS. It can be seen that the estimated trace generated by DSS is closer to the true one, and the corresponding tracking error of these two estimated traces are $0.1131m$ and $0.0575m$, respectively. For a larger network, the tracking results are shown in Fig. 3. The monitored area is $12 \times 12m^2$ with 48 sensor nodes deployed. It can be seen see that the DSS scheme still performs well under such conditions. On the other hand, PSS scheme almost lost the target during the first corner and at the end of the trace because of the empty detections. The corresponding tracking error of PSS and DSS are $0.3651m$ and $0.0534m$, respectively.



(a) Impact of node number on max error

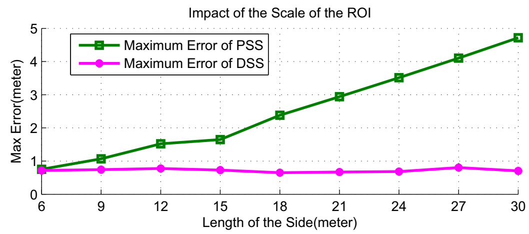


(b) Impact of node number on mean error

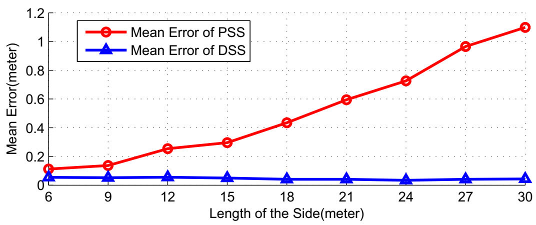
Fig. 4. Impact of Node Number on Tracking Error

A. Impact of the number of sensor nodes

We compare the DSS scheme with PSS scheme under a different setting of sensor quantity, ranging from 6 to 30. Fig. 4 shows that (i) the DSS scheme outperforms the PSS scheme, and (ii) with the increase number of deployed sensor nodes, the tracking error for both schemes are reduced, and it seems the error converge to a constant when the node number is very large. This is because the uncovered area is large when less number of nodes are deployed, which leads to high probability of losing the target. On the other hand, the area coverage is saturated when the node number is large enough, so that any more sensor deployment does not help in terms of tracking performance.

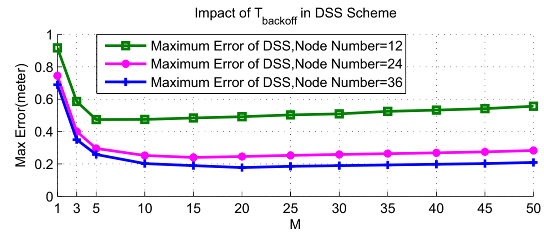


(a) Impact of ROI scale on max error

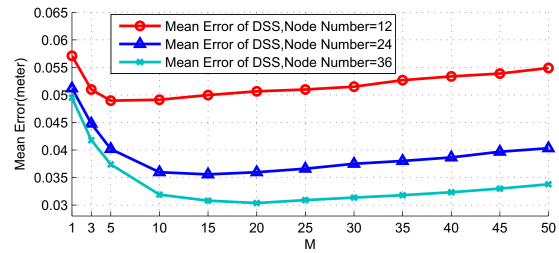


(b) Impact of ROI scale on mean error

Fig. 5. Impact of ROI Scale on Tracking Error



(a) Impact of $T_{backoff}$ on max error in DSS Scheme



(b) Impact of $T_{backoff}$ on mean error in DSS Scheme

Fig. 6. Impact of $T_{backoff}$ on Tracking Error in DSS Scheme

B. Impact of the scale of ROI

ROI means the region of interest, i.e. the monitored area. We compare the performance of both schemes under different scales of ROI, with same node density. For each scale, we define the number of nodes by

$$N_{node} = \alpha l^2$$

where α is a density factor, and l is the side length of the square monitored area. In the simulation, l increases from 6 meters to 30 meters in steps of 3 meters, and α is set to $1/3$.

Fig. 5 shows: (i) DSS scheme is very robust to the network scale (error keeps constant when the network scale increases) since it only schedules the activated nodes, while the error with the PSS scheme increases quickly because of the growing number of empty detections; and (ii) the DSS scheme is superior to the PSS scheme, especially when the network scale is very large.

C. Impact of M in distributed sensor scheduling scheme

Fig. 6 illustrates the impact of M to the DSS scheme. Simulation results indicate that: (i) there exists an optimal M_o, when M equals to M_o, the tracking error is the minimum; (ii) when M is larger than M_o, the tracking error increases with the increase of M because the average sampling period increases; (iii) when M is smaller than M_o, the tracking error reduces with the increase of M. The reason for this phenomenon is that the probability of more than one sensor nodes triggering their *delay timer* at the same time increases with the smaller M. Therefore, only nodes with the smallest ID will be selected to sense the target, which results in increase of accumulate error; and (iv) from the simulation results, we can see

$$M_o \propto \alpha \tag{6}$$

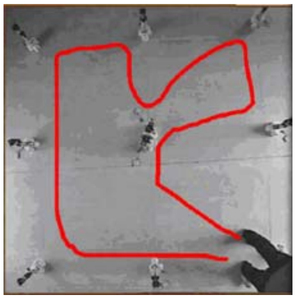
Equation 6 indicates that the optimal M_o is approximately proportional to the node density. This is because larger node density creates chances for more than one sensor nodes triggering their *delay timer* at the same time, so M should be properly chosen according to the node density.

IV. Testbed Experiment

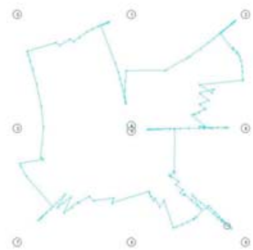
A $5.2m \times 5.2m$ testbed has been built, as shown in Fig. 7(a) to support testbed-scale experiments. We mount an overhead camera right up on the ceiling of the testbed to capture the experiments or extract accurate positions of interested objects. Each sensor node is integrated with one main board, one PIR sensor and three active ultrasonic sensors to realize the tracking. The main board, which is mainly composed of Atmel128L [14] as core unit and CC2420 [15] as communication chip is used as an communication/computation module. The PIR sensor can detect the changes of infrared energy radiation from the environment due to the movement of the target and outputs a high level signal which is used as an external interrupt to activate the node. The ultrasonic sensor has a maximum range of $3m$ and the effective angle is $\pi/3$. All the three ultrasonic sensor are connected with the main board via UART connection.

In our experiments, 10 sensor nodes are deployed in the testbed using 3×3 grid form with 2.2 meters displacement. Specifically, two nodes are placed in the middle of the area to provide with $\pm\pi$ coverage. The 3 ultrasonic sensors are fixed on the pipe with two different configurations, to cover either $\pi/2$ or π angle range. The node that covers $\pi/2$ is placed at the corner of the field, and The nodes cover π are placed in other positions.

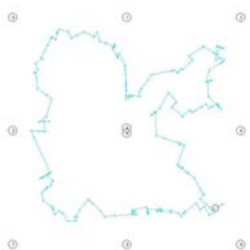
In the experiments, a person is considered as the mobile target to track. Fig. 7(a), taken by the overhead camera, shows the sensor node deployment and target trajectory. We put



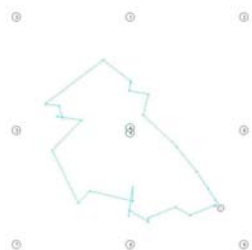
(a) Real human trajectory



(b) Tracking result of PSS scheme in 10 nodes scale



(c) Tracking result of DSS scheme



(d) Tracking result of PSS scheme in 40 nodes scale

Fig. 7. Testbed experiments for PSS and DSS schemes

a sequence of marks on the testbed as the trails to follow. And the person, put his feet on the marks at one pace per second, to generate identical target trails for each test. The estimated trajectories of PSS scheme and DSS scheme are shown in Fig. 7(b) and Fig. 7(c). Compared with the true trajectory, we see both estimated trajectories are very close to the true one. The overall tracking error is about $19cm$ for DSS and $28cm$ for PSS. Due to the relatively small network scale, using PSS scheme gives enough effective measurements for target updates. Next we evaluate the number of effective measurements. The PSS gave us 97 effective measurements and the DSS gave us 216, which reveals the existence of empty detection for PSS scheme. When the target moves within the field, DSS scheme always activates sensors in the vicinity of the target and thus makes most of the measurements effective. The PSS scheme, however, emphasizes fairness among sensors regardless of the target position and yields large number of empty detections.

We further test the performance of PSS scheme under larger scale network settings. Besides the 10 deployed sensors, we add another 30 *virtual* nodes working around the testbed with the same deployment and sharing the channel. The tracking result is shown in Fig. 7(d). Noticeably, the tracking error of PSS is about $1.15m$, which is very large and the corresponding number of effective measurements is only 27.

V. CONCLUSION

In this paper, we have presented a distributed sensor scheduling (DSS) scheme, which is inspired by CSMA, for active sensors in wireless sensor network to solve the ISI problem. By with the scheme, the computation burden for each sensor is reduced significantly and the scalability can be guaranteed. It has been demonstrated that tracking with DSS scheme outperforms the PSS scheme, especially when the network is very large. There are several issues remaining for future study, our scheme focuses on the effective measurement only with the consideration of the target presence. In addition, the DSS scheme selects the node with the smallest ID when there is a collision, and this may lead to accumulated error. The research to develop more impactful strategies is under way. More efficient sensor scheduling scheme could be designed with the estimation/pridiction techniques such as EKF, unscented Kalman filter and particle filter. Tracking and adaptive sensor scheduling for multiple targets using multiple modalities are our future study.

ACKNOWLEDGEMENT

The research is supported in part by NSFC-Guangdong joint Project grant no. U0735003; NSFC grant nos. 60736021 and 60974122; and the Fundamental Research Funds for the Central Universities no.2009QNA5007

REFERENCES

[1] G. Simon, M. Maróti, Á. Lédeczi, G. Balogh, B. Kusý, A. Nádas, G. Pap and et al. Sensor Network-based Countersniper System. In *Proceedings of ACM SenSys*, Baltimore, MD, USA, Nov. 2004.

[2] G. Werner-Allen, J. Johnson, M. Ruiz, J.Lees. Monitoring Volcanic Eruptions with a Wireless Sensor Network. In *Proceedings of EWSN*, pages 108-120, Istanbul, Turkey, Jan.-Feb. 2005.

[3] A. Terzis, A. Anandarajah, K. Moore, I-J. Wang. Slip Surface Localization in Wireless Sensor Networks for Landslide Prediction. In *Proceedings of IPSN*, pages 109-116, Nashville, Tennessee, USA, Apr. 2006.

[4] A. Smith, H. Balakrishnan, M. Goraczkoet, and N. Priyantha. Tracking Moving Devices with The Cricket Location System. In *Proceedings of ACM MobiSys*, pages 190-202, Boston, MA, USA, Jun. 2004.

[5] B. Kusy, A. Ledeczi, and X. Koutsoukos. Tracking Mobile Nodes Using RF Doppler Shifts. In *Proceedings of ACM SenSys*, pages 29-42, Sydney, Australia, Nov. 2007.

[6] L. Klingbeil, T. Wark. A Wireless Sensor Network for Real-time Indoor Localisation and Motion Monitoring. In *Proceedings of IPSN*, pages 39-50, St. Louis, Missouri, USA, Apr. 2008.

[7] H. Li, D. Miao, J. Chen, Y. Sun. X. Shen. Networked Ultrasonic Sensors for Target Tracking: An Experimental Study. In *Proceedings of IEEE GlobeCom*, Honolulu, Hawaii, USA, Nov.-Dec. 2009.

[8] Y. He, K. P. Chong. Sensor Scheduling for Target Tracking in Sensor Networks. In *Proceedings of IEEE CDC*, pages 743-748, Atlantis, Paradise Island, Bahamas, Dec. 2004.

[9] J. Chen, K. Cao, Y. Sun and X. Shen. Adaptive Sensor Activation for Target Tracking in Wireless Sensor Networks. In *Proceedings of IEEE ICC*, Dresden, Germany, Jun. 2009

[10] M. F. Huber, U. D. Hanebeck. Priority List Sensor Scheduling using Optimal Pruning. In *Proceedings of The 11th International Conference on Information Fusion*, Cologne, Germany, Jun.-Jul. 2008.

[11] V. Gupta, T. H. Chung, B. Hassibi, and R. M. Murray. On a Stochastic Sensor Selection Algorithm with Applications in Sensor Scheduling and Sensor Coverage. *Automatica*, vol. 42, no. 2, pages 251-260, Feb. 2006.

[12] W. Xiao, J. Wu, L. Xie, L. Dong. Sensor Scheduling for Target Tracking in Networks of Active Sensors. *Acta Automatica Sinica*, vol. 32, no. 6, pages 922-928, Nov. 2006.

[13] D. B. Johnson and D. A. Maltz. Dynamic Source Routing in Ad hoc Wireless Networks. In Tomasz Imielinski and Hank Korth, editors, *Mobile Computing*, vol. 353, Kluwer Academic Publishers, 1996.

[14] Atmel Inc. Atmel Atmega128L Low-Power Mricrocontroller, *http://www.atmel.com/dyn/resources/prod documents/2467S.pdf*.

[15] Texas Instruments. CC2420 Single-Chip 2.4 GHz RF Transceiver, *http://focus.ti.com/docs/prod/folders/print/cc2420.html*.