# An Interactive Optimal Sensor Selection Method for Wireless Sensor Networks

Zhen Song
CSOIS, Department of Electrical and Computer Engineering
Utah State University
Logan, UT, 84322
zhensong@cc.usu.edu

Chellury Sastry
Siemens Corporate Research
Princeton, NJ, 08540
Chellury.Sastry@siemens.com

N. Cihan Taş
Siemens Corporate Research
Princeton, NJ, 08540
Nazif.Tas@siemens.com

***Abstract*— It is important to balance the tradeoff between the energy efficiency and estimation precision for wireless sensor networks (WSNs). Energy costs can be reduced significantly by selecting proper sensors. In this paper, we propose an interactive optimal sensor selection (IOSS) algorithm, to choose proper sensors to observe parameters of physical systems, such as the position of a target. Since the computation costs on the original sensor selection problem are very high, we introduce carefully chosen approximations. Then we formulate the sensor selection problem as a continuous convex optimization problem, which can be solved efficiently. The proposed solution, the IOSS algorithm, is simple, fast, and memory efficient. Our experiments indicate that the algorithm selects the minimum number of sensors allowed by the Caratheodory's theorem. In addition, the algorithm approximates the minimal estimation error predicted by the Cramer-Rao lower bound (CRLB). The algorithm and the approximations has been tested extensively by our simulations and hardware experiments.**

## I. Motivation

### A. Introduction to Wireless Sensor Networks

Wireless sensor networks (WSNs) or smart sensor networks are a key component of machine-to-machine (M2M) communication. Such networks can be used in a wide variety of remote monitoring and control applications that range from environmental and human body monitoring to military surveillance, building automation, industrial monitoring and control, homeland security, air pollution detection [1], wild fire monitoring [2], detection of persons and vehicles in open areas [3], and reconnaissance tasks [4], etc. In typical remote monitoring applications, sensor nodes are scattered in an ad hoc manner over an area of interest. Individual sensor nodes can sense information (measurements) and communicate (by radio) sensed information to other sensors or to a gateway (also referred to as base station or sink in WSN literature). Furthermore, sensor nodes have a limited ability to process information on an on-board CPU, and can store that information in memory. This is the reason why such wireless sensors are sometimes referred to as smart sensors or smart dust. The on-sensor processing and on-sink processing should cooperatively interpret sensor data to observe environments in energy efficient approaches. Although each individual node has limited capability, several such nodes can cooperate to accomplish complex tasks, which are called sensor fusion.

Parameter estimation is an important application for WSNs, where sensor fusion plays an important role. Many physical quantities of interest are either expensive or even impossible to measure by small sensors. For example, we may want to use several low-cost temperature sensors to estimate the position of a fire, instead of using an expensive IR camera. We may also want to use several gas sensors to locate the leaking source of certain invisible gas. The location can not be measured by a physical sensor directly.

To put it succinctly, WSNs provide the ability to connect the physical world to enterprise computing systems, thereby improving business processes and facilitating efficient decision making.

In many applications involving WSNs, there is a delicate trade off between the need to sense and communicate useful information and the requirement to do so in an energy efficient manner. It is expected that a WSN, once deployed, will work for long periods of time with minimal human intervention. In most remote monitoring applications, individual sensor nodes run on a limited supply of energy (from batteries), and hence, excessive on-board processing and (perhaps more importantly) frequent invocation of the sensors radio, rapidly depletes the sensors energy source. Thus, to meet these challenges, in recent years, a major research effort in modern computing has been devoted to the development of sensor network protocols and algorithms with energy-efficient and self-organizing capabilities.

### B. Prior Art

Sensor selection (or sensor scheduling) methods are discussed recently under to context of WSN. Some model free protocols, such as S-MAC protocol [5], schedule sensor's sleep period for energy efficiency purposes. There are also many model-based sensor selection methods. The ideas behind the methods are similar: many physical quantities are distributed and constrained by physical laws. Thus some sensor data can be inferred from other data. The brute force sensor selection approach for target tracking is too expensive, in term of computation and hardware costs. Each sensor has two states: on and off. Thus, the solution space for $n$ sensors has $2^n$ states. It is impossible to use the brute force approach in practice.

In [6], the sensor selection is formulated as a binary integer programming problem and solved by the branch and bound (B&B) method. Based on the tree search, the B&B method is much faster than the brute force method. A sensor selection algorithm for camera-like sensors is proposed in [7], where geometric relations are used to speed up the computation. In the paper, it is observed that "a small number of sensors is sufficient to obtain good estimates."

Some papers consider the sensor selection problem from the aspect of information or estimation theory. The paper [8] presents a sensor selection method based on the Bayesian filter. This is a grid-based method: the region of interest is segmented into many small cells, and the probability of the target within each cell is repetitively updated according to the Bayesian theorem. The method is closely related to entropy-based sensor selection [9] approach.

The rest of the paper is organized as the followings: Sec.II presents our formulation on the sensor selection problem (Sec. II-B), our solution, and the IOSS algorithm (Sec. II-C). Sec. III describes our simulations (Sec. III-A) and hardware experiments (Sec. III-B) with analysis (Sec. III-C). Sec. IV concludes the paper.

## II. Sensor Selection Problem and the Solution

### A. Overview on Our Strategy

As aforementioned, the original sensor selection problem has a large solution space and it is hard to solve. Thus, our strategy is to simplify the problem by several reasonable approximations and solve the approximated problem by a fast optimization approach. The solution of the approximated problem should achieve the minimum estimation errors with the least number of sensors. In other words, we want to approach both the lowest estimation error bound, which is limited by Cramer-Rao lower bound (CRLB), and the minimum number of selected sensors that allowed by the Caratheodory's theorem.

We do not add the two cost functions (cost for estimation errors and number of selected sensors) together and minimize the summation, due to the high computation cost of integer programming: the number of selected sensors is an integer. It will take much more computations to minimize that cost function than our proposed, continuous cost function.

In this paper, we assume that the positions of each sensor node in the network are known. Admittedly, this may be a strong assumption, but we refer to a large volume of prior art that deals with the problem of sensor node localization in a WSN [10], [11].

### B. Problem Formulation

Our method follows the framework of experiment design and it is applicable to general parameter estimation problems, not just limited to the target tracking. However, for presentation purpose, we use a lamp tracking case has the example to interpret our formulation. This is the exact working scenario of our hardware testbed.

Assuming the true positions of the lamp at the time instance $k$ is $\mathbf{q}^*[k]$, and the positions of the $i$th sensor is $\mathbf{r}_i$, where $\mathbf{q}^*[k], \mathbf{r}_i \in \mathbb{R}^m$. The sensor model is in the following form:

$$\begin{aligned} y_i[k] &= f(\mathbf{q}^*[k]; \mathbf{r}_i), i = 1, 2, \cdots, n, && (1)\\ s_i[k] &= y_i[k] + v_i[k], && (2) \end{aligned}$$

where $y_i[k]$ is the nominal reading from the $i$th sensor, and the associated real sensor reading is $s_i[k]$, which is corrupted by the Gaussian noise $v_i[k]$. A widely used energy sensor model is adopt.

*Definition 2.1 (energy model):*

$$\begin{aligned} y_i[k] &= \frac{c_1}{h^2 + d_i^2[k]}, && (3)\\ v_i[k] &\sim \mathcal{N}(0, \sigma_i), && (4)\\ d_i[k] &= \|\mathbf{r}_i - \mathbf{q}^*[k]\|, && (5) \end{aligned}$$

where $h$ is the height of the lamp, $d_i[k]$ is the distance from the sensor to the exact position under the lamp, $c_1$ is a constants, and $\sigma$ is the standard deviation.

In order to reduce the noise $v_i$, the $i$th sensor node measures the $n_i$ light values in the time slot $t_S$, averages them and sends it back to the sink. The averaged light value is $\bar{s}_i[k]$, whose standard deviation is smaller than that of the raw data: $\bar{\sigma}_i^2[k] = \sigma_i^2[k]/n_i$. If we can afford infinite number of samples, we can reject the noise totally. Of course, there is an upper limit on sampling number in practice. For simplicity, we set $n_S$ as the upper limit on the total number of samples for all the sensors in the time slot $t_S$. The intuitive interpretation is as follows. We estimate the position of an event based on sensor measurements. Since sensor measurements are noisy, the event observations are not perfect. Thus, the accuracy of the event position estimate, which is based on sensor measurements, depends on their accuracy. Now, noise in the sensor measurements can be reduced either by modifying sensor design (use better hardware) or by filtering away sensor noise. In the latter approach, one approach could be to take a large number of sensor measurements (samples) and then average them by each sensor to eliminate as much noise as possible.

After receiving all the sensor data, the sink estimates the lamp's position by the standard nonlinear least square (LS) method, and the output is $\hat{\mathbf{q}}_A$, which is also called the 1st position. For a network of $n$ sensors, the $\hat{\mathbf{q}}_A$ is the following:

$$\hat{\mathbf{q}}_A[k] = \text{argmin}_{\mathbf{q}} \frac{1}{2} \sum_{i=1}^{n} (\bar{s}_i[k] - y_i(\mathbf{q}; \mathbf{r}_i))^2. \quad (6)$$

Now, we introduce several approximations to simplify the problem.

- Instead of assigning each sensor a binary value that indicates the "selected" and "non-selected" state, we assign a normalized sampling rate $p_i$ to sensor $i$. That is, $p_i[k] \in [0, 1]$ and $\sum_i p_i[k] = 1$. Thus, the integer programming problem is approximated by a continuous design problem.
- Our cost function is based on Fisher information matrix (FIM), $M$, whose inverse is the CRLB. Ideally, the optimal sampling rate is in this form: $\mathbf{p}^* = \text{argmin}_{\mathbf{p}} \Psi(M(\mathbf{p}; \hat{\mathbf{q}}^*[k]))$. Since $\hat{\mathbf{q}}^*[k]$ is unknown, we replace it by $\hat{\mathbf{q}}_A[k]$.
- The nonlinear sensor model is linearized as $\mathbf{y}[k] = A^T \hat{\mathbf{q}}_A[k]$.

Thus, we simplified the original problem to the following sampling rate optimization problem.

*Definition 2.2 (sampling rate optimization problem):*

$$\begin{aligned} \hat{\mathbf{p}}[k] &= \text{argmin}_{\mathbf{p}} \Psi(M(\mathbf{p}; \hat{\mathbf{q}}_A[k])), && (7)\\ \Psi(M) &= -\ln\det(M), && (8)\\ \text{subject to : } \mathbf{p} &\geq 0, && (9)\\ \mathbf{1}^T\mathbf{p} &= 1, && (10)\\ \mathbf{y}[k] &= A^T \hat{\mathbf{q}}_A[k], && (11)\\ A &= \nabla_{\mathbf{q}} \mathbf{y}|_{\mathbf{q}=\hat{\mathbf{q}}_A[k]} && (12)\\ M &= A\Sigma^{-1}A^T, && (13)\\ \Sigma^{-1} &= \begin{bmatrix} \bar{\sigma_1}^{-2}[k] & 0 & \\ 0 & \bar{\sigma_2}^{-2}[k] & \\ & & \ddots \end{bmatrix}, && (14)\\ \bar{\sigma_i}^{-2}[k] &= \sigma_i^{-2}[k] p_i[k]. && (15) \end{aligned}$$

A natural question to ask is why the $\Psi$ function is defined as $\Psi(M) = -\ln\det(M)$. This function is called D-optimality criterion in the literature of experiment design. Even though there are other optimality criteria available, the unique feature of D-optimization compared to the rest is that the result of the D-optimization is not affected by linear transforms. Remember that $\det(M)$ is the volume of matrix $M$. Actually, when $M$ is a 3 by 3 matrix, $\det(M)$ is the volume of the

parallelepiped constructed by the three column vectors of $M$. In other words, $\det(M)$ is a metric to measure the size of $M$. Since $\det(M^{-1}) = 1/\det(M)$, $\det(M^{-1})$ is minimized when $-\ln\det(M)$ is minimized.

Now we prove that the FIM, $M$, is the inverse of the covariance of estimation error $\mathbf{e}$.

*Theorem 1:* For the linear system $\mathbf{s} = A^T\mathbf{q}^* + \mathbf{v}$, where $v_i$ is a zero mean noise with a standard deviation of $\sigma_i$, we have $M^{-1}$ equals the covariance matrix of the estimation error. That is

$$cov(\mathbf{e}) = M^{-1},$$

where $M = A\Sigma^{-1}A^T$, $\mathbf{e} = \hat{\mathbf{q}} - \mathbf{q}^*$ and $\hat{\mathbf{q}}$ is the best linear unbiased estimator (BLUE) of $\mathbf{q}^*$.

*Proof:* It is well known that $\hat{\mathbf{q}} = (A\Sigma^{-1}A^T)^{-1}A\Sigma^{-1}\mathbf{s}$, if the cost function of weighted LS (WLS) estimation [1] is $\mathcal{J} = \min(A^T\mathbf{q} - \mathbf{s})^T\Sigma^{-1}(A^T\mathbf{q} - \mathbf{s})$.

Because $\hat{\mathbf{q}}$ is BLUE, $E\{\hat{\mathbf{q}}\} = \mathbf{q}^*$.

$$\begin{aligned} cov(\mathbf{e}) &= E\{(\mathbf{e} - E\{\mathbf{e}\})(\mathbf{e} - E\{\mathbf{e}\})^T\} \\ &= E\{(\hat{\mathbf{q}} - \mathbf{q}^* - 0)(\hat{\mathbf{q}} - \mathbf{q}^* - 0)^T\} \\ &= cov(\hat{\mathbf{q}}) \\ &= E\{(M^{-1}M\mathbf{q}^* + M^{-1}A\Sigma^{-1}\mathbf{v} - \mathbf{q}^*) \\ &\quad (M^{-1}M\mathbf{q}^* + M^{-1}A\Sigma^{-1}\mathbf{v} - \mathbf{q}^*)^T\} \\ &= M^{-1}A\Sigma^{-1}E\{\mathbf{v}\mathbf{v}^T\}\Sigma^{-T}A^TM^{-1} \\ &= M^{-1}. \end{aligned}$$

In summary, $cov(\mathbf{e}) = M^{-1}$. ■

Since $M^{-1}$ equals to the CRLB [12], [15], and $\Psi = \ln\det(M^{-1})$, the Eq. 7 minimizes the estimation error and pushes down the estimation errors close to the CRLB.

It is known that, under certain assumptions the D-optimization has a "sensor clustering" effect [15], i.e., after the optimization, most sensors have sampling rates close to 0. In the current literature, this effect is considered undesirable, and different methods have been proposed to compensate this effect [16]. However, we notice that this effect is ideal for our sensor selection purpose.

After the sampling rate optimization, we select sensors whose sampling rates are higher than a threshold $h_S$. The set of selected sensor is $\mathbb{S}_S$.

$$\mathbb{S}_S[k] = \{i | \hat{p}_i[k] \geq h_S\}. \tag{16}$$

Then the sink turns off the sensors that have not been selected and continuously collects the data from those selected sensors. Finally, the so called 2nd position of lamp is estimated by LS method again.

$$\hat{\mathbf{q}}_B[k] = \text{argmin}_{\mathbf{q}} \frac{1}{2} \sum_{i \in \mathbb{S}_S[k]} (\bar{s}_i[k] - y_i(\mathbf{q}; \mathbf{r}_i))^2. \tag{17}$$

The system keep estimating the lamp by the selected sensors, until at certain time, $k+i$, when the 2nd position estimate, $\hat{\mathbf{q}}_B[k+i]$, has a too big error bound, then we restart from the 1st estimation again. In fact, if the target is smoothly moving, we can also restart from the sampling rate optimization.

[1] It is proved that the $\hat{\mathbf{q}}$ computed based on this WLS cost function is better, in the sense that it is closer to CRLB than the LS cost function. The weight $\Sigma$ is optimal [12].

**Part 1: On-sensor computation.**

Receive $t_S$ and $n_i$ from the sink;
Collect $n_i$ samples in the time slot $t_S$, and $\bar{s}_i$ is the average of those samples;
Wait for a small random time, then send $\bar{s}_i$ to sink;

**Part 2: On-sink computation.**

Initially $p_i = 1/n$ and `State`←`selection`;
**if** `State=selection` **then**
  Send $n_i$, $t_S$ to the $i$th sensor;
  Wait for time $t_S$, collect $\bar{s}_i$;
  Estimate parameter $\mathbf{q}^*$ by the nonlinear LS method, and the result is $\hat{\mathbf{q}}_A$;
  **while** *true* **do**
    **if** $\phi_i(\mathbf{p}[k+1]) < m + \eta, i = 1, 2, \cdots, N$ **then**
      exit the while loop;
    **else**
      $p_i[k+1] = p_i[k]\frac{\phi_i(\mathbf{p}[k])}{m}$ ;
    **end**
  **end**
  **if** $p_i \geq h_S$ **then** $n_i = p_i \times t_S$ **else** $n_i = 0$;
  Send $\mathbf{n}$ to the proper sensors and `State`←`tracking`;
**end**
**if** `State=tracking` **then**
  Receive sensor reading $\bar{s}_i$;
  Estimate parameter $\mathbf{q}^*$ by the least square method. The result is $\hat{\mathbf{q}}_B$ and its associated FIM is $M_B$;
  **if** $M_B^{-1}$ *is big* **then** `State`←`selection`;
**end**

**Algorithm 1**: Interactive optimal sensor selection (IOSS) algorithm.

### C. Solution: Interactive Optimal Sensor Selection (IOSS)

The optimization Eq. 7 is formulated as a continuous design problem, which can be solved by a multiplicative algorithm which updates $p_i[k+1]$ as $p_i[k]\phi_i(\mathbf{p}[k])/m$, where $\phi(\mathbf{p}[k]) = \nabla_{\mathbf{p}}\Psi$. Remind $m$ is the dimension of the parameter under estimation. For 2D tracking, $m = 2$. It is proved that this method minimizes the D-optimality criterion [15], [17]. We use this method as a part of our interactive optimal sensor selection (IOSS) algorithm, which is listed as Algorithm 1.

Due to the limited space, we do not present the details on why the D-optimization, which is solved by the multiplicative algorithm, has the sensor clustering effect. We just remind that the Caratheodory's theorem predicts a lower limit on the required number of sensors.

*Theorem 2.3 (Caratheodory's theorem, based on [18], p72):* Let $\mathbb{S}$ be a subset of $\mathbb{R}^n$. Every element $\mathbf{x}$ in $\mathbb{S}$ can be expressed as a convex combination of no more than $n+1$ elements of $\mathbb{S}$. If $\mathbf{x}$ is on the boundary of $\mathbb{S}$, $n+1$ can be replaced by $n$.

**Remark** For our problem, the matrix $M$ is represented by a convex combination of $\mathbf{a}_i\mathbf{a}_i^T$. Here, $\mathbf{a}_i$ is defined as $A^T = [\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_n]$. Since $\mathbf{a}_i \in \mathbb{R}^m$, where $m$ is the number of parameters for the estimation, it is easy to see that $M$ has $m(m+1)/2$ unique entries. For 2D tracking problems, $m = 2$, and $m(m+1)/2 = 3$. The Caratheodory theorem tells us that it is possible to achieve the optimal solution with no more than 4 sensors. It is interesting to see that paper [7] observers that "the estimates obtained by four best sensors are as good as the estimates obtained from all sensors" in their tracking experiments. It seems that this observation is in consistent with ours.

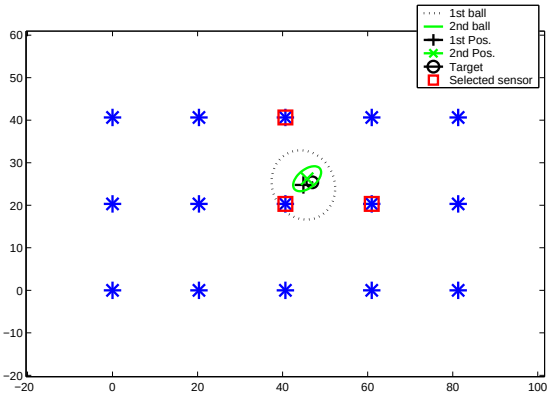


Fig. 1. A simulation based on our IOSS algorithm.

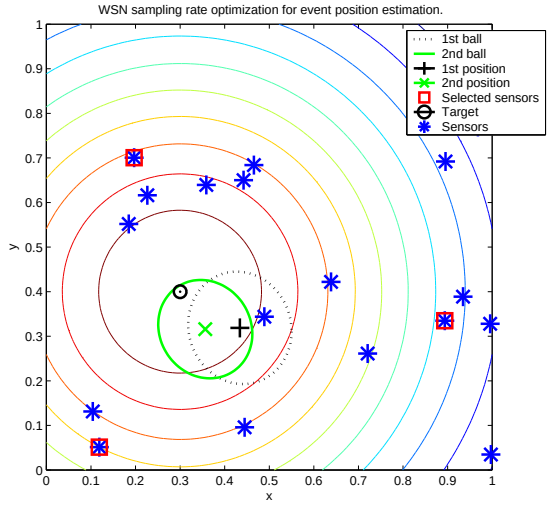


Fig. 2. An example of applying IOSS to randomly placed sensors.

## III. Experiment Results

### A. Simulations

Simulation results are shown in Figs. 1,2. In those simulations, 15 light sensors are spread 20.32 cm apart from each other. The sampling period $t_S$ is 1 sec, and the total sample number is 100. The signal noise ratio (SNR) is 8 dbm. The 1st position in the figure is $\mathbf{q}_A$. The 1st ball is the confidence ellipse associated with $\mathbf{q}_A$. The real target is believed within the ellipse. The 2nd position is $\mathbf{q}_B$, which is computed after the sensor selection. It is clear that the 2nd estimation is more precise in this case. The location error of the 1st estimation is 5.5883 cm, while the error of the 2nd is 3.7749 cm. Note that the IOSS algorithm does not requires the sensors to be placed uniformly. 20 sensors are randomly placed for the example in Fig. 2.

### B. Hardware Experiments

Since we introduced several approximations in the our algorithm, it is important to verify the validity of those approximations by our physical testbed. A picture of the testbed in shown in Fig. 3. 15 Tmote sky sensor nodes are placed under a lamp. The sink Tmote sky node (not in the picture) is connected to a PC. A GUI is running on the PC.

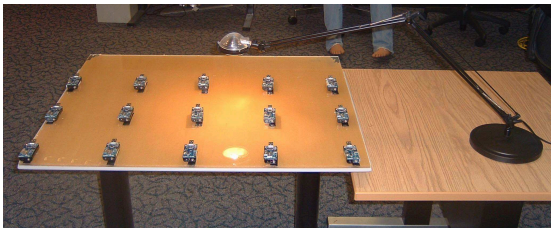

Fig. 3. A picture of our sensor selection testbed.

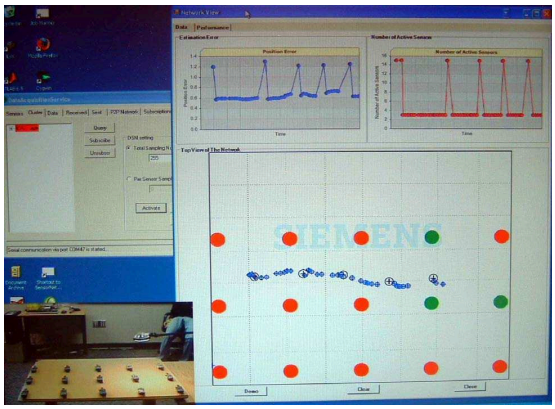

Fig. 4. A screen shoot of our sensor selection testbed.

Figure 4 is a frame from our movie [2] that demonstrates the testbed. The picture at the left bottom is from a video taken by a camcorder. The selected sensors are the green dots on the screen, the not-selected sensors are the red dots. No matter the lamp is smoothly moving or suddenly shifting, the system can always track the lamp.

Comparing to the simulation, the hardware implementation is more challenging due to the following reasons:

- Imperfect communications introduce packet drops.
- Disturbances from the ambient light.
- The light bulb is not a perfect point light source. In fact, the size of the light bulb is about 3 cm, which is comparable to the tracking error of our testbed.

Despite of the difficulties, we still achieve our goals: the IOSS algorithm selects 3 sensors and, statistically, the 2nd estimate is more precise than the 1st estimate. We measure the positioning errors on 21 points on the testbed. After placing the lamp on one position, we manually measure the position and take it as the real position. The 1st and 2nd positions are logged in a data file. Figures 5,6 are plotted based on those experiment data. The mean tracking error for the 1st estimate is 3.1387 cm, and the mean error for the 2nd estimate is 3.0269 cm. On Fig. 6 the ratio of the 2nd estimation error over the 1st estimation error is plot as a surface. Thus, the IOSS improves estimation precision at those places where that ratio is lower than 1. In the figure, we see that on most positions the error of the 2nd estimate is smaller than that of the 1st estimate. Remind that we introduce several approximates when we formulate the problem, also there are hardware imperfectness. They are the reasons why the 2nd estimate is not always better than the 1st estimate.

[2]This video and other experiment results are available at `http://cc.usu.edu/~zhensong/SensorSelection`.

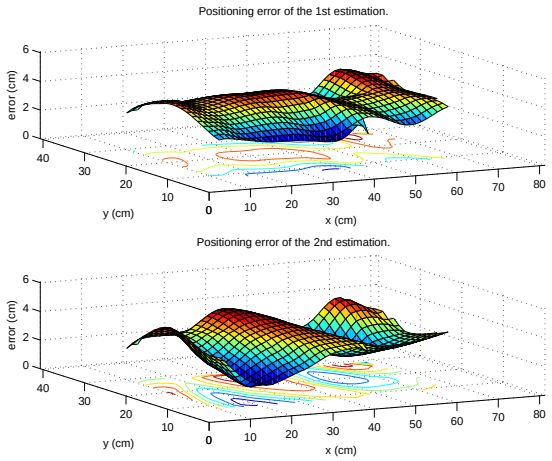


Fig. 5. Estimation errors of our sensor selection testbed.

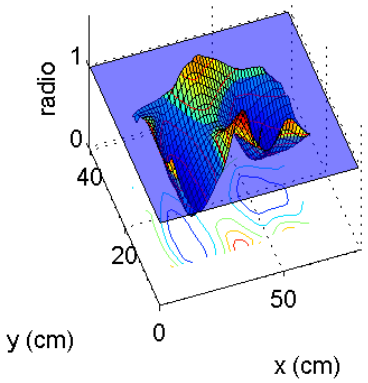


Fig. 6. A "before-and-after" comparison of our sensor selection testbed: ratio of the 2nd estimation errors (after sensor selection) over the 1st (before sensor selection) estimation errors.

### C. Remarks on the Results

The IOSS algorithm is simple, fast, and memory efficient.

In paper [6], the B&B integer programming method "allows us to schedule up to 50-60 sensors typically in the order of seconds." We use the Profile tool from Matlab and test the speed of our Matlab version IOSS algorithm. On a 3Ghz Pentium 4 PC with 1Gb memory, it takes the IOSS algorithm around 8 ms to optimize a network with 60 sensors. Thus, IOSS algorithm is roughly about 1000 times faster.

The IOSS algorithm requires memory to store $\nabla_{\mathbf{q}}\Psi$, $M$, and $\mathbf{a}_i$. Thus, it requires memory for $2mn + m^2$ float or double variables, where $m$ is the number of parameters and $n$ is the sensor number. Since $m$ is small ($m = 2$ or 3 for tracking problems) the required memory is not much. Considering grid-based Bayesian or entropy approaches, those methods need memory to store $k^2$ or $k^3$ cells, where $k$ is the number of cells in each dimension. In [8], it is claimed that (for the IDSQ sensor selection method) "we need to compute mutual information from three-dimensional joint density ... This may be computationally intensive given the limited ability of sensor nodes."

## IV. Conclusion and Future Work

In this paper, we study the sensor selection problem for WSNs. As the complexity of the original problem is very large, we introduce some reasonable approximations and formulate it as a continuous convex optimization problem. An IOSS algorithm is proposed to find the optimal solution for the approximated problem. Our experiments indicate that the algorithm selects the minimum number of sensors allowed by the Caratheodory's theorem. The algorithm also pushes the estimation errors closer to the theoretical limit, i.e., the Cramer-Rao lower bound. This algorithm is simple, fast, and memory efficient. After extensive simulation and hardware experiments, we conclude that the approximations are reasonable for engineering practices. It is worthwhile to significantly improve the performance by introducing those approximations.

In future, we will develop fully distributed IOSS and implement it on low-cost sensor nodes for real-time target tracking. We will also consider the effects of localization errors to our algorithm.

## V. Acknowledgement

We appreciate Prof. YangQuan Chen, and Mr. Cagri Gungor for the fruitful discussions.

## References

[1] J. Liu, P. Cheung, L. Guiba, and F. Zhao, "A dual-space approach to tracking and sensor management in wireless sensor networks," Tech. Rep. P2002-10077, Palo Alto Research Center, March 2002.

[2] D. M. Doolina and N. Sitara, "Wireless sensors for wildfire monitoring," in *Proceedings of SPIE Symposium on Smart Structures & Materials/NDE 2005*, 2005.

[3] P. K. Dutta and A. K. Arora, "Integrating micropower impulse radar and motes," Tech. Rep. OSU-CISRC-12/03-TR67, The Ohio State University Technical Report, 2004.

[4] F. Zhao, J. Liu, J. Liu, L. Guibas, and J. Reich, "Collaborative signal and information processing: An information-directed approach," *IEEE Proceedings*, vol. 91, pp. 1199–1209, Aug. 2003.

[5] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *In Proceedings of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002)*, (New York, NY, USA), June 2002.

[6] A. S. Chhetri, D. Morrell, and A. Papandreou-Suppappola, "Sensor scheduling using a 0-1 mixed integer programming framework," in *IEEE Workshop on Sensor Array and Multichannel Processing*, (Waltham, Massachusetts), 2006.

[7] V. Isler and R. Bajcsy, "The sensor selection problem for bounded uncertainty sensing models," *IEEE transaction on automation and Science and Engineering*, 2005. Accepted.

[8] J. Liu, J. Reich, and F. Zhao, "Collaborative in-network processing for target tracking," *EURASIP, Journal on Applied Signal Processing*, 2002.

[9] H. Wang, K. Yao, G. Pottie, and D. Estrin, "Entropy-based sensor selection heuristic for target localization," in *Proceedings of the third international symposium on Information processing in sensor networks (IPSN)*, pp. 36–45, ACM Press, 2004.

[10] N. Patwari, J. Ash, S. Kyperountas, R. Moses, N. Correal, and I. Hero, A.O., "Locating the nodes: cooperative localization in wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 22, pp. 54– 69, July 2005.

[11] D. Niculescu, *Forwarding and Positioning Problems In Ad Hoc Networks*. PhD thesis, Rutgers, The State University of New Jersey, 2004.

[12] K. W. Cheung, H. C. So, W. K. Ma, and Y. T. Chan, "Least squares algorithms for time-of-arrival-based mobile location," *IEEE Transactions on Signal Processing*, vol. 52, no. 4, pp. 1121–1130, 2004.

[13] K. W. Cheung, H. C. So, W. K. Ma, and Y. T. Chan, "Received signal strength based mobile positioning via constrained weighted least squares," in *IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03)*, vol. 5, pp. V–137–40, 2003.

[14] Y. T. Chan and K. C. Ho, "A simple and efficient estimator for hyperbolic location," *IEEE Transactions on Signal Processing*, vol. 42, no. 8, pp. 1905–1915, 1994.

[15] D. Uciński, *Optimal measurement methods for distributed parameter system identification*. Boca Raton, Florida: CRC press, 2005.

[16] V. V. Fedorov, "Design of spatial experiments: Model fitting and prediction," tech. rep., Ork Ridge National Lab, 1996.

[17] A. Pazman, *Foundations of optimum experimental design*. D.Reidel Publishing Company, 1986.

[18] S. D. Silvey, *Optimal design. An introduction to the theory for parameter estimation*. Chapman and Hall, 1980.