Directional Sensor Control for Maximizing Information Gain

Shankarachary Ragi[a], Hans D. Mittelmann[b], and Edwin K. P. Chong[a]

[a]Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523-1373, USA;

[b]School of Mathematical and Statistical Sciences, Arizona State University, Tempe, AZ 85287-1804, USA

ABSTRACT

We develop tractable solutions to the problem of controlling the directions of 2-D directional sensors for maximizing information gain corresponding to multiple targets in 2-D. The target locations are known with some uncertainty given by a joint prior distribution (Gaussian). A sensor generates a (noisy) measurement of a target only if the target lies within the field-of-view of the sensor, and the measurements from all the sensors are fused to form global estimates of target locations. This problem is hard to solve exactly—the computation time increases exponentially with the number of sensors. We develop heuristic methods to solve the problem approximately and provide lower and upper bounds on the optimal information gain. We improve the solutions from these heuristic approaches by formulating the problem as a dynamic programming problem and solving it using a rollout approach.

Keywords: Directional sensor control, maximizing information gain, rollout on heuristic methods

1. INTRODUCTION

Directional sensors are a class of sensors that have a limited/restricted field-of-view, e.g., surveillance cameras, infrared sensors, and ultrasound sensors. These sensors are becoming increasingly important due to a wide range of applications like surveillance, detection (e.g., human feature detection), and tracking. In this study, we develop tractable solutions to the problem of controlling the directions of multiple 2-D directional sensors for maximizing information gain corresponding to multiple targets in 2-D. Directional sensor control has been studied before in various contexts;[1–4] a general survey of this topic can be found in Ref. 5, where the focus is on the coverage issues in directional sensor networks.

We assume that the locations of targets are known with some uncertainty given by a joint prior distribution (Gaussian), and the sensor locations are known exactly. A directional sensor has a limited field-of-view (FOV), where the area sensed by the sensor is given by a sector in a circular region around the sensor as depicted in Figure 1. The direction of the FOV of a sensor can be changed by changing the direction of the sensor. The direction of a sensor can take several discrete values in the interval $[0, 2\pi)$. A directional sensor generates a measurement of a target if and only if the target lies within the FOV of the sensor. We assume that there is a notional fusion center where the measurements from all the sensors are fused, and the estimates of target locations are evaluated. Our goal is to assign each sensor to a particular direction such that the overall information gain is maximized. This problem is hard to solve exactly because of its combinatorial nature—the computation time increases exponentially with the number of sensors. In this study, we develop heuristic approaches that are tractable, and also provide bounds on the optimal information gain. We apply rollout on these heuristic approaches (as in Ref. 6) via the dynamic programming formulation[7] to improve the solutions from the heuristics.

Further author information: (Send correspondence to S.R.)
S.R.: E-mail: shankar.ragi@colostate.edu
H.D.M.: E-mail: mittelmann@asu.edu
E.K.P.C.: E-mail: edwin.chong@colostate.edu

Signal and Data Processing of Small Targets 2013, edited by Oliver E. Drummond, Richard D. Teichgraeber, Proc. of SPIE Vol. 8857, 88570J · © 2013 SPIE
CCC code: 0277-786X/13/$18 · doi: 10.1117/12.2022451

2. PROBLEM SPECIFICATION

Targets. There are N targets in 2-D, where $(\chi_1, \chi_2, \ldots, \chi_N)$ represent the locations of the targets. The target locations are not known exactly, however the locations are known with some uncertainty given by a (prior) joint Gaussian distribution.

Directional Sensors. There are M directional sensors in 2-D, where $(s_1, s_2, \ldots, s_M)$ represent the locations of the sensors. The sensor locations are known exactly. The direction of each sensor can take one of the K discrete values in the interval $[0, 2\pi)$. Let $\Theta = \{1, \ldots, K\}$ be the set of directions each sensor can take. Let $u = (u_1, \ldots, u_M)$ be the control vector, where $u_i \in \Theta$, $i = 1, \ldots, M$, is the direction of the ith sensor. The field-of-view of a sensor, pointed at a particular direction $\theta \in \Theta$, is shown in Figure 1, where r and α are the radial and angular sensing ranges of the sensor respectively.

Measurement Errors. Each sensor generates a 2-D position measurement of a target only if the target lies within the FOV of the sensor. These measurements are corrupted by random errors that depend on the relative location of the target with respect to the sensor and the direction of the sensor. The measurement of the jth target at the ith sensor is given by

$$z_{ij} = \begin{cases} \mathbf{H}\chi_j + n_{ij} & \text{if target lies within} \\ & \text{FOV of sensor,} \\ \text{no measurement} & \text{otherwise,} \end{cases} \tag{1}$$

where $n_{ij} \sim \mathcal{N}(0, \mathbf{Z}(s_i, u_i, \chi_j))$, $\mathbf{H}$ is an observation model, and $\mathbf{Z}(\cdot)$ is the measurement error-covariance matrix, which depends on the direction of the sensor and the locations of the target and the sensor.

Fusion. The observations obtained from the sensors are fused to form a global estimate for each target. Let $\mathcal{N}(\xi_j^{\text{prior}}, \mathbf{P}_j^{\text{prior}})$, $j = 1, \ldots, N$, be the prior distributions (Gaussian) of the target-locations. Given the observations and the prior distributions, we evaluate the posterior distribution of the target-locations by fusing the observations. The target observations are not Gaussian; the evaluation of the true Bayesian posterior distribution is not tractable. Therefore, we approximate the posterior distribution of jth target as $\mathcal{N}(\xi_j, \mathbf{P}_j)$, $j = 1, \ldots, N$, where ξ_j and $\mathbf{P}_j$ are evaluated according to Algorithm 1, where z_{ij} is the observation generated at sensor i.

Algorithm 1 Approximate Posterior Distribution

$A = \left[\mathbf{P}_j^{\text{prior}}\right]^{-1}$
$y = A\xi_j^{\text{prior}}$
for $i = 0$ to M **do** ▷ Information filtering equations
  if Sensor i generates observation **then**
    $A \leftarrow A + \mathbf{H}^{\mathrm{T}} \left[\mathbf{Z}\left(s_i, u_i, \xi_j^{\text{prior}}\right)\right]^{-1} \mathbf{H}$
    $y \leftarrow y + \mathbf{H}^{\mathrm{T}} \left[\mathbf{Z}\left(s_i, u_i, \xi_j^{\text{prior}}\right)\right]^{-1} z_{ij}$
  end if
end for
$\mathbf{P}_j = A^{-1}$
$\xi_j = A^{-1}y$

Objective. The objective is to compute u, i.e., the directions for each sensor, such that the following reward function (based on information gain) is maximized:

$$-\,\mathrm{E}\left[\sum_{j=1}^{N} \log\det(\mathbf{P}_j)\right], \tag{2}$$

where the expectation is over the prior joint-distribution of target locations, and $\mathbf{P}_j$ is the posterior distribution of the jth target, which is evaluated using Algorithm 1 given the locations of the targets—these target locations are used only to check if the targets fall within the FOV of the sensors.

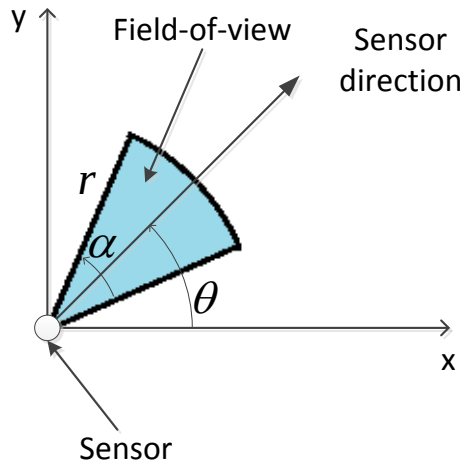

Figure 1. Field-of-view of a sensor

3. OPTIMAL SOLUTION

The optimal control-action is given by

$$u^* = \arg \max_{u \in \Theta^M} R(u), \tag{3}$$

where

$$R(u) = -\mathrm{E}\left[\sum_{j=1}^{N} \log\det(\mathbf{P}_j(u))\right]$$

and $\mathbf{P}_j(u)$, $j = 1, \ldots, N$, are evaluated according to Algorithm 1 given the control vector u. We approximate the expectation by a Monte Carlo method. Specifically, we generate several samples from the joint prior distribution of the target locations, and we compute the average (over the samples) objective value for a given control action. The above problem is a combinatorial optimization problem, where the computational time required to find the optimal solution is $O(K^M)$. Since the computational time increases exponentially with the number of sensors M, we are interested in deriving tractable heuristic methods to find an approximate solution.

4. APPROXIMATE SOLUTIONS AND BOUNDS

4.1 Continuous Optimization

We obtain an upper bound on the optimal objective value by "relaxing" the discrete property of our problem and solving its continuous version. The continuous version of our combinatorial optimization problem is stated as follows:

$$\begin{aligned} \underset{(u_1,\ldots,u_M)}{\text{maximize}} \quad & R(u_1, \ldots, u_M) \\ \text{subject to} \quad & 0 \leq u_i < 2\pi, \ i = 1, \ldots, M. \end{aligned}$$

We can use non-linear programming (NLP) solvers to obtain the optimal solution to the above problem, where the optimal objective value stands as an upper bound to the optimal objective value of our original problem. Specifically, we adopt the simulated annealing algorithm to solve the above problem.

4.2 Heuristic Approaches

Let $u = (u_1, u_2, \ldots, u_M)$ represent a solution to our problem. The optimal solution is given by

$$u^* = \arg \max_{u_i \in \Theta, i=1\ldots,M} R(u_1, \ldots, u_M), \tag{4}$$

where $\Theta = \{1, \ldots, K\}$. The computation complexity to obtain the optimal solution through (4) is $O(K^M)$, which is exponential in the number of sensors M. Therefore, we develop heuristic approaches that are polynomial in the number of sensors. The following algorithm (henceforth called $\mathcal{H}_1$) generates the solution $u_{\mathcal{H}_1} = (\bar{u}_1, \ldots, \bar{u}_M)$, where

$$\begin{aligned}
\bar{u}_1 &= \arg \max_{u \in \Theta} R(u, \emptyset, \ldots, \emptyset), \\
&\vdots \\
\bar{u}_k &= \arg \max_{u \in \Theta} R(\bar{u}_1, \ldots, \bar{u}_{k-1}, u, \emptyset, \ldots, \emptyset), \\
&\vdots \\
\bar{u}_M &= \arg \max_{u \in \Theta} R(\bar{u}_1, \ldots, \bar{u}_{M-1}, u),
\end{aligned}$$

where $\emptyset$ at any jth location in the solution $u = (u_1, \ldots, u_M)$, i.e., u_j replaced by $\emptyset$, means that the senor j is ignored while computing the reward function (as if the sensor j does not generate any observations and does not influence the reward function). Therefore, the algorithm $\mathcal{H}_1$ generates an approximate solution $u_{\mathcal{H}_1} = (\bar{u}_1, \ldots, \bar{u}_M)$, and the reward due to $\mathcal{H}_1$ is $R(\bar{u}_1, \bar{u}_2, \ldots, \bar{u}_M)$. The computational complexity of $\mathcal{H}_1$ is $O(KM)$.

We now present another heuristic approach (henceforth called $\mathcal{H}_2$) to solve the above problem. Let $\mathcal{H}_2$ generate the solution $u_{\mathcal{H}_2} = (\hat{u}_1, \ldots, \hat{u}_M)$. In contrast to the previous heuristic approach, this approach may not generate the elements of the solution vector in the order $\hat{u}_1, \ldots, \hat{u}_M$. Let $(p_1, \ldots, p_M)$ be the order in which the elements of the solution vector are generated, where $(p_1, \ldots, p_M)$ is a permutation of $\{1, \ldots, M\}$. The algorithm $\mathcal{H}_2$ is described as follows:

1. Here we abuse the notation slightly as follows. If we are evaluating the reward function with only one active sensor i (while ignoring other sensors), then the reward function is represented by $R(\emptyset, u_i, \emptyset)$. For each sensor i, $i = 1, \ldots, M$, while ignoring the rest of the sensors, we obtain the direction u_i for which the reward $R(\emptyset, u_i, \emptyset)$ is maximized. Among these sensor-direction pairs, we choose the pair with maximum corresponding reward (as evaluated above), and let $(p_1, \hat{u}_{p_1})$ be the chosen sensor-direction pair. We assign the above chosen direction $\hat{u}_{p_1}$ to the senor p_1. This assignment is fixed in rest of the algorithm.

2. For each sensor, except p_1, we repeat the above step and obtain the sensor-direction pair $(p_2, \hat{u}_{p_2})$. While incorporating the assignment of sensor p_1 to the direction $\hat{u}_{p_1}$, we compute the reward function to generate the sensor-direction pairs (as in step 1). We repeat this until the last sensor is assigned to a direction.

3. At any typical step, we generate sensor-direction pairs (only for those sensors that are not previously assigned to any direction) while incorporating the previous assignments (sensors to directions) in computing the reward for each sensor-direction pair. At the end of the step, a sensor (previously not assigned to any direction) is assigned to a particular direction. At the end of the algorithm, we are left with the solution $(\hat{u}_1, \ldots, \hat{u}_M)$.

The computational complexity of this approach is $O(KM^2)$. In both the heuristics $\mathcal{H}_1$ and $\mathcal{H}_2$, since we do not search all possible directions exhaustively, the objective values from these approaches stand as lower bounds on the optimal value.

4.3 Rollout on a Heuristic Approach

Given a heuristic approach that solves a combinatorial optimization problem (like $\mathcal{H}_1$ in the previous subsection), the authors of Ref. 6 have applied rollout approaches on the heuristic to improve the solution. We adopt the same approach, and apply rollout approaches on our heuristics to improve the solutions.

We can obtain the optimal solution (step-wise) using the dynamic programming[7] approach as follows. We start at a dummy (artificial) initial state; the state at the 1st stage is (u_1). The state at the kth stage is of the form $(u_1, \ldots, u_k)$ (also called k-solution), and the terminal state (final stage) is $(u_1, \ldots, u_M)$. The control variable at state $(u_1, \ldots, u_{k-1})$ is $u_k \in \Theta$. We get a reward at the end of the Mth step called terminal reward, which is represented by $R(u_1, \ldots, u_M)$. Let $J^*(u_1, \ldots, u_k)$ be the optimal value-to-go (see Ref. 6 for details) starting from the k-solution. The optimal solution to our control problem $(u_1^*, u_2^*, \ldots, u_M^*)$ can be obtained from the following equations:

$$u_k^* = \arg\max_{u \in \Theta} J^*(u_1^*, \ldots, u_{k-1}^*, u), \quad k = 1, \ldots, M. \tag{5}$$

In general, the optimal value-to-go $J^*(\cdot)$ is hard to obtain. Therefore, we replace $J^*(\cdot)$ with a heuristic value-to-go $\bar{J}(\cdot)$, which is typically easy to obtain.

Let $\mathcal{H}$ be any heuristic algorithm, which generates the path $(\bar{i}_1, \bar{i}_2, \ldots, \bar{i}_M)$, where $\bar{i}_k = (\bar{u}_1, \ldots, \bar{u}_k)$. Let $\bar{J}(\bar{i}_k)$ represent the value-to-go starting from the k-solution $\bar{i}_k = (\bar{u}_1, \ldots, \bar{u}_k)$, from the algorithm $\mathcal{H}$ (i.e., we use $\mathcal{H}$ to evaluate the value-to-go). The value-to-go due to the algorithm $\mathcal{H}$ is equal to the terminal reward, i.e., $\bar{J}(\bar{i}_k) = R(\bar{u}_1, \bar{u}_2, \ldots, \bar{u}_M)$. Therefore, the following is true: $\bar{J}(\bar{i}_1) = \bar{J}(\bar{i}_2) = \ldots = \bar{J}(\bar{i}_M)$. We use this heuristic value-to-go in (5) to find an approximate solution to our problem. We call this approximation algorithm "Rollout on $\mathcal{H}$" ($\mathcal{RH}$ in short) due to its structure, which is similar to *rollout algorithms*. The $\mathcal{RH}$ algorithm starts with the original dummy state, and generates the path $(i_1, i_2, \ldots, i_M)$ according to the following equation:

$$i_k = \arg\max_{j \in N(i_{k-1})} \bar{J}(j), \quad k = 1, \ldots, M$$

where, $i_{k-1} = (u_1, \ldots, u_{k-1})$, and

$$N(i_{k-1}) = \{(u_1, \ldots, u_{k-1}, u) | u \in \Theta\}, \quad k = 1, \ldots, M.$$

PROPOSITION 4.1. *The algorithm $\mathcal{RH}$ is sequentially improving with respect to $\mathcal{H}$, i.e., $\bar{J}(i_1) \leq \bar{J}(i_2) \leq \ldots \leq \bar{J}(i_M)$, where $(i_1, i_2, \ldots, i_M)$ is the path generated by the $\mathcal{RH}$ algorithm.*

Proof. Let $(i_1, i_2^{i_1}, \ldots, i_M^{i_1})$ be the complete solution from $\mathcal{H}$ given i_1 (obtained from the first step of $\mathcal{RH}$). We can easily verify that $\bar{J}(i_1) = \bar{J}(i_2^{i_1})$. Let i_2 is the solution obtained from the second step of $\mathcal{RH}$, i.e.,

$$\bar{J}(i_2) = \max_{j \in N(i_1)} \bar{J}(j).$$

But

$$\max_{j \in N(i_1)} \bar{J}(j) \geq \bar{J}(i_2^{i_1}) = \bar{J}(i_1).$$

Therefore, $\bar{J}(i_2) \geq \bar{J}(i_1)$. We can extend this argument for the rest of the steps in $\mathcal{RH}$, which proves the result $\bar{J}(i_1) \leq \bar{J}(i_2) \leq \ldots \leq \bar{J}(i_M)$. □

THEOREM 4.2. *The algorithm $\mathcal{RH}$ outperforms $\mathcal{H}$, i.e., $R(\bar{i}_M) \leq R(i_M)$, where $\bar{i}_M$ and i_M are the final paths generated by the algorithms $\mathcal{H}$ and $\mathcal{RH}$ respectively.*

Proof. We can easily verify that $\bar{J}(\bar{i}_1) = \bar{J}(\bar{i}_2) = \ldots = \bar{J}(\bar{i}_M) = R(\bar{i}_M)$. Since $(i_1, \ldots, i_M)$ is the path generated by $\mathcal{RH}$, and since

$$i_1 = \arg\max_{j \in \Theta} \bar{J}(j),$$

the following is true: $\bar{J}(i_1) \geq \bar{J}(j)$ for all $j \in \Theta$. Since $\bar{i}_1 \in \Theta$, therefore $\bar{J}(i_1) \geq \bar{J}(\bar{i}_1)$. Therefore, from the above result and Proposition 4.1, we can deduce the following:

$$R(i_M) = \bar{J}(i_M) \geq \ldots \geq \bar{J}(i_1) \geq \bar{J}(\bar{i}_1) = R(\bar{i}_M).$$

The above rollout approach can be viewed as a one-step lookahead approach (or simply one-step rollout), as we optimize, at every stage, the control for the current step by maximizing the value-to-go given the control for the current step. At the expense of increased computational intensity, we can further improve the solution from the above rollout by the following approach: optimize the controls for the current and the next steps combined (i.e., for two steps) by maximizing the value-to-go given the controls for the current and the next steps. This can be viewed as a two-step rollout. Therefore, we can generalize this to an m-step rollout; however as m increases, the computational requirement also increases. Therefore, when $m = M$, the rollout approach finds the exact optimal solution by exhaustively searching through all possible controls/directions, with computational complexity $O(K^M)$ as in the case of (4). It is also clear that the rollout on a heuristic provides a tighter lower bound on the optimal objective value compared to the bound obtained from the original heuristic approach.

5. SIMULATION RESULTS

We implement the heuristic approaches presented in the previous section in MATLAB for a scenario with four sensors and nine targets, where each sensor can take 10 possible directions $\{0, 2\pi/10, 2(2\pi/10), \ldots, 9(2\pi/10)\}$. The FOV of each sensor is $\pi/5$. We approximate the expectation by the following Monte Carlo method. We generate 50 samples from the (joint) target location distribution. For a given control vector u, we compute the reward function $R(u)$ from each sample; the objective value for the given control vector is given by the average of these 50 rewards. Figures 2, 3, 4, and 5 depict the locations of targets, sensors, and the solutions from the approaches $\mathcal{H}_1$, $\mathcal{RH}_1$, $\mathcal{H}_2$, and $\mathcal{RH}_2$ respectively. In these figures, the sensors are represented by small circles, the target (prior) distributions are represented by the error concentration ellipses, and the FOVs are represented by 2-D cones. The solution from each of these approaches are the directions computed for the sensors (can be interpreted from the FOVs of sensors shown in these figures). Table 1 compares the objective values of the solutions obtained from each of these approaches along with the objective value from continuous (relaxed) optimization. This table demonstrates that the rollout approaches on heuristics outperform the original heuristic approaches, as predicted in Theorem 4.2. This table also demonstrates that the objective values from the heuristics are relatively close to that of the relaxed problem, which implies that the solutions from our heuristic approaches and the rollout approaches are close to optimal.

Table 1. Objective values from various approaches

Approach	**Objective value**
Relaxed	39.19
$\mathcal{H}_1$	34.38
$\mathcal{RH}_1$	34.75
$\mathcal{H}_2$	34.56
$\mathcal{RH}_2$	34.75

6. CONCLUSIONS AND FUTURE WORK

We developed tractable solutions to the problem of controlling the directional sensors for maximizing the information gain corresponding to the locations of the targets. We identified that this problem is a combinatorial optimization problem, and developed heuristic approaches to solve the problem approximately. We further improved the performance of our heuristic approaches by applying rollout on the heuristics. We proved that the rollout on heuristic approaches outperformed the original heuristic approaches, and our empirical results are in agreement with this. In our future work, we will extend these heuristic approaches to the problem of dynamically controlling directional sensors for tracking moving targets.

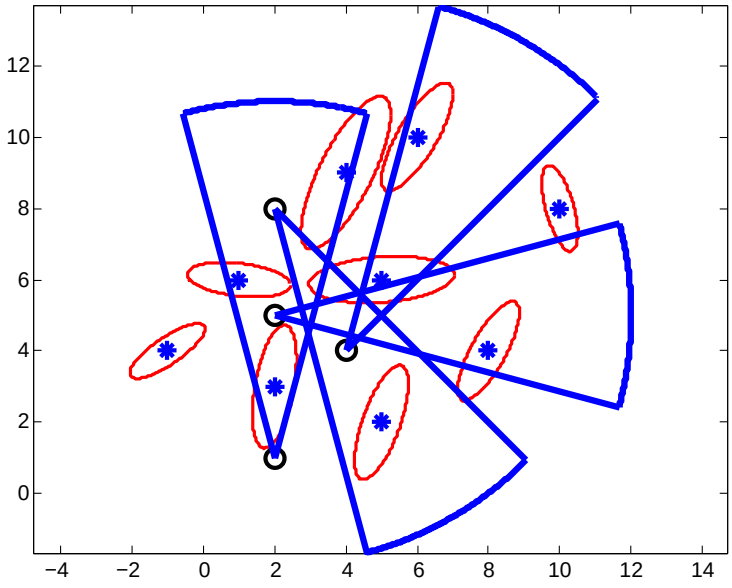

Figure 2. Solution from $\mathcal{H}_1$

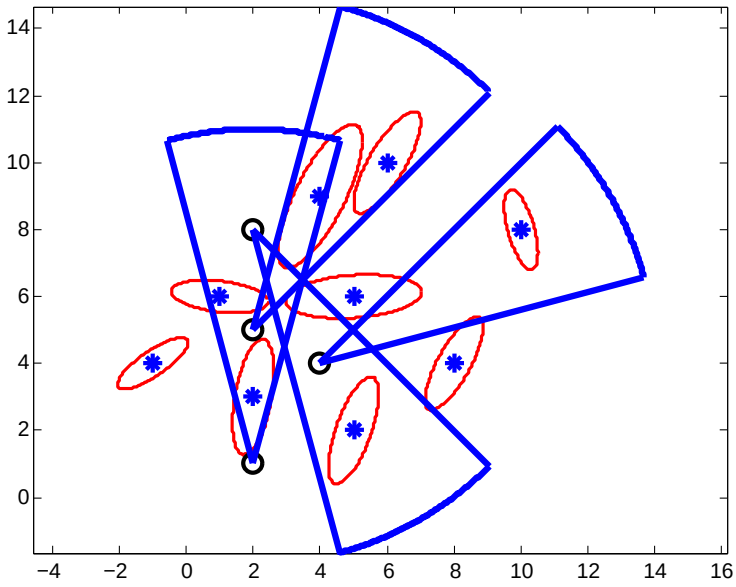

Figure 3. Solution from $\mathcal{RH}_1$

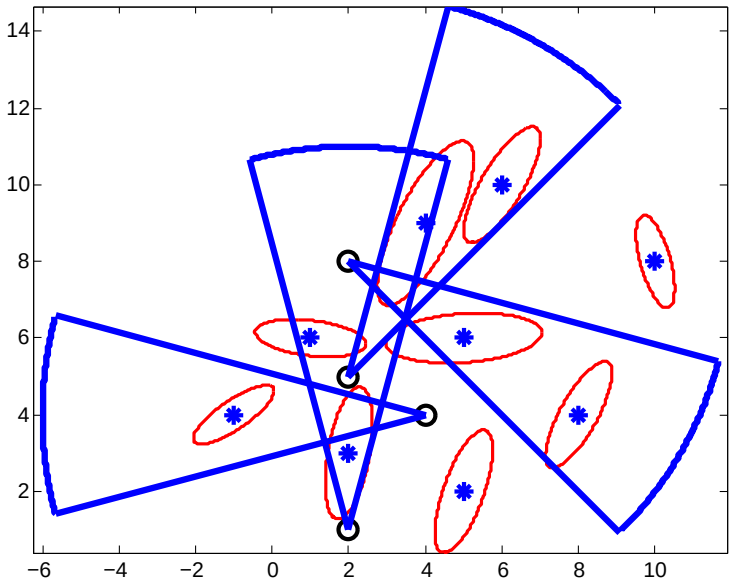

Figure 4. Solution from $\mathcal{H}_2$

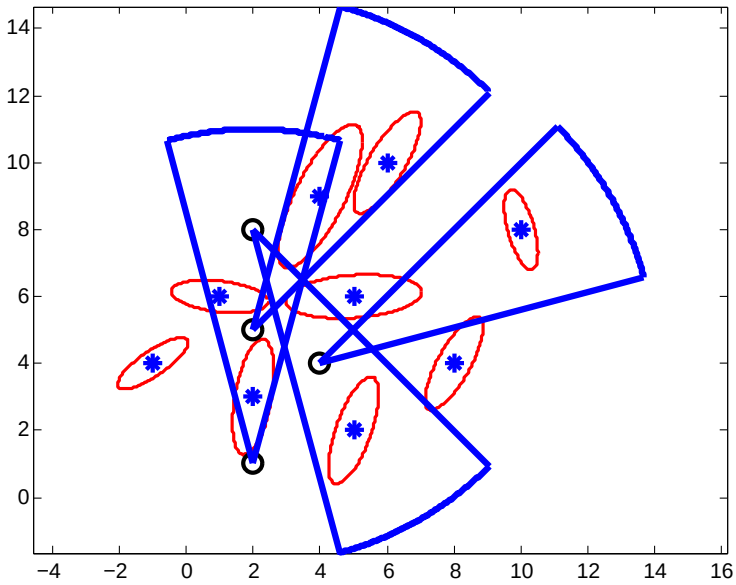

Figure 5. Solution from $\mathcal{RH}_2$

ACKNOWLEDGMENTS

The work of the second author was supported in part by AFOSR under grant FA9550-12-1-0153.

REFERENCES

[1] Wang, Y. and Cao, G., "Minimizing service delay in directional sensor networks," *Proc. 2011 IEEE INFOCOM*, 1790–1798 (Apr. 2011).

[2] Ma, H. D. and Liu, Y. H., "Some problems of directional sensor networks," *Int. J. Sensor Netorks* **2**, 44–52 (2007).

[3] Ai, J. and Abouzeid, A. A., "Coverage by directional sensors in randomly deployed wireless sensor networks," *J. Comb. Optim.* **11**, 21–41 (2006).

[4] Fusco, G. and Gupta, H., "Placement and orientation of rotating directional sensors," *Proc. 7th Annu. IEEE Communications Society Conf. SECON* (June 2010).

[5] Guvensan, M. A. and Yavuz, A. G., "On coverage issues in directional sensor networks: A survey," *Ad Hoc Networks* **9**, 1238–1255 (2011).

[6] Bertsekas, D. P., Tsitsiklis, J. N., and Wu, C., "Rollout algorithms for combinatorial optimization," *Journal of Heuristics* **3**, 245–262 (1997).

[7] Bellman, R., [*Dynamic Programming*], Princeton University Press, Princeton, NJ (1957).