Target Tracking with Distributed Sensors: The Focus of Attention Problem ⋆

Volkan Isler [a,*] Sanjeev Khanna [b] John Spletzer [c]

Camillo J. Taylor [b]

[a] *Center for Information Technology Research in the Interest of Society (CITRIS), University of California, Berkeley, CA, 94720-1764, USA*

[b] *Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, 19104, USA*

[c] *Department of Computer Science and Engineering, Lehigh University, Bethlehem, PA, 18015, USA*

Abstract

In this paper, we consider the problem of assigning sensors to track targets so as to minimize the expected error in the resulting estimation for target locations. Specifically, we are interested in how disjoint pairs of bearing or range sensors can be best assigned to targets in order to minimize the expected error in the estimates. We refer to this as the **focus of attention (FOA)** problem.

In its general form, FOA is NP-hard and not well approximable. However, for specific geometries we obtain significant approximation results: a 2-approximation algorithm for stereo cameras on a line, a $(1+\epsilon)$-approximation algorithm for any constant ϵ when the cameras are equidistant, and a 1.42-approximation algorithm for equally spaced range sensors on a circle. In addition to constrained geometries, we further investigate the problem for general sensor placement. By reposing as a

maximization problem - where the goal is to maximize the number of tracks with bounded error - we are able to leverage results from maximum set-packing to render the problem approximable.

We demonstrate the utility of these algorithms in simulation for a target tracking task, and for localizing a team of mobile agents in a sensor network. These results provide insights into sensor/target assignment strategies, as well as sensor placement in a distributed network.

Key words: Focus of Attention, sensor assignment, approximation algorithms

1 Introduction

Sensor networks are the enablers of a technology which can best be described as *omnipresence*. Small, inexpensive, low power sensors distributed throughout an environment can provide ubiquitous situational awareness. The technology lends itself well to surveillance and monitoring tasks - including target tracking - and it is in this application where our interests lie. Unfortunately, the sensors used for these tasks are inherently limited, and individually incapable of estimating the target state. Without additional constraints, a minimum of two bearing sensors (such as cameras) are required to estimate the position of a target. For range sensors, three are required to localize a target (although

⋆ A preliminary version of this paper appeared in the Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2003.

* Corresponding Author.
Email addresses: isler@eecs.berkeley.edu (Volkan Isler), sanjeev@cis.upenn.edu (Sanjeev Khanna), spletzer@cse.lehigh.edu (John Spletzer), cjtaylor@cis.upenn.edu (Camillo J. Taylor).

this can be reduced to two using filtering techniques). Noting that the measurements provided by these sensors are also corrupted by noise, we realize that the choice of which measurements to combine can greatly influence the accuracy of our tracking estimates.

Consider a distributed set of such sensors charged with tracking groups of targets as illustrated in Figure 1. In this paper, we restrict our investigation to the case where each sensor is capable of tracking a single target at any given point in time. Such an assumption is appropriate for active sensors such as pan-tilt-zoom (PTZ) cameras, which are finding significantly increased usage in the current geopolitical climate [1], or when the computational requirements of tracking algorithms support only a single target assignment. With this in mind, our problem can be viewed as an optimal allocation of resources for target tracking. How should disjoint pairs of sensors be assigned to targets so that the sum of errors in target position estimates is minimized? We refer to this as the *focus of attention* problem for distributed sensors.

2 Related Work

Since the measurements of multiple sensors are combined to estimate target pose, our work relates strongly to research in sensor fusion. Fusing measurements from multiple sensors for improving tracking performance has been the subject of significant research [2]. However, the focus has been on combining measurements from sensors (radars, laser range-finders, etc.) individually capable of estimating the target state (position, velocity, etc.). As our sensors require the fusion of pairs of measurements, we desire instead an optimal assignment of *disjoint sensor pairs* to targets. This added dimension changes the

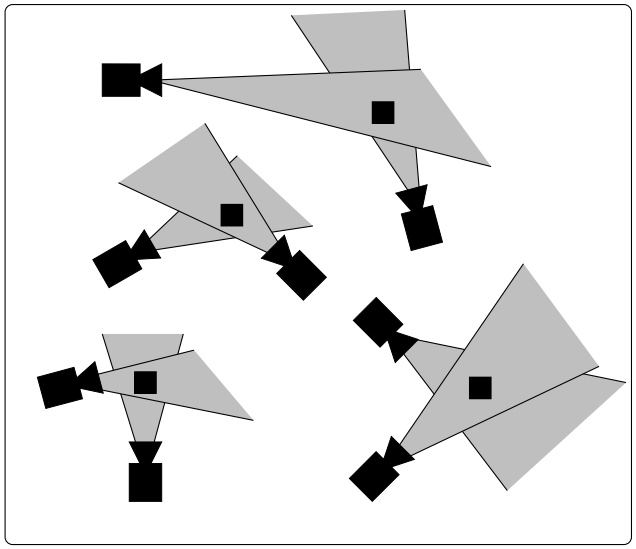


Fig. 1. An instance of the Focus of Attention problem where eight cameras are tracking four targets. Two cameras are needed to estimate the position of a target and a camera cannot track more than one target.

complexity of the problem entirely, and distinguishes our work from previous approaches.

Within the robotics community, Durrant-Whyte *et al.* pioneered work in sensor fusion and robot localization. This yielded significant improvements to methods used in mobile robot navigation, localization and mapping [3,4]. Thrun *et al.* have also contributed significant research to these areas [5,6]. However, our work distinguishes itself from traditional data fusion techniques in that the sensors themselves are actively managed to improve the quality of the measurements obtained *prior* to the data fusion phase, resulting in corresponding improvements in state estimation.

There has been other related research under the heading of sensor networks. Cortes *et al.* investigated the issue of sensor coverage [7]. In their model a single sensor is sufficient to cover a point, however the quality of coverage decreases with distance. This research focused on the movement of sensor

networks while ensuring optimal coverage. Our work begins where the sensor coverage problem leaves off, and is applicable when multiple sensors are required for monitoring a single region. Jung and Sukhatme examined a heterogeneous network of static and mobile sensors for target tracking [8]. Using a region based approach, each robot attempted to maximize the number of tracked targets per region. In contrast to our work, data fusion issues were not considered. Lastly, Horling *et al.* [9] focused on network management optimization to ensure target observability and synchronized sensor observations in order to better estimate target position. In sharp contrast, our approach optimizes explicit sensor error metrics to obtain an optimal or near optimal sensor-target assignment.

3 The Focus of Attention Problem

The following are the standard definitions used for analysis of approximation algorithms [10] that will be used in the paper:

Definition 1 *A polynomial algorithm, $\mathcal{A}$, is said to be an α-approximation algorithm, if for every problem instance I, $\mathcal{A}$ produces a solution whose cost is within a factor α of the cost of the optimal solution.*

Definition 2 *A polynomial-time approximation scheme (PTAS) is a family of algorithms $\mathcal{A}_\epsilon : \epsilon > 0$ such that for each $\epsilon > 0$, $\mathcal{A}_\epsilon$ is a $(1+\epsilon)-$approximation algorithm which runs in polynomial time in input size for fixed ϵ.*

3.1 Problem Definition

The focus of attention problem (FOA) is formally defined as as follows: The input consists of n targets, $2n$ sensors and a cost function $c(i, j, k)$ which indicates the cost of tracking target k using sensors i and j where $i, j \in \{1, \ldots, 2n\}$ and $k \in \{1, \ldots, n\}$. In the sequel, this cost represents the expected error associated with a position estimate obtained by fusing the information from sensors i and j. We are required to output an assignment: a set of n triples such that each target is tracked by two sensors, no sensor is used to track more than one target and the sum of errors associated with triples is minimized.

FOA is closely related to the following problem [11]:

Definition 3 (3D-Assignment) *Given three sets X, Y and W and a cost function $c : X \times Y \times W \to N$, find an assignment A (i.e. a subset of $X \times Y \times W$ such that every element of $X \cup Y \cup W$ belongs to exactly one element of A) such that $\sum_{(i,j,k)\in A} c(i, j, k)$ is minimized.*

3D-Assignment (3DA) is NP-hard [12] and inapproximable [13]. It is easy to see that any instance of 3DA can be reduced to an instance of FOA just by setting $c_{FOA}(i, j, k) = c_{3DA}(i, j, k)$ whenever $c_{3DA}(i, j, k)$ is defined and infinite otherwise. Moreover, since this reduction is approximation preserving, FOA with arbitrary costs is not approximable as well.

However, usually the error is not arbitrary but a function of the camera/target geometry. In the next two sections, we consider two error metrics for specific sensor configurations: cameras on the line and range sensors on the circle.

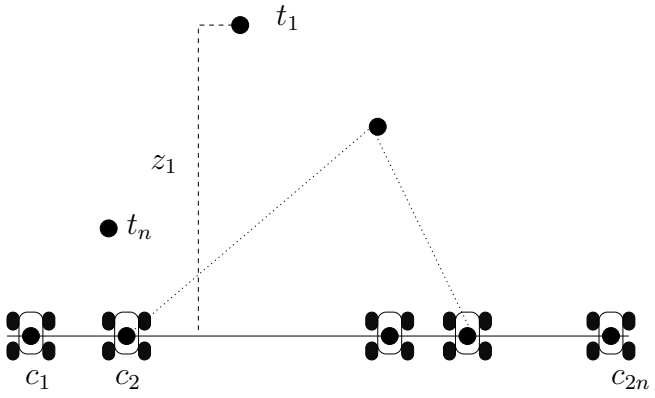


Fig. 2. The Focus of Attention Problem on the line.

3.2 Cameras on a line

In this section, we consider collinear cameras located on line l tracking targets on the plane. The error associated with cameras i and j tracking target k is $\frac{Z_k}{l_{ij}}$ where Z_k is the vertical distance of the target k to the line l and l_{ij} is the baseline, the distance between the two cameras (see Figure 2). This metric can be used to gauge the error in the stereo reconstruction [1] and gives a good approximation when the targets are not too close to the cameras. Note that this error metric fails if the targets are very close to the line l, therefore in this section we assume that there exists a minimum clearance δ such that $Z_i > \delta$, for all targets [2].

Suppose that the cameras are sorted from left to right and let c_i be the coordi-

[1] In fact, a better approximation is $\frac{Z_k^2}{l_{ij}}$. However, when all the cameras are collinear, the depth of a target is the same for all cameras and therefore for simplicity we assume that the depths are squared and the error is $\frac{Z_k}{l_{ij}}$. A brief discussion on the derivation of this error measure can be found in Appendix A.1.

[2] Recently, Goossens and Spieksma [14] obtained the following results: The Focus of Attention problem for cameras on the line remains NP-hard and this results holds even if the cameras are equidistant.

nate of the i^{th} camera. The following lemma enables us to separate matching cameras from matching targets to pairs.

Lemma 4 *Let Z_i be the depths of targets, $Z_1 \leq Z_2 \leq \ldots \leq Z_n$ and l_i be the baselines in an optimal assignment sorted such that $l_1 \leq l_2 \leq \ldots \leq l_n$. There exists an optimal matching such that the target at depth Z_i is assigned to the pair with baseline l_i.*

Proof: Suppose not. Then, in the optimal solution there exists two assignments (Z_i, l_j) and (Z_k, l_i) such that $Z_i > Z_k$ and $l_j < l_i$. Consider a new assignment obtained by modifying the optimal assignment by switching the targets of the two assignments. Since we have,

$$\frac{Z_i}{l_j} + \frac{Z_k}{l_i} > \frac{Z_i}{l_i} + \frac{Z_k}{l_j}$$
$$\frac{(Z_i - Z_k)}{l_j} > \frac{(Z_i - Z_k)}{l_i}$$

this new assignment produces less error than the optimal solution – a contradiction! □

Lemma 4 implies that once we choose pairs of cameras, the targets can be assigned to these pairs in a sorted fashion where further targets are assigned to pairs with larger baselines.

3.2.1 Performance of known heuristics

It is easy to see that a greedy assignment that assigns the furthest target the maximum available baseline can be arbitrarily far from optimal value: Consider the setting in Figure 3 with four cameras where the two cameras

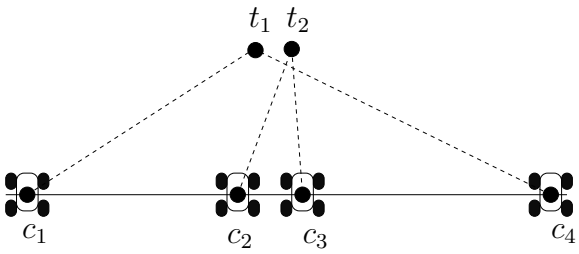


Fig. 3. A greedy assignment assigns c_1 and c_4 to target t_1 and gets stuck with the pair (c_2, c_3). The optimal assignment in this case is to assign t_1 to (c_1, c_3) and t_2 to (c_2, c_4).

in the middle are very close to each other. In this configuration, the greedy algorithm can produce an assignment that is arbitrarily more costly than the optimal assignment: $(t_1, c_1, c_3), (t_2, c_2, c_4)$.

Perhaps not so obvious is the performance of the following algorithm: Find a matching between the cameras that maximizes the sum of the baselines and assign these pairs to targets. This algorithm, which we call *Match-Assign*, gives a 3/2 approximation for the 3D-Assignment problem when the cost of a triple is the perimeter of the triangle formed by the points in the triple [12].

The Match-Assign Algorithm can also be arbitrarily bad: Suppose there is one target at $Z = \mathcal{Z}$ and $n-1$ targets at $Z = \epsilon$. Each camera c_i is located at $x = \frac{i}{2n-1}$, $i = [1, \ldots, 2n]$.

First consider the matching (c_1, c_{2n}) and (c_{4i-2}, c_{4i}), (c_{4i-1}, c_{4i+1}) for $i = 1, 2, \ldots, (2n-2)/4$. The cost of this matching is $1 + \frac{2(n-1)}{2n-1} \approx 2$ and the total error is $\mathcal{Z} + (n-1)\frac{2\epsilon}{2n-1}$.

Next, consider the matching that matches c_i with c_{i+4}. The cost of this matching is also $\frac{4n}{2n-1} \approx 2$ but the total error is $(2n-1)\frac{\mathcal{Z}}{4} + (n-1)\frac{(2n-1)\epsilon}{4}$.

Therefore, two matchings with equal sum of baselines may lead to errors such

that one can be made arbitrarily larger than the other and the Match-Assign algorithm cannot be used to obtain a good approximation.

3.2.2 A 2-Approximation Algorithm

In this section we present a 2-approximation algorithm for the previous assignment problem. The algorithm simply assigns camera i to camera $n+i$ and these pairs are then assigned to the targets according to Lemma 4. The algorithm is summarized in Table 1. Let l_i be the baselines generated by our algorithm. Let OPT be any optimal solution and l_j^* be the baselines in OPT. The following lemmas show that we can find a one-to-one correspondence between l_i and l_j^* such that l_i are longer than half of their corresponding pairs in the optimal solution.

CamerasOnALine($c_1, \ldots, c_{2n}$: camera positions, $z_1, \ldots, z_n$: target depths **)**
for i=1 **to** n
$p_i \leftarrow (c_i, c_{n+i})$
$\{p_i'\} \leftarrow$ Sort $\{p_i\}$ so that p_i' is the i^{th} largest baseline
$\{z_i'\} \leftarrow$ Sort $\{z_i\}$ so that z_i' is the i^{th} largest depth
for i=1 **to** n
assign p_i' to z_i'

Table 1
A 2-approximation algorithm for assigning cameras on a line to targets.

Lemma 5 *For all i, there exists an index j such that $l_i \geq l_j^*$.*

Proof: Let k be the the pair such that $|(c_k, c_{n+k})| = l_i$.

Let $A = \{c_k, c_{k+1}, \ldots, c_{n+k}\}$. Since $|A| = n+1$, in the optimal matching there must be two cameras in A that match with each other and the baseline of that

match is at most l_i. $\square$

Lemma 6 *Let $S = \{l_1, \ldots, l_n\}$ and $OPT = \{l_1^*, \ldots, l_n^*\}$. For any $A \subseteq S, |A| = k$, there exists a subset $B \subseteq OPT, |B| = k$ and a bijection $\sigma_k : A \rightarrow B$ such that $l_i \geq \sigma_k(l_i)/2$ for all $l_i \in A$.*

Proof: The lemma is proven by induction on $|B| = k$.

Basis: Existence of σ_1 for $k = 1$ is a corollary of Lemma 5.

Inductive Step: Let c_i and c_j be the leftmost and rightmost cameras used by the edges in A. Without loss of generality, assume that $|c_i c_{n+i}| \geq |c_j c_{n+j}|$. Let Y be the subset of pairs in OPT that matches cameras in the set $C = \{c_i, c_{i+1}, \ldots, c_j\}$.

We first observe that $|Y| \geq k$. This is because $|C| \geq n + k$ and hence at most $n - k$ cameras in C could be matched by OPT to cameras outside C.

The longest edge in B is easily seen to be at most $2|c_i c_{n+i}|$. We now recursively compute σ_{k-1} for $A' = A \setminus \{(c_i, c_{n+i})\}$. Let B' be the range of σ_{k-1}. Since $|Y| \geq k$, Y must have at least one pair, say l^*, not in B'. We match this pair to (c_i, c_{n+i}):

$$\sigma_k(l) = \begin{cases} \sigma_{k-1}(l), & \text{if } l \in A' \quad (1) \\ l^*, & \text{if } l = (c_i, c_{n+i}) \quad (2) \end{cases}$$

$\square$

Therefore by Lemma 6 there exists a mapping σ from S to the optimal matching such that $l_i \geq \frac{\sigma(l_i)}{2}$, $\forall l_i \in S$ which gives us the desired approximation

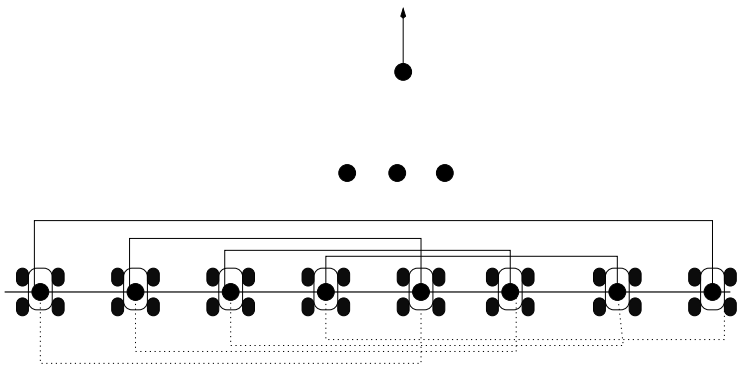


Fig. 4. The matchings produced by our algorithm (shown in dotted lines) can be twice as bad as the optimal matching (shown in solid lines) by moving the furthest target to infinity.

guarantee. This analysis is tight, there are instances where our algorithm can be twice as costly as the optimal cost:

The tight example consists of $n/4$ cameras at $x = 0$, $n/4$ cameras at $x = 1-\epsilon$, $n/4$ cameras at $x = 1+\epsilon$ and $n/4$ cameras at $x = 2$. There is one target at $Z = \mathcal{Z}$ and $n-1$ targets at $Z = \epsilon$ (see Figure 4).

The optimal cost in this case is $\frac{\mathcal{Z}}{2} + (n-2)\frac{\epsilon}{1+\epsilon} + \frac{\epsilon}{2\epsilon}$. This is achieved by matching c_1 to c_{2n} and $c_{\frac{n}{4}+1}$ to $c_{\frac{3n}{4}}$ and imitating our algorithm otherwise.

Our cost in this case is is $\frac{\mathcal{Z}}{1+\epsilon} + (n-1)\frac{\epsilon}{1+\epsilon}$ which becomes twice the optimal value as $\mathcal{Z}$ grows to infinity.

We summarize the main result of this section in the following theorem.

Theorem 7 *There exists an $O(n \log n)$-time algorithm that simultaneously gives a 2-approximation to minimizing the sum of errors metric as well as minimizing the maximum error metric when the cameras are aligned and the cost of assigning cameras i and j to target k is $\frac{Z_k}{l_{ij}}$ where l_{ij} is the distance between the cameras and Z_k is the distance of target k to the line that passes*

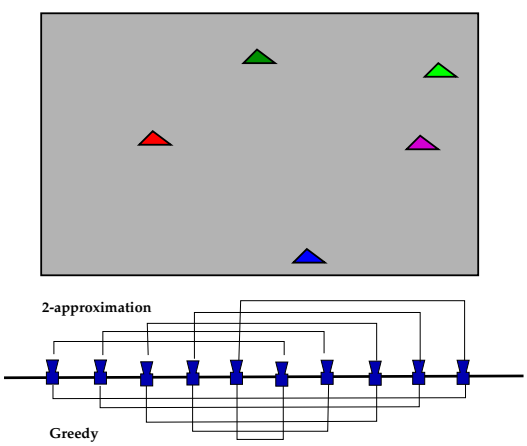
through the cameras.

3.2.3 Simulation results

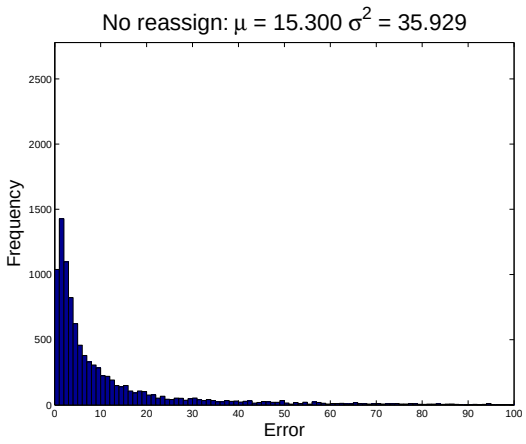
In this section, we present simulation results that contrast the performance of the 2-approximation algorithm empirically with a static and a dynamic/greedy approach. The simulation models the target tracking task as outlined in Section 3.2.2. Specifically, we consider 10 cameras charged with tracking 5 targets performing a random walk as shown in Figure 5 (a). The sensors measure bearings to targets. We assume that the sensor locations are known without error, and each sensor is constrained to tracking a single target at any given time. Measurements from pairs of sensors are then merged (via triangulation) to obtain an estimate of the position of the target. We modeled this scenario for two different algorithms.

Algorithm 1 initially assigned each target to the best available pair and kept this assignment fixed throughout the simulation. Algorithm 2 employed a greedy reassignment strategy whereas Algorithm 3 employed the 2-approximation algorithm presented in Section 3.2.2. In this approach, sensor pairs communicated target position estimates (requiring $O(n)$ communications), and sensor pair-target assignments were dynamically updated as necessary.

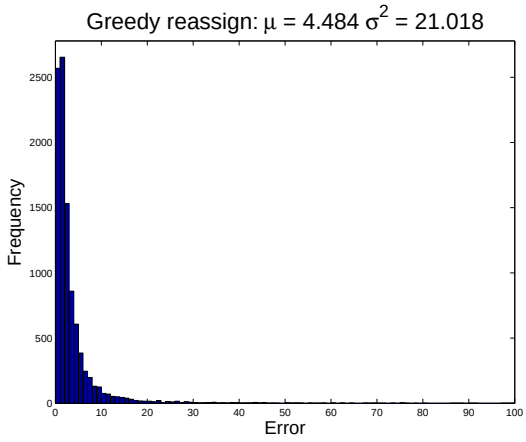
We simulated the performance of these three algorithms for 10000 iterations. The error in bearing was simulated by drawing samples from zero mean Gaussian with $\sigma = 1^\circ$. The histograms of average error for the methods is shown in Figure 5 (b-d). The mean squared error ($\mu = 2.69$), and the variance of the error ($\sigma^2 = 2.73$), produced by the 2-approximation algorithm is significantly lower than both the greedy ($\mu = 4.48, \sigma^2 = 21.02$) and the static



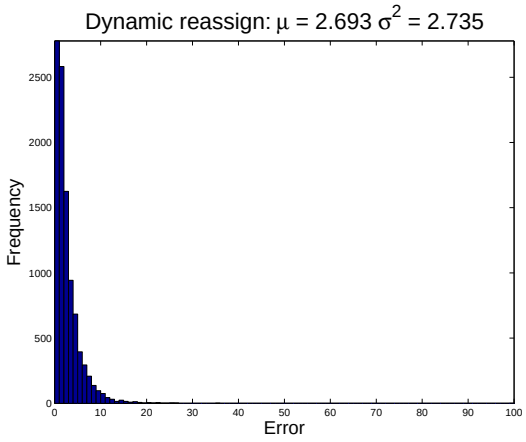
(a) Simulator



(b) No reassignment



(c) Greedy reassignment



(d) 2-approximation algorithm

Fig. 5. **Case (a):** A tracking scenario with targets performing a random walk. **Cases (b)-(d):** Comparison of three algorithms. Histograms of the mean-squared error (MSE) for tracking without reassignment (a), with greedy reassignment (b) and reassignment according to the 2-approximation algorithm (c).

$(\mu = 15.30, \sigma^2 = 35.93)$ approaches.

3.2.4 A PTAS for equidistant cameras

Our next result is a PTAS for equidistant cameras on the line. Specifically, we will show that given any $\epsilon > 0$, there is an algorithm to compute a $(1 + O(\epsilon))$-approximate solution in $n^{O(1/\epsilon^4)}$ time (polynomial for any fixed positive ϵ).

Without any loss of generality assume that the distance between two consecutive cameras is 1, hence the length of the line segment is $2n - 1$.

Note that we can view any solution as a matching between the cameras such that the weight on a matching edge (c_i, c_j) is equal to $\frac{Z_k}{|c_i c_j|}$ if and only if cameras c_i and c_j are assigned to the target k in the solution. We start with an overview of our PTAS. There are three main ingredients of our scheme:

- In any optimal matching, a camera in the left half (first n cameras) must be paired with a camera in the right half (last n cameras).
- Fix any optimal matching OPT. We distinguish between two types of camera pairings (edges) in OPT. We call an edge e *short* if its length is at most ϵn, and call it *long* otherwise. We show that all but a small fraction of the error in OPT is incident on the long edges.
- Finally, we show how to find a matching that allows us to approximate the length of each long edge in OPT to within a factor of $(1-\epsilon)$ and each short edge to within a factor of $1/2$. Since much of the error is concentrated on long edges, this suffices to get an overall $(1 + O(\epsilon))$-approximation.

In what follows, we formally develop this ideas.

Lemma 8 *In an optimal matching the leftmost n cameras match with the rightmost n cameras.*

Proof: Assume c_i is matched to c_j, $i, j \leq n$ in an optimal solution, OPT. This implies that among the rightmost n cameras at least two of them match with each other, say c_k and c_l. But then, this matching can be improved by pairing c_i with c_k and c_j with c_l which contradicts the optimality of OPT. $\square$

EquidistCamsOnALine($c_1, \ldots, c_{2n}$: camera positions,
$z_1, \ldots, z_n$: target depths, ϵ: error parameter)

$p \leftarrow \epsilon^2 n$; $q \leftarrow 1/\epsilon^2$
Partition leftmost cameras into $L_1, \ldots, L_q$ (Figure 6)
Partition rightmost cameras into $R_1, \ldots, R_q$
$\mathcal{M} \leftarrow$ set of all matchings of $\{L_i\}$ and $\{R_i\}$
for each matching M in $\mathcal{M}$
  $\{p_i\} \leftarrow$ Camera pairing produced by ComputeCameraPairs(M)
  $\{p'_i\} \leftarrow$ Sort $\{p_i\}$ so that p'_i is the i^{th} largest baseline
  $\{z'_i\} \leftarrow$ Sort $\{z_i\}$ so that z'_i is the i^{th} largest depth
  Error(M) $\leftarrow$ error produced by assigning p'_i to z'_i
return the best assignment

Subroutine ComputeCameraPairs(M: A partition matching)
$S \leftarrow \emptyset$
unmark all cameras
for $i = 1$ **to** q
  for $j = 1$ **to** q
    $m \leftarrow$ weight of matching edge type (i, j)
    for i = 1 **to** m
      $c_1 \leftarrow$ leftmost unmarked camera in L_i
      $c_2 \leftarrow$ leftmost unmarked camera in R_j
      $S \leftarrow S \cup \{(c_1, c_2)\}$
      mark c_1 and c_2
return S

Table 2
A (1+ϵ)-approximation algorithm for assigning equidistant cameras on a line to targets.

Let $p = \epsilon^2 n$ and $q = 1/\epsilon^2$. Partition the n points on the left into equal sized blocks $L_1, \ldots, L_q$ so that each block has p consecutive cameras. Similarly, we partition the points on the right into equal sized blocks $R_1, \ldots, R_q$. Consider a camera pairing (x, y) in OPT. We call it of type (i, j) if x is in L_i and y is in R_j.

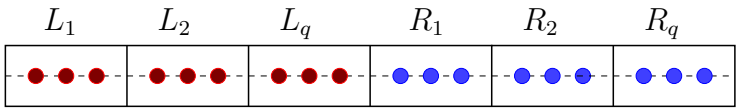


Fig. 6. Partitioning the line segment: An edge that connects a camera in block L_i to a camera in block R_j is called an edge of type (i, j).

Recall that an edge is called short if its length is no more than ϵn.

Lemma 9 *The number of short edges is at most ϵn.*

Proof: The lemma follows from the fact that the short edges may involve at most $1/\epsilon$ left blocks connected to the $1/\epsilon$ right blocks. □

There are at most q^2 (i.e. constant, for a given ϵ) different types of matching edges in OPT. We can assume from here on that we know the number of edges of each type in OPT. This is easily done by enumerating over all possible $O(n^{q^2})$ vectors of matching edge types. Note that we are not enumerating over all possible matchings of cameras (the number all such matchings is n factorial). Instead, the enumeration is over all possible types of edges. This gives us a $q \times q$ matrix T where the entry $T(i, j)$ is equal to the number of edges of type (i, j).

Given the matrix T, we use the following scheme to match the cameras. Initially, all cameras are unmarked. Starting with the block L_1, we do the following for each block L_i. Suppose $T(i, 1) = x_1, \ldots, T(i, q) = x_q$. We first pair the x_1 leftmost cameras in L_i to x_1 leftmost unmarked cameras in R_1. We mark all paired cameras. Then we pair x_2 leftmost unmarked cameras in L_i to x_2 leftmost unmarked cameras in R_2 and so on. See Figure 7.

Next, we show that the matching scheme described above produces camera

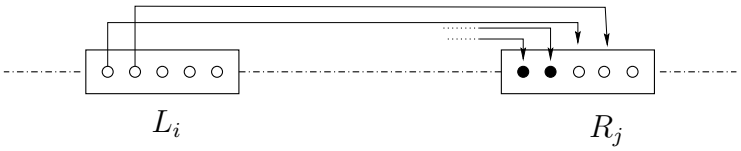


Fig. 7. Assigning two edges of type (i, j) from L_i to R_j. Note that the first two cameras in R_j are already marked – i.e. have already been assigned to a camera in L_k for some $k < i$.

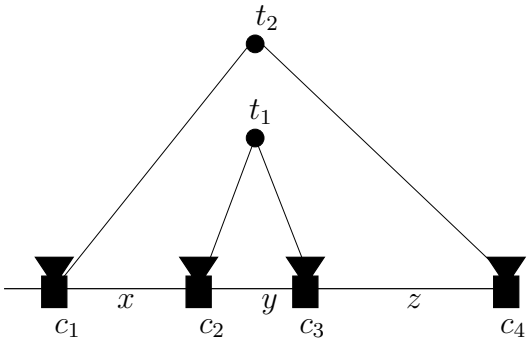


Fig. 8. Figure for Lemma 11

pairs that are not much shorter than the camera pairs in OPT.

Lemma 10 *Given the matrix T that specifies number of edges of each type in OPT, the matching scheme above decreases the length of each short edge by a factor $1/2$ at most, and each long edge by a factor of $(1 - \epsilon)$ at most.*

Proof: First observe that lengths of any two edges pairing a camera in some block L_i to another block R_j differ from each other by at most an additive $\epsilon^2 n$ term. This immediately gives us the error bound for long edges as well as for short edges whose length exceeds $\epsilon^2 n$. All that remains to consider are edges that match vertices in L_q to R_1. Observe that in our greedy matching scheme, we pair cameras in L_q only after all cameras in $L_1, ..., L_{q-1}$ have been paired. Thus we pair cameras in L_q to the rightmost possible cameras in R_1, doing at least as well as the pairing in OPT for this group. □

Lemma 11 *Let c_1, c_2, c_3 and c_4 be four cameras ordered from left to right, $x = |c_1c_2|$, $y = |c_2c_3|$, $z = |c_3c_4|$ with $z > x$. In addition, let t_1 and t_2 be two targets at distances z_1 and z_2 respectively (Figure 8). If (c_1, c_4, t_2) and (c_2, c_3, t_1) are triples in an optimal assignment then:*

$$\frac{Z_1}{y} \leq Z_2 \frac{(x+y)}{(x+y+z)(y+z)}$$

Proof: Consider the assignment obtained by crossing the pairs: (c_1, c_3, t_1) and (c_2, c_4, t_2) (see Figure 8). Due to optimality we have

$$\frac{Z_1}{y} + \frac{Z_2}{x+y+z} \leq \frac{Z_1}{x+y} + \frac{Z_2}{y+z}$$

and the lemma follows by simple algebraic manipulation. □

Finally, we show that all but a small fraction of the error weight in an optimal matching is incident on the long edges.

Lemma 12 *Let the weight (error) on an edge e for an assignment be $\frac{Z_i}{|e|}$ where Z_i is the depth of the target assigned to this edge and $|e|$ is the distance between the cameras connected by e. In any optimal assignment, the total weight on the short edges is at most an 64ϵ fraction of the overall weight.*

Proof: Let M and N be the leftmost and rightmost $\frac{3n}{4}$ cameras respectively. In an optimal matching, due to Lemma 8, the edges in M match with rightmost n edges and at least $\frac{n}{2}$ of them are in N. Let $B = \{b_1, \ldots, b_{\frac{n}{2}}\}$ be the set of any $\frac{n}{2}$ "big" edges that match cameras from M to cameras in N and $S = \{s_1, \ldots, s_k\}$ be the set of "small" edges. By Lemma 9, $k \leq \epsilon n$.

Partition B into $\frac{n}{2k} \geq \frac{1}{2\epsilon}$ groups B_i of size k arbitrarily.

We pick any group B_j and match the edges $b_i \in B_j$ to edges in S arbitrarily. Let Z_i^s and Z_i^b be the depths of targets assigned to s_i and b_i respectively. By Lemma 11 with $x + y \leq n + \epsilon n$, $x + y + z \geq \frac{n}{2}$ and $y + z \geq \frac{n}{4}$ we get:

$$\frac{Z_i^s}{s_i} \leq Z_i^b \frac{(n + \epsilon n)}{\frac{n}{2}\frac{n}{4}} = Z_i^b \frac{8(1+\epsilon)}{n} \leq \frac{16 Z_i^b}{n}$$

Let $w(S)$ be the total error in set S. Since a baseline can be of length at most $2n - 1$, by summing up over the elements in S, we get $w(S) \leq 32 w(B_j)$.

Therefore we conclude:

$$w(B) \geq \frac{w(B_i)}{2\epsilon} \geq \frac{w(S)}{64\epsilon}$$

since the total weight is greater than $w(B)$, the lemma follows. □

Theorem 13 *There exists a PTAS for assigning equidistant cameras on a line.*

Proof: The matching described ensures that short edges in OPT are reduced by at most a factor of 2 and long edges are within a factor of $(1 + \epsilon)$. Using Lemma 12 above, by combining these matchings, we get an overall $1 + O(\epsilon)$-approximation. □

3.3 Range-Sensors on a Circle

In this section, we consider range-sensors located on a circle $\mathcal{C}$ at equidistant intervals, tracking targets that are located inside $\mathcal{C}$. The error associated with

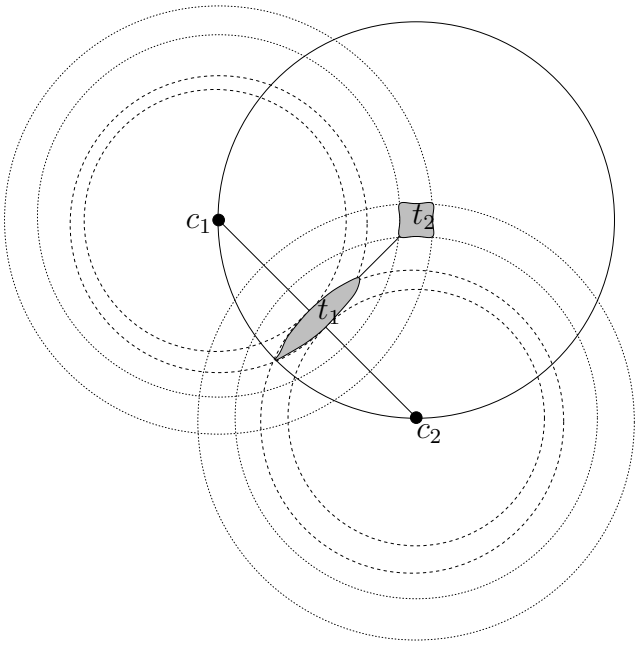


Fig. 9. Geometric Dilution of Precision (GDOP): Each sensor c_1 and c_2 measures its distance to target with some precision. The error in position estimation is shown for two target locations t_1 and t_2. The error reduces as the angle $\theta = \angle c_1 t_i c_2$ gets closer to $\frac{\pi}{2}$.

a pair of range sensors (c_1, c_2) and a target t is approximated by $\frac{1}{\sin\theta}$ where $\theta = \angle c_1 t c_2$ (Figure 9). This is the Geometric Dilution of Precision (GDOP) for sensors that measure distances from the targets. A brief discussion on the derivation of this error measure can be found in Appendix A.2.

In practice three range sensors are required for explicit target localization. However, target-tracking need not be an adversarial task. Consider a team of mobile robots negotiating a sensor network. Pairs of sensor measurements could be paired with heading information to enable localization. In this application, identifying optimal pairs would prove useful for providing optimal position estimates while minimizing network transmissions.

For simplicity, assume there are $4n$ sensors and $2n$ targets. Let S be the set

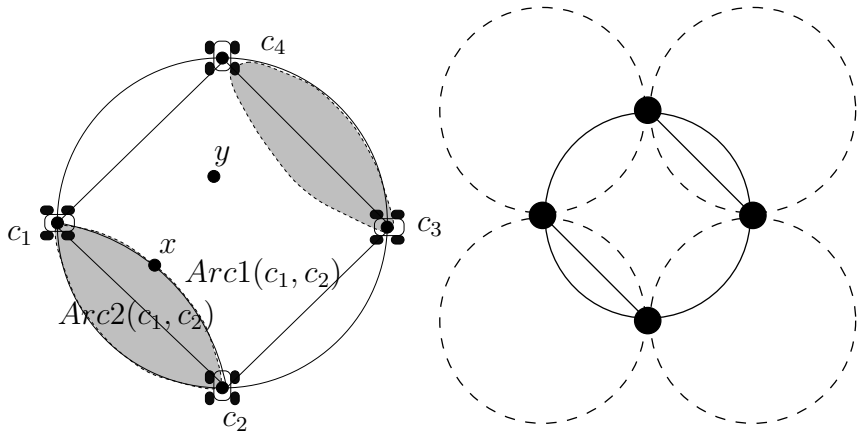


Fig. 10. Sensors on circle: **LEFT:**The defective region for sensors c_1 and c_2 is the shaded area defined by arcs $Arc1(c_1, c_2)$ and $Arc2(c_1, c_2)$. **RIGHT:** The defective regions are disjoint.

of pairs generated by matching sensor i with sensor $i + n$ which is 90 degrees away clockwise from i. Assign the targets arbitrarily to pairs.

For two sensors c_1 and c_2, let x be a point inside C such that $\angle c_1 x c_2 = \frac{3\pi}{4}$ (see Figure 10). Let $Arc1(c_1, c_2)$ be the arc defined by c_1, c_2 and x and $Arc2(c_1, c_2)$ be the arc axially symmetric with respect to the the chord c_1c_2. Note that $Arc2$ lies on $\mathcal{C}$.

Let A_{ij} be the region inside $Arc1(c_i, c_j)$ and $Arc2(c_i, c_j)$. Since any target outside the region A_{ij} is viewed by an angle less than $\frac{3\pi}{4}$ and greater than $\frac{\pi}{4}$ degrees from (c_i, c_j), we call the region A_{ij} a *defective* region for the pair (c_i, c_j). This angle is enough to guarantee a 1.42-approximation since $1/\sin(\frac{3\pi}{4}) < 1.42$ and the least error possible in this metric is 1. A target is *good*, if it is assigned to sensors c_1 and c_2 and located outside the defective region of (c_1, c_2). We summarize the properties of defective regions in the following propositions, which can be proven using basic geometric formulas.

Proposition 14 *Any target outside the defective region of sensors c_1 and c_2*

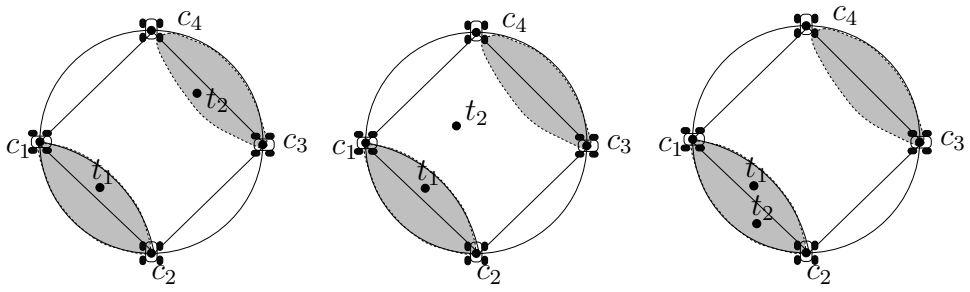


Fig. 11. Three possible cases for the location of target t_2.

is viewed by an angle less than $\frac{3\pi}{4}$ *and greater than* $\frac{\pi}{4}$ *from* c_1 *and* c_2.

Proposition 15 *Let* c_1, c_2, c_3 *and* c_4 *be four sensors* $\frac{\pi}{2}$ *degrees apart. Defective regions of* $(c_1, c_2), (c_2, c_3), (c_3, c_4)$ *and* (c_4, c_1) *are disjoint (Figure 10 right).*

Having assigned the targets to sensors $\frac{\pi}{2}$ degrees apart we proceed as follows: We scan the pairs assigned to each target t_i. Suppose the current pair is (c_1, c_2).

Now suppose that t_1 assigned to (c_1, c_2) is defective (i.e. in the defective region A_{12} of c_1 and c_2). Consider the pair (c_3, c_4), such that c_3 (resp. c_4) is the antipodal of c_1 (resp. c_3) and the target t_2 assigned to (c_3, c_4). Figure 11 illustrates the three possibilities based on the location of t_2.

- if $t_2 \in A_{34}$, we swap targets: the new assignment is (c_1, c_2, t_2) and (c_3, c_4, t_1).
- if t_2 is good and outside A_{12} again we swap targets: the new assignment is (c_1, c_2, t_2) and (c_3, c_4, t_1).
- if t_2 is good and inside A_{12}, we swap pairs: the new assignment is (c_1, c_4, t_1) and (c_2, c_3, t_2).

The reason we picked the angle as $\frac{3\pi}{4}$ is to make the defective regions disjoint: As the right illustration in Figure 10 shows, by construction the defective regions only intersect at the sensors. This makes each assignment have an error of 1.42 at most. In addition, once an assignment is modified we never

return to it. Therefore this algorithm gives a 1.42-approximation for $1/\sin\theta$ error metric.

The main result of this section is summarized in the following theorem:

Theorem 16 *There exists an $O(n)$-time algorithm that simultaneously gives a 1.42-approximation to minimizing the sum of errors metric as well as minimizing the maximum error metric when the $4n$ sensors are equally spaced on a circle and the cost of assigning sensors i and j to target k is $\frac{1}{\sin\angle ikj}$.*

3.3.1 Simulation results

In this section, we demonstrate the utility of the algorithm for range sensors on circle for a cooperative localization task.

Target tracking need not be adversarial. In this simulation, n robots are operating within a sensor network defined by $2n$ range sensors on a circle. The robots rely on pairs of sensor measurements to estimate position. The resulting pair of position estimates are then merged with odometry information to eliminate pose ambiguity. Both the sensor and odometry measurements are corrupted with random Gaussian noise.

Again, three algorithms were modeled. Each initiated with a globally optimal assignment of sensor pairs to targets. In the first, this assignment was maintained throughout the simulation. The second employed a greedy reassignment strategy at each time step. Finally, Algorithm 3 followed the 1.42-approximation presented in Section 3.3. In this case, the sensor pairs assigned to every two targets were chosen from the initial four sensors assigned to the targets. Localization then proceeded with each robot transmitting a position

estimate to its assigned sensor pair. The sensor pair in turn transmitted range measurements to the target. These measurements, the knowledge of sensor positions and odometry measurements allowed each robot to estimate its position at each timestep. The procedure then iterated.

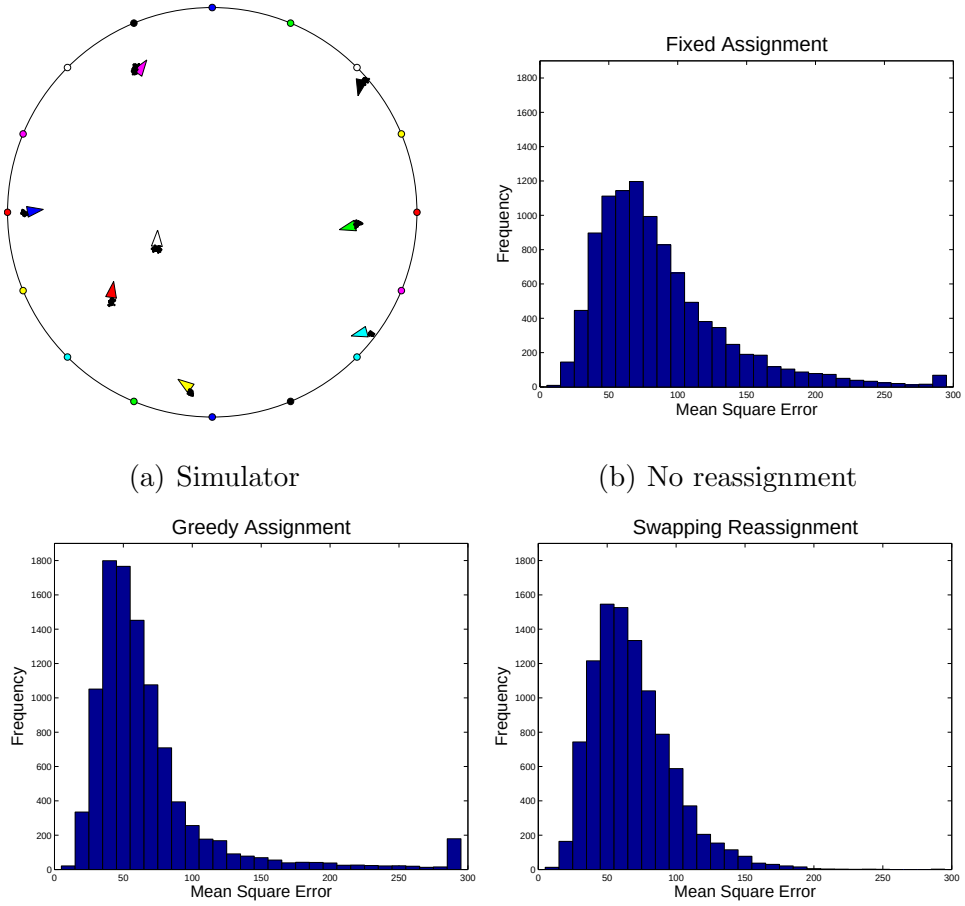
Localization performance for the three algorithms is reflected in Figure 12. In this example, 8 robots were tracked by 16 sensors over 10,000 time steps. The robots localized while following pseudo-random trajectories through the network. Results indicate significant improvements in localization performance can be achieved by intelligently assigning targets to sensors.

Both our 1.42 approximation (MSE=68.4, $\sigma^2 = 29.3$) as well as greedy reassignment (MSE=70.9, $\sigma^2 = 66.8$) outperformed static assignment (MSE=87.5, $\sigma^2 = 49.7$). It is interesting to note that while greedy reassignment and our 1.42 approximation have nearly identical performance with respect to mean square error, the performance guarantee of our approach results in a significantly lower variance. It also scales far better computationally ($O(n)$ vs. $O(n^3)$) and requires far fewer reassignments (2,206 vs. 61,241 out of 80,000 possible assignments).

3.4 Discussion: Universal Placement

Note that the analysis above shows that the equidistant placement for $\frac{1}{\sin\theta}$ metric is *universal*: When the cameras are placed equidistantly, no matter where the targets are located, our algorithm guarantees a 1.42-approximation to the optimal matchings generated by *any* placement of sensors on circle.

Similarly, a universal placement for cameras on a line segment $[x, y]$ for the



(a) Simulator (b) No reassignment

(c) Greedy reassignment (d) 1.42-approximation

Fig. 12. **Case (a):** A tracking scenario showing robot and sensor positions. **Cases (b)-(d):** Comparison of three algorithms. Histograms of the mean-squared error (MSE) for tracking without reassignment (a), with greedy reassignment (b) and reassignment according to the 1.42-approximation algorithm (c). The latter reduces both MSE and error variance over the alternative approaches.

Z/b metric would be to put half of the cameras on x and the other half on y, which guarantees an optimal assignment for this metric.

3.5 Arbitrary Sensor Assignment

The inapproximability of FOA for general sensor assignment lead us to repose it as its "dual" maximization problem. To do this, we define the notion of a *valid track*. An assignment (c_i, c_j, t_k) is considered a valid track if $Err(c_i, c_j, t_k) \leq \delta_0$, where δ_0 represents an acceptable error threshold predefined by the user. The problem then becomes: Given a set of sensors C with $c_i \in C$, a set of targets T with $t \in T$, and an error threshold δ_0, construct a set of disjoint assignments A, where $(c_i, c_j, t_k) \in A$ iff $Err(c_i, c_j, t_k) \leq \delta_0$, such that $|A|$ is maximized.

In addition to arbitrary error metrics, this formulation allows us to deal with occlusions in the scene: If target t_j is not visible from camera c_i, we delete all triples that contain both c_i and t_j from the set of valid tracks.

GreedyAssign(A = $\{(c_i, c_j, t_k)\}$: valid assignments)

$S \leftarrow \emptyset$
for all assignments $a \in A$
  if a does not conflict with any assignment in S
  $S \leftarrow S \cup \{a\}$
return S

Table 3
Greedy algorithm for selecting valid assignments.

When the error metric is arbitrary, this problem is equivalent to *Maximum 3-Set Packing* [3], which is known to be NP-hard [11]. It is also known that a simple greedy heuristic (Table 3) which adds any set to the solution as long as it does not conflict is within a factor of 3 of the optimal solution. A *2-locally-*

[3] Given a 3-set system (S, C) – a set S and a collection C of size 3 subsets of S, find a maximum cardinality collection of disjoint sets in C.

optimal solution is defined as a maximal solution that cannot be improved further by removing any item from the current solution, and attempting to insert two non-conflicting items (Table 4). It has been shown that any 2-locally optimal solution provides a $\frac{5}{3}$ approximation [15,16]. If $N = O(n^3)$ is the number of valid tracks, the greedy algorithm can be implemented in $O(N)$ time, whereas finding a 2-locally optimal solution takes $O(N^2)$ time. The details of the latter implementation can be found in [15].

2-LocalAssign(A = $\{(c_i, c_j, c_k)\}$: valid assignments)

$S \leftarrow GreedyAssign(A)$
improved $\leftarrow$ true
while improved
  improved $\leftarrow$ false
  for all assignments $a \in S$
    for all assignments $b, c \in A \setminus S$
      if $S \setminus \{a\} \cup \{b, c\}$ is a valid solution
        $S \leftarrow (S \setminus \{a\}) \cup \{b, c\}$
        improved $\leftarrow$ true
return S

Table 4
A 2-local algorithm for selecting valid assignments. A solution is called a *valid solution* if no two of assignments in the solution contain the same target or camera.

One might suspect that a 2-locally optimal solution would yield an approximation factor better than $\frac{5}{3}$ for restricted error metrics. However, this is not the case, even for equidistant cameras on the line: Consider the example in Figure 13 with cameras on a line, Z/b as our error metric and an error threshold $\delta_0 = 1$. There are five targets, $t_1, \ldots, t_5$ and $z_1 = 9, z_2 = 7$, $z_3 = 5, z_4 = 3$ and $z_5 = 1$. Ten cameras c_1 to c_{10} are located at $x = 1, \ldots, 10$. Optimum packing is five targets with $(t_1, 1, 10), (t_2, 2, 9)$, $(t_3, 3, 8), (t_4, 4, 7)$ and $(t_5, 5, 6)$, represented by the nodes of the conflict graph shown on the left in Figure 13.

Suppose our solution is $(t4, 3, 6)$, $(t5, 8, 9)$ and $(t3, 5, 10)$. Note that it is not possible to remove a triple from this solution and insert two, therefore it is 2-locally optimal. But this implies a $\frac{5}{3}$ approximation, which shows that the analysis is tight. It is possible to generalize this example to $5k$ targets, just by replicating k instances of the same example and putting them on the top of each other. Thus, the $\frac{5}{3}$ lower bound is tight even for equi-spaced cameras on a line.

In the next section, we investigate the utility of the greedy and 2-local algorithms through simulations.

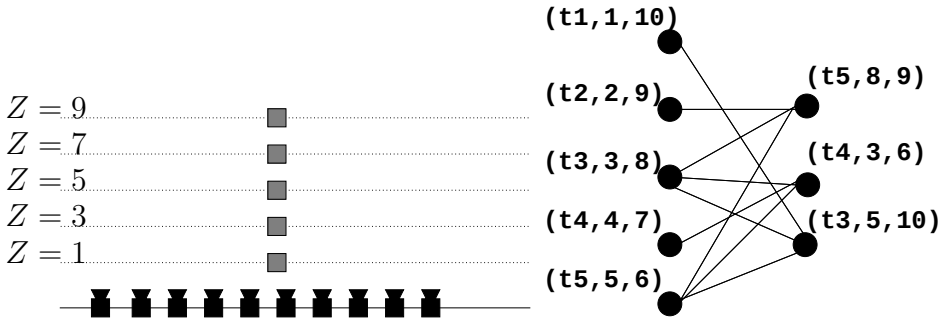


Fig. 13. **Left:** An instance of FOA. **Right:** The conflict graph for the optimal solution and a 2-local solution: The nodes on the left (resp. right) represent the optimal (resp. 2-local) solution. There is an edge between two nodes if the assignments conflict. In this case, a 2-local solution gives a $\frac{5}{3}$ approximation, showing that the analysis is indeed tight.

3.5.1 Simulation results

In this last simulation, we examined the arbitrary sensor placement problem as outlined in Section 3.5. For this example, 20 cameras were distributed roughly uniformly on the plane and charged with tracking 10 targets. Here, the objective was to maximize the number of valid tracks, in contrast to the error minimization objective of previous simulations. Targets followed random

trajectories, and were tracked in simulation using particle filters. The respective particle sets were employed to generate a numerical error metric for the targets as discussed in [17].

Two algorithms were investigated for this maximization approach. The first employed a greedy assignment strategy, and the second a 2-locally optimal approach as discussed in Section 3.5. The latter took the greedy solution as input, and as a consequence could only improve on its performance. Reassignment was made for both algorithms at each time-step. Several trials were conducted corresponding to sparse and dense solution sets. Data from a representative trial can be found in Figure 14.

In each trial, the 2-local solution improved over greedy by 5-15%. As expected, the larger improvements corresponded to dense solution sets - i.e. when there were more opportunities for finding local improvements. These results are by no means encompassing, and provide only insights into expected performance which is a function of too many variables to address here. However, they imply that unless the guarantee of improved performance is critical, the greater computational complexity of 2-local may not be warranted by the expected performance improvement over greedy for real-time applications.

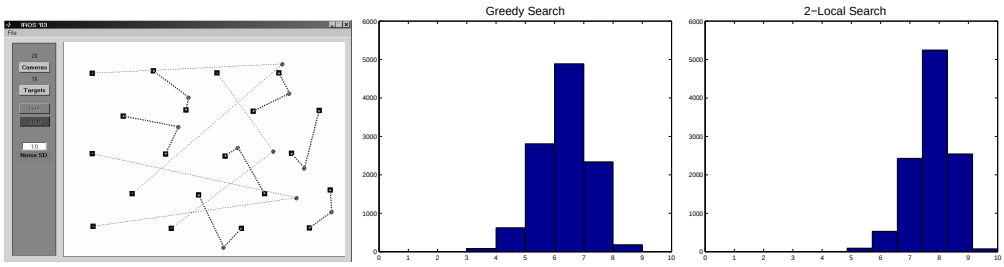


Fig. 14. **Left:** Simulator snapshot for 2-local assignment trial. **Center, Right:** The number of valid tracks recovered for greedy and 2-local search strategies. In this example, 2-local improved over greedy by on average 15%.

4 Conclusions and Discussion

In this paper, we have introduced the focus of attention problem and studied the algorithmic aspects of managing the attention of a group of distributed sensors. We observed that for a general cost metric, the problem is NP-hard and not well approximable. However, for constrained geometric cases we were able to exploit relations between the sensor geometry and corresponding error metrics. From this, we obtained: a 2-approximation for stereo cameras constrained to the same baseline, a PTAS solution for the same geometry when the cameras are spaced equidistantly, and a 1.42-approximation for $4n$-range sensors equi-spaced on the circle. For arbitrary sensor placement, we reposed the problem in a maximization vein. Using results from maximum set-packing, we obtained a $\frac{5}{3}$-approximate solution. This was implemented in simulation, and its performance contrasted against a greedy approach.

Adjusting the focus of attention of the sensors reduces the overall tracking error significantly. This is true when applied to both direct and filtered estimation problems, as demonstrated in sections 3.2.3 and 3.3.1, respectively. The 2-approximation for stereo cameras and the 1.42-approximation for range sensors have several desirable attributes. Their matchings have twofold approximation guarantees; the sums of errors are bounded, as are the individual target errors. Additionally, they are readily implemented, and are inexpensive both computationally ($O(n \log n)$ and $O(n)$, respectively) and in terms of network communications ($O(n)$). In simulation, both showed significant improvements in performance over greedy/static assignment strategies. The constraints to geometry are restrictive but still useful, and we are currently working to extend these to additional configurations.

Empirical results for arbitrary sensor placement simulations indicate on average a 5-15% improvement for the $\frac{5}{3}$-approximate solution over a greedy approach. However, the former is more expensive computationally (quadratic vs. linear in the number of valid tracks). As a consequence, a greedy strategy may be preferred for real-time applications.

While we feel these results provide a solid theoretical footing for the FOA problem, there is still significant room for future work. Using our error metrics, sensor configurations are limited to restricted geometries. This is addressed to a point by our $\frac{5}{3}$-approximate solution for arbitrary sensor placement. However, it relies upon an alternate objective function (maximizing low error tracks rather than minimizing the total error). Strategies for minimizing the tracking error under general sensor placement is the topic of ongoing work.

In the general FOA formulation, we have not addressed the subject of occlusions. We have assumed that the individual targets are neither occluded by one another, nor by clutter in the environment. Such an assumption is unrealistic. However, the estimates from each sensor/target triple need not be used as direct estimates for target positions. Rather, they can serve as inputs to traditional Bayesian filters (i.e. Kalman or particle) which yield estimates of the target pose. The latter has shown robustness for tracking features in a cluttered image [18]. For the arbitrary sensor placement problem, general visibility constraints (range, occlusions, *etc.*) can also be accommodated by merely eliminating the corresponding triples from the feasible set of assignments.

Finally, we have only considered the case where each sensor is constrained to track a single target during each time step. Such an approach is relevant for

pan-tilt-zoom cameras, or when the computational requirements of the tracking algorithm support only a single target assignment. We have not addressed the case where multiple targets are visible within a single camera's field of view. We hope to answer such questions in our future research.

A A note on error measures

In this section, we present a brief discussion on the error functions used in the paper. Our presentation is based on [19].

Suppose we have a function $y = f(x)$ that is used for estimating the position of the targets, y, given the observable x. Here, both x and y can be vectors. If the error is unbiased, a small error ϵ in the measurement x propagates to our estimation as $y' = f(x + \epsilon) \approx f(x) + J\epsilon$ where J is the Jacobian of f. Throughout the paper we use the determinant of the covariance of the transformed variable y as an analytical error function:

$$Cov_y = J \cdot Cov_x \cdot J^T \tag{A.1}$$

A.1 Stereo cameras

Given a point $X = [x\ y\ z]$ in 3D, let (u_1, v_1) and (u_2, v_2) be the coordinates (in pixels) of the projection of X onto the image plane of two cameras. If the two images are rectified, we have $v_1 = v_2$. It is commonly assumed that the observation in v is error-free. Hence, the depth of the world point, z, can be

estimated by

$$z = \frac{bf}{|u_1 - u_2|} \tag{A.2}$$

where f is the focal length of the cameras, b is the baseline and the quantity $d = |u_1 - u_2|$ is called the disparity. By defining $\delta = d/f$ and taking the derivatives with respect to δ, we obtain the standard deviation of the depth as:

$$\sigma_z = \frac{Z^2}{b}\sigma_\delta \tag{A.3}$$

Hence, the propagation of error in disparity measurements to the depth estimation is proportional to the quantity $\frac{z^2}{b}$.

A.2 Range sensors

Let $\vec{r_1} = [x_1, y_1]$ and $\vec{r_2} = [x_2, y_2]$ be the coordinates of two sensors that measure their distances, r_1 and r_2, to the target located at $\vec{r} = [x, y]$. Hence, we have the measurements:

$$r_i = \sqrt{(x - x_i)^2 + (y - y_i)^2}, \qquad i = 1, 2 \tag{A.4}$$

By taking derivatives with respect to x and y, the determinant of the jacobian can be shown to be:

$$|J|^{-1} = \frac{|\vec{r_1}||\vec{r_2}|}{\vec{r_1} \times \vec{r_2}} = \frac{1}{\sin(\theta)} \tag{A.5}$$

where θ is the angle between the two vectors $\vec{r} - \vec{r_1}$ and $\vec{r} - \vec{r_2}$.

References

[1] DHS HSARPA automated scene understanding bidders conference, Washington, DC (April 2004).

[2] Y. Bar-Shalom, X. Li, T. Kirubarajan, Estimation with applications to tracking and navigation, John Wiley, 2001.

[3] S. Majumder, S. Scheding, H. Durrant-Whyte, Multi-sensor data fusion for underwater navigation, Robotics and Autonomous Systems 35 (1) (2001) 97–108.

[4] G. Dissanayake, P. Newman, H. Durrant-Whyte, M. Csorba, A solution to the simultaneous localization and map building, IEEE Transactions on Robotics and Autonomation 17 (3) (2001) 229–241.

[5] S. Thrun, A probabilistic online mapping algorithm for teams of mobile robots, International Journal of Robotics Research 20 (5) (2001) 335–363.

[6] S. Thrun, D. Fox, W. Burgard, F. Dellaert, Robust monte carlo localization for mobile robots, Artificial Intelligence 128 (1-2) (2000) 99–141.

[7] J. Cortés, S. Martínez, T. Karatas, F. Bullo, Coverage control for mobile sensing networks, in: Proceedings of the IEEE International Conference on Robotics and Automation, Arlington, VA, 2002, pp. 1327–1332.

[8] B. Jung, G. Sukhatme, Multi-target tracking using a mobile sensor network, in: Proceedings of the IEEE International Conference on Robotics and Automation, Washington, DC, 2002, pp. 1050–1055.

[9] B. Horling, R. Vincent, R. Mailler, J. Shen, R. Becker, K. Rawlins, V. Lesser, Distributed Sensor Network for Real Time Tracking, in: Proceedings of the 5th International Conference on Autonomous Agents, ACM Press, Montreal, 2001,

pp. 417–424.

URL http://mas.cs.umass.edu/paper/199

[10] D. S. Hochbaum, Approximation algorithms for NP-hard problems, Boston : PWS Pub. Co., 1997.

[11] M. R. Garey, D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, San Francisco, CA, 1979.

[12] Y. Crama, F. C. R. Spieksma, Approximation algorithms for three-dimensional assignment problems with triangle inequalities, European J. Oper. Res. 60 (1992) 273–279.

[13] R. Burkard, R. Rudolf, G. Woeginger, Three-dimensional axial assignment problems with decomposable cost coefficients, Discrete Applied Mathematics 65 (1996) 123–139.

[14] D. Goossens, F. Spieksma, On the focus of attention problem, personal communication (2004).

[15] M. Halldorsson, Approximating discrete collections via local improvements, Proc. of Sixth SIAM/ACM Symposium on Discrete Algorithms (1995) 160–169.

[16] G. Yu, O. Goldschmidt, Local optimality and its application on independent sets for k-claw free graphs, Journal of Combinatorial Optimization (1997) 151–164.

[17] J. Spletzer, C. Taylor, A framework for sensor planning and control with applications to vision guided multi-robot systems, in: Computer Vision and Pattern Recognition Conference, Kauai, Hawaii, 2001.

[18] M. Isard, A. Blake, Condensation – conditional density propagation for visual tracking, International Journal of Computer Vision 29-1 (1998) 5–28.

[19] A. Kelly, Introduction to mobile robots–lecture notes (2001).