Target Tracking with Distributed Sensors: The Focus of Attention Problem

University of Pennsylvania Technical Report MS-CIS-03-23

Volkan Isler, John Spletzer, Sanjeev Khanna, and C.J. Taylor

Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, 19104
{isleri, spletzer}@grasp.cis.upenn.edu, {sanjeev, cjtaylor}@cis.upenn.edu

Abstract

In this paper, we investigate data fusion techniques for target tracking using distributed sensors. Specifically, we are interested in how 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 PTAS for when the cameras are equidistant, and a 1.42 approximation 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 these 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.

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 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. It would be unrealistic to assume that each sensor could track multiple targets or that the network possessed unlimited computational power and bandwidth. With this in mind, our problem can be viewed as an optimal allocation of resources for target tracking. How should 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 [1]. 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 sensors pairs* to targets. This added dimension changes the 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 [12, 5]. Thrun *et al* have also contributed significant research to these areas [14, 15]. 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 [3]. This 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 [10]. 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* [6] 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 [9] that will be used in the paper:

Definition 1 *A polynomial algorithm, $\mathcal{A}$, is said to be a α-approximation algorithm, if for every problem instance I, $\mathcal{A}$ produces a solution within a factor of α 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 is a cost function $c(i,j,k)$ which indicates the cost of tracking target k using sensors i and j where $i,j \in [1 \dots 2n]$ and $k \in [1 \dots n]$. 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 [7]:

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 (that is 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 [4] and inapproximable [2]. 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.

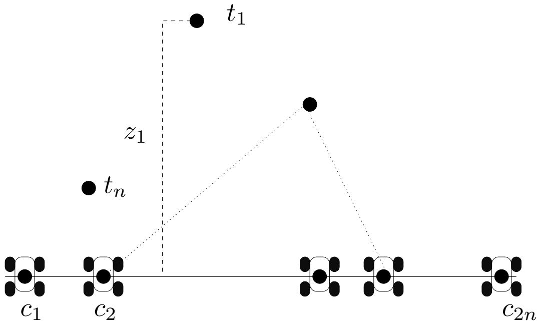

Figure 1: The Focus of Attention Problem on the line.

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

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 $\frac{Z_k}{b_{ij}}$ where Z_k is the vertical distance of the target k to the line l and b_{ij} is the baseline, that is the distance between the two cameras (see figure 1). 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 [11] . Note that this error metric fails if the targets are very close to l, therefore in this section we assume that there exists a minimum clearance δ such that $Z_i > \delta$, for all targets i.

Suppose that the cameras are sorted from left to right and let c_i be the coordinate 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, $Z_1 \leq Z_2 \leq \dots \leq Z_n$ and b_i be the baselines in an optimal assignment sorted such that $b_1 \leq b_2 \leq \dots b_n$. There exists an optimal matching such that the target at depth Z_i is assigned to the pair with baseline b_i.*

Proof: Suppose not. Then there exists two assignments (Z_i, b_j) and (Z_k, b_i) such that $Z_i > Z_k$ and $b_j < b_i$. But

[1] In fact, a better approximation is Z^2/b, but 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 Z/b

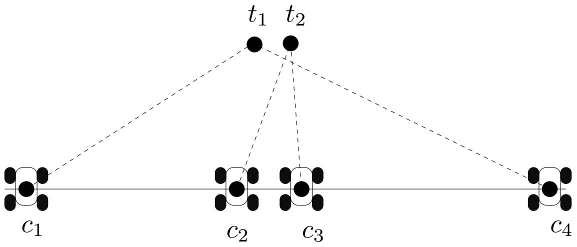

Figure 2: A greedy assignment assigns c_1 and c_4 to target t_1 and gets stuck with the pair (c_2, c_3). The optimal assignment in this case is to assign t_1 to (c_1, c_3) and t_2 to (c_2, c_4).

then

$$\begin{aligned} \frac{Z_i}{b_j} + \frac{Z_k}{b_i} &> \frac{Z_i}{b_i} + \frac{Z_k}{b_j} \\ \frac{(Z_i - Z_k)}{b_j} &> \frac{(Z_i - Z_k)}{b_i} \end{aligned}$$

which is always true. Therefore we could improve the optimal matching by swapping the pairs leading to a contradiction. ■

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: Consider the setting in figure 2 with four cameras where the two cameras 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 [4].

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

First consider the matching (c_1, c_{2n}) and (c_{4i-2}, c_{4i}), (c_{4i-1}, c_{4i+1}) for $i = 1, 2, \dots, (2n-2)/4$. 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 can not 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. Let l_i (resp. l^*_j) be the baselines of the pairs generated by our (resp. optimal) algorithm. 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.

Lemma 5 *$\forall i \exists j$ such that $l_i \geq l^*_j$.*

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

Let $A = \{c_k, c_{k+1}, \dots, c_{n+k}\}$. 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_k. ■

Lemma 6 *Let $S = \{l_1, \dots, l_n\}$ and $OPT = \{l^*_1, \dots, l^*_n\}$. 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 (by induction): 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. W.l.o.g. 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 $C = \{c_i, c_{i+1}, \dots, c_j\}$.

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}):

$$\sigma_k(l) = \begin{cases} \sigma_{k-1}(l), & \text{if } l \in A' \quad (3.1) \\ l^*, & \text{if } l = (c_i, c_{n+i}) \quad (3.2) \end{cases}$$

■

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

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 3).

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.

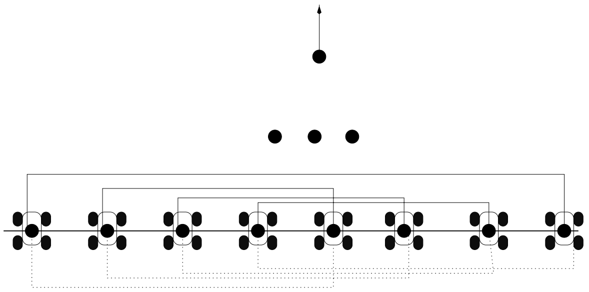

Figure 3: The matchings produced by our algorithm (shown in dotted lines) can be twice as bad as the optimal matching (shown in solid lines) by moving the furthest target to infinity.

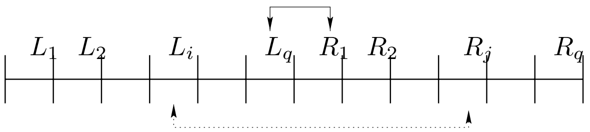

Figure 4: Partitioning the line segment: A small edge (solid line) and an edge of type (i, j) (dotted line).

Our cost in this case is is $\frac{\mathcal{Z}}{1+\epsilon} + (n-1)\frac{\epsilon}{1+\epsilon}$ which is $2OPT$ for large enough $\mathcal{Z}$.

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

Theorem 7 *There exists an $O(n)$-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 $\frac{Z_k}{b_{ij}}$ where b_{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 A PTAS for equidistant cameras

Our next result is a PTAS for equidistant cameras on the line. Let $\epsilon > 0$ be a fixed constant. We are going to present an algorithm that computes a $(1 + O(\epsilon))$-approximate solution. Without any loss of generality assume that the distance between two consecutive cameras is 1, hence the length of the line segment is $2n$.

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

Proof: Assume c_i is matched to c_j, $i, j \leq n$ in an optimal matching. 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. ■

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

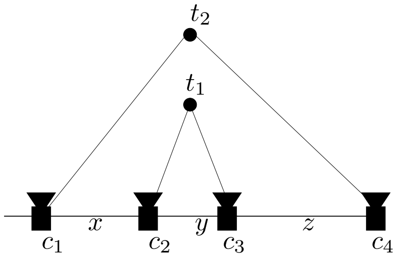

Figure 5: Figure for lemma 10

block has p 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 L_i and y is in R_j.

Clearly, there are q^2 (i.e. constant, for a given ϵ) number of different types. We will enumarate all possible matchings by guessing the number of edges in each type.

Lemma 9 *An edge is called small if its length is less than ϵn. The number of small edges is at most ϵn.*

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

Given a guess, we use the following rule to match the cameras. Fix a block on the left, say L_i. Suppose L_i is connected to x_1 vertices in R_1, x_2 vertices in R_2, etc. Pair the x_1 leftmost vertices in L_i to x_1 leftmost vertices in R_1. Then x_2 leftmost among remaining ones and so on.

This ensures that small edges in OPT are reduced by at most a factor of 2.

Lemma 10 *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 5). 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 5). 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. ■

Lemma 11 *Let the weight of 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. The total weight on the small edges is at most an 64ϵ fraction of the overall weight in any optimal assignment.*

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 $B = \{b_1, \dots, b_{\frac{n}{2}}\}$ be the set of any $\frac{n}{2}$ "big" edges that match cameras from M to cameras in N and $S = \{s_1, \dots, s_k\}$ 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 10 with $x + y \leq n + \epsilon n$, $x + y + z \geq \frac{n}{2}$ and $y + z \geq \frac{n}{4}$ we get:

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

Let $w(S)$ be the total weight in set S. Since a baseline can be of length at most $2n$, 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 12 *There exists a PTAS for assigning equdistant cameras on a line.*

Proof: The matching described ensures that small edges in OPT are reduced by at most a factor of 2 and long edges are within a factor of $(1 + \epsilon)$. Using Lemma 11 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 a pair of range sensors (c_1, c_2) and a target t is approximated by $\frac{1}{\sin\theta}$ where $\theta = \angle c_1 t c_2$. This is the Geometric Dilution of Precision (GDOP) for sensors that measure distances from the targets [11]. 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 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 [2] .

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 6). 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 $c_1 c_2$. Note that $Arc2$ lies on $\mathcal{C}$.

We call the region inside $Arc1(c_1, c_2)$ and $Arc2(c_1, c_2)$ a *defective* region for the pair (c_1, c_2), because any target outside this region is viewed by an angle less than $\frac{3\pi}{4}$ and greater than $\frac{\pi}{4}$ degrees from (c_1, c_2). 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. We summarize the properties of defective regions in the following propositions, which can be proven using basic geometric formulas.

Proposition 13 *Any target outside the defective region of sensors c_1 and c_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 14 *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 6 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 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).

- if t_2 is also defective, 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 the defective region of (c_1, c_2) 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 the defective region of (c_1, c_2) 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 figure in figure 6 illustrates, by construction the defective regions only intersect at the sensors. This makes each assignment to have an error of 1.42 at most. In addition, once an assignment

[2] In practice one would run a matching algorithm, however this does not affect the analysis.

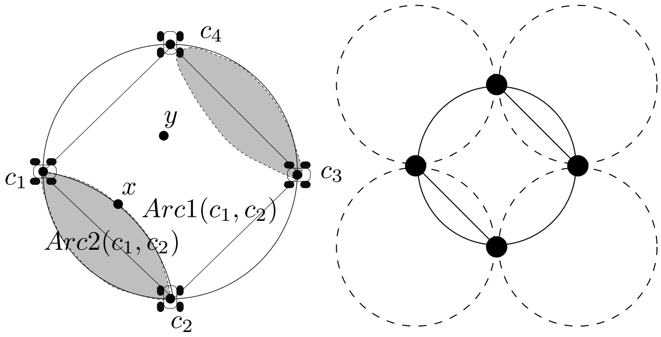

Figure 6: Sensors on circle: **LEFT:**The defective region for sensors c_2 and c_2 is the shaded area defined by arcs $Arc1(c_1, c_2)$ and $Arc2(c_1, c_2)$. **RIGHT:** The defective regions are disjoint.

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 15 *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 $\frac{1}{\sin \angle ikj}$.*

Discussion: Universal Placement Note that the analysis above shows that the equidistant placement for $\frac{1}{\sin\theta}$ metric is *universal*: No matter where the targets are located, our algorithm guarantees a 1.42-approximation for 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 Z/b metric would be to put half of the cameras to x and the other half on y, which guarantees an optimal assignment for this metric.

3.4 Arbitrary Sensor Placement

The inapproximability of FOA for general sensor placement 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.

When the error metric is arbitrary, this problem is equivalent to *Maximum 3-Set Packing*[3], which is known to be NP-hard [7]. It is also known that a greedy solution is within a factor of 3 of optimal. A "2-locally-optimal" solution is defined as a maximal solution that can not be improved further by removing any item from the current solution, and attempting to insert 2 non-conflicting items. It has been shown that any 2-locally optimal solution provides a $\frac{5}{3}$ approximation [8, 16].

One might suspect that a 2-locally optimal solution would yield better performance for restricted error metrics. However, this is not the case: Consider the example in figure 7 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 on the left in figure 7. 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 equispaced cameras on a line.

We further investigate the utility of the greedy and 2-local algorithms in Section 4.

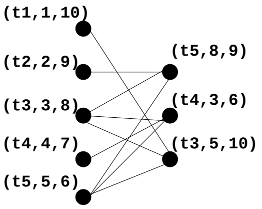

Figure 7: The conflict graph for the optimal solution and a 2-local solution: The nodes on the left (resp. right) represent the optimal (resp. 2 local) solution. There is an edge between two nodes if the assignments conflict. In this case, a 2-local solution gives a $\frac{5}{3}$ approximation, showing that the analysis is indeed tight.

4 Simulation Results

We implemented simulations for several of our results in both target tracking and cooperative localization tasks. We contrasted the performance of these empirically with greedy approaches. Each sensor was constrained to tracking a single target at any given time. For the line and circle cases, no limitations were assumed regarding sensor range.

[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.

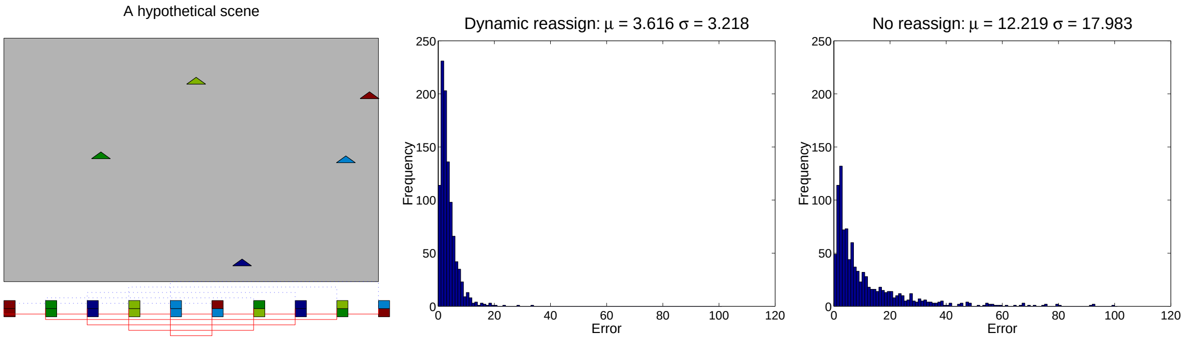

Figure 8: **Left:** A tracking scenario with targets performing a random walk. **Middle, Right:** Histograms of the mean-squared error (MSE) for 2-approximation and greedy algorithm with no re-assignment.

4.1 Cameras on the line

In this simulation, we modeled the target tracking task as outlined in Section 3.2. Specifically, we considered 10 cameras charged with tracking 5 targets performing a random walk as shown in Figure 8. The sensors measured bearings to targets. Measurements from pairs of sensors were 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 the 2-approximation algorithm presented in Section 3.2. In this approach, sensor pairs communicated target position estimates (requiring $O(n)$ communications), and sensor pair-target assigments were dynamically updated as necessary.

We simulated the performance of these two algorithms for 1000 iterations. The error in bearing was simulated by drawing samples from zero mean Gaussian with $\sigma = 1^\circ$. The middle figure in Figure 8 is the histogram of the average error for the dynamic update method. The mean squared error is 3.62 and the standard deviation of the error is 3.22. In this simulation, the 2-approximation algorithm performs better than the no-optimization version (given in Figure 8 right), whose mean error is 12.22 and the standard deviation of the error is 17.98.

4.2 Sensors on the circle

Target tracking need not be adversarial. We demonstrate this in a cooperative localization task. 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 fuse with odometry information. Both the sensor and odometry measurements are corrupted with random Gaussian noise. Additionally, the odometry measurements have an unmodeled bias (to reflect wheel imperfections, for example). Each robot employs a particle filter to fuse the imperfect odometry and sensor measurements to estimate its position.

Again, we modeled two algorithms for this scenario. Both initiated with a globally optimal assignment of sensor pairs to targets. In Algorithm 1, this assignment was maintained throughout the simulation. Algorithm 2 followed the 1.42-approximation as outlined in Section 3.3. In this case, reassignment of sensor pairs to targets was constrained to within the initial 4 sensor/2 target assignment. 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, and the knowledge of sensor positions, allowed each robot to condition its particle filter set for improved position estimation. The procedure then iterated.

Localization performance for both algorithms is reflected in Figure 9. In this example, 8 robots were tracked by 16 sensors. The robots localized while following pseudo-random trajectories through the network. As expected, results indicate that although both approaches rely on identical filtering techniques, significant improvements in localization performance can be achieved by intelligently assigning targets to sensors prior to the data fusion phase.

4.3 Arbitrary Sensor Placement

In this last simulation, we examined the arbitrary sensor placement problem as outlined in Section 3.4. 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 [13].

Two algorthms were investigated for this maximization

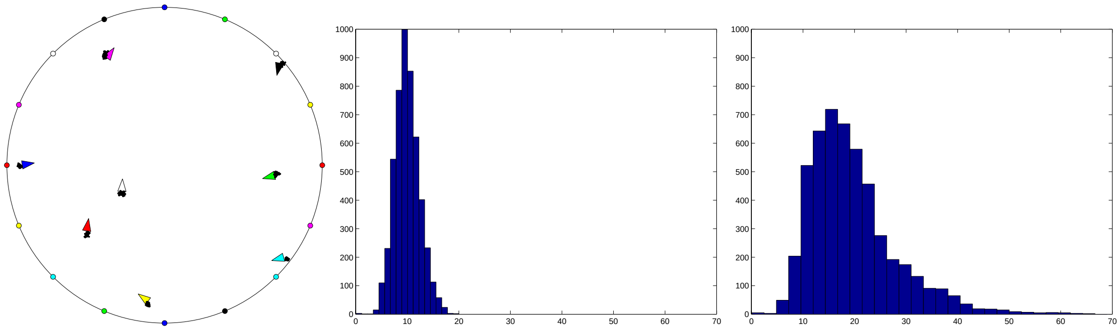

Figure 9: **Left:** Simulator snapshot showing robot positions overlaid with corresponding particle set estimates. **Center, Right:** MSE histograms for the 1.42-approximation and static assignment trials. The former reduces MSE in position by 50%.

approach. The first employed a greedy assignment strategy, and the second a 2-locally optimal approach as discussed in Section 3.4. 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 timestep. Several trials were conducted corresponding to sparse and dense solution sets. Data from a representative trial can be found at figure 10.

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 significantly greater computational complexity of 2-local may not be warranted by the expected performance improvement over greedy for real-time applications.

5 Conclusions and Discussion

In this paper, we have introduced the focus of attention problem for 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.

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. As a consquence, a greedy strategy may be preferred for real-time applications.

References

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

[2] R.E. Burkard, R. Rudolf, and G.J. Woeginger. Three-dimensional axial assignment problems with decomposable cost coefficients. *Discrete Applied Mathematics*, 65:123–139, 1996.

[3] J. Cortés, S. Martínez, T. Karatas, and F. Bullo. Coverage control for mobile sensing networks. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1327–1332, Arlington, VA, May 2002.

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

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

[6] B. Horling et al. Distributed sensor network for real time tracking. In *Proceedings of the 5th International Conference on Autonomous Agents*, pages 417–424, 2001.

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

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

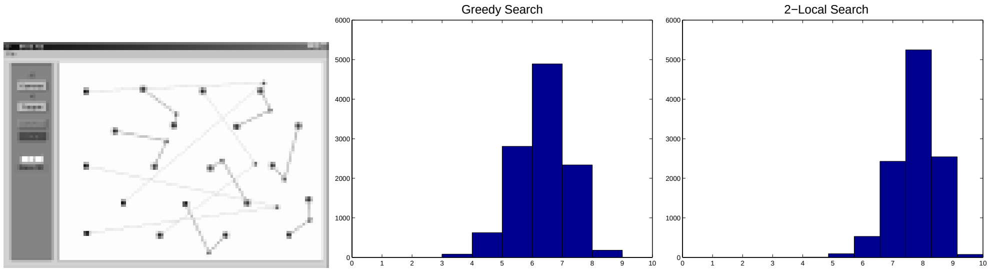

Figure 10: **Left:** Simulator snapshot for 2-local assignment trial. **Center, Right:** The number of valid tracks recovered for greedy and 2-local search strategies. In this example, 2-local improved over greedy by on average 15%.

[9] Dorit S. Hochbaum. *Approximation algorithms for NP-hard problems*. Boston : PWS Pub. Co., 1997.

[10] B. Jung and G. Sukhatme. Multi-target tracking using a mobile sensor network. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1050–1055, Washington, DC, May 2002.

[11] A. Kelly. Introduction to mobile robots–lecture notes, 2001.

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

[13] J. Spletzer and C.J. 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, Dec 2001.

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

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

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