

Triangulation Based Multi Target Tracking with Mobile Sensor Networks

Seema Kamath, Eric Meisner and Volkan Isler

Abstract—We study the problem of designing motion-planning and sensor assignment strategies for tracking multiple targets with a mobile sensor network. We focus on triangulation based tracking where two sensors merge their measurements in order to estimate the position of a target. We present an iterative and distributed algorithm for the tracking problem. An iteration starts with an initialization phase where targets are assigned to sensor pairs. Afterwards, assigned sensors relocate to improve their estimates. We refer to the problem of computing new locations for sensors (for given target assignments) as one-step tracking. After observing that one-step tracking is computationally hard, we show how it can be formulated as an energy-minimization problem. This allows us to adapt well-studied distributed algorithms for energy minimization. We present simulations to compare the performance of two such algorithms and conclude the paper with a description of the full tracking strategy. The utility of the presented strategy is demonstrated with simulations and experiments on a sensor network platform.

I. INTRODUCTION

A mobile sensor network is a network of mobile devices equipped with sensing, communication and computation capabilities. In this paper, we study a typical application of mobile sensor networks: tracking multiple targets. Imagine that we are given (i) a mobile sensor network specified by the locations of the sensor nodes and (ii) a communication graph whose edges correspond to nodes that can communicate with each other. Our goal is to estimate the positions of (possibly many) targets as they move around in the work space. In this scenario, the following questions arise:

- i) Target-sensor assignment: which sensors should track which targets?
- ii) Motion planning: how should the sensors move, so that the overall quality of the estimation improves?

In answering these questions, one must address the following issues:

The coupling between assignment and motion planning. As the sensors move, the optimal target-sensor assignments will change. Therefore, these assignments must be updated in a dynamic fashion.

Dependencies between sensors. Sensor network nodes are typically quite modest in terms of their sensing capabilities. Consequently, sensor nodes must collaboratively obtain position estimates. However, this raises the following issue. Suppose three sensors s_1, s_2 and s_3 are tracking targets t_1, t_2 and t_3 . Sensor pair (s_1, s_2) is assigned to t_1 , (s_2, s_3) to t_2 and (s_1, s_3) to t_3 . The quality of tracking t_1 depends

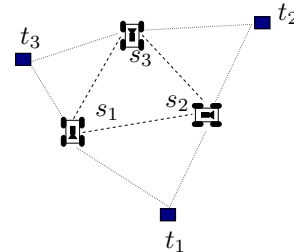


Fig. 1. A setup where three targets are tracked by three sensors. Sensors s_1 and s_2 are tracking target t_1 , s_2 and s_3 are tracking t_2 , and s_3 and s_1 are tracking t_3 .

also on the position of s_2 . Therefore s_1 must negotiate a good position with s_2 . But then s_2 must negotiate a good location with s_3 (because of t_2) who, in turn, must negotiate a good location with s_1 because of t_3 ! Performing these cyclic negotiations in a distributed fashion while improving the quality of the estimates is one of the primary issues addressed in our present work.

Communication and energy constraints. Battery power is the primary limitation determining the lifetime of the network. An important source of power consumption is communication. When two nodes are tracking a target, they will need to exchange information. Therefore, it is desirable that the energy consumption required to transfer the information from one node to another is minimized.

Our results and organization of the paper: In this paper, we study algorithms to compute sensor assignment and motion planning strategies for mobile sensor networks. We focus on triangulation based position estimation where two sensors are utilized to estimate the position of a target.

We start with an overview of related work and continue with a formal statement of the one-step tracking problem (Section III) for which we present an iterative and distributed algorithm. In Section IV-A, we investigate the computational complexity of one-step tracking and observe that the general form is not only NP-hard but also hard to approximate.

In Section IV-B, we show how one-step tracking can be formulated as an energy minimization problem. Even though energy minimization with pairwise interactions is hard to approximate, the problem has been extensively studied and there are algorithms in the literature which work well in practice. In particular, two algorithms, loopy belief propagation and tree-reweighted message passing, have been demonstrated to solve various problems (e.g. data association) in sensor-networks efficiently. After demonstrating the performance of these two algorithms with simulations,

The authors are with the Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY 12180, USA. Emails: {kamath, meisne, isler}@cs.rpi.edu

we present the complete tracking algorithm in Section V. We conclude the paper with results from real experiments conducted on a sensor network.

II. RELATED WORK

Tracking is a fundamental problem in sensor networks and therefore the problem received significant attention recently. In [1], an *information driven sensor query* approach was proposed. In this approach, at any given time, only a single sensor (leader) is active. After obtaining a measurement, the leader selects the most informative node in the network and passes its measurement to this node which becomes the new leader. In subsequent work, researchers addressed leader election, state representation, and aggregation issues [2], [3]. A sensor selection method based on the mutual information principle is presented in [4]. Recently, an *entropy based heuristic approach* was proposed [5] which greedily selects the next sensor to reduce overall uncertainty. Other aspects of tracking in sensor networks have received significant attention as well. In [6], the problem of estimating target's location and velocity using minimal information has been addressed. The problem of assigning n disjoint pairs of sensors to n targets so as to minimize the overall error in the estimation has been studied in [7]. The problem of choosing the best subset of cameras for a given placement has been studied recently in [8]. A related line of research is cooperative localization, where a group of robots or network-nodes localize themselves by collecting information over the network [9]–[11].

For mobile sensor networks, the problem studied in our present work is related to the coverage problem. The problem of relocating sensors to improve coverage has been studied in [12]. In this formulation, the sensors can individually estimate the positions of the targets. However, the quality of coverage decreases with increasing distance. The algorithm presented in this paper studies a similar problem but for sensors which cannot estimate the targets' positions by themselves.

The problem of controlling the configuration of a sensor team which employs triangulation for estimation has been studied in [13] where the authors present a numerical, particle-filter based framework. A recent related result was presented in [14] where the problem of relocating a sensor team whose members are restricted to lie on a circle and charged with jointly estimating the location of the targets was studied.

The one-step tracking problem (Section III) has been studied recently in [15] for the special case where the communication graph is a tree. In this paper, we present (i) a solution to the general case and (ii) a full description of a distributed, iterative tracking algorithm. In solving the one-step tracking problem we utilize message-passing based distributed estimation algorithms. These algorithms have been utilized in solving other problems in sensor networks and demonstrated to work well in practice [16]–[19].

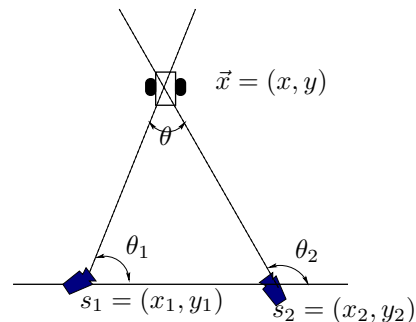


Fig. 2. The uncertainty in estimating the position of the target at x is given by: $U(s_1, s_2, x) = \frac{d(s_1, x) \times d(s_2, x)}{\sin \theta}$

III. PROBLEM STATEMENT

Let $S(t) = [s_1(t), s_2(t), \dots, s_n(t)]$ be the state of the sensor network at time t where $s_i(t)$ is the state (position) of the i^{th} sensor at the time. Let us construct a dynamic graph $G(t) = (V(t), E(t))$ whose nodes $V(t)$ correspond to the set of sensors at time t and there is an edge between two nodes if the corresponding nodes in the network can communicate without spending more energy than a given budget. This energy budget can be specified as, for example, the number of hops required to transmit a message.

In this paper, we focus on triangulation based estimation where two sensors s_i and s_j can estimate the position of an object at location x . However, the uncertainty in this estimation is given by a function $U(s_i, s_j, x)$. For example if the sensors measure bearing then $U(s_i, s_j, x) = \frac{d(s_i, x)d(s_j, x)}{\sin \angle s_i x s_j}$ where $d(s_i, x)$ is the Euclidean distance between s_i and x and $\angle s_i x s_j$ denotes the angle between the target and the sensors [20]. Therefore if the target is collinear with the sensors, the uncertainty becomes infinite which leads to an arbitrarily bad estimation (Figure 2). We will use this uncertainty measure throughout the paper to demonstrate our results.

Suppose at time t we are given the location $x(t)$ of a target, the uncertainty function, the state of the sensor network $S(t)$ and the communication graph $G(t)$. We can define a function

$$\text{assign}(x(t), S(t), G(t)) = \arg \min_{(s_i, s_j) \in E(t)} U(s_i, s_j, x(t))$$

which returns the best sensor pair in the network for tracking the target at $x(t)$. Similarly, we define the error in tracking this target as

$$\text{error}(x(t), S(t), G(t)) = U(\text{assign}(x(t), S(t), G(t)), x(t))$$

which is simply the uncertainty achieved by the optimal selection. Suppose there are m targets and let $x_i(t)$ be the position of the i^{th} target at time t . We are now ready to define the *tracking problem*:

Compute an admissible trajectory $s_i(t)$ for each sensor in the network so as to minimize the overall uncertainty in tracking given by:

$$\frac{1}{mT} \sum_{i=1}^m \int_{t=0}^T \text{error}(x_i(t), S(t), G(t)) \quad (1)$$

where T denotes a (given) terminal time.

Clearly, the value given by the Equation 1 for the optimal solution of the tracking problem is the best possible error that can be achieved by the network. However, in practice, there are significant challenges in computing trajectories that achieve this value. First of all, the target trajectories may not be available. Second, the sensor pair for a target can change from one sensor pair to another instantaneously and hence computing the best pair for the next time instance in a distributed fashion can be very difficult. A related issue is that the graph $G(t)$ depends on the locations of the nodes as well as the environment (occlusions may prevent communication) and the nodes may not have access to the entire graph at all times.

To deal with these issues, we propose a solution where the time interval $[0, T]$ is divided into epochs of possibly varying length. Each epoch starts with an initialization phase, where the network computes target/sensor-pair assignments based on the estimated positions of the targets and its current state (we defer the details of initialization and a full description of the sequence of events in an epoch to Section V). Once the initialization phase is complete, we ask the question: Given target/sensor-pair assignments, where should each sensor move to minimize the error until the end of the epoch? We call this problem the *one-step tracking problem*. Formally, the one-step tracking problem is to compute a location s_i for each sensor such that s_i is reachable within the epoch, so as to minimize

$$\sum_{i=1}^m U(s_i^1, s_i^2, x_i) \quad (2)$$

where x_i is the position of target i at the beginning of the epoch and s_i^1 and s_i^2 are the computed locations of the two sensors assigned to target i during this epoch.

In essence, we force the target/sensor-pair assignments to remain static within an epoch. This is justified, because as we will see shortly, the length of an epoch is required to be only long enough to pass a few rounds of messages across the network – which is much shorter than the time it takes for either the sensor nodes or the targets to travel a significant distance. Therefore, the solutions to one-step tracking problems can be combined into a filter to solve the tracking problem: compute the assignments, compute the next locations, repeat.

Before we present the details of the overall algorithm, let us start with the one-step tracking problem.

IV. ONE-STEP TRACKING PROBLEM

In this section, we first establish the computational complexity of the one-step tracking problem. Next, we show how to formulate the problem as an instance of energy minimization with pairwise constraints. This allows us to utilize distributed message passing algorithms for energy

minimization. We show how two of these algorithms which have been extensively studied in the literature can be adapted to solve the one-step tracking problem and compare them in the context of one-step tracking with simulations.

A. Hardness results

In this section, we present computational lower bounds on the hardness of the one-step tracking problem. One-step tracking is closely related to the well-known graph coloring problem.

In the graph k -coloring problem, we are given a graph $G = (V, E)$ and the goal is to choose one of the k available colors for each vertex in such a way that adjacent vertices receive different colors. The coloring problem is NP-hard. Further, unless $P = NP$, it is not possible to find efficient algorithms to approximate coloring within a factor better than $o(|V|)$ [21].

Graph coloring can be reduced to one step-tracking as follows: Given a graph G with n vertices, we produce an instance of one-step tracking with n sensors. The communication graph is identical to G . We assign a target for each edge. There are k available locations where sensors can move (each location corresponds to a color). The cost of tracking a target is infinite if the sensors move to the same location. It is zero otherwise. It is easy to see that G is k colorable if and only if there is a solution to the one-step tracking problem that achieves zero error.

Since graph coloring can not be approximated in general, it is also not possible to find a general solution to solve the one-step tracking problem. In the next section, we show how one-step tracking can be formulated as an energy minimization problem.

B. Energy minimization formulation

Energy minimization with pairwise constraints, also known as the graph labeling problem is defined as follows. We are given a graph $G = (V, E)$ and a finite set of labels $L = \{l_1, \dots, l_k\}$. The objective is to assign labels to the nodes of the graph to minimize the energy given by

$$\sum_{v \in V} E_1(v, l(v)) + \sum_{(u,v) \in E} E_2(u, l(u), v, l(v)) \quad (3)$$

In the equation above $l(v)$ denotes the label of node v . The function E_1 is the cost of assigning a label to a node and is known as the data cost. The function E_2 is the smoothness term which is a function of the labels assigned to the endpoints of edges.

We express the sensor placement task as a graph labeling problem as follows: the labels are possible locations for nodes to travel until the end of an epoch. The data cost is the energy spent in traveling from the node's current location to the assigned location. The smoothness function is obtained by summing the quality of coverage $U(\cdot)$ on all targets for a given placement of the sensors (see Figure 2 for an example uncertainty function).

Energy minimization on graphs with cycles is NP-hard (it is harder than the graph coloring problem.). Nevertheless,

the problem has been extensively studied and researchers have designed algorithms which work well in practice by exploiting substructures in the graph. In particular, two algorithms, loopy belief propagation (LBP) and tree-reweighted message passing (TRW), have been demonstrated to solve various problems (e.g. data association) in sensor-networks. Descriptions of the LBP and TRW algorithms can be found in [22] and [23] respectively. In Section III, we evaluate the performance of these algorithms for the one-step tracking problem using simulations. We first describe the full tracking algorithm in the next section.

V. FULL TRACKING ALGORITHM

The tracking algorithm consists of a series of epochs. The sequence of events within an epoch is shown in Figure 3. We assume that during an epoch a spanning tree T of the communication graph is available and, further, that this tree is rooted at an arbitrary node to establish child-parent relationships. Building a spanning tree in distributed manner is a well-studied problem. We refer the reader to [16] for details.

We use the notation $T(u)$ to denote the subtree of T rooted at node u .

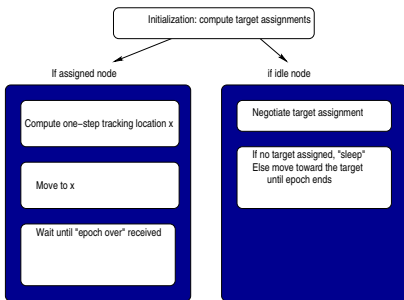


Fig. 3. Sequence of events in an epoch.

The first step in an epoch is *initialization*. All nodes contribute to the initialization phase. The process starts from the leaves. Let u be a leaf node and v be its parent. Node u passes a message to v . The message consists of u 's own location along with a list of targets that u can contribute in tracking. The initialization process propagates up to the root as follows: Let w be an internal node and $U = \{u_1, \dots, u_k\}$ be its children. Node w will compute a list L_w which has an entry for each target that can be tracked by nodes in $T(w)$. The entry $L_w(x)$ for a target at x contains the best pair in $T(w)$ to track x and the expected error in tracking x with this pair. The value of the error is computed by the position of the target and the sensor at the time of the computation.

The computation of $L_w(x)$ is achieved as follows: Let $G(w)$ be the subgraph of the communication graph G induced by the vertex set $U \cup \{w\}$. Upon receiving tables L_{u_i} from all of its children, node w compares the values in L_{u_i} with the values obtained from the pairs in $G(w)$ and picks the best among them. When all entries of L_w are filled, node w passes L_w to its parent.

After the root node computes its table, the best sensor pair for each target has been computed. The root propagates these assignments downwards toward the leaves. Nodes which are assigned to targets begin the one-step tracking algorithm described in Section IV. When a node reaches its destination, and receives “epoch over” signals from its children, it passes the “epoch over” signal to its parent. The epoch ends when the root has reached its destination and received “epoch over” signals from its children. Nodes which are not assigned may be kept idle to conserve power, or commanded to move toward targets which may lead to favorable configurations in the future.

VI. SIMULATIONS

In this section, we compare the performance of LBP and TRW for the one-step tracking problem. In the first simulation, the targets remained stationary throughout the simulation. In this experiment, the target assignments were not updated dynamically. The evolution of the total uncertainty as a function of the number of epochs in the networks is shown in Figure 4-left. In this experiment, LBP achieved near optimal performance and in fact converged to the value achieved by OPT: the optimal solution obtained by enumeration.

In the next simulation, we updated the target assignments as the sensors moved. Comparing the results with the previous simulation, we observed that target reassignment improves the performance of tracking. The evolution of the total uncertainty as a function of the number of epochs in the networks is shown in Figure 4-middle.

Finally in the last simulation we studied a scenario where the targets are moving. The evolution of the total uncertainty as a function of the number of epochs in the networks is shown in Figure 4-right.

In all these experiments we observed that LBP consistently outperformed TRW and its performance was close to OPT. To further test this observation, the outcomes of 25 randomly generated tracking instances is reported in Figure 5. It is seen that when target reassignment was not utilized, on the average LBP was within factor 1.2 of OPT's performance whereas TRW's average performance was a factor of 1.35 (in these experiments OPT did not utilize reassignment as well.) With reassignment, LBP was within a factor of 1.45 which is better than TRW's performance of 2.1.

It is interesting to note that both LBP and TRW occasionally converged to a better value than OPT. Therefore optimal local performance does not necessarily imply the best performance in time.

Since the average performance of LBP is better than our implementation of TRW, LBP seems to be a better choice for the one step tracking problem in our experiments.

VII. EXPERIMENTS

The experiments were conducted on a sensor network platform consisting of Acroname Garcia robots equipped with Intel Stargate boards and Ambicom wireless cards. Logitech Pro 4000 webcams were mounted on each of these

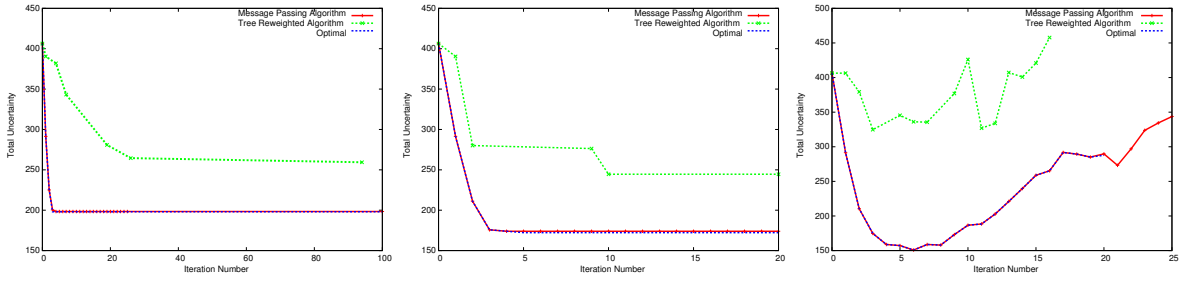


Fig. 4. Each figure shows the evolution of total uncertainty for the three algorithms OPT, LBP and TRW. The leftmost figure corresponds to the case when the targets are static. The second figure from the left shows the total uncertainty over successive epochs when the sensor assignment were updated at the end of every epoch. The rightmost figure shows the total uncertainty over epochs for moving targets with sensor assignments being updated at the end of each epoch.

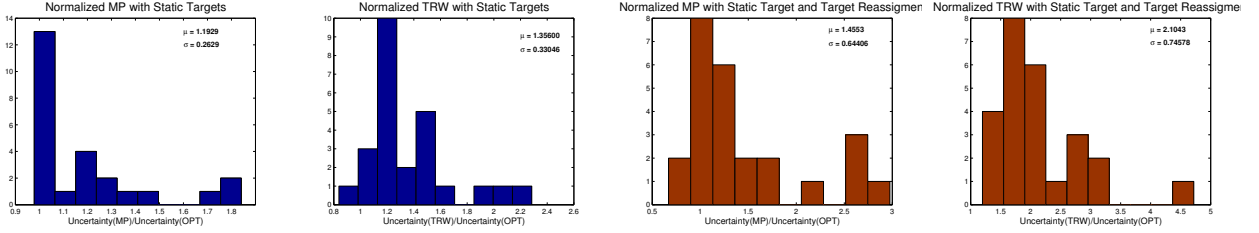


Fig. 5. **LEFT TWO FIGURES:** Static targets, no reassignment. Performance of LBP (left) and TRW (right). **RIGHT TWO FIGURES:** Static targets, with reassignment. Performance of LBP (left) and TRW (right). The numbers on the x-axes indicate the performance of an algorithm normalized with the performance of OPT.

robots and used as the sensor. A Point Grey Bumblebee camera was mounted on the ceiling and was used to localize the robots.

The setup for the experiment consisted of three robots (r_1, r_2, r_3) tracking one target (t) as shown in Figure 7. The communication graph for the robot network was static and had links between r_1 and r_2 and r_1 and r_3 . There were no loops in the network. The tracking algorithm was run for the case where the target is stationary over consecutive epochs. The robots were placed such that $\angle r_1 t r_2$ was approximately 90 degrees and $\angle r_1 t r_3$ was approximately 45 degrees. The robots were localized using an external stereo camera and triangulation was performed using correspondences in the images of the target obtained from webcams mounted on the robots. Using the estimated positions of the target due to the two robot pairs got from the triangulation step and the positions of the robots from the stereo camera, the tracking algorithm (using Loopy Belief Propagation) was run and the best position to move to in the next epoch was estimated. This was done by calculating the distances of the robots to the estimated position of the target and calculating $\angle r_1 t r_2$ and $\angle r_1 t r_3$.

The uncertainty measure was calculated for each robot pair and the target (t) was assigned to the pair having minimum uncertainty. The experiment showed that the robots r_1 and r_2 were assigned to track target t since the uncertainty in estimation due to these robots was the lowest. The ground truth for the position of the target was obtained from the stereo camera. The errors between the estimated target position and the ground truth for the target were calculated for both the robot pairs. It was seen that the error in triangulation due to the robots r_1 and r_2 was less than that due to robots r_1

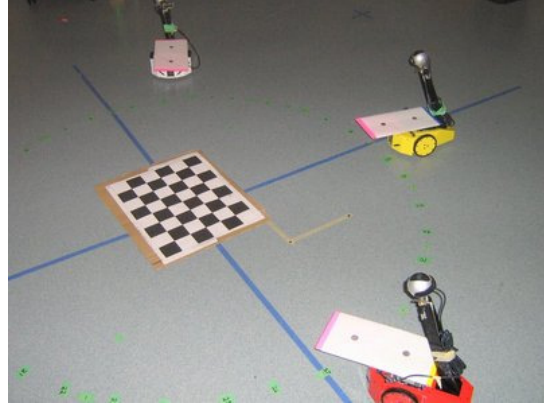


Fig. 7. The experimental setup in which a network of three mobile robots with webcams are tracking a target (checkered board)

and r_3 . The experimental results also clearly showed that the uncertainty due to r_1 and r_2 was less than that due to r_1 and r_3 . The robots then moved to the calculated best position for the next epoch and the experiment was repeated using the new positions of the robots and the pictures taken from the webcams from the new positions.

VIII. CONCLUSION

In this paper, we studied the problem of designing motion-planning and sensor assignment strategies for tracking multiple targets with a mobile sensor network. We presented a distributed strategy for triangulation based tracking where two sensors merge their measurements in order to estimate the position of a target.

An important problem for future research is to incorporate

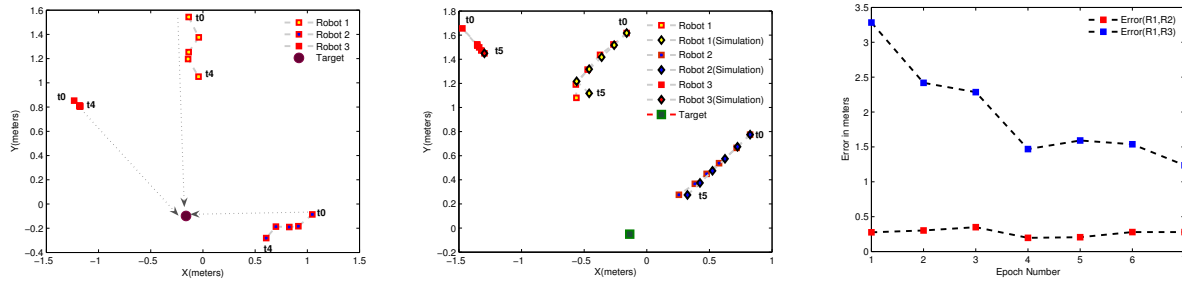


Fig. 6. The leftmost figure shows the path for the robots r_1 , r_2 and r_3 for the experiment where the target is stationary. The center figure shows the path of the robots from simulation and those that are obtained from experiments taking into account the error in robot motion. The rightmost figure shows the plot of the distance between the ground truth for the position of the target and the estimate of the target position obtained from triangulation over successive epochs for the two robot pairs.

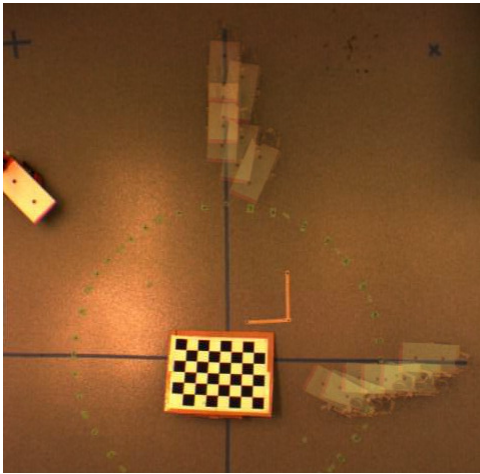


Fig. 8. Snapshots of the robot positions at each epoch superimposed on another. This figure corresponds exactly to the leftmost figure in Figure 6. The target used is the checkered board.

target dynamics into the strategy. Currently, we treat the targets as static within an epoch. However, if accurate representations of the targets' motion are available, then the strategy can be optimized in the time dimension as well. We plan to address this issue in the context of tracking humans in our future work. We are currently conducting extensive experiments using different network configurations.

ACKNOWLEDGMENT

This work is supported in part by NSF CCF-0634823. The authors would like to thank Kostas Daniilidis and Chuck Stewart for useful discussions.

REFERENCES

- [1] J. Liu, J. Reich, and F. Zhao, "Collaborative in-network processing for target tracking," *EURASIP JASP*, vol. 4, no. 378-391, 2003.
- [2] F. Zhao, J. Liu, J. Liu, L. Guibas, and J. Reich, "Collaborative signal and information processing: An information directed approach," *Proc. of the IEEE*, vol. 91, no. 8, 2003.
- [3] J. Liu, M. Chu, J. Liu, J. Reich, and F. Zhao, "Distributed state representation for tracking problems in sensor networks," in *Proc. IPSN 2004*. ACM Press, pp. 234-242.
- [4] E. Erten, J. W. F. III, and L. C. Potter, "Maximum mutual information principle for dynamic sensor query problems."
- [5] H. Wang, K. Yao, G. Pottie, and D. Estrin, "Entropy-based sensor selection heuristic for target localization," in *Proc. IPSN 2004*. ACM Press, pp. 36-45.
- [6] J. Aslam, Z. Butler, F. Constantin, V. Crespi, G. Cybenko, and D. Rus, "Tracking a moving object with a binary sensor network," in *Proc. First intl'l conf. on Embedded networked sensor systems*. ACM Press, 2003, pp. 150-161.
- [7] V. Isler, J. Spletzer, S. Khanna, and C. Taylor, "Target tracking in sensor networks: the focus of attention problem," *Computer Vision and Image Understanding Jnl.*, vol. 100, pp. 225-246, October 2005.
- [8] V. Isler and R. Bajcsy, "The sensor selection problem for bounded uncertainty sensing models," *IEEE Tran. Automation Science and Engineering*, 2006.
- [9] L. Doherty, K. S. J. Pister, and L. E. Ghaoui, "Convex position estimation in wireless sensor networks," in *Proc. IEEE Infocom*, 2001.
- [10] A. Howard, M. J. Matarić, and G. S. Sukhatme, "Cooperative relative localization for mobile robot teams: An ego-centric approach," in *Proc. Naval Research Laboratory Workshop on Multi-Robot Systems*, Washington, D.C., Mar 2003.
- [11] J. Spletzer, A. Das, R. Fierro, C. Taylor, V. Kumar, and J. Ostrowski, "Cooperative localization and control for multi-robot manipulation," in *Proc. IROS 2001*.
- [12] J. Cortés, S. Martínez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 2, pp. 243-255, 2004.
- [13] J. Spletzer and C. Taylor, "Dynamic sensor planning and control for optimally tracking targets," *IJRR*, vol. 22, no. 1, pp. 7-20, 2003.
- [14] S. Aranda, S. Martínez, and F. Bullo, "On optimal sensor placement and motion coordination for target tracking," in *Proc. ICRA*, Barcelona, Spain, Apr. 2005.
- [15] V. Isler, "Placement and distributed deployment of sensor teams for triangulation based localization," in *Proc. IEEE ICRA*, 2006.
- [16] M. A. Paskin, C. E. Guestrin, and J. McFadden, "A robust architecture for inference in sensor networks," in *IPSN 2005*.
- [17] A. T. Ihler, J. W. Fisher III, R. L. Moses, and A. S. Willsky, "Non-parametric belief propagation for self-calibration in sensor networks," *IEEE Jnl. of Selected Areas in Communication*, 2005.
- [18] R. Biswas, S. Thrun, and L. J. Guibas, "A probabilistic approach to inference with limited information in sensor networks," in *IPSN 2004*. New York, NY, USA: ACM Press, pp. 269-276.
- [19] L. Chen, M. Wainwright, M. Cetin, and A. Willsky, "Multitarget-multisensor data association using the tree-reweighted max-product algorithm," in *Proc. SPIE Aerosense conf.*, Orlando, FL, March 2003.
- [20] A. Kelly, "Precision dilution in mobile robot position estimation," in *Intelligent Autonomous Systems*, Amsterdam, Holland, 2003.
- [21] M. Bellare, O. Goldreich, and M. Sudan, "Free bits, pcps and non-approximability-towards tight results," in *FOCS '95: Proc. 36th Annual Symposium on Foundations of Computer Science (FOCS'95)*. Washington, DC, USA: IEEE Computer Society, 1995, p. 422.
- [22] K. P. Murphy, Y. Weiss, and M. I. Jordan, "Loopy belief propagation for approximate inference: An empirical study," in *UAI*, 1999, pp. 467-475.
- [23] M. Wainwright, T. Jaakkola, and A. Willsky, "Tree-reweighted belief propagation algorithms and approximate ml estimation via pseudo-moment matching," in *Proc. AISTATS*, 2003.