2011 50th IEEE Conference on Decision and Control and
European Control Conference (CDC-ECC)
Orlando, FL, USA, December 12-15, 2011

Consensus-based Distributed Estimation for Target Tracking in Heterogeneous Sensor Networks

Antonio Petitti, Donato Di Paola, Alessandro Rizzo, Grazia Cicirelli

Abstract— **In this paper the problem of distributed target tracking is considered. A network of heterogeneous sensing agents is used to observe a maneuvering target and, at each iteration, all the agents are able to agree about the estimate of the target position, despite the fact that only a small percentage of agents can sense the target at each time instant. Our *Consensus-based Distributed Target Tracking (CDTT)* is a fully distributed iterative tracking algorithm, in which each iteration is based on two phases: an *estimation phase* and a *consensus* one. As a result, the estimated trajectories are identical for all the agents at each time instant. Numerical simulations and comparison with another target tracking algorithm are carried out to show the effectiveness and feasibility of our approach.**

I. Introduction

In this paper we focus our attention on the *distributed target tracking* problem: a pool of networked agents try to track a mobile target which moves on a given field. Usually, each agent individually performs an estimate of the target position; then a global estimate is carried out through computing and communicating between the network nodes. It is assumed that all the sensors have a limited sensing range, and consequently that, at a given time, the target is sensed by only a subset of the sensors in the network. Several target tracking algorithms presented in literature rely on some form of *centralization,* even though a local connection scheme among agents is very often assumed. Some algorithms make use of a data fusion center [1], some others on local filtering associated with all-to-all communication schemes, which can introduce an heavy communication overhead [2], [3], [4]. An important step towards the realization of distributed approaches has been made in [5], through the introduction of a new generation of distributed Kalman filtering algorithms, also referred as Kalman-Consensus Filters, which rely on a peer-to-peer architecture that reaches a consensus on estimates of local Kalman Filters. The disagreement among local filters is reduced by applying a consensus step right after the estimation step. An extension of this work is then presented in [6] where a hybrid architecture is proposed. Even though this algorithm joins a good performance with an acceptable communication overload, it keeps some aspects of centralized computation.

This work was partially funded by Regione Puglia, A.Q.P. Ricerca, Project Modelli Innovativi per Sistemi Meccatronici, Del. CIPE 20/04, DM01

A. Rizzo is with Dipartimento di Elettrotecnica ed Elettronica (DEE), Politecnico di Bari, 70126 Bari, Italy rizzo@deemail.poliba.it

A. Petitti, D. Di Paola and G. Cicirelli are with the Institute of Intelligent Systems for Automation (ISSIA), National Research Council (CNR), 70126 Bari, Italy antonio.petitti@ge.issia.cnr.it, [dipaola,grace]@ba.issia.cnr.it

In this paper we present a target tracking strategy which is *fully distributed*. The sensors have limited communication and sensing ranges, and only a fraction of them can sense the target at a given time. Moreover, a certain degree of heterogeneity of the sensors is introduced, by considering sensors with different sensing ranges [7]. This assumption implies that the measurement reliability changes along the space. Each iteration of the proposed algorithm consists of two phases: an *estimation phase* and a *consensus* one. In the former phase, each network node estimates the position of the target. If the node can perform a measurement, then it will estimate the target position through the measurement, improved by a Kalman filter. Otherwise, the node will predict the target motion according to the embedded linear motion model of the Kalman filter. In the latter phase, the estimates performed individually by each node converge on a common value by means of a suitable consensus strategy. In our approach, we suitably apply a max-consensus algorithm [8] such that the best individual estimate can propagate through the whole network in a finite number of steps. In particular, each individual estimate made by a node is associated to a *perception confidence value* which quantifies the reliability of the target position estimate. Then, a max-consensus algorithm applied to the perception confidence value will allow all the network nodes to agree on the best estimate available. Simulation results are totally satisfactory, both in absolute terms and when compared to the Kalman-Consensus Filter [6] which, we remind, retains some aspects of centralized computation. Finally, our framework is quite general and is easily extendable to many other problems of distributed estimation.

II. Background

The communication structure of a network is usually represented by a *graph*. A graph $\mathcal{G}$ is a pair $(\mathcal{I}, \mathcal{E})$, where $\mathcal{I} \stackrel{\triangle}{=} \{1, \ldots, n\}$ is a finite nonempty set of *nodes* and $\mathcal{E} \subset \mathcal{I} \times \mathcal{I}$ is a set of ordered pairs of nodes, called *edges*. The existence of an edge $(i, j) \in \mathcal{E}$ denotes that node j can obtain information from node i, but not necessarily vice versa. If the pairs of nodes are ordered the graph is said *directed* (also known as a *digraph*), otherwise if node i and j can always obtain information from each other, that is pairs of nodes are unordered and therefore the existence of link $(i, j) \in \mathcal{E}$ implies that of link $(j, i) \in \mathcal{E}$, the graph is said *undirected*. Let $\mathcal{G} = (\mathcal{I}, \mathcal{E})$ be an undirected graph with *n* nodes; the set of *neighbors* of node i is defined by $\mathcal{N}_i = \{j \in \mathcal{I} : (j, i) \in \mathcal{E}\}$.

978-1-61284-799-3/11/$26.00 ©2011 IEEE

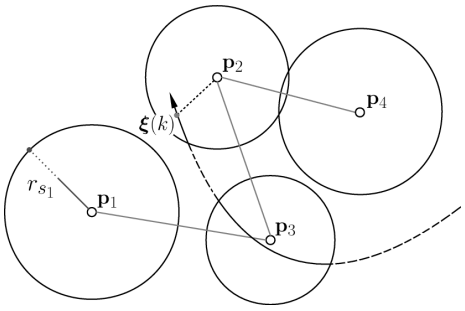

Fig. 1. Abstract representation of the the distributed tracking scenario. The case of four nodes $\{\mathbf{p}_1, \cdots, \mathbf{p}_4\}$ randomly placed in the environment is shown. The communication topology, indicated by the gray links, depends on the communication range r_c (not depicted for clarity). The local sensing area (circles around each node) is defined by the range r_{s_i}, indicated, for clarity sake, only for the node 1, r_{s_1}. The target trajectory $\boldsymbol{\xi}(k)$ over time is depicted with the bold line, that is continuous when the target is in the sensing range of at least one node, and dashed on the opposite case.

The aim of a *consensus protocol* is to make all the nodes in a network agree on some quantity of interest, based only on local communication. In this work we exploit the features of a particular class of algorithms called *max-consensus* [8]. The max-consensus makes all the nodes agree on the maximum value of their initial states. In the discrete time, its update rule is very simple:

$$x_i(k+1) = \max_{j \in \mathcal{N}_i \cup i} \{x_j(k)\}, \tag{1}$$

where x_i is the state variable of the node i and k is the discrete time index.

In accordance with [8], we define the achievement of max-consensus as:

***Definition** 2.1:* Given a directed graph $\mathcal{G}$, an initial vector of information states $x(0) := (x_1(0), \ldots, x_n(0))^T$ and algorithm (1), *max-consensus* is said to be achieved if $\exists l \in \mathbb{N}_0$ such that

$$\begin{aligned} x_i(k) &= x_j(k) \\ &= \max\{x_1(0), \ldots x_n(0)\}, \forall k \geq l, \end{aligned} \tag{2}$$

for all nodes $i, j \in \mathcal{I}$.

If (2) holds for all $x(0)$, a *strong max-consensus* is achieved. If (2) only holds for a subset of all possible $x(0)$, *weak max-consensus* is achieved. For our aims, we will exploit two important properties derived in [8]:

1) given a graph $\mathcal{G}$, strong max-consensus (2) is achieved if and only if $\mathcal{G}$ is strongly connected[1];
2) if strong max-consensus is achieved, then it will be achieved in a number of steps $l \leq \mathcal{D}(\mathcal{G}) \leq n-1$,

where $\mathcal{D}(\mathcal{G})$ is the graph diameter [9].

III. Problem Formulation

In this paper we deal with the *distributed tracking* of a *maneuvering target,* performed by a sensor network in which each node has *limited sensing and communication ranges,* that is to say, only a subset of the nodes in the network can sense the target, and that each node is directly connected to a limited fraction of the nodes in the network. Moreover, each sensor has its own sensing range that differs from sensor to sensor. Fig. 1 shows an abstract representation of the network in the case of four nodes.

We consider a sensor network composed by n nodes which have the objective to track a target moving in the environment $E \triangleq [-L/2, L/2] \times [-L/2, L/2] \subset \mathbb{R}^2$, where $L > 0$, that is a square field[2] with side length *L*. In the following, we will refer to the generic node of the sensor network with the term *agent,* to emphasize the fact that each node of the sensor network must have sensing, computation and communication capabilities.

The target moves along a piece-wise linear trajectory which can be described by a linear switching system as in [6]. In practice, the target moves on a blend of random linear tracts, and when it reaches the boundary of the region, it is pushed back by a force which is orthogonal to the region boundary.

We will refer to the target trajectory as $\boldsymbol{\xi}(k)$, where $\boldsymbol{\xi}(k) \in E$ and k is the discrete time. An agent $i \in \mathcal{I} \triangleq \{1, \cdots, n\}$, is described by the tuple:

$$\langle \mathbf{p}_i, \hat{\boldsymbol{\xi}}_i(k), M_i, \gamma_i(k), r_{s_i} \rangle \tag{3}$$

where the vector $\mathbf{p}_i \in E$ denotes the position of the agent in the environment; the vector $\hat{\boldsymbol{\xi}}_i(k) \in E$ is the estimated position of the target at time k made individually by agent i; the entity M_i is the i-th agent's memory, which contains all the global estimates (*i.e.,* after the agreement of the whole network) of the target position over time. The size of M_i is equal to the number of time instants during which the target is tracked, k_f. In the following we will indicate with $M_i(k)$ the global estimate of the target state performed by the network at time k. Note that the difference between $\hat{\boldsymbol{\xi}}_i(k)$ and $M_i(k)$ is that the former is the target state estimation performed individually by agent i at time k, whereas the latter is the estimate of the target state performed globally over the sensor network and stored in agent i at time k. The value $\gamma_i(k) \in \mathbb{R}$ is called *perception confidence value*, and quantifies the reliability of the information about the estimate of the target position $\hat{\boldsymbol{\xi}}_i(k)$ made by the i-th agent. Its working principle will be clarified later in the paper. The value $r_{s_i} \in \mathbb{R}$ is the sensing range of agent i. Each agent senses the target at time k if and only if $|| \mathbf{p}_i - \boldsymbol{\xi}(k)|| \leq r_{s_i}$.

The network topology is determined by the communication range $r_c \in \mathbb{R}$, that we assume equal for each agent, such that two agents, i and $j \in \mathcal{I}$, can communicate, i.e. a link between them exists, if and only if $|| \mathbf{p}_i - \mathbf{p}_j || \leq r_c$. In this way, the communication scheme of the network is described by an undirected graph $\mathcal{G} = \{\mathcal{I}, \mathcal{E}\}$, where $\mathcal{I}$ is the set of agents and $\mathcal{E} \subseteq \mathcal{I} \times \mathcal{I}$ is the set of edges (links) representing the point-to-point communication channels. We assume a one-hop communication among agents, that is each agent can send and receive messages only with its direct neighbors.

[1]In this paper we make only use of undirected graphs, so the fact of being connected implies that of being strongly connected.

[2]This can be easily extended to convex polyhedra.

Finally, among the many possible aspects of the performance assessment in the target tracking framework [10], in this work we will focus our attention on the *tracking accuracy,* by evaluating the mean square error between estimated and actual target trajectory.

IV. Consensus-based Distributed Target Tracking

The proposed Consensus-based Distributed Target Tracking (CDTT) algorithm is an iterative strategy that, at each iteration, consists of two main phases: *estimation* and *consensus*. During the *estimation phase* each agent performs an individual estimate of the target position at the current time step. If the target is in the sensing range of a given agent, its estimate is carried out through a measurement, improved by a Kalman Filter. On the contrary, the measurement will not be available to the agent at that time, and a prediction will be performed according to the prediction part of the Kalman Filter only. We will use the term *sensing agent* for those agents for which the target lays in their sensing range; the term *predicting agent* otherwise. At the end of the estimation phase, each agent updates its *perception confidence value* $\gamma_i(k)$ on the basis of the *a posteriori* estimate covariance matrix of the Kalman Filter. In the second phase, a max-consensus algorithm on the perception confidence value is run, to make all the agents agree on the agent (or agents) which performed the best estimation of the target position. At the same time, during the execution of the max-consensus algorithm, the best estimate will be propagated among all agents. The CDTT algorithm, running at each iteration the two phases, guarantees the agreement of the whole network on the target position over time, exploiting the convergence property of the max-consensus algorithm, making our approach fully distributed. At the end of the consensus phase, each agent will possess the same information about the target state, that is to say, the best estimate of the target position and the best estimate covariance matrix. Then, a new estimation phase will be started again. To improve the estimation performance, the state of each individual Kalman Filter will be reset to the best available measurement available in the network, and the correspondent best covariance estimate matrix will be used to compute the new Kalman gain.

Note that our algorithm requires the synchronization of agent clocks to rely on a common discrete timeline. This need does not invalidate the distributed nature of our approach in real applications, because the sensor network itself can be effectively exploited to achieve the global synchronization of agent's clocks [11]. Let us now describe in detail the two phases of each iteration. We describe the computation of iteration at time k assuming, therefore, that each agent has available the information in its tuple (3) up to time $k-1$.

A. Phase 1: Estimation Phase

In this phase each agent estimates the target position.

If the agent i is a *sensing agent*, this means that the target is within the sensing range of the agent, *i.e.,* $d_i(k) = || \mathbf{p}_i - \boldsymbol{\xi}(k)|| \leq r_s$, thus the position of the target is measured by the sensing system. The challenge lies in that the measures are inevitably inaccurate and also the speed cannot be measured directly. Then for each *sensing agent* we employ a Kalman Filter to obtain an optimized estimate of target position.

Let us define $\mathbf{x}(k) = [\xi^{[1]}(k), \xi^{[2]}(k), v^{[1]}(k), v^{[2]}(k)]^T$ the target state vector, where $\xi^{[1]}(k), \xi^{[2]}(k)$ are the coordinates of target position and $v^{[1]}(k), v^{[2]}(k)$ the velocity components at time instant k. Assuming that the target moves according to a linear dynamic process, a classical Kalman Filter is used.

To this aim, we assume that the linear dynamic equation for the target movement is given as follows:

$$\mathbf{x}(k) = A\mathbf{x}(k-1) + \omega(k-1)$$

where

$$A = \begin{bmatrix} 1 & 0 & \epsilon & 0 \\ 0 & 1 & 0 & \epsilon \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

is the state transition matrix, ϵ is the time step, and ω is a noise term, that is white Gaussian noise with zero mean and covariance matrix Q, *i.e.,* $\omega(k-1) \sim N(0, Q)$, $Q = \text{diag}(\sigma_Q^2, \sigma_Q^2, \sigma_Q^2, \sigma_Q^2)$.

Let us denote with $\mathbf{z}_i(k)$ the measurement of the target position performed by the agent i at time k as follows:

$$\mathbf{z}_i(k) = H\mathbf{x}(k) + \nu_i(k) \tag{4}$$

where H is the measurement output matrix, defined as:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

and ν_i is the measurement noise, which is white Gaussian with zero mean and covariance matrix R_i, *i.e.,* $\nu_i(k) \sim N(0, R_i(k))$. Supposing that for each agent $\xi^{[1]}$ and $\xi^{[2]}$ are independent variables, the covariance matrix $R_i(k)$ can be defined as $R_i(k) = \text{diag}(\sigma_{R_i}^2(k), \sigma_{R_i}^2(k))$. where we set $\sigma_{R_i}^2(k) = \frac{d_i(k)}{r_s}$, *i.e.,* the higher the relative distance between the agent i and the target, the higher the uncertainty on measurement. Using the above process formulation the following update equations are applied for each *sensing agent* i:

State Prediction

$$\hat{\mathbf{x}}_i^-(k) = A\overline{\mathbf{x}}_i(k-1) \tag{5}$$

Error Covariance Prediction

$$P_i^-(k) = A\overline{P}_i(k-1)A^T + Q \tag{6}$$

Kalman Gain Matrix

$$K_i(k) = P_i^-(k)H^T[HP_i^-(k)H^T + R_i(k)]^{-1} \tag{7}$$

State Estimation

$$\hat{\mathbf{x}}_i(k) = \hat{\mathbf{x}}_i^-(k) + K_i(k)[z_i(k) - H\hat{\mathbf{x}}_i^-(k)] \tag{8}$$

Error Covariance Estimation

$$P_i(k) = [I - K_i(k)H]P_i^-(k) \tag{9}$$

Please note that $\overline{\mathbf{x}}_i(k-1)$ and $\overline{P}_i(k-1)$ are the global (best) estimate of the target position and the global (best) estimate covariance matrix, respectively, computed at time $k-1$ through the max-consensus phase. Their computation will be explained in the next Section.

On the other hand, if the target is not within the sensing range of the i-th agent, that is, i is a *predicting agent*, the target state is estimated by using only the state prediction equation (5).

In both cases, i.e. agent i is a either *sensing agent* or a *predicting* one, the initial guess of target state $\overline{\mathbf{x}}_i(0) = [\hat{\boldsymbol{\xi}}_i(0)^T \hat{\boldsymbol{v}}(0)^T]^T$ is set to $\overline{\mathbf{x}}_i(0) = \mathbf{0}$; this means that the initial estimation of the position of the target coincide with the center of the field, and the initial target velocity is considered null.

Phase 1 ends up with each agent possessing an individual estimate of the target position, $\hat{\boldsymbol{\xi}}_i(k) = \hat{\mathbf{x}}_i(k)$, obtained either by measurement or by prediction. Phase 2 will make all agents agree on a common estimate of the target position, via a max-consensus algorithm. To this aim, the *perception confidence value* $\gamma_i(k)$ is conveniently set during Phase 1 for each agent, in order to quantify the reliability of the estimate of each agent, which will determine the influence of the single agent estimate on the final outcome of Phase 2. In particular $\gamma_i(k)$ is defined as following:

$$\gamma_i(k) = \begin{cases} \dfrac{1}{\mathrm{Tr}\,(P_i(k))} & \text{if } i \text{ is a } \textit{sensing agent} \\ 0 & \text{if } i \text{ is a } \textit{predicting agent} \end{cases} \tag{10}$$

where $\mathrm{Tr}(\cdot)$ is the matrix trace operator. It is clear that $\gamma_i(k)$ grows with the reliability of the estimation performed by the agent i at time k.

B. Phase 2: Consensus Phase

In this phase, a max-consensus based algorithm is run in order to select, in a totally distributed fashion, the agent with most reliable estimates of the target position and to propagate its correspondent estimate through the whole network. This Phase is schematized in Algorithm 1. Agent i sets its internal variables $\zeta_i \in \mathbb{R}$, $\chi_i \in \mathbb{R}^4$, and $\Pi_i \in \mathbb{R}^{4,4}$ to the values obtained from Phase 1, $\gamma_i(k)$, $\hat{\mathbf{x}}_i(k)$, $P_i(k)$, respectively. Then, a max-consensus protocol allows the variable ζ_i to converge to the maximum of the perception confidence values all over the network, that is to say, to select the agent which performed the best estimate of the target position. In correspondence with this selection, suitable variable assignments will allow the agent i to store in its variables, $\chi_i(k)$ and $\Pi_i(k)$, the corresponding estimate and covariance estimate matrix, respectively. It is straightforward to note that the convergence condition of Algorithm 1 is the same of the max-consensus of eq. (1), that is to say, the network must be strongly connected (connected, in our case, in which undirected graphs are adopted). According to [8], the algorithm will converge in at last $n-1$ iterations, so we will make the Algorithm run for the maximum number of cycles needed for convergence. The sampling time connected with the discrete time index t of Phase 2 must be chosen such that Phase 2 is completed before a new instance of Phase 1 can start, that is to say, the sampling time of Phase 2, dubbed as ε, must be such that $t_p + (n-1)\varepsilon < \epsilon$, where t_p is the time required to perform the perception phase.

Algorithm 1 Consensus phase for agent i at iteration k

1: **Input:** $\gamma_i(k)$, $\hat{\mathbf{x}}_i(k)$, $P_i(k)$
2: **Output:** $\overline{\mathbf{x}}_i(k)$, $\overline{P}_i(k)$
3: $\zeta_i(0) = \gamma_i(k)$
4: $\chi_i(0) = \hat{\mathbf{x}}_i(k)$
5: $\Pi_i(0) = P_i(k)$
6: **for** $t = 1$ to $n-1$ **do**
7: $\quad \zeta_i(t) = \max_{j \in \mathcal{N}_i \cup i} \{\zeta_j(t-1)\}$
8: $\quad \nu_i(t) = \arg\max_{j \in \mathcal{N}_i \cup i} \{\zeta_j(t-1)\}$
9: $\quad \chi_i(t) = \zeta_{\nu_i}(t-1)$
10: $\quad \Pi_i(t) = \Pi_{\nu_i}(t-1)$
11: **end for**
12: $\overline{\mathbf{x}}_i(k) = \chi_i(n-1)$
13: $\overline{P}_i(k) = \Pi_i(n-1)$

At the end of the consensus phase, an agreement both on the best estimation of the target state and on the related a posteriori covariance matrix is achieved. Each agent will store these values in its variables $\overline{\mathbf{x}}_i(k)$ and $\overline{P}_i(k)$. These two variables will be fed back to the next Phase 1 of the algorithm, in order to let the Kalman filter of each agent start from the best available estimate and therefore improve the individual prediction performance.

Remark 4.1: It is clear that using a network with a large number of sensors can lead to a detriment of the algorithm performance, in terms of an increasing in the sampling time ϵ. To overcome this drawback, one can keep the number of sensors relatively small and increase the sensing radii, and consequently the coverage ratio ρ, without affecting the algorithm performance (see Section V). Another strategy can be that of adopting networks with a large number of sensors but a relatively small diameter, and strictly limit the number of steps of the consensus algorithm to the network diameter (which must be known at the time the sensor network is designed).

Remark 4.2: We remark that at step 8 of Algorithm 1, a tie-break rule is needed in the case that the maximum value for $\zeta_j(t-1)$ is allocated in more than an agent in the agent's neighbors. This case is of particular interest when the target is out of the sensing range of every agent, that is to say $\gamma_i(k) = 0, \forall i \in \mathcal{I}$. The tie-break rule could be, for example, to choose an index at random among those corresponding to the maximum, or perform, on steps 9 and 10, an average along the involved indices ν_i, rather than a variable assignment.

V. Numerical Results

The performance of the CDTT are analyzed by running a number of Monte Carlo simulations for a generic heteroge-

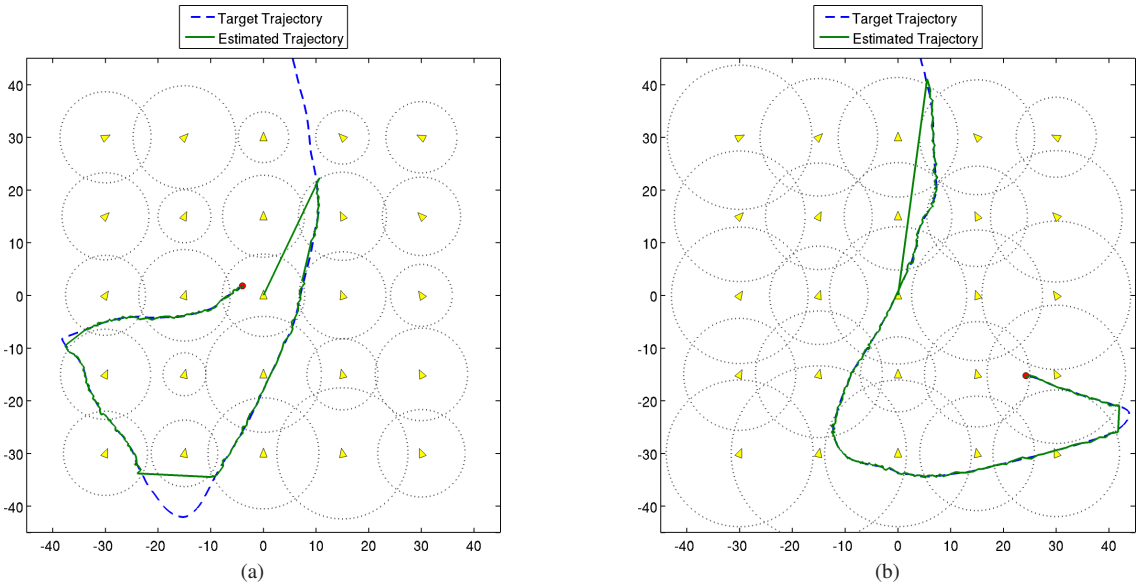

Fig. 2. Two CDTT simulations of random generated target trajectories for a network of $n = 25$ heterogeneous agents. The target comes from the top side of the environment and the filled red circle indicates its last position. (a) A simulation of the network with $\rho = 55\%$. Note that until the target is not sensed by any agent the algorithm does not start.(b) The same network with $\rho = 85\%$. In the latter case, due to the greater number of sensing agent, the global error in the estimated trajectory is reduced.

neous sensor network. We simulate a sensor network tracking the position of a maneuvering target which moves on a square field E with side length $L = 90$, with a motion model as described in Section III, and, for comparison purposes with [6], with communication radius $r_c = (3\lceil\sqrt{n}\rceil) + 2$. Please note that, to satisfy the hypothesis of convergence of the algorithm, one has to check that the network is connected. Heterogeneity in the sensing range of each agent is obtained by picking at random a sensing range for each agent, through a Gaussian stochastic process with average $0.1L$ and standard deviation $0.03L$. The initial guesses of the estimated target position and velocity are $\widetilde{\boldsymbol{\xi}}_i(0) = [0, 0]^T$ and $\widetilde{\mathbf{v}}(0) = [0, 0]^T$. Moreover, the covariance matrix of the Kalman Filter used in Phase 1, is chosen as $Q = k_w^2 I_4$, where $k_w = 5$ and I_m is the m-dimensional identity matrix.

In order to evaluate the global performance of the CDTT, we define two additional parameters. The former, ρ, represents the ratio between the coverage sensing area and the total area of the field E. The latter, φ, represents the average percentage of sensing agents during a single run.

As a metric of target tracking accuracy, the following mean square error (in norm) is computed:

$$\alpha = \frac{1}{k_f} \sum_{k=1}^{k_f} ||M(k) - \boldsymbol{\xi}(k)||^2 \qquad (11)$$

where k_f is the length (in time samples) of the target trajectory, $\boldsymbol{\xi}(k)$ is the actual target position at time k, $M(k)$ is any of the global estimates $M_i(k)$ (which, thanks to the consensus protocols, are identical at any given time) at time k.

We now evaluate the performance of the CDTT algorithm in different setups of the simulation scenario. Table I lists the values obtained for these parameters in different simulations. As can be noted, a comparable improvement in the tracking performance can be achieved either by increasing the number of sensors, or by increasing the coverage ratio (*i.e.,* increasing the sensing radii). Moreover, the values of φ confirm that only a small number of sensors senses the target at a given time.

Two cases of the aforementioned simulation campaign are depicted in Fig. 2. In particular, two trajectories estimated by CDTT for a network of 25 agents are considered. In these pictures, in order to show the improvement in CDTT trajectory against the real one, two values of coverage percentage are considered. It can be noticed that when the target is not sensed by any of the agents the algorithm does not stop, the prediction follows the linear model defined in Section IV.B, and when the target comes back into any of the sensing ranges of at least one sensor, the prediction suddenly starts again to follow the target trajectory.

TABLE I

VARIABILITY OF PARAMETERS ρ, φ AND α IN DIFFERENT SIMULATION CASES.

n	ρ	φ	α
25	50%	0.78%	10.39
	75%	1.40 %	2.33
	100%	19.95%	0.08
50	50%	1.24%	6.01
	75%	1.97%	2.25
	100%	12.54%	0.08
75	50%	2.43 %	0.89
	75%	4.77%	0.16
	100%	6.25%	0.06

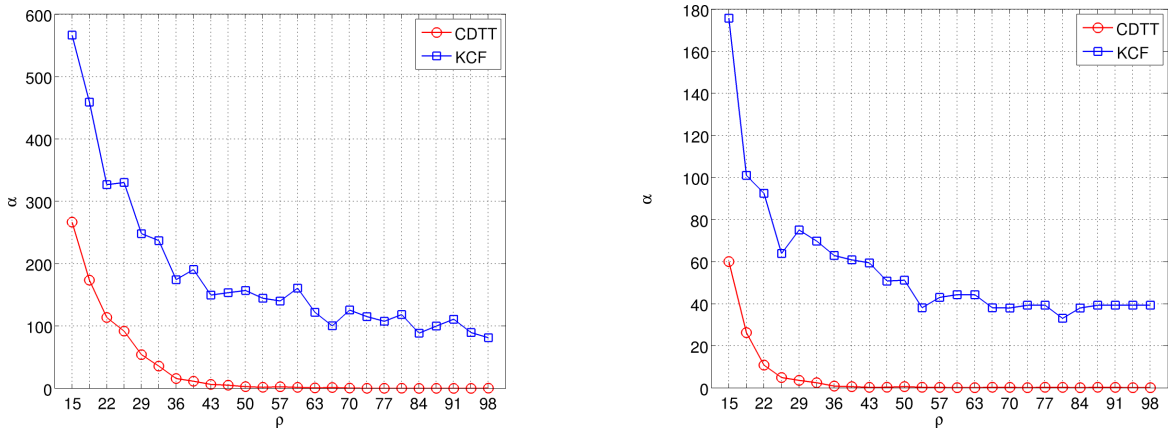

Fig. 3. Comparison between CDTT and KCF: Values of α as function of the parameter ρ for a network of $n = 25$ agents (a) and $n = 50$ agents (b). Each point is computed by averaging 50 random trajectories.

A. Comparison with the KCF algorithm

In order to evaluate the tracking performance of our CDTT algorithm we compare it to the hybrid architecture of the Kalman Consensus Filter with message passing (KCF) as described in [6]. This method is based on distributed *microfilters* with message-passing between agents and the use of a high-level fusion center for aggregating the estimates of a suitably selected set of agents, specifically, those who show the best agreement in the trajectory estimate (this is done by evaluating the covariance matrix of the local estimates). As detailed in [6], we set the following KCF parameters: $R_i = k_v^2 I_2$, $Q = k_w^2 I_4$, and $P_0 = 10 k_w^2 I_4$, where $k_v = 3$ and $k_w = \sigma_0 = 5$.

The comparison between CDTT and KCF is performed through several Monte Carlo simulations based on 50 repeated random trajectories for different values of ρ. More specifically, the average value of α over the test trajectories has been computed for each value of ρ and compared with the correspondent performance parameter obtained through the KCF algorithm trajectories. The results of this comparison are depicted in Fig. 3. Our algorithm clearly outperforms the KCF algorithm. We performed many other simulations with different values of the simulation parameters, which are not reported in this paper due to space constraints. Moreover, our algorithm is less dependent on the number of the sensing agents, as it selects only the best estimate at each iteration. On the other hand, to achieve good performance, KCF algorithm needs a certain number of sensing agents to achieve a satisfactory performance after the application of the sensor fusion phase. Conversely, our algorithm imposes some limitations on the sampling time of the tracking. In fact, the sampling time ϵ must be chosen such that the consensus phase ends before a new iteration of the tracking algorithm. Nevertheless, this limitation can be mitigated by estimating the network diameter at the time of the sensor network setup (*i.e.,* when checking the network connectivity). This assumption is reasonable in practical applications as long as the network topology is fixed in time, as in our case.

VI. Conclusion

In this paper we have addressed the problem of distributed target tracking by a network of heterogeneous sensory agents. The approach relies on an iterative strategy based on the execution of a perception phase, which makes use of a Kalman filter, followed by a max-consensus phase, which allows all the agents to agree on the best estimate of the target position through the whole network. The novel aspect of our approach is the full distribution of the algorithm. Moreover, the performance is good despite the fact that a few agents can actually sense the target at each iteration, and the environment is not fully covered by the sensor ranges.

References

[1] A. Ghaffarkhah and Y. Mostofi, "Communication-aware target tracking using navigation functions - centralized case," in *Proc. Int. Conf. on Robot Communication and Coordination*, Odense, Denmark, April 2009.

[2] B. S. Y. Rao, H. F. Durrant-Whyte, and J. A. Sheen, "A fully decentralized multi-sensor system for tracking and surveillance," *Int. Journal of Robotics Research*, vol. 12, no. 1, 1993.

[3] S. Oruc, J. Sijs, and P. van den Bosch, "Optimal decentralized kalman filter," in *Proc. Mediterranean Conf. on Control and Automation*, Thessaloniki, Greece, June 2009.

[4] C.-L. Wang and D.-S. Wu, "Decentralized target tracking based on a weighted extended kalman filter for wireless sensor networks," in *Proc. IEEE Global Telecommunication Conf.*, New Orleans, USA, Dec. 2008.

[5] R. Olfati-Saber, "Distributed kalman filtering for sensor networks," in *Proc. IEEE Conf. on Decision and Control*, New Orleans, USA, Dec. 2007.

[6] R. Olfati-Saber and N. F. Sandell, "Distributed tracking in sensor networks with limited sensing range," in *Proc. American Control Conf.*, Seattle, USA, June 2008.

[7] L. Lazos, R. Poovendran, and J. A. Ritcey, "Analytic evaluation of target detection in heterogeneous wireless sensor networks," *ACM Trans. on Sensor Networks*, vol. 5, no. 2, March 2009.

[8] B. M. Nejad, S. A. Attia, and J. Raisch, "Max-consensus in a max-plus algebraic setting: The case of fixed communication topologies," in *Int. Symposium on Information, Communication and Automation Technologies*, 2009, pp. 1–7.

[9] C. Godsil and G. Royle, *Algebraic Graph Theory*, ser. Graduate Texts in Mathematics. Springer, 2001.

[10] F. Zhao and L. Guibas, *Wireless Sensor Networks: An Information Processing Approach*. Morgan Kaufmann, 2004.

[11] Q. Li and D. Rus, "Global clock synchronization in sensor networks," *IEEE Trans. on Computers*, vol. 55, 2006.