J Intell Robot Syst (2017) 87:471–485
DOI 10.1007/s10846-017-0499-4

Collaborative Multi-MSA Multi-Target Tracking and Surveillance: a Divide & Conquer Method Using Region Allocation Trees

Emrah Adamey · Abdullah Ersan Oğuz · Ümit Özgüner

Received: 25 September 2015 / Accepted: 24 January 2017 / Published online: 4 May 2017

Abstract This paper presents a concurrent region decomposition and allocation algorithm that solves the multi-MSA coordination problem within the context of multi-target tracking and surveillance missions. Our collaboration approach achieves favorable computational characteristics, compared to its alternatives, by taking advantage of a data structure we named region allocation tree and the recursive processing strategy it allows. The region allocation tree data structure identifies the candidate regions, organizes information pertaining to tracking uncertainties and mobile sensor agent assignments, and allows for region decomposition and allocation simultaneously in a single depth-first sweep. Our collaboration approach, here, is used in conjunction with a Bayesian tracking algorithm–as the decision making is carried out in the belief space. Our contributions are also located within the wider discourse on multi-robot coordination. The simulation results demonstrate the effectiveness of our multi-MSA coordination approach.

Keywords Mobile sensor agents · Multi-robot systems · Task allocation · Region allocation · Target tracking · Area surveillance · Decision making · Bayesian tracking · Region allocation tree · Multi-MSA collaboration · Multi-UAV collaboration · Active information gathering

E. Adamey (✉) · Ü. Özgüner
Center of Intelligent Transportation Research (CITR)
Center of Automotive Research (CAR), The Ohio State University, Columbus, OH, USA
e-mail: adamey.1@osu.edu

Ü. Özgüner
e-mail: ozguner.1@osu.edu

A. E. Oğuz
Turkish Air Force Academy, Bakirköy/Istanbul, Turkey
e-mail: ae.oguz@hho.edu.tr

1 Introduction

Mobile sensor agents (i.e. MSAs) are a subspecies of cyber-physical systems, and are characterized by their limited sensing, mobility, and communication capabilities. Such systems include unmanned aerial, ground, and underwater vehicles [1–4]. They're often employed, as multi-MSA teams, in active information gathering missions–such as tracking, surveillance, coverage, and exploration [5]. In such missions, the effective coordination of the agents becomes particularly important. In order to render the solutions computationally feasible, the multi-MSA coordination problem is often divided into two subproblems: (1) task decomposition and (2) task allocation. In task decomposition, the objective is to decompose the overall mission into separate tasks, each of which contribute to the achievement of the overall mission. Here, the frontier-based approaches, and their variants, in

multi-robot exploration provide a paradigmatic example [6–8]. In task allocation, the objective is to assign these tasks to individual agents. Here, the literature on multi-robot task allocation (i.e. MRTA) is primarily informative [9–12]. The market-based multi-robot task allocation approaches, with their efficient auctioning and bidding mechanisms, are proved capable in a variety application domains [13–17].

In this paper, we focus on the multi-MSA multi-target tracking and surveillance problem [18–23]. Figure 1 illustrates the problem, where a number of moving targets are situated in a bounded environment, and a team of mobile sensor agents, with limited sensor footprints, are employed in an active information gathering mission to effectively track the targets. The number of mobile sensor agents are assumed to be smaller than the number of targets–so as to prevent the trivial solution of dedicating individual agents to chasing specific targets. The mission success is evaluated against total team entropy–an information-theoretic performance criterion we developed in our previous works [22, 23]. A particularly important issue in the multi-target tracking and surveillance problem is the choice pertaining to the mode of representation of the belief spaces of the mobile sensor agents–as the decision making procedure executes in the belief space. Here, we use a nonlinear Bayesian tracking method that leverages grid-based state discretization–a tracking scheme widely used in the literature [24, 25]. However, our coordination approach does not rely on the specifics of this filtering method. Our choice for the grid-based representation is rather preferred for its generality–and also for its suitability for being over-imposed on alternative modes of representation.

Here, we propose a multi-MSA coordination approach that solves the task decomposition and the task allocation subproblems concurrently in a computationally efficient manner–which puts our approach in a comparatively advantageous position over its alternatives. Our coordination approach relies on the use of a region allocation tree for simultaneously divides the environment into regions of interest based on the probability distributions of the target tracks and allocates the mobile sensor agents to these regions of interest based on their proximities. The region allocation tree is in the form of a binary search tree

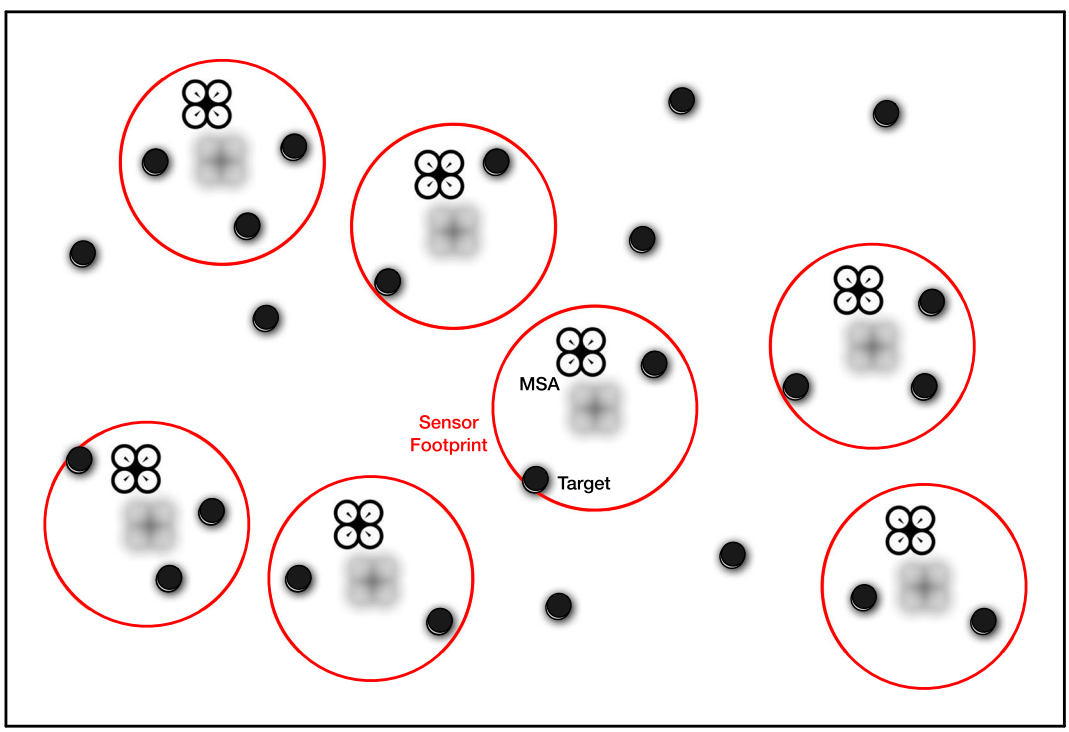

Fig. 1 Multi-MSA multi-target tracking problem: an overall description

[26, 27] and therefore allows for a divide-and-conquer approach in achieving the aforementioned functions in a single sweep. The computational complexity of the resulting multi-MSA coordination algorithm scales linearly with the number of agents and targets and logarithmically with the candidate regions encoded in the region allocation tree. We tested our coordination approach in a number of simulation runs using MATLAB. The results demonstrate the effectiveness and efficiency of our approach.

This paper is divided into multiple parts. In Section 2, we present the workings of Bayesian multi-target tracking. It is intended largely for providing an insight into the mode of belief representation our coordination algorithm operates on and an understanding of the Bayesian tracking framework. In Section 3, we present our multi-MSA coordination approach. Here, the main contribution of our paper is explained in detail. The focus is on the region allocation tree data structure and the coordination algorithm that operates on it. In Section 4, we explain our MATLAB-based simulation setup, define our evaluation scheme, and present our simulation results. Further remarks are also provided in this section. And, finally, in Section 5, we conclude our work and provide directions for future research.

2 Bayesian Multi-Target Tracking

The Bayesian filtering framework is a dominant paradigm in the multi-sensor data fusion literature [28, 29]. It is widely used in robotic applications [30–34]. Its specific forms, such as Kalman filters [35, 36] and particle filters [37, 38], are extensively used in a broad range of applications that are concerned with wringing information coming from separate sources into coherent and reliable estimates of hidden states. Here, we're interested in Bayesian filtering within the context of the multi-target tracking problem [24, 25]. Specifically, we're focusing on the discrete-state implementations of Bayesian multi-target tracking [18, 21, 24]. The advantages of discrete Bayes filter include the ease of implementation, the adaptability of the models employed, and the usefulness of the output for decision making procedures.

$$bel\left(q_k^1, q_k^2, \ldots, q_k^m\right) = \prod_{j=1}^{m} bel\left(q_k^j\right) \tag{1}$$

In Bayesian multi-target tracking, the first design choice rests in deciding whether to estimate target states separately or in a joint state space. Here, in our approach, by assuming that target motions are mutually independent, we maintain target tracks separately. This assumption leads to a more conservative way of dealing with uncertainties, but simplifies the implementation and reduces the computational burden that arises in joint-state estimation. The mutual independence assumption is described in Eq. 1, where q_k^j represents the state of target j at the time-step k and *bel* refers to the belief function. Under this formulation, the multi-target tracking problem is decomposed into separate single-target tracking problems.

The second design choice we face pertains to the state space of target tracks. For our purpose, estimating the latitudinal and longitudinal positions of the targets is sufficient–although a more comprehensive state space is also compliant with our decision making procedure. (Note that this choice is driven by the requirements of the decision making approach which relies on the tracking result.) We also need to identify the boundaries of the state space and the granularity of its discretization. Here, as we assume a bounded region within which targets are tracked, we select the boundaries of the state space as the boundaries of the region $\mathcal{R}$. The choice of discretization granularity is a trade-off between estimation accuracy and computational complexity. Here, the state-space $\mathcal{S}$ is divided into discrete states of $s \in \mathcal{S}$. The Bayes filter tracks the target by maintaining probabilities for the target being at a specific discrete state (i.e. cell), as $p(s_k|z_{1:k})$, by fusing all the measurement data

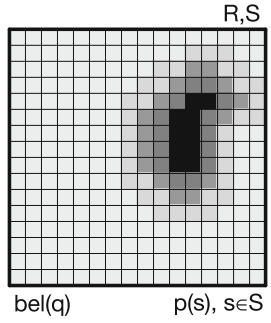

Fig. 2 The track for target q, as $bel(q)$, is represented using a grid structure where $\mathcal{R}$ and $\mathcal{S}$ determine the boundary and granularity of the grid structure, and $p(s)$ represent the probability of target being at the cell $s \in \mathcal{S}$. Darker regions correspond to the high-probability cells

available $z_{1:k}$. Hence, target tracks are maintained in grid-based representation, as depicted in Fig. 2. As the target remains within the region, the probabilities of each cell sum up to 1 for each target as in Eq. 2. (Here, we drop maintaining the superscripts for target identifiers, j, for notational simplicity.)

$$\sum_{s \in S} p\left(s_k | z_{1:k}\right) = 1 \tag{2}$$

The Bayes filter has two consecutive steps of prediction and correction. In the prediction step (i.e. time update), as shown in Eq. 3, target tracks are updated by using a Markovian motion model, $p\left(s_k | \tilde{s}_{k-1}\right)$, defined as from a given cell $\tilde{s}_{k-1}$ from the previous time-step to a given cell s_k at the current time-step. (As this estimate is generated prior to using the measurement at the time step, it is also called *a priori* estimate.) In most applications, and here as well, a simple heat dissipation model is sufficient as the motion model, and it can be represented as a convolution operation over the grid structure. More complex models involving hospitability and inclination maps are also used in the literature [18, 21]. Here, note that the Bayes filter requires a prior distribution for the first time step, before the first measurement is acquired. In our application, the prior distribution is selected as a uniform distribution over the entire region–which corresponds to the maximum-uncertainty condition.

$$p\left(s_k | z_{1:k-1}\right) = \sum_{\tilde{s} \in S} p\left(s_k | \tilde{s}_{k-1}\right) p\left(\tilde{s}_{k-1} | z_{1:k-1}\right) \tag{3}$$

$$p\left(s_k | z_{1:k}\right) = \eta \; p\left(z_k | s_k\right) p\left(s_k | z_{1:k-1}\right) \tag{4}$$

In the correction step (i.e. measurement update), as shown in Eq. 4, target tracks are updated by using measurements acquired at that time step and a measurement likelihood model $p\left(z_k | s_k\right)$. The variable η refers to the normalization constant, ensuring that the cell probabilities sum up to one for each target posterior. (The resulting estimate is called *a posteriori* estimate.) Here, as shown in Eq. 5, we're using a likelihood model which incorporates both positive and negative information. Positive information refers to the case where a target is observed at the given time step, which is informative about where the target is. This case is modeled using a multivariate normal distribution $\mathcal{N}\left(h\left(x_k, s_k\right), R_k\right)\left(z_k\right)$. Here, $h\left(x_k, s_k\right)$ refers to the measurement model for target detection, where x_k refers to the observing agent, s_k refers to a given grid cell, z_k an observed measurement, and R_k the measurement uncertainty. Negative information, on the other hand, refers to the case where a target is not observed, which is informative about where the target is not. We model this case by evaluating detection probabilities for each cell, $\alpha\left(x_k, s_k\right)$, given the information about the sensor footprint of the MSA. As such, by incorporating both positive and negative information obtained by several MSAs regarding each target, Bayesian filtering provides an effective multi-target tracking algorithm.

$$p(z_k | s_k) = \begin{cases} \mathcal{N}\left(h\left(x_k, s_k\right), R_k\right)\left(z_k\right) & \text{if target is detected} \\ 1 - \alpha\left(x_k, s_k\right) & \text{if target is not detected} \end{cases} \tag{5}$$

In this framework, any knowledge regarding the hospitality of environment can be encoded in a map-based prior $p\left(s_k | m\right)$. This prior can then be fused with the tracking posterior $p\left(s_k | z_{1:k}\right)$ according to the Bayesian fusion equation given in Eq. 6. Since the map-independent prior on target position $p\left(s_k\right)$ is uniform, this equation simplifies to Eq. 7 where γ is the normalizing constant.

$$p\left(s_k | z_{1:k}, m\right) = \frac{p\left(s_k | z_{1:k}\right) p\left(s_k | m\right)}{p\left(s_k\right)} \tag{6}$$

$$= \gamma p\left(s_k | z_{1:k}\right) p\left(s_k | m\right) \tag{7}$$

The Bayesian tracking approach described above is summarized in Algorithm 1. The mutual-independence assumptions of target motions and of target observations result in the nested algorithmic structure, and thereby significantly reducing the computational complexity. As the complexity of processing of likelihood models scales linearly with the number of grid cells defined in the discrete state space, the overall computational complexity of this algorithm can be stated as $\mathcal{O}(NMC)$ where N, M, and C represent the number of agents, the number of targets, and the number of grid cells correspondingly.

Note that our region allocation tree based collaboration scheme, which is described in detail in Section 3, does not rely on the specifics of the Bayesian tracking algorithm proposed in this section. One can use other variations of discrete-state Bayesian multi-target tracking [24, 25]. Moreover, one can use other nonlinear filtering approaches, such as particle filters [38], so long as a grid structure is overlaid upon the resulting state estimate. The main concern

Algorithm 1 Bayesian tracking algorithm

```
1: procedure BAYESIAN–TRACKING
2:     for all targets do
3:         Use motion model to update target track
4:         for all agents do
5:             if the target is detected by the agent then
6:                 Use likelihood model for the positive information case to update track
7:             else
8:                 Use likelihood model for the negative information case to update track
9:             end if
10:        end for
11:        Refine the target posterior using the map dependent prior
12:        Normalize the probabilities of the grid cells so that they sum up to 1
13:    end for
14: end procedure
```

for Section 2 is to demonstrate the workings of the mode of representation chosen for target tracking upon which our region allocation approach depends.

3 Multi-MSA Coordination

As we explained in Section 1, our work can be understood as a subspecies of multi-robot coordination within the context of active information gathering missions. In the multi-robot coordination literature, there are three important schemes of classification [9–11]. Here, we first frame our work within the context of the wider discourse on multi-robot coordination, by using these classification schemes, and thereby explicitly define our problem and our proposed solution, together with their virtues, using the conceptual framework provided by the literature. And, after framing our work thusly, we proceed by explaining the details of our region allocation tree data structure and our concurrent region decomposition and allocation algorithm that operates on it.

The multi-robot collaboration taxonomy introduced by Farinelli et al. in [9], depicted in Fig. 3, uses a three-level classification: (1) knowledge level, (2) coordination level, and (3) communication level. At the first level, since the mobile sensor agents are actively sharing information with each other, including target measurements and region allocations, our approach falls within the aware category. Thus, following to the second level of classification, as our approach relies on a coordination protocol, namely the

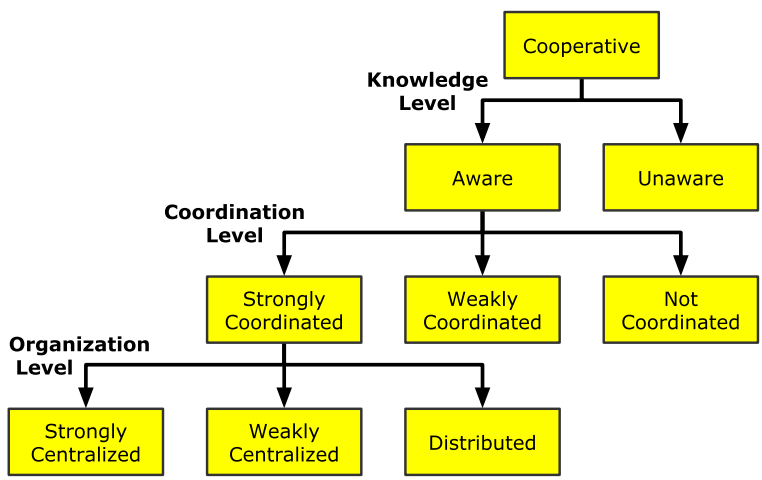

Fig. 3 Multi-robot systems taxonomy proposed by Farinelli et al. in [9]

allocation of regions to mobile sensor agents, it is classified as a strongly coordinated system. Finally, at the last level of classification, as region allocation decisions are made by a leader agent and as the leader agent can be any agent in the mobile sensor team, our collaboration approach is classified as weakly centralized. Any agent, at any time, can take the role of the leader agent. And, although the leader agent is responsible for the adaptive re-allocation of regions to the multi-agent team, each agent can carry out its local decisions without needing supervision from the leader agent. Note that the collaborative multi-MSA multi-target tracking and surveillance problem can be solved with different strategies that may fall within a different category–therefore the weakly-centralized classification is specific to the collaboration approach proposed in this paper.

A second important scheme of classification of multi-robot systems is the multi-robot task allocation (i.e. MRTA) taxonomy proposed by Gerkey and Mataric in [10]. Here, as shown in Fig. 4, MRTA systems are classified based on where they fall in a three-dimensional solution space. The axes defined are: (1) single-task (ST) versus multi-task (MT) robots, (2) single-robot (SR) vs multi-robot (MR) tasks, and (3) instantaneous (IA) vs time-extended (TA) assignment. According to this framework, our approach can be classified as an instance of ST-SR-IA class. Note that this classification focuses on the task allocation step, and not on the task decomposition step. Note also that a primary feature of our approach is the concurrent solution of task allocation and task decomposition problems.

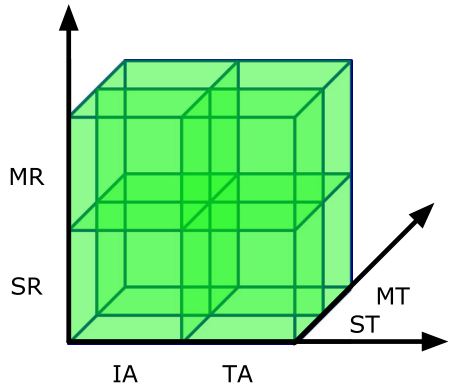

Fig. 4 Multi-robot task allocation (MRTA) taxonomy of Gerkey and Mataric [10]

In the multi-robot coordination taxonomy introduced by Korsah et al. [11], the focus is shifted towards the task decomposition. Here, the authors correctly diagnosed the main shortcomings of the MRTA framework, namely its restrictive assumption of task independence and its failure to deal with interrelated utilities and constraints between tasks. In their own framework, iTax, they used the terminology of task decomposability, multiple decomposability, elemental/atomic tasks, decomposable simple tasks, compound tasks, and complex tasks. Without going into much detail, we point out that, in our approach, we formulate the overall problem as a complex task and we actively exploit the multiple decomposability property in our method of concurrent task decomposition and allocation. Their taxonomy addresses the problem of interrelated task utilities and constraints by proposing four distinct classes of dependencies: (1) no dependencies (ND), (2) in-schedule dependencies (ID), (3) cross-schedule dependencies (XD), and (4) complex dependencies (CD). Here, we note that our approach frames the problem in a way that there are complex dependencies (CD) between tasks–and in particular task utilities. This can be understood using a simple example: a target with unknown location, i.e. with near-maximal entropy, has a probability distribution that is spread throughout the bounded environment, contributing to the utilities of each candidate region. This effectively influences the manner in which the bounded region is divided into separate regions (i.e. task decomposition) and the manner in which those regions are assigned to specific agents (i.e. task allocation). Moreover, when an agent finds the target in question in its assigned region, the utility landscape changes and that influences the entire task decomposition and allocation process in the next time-step. Hence, we identify our approach as dealing with complex tasks which present a particular challenge as they decouple the task decomposition and task allocation problems. Therefore, we conclude that our collaboration approach is an instance of a weakly-centralized approach using an ST-SR-IA method in solving a complex task involving complex dependencies, according to the three taxonomies mentioned [9–11].

In our previous works [22, 23], we proposed multi-MSA coordination mechanisms for multi-target tracking and surveillance missions. These coordination approaches were based on a region allocation algorithm that used two consecutive steps: (1) the

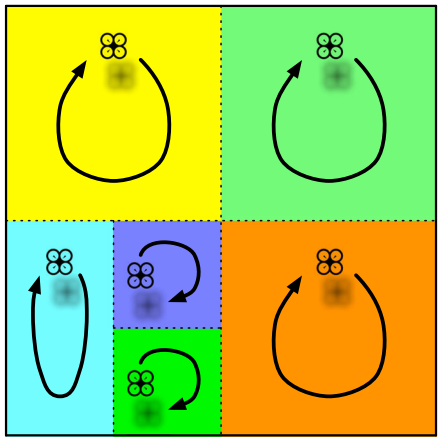

Fig. 5 Region allocation based multi-MSA coordination for multi-target tracking and surveillance. Each different color represent a different region assigned to a specific agent

clustering of particle sets describing target tracks into separate regions, and (2) the assignment of individual MSAs to the these regions/clusters. Each MSA, then, plans its trajectory in its assigned region so as to minimize the uncertainty contained in that region. As new target measurements are obtained and the underlying uncertainty characteristics of the environment changes, the regions and their allocations are updated at the next region allocation cycle. We demonstrated that this region-based coordination strategy yields satisfying tracking performance–as measured against the criterion of total team entropy. Its computational complexity is described as $\mathcal{O}\left(NMk + N^3\right)$, where N, M, and k represent the number of MSAs, the number of targets, and the maximum number of steps allowed in clustering respectively. Here, in this paper, we present a coordination method that uses a similar region allocation strategy. Our new approach accomplishes effective coordination with significantly better computational characteristic–which is described as $O(NM \log q)$, where q represents the number of candidate regions defined *a priori*. This favorable result is achieved by taking advantage of a data structure we coined as region allocation tree–and the recursive node-processing procedure it allows. Figure 5 illustrates the main idea behind region-based coordination approaches.

Figure 6 illustrates the region allocation tree data structure. It is devised similar to a binary tree data structure [26, 27]. Here, each node in the region allocation tree defines a candidate region to be considered in region allocation. The top node describes the entire bounded environment to be surveilled. At each level, a given region is divided into two sub-regions, as its children nodes. The leaf nodes determine the granularity of the decomposition of the overall environment into separate regions. Here, the bounded environment is defined as a rectangular shape, and the region allocation tree is defined as a balanced tree where

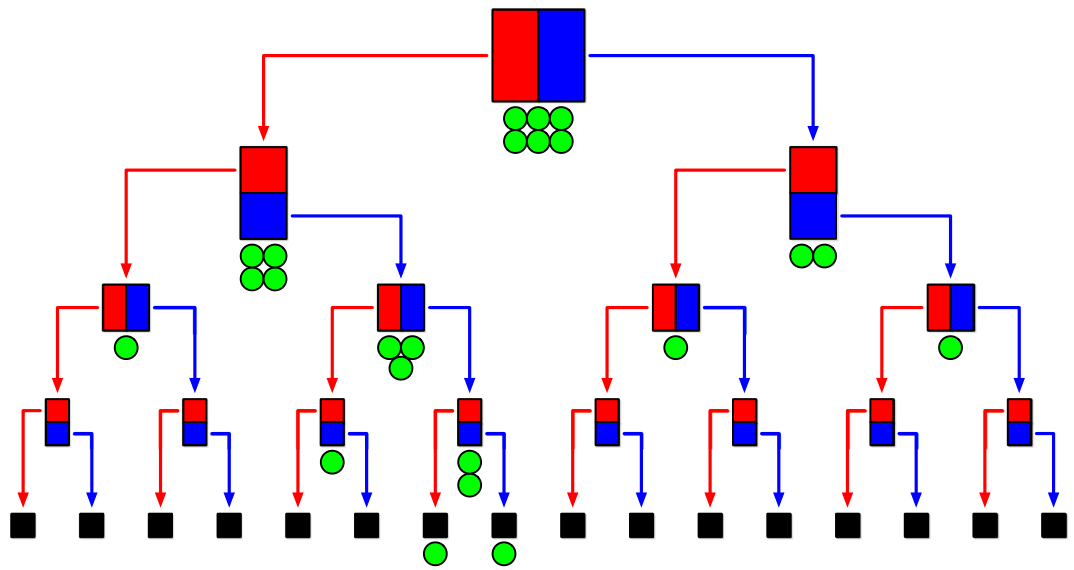

Fig. 6 Region allocation tree, with 4 levels of depth. The top node represents the entire region, divided into two regions at each level. All the MSAs are assigned to the top node (i.e. entire region) and region assignments are determined by a divide-and-conquer approach starting from the top node

regions are divided into rectangular shapes at each level. This particular decomposition strategy is not a necessary scheme, although computationally efficient, and can be replaced with any arbitrary decomposition scheme so long as a tree structure is maintained as a constraint between candidate regions.

$$H\left(r_k\right) = \sum_{s_k \in r_k} H\left(s_k\right) \tag{8}$$

$$H\left(s_k\right) = \sum_{j=1}^{M} H\left(s_k^j\right) \tag{9}$$

$$H\left(s_k^j\right) = -p\left(s_k^j | z_{1:K}\right) \log p\left(s_k^j | z_{1:K}\right) \tag{10}$$

In the region allocation tree, each node (i.e. each candidate region) is assigned a utility value and a set of agents allocated to it. The utility values represent the amount of uncertainty contained in the corresponding candidate regions, and the uncertainties are understood as tracking entropies–an information-theoretic measure of uncertainty. The tracking entropy of a given candidate region r_k, $H\left(r_k\right)$, is defined as the summation of the entropies of each grid cell s_k that resides in it, $H\left(s_k\right)$, as given in Eq. 8. The entropy of a given grid cell is, then, computed as the summation of the entropies of each target track for that grid cell, $H\left(s_k^j\right)$, as given in Eq. 9. The entropy of a given grid cell for a given target track is then computed according to the standard entropy equation given by Eq. 10. A computationally efficient method to compute utilities is given by Eqs. 11 and 12, where the entropies of the leaf nodes r_k^0 are computed first, and then the entropy values of the non-leaf nodes r_k^+ are computed as the summations of the entropies of their children nodes r_k^L and r_k^R by moving upwards in the tree. Algorithm 2 summarizes the recursive computation of the entropies of all the nodes in the region allocation tree in a bottom-up fashion. The complexity of the resulting agglomerative entropy calculation procedure scales linearly with the number of cells in the grid map and also linearly with the number of nodes in the region allocation tree.

$$H\left(r_k^0\right) = \sum_{s_k \in r_k} H\left(s_k\right) \tag{11}$$

$$H\left(r_k^+\right) = H\left(r_k^L\right) + H\left(r_k^R\right) \tag{12}$$

After the utility values are computed for all the candidate regions, the region allocations are initialized thus: (1) all the MSAs are assigned to the top node, and (2) all other nodes are cleared of their assigned MSAs. Then the region allocation tree is processed, node-by-node, starting from the top node in a recursive fashion. At each node, the MSAs assigned to that node are further assigned to its children nodes according to their utilities and their distances from the MSAs under consideration. This is achieved in two consecutive steps. First, the number of MSAs to be allocated for each child node is decided by comparing the entropies of the two children nodes, as given in Eqs. 13 and 14. Here, n_k^p, n_k^L, and n_k^R refer to the number of MSAs assigned to the parent node, to the left-child node, and the right-child node respectively. The function [.] refers to rounding to the nearest integer number.

$$n_k^L = \left[\frac{H\left(r_k^L\right)}{H\left(r_k^R\right)} n_k^p\right] \tag{13}$$

$$n_k^R = n_k^p - n_k^L \tag{14}$$

After the numbers of MSAs for each child node are decided, the second step is to sort the MSAs assigned to the parent node according to their distances to the

Algorithm 2 Entropy calculation for the region allocation tree

```
1: procedure AGGLOMERATIVE ENTROPY CALCULATION
2:     update the entropies of the leaf nodes using Equation 10
3:     while root node is not reached do
4:         move up a level in the region allocation tree
5:         update the entropies of all the nodes in this level using Equation 12
6:     end while
7: end procedure
```

Algorithm 3 Region allocation tree

```
1: procedure REGION–DECOMPOSITION–AND–ALLOCATION
2:     update the utilities of the region allocation tree
3:     initialize the MSA assignments of the nodes
4:     process the top node in the region allocation tree
5: end procedure
6: procedure NODE–PROCESSING
7:     if it is not a leaf node then
8:         if it has more than one MSA allocated then
9:             assign MSAs to children nodes
10:            process the left-child node of the current node
11:            process the right-child node of the current node
12:        end if
13:    end if
14: end procedure
```

left-child node, and then the first n_k^L agents in the sorted list are assigned to the left-child while the remaining n_k^R agents are assigned to the right-child node. After this is accomplished the children nodes are executed, moving downwards in the region allocation tree–until a leaf node or a non-leaf node with only one agent assigned is reached. This procedure results in a breath-first search that concurrently accomplishes region decomposition and region allocation.

Algorithm 3 summarizes our concurrent region decomposition and allocation algorithm operating on the region allocation tree. The first procedure, *region decomposition and allocation*, describes the main algorithm for processing the region allocation tree at any given decision making cycle. The second procedure, *node processing*, is called by the main procedure and by itself recursively as the region allocation tree is processed in a breath-first fashion. An illustration of the algorithms working is provided in Fig. 6 together with its resulting region-agent pairs in Fig. 5. The computational complexity of this algorithm is $\mathcal{O}(NM \log q)$, where N, M, and q refer to the number of agents, the number of targets, and the number of candidate regions respectively. The term $\log q$ emerges from the breath-first processing of the tree, as it describes the depth of the tree. The term N emerges from the sorting of the MSAs at each level–where linear complexity can be accomplished by using the selection algorithm [26]. Finally, the term M emerges from the utility calculation step executed during the initialization of the region allocation tree. Note that this collaboration scheme results in a significantly more favorable computational characteristic when compared against our previous work [22, 23].

4 Simulation Results

We tested our collaboration scheme using MATLAB in a simulation setup where $N \in 6, 8, 10, 15, 20$ mobile sensor agents are given the mission of tracking $M \in 6, 8, 10$ moving targets in a 1200m $*$ 1200m bounded environment. The granularity of the grid cells is chosen as 5m$*$5m, which yields tracking grids of the size of $240 * 240$. A number of obstacles are randomly sampled in the region. The region allocation tree is chosen to have $l = 4$ levels of depth, and therefore has $q = 2^{l+1} - 1 = 31$ candidate regions where $2^l = 16$ of them are leaf nodes. Hence, each leaf node covers a region of $60 * 60$ cells. The maximum speeds of targets and agents are chosen as 5 and 20 m/s respectively. The detection range of the mobile sensor agents are chosen as 100 meters. Targets start at random locations and then they move towards randomly selected goal locations, while avoiding obstacles, and then randomly choose new goal locations. MSAs also start at random locations and quickly adjust their region allocations to their belief space. Figure 7 demonstrates a randomly sampled simulated world, where the obstacles are shown in grey and the trajectories for 10 moving targets are shown in different colors.

Figure 8 illustrate the map, the targets, the agents, their sensor footprints, and allocated regions in an

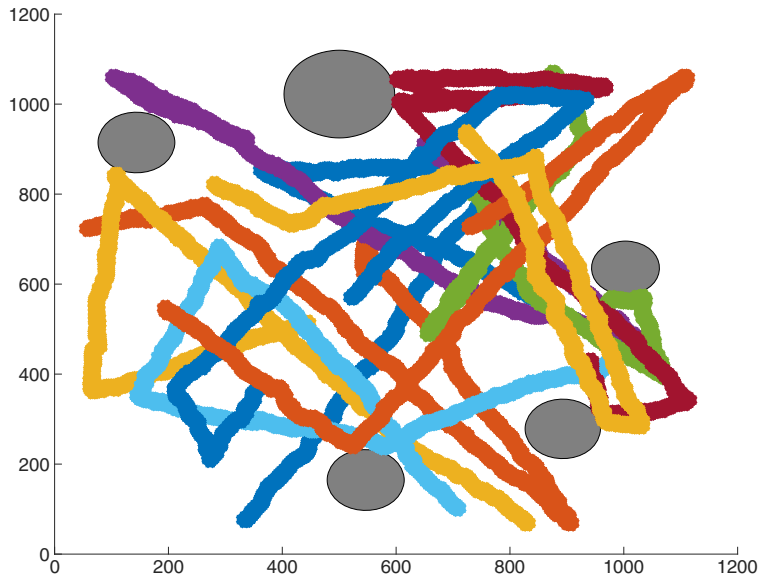

Fig. 7 The simulated world and the simulated target trajectories

early step of a 6-agent 10-target simulation setup. Here, colored dots represent the locations of the targets that are being simulated. The agents and their sensor footprints are illustrated as black dots and cyan circles around them respectively. The allocated regions are shown as rectangular decompositions of the entire map, and the centroids of each region, shown as black circles, are linked to the corresponding sensor agents to which they are assigned. Note that each agent is assigned to a nearby region, such that the collective allocations are cost-efficient in terms of agent-to-region distances.

Our grid-based tracking method fuses both positive and negative measurements from all agents and the map priors in target posteriors. Figure 9 demonstrates the posterior for an unobserved target for the timestep corresponding to the simulated world in Fig. 8. Here, the coloring is such that the black

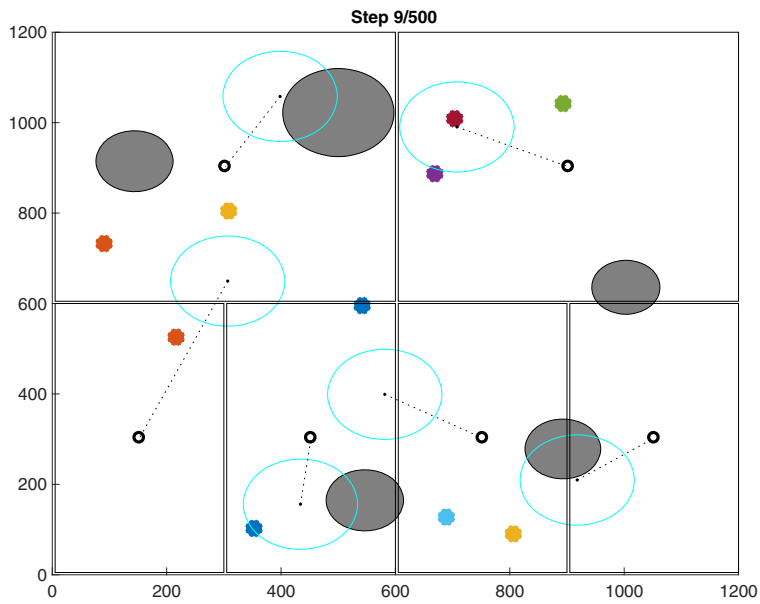

Fig. 8 The simulated world at a given timestep and the corresponding region assignments

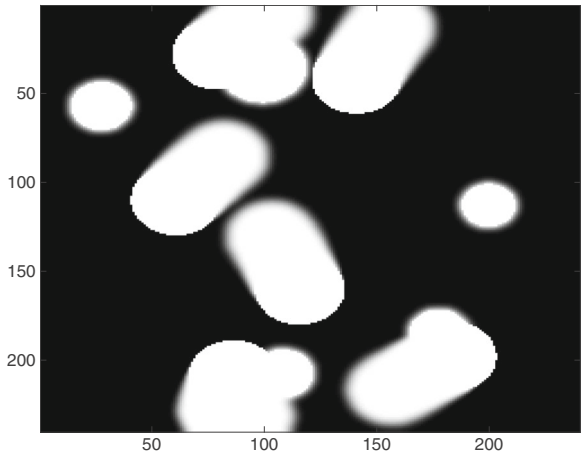

Fig. 9 The posterior for a target during the initial search phase

and white regions correspond to higher and lower probabilities correspondingly. Note that the areas corresponding to environmental obstacles have almost zero probability for the target being there, as encoded by the map-dependent priors. We also observe that the negative information from the recent trajectories of the agents also contribute to the refinement of the target posterior.

In our simulations, we observed that there are two separate phases in our multi-target tracking and surveillance approach for each target: (1) the search-and-find phase, and (2) the regular tracking phase. In the search-and-find phase, the probability distribution of the target is near uniform throughout the bounded environment, and therefore provide limited guidance for agent coordination. In this phase, until a measurement is obtained about the agent, the MSA team largely rely on negative information–i.e. information regarding where the target is not–to focus their search-and-find efforts guided by the region decomposition and allocation mechanism. Figure 9 illustrates this phase: the progression of the search-and-find phase guided by negative information. Once the target is located in the tracking grid, then the second phase takes over. When tracking a set of targets in the regular tracking phases, the mobile sensor agents switch their attentions from one subset of targets to another based on their entropy characteristics. As this process is more informed, i.e. the agent knows where to look, it proceeds smoothly and the agents do not loose track of the targets.

In Fig. 10, we see the tracking entropies of each target throughout the simulation. Here, we observe the two phases clearly. In the search-and-find phase, target entropies decrease slightly based on the contributions of the negative information acquired by the agents. When a target is first observed, its entropy decreases sharply as it is accurately located. After that, the regular tracking phase becomes active. In this phase, there are three factors in play: (1) entropy increase due to target motions, (2) entropy decrease due to negative information, and (3) entropy decrease due to positive information. Here, we observe that when the target is not observed the entropy increase due to target motion overwhelms the entropy decrease due to negative information. (There is actually a convergence point, to which the target entropies converge in negative information case–both in the search-and-find phase and in the regular tracking phase.) We also observe that the multi-MSA team switches between the targets in a way that is adaptive to their entropies.

Figure 11 models the dynamics of the changes in target tracking entropies. Since the targets are at unknown locations in the beginning, their tracking entropies start with near-maximum theoretical

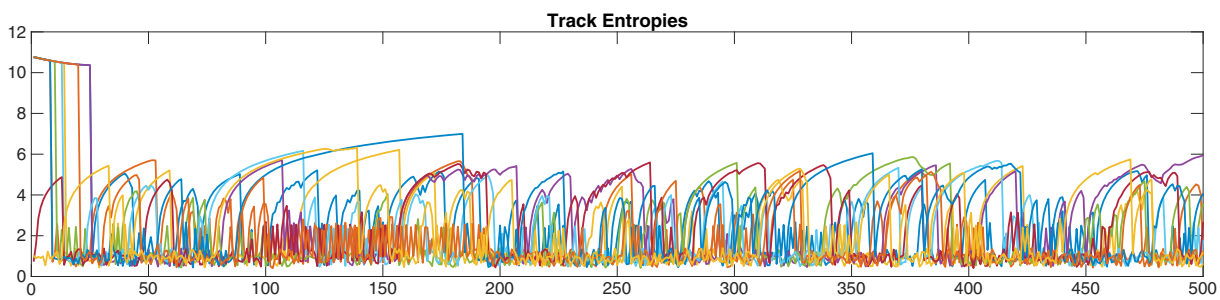

Fig. 10 Individual track entropies throughout the simulation, for $N = 6$ and $M = 10$

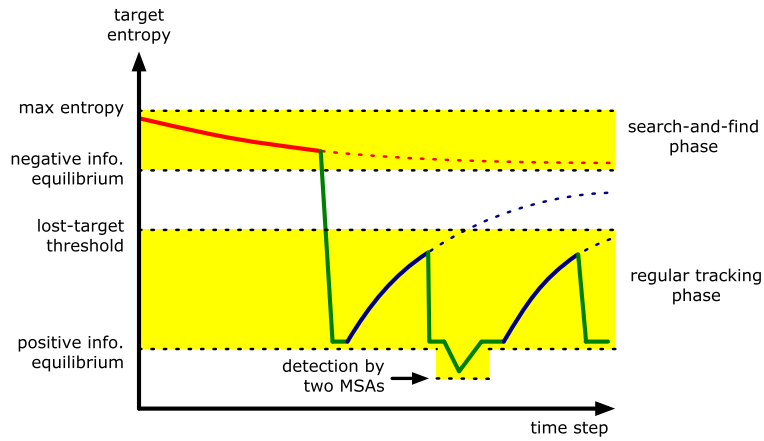

Fig. 11 The dynamics of target tracking entropy changes, for a 6-agent and 10-target scenario

value. Based on the availability of target detections/measurements, the tracking algorithm proceeds in either the positive information or the negative information case. Here, positive and negative information refers to the availability of information regarding where the target is and is not respectively. And, consequently, the tracking entropy of a given target approaches one of the two equilibria: the negative information equilibrium and the positive information equilibrium. Therefore, in the search-and-find phase, the target track starts from a near-max entropy and slowly converges toward the negative information equilibrium–as the search continues–and whence the target is detected, a sharp decrease in entropy is observed, reaching the positive information equilibrium. Once the target is found, then the regular tracking phase is observed where target tracks oscillate between moving towards the positive information equilibrium and the negative information equilibrium. This stems from the fact that the number of targets is greater than the number of agents, $M > N$, therefore the agents need to switch their attention from one target to another based on their tracking entropies. Here, during this regular tracking phase, by successful re-allocation of the MSAs over different regions, the target tracks are maintained within an entropy band that resides between the positive equilibrium and a threshold for declaring the target as lost. (Note

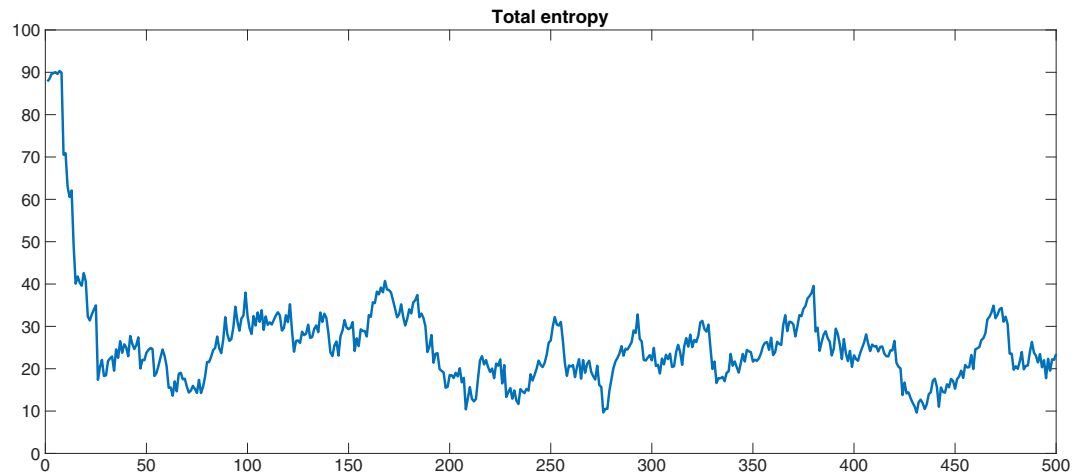

Fig. 12 The total team entropy throughout the simulation, for a 6-agent and 10-target scenario

Table 1 The average track entropy under different scenarios with different numbers of agents (N) and targets (M)

		M				
		6	8	10	15	20
N	6	2.4986	2.7783	2.8959	2.9879	3.3892
	8		2.1960	2.3257	2.7013	3.0096
	10			2.0987	2.2836	2.3440

that the positive information equilibrium can shift downwards, as seen in Figs. 10 and 11, if two or more MSAs detect/observe a given target at the same time.)

The total target tracking entropy, our central performance criterion, is shown in Fig. 12 for throughout the simulation. We observe that the multi-MSA team first succeeds in the search-and-find phase and significantly reduces the total entropy, and then succeeds in the regular tracking phase by keeping the total entropy within an acceptable level.

In Table 1, we provide the average track entropies under different simulation setups with different numbers of agents and targets. We simulated for cases where the number of targets are equal to or larger than the number of agents (i.e. $N <= M$). As seen from the results, the performance of our multi-agent multi-target tracking approach improves consistently as more agents are added, and decreases as more targets are introduced. Similarly, Table 2 illustrates the average of the worst track entropies for the same simulation scenarios. Here, the focus is on the least-known targets at any given time, and the results suggest a similar performance increase and decrease as more agents and targets are introduced respectively.

Table 2 The average worst track entropy under different scenarios with different numbers of agents (N) and targets (M)

		M				
		6	8	10	15	20
N	6	4.8084	5.7232	6.0133	6.1840	6.5210
	8		4.8210	5.3173	5.7339	6.1045
	10			4.9847	5.5630	5.8889

The results demonstrate the effectiveness of our region-based approach for collaborative multi-agent multi-target tracking.

5 Conclusion

In this paper, we presented a novel collaborative multi-MSA multi-target tracking and surveillance approach. It is developed in conjunction with the Bayesian tracking framework, as the decision making procedure relies on the belief space. Our approach is based on decomposing the entire bounded environment into separate regions, in light of the underlying uncertainty characteristics of the belief space, and then allocating these regions to individual mobile sensor agents to be explored. Our approach utilizes a data structure called region allocation tree, which encodes all the candidate regions, organizes them in accordance with their subset/superset relationships, and keeps track of their exploration utilities. Our coordination algorithm solves the region decomposition and the region allocation problems simultaneously in a single depth-first sweep of the region allocation tree. Therefore, it achieves favorable computational characteristics.

We tested our proposed multi-MSA collaboration approach in a MATLAB based simulation setup. We observed two phases for each target track: (1) the search-and-find phase, and (2) the regular tracking phase. We also observed two equilibrium points for the entropies of each target track: (1) the negative information equilibrium, and (2) the positive information equilibrium. We further investigated the dynamics behind the target tracking entropies based on these observations. We demonstrated that our approach succeeds in accurately finding the moving targets in the environment and then reliably maintaining the uncertainties corresponding to each target track below a certain threshold. The results demonstrate the effectiveness of our approach.

One limitation of our collaboration approach stems from the fact that candidate regions are defined *a priori*. We acknowledge that, although this reduces complexity by effectively restricting the search space for region decomposition, it can also lead to inefficiencies if the candidate regions are not chosen carefully enough. We identify the choice of candidate regions as a design choice that involves trade-offs between com-

plexity and effectiveness. Future directions for our work thus involves investigating the effects of and providing guidelines for candidate region selection.

References

1. Feron, E., Johnson, E.N.: Aerial Robotics. In: Springer Handbook of Robotics (2008)
2. Broggi, A., Zelinsky, A., Parent, M., Thorpe, C.E.: Intelligent Vehicles. In: Springer Handbook of Robotics (2008)
3. Ozguner, U., Acarman, T., Redmill, K.: Autonomous Ground Vehicles. Artech House (2011)
4. Antonelli, G., Fossen, T.I., Yoerger, D.R.: Underwater Robotics. In: Springer Handbook of Robotics (2008)
5. Parker, L.E.: Multiple Mobile Robot Systems. In: Springer Handbook of Robotics (2008)
6. Fox, D., KO, J., Konolige, K., Limketkai, B., Schulz, D., Stewart, B.: Distributed multirobot exploration and mapping. Proc. IEEE **94**(7), 1325–1339 (2006)
7. Burgard, W., Moors, M., Stachniss, C., Schneider, F.E.: Coordinated multi-robot exploration. IEEE Trans. Robot. **21**(3), 376–386 (2005)
8. Wurm, K.M., Stachniss, C., Burgard, W.: Coordinated multi-robot exploration using a segmentation of the environment. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS 2008, pp 1160–1165 (2008)
9. Farinelli, A., Iocchi, L., Daniele, N.: Multirobot systems: a classification focused on coordination. Systems, Man, and Cybernetics, Part B: IEEE Transactions on Cybernetics **34**(5), 2015–2028 (2004)
10. Gerkey, B.P., Matarić, M.J.: A formal analysis and taxonomy of task allocation in multi-robot systems. Int. J. Robot. Res. **23**(9), 939–954 (2004)
11. Ayorkor Korsah, G., Stentz, A., Bernardine Dias, M.: A comprehensive taxonomy for multi-robot task allocation. Int. J. Robot. Res. **32**(12), 1495–1512 (2013)
12. Sariel-Talay, S., Balch, T.R., Erdogan, N.: A generic framework for distributed multirobot cooperation. J. Intell. Robot. Syst. **63**(2), 323–358 (2011)
13. Bernardine Dias, M., Zlot, R., Kalra, N., Anthony, S.: Market-based multirobot coordination: A survey and analysis, Proceedings of the IEEE **94**(7), 1257–1270 (2006)
14. Gerkey, B.P., Matari, M.J.: Sold! auction methods for multirobot coordination. IEEE Trans. Robot. Autom. **18**(5), 758–768 (2002)
15. Zlot, R., Stentz, A.: Market-based multirobot coordination for complex tasks. Int. J. Robot. Res. **25**(1), 73–101 (2006)
16. Choi, H.-L., Brunet, L., How, J.P.: Consensus-based decentralized auctions for robust task allocation. IEEE Trans. Robot. **25**(4), 912–926 (2009)
17. Kivelevitch, E., Cohen, K., Kumar, M.: A market-based solution to the multiple traveling salesmen problem. J. Intell. Robot. Syst. **72**(1), 21–40 (2013)
18. Tang, Z., Ozguner, U.: Sensor fusion for target track maintenance with multiple UAVs based on Bayesian filtering method and hospitability map. In: 42nd IEEE International Conference on Decision and Control, pp. 19–24 (2003)
19. Tang, Z., Ozguner, U.: Motion planning for multitarget surveillance with mobile sensor agents. IEEE Trans. Robot. **21**(5), 898–908 (2005)
20. Tang, Z., Ozguner, U.: PF-HMap: A Target Track Maintenance Approach for Mobile Sensor Platforms with Intermittent and Regional Measurements. In: 45th IEEE Conference on Decision and Control, pp. 6757–6762. IEEE (2006)
21. Kassas, Z.M., Ozguner, U.: A nonlinear filter coupled with hospitability and synthetic inclination maps for in-surveillance and out-of-surveillance tracking. Syst., Man, and Cybern., Part C: IEEE Trans. Appl. Rev. **40**(1), 87–97 (2010)
22. Adamey, E., Ozguner, U.: Cooperative multitarget tracking and surveillance with mobile sensing agents: A decentralized approach. In: 2011 4th International IEEE Conference on Intelligent Transportation Systems. ITSC 2011, pp. 1916–1922 (2011)
23. Adamey, E., Ozguner, U.: A decentralized approach for multi-UAV multitarget tracking and surveillance. Proceedings SPIE 8389, Ground/Air Multisensor Interoperability, Integration, and Networking for Persistent ISR III, 838915 (2012)
24. Stone, L.D., Streit, R.L., Corwin, T.L., Bell, K.L.: Bayesian Multiple Target Tracking Second Edition. Artech House (2013)
25. Praveen Babu Choppala: Bayesian multiple target tracking. Victoria University of Wellington, PhD thesis (2014)
26. Cormen, T.H.: Introduction to Algorithms. MIT Press (2009)
27. Skiena, S.S.: The Algorithm Design Manual. Springer (2008)
28. Khaleghi, B., Khamis, A., Karray, F.O., Razavi, S.N.: Multisensor data fusion a review of the state-of-the-art. Information Fusion **14**(1), 28–44 (2013)
29. Chen, Z.: Bayesian filtering: From Kalman filters to particle filters, and beyond. Statistics **182**(1), 1–69 (2003)
30. Russell, S., Norvig, P.: Artificial Intelligence Prentice Hall (2009)
31. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics. MIT Press (2005)
32. Hahnel, D., Triebel, R., Burgard, W., Sebastian, T.: Map building with mobile robots in dynamic environments. IEEE International Conference on Robotics and Automation. IEEE ICRA 2003. Proceedings, pp. 1557–1563 (2003)
33. Thrun, S.: Learning occupancy grid maps with forward sensor models. Auton robot. **15**(2), 111–127 (2003)
34. Collins, T., Collins, J.J., Ryan, C.: Occupancy grid mapping: An empirical evaluation. 2007 Mediterranean Conference on Control & Automation. pp. 1–6. IEEE (2007)
35. Kalman, R.E.: A new approach to linear filtering and prediction problems. J. Basic Eng. **82**(Series D), 35–45 (1960)
36. Simon, D.: Optimal State Estimation. Wiley-Interscience (2006)
37. Doucet, A., Johansen, A.M.: A tutorial on particle filtering and smoothing: Fifteen years later. Handbook of Nonlinear Filtering **12**(3), 656–704 (2009)
38. Ristic, B., Arulampalam, S., Gordon, N.J.: Beyond the Kalman Filter. Artech House (2004)

Emrah Adamey received the B.S. degree in Control and Automation Engineering at Istanbul Technical University of Turkey in 2009. He worked as a graduate research assistant in 2010-2015 at the Center of Intelligent Transportation (CITR) Lab in the Ohio State University. He received his masters degree from the Ohio State University in 2015, and is currently a PhD candidate. His research interests include Multi-Object Tracking, Multi-Sensor Data Fusion, and Bayesian Filtering.

Abdullah Ersan Oğuz is an assistant professor at the Turkish Air Force Academy. He received the BS degree Electronic Engineering from Ankara University in 1994, MSc degree Electronics Engineering from the Gazi University in 2005 and his PhD degree in Electronics Engineering from the Aeronautics and Space Technologies Institute in 2012. He was a post doctoral researcher at the Ohio State University between 2014-2015. His current research interests include Simultaneous Mapping and Localization, Kalman Filter, Navigation, Control and UAVs.

Ümit Özgüner (LF'13) received the Ph.D. degree from University of Illinois. He held positions at I.B.M. Research Labs, University of Toronto, and Istanbul Technical University. Since 1981, he has been with The Ohio State University, where he is currently a Professor of electrical and computer engineering and a Senior Fellow of the Center for Automotive Research. He holds the TRC Inc. Chair on intelligent transportation systems (ITS). He is the author or coauthor of over 400 publications, including a 2011 book on autonomous ground vehicles. He has advised over 40 students in their M.S. research and over 25 students in their Ph.D. research. His research interests include ITS, and decentralized control and autonomy in large systems.