Motion Planning for Multitarget Surveillance With Mobile Sensor Agents

Zhijun Tang and Ümit Özgüner, *Member, IEEE*

***Abstract*—In the surveillance of multiple targets by mobile sensor agents (MSAs), system performance relies greatly on the motion-control strategy of the MSAs. This paper investigates the motion-planning problem for a limited resource of M MSAs in an environment of N targets ($M < N$). The kinematics of the MSA is modeled as a point mass moving at a constant speed with a bounded turning radius. Based on the fact that the track information of each target degrades over time, the motion-planning problem is formulated as an optimization problem whose objective is to minimize the average time duration between two consecutive observations of each target. In the case of a single MSA, the motion-planning problem is further interpreted so as to find a time-optimal loop path to traverse the targets. A gradient-approximation algorithm is then proposed to generate a suboptimal loop path for a mobile agent to traverse a sequence of target points. For the multi-MSA-multitarget case, a cooperative online motion-planning approach is developed.**

***Index Terms*—Cooperative motion control, mobile sensor agents (MSAs), multi-MSA-multitarget (MMMT) tracking, robot motion planning, sensor management, uninhabited air vehicles (UAVs).**

I. Introduction

THERE has been growing interest in performing target surveillance with mobile sensor platforms in recent years. In complex and dynamic environments such as transportation systems, flow networks, or battlefields, networked mobile sensor platforms are expected to be capable of adapting themselves to changes of the environment as well as target locations. Different from traditional sensor platforms, modern mobile sensor agents (MSAs) usually have a smaller field of view around a local area, and cannot cover the whole domain of interest all the time. As a result, the motion control of the MSAs becomes a crucial part of a successful surveillance system, especially when the resources of MSAs are limited (i.e., the number of MSAs $<$ the number of targets). Meanwhile, the possible nonholonomic constraints (e.g., minimum/maximum speed, minimum turning radius) on the MSAs, which is often the case in reality, complicates the corresponding solution strategies.

This paper studies the cooperative motion-planning problem for monitoring N targets by M MSAs ($M < N$). The work is motivated by the application for uninhabited air vehicles (UAVs) in both military [1]–[3], [7] and civilian surveillance systems [5]. The MSA here is modeled as a nonholonomic point mass moving on a two-dimensional (2-D) plane at a constant speed with a bounded turning curvature. This model is also called the *Dubins car* in the literature [10], [13], [14], [17], [19], [22], and has been widely used as the kinematic model of UAV by researchers [3], [4], [21], [39]. Each MSA is assumed to have an onboard sensor with a limited local field of view around itself, as illustrated in Fig. 1(a). It is also assumed that the MSAs move much faster than the targets do, which agrees with reality in UAV applications.

An example of the *multi-MSA-multitarget* (MMMT) scenario is shown in Fig. 1(b). In order to keep the targets in surveillance with limited MSA resources, the members of the MSA team have to fly back and forth to update the targets' status. Therefore, the motion planning in such an MMMT environment has to consider not only how each MSA goes from one point to another, but also which target (or target set) each MSA should take care of. This is essentially a combination of the problems of *sensor resource management* and *robot motion planning*. The main purpose of this work is to seek a systematic framework for designing a real-time implementable motion-planning approach for the MMMT scenario.

A. Related Work

Although both the problems of sensor resource management and robot motion planning have been intensively studied for decades, very few available methods can be directly applied to the MMMT scenario.

Most previous robot motion-planning approaches assume that a simple, specific origin-to-destination configuration is given to each agent, so that their main focus is usually on the optimal path generation between two predefined positions [10]–[19], [22], [23], [25]–[28]. However, with limited resources of MSAs, the task for each MSA often covers more than one target, which cannot be interpreted as a simple end-to-end configuration problem. Meanwhile, distinct from many traditional motion-planning applications (e.g., target search [1], [2], [8], [9] and target engagement [3], [24]), the surveillance job here does not end after each target is visited. The MSAs have to come back to the targets repetitively to update their status. The optimal path for a given cycle is not necessarily optimal in the long run.

Manuscript received August 20, 2004; revised December 29, 2004. This paper was recommended for publication by Associate Editor L. Parker and Editor H. Arai upon evaluation of the reviewers' comments. This work was supported by the Air Vehicles Directorate of the Air Force Research Laboratory and the Air Force Office of Scientific Research Collaborative Center of Control Science under Contract F33615-01-2-3154.

The authors are with the Department of Electrical and Computer Engineering, The Ohio State University, Columbus, OH 43210 USA (e-mail: tang.89@osu.edu; ozguner.1@osu.edu).

Digital Object Identifier 10.1109/TRO.2005.847567

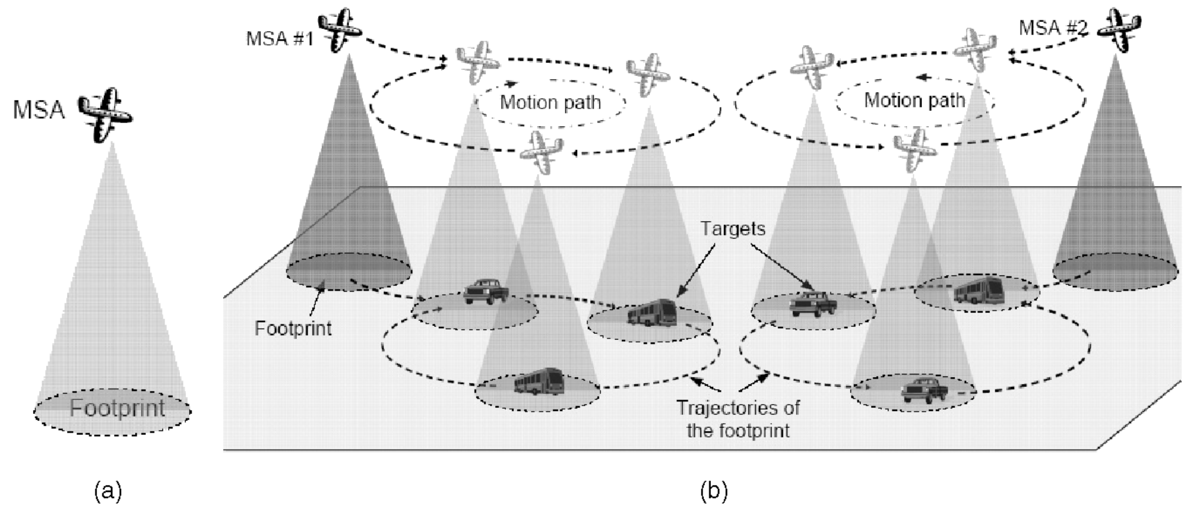

Fig. 1. (a) MSA and its footprint. (b) Example of the MMMT scenario.

On the other hand, the majority of previous work in the area of sensor resource management treats the sensor-control problem as (or similar to) a sensor-scheduling problem [29]–[32]. The cooperation of multiple sensors is often achieved by choosing different sensors (or sensor modes) for different tasks (targets) at different times. Motion planning is not involved in most previous approaches, which may not be a problem, as these approaches are applied to sensor platforms with large global coverage. As for mobile sensor platforms, however, controlling the sensors is not merely to assign them tasks or schedules, but to find them optimal motion plans.

Among the very few approaches in dealing with cooperative multisensor motion planning, Parker *et al.* [34] formulated the motion-planning problem as an optimization problem, whose objective is to maximize the collective time during which each target is monitored by at least one sensor agent. An approximation method based on the ALLIANCE architecture has been proposed in [34]. The motion control of each agent is achieved in an implicit way by a force vector. The force vector is essentially a tradeoff among different subgoals to keep the agents within a certain distance of the targets, as well as away from each other. Real-world experiments have demonstrated the feasibility of this approach with sufficient sensor resources. Jung *et al.*, on the other hand, suggest a region-based approach for cooperative multitarget tracking in a structured environment in [35], in which the whole area of interest is assumed to be divided into topologically simple regions. The objective of individual motion control is formulated to locate each agent a certain distance from the center of gravity of targets that it is tracking. However, the cooperation among the robots within the same region for multitarget tracking is not mentioned. Cortes *et al.* [36] studied the multisensor localization problem in a polygonal environment and developed a gradient-descent algorithm to realize optimal coverage and sensing policies. Each sensor agent is expected to converge to its optimal location and stay there. Similar to the other two methods mentioned previously, no motion constraints are considered, which makes it very difficult to apply these approaches directly to nonholonomic sensor agents with a minimum-speed constraint, such as the Dubins car. Walker *et al.* addressed the multi-agent-multitarget path-planning problem for the Dubins car in [39]. In their approach, the coupled target assignment and path-planning problems are solved at the same time by searching over a tree of stepwise feasible flying paths. The real-time A^* algorithm [33] is used to find a suboptimal path for each agent. By considering the effect of sensor footprint, this approach is capable of finding a suboptimal path for each agent, which is not necessary to pass the targets, as long as they will be covered by the sensor footprint. Unfortunately, it is still difficult to fit this method into the multitarget surveillance problem addressed here due to the following reasons. First, although this method allows the targets to be visited multiple times, the number of visits on each target has to be predetermined before path planning. Meanwhile, visiting a target multiple times does not necessarily lead to a good tracking performance. It is how these multiple observations are made in the time domain that determines the tracking performance. Furthermore, all of the targets are assumed to be strictly static in Walker's approach [39]. Thus, the objective of path planning is to generate the shortest feasible paths for the MSAs to traverse the targets once or multiple times. There is no replanning scheme in dealing with a dynamic environment with moving targets. There is some other research work that is related to the problem or a subproblem of the problem addressed here, such as [9], [20], and [21].

B. Main Contributions

As pointed out in [34], the general MMMT motion-planning problem is NP-hard, both in the number of targets and in the number of sensor agents. Thus, looking for the global optimal solution is not only computationally prohibitive, but also unsuitable for a dynamic environment with moving targets. To find a practical method that is feasible in real-world applications, a suboptimal approach with much less computational load is proposed in this paper.

The main contributions of this paper are the following.

1) Based on the fact that the uncertainty of a target status is proportional to how frequently the target track is updated, the motion-planning problem is modeled as an optimization problem, whose objective is to minimize the average time duration (ATD) between two consecutive observations of each target. This formulation is distinct from previous approaches [32]–[36], [39], and can be applied to general surveillance and information-gathering problems with limited sensor resources, in which one target can be a building, an intersection, a car under surveillance, or a military unit.
2) A computationally efficient gradient-based method is developed to determine a suboptimal loop path for a single MSA to traverse multiple target points (i.e., the single-MSA-multitarget (SMMT) case). The path search is conducted in a reduced search space, in which we have proved that the global optimal path always lies. This method can also be applied to the general multitarget-engagement problem.
3) A decentralized online motion-planning algorithm for multitarget tracking by multi-MSAs is proposed. At each time instant, we further decouple the MMMT motion-planning problem into several disjoint SMMT problems based on heuristic but computationally efficient rules, which links the traditionally separately studied sensor resource-management problem and robot motion-planning problem together. Also, an online replanning scheme is developed to deal with moving targets.

The rest of the paper is organized as follows. Section II gives the problem statement. The existing methods for the simplest single-MSA-single-target (SMST) case is briefly reviewed in Section III. Then, a suboptimal path-generation approach for an MSA of type Dubins to traverse a set of target points in minimum time (i.e., the SMMT case) is introduced in Section IV. After that, a target-based online motion-planning algorithm for the MMMT case is proposed in Section V, followed by the coverage stability of the approach discussed in Section VI. Simulations and results are shown in Section VII. Finally, conclusions are given in Section VIII.

II. Problem Statement

Consider a team of M homogeneous MSAs, each of which is modeled as a nonholonomic Dubins car [10]

$$\begin{cases} \dot{x}_j(t) = V_M \cos\vartheta_j(t) \\ \dot{y}_j(t) = V_M \sin\vartheta_j(t) \\ \dot{\vartheta}_j(t) = u_j(t), & |u_j(t)| \leq \frac{V_M}{R_M} \end{cases} \tag{1}$$

where $s_j(t) = (x_j(t), y_j(t)) \in \mathcal{R}^2$ and $\vartheta_j(t) \in [0, 2\pi]$ denote the horizontal position and orientation of agent j, $j = 1, \ldots, M$ at time instant t, respectively. V_M is the speed of the MDAs, R_M is the minimum turning radius, and $u_j(t)$ is the control input.

The task of the MSAs is to monitor a number of ground targets $Q = \{q_i\}$, $i = 1, \ldots, N$, where $q_i = (q_{ix}, q_{iy})$ denotes the expected ground position of target i. Without further notice, we will use i and j as the indexes for the targets and the MSAs, respectively, in the rest of this paper. Each MSA is assumed to have an onboard sensor with a restricted local field of view around itself. In this paper, the footprint of MSA j at time instant t is defined as a small circle centered at $s_j(t)$, as shown in Fig. 1(a). However, the general result developed here can be extended to other footprint shapes. The sensors are assumed to be reliable, so that a target is observed when it is inside the footprint of any one of the MSAs. Here we further assume that when one MSA is going to check the status of target i, it will come over on top of the estimated position of the target (q_i). In other words, it will locate the center of its footprint at q_i. One may argue that it is not necessary to have the sensor footprint centered at the target to make the observation, and a better path may be achieved by considering taking advantage of the effect of the footprint [33], [39]. As we will show later in this paper (Section VI), this approximation not only simplifies the problem, but also increases the robustness and stability of a motion plan for the MSA in dealing with the undeterministic motion of moving targets.

With limited resources of MSAs (i.e., $M < N$), the targets may not always be inside the footprints of the MSAs, in general. Assuming that the speed of the MSA is much faster than that of the targets, what the MSAs have to do is to cooperatively hover around the targets to keep them under surveillance.

To mathematically formulate this motion-planning problem, we start from the simplest single-target case. An important feature of the target track is that the uncertainty of the track information increases as the target is off observation until it is recaptured by the MSAs, which means the tracking performance depends on how frequently the target track is updated. Therefore, the motion-planning problem here can be posed as an optimization problem whose objective is to minimize the ATD between two consecutive observations of the target. In the case of multiple targets, the optimization goal is to minimize the mean of the ATD between two consecutive observations of each target, as follows:

$$J = \frac{1}{N} \sum_{i=1}^{N} \mathrm{ATD}_i \tag{2}$$

where ATD_i is the ATD between two consecutive observations of the ith target.

Equation (2) can be regarded as a generic form for the general information-gathering problem with limited sensor resources, in which one target can be a building, an intersection, a car under surveillance, or a military unit. This formulation is distinct from previous approaches [32], [34]–[36], which conforms to the fact that with limited MSA resources, the desired motion plan of the MSAs should appropriately distribute their effort/time not only in the target domain (or space domain), but also in the time domain.

In the rest of the paper, we shall address the motion-planning problem described by (2) step by step, starting from the simplest SMST case (Section III), to the more complicated SMMT case (Section IV), and finally to the general MMMT case (Section V).

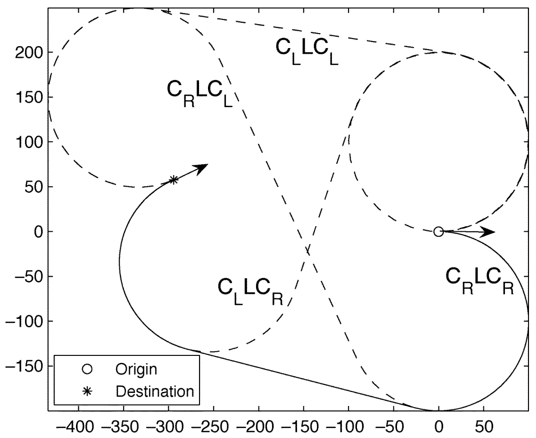

Fig. 2. Example of finding the minimum-time trajectory.

III. Time-Optimal Motion Planning for Single-Target Engagement (SMST Case)

When there is only one MSA and one target, the motion-planning problem (2) is equivalent to the traditional single-target engagement problem [10]–[19], [22]

$$\text{Minimize } J = \int_{t_0}^{t_f} dt \text{ subject to (1)}$$
$$\text{with } s(t_0) = q_0, \vartheta(t_0) = \vartheta_0,\ s(t_f) = q_1,\ \vartheta(t_f) = \vartheta_1. \quad (3)$$

Note that in different applications, the initial and final conditions (i.e., q_0, ϑ_0, q_1, and ϑ_1) can be either fixed or free. The earlier work of Dubins [10] has proved the existence of a time-optimal path for a system of type (1). More recently, Reeds and Shepp [11] have extended the work to the case that the robot can move both forward and backward. Meanwhile, Sussmann [16] and Boissonnat [17] have solved the problem using Pontryagin's Maximum Principle, which coincides with the following theorem given by Dubins in [10].

Lemma 1: For the time-optimal control problem described in (3) with any initial and final configurations, there exists a minimum-time trajectory χ^* for system (1), which is a combination of arcs from circles (which we shall denote as C) and straight-line segments (which we shall denote as L). More specifically, the time-optimal path χ^* is a subpath of a path of type circle–line–circle (CLC) or of type circle–circle–circle (CCC).

Practically, the minimum-time trajectory, as well as the corresponding control u^*, can be determined by simple geometric methods, which basically choose the shortest path from a finite set of extremal trajectories. An example of seeking the optimal trajectory is shown in Fig. 2. In this example, there are only four candidate paths that satisfy *Lemma 1*, which are C_LLC_R (i.e., a left-turn arc followed by a straight line and a right-turn arc), C_LLC_L, C_RLC_L, and C_RLC_R. The minimum-time trajectory in this case is C_RLC_R. More complete discussions on how to geometrically synthesize the optimal trajectory can be found in [13]–[16].

Remark 1: The geometric method provides us a convenient way to specify the minimum-time trajectory for an end-to-end problem. This computationally low-cost feature of the geometric solution to the SMST problem is very helpful in developing our algorithm for the multitarget case.

IV. A Suboptimal Motion-Planning Approach for the SMMT Case ($M = 1$, $N > 1$)

In this section, we focus on the case of a single MSA $(M = 1)$ monitoring multiple targets $(N > 1)$. Given a specific list of targets, an MSA has to move around and update the status of the targets one after another. Here, we further assume that the targets are spread in the field of interest, with considerable distance between each other. In the case that several targets are very close to each other, we can replace them by a pseudo target. Thus, the corresponding motion plan for the MSA is a *traverse* path. By *traverse*, here we mean one MSA visiting each target once along such a path. Since the surveillance job is not finished after one cycle, what we are looking for is actually a loop path. To minimize (2), the MSA has to execute the traverse loop as fast as possible. Similar to (3), we can rewrite (2) for this case as the following time-optimal control problem:

$$\arg\min_{Q',u} J\left(Q'|Q' = \{q'_i\} \in \Omega\right) = \frac{1}{N}\sum_{i=1}^{N} T_i \text{ subject to (1)}$$
$$\text{with } s(t_0) = q'_0,\ s(t_0 + T_i) = q'_i \quad (4)$$

where $q'_N = q'_0$, Ω is the collection of all permutations of the target set $Q = \{q_i\}$, and T_i is the flying time between q'_{i-1} and q'_i due to the controller u.

Although motion planning for the single-target case is quite a mature area, the extension of the available methods to the multitarget case is a nontrivial problem. As (4) implies, to find the optimal motion plan, one has to search for the best traverse order, as well as the time-optimal trajectory for it. The number of possible suboptimal trajectories in the search space increases exponentially as the number of targets increases. For example, a set of N targets will lead to $(N-1)!$ distinct traverse order (rather than $N!$, due to the symmetry of a loop), which makes it very difficult to achieve the optimal path, since the search space is literally $(N-1)!$ times bigger. Fortunately, since we are looking for a loop path in the MSA-target scenario, the desired path is usually in a circle pattern around the geometric center of the targets. Based on this heuristic observation, here we develop the following geometric method with much less computational complexity $(O(N \log N))$ to determine a suboptimal traverse order first.

A. Determination of the Traverse Order

The geometric method to determine the traverse order consists of three steps.

1) Calculate the geo-center $(\bar{q})$ of the target set $Q = \{q_i\}$.
2) Calculate the orientation (α_i) of each target point (q_i) with respect to the center $\bar{q}$, as shown in Fig. 3.
3) Sort the target points by $\{\alpha_i\}$, and the result is chosen as the traverse order.

After determining the traverse order, the motion-planning problem for the SMMT case further reduces to a preordered multitarget-engagement problem.

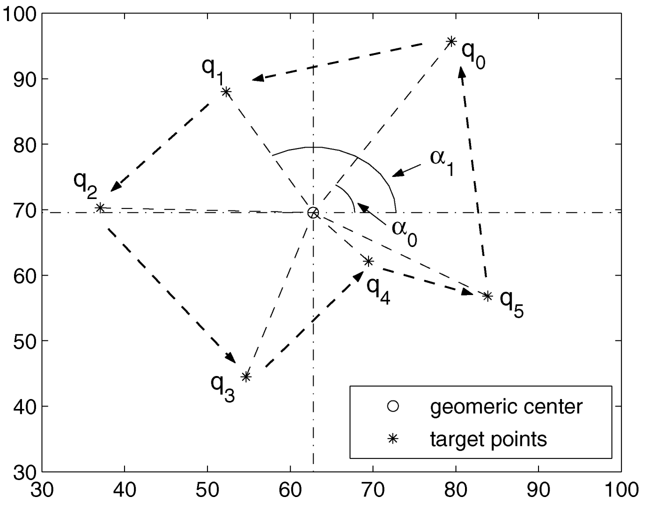

Fig. 3. Example of determining the traverse order.

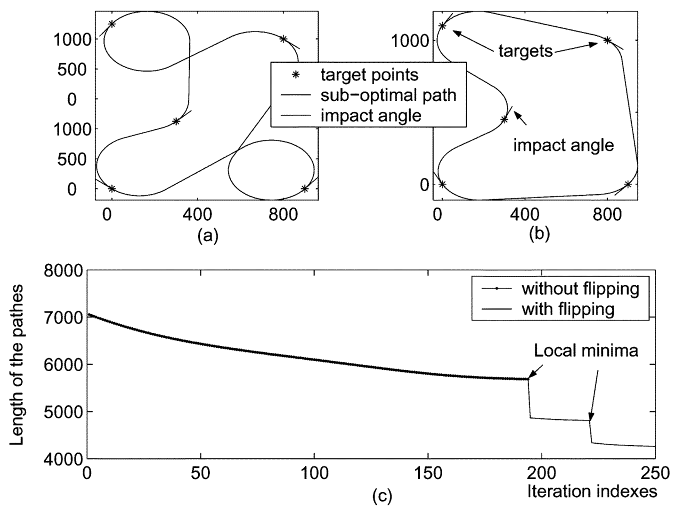

Fig. 4. Example for the SMMT case ($N = 5$). (a) Suboptimal path without flipping. (b) Suboptimal path with flipping. (c) Length comparison.

B. Motion Planning for the SMMT Case With a Given Traverse Order by Approximated Gradient Decent

Definition 1: Given a target set $Q = \{q_i\}_{i=0,\dots,N-1}$, an *impact angle configuration* Θ is the set of impact angles $\Theta = \{\theta_i\}_{i=0,\dots,N-1}$, where θ_i denotes the orientation of the MSA as it passes target i.

Definition 2: A *motion plan* $\chi(Q,\Theta) = \{\chi_i(Q,\Theta)\}$ is an admissible motion trajectory for the MSA to traverse a pre-ordered target sequence $Q = \{q_i\}_{i=0,\dots,N-1}$, where $\chi_i(Q,\Theta)$ denotes the subpath of $\chi(Q)$ from q_i to q_{i+1}. Note that in the case of a loop path, $q_N = q_0$.

The time for the MSA to execute a motion plan χ is then denoted as $J(\chi)$.

Definition 3: A motion plan $\chi(Q,\Theta)$ is a Dubins path if each subpath $\chi_i(Q,\Theta)$ of $\chi(Q,\Theta)$ is the time-optimal path between q_i and q_{i+1} that satisfies (3).

Lemma 2: Given a target set Q and its impact angle configuration Θ, there exists a motion plan that is a Dubins path, which is denoted as $\bar{\chi}(Q,\Theta)$.

Lemma 2 is a direct result from *Lemma 1*, which also leads to the following corollary and proposition.

Corollary 1: Given two impact angle configurations (Θ,Θ') and the corresponding Dubins paths $\bar{\chi}(Q,\Theta)$ and $\bar{\chi}(Q,\Theta')$, if $\theta_i = \theta'_i$ for all $i = 0,\dots,N-1$, except for $i = i^*$, then we have $\bar{\chi}_i(Q,\Theta) = \bar{\chi}'_i(Q,\Theta)$ for all $i = 0,\dots,N-1$, except for $i = i^* - 1$ and i^*.

Proposition 1: Define $\Gamma(Q)$ as the collection of Dubins paths for all possible impact angle configurations: $\Gamma(Q) = \bigcup_\Theta\{\bar{\chi}(Q,\Theta)\}$, where $\bar{\chi}(Q,\Theta)$ denotes the Dubins path that traverses Q with impact angle configuration Θ. If a motion plan χ^* is the global minimum-time plan, it follows that $\chi^* \in \Gamma(Q)$.

Remark 2: Proposition 1 tells us that given any impact angle configuration Θ, the only candidate for the global time-optimal path is the Dubins path $\bar{\chi}(Q,\Theta)$.

Therefore, in order to obtain the time-optimal path, we can minimize $J(\bar{\chi}(Q,\Theta))$ by searching in the reduced path space $\Gamma(Q)$. An intuitive way to realize that is to use the gradient method. However, the nature of the Dubins path is space-dependent, which cannot be formulated in a simple explicit way, and so is $J(\bar{\chi}(Q,\Theta))$. Meanwhile, the cost function $J(\bar{\chi}(Q,\Theta))$ may not even be differentiable, as some subpaths of $\bar{\chi}$ change their types. To deal with this problem, we use the following gradient approximation method.

Consider two different impact angle configurations $\Theta = \{\theta_0,\theta_1,\dots,\theta_i,\dots,\theta_{N-1}\}$ and $\Theta' = \{\theta_0,\theta_1,\dots,\theta'_i,\dots,\theta_{N-1}\}$. Let $\chi = \bar{\chi}(Q,\Theta)$ and $\chi' = \bar{\chi}(Q,\Theta'))$ be the corresponding Dubins paths. According to *Corollary 1*, we have

$$J(\chi) - J(\chi') = J(\chi_{i-1}) + J(\chi_i) - J\left(\chi'_{i-1}\right) - J\left(\chi'_i\right). \quad (5)$$

Based on (5), we can approximate the gradient function $J_{\theta_i}(\chi)$ as

$$\frac{\partial J(\chi)}{\partial\theta_i} \approx \frac{J(\chi^+)-J(\chi^-)}{2\Delta\theta} = \frac{J\left(\chi^+_{i-1}\right)+J\left(\chi^+_i\right)-J\left(\chi^-_{i-1}\right)-J\left(\chi^-_i\right)}{2\Delta\theta} \quad (6)$$

where $\chi^+ = \bar{\chi}(Q,\{\theta_0,\dots,\theta_i+\Delta\theta,\theta_{i+1},\dots,\theta_{N-1}\})$ and $\chi^- = \bar{\chi}(Q,\{\theta_0,\dots,\theta_i-\Delta\theta,\theta_{i+1},\dots,\theta_{N-1}\})$.

Remark 3: One important feature of (6) is that the change of the impact angle of one target point q_i only affects two subpaths of $\bar{\chi}$: $\bar{\chi}_{i-1}$ (from q_{i-1} to q_i) and $\bar{\chi}_i$ (from q_i to q_{i+1}). Thus, the computational complexity of calculating the approximated gradients $\nabla\tilde{J}_\Theta$ is just $O(N)$, which is equivalent to that of the traditional gradient method, rather than $O(N^2)$.

With the help of (6), we can search for a suboptimal path in a recursive way, as the traditional gradient method does. The corresponding update equation for Θ is

$$\Theta(k+1) = \Theta(k) - \eta\nabla\tilde{J}_\Theta \quad (7)$$

where η controls the convergence speed of the algorithm.

Fig. 4(a) shows an example of suboptimal path obtained by the approximated-gradient method introduced above. The initial condition $\Theta(t_0)$ is randomly chosen. Although the global optimal path is always contained in the reduced search space $\Gamma(Q)$, and the approximated-gradient method provides us a fast way to find a local optimum, the convergence of this method toward

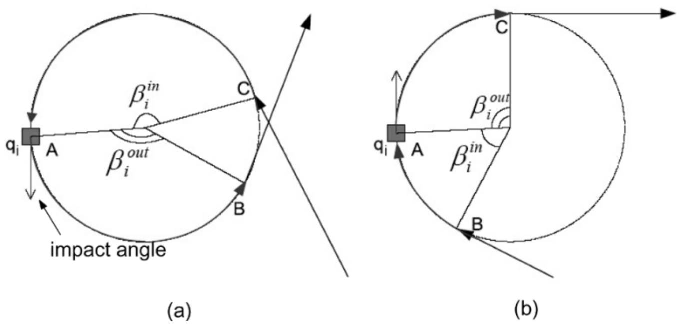

Fig. 5. Illustration of the check-and-flip procedure. (a) Before flipping. (b) After flipping.

the global optimum is not guaranteed. Unlike the single-target case, there are many local minima for the multitarget-traverse problem. Actually, the global minimum-time path in this example is shown in Fig. 4(b). In [21], a set of empirical initial angular configurations is chosen to preserve a good search performance. In this approach, we revise the optimization procedure above by adding a *check-and-flip* procedure, as follows.

C. Revised Algorithm

By carefully investigating the subpaths that pass each target (as shown in Fig. 5), it is noticed that most of the local minimal paths have at least one such subpath that is an arc larger than π [Fig. 5(a)]. For example, the suboptimal path shown in Fig. 4(a) has two large arcs as it passes the upper-left target and the bottom-right one. For each of these subpaths [e.g., Fig. 4(a)], a better configuration often comes from a totally opposite impact angle [e.g., Fig. 5(b)]. Based on this observation, here we revise the approximated-gradient method by adding an extra step to check the characteristics of a "suboptimal" path when it converges to a local minimum.

According to *Lemma 1*, the subpaths that come into and go out from one target point q_i are two arcs ($\widehat{AB}$ and $\widehat{AC}$ in Fig. 5). Denote the radians of the two arcs as β_i^{in} and β_i^{out}, as shown in Fig. 5. We revise the search algorithm as follows.

1) Search the optimal path by the approximated-gradient method until it converges to a local minimum.
2) Calculate the radians of the arcs passing each target point: $\rho(i) = |c_i^{\text{in}}\beta_i^{\text{in}} + c_i^{\text{out}}\beta_i^{\text{out}}|$.
3) $i^* = \text{argmax}_i\{\rho(i)\}$, where c_i^{in}, $c_i^{\text{out}} = 1$ if the arc is clockwise, and -1, otherwise.
4) If $\rho(i^*) \leq \pi$ or $k > T_k$, quit, where k is the counter of the number of flips.
5) Flip the impact angle of target i^* over: $\theta_{i^*} \leftarrow \theta_{i^*} + \pi$, $k = k + 1$, and go back to step 1).

The additional steps 2)–5) in the algorithm above is called the procedure of *check-and-flip*, which reverses the impact angle on a target if the subpath that passes it includes an arc larger than π. As the example illustrated in Fig. 4 shows, the revised algorithm evolves exactly the same way as the approximated-gradient method does until a local minimum is found. Without the check-and-flip procedure, the search will stop and a suboptimal path is obtained [Fig. 4(a)]. Because of the flipping, this revised algorithm is capable of escaping from this local optimum and another one following that [Fig. 4(c)], which helps the search converge to a shorter path (the global optimum in this case), as shown in Fig. 4(b).

Remark 4: The check-and-flip procedure here reduces the possibility of local minima, but still does not guarantee the convergence to the global optimum. Especially when two target points are close to each other, the minimum-time path between them often results in an arc larger than π, which will always cause the flipping. Therefore, in practice, we also limit the number of check-and-flip procedures within a certain threshold T_k, as indicated in step 4). The shortest path among the multiple local minima that are caused by the check-and-flip procedures is then selected as the search result.

Remark 5: The path-generation algorithm introduced here can be extended to the general multitarget-engagement scenario with a given order, which is not necessarily a loop (i.e., $q_N \neq q_0$). The geometric method to determine the traverse order introduced earlier, however, is not suitable for the general multitarget-engagement problems without a predetermined order.

V. Decentralized Cooperative Motion-Planning Approach for the MMMT Case

The general MMMT motion-planning problem is NP-hard, both in the number of sensor agents (M) and the number of targets (N) [34]. It is computationally prohibitive to find the optimal solution for motion planning. Furthermore, due to the nondeterministic motion of the targets, it is even more difficult to make an optimal long-term motion plan for the MSAs. In this section, we propose a suboptimal online motion-planning approach with a lower computational load, as follows.

In this scenario, the MSAs are the only sensor sources available. Without any prior knowledge, the motion of the targets is unpredictable. Here, we use the most previous positional measurement of each target as its estimated position until it is renewed by the MSAs. Since we assume the MSA moves much faster than the targets, the possible movements of the targets are then regarded as system perturbations, which are adapted by adjusting the motion plans through online replanning.

At each time, the motion planning consists of two steps: *task decomposition* and *individual path generation*.

In task decomposition, the whole task of tracking N targets is divided into M disjoint task assignments. Then, given a task assignment, each MSA makes its own motion plan, which can be achieved by the approach introduced in the previous section.

A. Task Decomposition

Given a set of N elements, the total number of ways to partition these elements into M nonempty subsets is called the *Stirling set number* $S(N, M)$. We shall use this notion with the elements representing targets. The Stirling set number can be obtained by the following equation [41]:

$$S(N, M) = \frac{1}{M!}\sum_{j=0}^{M-1}(-1)^j C(M, j)(M - j)^N \tag{8}$$

where $C(M, j)$ is a binomial coefficient.

Equation (8) indicates that $S(N, M)$ increases exponentially as N or M increases, which makes the task decomposition an NP-hard problem by itself. Since the desired motion plan for each MSA is a traverse loop, and the ultimate goal is to have the MSAs execute their loop paths in the shortest time, it is more likely that the optimal partitioning divides the targets into clusters. Based on this consideration, we use the K-means clustering method here to realize the task decomposition in a recursive way. The details of the K-means clustering method can be found in [45].

After the task decomposition, we still have to distribute the M tasks (i.e., target groups) among the M MSAs. An optimal assignment is defined as the one that requires the shortest total time for the MSAs to catch up their tasks. To achieve such an optimal M-to-M assignment, we use Murty's k-*best* algorithm [42].

Remark 6: Although the M-to-M assignment still yields a considerable computational load, it only happens once in the initialization stage of the MMMT motion-planning problem. With the assumption that the MSAs move much faster than the targets, the majority of the new target assignments generated by online replanning (see Section V-B) will be consistent with the previous ones. Most MSAs will stay with their current target groups after replanning, except that some targets may be switched to other groups. In implementation, the cluster center $\bar{q}_j$ from the previous planning cycle can be used as the default initial cluster center for MSA j's new motion plan. No M-to-M assignment is needed in the replanning. Therefore, as far as the long-term performance is concerned, the cost of assignment initialization can be neglected. There are also several approaches suggesting an efficient way to implement the k-best assignment algorithm [43], [44].

B. Online Motion Planning

The motion plan for each MSA is essentially a function of the expected positions of the targets (Q). In reality, a target may move away from its previous spot. It is necessary to have a corresponding replanning scheme as the target information is updated. In this approach, the motion replanning is achieved in a decentralized way.

Define $Q_j(k) = \{q_i^j(k)\}$, $i = 1, \ldots, n_j$, as the task assignment for MSA j at time t_k. Since the status of one target $q_i^j(k)$ can only be updated when it is observed by MSA j, it is reasonable for the MSA to renew its traverse plan once the status of each target in its task assignment is updated (from $Q_j(k)$ to $Q_j(k+1)$). As one MSA reconsiders its motion plan, it may decide to change its task assignment, which means the MSA may request some targets from or pass some targets to other MSAs. The exchange of target assignment here is called a procedure of *target handoff.* In summary, a replanning process is activated on one MSA by either one of the following events.

- **Event 1: An MSA finishes its current traverse loop.**
  In this case, the MSA will reconsider its task assignment by regrouping the targets, based on its own target information and the most recent information it obtained from other MSAs. Here we assume that the target information is shared by the whole MSA team. Note that to guarantee such a communication capability, the communication channel among the MSAs has to have a bandwidth no less than $O(MN)$ [34].
  If there is no change in the task assignment (i.e., $Q_j(k+1) = Q_j(k)$), the MSA only has to adjust its own traverse loop path. Otherwise, a target handoff is triggered.
- **Event 2: An MSA is requested for a target handoff.**
  In this case, the MSA will renew its target list $(Q_j(k+1))$, and then replan the optimal path for the new task. Note that there will be no negotiation between the MSAs in this approach. An MSA involved in a target handoff will unconditionally accept the change request of its task assignment. There are several advantages of this nonnegotiation scheme. First of all, there will be no deadlock in the decision making among MSAs. Second, the task partitioning is always complete, since no target will be abandoned accidently during the target handoff.

Remark 7: It is worth noting that for **Event 1**, one MSA will go through the whole motion-planning procedure, including task redecomposition (based on its own global information) and its own path regeneration. As for the result of **Event 2**, however, one MSA only has to update its own path. In summary, the online replanning is achieved asynchronously by the MSAs. One MSA reconsiders its motion plan only when either of the two events above happens.

Once a replanning process is activated, the MSA will renew its traverse path according to the new task assignment and catch it up from its current configuration, which is briefly summarized as follows.

1) Renew the traverse order from $Q_j(k)$ to $Q_j(k+1)$.
   In case of **Event 1**, keep $q_0^j(k)$ as the start point $q_0^j(k+1) \leftarrow q_0^j(k)$.
   In case of **Event 2**, choose the next unvisited target point in the previous plan as $q_0^j(k+1)$.
2) Renew the traverse path $\bar{\chi}(Q_j(k+1), \Theta_j(k+1))$.
3) Obtain the minimum-time path from the current configuration of the MSA $\{s(t_k), \vartheta(t_k)\}$ to the start point of the new traverse loop $\{q_0^j(k+1), \theta_0^j(k+1)\}$.
4) Execute the new traverse path $\bar{\chi}(Q_j(k+1), \Theta_j(k+1))$ from $q_0^j(k+1)$ to $q_{n_j-1}^j(k+1)$ until a new event is triggered.

VI. Coverage Stability Analysis

Stability and robustness is very important to a motion plan, and show how reliable the motion plan is in response to system perturbations. Some perturbations are caused by numerical errors when we generate the desired trajectory coordinates from the motion plan or by process noises, as we realize the trajectory in physical maneuvering. A good analysis on the stability and robustness of the time-optimal trajectory generation for a Dubins car can be found in [19]. To deal with the high sensitivity to modeling and measurement noises that a time-optimal path inherits, Turnau *et al.* proposed a near time-optimal planning method in [38]. Our aim in this section, however, is to investigate the stability in the sense of coverage, which is motivated by the question of whether the motion plan is able to maintain the target-tracking job as the targets are moving.

Without any prior knowledge, the motion of each target is unpredictable when it is outside the footprints of the MSAs.

When one MSA traverses the estimated target positions (the previous positional measurements in this case), the hope is that each target is not far away, and is still covered by the footprint of the MSA. Obviously, this coverage is not unconditionally guaranteed by any motion plan. The more frequently the MSA comes back to renew the target track, the more likely the target is close to the estimated position. If the target is not inside the sensor footprint, the MSA has to search around for the target, rather than just take a single look, as planned. As result, it will take more time to finish a traverse cycle for the MSA and lead to more possibility of missing the target in the future, which means that the motion plan is unstable in this sense.

Consider a target observed at $q = (q_x, q_y)$. Assuming that the motion plan is to recheck the status of this target after T, it obviously that the coverage is guaranteed if $\rho(q, T)$ can still be covered by the sensor footprint, in which $\rho(q, T)$ is the collection of all the possible positions of the target after T.

Proposition 2: The coverage of a motion plan is stable if

$$TD_i \leq \frac{D}{2v_{\max}}, \quad \text{for all of the } i = 1, \ldots, N \tag{9}$$

where TD_i is the time duration between two consecutive observations of target i. $v_{\max}$ is the maximum speed of a target, and D is the minimal diameter of the footprint of the MSA.

The proof of *Proposition 3* is fairly straightforward. As long as (9) holds, $\rho(q_i, T_i)$ will be a circle of radius $v_{\max}T_i$, centered at its previous position q_i. Thus, the MSAs just have to pass by the previous position of each target. No matter how the targets move, they will always be covered by the MSAs.

Obviously, the coverage stability is not preserved in an open field as long as $N > M$, in which the targets can move in opposite directions to break the coverage. Nevertheless, *Proposition 2* gives us a clue to evaluate the feasibility of a motion plan, which can be applied to other important sensor-management issues beyond the scope of this paper, such as task assessment and high-level sensor resource management.

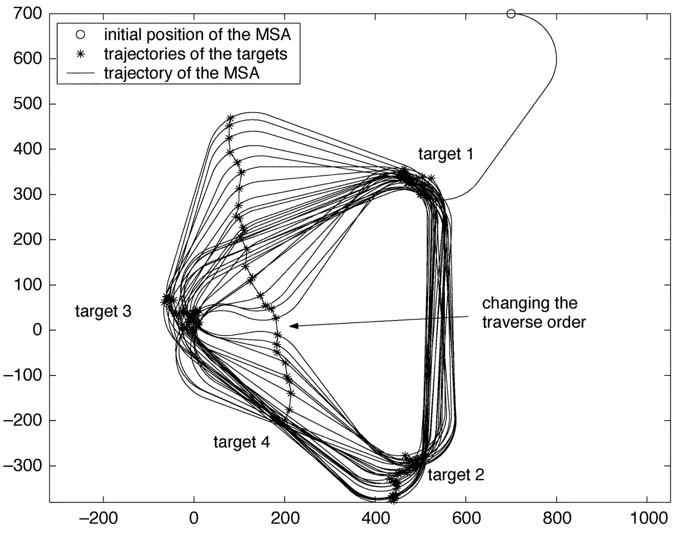

Fig. 6. Simulation 1: One MSA, three randomly walking targets, and one maneuvering target.

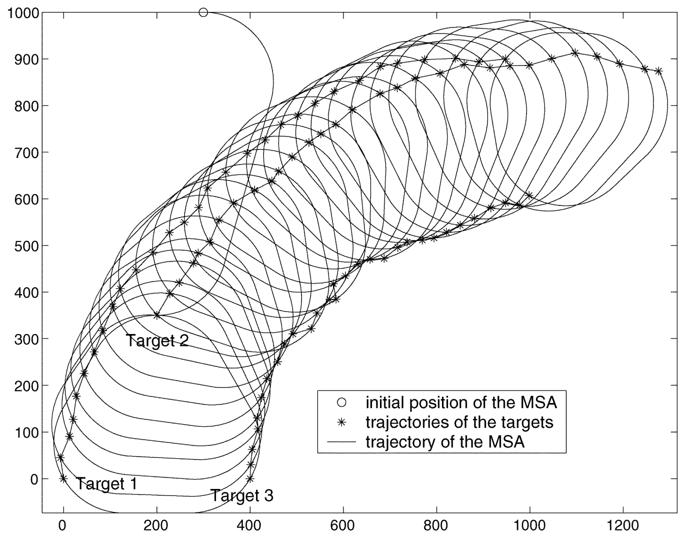

Fig. 7. Simulation 2: One MSA and a moving convoy of three targets.

VII. Experiments and Results

A. Path Generation for the SMMT Case

Two simulations for the SMMT case and their results are shown in Figs. 6 and 7. In simulation 1 (Fig. 6), four targets are generated. Three of them (targets 1–3) are modeled as random walks, while the other one (target 4) is a maneuvering target. As Fig. 6 shows, the MSA is able to smoothly adjust its motion plan as the targets are moving around. Note that at some point, the MSA changes the traverse order from 1-3-4-2 to 1-4-3-2 to achieve a shorter traverse loop. Fig. 7 shows another example, in which there is a convoy of three moving targets. As the convoy moves, a smooth trajectory is generated for the MSA to follow up the convoy and keep updating the status of the targets.

The performance of the suboptimal method has been examined by Monte Carlo simulations. Fig. 8 shows the results of one experiment, in which we chose $N = 6$ and conducted 500 Monte Carlo simulations. The histogram of approximation errors is shown in Fig. 8. The actual minimum-time path for each simulation is achieved by exhaustively searching over all of the permutations of the target points. In this experiment, most of the suboptimal paths are no more than 10% longer than the actual one. The average error is only 2.44%, which is quite satisfactory.

B. Cooperative Online Motion Planning for the MMMT Case

In this experiment (Fig. 9), four MSAs, 16 randomly walking targets, and two maneuvering targets are generated. The parameters are chosen as $R_M = 72$ m, $V_M = 96$ mph, and $v_{\max} = (1/20)V_M$. The MSAs all start from the center of the field for simplicity, whose footprint is defined as a circle with a 200-yd diameter (i.e., $D = 200$ yd). As Fig. 9 shows, two target handoff events are triggered as the two maneuvering targets move across the field. Note that as a decentralized algorithm, each MSA will asynchronously update its own motion plan if there is no target handoff. When a target handoff happens, it only affects the motion plans of the MSAs that are involved in this handoff.

The ATD between two consecutive observations of each target is shown in Fig. 10. Note that because the targets are moving, each TD_i is time-variant. The result shows that each

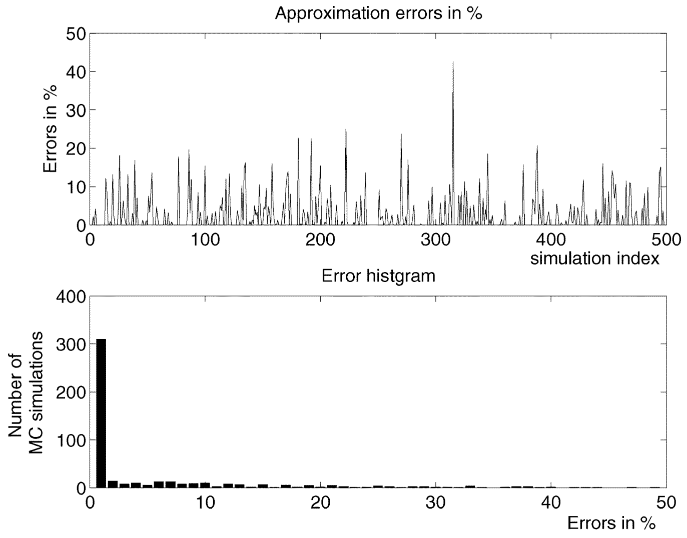

Fig. 8. Performance evaluation for the geometrical method to determine the traverse order ($N = 6$, 500 Monte Carlo simulations).

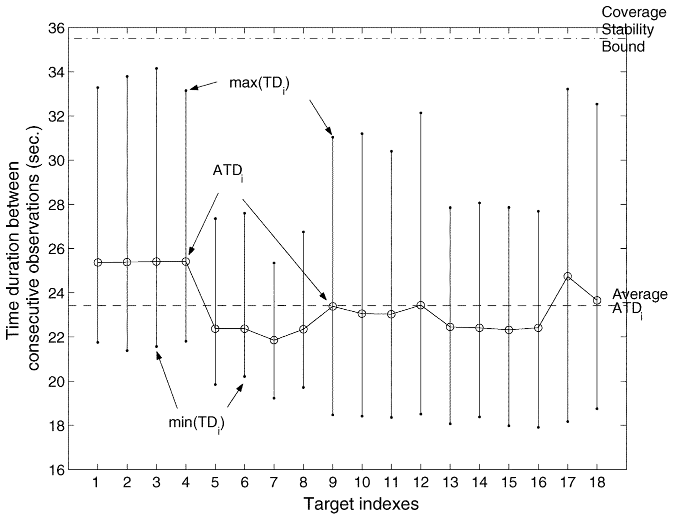

Fig. 10. Time durations between consecutive observations of each target in simulation 3.

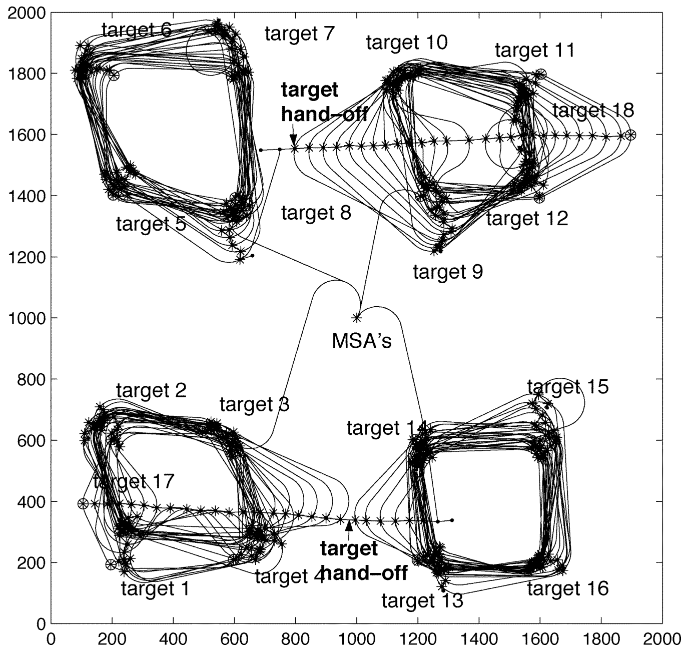

Fig. 9. Simulation 3: Four MSAs and 18 targets (16 randomly walking and two maneuvering targets).

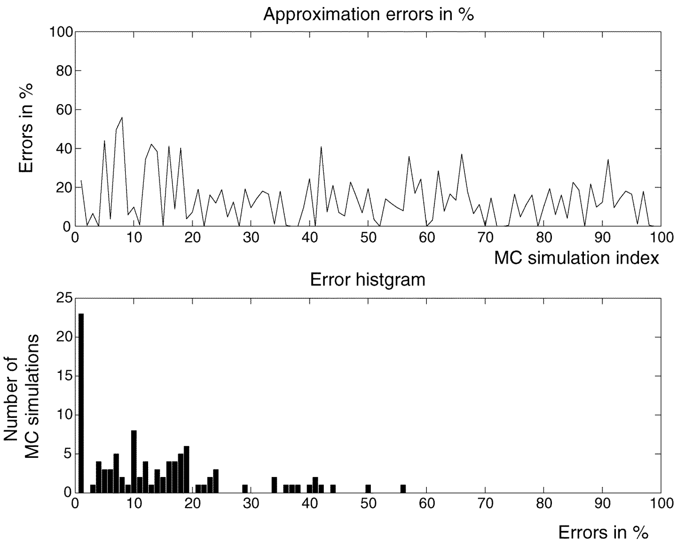

Fig. 11. Performance evaluation for task assignment by K-means clustering ($N = 12, M = 3$, 100 Monte Carlo simulations).

TD_i is less than $D/2v_{\max} = 35.5$ s, which satisfies *Proposition 2*. The average ATD is 23.4 s. We also compare this result with a pseudostationary sensor platform, whose field of view is equal to the summation of the footprints of the MSAs. Assume that the sensor is scanning the whole field (2000 yd $\times$ 2000 yd in this simulation) with the same speed of the MSA. It will take the pseudosensor $2000 \times 2000/(MDV_M) = 75.8$ s to scan the whole field once. In other words, the pseudosensor needs at least 75.8 s to update the track information of each target, which is much longer than the result of the proposed approach with multiple MSAs (which is 23.4 s).

The performance of the task partitioning method has also been evaluated by Monte Carlo simulations. Fig. 11 shows the results of one experiment, in which $(N, M) = (12, 3)$ and 500 Monte Carlo simulations are conducted. The results of the suboptimal method based on K-means clustering is compared with those of the actual optimal partitions. The optimal partition is also achieved by an exhaustive search. In this experiment, most of the suboptimal paths are no more than 15% longer than the actual one. The average error is 11.21%.

VIII. Conclusion

This paper addresses the motion-planning problem for multiple target surveillance with limited resources of MSAs. The kinematics of the MSA is modeled as a nonholonomic UAV of type Dubins. Based on the fact that the track information of each target degrades over time until it is renewed by the MSAs, the motion-planning problem here is formulated as an optimization problem, whose objective is to minimize the average time period between two consecutive observations of each target. Since the general optimal motion-planning problem for the MMMT case is NP-hard, a computationally efficient suboptimal approach is proposed in this paper.

The motion planning consists of two stages: task decomposition and individual path generation. In task decomposition, the whole task of tracking N targets is divided into M disjoint task assignments, which is achieved by a heuristic method based on K-mean clustering. Then, given a task assignment, each MSA makes its own motion plan, which is an SMMT motion-planning problem. The desired motion plan in the SMMT case is formulated as a time-optimal loop path to traverse the targets. To find such a loop path, a particular family of trajectories is selected to compose a reduced search space, in which we have proved that the optimal trajectory is always contained. Based on that, a gradient-based suboptimal path-generation algorithm is proposed for a mobile agent of type Dubins to traverse a sequence of target points. Meanwhile, a check-and-flip procedure is introduced to reduce the possibility of local minima. Furthermore, a decentralized online replanning approach is also developed to deal with the situation where the targets are moving.

Experiments and simulations have demonstrated the effectiveness and efficiency of the proposed methods. The adoption of the proposed motion-planning method to a real-world test bed consisting of UAVs and ground robots that are under development is part of our ongoing research work.

The major results of this paper are not limited to this application only. The target-based motion-planning scheme can be extended to the cooperative control of multiple MSAs in the general information-gathering scenario, in which one target can be a building, an intersection, a car under surveillance, or a military unit. Meanwhile, the motion-planning algorithm for the SMMT case can be applied to the general multitarget-engagement problem.

On the other hand, the extension of the proposed approach to more complicated situations with heterogeneous sensor agents and heterogeneous targets is open for further research. The discussions on coverage stability in this paper can also be extended to related topics, such as task assessment and high-level sensor resource management.

REFERENCES

[1] M. M. Polycarpou, Y. Yang, and K. M. Passino, "A cooperative search framework for distributed agents," in *Proc. IEEE Int. Symp. Intell. Control*, 2001, pp. 1–6.

[2] M. Baum and K. Passino, "A search-theoretic approach to cooperative control for uninhabited air vehicles," in *Proc. AIAA GNC Conf.*, Aug. 2002, Paper 2002–4589.

[3] R. W. Beard, T. W. McLain, M. A. Goodrich, and E. P. Anderson, "Coordinated target assignment and intercept for unmanned air vehicles," *IEEE Trans. Robot. Autom.*, vol. 18, no. 6, pp. 911–922, Dec. 2002.

[4] S. Rathinam and R. Sengupta, "A safe flight algorithm for unmanned aerial vehicles," in *Proc. IEEE Aerosp. Conf.*, vol. 5, Big Sky, MT, 2004, pp. 3025–3031.

[5] R. Kumar, H. Sawhney, S. Samarasekera, S. Hsu, H. Tao, Y. Guo, K. Hanna, A. Pope, R. Wildes, D. Hirvonen, M. Hansen, and P. Burt, "Aerial video surveillance and exploitation," *Proc. IEEE*, vol. 89, no. 10, pp. 1518–1539, Oct. 2001.

[6] T. W. McLain, P. R. Chandler, and M. Pachter, "A decomposition strategy for optimal coordination of unmanned air vehicles," in *Proc. Amer. Control Conf.*, vol. 1, 2000, pp. 28–30.

[7] P. R. Chandler, M. Pachter, and S. Rasmussen, "UAV cooperative control," in *Proc. Amer. Control Conf.*, vol. 1, 2001, pp. 50–55.

[8] K. E. Nygard, P. R. Chandler, and M. Pachter, "Dynamic network flow optimization models for air vehicle resource allocation," in *Proc. Amer. Control Conf.*, vol. 3, 2001, pp. 1853–1858.

[9] C. Schumacher, P. R. Chandler, and S. R. Rasmussen, "Task allocation for wide area search munitions," in *Proc. Amer. Control Conf.*, vol. 3, 2002, pp. 1917–1922.

[10] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *Amer. J. Math.*, vol. 79, no. 3, pp. 497–516, 1957.

[11] J. A. Reeds and L. A. Shepp, "Optimal paths for a car that goes both forward and backward," *Pacific J. Math.*, vol. 145, no. 2, pp. 367–393, 1990.

[12] G. M. Siouris and A. P. Leros, "Minimum-time intercept guidance for tactical missiles," *Control Theory Adv. Technol.*, vol. 4, no. 2, pp. 251–263, 1988.

[13] P. Soueres and J.-P. Laumond, "Shortest paths synthesis for a car-like robot," *IEEE Trans. Autom. Control*, vol. 41, no. 5, pp. 672–688, May 1996.

[14] A. M. Shkel and V. J. Lumelsky, "Classification of the Dubins set," *Robot. Auton. Syst.*, vol. 34, pp. 179–202, 2001.

[15] G. Desaulniers and F. Soumis, "An efficient algorithm to find a shortest path for a car-like robot," *IEEE Trans. Robot. Autom.*, vol. 11, no. 6, pp. 819–828, Dec. 1995.

[16] H. J. Sussmann and W. Tang, *Shortest Paths for the Reeds-Shepp Car: A Worked Out Example of the Use of Geometric Techniques in Nonlinear Optimal Control*. New Brunswick, NJ: Rutgers Univ. Press, 1991.

[17] J. D. Boissonnat, A. Cerezo, and J. Leblond, "Shortest paths of bounded curvature in the plane," in *Proc. IEEE Int. Conf. Robot. Autom.*, 1992, pp. 2315–2320.

[18] T. L. Song and S. J. Shin, "Time-optimal impact angle control for vertical plan engagement," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 35, no. 2, pp. 738–742, Apr. 1999.

[19] A. Balluchi, A. Bicchi, B. Piccoli, and P. Soueres, "Stability and robustness of optimal synthesis for route tracking by Dubins' vehicle," in *Proc. 39th IEEE Conf. Decision Control*, 2000, pp. 581–586.

[20] G. Yang and V. Kapila, "A dynamic-programming-styled algorithm for time-optimal multi-agent task assignment," in *Proc. 40th IEEE Conf. Decision Control*, vol. 2, 2001, pp. 1959–1964.

[21] ——, "Optimal path planning for unmanned air vehicles with kinematic and tactical constraints," in *Proc. 41st IEEE Conf. Decision Control*, vol. 2, Dec. 2002, pp. 1301–1306.

[22] J.-P. Laumond, *Robot Motion Planning and Control*. Berlin, Germany: Springer-Verlag, 1998.

[23] J. Bellingham, A. Richard, and J. P. How, "Receding horizon control of autonomous aerial vehicle," in *Proc. Amer. Control Conf.*, Anchorage, AK, May 2002, pp. 3741–3746.

[24] J. S. Bellingham, M. Tillerson, M. Alighanbari, and J. P. How, "Cooperative path planning for multiple UAV's in dynamic and uncertain environments," in *Proc. 41st IEEE Int. Conf. Decision Control*, vol. 3, 2002, pp. 2816–2822.

[25] Y. Guo and L. E. Parker, "A distributed and optimal motion planning approach for multiple mobile robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2002, pp. 2619–2626.

[26] A. Bicchi and L. Pallottino, "On optimal cooperative conflict resolution for air traffic management systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 1, no. 4, pp. 212–222, Dec. 2000.

[27] S. M. LaValle and S. A. Hutchinson, "Optimal motion planning for multiple robots having independent goals," *IEEE Trans. Robot. Autom.*, vol. 14, no. 6, pp. 912–925, Dec. 1998.

[28] I. K. Nikolos, K. P. Valavanis, N. C. Tsourveloudis, and A. N. Kostaras, "Evolutionary algorithm-based offline/online path planner for UAV navigation," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 33, no. 6, pp. 898–912, Jun. 2003.

[29] D. A. Castanon, "Approximated dynamic programming for sensor management," in *Proc. IEEE Conf. Decision Control*, San Diego, CA, Dec. 1997, pp. 1202–1207.

[30] D. J. Cool, P. Gmytrasiewicz, and L. B. Holder, "Decision-theoretic cooperative sensor planning," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, no. 10, pp. 892–902, Oct. 1996.

[31] M. Kalandros, L. Y. Pao, and Y. Ho, "Randomization and super-heuristics in choosing sensor sets for target tracking applications," in *Proc. IEEE Conf. Decision Control*, vol. 2, Phoenix, AZ, Dec. 1999, pp. 1803–1808.

[32] P. Vanheeghe, E. Duflos, P. E. Dumont, and V. Nimier, "Sensor management with respect to danger level of targets," in *Proc. IEEE Conf. Decision Control*, vol. 5, Orlando, FL, 2001, pp. 4439–4444.

[33] J. Howlett, M. Goodrich, and T. McLain, "Learning real-time A^* path planner for sensing closely spaced targets from an aircraft," in *Proc. AIAA Guid., Navigat., Control Conf.*, Austin, TX, Aug. 2003, Paper 2003-5338.

[34] L. E. Parker and B. A. Emmons, "Cooperative multi-robot observation of multiple moving targets," in *Proc. IEEE Int. Conf. Robot. Autom.*, 1997, pp. 2082–2089.

[35] B. Jung and G. S. Sukhatme, "A region-based approach for cooperative multi-target tracking in a structured environment," in *Proc. IEEE Int. Conf. Intell. Robot. Syst.*, 2002, pp. 2764–2769.

[36] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Trans. Robot. Autom.*, vol. 20, no. 2, pp. 243–255, Apr. 2004.

[37] R. Morselli and R. Zanasi, "Positioning trajectory generator with non-linear constraints," in *Proc. IEEE Int. Conf. Control Appl.*, Glasgow, U.K., Sep. 2002, pp. 1177–1182.

[38] A. Turnau, M. Szymkat, and A. Korytowski, "Robust near time-optimal trajectory planning by intermediate targets assignment," in *Proc. IEEE Int. Conf. Control Appl.*, Glasgow, U.K., Sep. 2002, pp. 1159–1164.

[39] D. Walker, T. McLain, and J. Howlett, "Cooperative UAV target assignment using distributed calculation of target-task tours," in *Theory and Algorithms for Cooperative Control*, Singapore: World Scientific, 2004.

[40] M. Orlov. (2002) Efficient generation of set partitions. [Online]. Available: http://www.cs.bgu.ac.il/orlovm/papers/partitions.pdf

[41] M. Abramowitz and I. A. Stegun, "Stirling numbers of the second kind," in *Handbook of Mathematical Functions With Formulas, Graphs, and Mathematical Tables*. New York: Dover, 1972, pp. 824–825.

[42] K. G. Murty, "An algorithm for finding all the assignments in order of increasing cost," *Oper. Res.*, vol. 16, pp. 682–687, 1968.

[43] M. L. Miller, H. S. Stone, and I. J. Cox, "Optimizing Murty's ranked assignment method," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 33, no. 3, pp. 851–886, Jul. 1997.

[44] D. Eppstein, "Finding the k shortest paths," *SIAM J. Comput.*, vol. 28, no. 2, pp. 652–673, 1999.

[45] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford, U.K.: Oxford Univ. Press, 1995.

Zhijun Tang received the B.S. and M.S. degrees from Zhejiang University, Hangzhou, China, in 1997 and 2000, respectively. He is currently working toward the Ph.D. degree at The Ohio State University, Columbus.

His research interests include computer vision, target tracking, sensor fusion, and robot motion planning with their applications in intelligent transportation systems and mobile sensor networks.

Ümit Özgüner (S'72–M'75) received the Ph.D. degree from the University of Illinois at Urbana-Champaign, Urbana, in 1975.

He has held research and teaching positions at IBM T. J. Watson Research Center, Yorktown Heights, NY, the University of Toronto, Toronto, ON, Canada, and Istanbul Technical University, Istanbul, Turkey. He has been with The Ohio State University, Columbus, since 1981, where he is presently a Professor of Electrical and Computer Engineering, and holds the TRC Inc. Chair on Intelligent Transportation Systems (ITS). His areas of research interest are ITS, decentralization and autonomy issues in large systems, and applied automotive control. He is the author of over 300 publications which have appeared in journals, as book chapters, and in conference proceedings. He has been an invited plenary speaker on ITS topics in many international meetings in the U.S., Japan, China, India, Turkey, and Italy. The team he coordinated participated successfully in the (U.S. D.O.T.-supported) 1997 Automated Highway System Technology Demonstration in San Diego, CA, where they demonstrated three fully automated cars doing lane-keeping, convoying, and passing using radar- and vision-based guidance. Recently, he has been leading a group developing an autonomous off-road vehicle to participate in the DARPA Grand Challenge to autonomously cross the desert.

Prof. Özgüner was the first President of the IEEE ITS Council in 1999, and was reelected in 2000. He has also served the IEEE Control Society in many positions and was an elected member of its Board of Governors (1999–2001). He participated in the organization of many conferences (including the IFAC Congress), and was the General Chair of the 2002 CDC. He was the Program Chair for the 1997 IEEE ITS Conference, which he helped initiate, and served as the General Chair for the 2003 Intelligent Vehicles Symposium.