Event-Driven Cooperative Receding Horizon Control for Multi-Agent Systems in Uncertain Environments

Yasaman Khazaeni, *Member, IEEE* and Christos G. Cassandras, *Fellow, IEEE*

***Abstract*—We propose an event-driven Cooperative Receding Horizon (CRH) controller to solve maximum reward collection problems (MRCP) where multiple agents cooperate to maximize the total reward collected from a set of targets in a given mission space. In previous work, a CRH controller was developed and in this paper, we overcome several limitations of this controller, including potential instabilities in the agent trajectories and poor performance due to inaccurate estimation of a reward-to-go function. Rewards are non-increasing functions of time and the environment is uncertain with new targets detected by agents at random time instants. The controller sequentially solves optimization problems over a planning horizon and executes the control for a shorter action horizon, where both are defined by certain events associated with new information becoming available. In contrast to the earlier CRH controller, we reduce the originally infinite-dimensional feasible control set to a finite set at each control update event. We prove some properties of this new controller and include simulation results showing its improved performance.**

***Index Terms*—Cooperative control, event-driven control, receding horizon control, model predictive control, multi-agent systems, optimization algorithms, path planning, traveling salesman problem, vehicle routing problem.**

I. Introduction

COOPERATIVE control of multi-agent systems has been the focus of many recent research works, [1]–[3]. These can mostly be described in terms of systems where a set of controllable mobile agents with limited sensing, communication and computational capabilities seeks to achieve objectives defined globally or individually. The uncertain environments where these agents live in, further require the agents to respond to random events. Examples arise in areas of wireless sensor network and robotics in autonomous vehicles control, cooperative classification, mobile robot coordination, rendez-vous problems, task assignment, persistent monitoring, coverage control and consensus problems; see [4]–[12] and references therein.

Manuscript received July 15, 2016; revised September 18, 2016; accepted September 20, 2016. Date of publication October 4, 2016; date of current version March 16, 2018. This work was supported in part by the NSF under Grants CNS-1239021, ECCS-1509084, and IIP-1430145, by AFOSR under Grant FA9550-15-1-0471, and by ONR under Grant N00014-09-1-1051. Recommended by Associate Editor Q. Zhao.

Y. Khazaeni was with the Division of Systems Engineering and Center for Information and Systems Engineering, Boston University, Brookline, MA 02446 USA. She is now with IBM Research, Cambridge, MA 02142 USA (e-mail: yasaman.khazaeni@us.ibm.com).

C. G. Cassandras is with Boston University, Brookline, MA 02446 USA (e-mail: cgc@bu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCNS.2016.2615162

Both centralized or decentralized control approaches are used to solve these problems. The former are usually based on the solution of an optimal control problem in a central station via non-linear programming [13], game theoretic frameworks [14] or semi-definite programming [15]. In the latter case, identification of the information to be shared between agents along with the communication between the agents in order to make collaborative decisions are critical steps in the formulation of a cooperative control problem; see [16]–[19].

In this paper, we consider Maximum Reward Collection Problems (MRCP) where N agents are collecting time-dependent rewards associated with M targets (e.g., data associated with each target) in an uncertain environment. In a deterministic setting with equal target rewards, a one-agent MRCP is an instance of a Traveling Salesman Problem (TSP), [20], [21]. The multi-agent MRCP is similar to the Vehicle Routing Problem (VRP) [22]. These are combinatorial problems for which globally optimal solutions are found through integer programming. For example, in [23], [24], a deterministic MRCP with a linearly decreasing reward model is cast as a dynamic scheduling problem and solved via heuristics. The VRP and Dynamic VRP (DVRP) literature is extensive. In [25], a comprehensive review is provided which points to the computational complexity of these problems. Beyond dynamic programming, various heuristics are described, including methods such as Genetic Algorithms and Ant Colony Systems for different versions of DVRPs. A broad taxonomy of solution approaches may be found in [26]. In [27], a variety of VRPs is considered from a queueing theory point of view and solution algorithms are given that provide some performance guarantees.

Because of the MRCP complexity, it is natural to resort to decomposition techniques. One approach is to seek a functional decomposition that divides the problem into smaller subproblems [28], [29] which may be defined at different levels of the system dynamics. An alternative is a time decomposition where the main idea is to solve a finite horizon optimization problem, then continuously extend this *planning horizon* forward (either periodically or in event-driven fashion), an example of event triggered cooperative control is used for the consensus problem in [30] with centralized and decentralized approaches. This time decomposition is in the spirit of receding horizon techniques used in Model Predictive Control (MPC) to solve optimal control problems for which obtaining infinite horizon feedback control solutions is extremely difficult [31]. In such methods, the current control action is calculated by solving a finite horizon open-loop optimal control problem using the current state of the system as the initial state. At each instant, the optimization yields an optimal control sequence executed over a shorter *action horizon* before the process is repeated. In the context of multi-agent systems, a Cooperative Receding Horizon (CRH) controller was introduced in [32] with the controller steps defined in event-driven fashion (with events dependent on the observed system state) as opposed to being invoked periodically, in time-driven fashion. The method is extended into a graph representation with a switching CRH controller in [33]. A decentralized version of the CRH controller is also introduced in [18]. A key feature of this controller is that it does not attempt to make any explicit agent-to-target assignments, but only to determine headings that at the end of the current planning horizon, place agents at positions such that a total expected reward is maximized. Nonetheless, as shown in [32], a stationary trajectory for each agent is guaranteed under certain conditions, in the sense that an agent trajectory always converges to some target in finite time.

In this paper, we develop and formalize a new centralized CRH controller first introduced in [34]. In particular, we consider MRCPs in uncertain environments where we have no *a priori* information about the appearance of targets. This means targets appear/disappear at random times and a target may have a random initial reward and a random reward decreasing rate. The new CRH controller allows us to overcome several limitations of the controller in [32], including potential instabilities in the agent trajectories and poor performance due to inaccurate estimation of the reward-to-go function. We accomplish this by (*i*) reducing, at each event-driven control evaluation step, the originally infinite-dimensional feasible control set to a finite set and (*ii*) by improving the estimation process for the reward to go, including a new "travel cost factor" for each target which accommodates different target configurations in a mission space. We also establish some properties of this new controller whose overall performance is significantly improved relative to the original one, as illustrated through various simulation examples. Although the work here is focusing on a centralized CRH controller, it has been shown in [18] that a distributed version of [34] may be obtained and ongoing work is intended to do the same for the controller in this paper.

In Section II, the MRCP is formulated and in Section III we place the problem in a broader context of event-driven optimal control. In Sections IV and V the original CRH controller is reviewed and the proposed new controller and some of its properties are established. Section VI analyzes the special case of one agent and two targets while in Section VII simulation examples are presented and future research directions are outlined in the conclusions.

II. Problem Formulation

We consider a MRCP where agents and targets are located in a mission space S. There are M targets defining a set $\mathcal{T} = \{1, \ldots, M\}$ and N agents defining a set $\mathcal{A} = \{1, \ldots, N\}$. The mission space S may have different topological characteristics. In a 2-D Euclidean topology, $S \subset \mathbb{R}^2$ as illustrated in Fig. 1 with a triangle denoting a base, circles are mobile agents and squares are targets. In this case, the distance metric $d(x, y)$ is a simple Euclidean norm such that $d : S \times S \to \mathbb{R}$ is the length

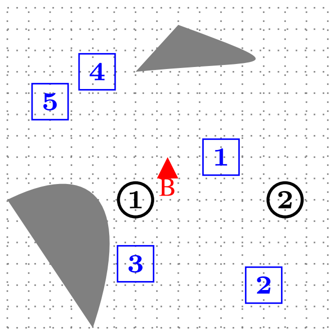

Fig. 1. Sample mission space, gray regions as obstacles.

of the shortest path between points $x, y \in S$. Moreover, the feasible agent headings are given by the set $\mathbf{U}_j(t) = [0, 2\pi]$, $j \in \mathcal{A}$. If there are obstacles in S, the feasible headings and the shortest path between two points should be defined accordingly. Alternatively, the mission space S may be modeled as a graph $\mathcal{G}(E, V)$ with V representing the location of targets and the base. Feasible headings are defined by the (directed) edges at each node and the distance $d(u, v)$ is the sum of the edge weights on the shortest path between u and v. In this paper, we limit ourselves to a Euclidean mission topology. Targets are located at points $\mathbf{y}_i \in S$, $i \in \mathcal{T}$. Target i's reward is denoted by $\lambda_i \phi_i(t)$ where λ_i is the initial maximum reward and $\phi_i(t) \in [0, 1]$ is a non-increasing discount function. By using the appropriate discounting function we can incorporate constraints such as hard or soft deadlines for targets. An example of a discount function is

$$\phi_i(t) = \begin{cases} 1 - \frac{\alpha_i}{D_i} t & \text{if } t \le D_i \\ (1 - \alpha_i) e^{-\beta_i (t - D_i)} & \text{if } t > D_i \end{cases} \tag{1}$$

where α_i, β_i and D_i are given parameters. In this case D_i acts as a "deadline" which if exceeded, results in negligible or no reward. Agents are located at $\mathbf{x}_j(t) \in S$. Each agent has a controllable heading at time t, $u_j(t) \in \mathbf{U}_j(t) = [0\ 2\pi]$. The velocity of agent j is

$$\mathbf{v}_j(t) = V_j [\cos(u_j(t)), \sin(u_j(t))]^\top \tag{2}$$

where we assume that V_j is a fixed speed. This can be relaxed with V_j treated as another control parameter; it will increase the problem complexity, but can be handled by the proposed framework.

We define a *mission* as the process of the N agents cooperatively collecting the maximum possible total reward from M targets within a given mission time T. Upon collecting rewards from all targets, the agents return to a base located at $\mathbf{z} \in S$ and the mission is complete. Events occurring during a mission can be controllable (e.g., collecting a target's reward) or random (e.g., the appearance/disappearance of targets or changes in their location). The event-driven CRH controller we will develop, handles these random events by re-solving the optimal control problem as in the original CRH controller in [32]. In order to ensure that agents collect target rewards in finite time, we assume that each target has a radius $s_i > 0$ and that agent j collects reward i at time t if and only if $d(\mathbf{x}_j(t), \mathbf{y}_i) \le s_i$.

III. An Event-Driven Optimization View

We view the solution of a MRCP as a sequence of headings for all agents and associated heading switching times. We define a policy $\boldsymbol{\pi}$ as a vector $[\mathbf{u}, \xi]$ where $\xi = [\xi_1, \ldots, \xi_K]$ are the switching time intervals over which headings are maintained with $t_{k+1} = \sum_{l=1}^{k} \xi_l$, and $t_1 = 0$. Here, we have defined the switching intervals ξ_k to be part of the control; however, had we assumed a fixed value $\xi_k = \delta$ this would be the fully deterministic time decomposition with time step δ where $t_k = k\delta$. On the other hand, $\mathbf{u} = [\mathbf{u}_1, \ldots, \mathbf{u}_K]$ with $\mathbf{u}_k = [u_1(t_k), \ldots, u_N(t_k)]$ is the vector of all agent headings at time t_k. With M bounded, there exist policies $\boldsymbol{\pi}$ such that all targets are visited over a finite number of switching events. Each switching time t_k is either the result of a controllable event (e.g., visiting a target) or an uncontrollable random event. This is a complex stochastic control problem where the state space Ξ is the set of all possible locations of agents $\mathcal{X}_k = [\mathbf{x}_1(t_k), \ldots, \mathbf{x}_N(t_k)]$ and targets $\mathcal{Y}_k = [\mathbf{y}_1, \ldots, \mathbf{y}_{M_k}]$ along with $\mathcal{T}_k$ defined as the set of unvisited targets at time t_k and $M_k = \|\mathcal{T}_k\|$. As the mission evolves, M_k decreases and the mission is complete when either $M_k = 0$ or a given mission time T is reached. The complete system state at time t_k is $(\mathcal{X}_k, \mathcal{Y}_k) \in \Xi$. The total number of time steps in a mission is K_T which depends on T, the total mission time. We define the optimization problem $\mathbf{P}$ as

$$\max_{\boldsymbol{\pi}} \sum_{k=1}^{K_T} R_{\boldsymbol{\pi}}(t_k, \mathcal{X}_k, \mathcal{Y}_k) \tag{3}$$

where

$$R_{\boldsymbol{\pi}}(t_k, \mathcal{X}_k, \mathcal{Y}_k) = \sum_{i=1}^{M_k} \sum_{j=1}^{N} \lambda_i \phi_i(t_k) \mathbb{1}\{d(\mathbf{x}_j(t_k), \mathbf{y}_i) \leq s_i\}$$

where $\mathbb{1}(\cdot)$ is the usual indicator function. The time a target is visited is a controllable event associated with a heading switching. In a deterministic problem, there is no need to switch headings unless a target is visited, but in an uncertain setting the switching times are not limited to these events. We define a subsequence $\boldsymbol{\tau}^{\boldsymbol{\pi}} = \{\tau_1^{\boldsymbol{\pi}}, \tau_2^{\boldsymbol{\pi}}, \ldots, \tau_M^{\boldsymbol{\pi}}\}$ of $\{t_1, \ldots t_K\}$, $M \leq K$, so that $\tau_i^{\boldsymbol{\pi}}$ is the time target i is visited. Note that $\boldsymbol{\tau}^{\boldsymbol{\pi}}$ is not a monotonic sequence, since targets can be visited in any order. Therefore, (3) can be rewritten as

$$\max_{\boldsymbol{\pi}} \sum_{i=1}^{M} \lambda_i \phi_i(\tau_i^{\boldsymbol{\pi}}). \tag{4}$$

Defining the immediate reward as being collected during a time period ξ_k and the reward-to-go as being aggregated over all $t > t_k + \xi_k$, an optimality equation for this problem is

$$J(t_k, \mathcal{X}_k, \mathcal{Y}_k) = \max_{\mathbf{u}_k, \xi_k} \left[J_I(t_k, \mathcal{X}_k, \mathcal{Y}_k, \mathbf{u}_k, \xi_k) + J(t_{k+1}, \mathcal{X}_{k+1}, \mathcal{Y}_{k+1}) \right] \tag{5}$$

where $J(t_k, \mathcal{X}_k, \mathcal{Y}_k)$ denotes the maximum total reward at time t_k with current state $(\mathcal{X}_k, \mathcal{Y}_k)$ and $J_I(t_k, \mathcal{X}_k, \mathcal{Y}_k, \mathbf{u}_k, \xi_k)$ is the immediate reward collected in the interval $(t_k, t_{k+1}]$. Finally, $J(t_{k+1}, \mathcal{X}_{k+1}, \mathcal{Y}_{k+1})$ is the maximum reward-to-go at t_{k+1} assuming no future uncertainty, i.e., we avoid the use of an *a priori* stochastic model for the environment, opting instead to react to random events by re-solving (5) when this happens. Letting $\tau^* = \max_{i \in \mathcal{T}}\{\tau_i^{\boldsymbol{\pi}}\}$, we set $J(\tau^*, \mathcal{X}_K, \mathcal{Y}_K) = 0$. Henceforth, we write $J(t_k, \mathcal{X}_k, \mathcal{Y}_k) = J(t_k)$ for brevity.

Had we assumed a fixed value for ξ_k *a priori*, the optimization problem (5) could be solved using Dynamic Programming (DP) with the terminal state reached when no target is left in the mission space. However, a fixed ξ_k does not allow for real-time reactions to new events. This fact, along with the size of the state space renders DP impractical and motivates a receding horizon control approach where we set $\xi_k = H_k$ based on a *planning horizon* H_k selected at time step t_k. The selection of this planning horizon will be fully discussed in the next section. A finite horizon optimal control problem over $(t_k, t_k + H_k]$ is solved to determine the optimal control $\mathbf{u}_k^*$. This control is maintained for an *action horizon* $h_k \leq H_k$. A new optimization problem is re-solved at $t_{k+1} = t_k + h_k$ or earlier if any random event is observed. Following (5), the optimization problem $\mathbf{P}_k$ is

$$\max_{\mathbf{u}_k} [J_I(\mathbf{u}_k, t_k, H_k) + J(t_{k+1}, H_{k+1})] \tag{6}$$

where $J(t_{k+1}, H_{k+1})$ and $J_I(\mathbf{u}_k, t_k, H_k)$ were defined above assuming $\xi_k = H_k$. The immediate reward is zero if agents do not visit any target during $(t_k, t_k + H_k]$, otherwise it is the reward collected from the visited targets.

IV. CRH Control Scheme

In this section we briefly review the CRH controller introduced in [32] and identify several limitations of it to motivate the methods we will use to overcome them.

Cooperation Scheme: In [32] the agents divide the mission space into a *dynamic* partition at each mission step. The degree of an agent's responsibility for each target depends on the relative proximity of the agent to the target. A neighbor set is defined for each target which includes its b closest agents, $b = 1, 2, \ldots$, sharing the responsibility for that target until another agent moves closer. A value of $b = 2$ is used in the previous and current work for simplicity. Defining $c_{\mathrm{ij}}(t) = d(\mathbf{y}_i, \mathbf{x}_j(t))$ to be the direct distance between target i and agent j at time t, let $B^l(\mathbf{y}_i, t) \in \{1, \ldots, N\}$ be the lth closest agent to target i at time t. Formally

$$B^l(i, t) = \underset{j \in \mathcal{A}, j \neq B^1(i,t), \ldots, j \neq B^{l-1}(i,t)}{\operatorname{argmin}} \{c_{\mathrm{ij}}(t)\} \tag{7}$$

Let $\beta^b(i, t) = \{B^1(i, t), \ldots, B^b(i, t)\}$ be a neighbor set based on which a *relative distance* function is defined for all $j \in \mathcal{A}$

$$\delta_{\mathrm{ij}}(t) = \begin{cases} \dfrac{c_{\mathrm{ij}}(t)}{\sum_{k \in \beta^b(i,t)} c_{\mathrm{ik}}(t)} & \text{if } j \in \beta^b(i, t); \\ 1 & \text{otherwise} \end{cases}. \tag{8}$$

Obviously, if $j \notin \beta^b(i, t)$, then $\delta_{\mathrm{ij}}(t) = 1$. The *relative proximity function* $p(\delta_{\mathrm{ij}}(t))$ defined in [32] is as follows:

$$p(\delta_{\mathrm{ij}}(t)) = \begin{cases} 1, & \text{if } \delta \leq \Delta \\ \frac{1-\Delta-\delta}{1-2\Delta}, & \text{if } \Delta \leq \delta \leq 1 - \Delta \\ 0, & \text{if } \delta > 1 - \Delta \end{cases} \tag{9}$$

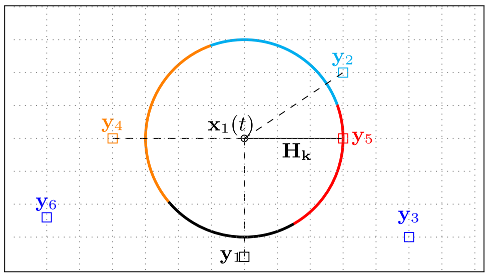

Fig. 2. The Active Target Set for agent 1: $S_1(\mathbf{x}_1(t_k), H_k) = \{1, 2, 4, 5\}$.

and is viewed as the probability that target i will be visited by agent j. Here, $\Delta \in [0, \frac{1}{2})$ defines the level of cooperation between the agents. By increasing Δ, an agent will take full responsibility for more targets, hence less cooperation. Each agent takes on full responsibility for target i if $\delta_{\rm ij}(t) \leq \Delta$. As shown in [32], this generalizes the Voronoi tessellations of the mission space where the borders of the partitions will be determined based on the value of Δ. When $\Delta = \frac{1}{2}$ the regions converge to the Voronoi tessellation of the mission space, with the location of agents at the centers of the Voronoi tiles. There is no cooperation region in this case and each agent is fully responsible for the targets within its own Voronoi tile. On the other hand, when $\Delta = 0$ no matter how close an agent is to a target, the two agents are still responsible for that target.

Planning and Action Horizons: In [32], H_k is defined as the earliest time of an event such that one of the agents can visit one of the targets:

$$H_k = \min_{l \in \mathcal{T}_k} \left\{ \frac{d(\mathbf{x}_j(t_k), \mathbf{y}_l)}{V_j} \right\}. \tag{10}$$

This definition of *planning horizon* for the CRH controller ensures no controllable event can take place during this horizon. It also ensures that re-evaluation of the CRH control is *event-driven*, as opposed to being specified by a clock which involves a tedious synchronization over the networked agents. Fig. 2 illustrates how H_k is determined when $V_j = 1$. The CRH control calculated at t_k is maintained for an *action horizon* $h_k \leq H_k$. In [32] h_k is defined either (*i*) through a random event that may be observed at $t_e \in (t_k, t_k + H_k]$ so that $h_k = t_e - t_k$, or (*ii*) as $h_k = \gamma H_k$, $\gamma \in (0, 1)$. It is also shown in [32] that under (10) the CRH controller generates a stationary trajectory for each agent in the sense that an agent trajectory always converges to some target in finite time, under the assumption that all minima of a an introduced potential function are at the target locations.

IV. Original CRH Controller Limitations

1. Instabilities in Agent Trajectories: The optimization problem considered in [32] uses a potential function which is minimized in order to maximize the total reward. If the assumption mentioned above (i.e., all minima of the potential function are at the target locations) does not hold, the stationary trajectory guarantee cannot be maintained. Due to this, the agents are directed toward the weighted center of gravity of all targets. This can specifically happen in missions where targets attain a symmetric configuration, leading to oscillatory behavior in the agent trajectories. An example is shown in Fig. 7(a) with the original CRH controller applied to a single agent, resulting in oscillations between three targets with equal rewards. This problem was addressed in [18] by introducing a monotonically increasing cost factor (or penalty) $C(u_j)$ on the heading u_j. While this prevents some of the instabilities, it has to be appropriately tuned for each mission. We show how to overcome this problem in Section V.

2. Hedging and Mission Time: The agent trajectories in [32] are specifically designed to direct them to positions close to targets but not exactly towards them unless they are within a certain "capture distance". The motivation is that no agent should be committed to a target until the latest possible time so as to hedge against the uncertainty of new, potentially more attractive, randomly appearing targets. This hedging effect is helpful in handling such uncertainties, but it can create excessive loss of time, especially when rewards are declining fast. This can be addressed by more direct movements towards targets, while also re-evaluating the control frequently enough. The feasible control set in the original CRH is the continuous set $[0, 2\pi]^N$. By appropriately reducing this to a discrete set of control values we will show how we can eliminate unnecessary hedging. This also reduces the complexity of the optimal control problem at each time step and facilitates the problem solution over a finite number of evaluations. This reduction in complexity is achieved with no loss of optimality in the objective function that will be defined in the next section.

3. Estimation of Reward-to-Go: In the original CRH control scheme, a target visit time is estimated as the earliest time any agent j would reach some target i, given a control $\mathbf{u}_k$ at time t_k and maintained over $(t_k, t_k + H_k]$. Thus, the estimated visit time $\tilde{\tau}_{\rm lj}(\mathbf{u}_k, t_k, H_k)$ for any $l \in \mathcal{T}_k$ is

$$\tilde{\tau}_{\rm lj}(\mathbf{u}_k, t_k, H_k) = t_k + H_k + d(\mathbf{x}_j(t_k + H_k, u_j(t_k)), \mathbf{y}_l)$$

where $\mathbf{x}_j(t_k + H_k, u_j(t_k))$ is the location of agent j in the next time step given the control $u_j(t_k)$. This is a *lower bound* for $\tilde{\tau}_{\rm ij}$ which is feasible only when $M_k \leq N$, leading also to a mostly unattainable upper bound for the total reward. We will show how this estimate is improved by a more accurate projection of each agent's future trajectory.

V. The New CRH Controller

In this section, we present a new version of the CRH controller in [32]. Using the agent location $\mathbf{x}_j(t_k + H_k, u_j(t_k))$ and assuming $V_j = 1$ for all agents, the feasible set for $\mathbf{x}_j(t_k + H_k, u_j(t_k))$ is defined as

$$\mathcal{F}_j(t_k, H_k) = \{\mathbf{w} \in S : \; d(\mathbf{w}, \mathbf{x}_j(t_k)) = H_k\}. \tag{11}$$

In a Euclidean mission space with no obstacles, $\mathcal{F}_j(t_k, H_k)$ is the circle centered at $\mathbf{x}_j(t_k)$ with radius H_k (see Fig. 2). We define $q_i(\mathbf{x}_j(t))$ as the indicator function capturing whether agent j visits target i at time t

$$q_i(\mathbf{x}_j(t)) = \mathbb{1}\{d(\mathbf{x}_j(t), \mathbf{y}_i) \leq s_i\}. \tag{12}$$

We then define the immediate reward at t_k:

$$J_{\mathbf{I}}(\mathbf{u}_k, t_k, H_k) = \sum_{j=1}^{N} \sum_{l=1}^{M_k} \lambda_l \phi_l(t_k + H_k) \cdot q_l(\mathbf{x}_j(t_k + H_k, u_j(t_k))). \tag{13}$$

Following the definition of τ_i as the visit time of target i used in (4), we define $\tilde{\tau}_{\text{ij}}$ as the estimated visit time of target i by agent j. Here, $\tilde{\tau}_{\text{ij}} > t_k$ and any of the agents in the mission space has a chance to visit target i. At time t_k we define an estimate of the reward-to-go $J(t_{k+1}, H_{k+1})$ under control $\mathbf{u}_k$ as

$$\tilde{J}(\mathbf{u}_k, t_{k+1}, H_{k+1}) = \sum_{j=1}^{N} \sum_{l=1}^{M_{k+1}} \lambda_l \phi_l(\tilde{\tau}_{\text{lj}}(\mathbf{u}_k, t_k, H_k)) \cdot q_l(\mathbf{x}_j(\tilde{\tau}_{\text{lj}}(\mathbf{u}_k, t_k, H_k))). \tag{14}$$

We previously mentioned that the original CRH control approach used a lower bound for estimating $\tilde{\tau}_{\text{ij}}$. We improve this estimate and at the same time address the other two limitations presented above through three modifications: (*i*) We introduce a new *travel cost* for each target, which combines the distance of a target from an agent, its reward, and a "local sparsity factor". (*ii*) We introduce an *active target set* associated with each agent at every control evaluation instant t_k. This allows us to reduce the infinite dimensional feasible control set at t_k to a finite set. (*iii*) We introduce a new event-driven action horizon h_k which makes use of the active target set definition. With these three modifications, we finally present a new CRH control scheme based on a process of looking ahead over a number of CRH control steps and aggregating the remainder of a mission through a reward-to-go estimation process.

1) Travel Cost Factor

At each control iteration instant t_k, we define $\zeta_i(t_k)$ for target i to measure the sparsity of rewards in its vicinity as follows. Let $\bar{D}_i > 0$ be the minimum time such that $\phi_i(\bar{D}_i) = 0$ for each $i \in \mathcal{T}$ and set $D_i = \min(\bar{D}_i, T)$. Thus, the average reward decreasing rate of i over the mission is given by λ_i / D_i. Let the set $\{1, 2, \ldots, I_k\}$ contain the indices of the I_k closest targets to i at time t_k. We then define the *sparsity factor* for target i as

$$\zeta_i(t_k) = \sum_{l=1}^{I_k} \gamma^l \frac{d(\mathbf{y}_i, \mathbf{y}_l)}{\lambda_l / D_l}. \tag{15}$$

The parameter I_k is chosen based on the number of targets in the mission space at time t_k and the computation capacity of the controller. In addition $\gamma \in [0\ 1]$ is used to shift the weight among the I_k targets. In (15), $\zeta_i(t_k)$ captures the effect of target clusters in the mission space. In topologies with a large number of targets in one cluster, higher values of γ produce a larger force attracting agents towards such clusters. On the other hand, if $\gamma \to 1$ and if I_k includes all available targets, then (15) becomes the sum of weighted distances between target pairs. In the limit $\gamma = 1$, this results in similar sparsity factors for all the targets.

Note that $\zeta_i(t_k)$ is time-dependent since the set of I_k closest targets changes over time as rewards are collected. A larger $\zeta_i(t_k)$ implies that target i is located in a relatively sparse area and vice versa. The main idea for $\zeta_i(t_k)$ comes from [35] where it was used to solve TSP problems with clustering.

Next, for any point in $\mathbf{x} \in S$, we define target i's travel cost at time t_k as

$$\eta_i(\mathbf{x}, t_k) = \frac{d(\mathbf{x}, \mathbf{y}_i)}{\lambda_i / D_i} + \zeta_i(t_k). \tag{16}$$

The travel cost is proportional to the distance metric, so the farther a target is from $\mathbf{x}$ the more costly is the visit to that target. It is inversely proportional to the reward's average decreasing rate, implying that the faster the reward decreases, the less the travel cost is. Adding $\zeta_i(t_k)$ gives a target in a sparse area a higher travel cost as opposed to one where there is an opportunity for a visiting agent to collect additional rewards from its vicinity.

2) Active Targets

At each control iteration instant t_k, we define for each agent j a subset of targets with the following property relative to the planning horizon H_k:

$$S_j(t_k, H_k) = \big\{\ell : \exists\, \mathbf{x} \in \mathcal{F}_j(t_k, H_k) \text{ s.t. } \ell = \operatorname*{argmin}_{i \in \mathcal{T}_k} \eta_i(\mathbf{x}, t_k + H_k),\ i = 1, 2, \ldots, M_k\big\}. \tag{17}$$

This is termed the *active target set* and (17) implies that $i \in \mathcal{T}_k$ is an active target for agent j if and only if it has the smallest travel cost using (16) from at least one point in the reachable set $\mathcal{F}_j(t_k, H_k)$. This means that every $\mathbf{x} \in \mathcal{F}_j(t_k, H_k)$ is associated with one of the active targets and, therefore, so does every feasible heading $u_j(t_k)$, which corresponds to active target l if and only if

$$l = \operatorname*{argmin}_{i \in \mathcal{T}_k} \eta_i\big(\mathbf{x}(t_k + H_k, u_j(t_k)), t_k + H_k\big). \tag{18}$$

When $d(x, y)$ is the 2-D Euclidean norm, active targets partition the reachable set $\mathcal{F}_j(t_k, H_k)$ into several arcs as illustrated in Fig. 2 where, for simplicity, we assume $\gamma = 0$ in (15) and all λ_i and $\phi_i(t)$, $i = 1, \ldots, M$ are the same. In this case, agent 1 has four active targets $S_1(t_k, H_k) = \{1, 2, 4, 5\}$ each of them corresponding to one of the arcs shown in Fig. 2. Consequently, the common feature of all points on an arc is that they correspond to the same active target with the least travel cost.

Construction of $S_j(t_k, H_k)$: For each target $l \in \mathcal{T}_k$ and each agent j, let $\mathcal{L}_k(\mathbf{x}_j(t_k), \mathbf{y}_l)$ be the set of points $\mathbf{x} \in S$ defining the shortest path from $\mathbf{x}_j(t_k)$ to $\mathbf{y}_l$. The intersection of this set with $\mathcal{F}_j(t_k, H_k)$ is the set of closest points to target l in the feasible set

$$\mathcal{C}_{l,j}(t_k, H_k) = \mathcal{L}_k(\mathbf{x}_j(t_k), \mathbf{y}_l) \cap \mathcal{F}_j(t_k, H_k). \tag{19}$$

In a Euclidean mission space, $\mathcal{L}_k(\mathbf{x}_j(t_k), \mathbf{y}_l)$ is a convex combination (line segment) of $\mathbf{x}_j(t_k)$ and $\mathbf{y}_l$, while $\mathcal{C}_{l,j}(t_k, H_k)$ is a single point where this line crosses the circle $\mathcal{F}_j(t_k, H_k)$ (see Fig. 2). The following lemma provides a necessary and sufficient condition for identifying targets which are active for an agent at t_k using $\mathcal{C}_{l,j}(t_k, H_k)$.

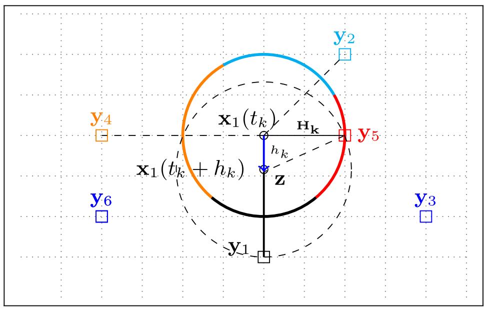

Fig. 3. Multiple-immediate-target event happens with agent at equal distance to targets 1 and 5.

Lemma 1: Target l is an active target for agent j at time t_k if and only if, $\forall i \in \mathcal{T}_k$

$$\eta_l(\mathcal{C}_{l,j}(t_k, H_k), t_{k+1}) \leq \eta_i(\mathcal{C}_{l,j}(t_k, H_k), t_{k+1}). \tag{20}$$

Proof: See Appendix. ■

3) Action Horizon

As mentioned in Section IV, h_k in [32] is defined either (*i*) through a random event that may be observed at $t_e \in (t_k, t_k + H_k]$ so that $h_k = t_e - t_k$, or (*ii*) as $h_k = \gamma H_k$, $\gamma \in (0, 1)$. This definition requires frequent iterations of the optimization problem through which $\mathbf{u}_k^*$ is determined in case no random event is observed to justify such action. Instead, when there are no random events, we define a new *multiple-immediate-target event* to occur when the minimization in (10) returns more than one target, i.e., the agent is at an equal distance from at least two targets. This is illustrated in Fig. 3 where the agent is moving toward target 1 and at point $\mathbf{z}$ it is equidistant to targets 1 and 5. In this case, we define $h_k = \|\mathbf{z} - \mathbf{x}_1(t_k)\|$ and the problem is re-solved at $t_k + h_k$. In general, we define h_k to be the shortest time until the first multiple-immediate-target event occurs in $(t_k, t_k + H_k]$

$$h_k = \min\Big\{H_k, \inf\big\{t > 0 : \exists l, l^* \in \mathcal{T}_k \text{ s.t.} \tag{21}$$
$$d(\mathbf{x}_j(t_k + t, u_j(t_k)), \mathbf{y}_l) = d(\mathbf{x}_j(t_k + t, u_j(t_k)), \mathbf{y}_{l^*})\big\}\Big\}.$$

Consequently, this definition of h_k eliminates any unnecessary control re-evaluation.

3) Look Ahead and Aggregate Processes

In order to solve the optimization problem $\mathbf{P}_k$ in (6) using the CRH approach, we need the estimated visit time $\tilde{\tau}_{\text{lj}}(\mathbf{u}_k, t_k, H_k)$ for each $\mathbf{u}_k$ through which $\tilde{J}(\mathbf{u}_k, t_{k+1}, H_{k+1})$ in (14) can be evaluated. This estimate is obtained by using a projected path for each agent. This path projection consists of a *look ahead* step and an *aggregate* step. In the first step, the active target set $S_j(t_k, H_k)$ in (17) is determined for each agent j. The remaining targets are partitioned using the relative proximity function in (9). We denote the target subset for agent j as $\mathcal{T}_{k,j}$ where

$$T_{k,j} = \big\{l : p(\delta_{\text{lj}}(t_k)) > p(\delta_{\text{lq}}(t_k)),\ \forall q \in \mathcal{A}\big\}. \tag{22}$$

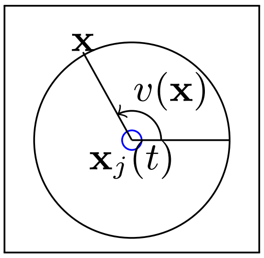

Fig. 4. Agent's heading in a Euclidean mission space.

Let $|\mathcal{T}_{k,j}| = M_{k,j}$. All $\tilde{\tau}_{\text{lj}}(\mathbf{u}_k, t_k, H_k)$ are estimated so that j visits targets in its own subset starting with the one with the least travel cost first. We define the agent j's tour as the permutation $\boldsymbol{\theta}^j(\mathbf{u}_k, t_k, H_k)$ specifying the order in which it visits targets in $\mathcal{T}_{k,j}$. For simplicity, we write $\boldsymbol{\theta}^j$ and let $\boldsymbol{\theta}_i^j$ denote the i^{th} target in agent j's tour. Then, for all $l \in \mathcal{T}_{k,j}$ and $t_{k+1} = t_k + H_k$

$$\eta_{\boldsymbol{\theta}_1^j}(\mathbf{x}_j(t_{k+1}, u_j(t_k)), t_{k+1}) \leq \eta_l(\mathbf{x}_j(t_{k+1}, u_j(t_k)), t_{k+1})$$

and with $n = 1, \ldots, M_{k,j} - 1$, for all $l \in \mathcal{T}_{k,j} - \{\theta_1^j, \ldots, \theta_n^j\}$

$$\eta_{\boldsymbol{\theta}_{n+1}^j}(\mathbf{y}_{\boldsymbol{\theta}_n^j}, \tilde{\tau}_{\boldsymbol{\theta}_n^j}(\mathbf{u}_k, t_k, H_k)) \leq \eta_l(\mathbf{y}_{\boldsymbol{\theta}_n^j}, \tilde{\tau}_{\boldsymbol{\theta}_n^j}(\mathbf{u}_k, t_k, H_k))$$

where

$$\tilde{\tau}_{\boldsymbol{\theta}_n^j}(\mathbf{u}_k, t_k, H_k) = t_k + H_k + \sum_{i=1}^{n-1} d(\mathbf{y}_{\boldsymbol{\theta}_i^j}, \mathbf{y}_{\boldsymbol{\theta}_{i+1}^j}). \tag{23}$$

This results in the corresponding $\tilde{\tau}_{\text{lj}}(\mathbf{u}_k, t_k, H_k)$ for all $l \in \mathcal{T}_{k,j}$. We can now obtain the reward-to-go estimate as

$$J_{\mathbf{A}}(\mathbf{u}_k, t_k, H_k) = \sum_{j=1}^{N} \sum_{l=1}^{M_{k+1,j}} \lambda_l \phi_l(\tilde{\tau}_{\text{lj}}(\mathbf{u}_k, t_k, H_k))$$
$$\cdot\, q_l(\mathbf{x}_j(\tilde{\tau}_{\text{lj}}(\mathbf{u}_k, t_k, H_k))) \tag{24}$$

where $q_l(\cdot)$ is the indicator function defined in (12). Recalling the immediate reward in (13), the optimization problem $\mathbf{P}_k$ becomes

$$\max_{\mathbf{u}_k \in [0\ 2\pi]^N} \big[J_{\mathbf{I}}(\mathbf{u}_k, t_k, H_k) + J_{\mathbf{A}}(\mathbf{u}_k, t_k, H_k)\big]. \tag{25}$$

Next, we will present a theorem that enables us to reduce the infinite feasible set $[0, 2\pi]^N$ to a finite set of headings for each agent. We do so by proving in Lemma 2 that the optimal heading for an agent in a single-agent mission is to go toward an active target in each time step. Then, Theorem 1 proves that the same holds in a multi-agent MRCP mission. In (11), we defined the feasible set $\mathcal{F}_j(t_k, H_k)$ for the location of agent j in the next step $t_{k+1} = t_k + H_k$. In a Euclidean mission space, each point $\mathbf{x} \in \mathcal{F}_j(t_k, H_k)$ corresponds to a heading $v(\mathbf{x}) \in [0, 2\pi]$ relative to the agent's location $\mathbf{x}_j(t_k)$. This is illustrated in Fig. 4. Recalling (19), $\mathcal{C}_{l,j}(t_k, H_k)$ is the closest feasible point to target l. We then define the set containing the headings that correspond to these feasible points for all the active targets of agent j

$$\mathcal{V}_j(t_k, H_k) = \big\{v(\mathbf{x}) : \mathbf{x} = \mathcal{C}_{l,j}(t_k, H_k),\ l \in S_j(t_k, H_k)\big\}.$$

Then, for all agents at time t_k, we define the set $\mathbf{V}_k$ as follows:

$$\mathbf{V}_k = \mathcal{V}_1(t_k, H_k) \times \mathcal{V}_2(t_k, H_k) \times \cdots \times \mathcal{V}_N(t_k, H_k).$$

In the next lemma, we prove that in a single-agent mission with the objective function defined in (25) the optimal control is $u_1(t_k) = v(\mathcal{C}_{l,1}(t_k, H_k)) \in [0, 2\pi]$ for some $l \in S_1(t_k, H_k)$.

Lemma 2: In a single agent $(N = 1)$ mission, if u_1^* is an optimal solution to the problem

$$\max_{u_k \in [0\ 2\pi]} \left[J_{\mathbf{I}}(u_k, t_k, H_k) + J_{\mathbf{A}}(u_k, t_k, H_k)\right] \tag{26}$$

then $u_1^* \in \mathcal{V}_1(t_k, H_k)$

Proof: See Appendix. ∎

The implication of this lemma is that we can reduce the set of feasible controls $[0, 2\pi]$ to a finite set defined by active targets as illustrated in Fig. 2. In this case the set $[0, 2\pi]$ is divided into four arcs each corresponding to one active target. The finite set of feasible controls are then the four direct directions toward the active targets.

Theorem 1: In a multi-agent MRCP mission, if $\mathbf{u}^* = [u_1^*, \ldots, u_N^*]$ is the optimal solution to the problem in (25) then $\mathbf{u}^* \in \mathbf{V}_k$.

Proof: See Appendix ∎

Theorem 1 reduces the problem $\mathbf{P}_k$ to a maximization problem over a finite set $\mathbf{V}_k$ of feasible controls

$$\max_{\mathbf{u}_k \in \mathbf{V}_k} \left[J_{\mathbf{I}}(\mathbf{u}_k, t_k, H_k) + J_{\mathbf{A}}(\mathbf{u}_k, t_k, H_k)\right]. \tag{27}$$

The significance of this is that the aggregation process not only improves the performance of the CRH control, but also reduces the size of the problem compared to the original CRH controller in [32]. Having the active target set, an agent will always eventually reach a point where the active target set is reduced to only one active target, which the agents would then move towards. This resolves the issue of stationarity in [32] and guarantees that all targets are visited by agents (unless of course their deadlines are missed in the case of an infeasible problem).

The following algorithm generates controls in this manner at each step t_k and is referred to as the "One-step Lookahead" henceforth abbreviated as 1-L (extended to a "K-step Lookahead" algorithm in what follows).

CRH 1-L Algorithm.

```
1) Determine $H_k$ through (10).
2) Determine the active target set $S_j(t_k, H_k)$ through (17)
   for all $j \in \mathcal{A}$.
3) Evaluate $J_{\mathbf{I}}$ and $J_{\mathbf{A}}$ for all $\mathbf{u}_k \in \mathbf{V}_k$ through (13)
   and (24)
4) Solve $\mathbf{P}_k$ in (27) and determine $\mathbf{u}_k^*$.
5) Evaluate $h_k$ through (21)
6) Execute $\mathbf{u}_k^*$ over $(t_k, t_k + h_k]$ and repeat Step 1 with
   $t_{k+1} = t_k + h_k$.
```

The 1-L CRH controller can be extended to a K-step Lookahead, henceforth referenced to as K-L controller with $K > 1$, by exploring additional possible future paths for each agent at each time step t_k. In the 1-L algorithm, the optimal reward-to-go is estimated based on a single tour over the remaining targets. The K-L algorithm estimates this reward by considering more possible tours for each agent as follows. For any feasible $u_j(t_k) \in \mathcal{V}_j(t_k, H_k)$ the agent is hypothetically placed at the corresponding next step location $\mathbf{x}_j(t_{k+1})$. This is done for all agents to maintain synchronicity of the solution. At $\mathbf{x}_j(t_{k+1})$, a new active target set is determined, implying that agent j can have $|S_j(t_k + H_k, H_{k+1})|$ possible paths. At this point, we can repeat the same procedure by hypothetically moving the agent to a new feasible location from the set $\mathcal{F}_j(t_{k+1}, H_{k+1})$ or we can stop and estimate the reward-to-go for each available path. Thus, for a 2-L controller, problem $\mathbf{P}_k$ becomes:

$$\max_{\mathbf{u}_k \in \mathbf{V}_k} \Bigg[J_{\mathbf{I}}(\mathbf{u}_k, t_k, H_k) + \max_{\mathbf{u}_{k+1} \in \mathbf{V}_{k+1}} \big[J_{\mathbf{I}}(\mathbf{u}_{k+1}, t_{k+1}, H_{k+1}) + J_{\mathbf{A}}(\mathbf{u}_{k+1}, t_{k+1}, H_{k+1})\big]\Bigg] \tag{28}$$

We extend the previous algorithm to the K-L controller, as shown below. For a K-L algorithm the optimization problem $\mathbf{P}_k$ includes a K-fold maximization as an extension of the two-fold optimization in (28).

CRH K-L Algorithm.

```
1) Determine $H_k$ through (10).
2) Determine the active target set $S_j(t_k, H_k)$ through (17)
   for all $j \in \mathcal{A}$.
3) Evaluate $J_I$ for all $\mathbf{u}_k \in \mathbf{V}_k$ through (13)
4) For each $\mathbf{u}_k \in \mathbf{V}_k$, Step forward to $t_{k+1} = t_k + H_k$
   and $\mathbf{x}_j(t_{k+1}, u_j(t_k))$ for all $j \in \mathcal{A}$.
5) For $K$ times repeat steps 1 to 4 for the new $t_{k+1}$ for all
   $\mathbf{u}_{k+1} \in \mathbf{V}_{k+1}$.
6) Evaluate $J_{\mathbf{A}}(\mathbf{u}_{k+K}, t_{k+K}, H_{k+K})$ for all
   $\mathbf{u}_{k+K} \in \mathbf{V}_{k+K}$ through (23) and (24)
7) Solve the $K$-L optimization problem $\mathbf{P}_k$ and
   determine $\mathbf{u}_k^*$.
8) Evaluate $h_k$ through (21)
9) Execute $\mathbf{u}_k^*$ over $(t_k, t_k + h_k]$ and repeat Step 1 with
   $t_{k+1} = t_k + h_k$.
```

This procedure can easily be repeated and the whole process can be represented as a tree structure where the root is the initial location of the agent and a path from the root to each leaf is a possible target sequence for the agent. In Fig. 5(a) a sample mission with 5 targets is shown with its corresponding tree in Fig. 5(b). A brute-force method involves $5! = 120$ possible paths, whereas the tree structure for this mission is limited to 11 paths. The active target set for agent 1 consists of targets 1, 2. Each of these active targets would then generate several branches in the tree, as shown. We calculate the total reward for each path to find the optimal one. Determining the complete tree for large K is time consuming. The K-L CRH controller

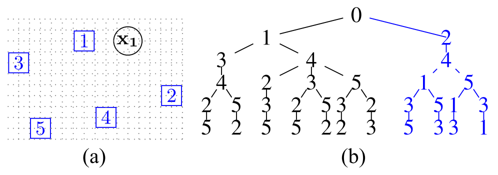

Fig. 5. (a) Five-target mission. (b) Tree structure.

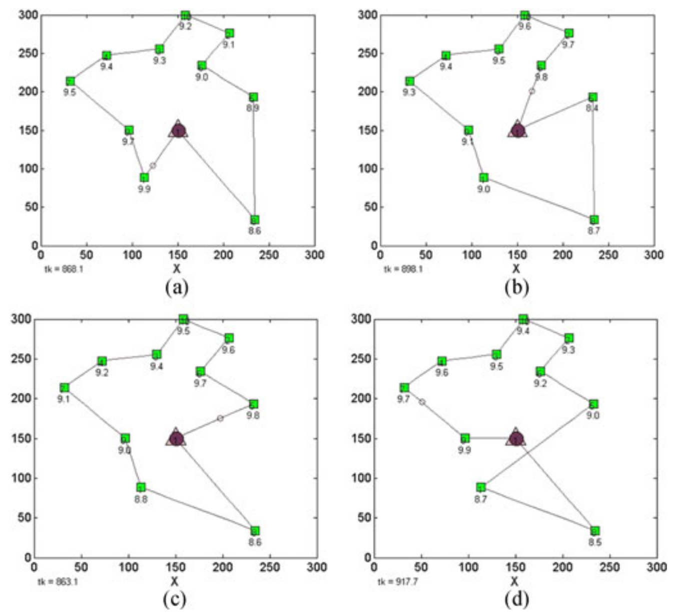

Fig. 6. Ten-target deterministic mission with different number of look ahead steps. (a) 1-L: $[1-9-7-4-3-10-2-6-5-8]$-Reward $= 92.6683$-Time $= 868$. (b) 3-L: $[6-2-10-3-4-7-9-1-8-5]$-Reward $= 92.5253$-Time $= 897$. (c) 5-L: $[5-6-2-10-3-4-7-9-1-8]$-Reward $= 92.6031$-Time $= 862$. (d) 6-L: $[9-7-4-3-10-2-6-5-1-8]$-Reward $= 92.7436$-Time $= 916$.

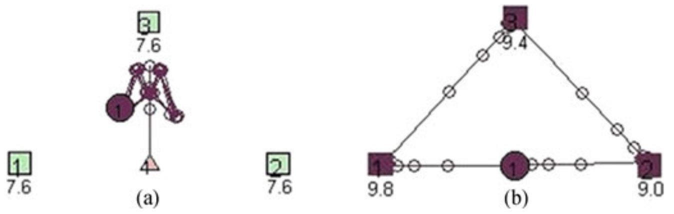

Fig. 7. Comparison of the two CRH controllers for a 3-target mission. (a) Original CRH oscillation. (b) New CRH optimal solution.

enables us to investigate the tree down to a few levels and then calculate an estimated reward-to-go for the rest of the selected path. However, there is no guarantee on the monotonicity of the results with more lookahead steps and, in some cases, the final result degrades with one more lookahead step. We illustrate this point through a counterexample with a single agent case and ten equally important targets (see Fig. 6 where the reward for each target at the time of the collection is shown below it). This is a straightforward TSP for which the optimal path can be obtained through an exhaustive search. For this case, the 1-L and 2-L CRH controllers find the same path with a reward of 92.6683. However, once we move up to 3-L, the CRH controller degrades to a worse path with a reward of 92.5253 reward, [see Figs. 6(a) and (b)]. The optimal path calculated through exhaustive search is obtained by the 6-L CRH controller [Fig. 6(d)]. The observation is that the non-monotonicity in the number of look ahead steps is a local effect and once we increase the number of such steps K, the controller can determine the optimal solution. This obviously is not always the case in deterministic missions.

3) *Complexity of the Algorithm*

In [32], it was shown that the complexity of the optimization algorithm solved at each receding horizon iteration is $O(G^N)$ where N is the number of agents and G is the resolution (i.e., discretization level) used to define agent headings over $[0\ 2\pi]$ (typically, $G = 16$ or 32). In the new approach, since the feasible control set is a finite set defined by the active targets, the complexity is $O(A^N)$ where $A \leq M$ (number of targets) is the maximum number of active targets. Observe that, in general, $A \ll G$ since the number of active targets is not likely to include every heading in a discretized $[0\ 2\pi]$ set, unless the targets have a very special topology. Moreover, A decreases as targets are visited. The value of A depends on determining the active target set $S_j(t_k, H_k)$ in (17) for agent j (so that $A = |S_j(t_k, H_k)|$) at each step. This is a sorting process over M targets which is of $O(M \log M)$. On the other hand, active target sets have to be determined for all K lookahead steps and this computation is of exponential complexity in K. Selecting K is topology-dependent and K can be tuned for any mission. Note, however, that in missions with uncertainties a small value of K is sufficient since the topology of the mission space can change at any time.

VI. Two-Target, One-Agent Case

The simplest case of the MRCP is that of one agent and two targets. Obviously, this is an easy routing problem whose solution is one of the two possible paths the agent can take. We prove that the 1-L algorithm solves the problem with any linearly decreasing reward function. Consider a mission with one agent and two targets with initial rewards and deadlines λ_1, D_1 and λ_2, D_2, respectively. The analytical solution for this case reveals whether path $\theta_1 = (1,2)$ or $\theta_2 = (2,1)$ is optimal. Following the previous analysis, we assume that $V_1 = 1$ and set $\mathbf{x}_1(t_k) = \mathbf{x}$ for the sake of brevity. We also assume the rewards are linearly decreasing to zero: $\phi_i(t) = 1 - \frac{t}{D_i}$. The two possible rewards are given by

$$\begin{aligned} R_{(1,2)} =\ & \lambda_1 \left[1 - \frac{d(\mathbf{x}, \mathbf{y}_1)}{D_1}\right] \\ & + \lambda_2 \left[1 - \frac{d(\mathbf{x}, \mathbf{y}_1) + d(\mathbf{y}_1 - \mathbf{y}_2)}{D_2}\right] \end{aligned} \tag{29}$$

$$\begin{aligned} R_{(2,1)} =\ & \lambda_2 \left[1 - \frac{d(\mathbf{x}, \mathbf{y}_2)}{D_2}\right] \\ & + \lambda_1 \left[1 - \frac{d(\mathbf{x}, \mathbf{y}_2) + d(\mathbf{y}_2 - \mathbf{y}_1)}{D_1}\right]. \end{aligned} \tag{30}$$

Therefore, if $R_{(1,2)} > R_{(2,1)}$, it follows that the following inequality must hold:

$$\begin{aligned} & \frac{\lambda_1}{D_1}\left[d(\mathbf{x}, \mathbf{y}_1) - d(\mathbf{x}, \mathbf{y}_2) + d(\mathbf{y}_2, \mathbf{y}_1)\right] \\ & \quad < \frac{\lambda_2}{D_2}\left[d(\mathbf{x}, \mathbf{y}_2) - d(\mathbf{x}, \mathbf{y}_1) + d(\mathbf{y}_1, \mathbf{y}_2)\right] \end{aligned} \tag{31}$$

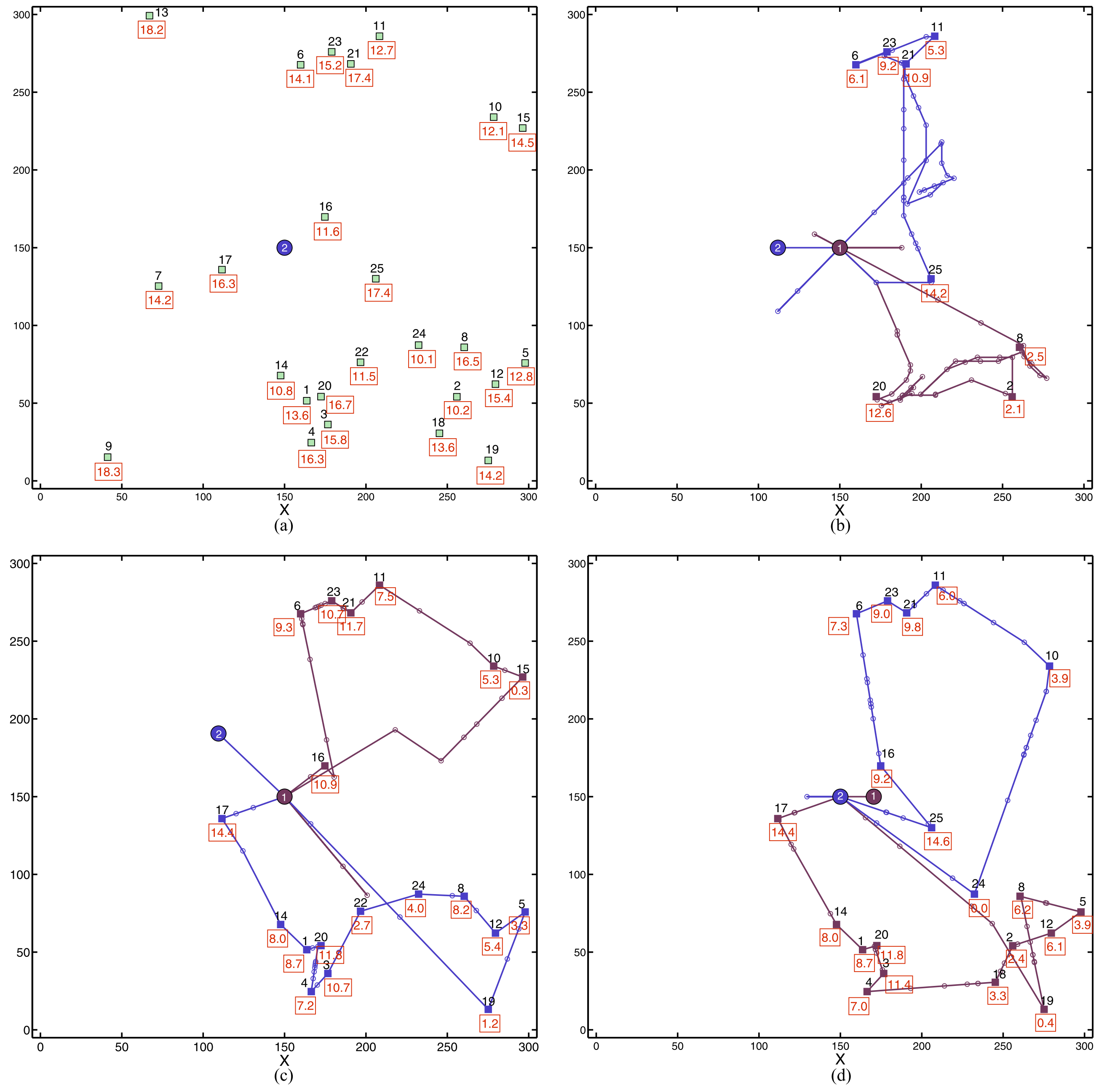

Fig. 8. Performance comparison of the original and new CRH algorithms (numbers in red show the reward for each target). (a) Initial mission. (b) Original CRH controller, 8 out of 25 targets visited prior to their deadlines. $\text{Reward} = 62.8$, $\text{Travel Time} = 714$, Computation Time: 108 s. (c) 3-L Controller, 19 out of 25 targets visited prior to their deadlines. $\text{Reward} = 141.29$, $\text{Travel Time} = 677$, Computation Time: 104 s. (d) 5-L Controller, 20 out of 25 targets visited prior to their deadlines. $\text{Reward} = 143.42$, $\text{Travel Time} = 657$, Computation Time: 1400 s.

and the optimal path is $\theta^* = \theta_1$. Letting θ^{CRH} denote the path obtained by the 1-L CRH controller, we show next that this controller recovers the optimal path θ^*.

Theorem 2: Consider a two-target, one-agent mission. If $\gamma = 0$ in (15) and target i's reward at time t is $\lambda_i(1 - \frac{t}{D_i})$, then $\theta^{\text{CRH}} = \theta^*$.

Proof: See Appendix. ■

VII. Simulation Examples

We provide several MRCP examples in which the performance of the original and new CRH controllers is compared. We have also applied the CRH control method to the deterministic and completely homogeneous case of benchmark TSP problems. We emphasize that the CRH controller is *not* designed for deterministic TSP problems, thus it is not expected to perform as well as highly efficient TSP algorithms. Nonetheless, it performs to a comparable level; detailed results can be found in [36]. In all examples that follow, we use the cooperation parameter $\Delta = 0$, mobile agent speed $V_j = 1$, and the parameters for target rewards are $\alpha_i = 1$, $\beta_i = 1$ in (1).

Addressing Instabilities: As already mentioned, the original CRH controller may give rise to oscillatory trajectories and fail to complete a mission. This is illustrated in Fig. 7(a) for a

TABLE I
20 TARGET-2 AGENT MISSIONS

Mission #	Original CRH		3-L CRH	
	Reward	Travel Time	Reward	Travel Time
1	33.92	412	45.24	536
2	41.48	439	52.4	426
3	30.93	476	41.19	483
4	32.08	389	37.24	457
5	41.5	444	47.25	537
6	44.61	389	47.91	471
7	23.93	528	35.48	462
8	38.68	415	50.91	489
9	30.92	478	34.08	429
10	36.81	458	44.26	476
Average	**35.48**	**443**	**43.53**	**479**
SD	**6.29**	**43.92**	**6.40**	**37.75**

TABLE II
20 TARGET-2 AGENT MISSIONS, WITH RANDOM TARGET APPEARANCE

Mission #	Original CRH		2-L CRH	
	Reward	Travel Time	Reward	Travel Time
1	51.63	704	58.61	736
2	46.53	716	67.57	632
3	37.59	646	54.42	691
4	31.71	929	53.13	941
5	60.05	668	81.31	528
6	58.16	609	67.91	688
7	45.81	739	61.29	760
8	49.71	722	59.47	732
9	42.64	822	47.95	818
10	40.32	648	58.01	868
Average	**46.42**	**720**	**60.97**	**739**
SD	**8.69**	**94.52**	**9.40**	**117.87**

TABLE III
EFFECT OF THE SPARSITY FACTOR ζ_i IN CLUSTERED MISSIONS

Mission #	$\gamma = 0$		$\gamma = 0.3$	
	Reward	Travel Time	Reward	Travel Time
1	40.62	552	61.9	413
2	64.89	447	64.64	420
3	35.24	471	63.8	461
4	63.78	465	64.64	478
5	25.42	493	26.5	449
6	22	454	22	454
7	44.1	458	46.84	449
8	34.26	466	61.21	472
Average	**41.29**	**475**	**51.44**	**449**
SD	**15.95**	**33.72**	**17.79**	**22.91**

simple mission with three linearly discounted reward targets. In Fig. 7(b), it is shown that the new CRH controller can easily determine the optimal path in this simple case.

Comparison of the Two CRH Controllers: A mission with 25 targets distributed uniformly and 2 agents starting at a base is considered as shown in Fig. 8(a), with uniformly distributed initial rewards: $\lambda_i \sim U(10, 20)$ and $D_i \sim U(300, 600)$ as in (1). In this case, the original CRH [Fig. 8(b)] significantly underperforms compared to a 3-L and 5-L CRH controller [Figs. 8(c), (d)] by a large margin, with the 3-L and 5-L controllers more than doubling the mission reward. We have used a value of $\gamma = 0.3$ and $I_0 = 25$ in (15). The value for I_k is equal to the number of remaining targets in the mission at time t_k, i.e., $I_k = M_k$. The best travel time is from the 5-L controller where the mission is completed in 657 units of time. It should be noted that minimizing time is not an objective of the MRCP considered here and reward maximization dictates the final length of the mission.

In Fig. 8 for the first ten iterations the original CRH average CPU time is around 3 s while for the 3-L and 5-L controllers it is 6 s and 114 s, respectively. However, in the last ten iterations the 3-L and 5-L CPU time goes down to 0.5 s, while it stays the same for the original CRH controller. In Fig. 8, it can be seen that in the original algorithm the red agent exhibits oscillatory behavior around targets 3, 20, 4, 1. This results in missing three of these targets. This behavior is one reason that the original controller underperforms in these cases. The other reason is that it has to solve a nonlinear optimization problem at each iteration whose complexity is fixed, whereas in the new CRH controller the computation time decreases drastically as the number of targets left in the mission space decreases. The computation times for all the controllers are shown in Fig. 8 (based on a 3.16 GHz 2-core CPU). In order to calculate the control for the original CRH controller, the complete infinite feasible control set is discretized into a finite number of headings instead of solving the original complex nonlinear optimization problem (in this case, we discretized to 16 headings). The total computation times for the original CRH and 3-L controllers are almost the same, while that of the 5-L controller is almost 10 times longer. In terms of computation time per control iteration, this is approximately the same in all original CRH controller updates, since it always involves evaluating the objective function for 16N controls in this case. In contrast, for the 3-L and 5-L controllers, computation times are longer in the first few control updates and become shorter in the final iterations due to the smaller number of targets remaining to be visited.

Randomly Generated Missions: To compare the overall performance of the new CRH controller, we generated ten missions, each with 20 targets that are uniformly located in a 300×300 mission space and two agents initially at the base. We have used $\lambda_i \sim U(2, 12)$ and $D_i = 300$. The results are shown in Table I, in which can be seen that in every single case, the new CRH controller collects a higher reward (22% average increase) with, sometimes, a longer travel time (8% average increase). This longer travel time is to be expected since the new controller is capable of visiting *more targets* before they disappear due to vanishing rewards. In another example, 10 missions were generated, each with 20 targets where 10 targets are only initially available to the agents. The other 10 targets appear at random time instants. We use an initial reward $\lambda_i \sim U(2, 12)$ and the parameter $D_i \sim U(300, 600)$. The comparison of the original and new CRH controller is shown in Table II. An increase of 31% is seen in the total reward with a slight 2% increase in the total mission time.

Sparsity Factor in Clustered Missions: We considered eight missions with 20 targets that are located uniformly in one case and in nine clusters in a second case. The goal here is to investigate the contribution of the sparsity factor ζ_i in (15). We have again used $\lambda_i \sim U(2, 12)$ and $D_i = 300$. We consider a case with $\gamma = 0$ which eliminates the effect of ζ_i and a second case with $\gamma = 0.3$ and $I_K = 5$ in (15). The results in Table III indicate that in the clustered missions rewards are improved by about 24%. It is worth pointing out that the change in the γ value did not have a significant effect in missions with targets distributed uniformly (not shown in Table III).

VIII. Conclusions and Future Work

A new CRH controller was developed for solving cooperative multi-agent problems in uncertain environments using the framework of previous work in [32]. We overcame several limitations of the controller developed in [32], including agent trajectory instabilities and inaccurate estimation of a reward-to-go function while improving the overall performance. The event-driven CRH controller is developed to solve the MRCP, where multiple agents cooperate to maximize the total reward collected from a set of stationary targets in the mission space. The mission environment is uncertain, for example targets can appear at random times and agents might have a limited sensing range. The controller sequentially solves optimization problems over a planning horizon and executes the control for a shorter action horizon, where both are defined by certain events associated with new information becoming available. Unlike the earlier CRH controller, the feasible control set is finite instead of infinite. In several numerical comparisons, we showed that the new CRH controller has a better performance than the original one.

In ongoing work, the same framework is applied to problems other than the MRCP, such as "data harvesting" where each target is generating data that should be collected and delivered to the base. Here, the base acts as a target with dynamic reward. Ongoing work is also focused on extending the CRH controller to a decentralized version where each agent is responsible for calculating its own control with limited information. This will be pursued along the lines of [18] where decentralization was done for the original CRH controller. In [18] the main idea is that the reward-to-go generates a potential field in the centralized algorithm and the same pattern is followed in the distributed version by evaluating the gradient of potential fields for each separate agent. In the new algorithm, we can evaluate J_A from the perspective of each agent. Calculating the immediate reward J_I for each agent is also trivial, given that the agent should at least be able to see the closest target to itself. Note, however, that we may lose the synchronization of agents and H_K may need to be defined for each agent separately.

Finally, we are pursuing extensions to mission space topologies that include obstacles or are better represented through graphs with agents restricted to move along edges of the graph.

Appendix

Proof of Lemma 1: From the definition of $\eta_i(x, t)$ in (16) and $\mathcal{C}_{l,j}(t_k, H_k)$ in (19) we have

$$d(\mathcal{C}_{l,j}(t_k, H_k), \mathbf{y}_l) \le d(\mathbf{x}, \mathbf{y}_l),\ \forall \mathbf{x} \in \mathcal{F}_j(t_k, H_k) \qquad (32)$$

Dividing both sides by $\lambda_l D_l^{-1}$ and adding $\zeta_l(t_k + H_k)$ we get, for all $\mathbf{x} \in \mathcal{F}_j(t_k, H_k)$

$$\eta_l(\mathcal{C}_{l,j}(t_k, H_k), t_k + H_k) \le \eta_l(\mathbf{x}, t_k + H_k) \qquad (33)$$

To prove the forward lemma statement, we use a contradiction argument and assume there exists a target r such that

$$\eta_l(\mathcal{C}_{l,j}(t_k, H_k), t_k + H_k) > \eta_r(\mathcal{C}_{l,j}(t_k, H_k), t_k + H_k).$$

Using (33), we get $\eta_r(\mathcal{C}_{l,j}(t_k, H_k), t_k + H_k) < \eta_l(\mathbf{x}, t_k + H_k)$ for all $\mathbf{x} \in \mathcal{F}_j(t_k, H_k)$. This implies that there exists no $\mathbf{x} \in \mathcal{F}_j(t_k, H_k)$ such that $l = \operatorname{argmin}_i \eta_i(\mathbf{x}, t_k + H_k)$. Therefore, l cannot be an active target, which contradicts the assumption, hence (20) is true.

To prove the reverse statement, we assume that (20) holds for any $i \in \mathcal{T}_k$, i.e.,

$$\eta_l(\mathcal{C}_{l,j}(t_k, H_k), t_k + H_k) < \eta_i(\mathcal{C}_{l,j}(t_k, H_k), t_k + H_k).$$

By the definition of active targets (17), we then know that l is an active target for agent j at time t_k. ■

Proof of Lemma 2: The active target set creates a partition of the set $\mathcal{F}_1(t_k, H_k)$ where each subset is an arc in a Euclidean mission space. For any active target $l \in S_1(t_k, H_k)$, let the arc associated with target l be $\mathcal{F}_1^l(t_k, H_k) \subset \mathcal{F}_j(t_k, H_k)$. For each $\mathcal{F}_1^l(t_k, H_k)$, we prove that for all $\mathbf{x} \in \mathcal{F}_1^l(t_k, H_k)$ the heading $\mathbf{v}^* = v(\mathcal{C}_{l,1}(t_k, H_k))$ satisfies

$$J(v(\mathbf{x}), t_k, H_k) \le J(\mathbf{v}^*, t_k, H_k). \qquad (34)$$

This means that among all headings $v(\mathbf{x})$, $\mathbf{x} \in \mathcal{F}_1^l(t_k, H_k)$ the heading towards the active target l is the optimal heading in that arc. There are two possible cases:

Case 1: $\mathbf{y}_l \in \mathcal{F}_1(t_k, H_k)$. This means $d(\mathbf{y}_l, \mathbf{x}_1(t)) = H_k$. Also, from (19) and (12), we know that $\forall r \in \mathcal{T}_k$

$$q_r(\mathcal{C}_{r,1}(t_k + H_k)) = \begin{cases} 1 & \text{if } r = l \\ 0 & \text{otherwise} \end{cases}$$

Setting $\tilde{\tau}_r(\mathbf{v}^*, t_k, H_k)) = \tilde{\tau}_r^*$, the visit time for target l is $\tilde{\tau}_l^* = t_k + H_k$ and reward from target l will be collected at time $t_k + H_k$. The estimated visit times $\tilde{\tau}_r^*$ for the rest of the targets are determined based on a tour $\boldsymbol{\theta}^*$ which is a permutation of the set $\mathcal{T}_{k+1}^* = \mathcal{T}_k - \{l\}$. This tour gives the order of visit for all remaining targets. The objective function for the control $\mathbf{v}^*$ is as follows:

$$\begin{aligned} J(\mathbf{v}^*, t_k, H_k) &= J_{\mathbf{I}}(\mathbf{v}^*, t_k, H_k) + J_{\mathbf{A}}(\mathbf{v}^*, t_k, H_k) \\ &= \lambda_l \phi_l(t_k + H_k) + \sum_{r=1}^{M_{k+1}^*} \lambda_r \phi_r(\tilde{\tau}_r^*) q_l(\mathbf{x}_1(\tilde{\tau}_r^*)). \end{aligned}$$

Here, $M_{k+1}^* = |\mathcal{T}_{k+1}^*|$. Next we calculate the objective function for any other heading $v(\mathbf{x})$ where $\mathbf{x} \in \mathcal{F}_1^l(t_k, H_k)$. Setting $\tilde{\tau}_r(v(\mathbf{x}), t_k, H_k)) = \tilde{\tau}_r$ and $\mathbf{x} \ne \mathcal{C}_{l,1}(t_k, H_k)$ so that $q_r(\mathbf{x}) = 0$ for all $r \in \mathcal{T}_k$, we have

$$\begin{aligned} J(v(\mathbf{x}), t_k, H_k) &= J_{\mathbf{I}}(v(\mathbf{x}), t_k, H_k) + J_{\mathbf{A}}(v(\mathbf{x}), t_k, H_k) \\ &= 0 + \sum_{r=1}^{M_{k+1}} \lambda_r \phi_r(\tilde{\tau}_r) q_l(\mathbf{x}_1(\tilde{\tau}_r)). \end{aligned}$$

In this case, no target is visited at time $t_k + H_k$ and the aggregated tour $\boldsymbol{\theta}$ is a permutation of the set $\mathcal{T}_{k+1} = \mathcal{T}_k$. By definition, the target with the least travel cost from point $\mathbf{x}$ is the active target l and this is the first target in the tour $\boldsymbol{\theta}$. The rest of the tour consists of targets in $\mathcal{T}_{k+1} - \{l\}$. Since in both tours $\boldsymbol{\theta}^*$ and $\boldsymbol{\theta}$ the starting point and the set of available targets are the same, the order of targets will be identical and we have $\boldsymbol{\theta} = [l, \boldsymbol{\theta}^*]$. The visit times in $\boldsymbol{\theta}^*$ are given by

$$\tilde{\tau}_{\boldsymbol{\theta}^*_n} = t_k + H_k + \sum_{i=1}^{n-1} d(\mathbf{y}_{\boldsymbol{\theta}^*_i}, \mathbf{y}_{\boldsymbol{\theta}^*_{i+1}}).$$

In $\boldsymbol{\theta}$, the visit time for the first target $\boldsymbol{\theta}_1 = l$ is $\tilde{\tau}_{\boldsymbol{\theta}_1} = t_k + H_k + d(\mathbf{x}, \mathbf{y}_l)$. For the rest of the targets, with $1 < n \leq M_{k+1}$

$$\tilde{\tau}_{\boldsymbol{\theta}_n} = t_k + H_k + d(\mathbf{x}, \mathbf{y}_l) + \sum_{i=1}^{n-1} d(\mathbf{y}_{\boldsymbol{\theta}_i}, \mathbf{y}_{\boldsymbol{\theta}_{i+1}}).$$

Expanding both objective functions in (34) we need to prove that

$$\lambda_l \phi_l(\tilde{\tau}_l) + \sum_{n=2}^{M_{k+1}} \lambda_{\boldsymbol{\theta}_n} \phi_{\boldsymbol{\theta}_n}(\tilde{\tau}_{\boldsymbol{\theta}_n}) \leq \lambda_l \phi_l(\tilde{\tau}^*_l) + \sum_{n=1}^{M^*_{k+1}} \lambda_{\boldsymbol{\theta}_n} \phi_{\boldsymbol{\theta}^*_n}(\tilde{\tau}_{\boldsymbol{\theta}^*_n}). \tag{35}$$

Recall that by assumption, $\phi_i(t)$ is a non-increasing function. Since for all $1 < n \leq M_{k+1}$, we have $\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}^*_n$ and $\tilde{\tau}_{\boldsymbol{\theta}_{n+1}} > \tilde{\tau}_{\boldsymbol{\theta}^*_n}$ it follows that

$$\phi_{\boldsymbol{\theta}_{n+1}}(\tilde{\tau}_{\boldsymbol{\theta}_{n+1}}) \leq \phi_{\boldsymbol{\theta}^*_n}(\tilde{\tau}_{\boldsymbol{\theta}^*_n}).$$

Summing over all the targets and observing that $M_{k+1} = 1 + M^*_{k+1}$ we have

$$\sum_{n=2}^{M_{k+1}} \lambda_{\boldsymbol{\theta}'_n} \phi_{\boldsymbol{\theta}_n}(\tilde{\tau}_{\boldsymbol{\theta}_n}) \leq \sum_{n=1}^{M^*_{k+1}} \lambda_{\boldsymbol{\theta}_n} \phi_{\boldsymbol{\theta}^*_n}(\tilde{\tau}_{\boldsymbol{\theta}^*_n}). \tag{36}$$

For target l under the control $\mathbf{v}^*$, we have $\tilde{\tau}^*_l = t_k + H_k$ and under $v(\mathbf{x})$, we have $\tilde{\tau}_l = (t_k + H_k + d(\mathbf{x}, \mathbf{y}_l))$. Using the non-increasing property of the function $\phi_i(t)$ we have

$$\phi_l(\tilde{\tau}_l) \leq \phi_l(\tilde{\tau}^*_l). \tag{37}$$

Combining (36) and (37) yields (35).

Case 2: $\mathbf{y}_l \notin \mathcal{F}_1(t_k, H_k)$. In this case, for any point $\mathbf{x} \in \mathcal{F}^l_1(t_k, H_k)$ we have a zero immediate reward. Thus, only the rewards-to-go need to be compared. Using (18), for any $\mathbf{x} \in \mathcal{F}^l_1(t_k, H_k)$ we know the aggregation tour $\boldsymbol{\theta}$ for any point $\mathbf{x}$ starts with target l and the rest of it would also be the same. Similarly, let us assume $\boldsymbol{\theta}^*$ is the tour for $\mathbf{v}^*$ and $\boldsymbol{\theta}$ is the tour for any other control $v(\mathbf{x})$. The estimated visit times for $\boldsymbol{\theta}^*$ are

$$\tilde{\tau}_{\boldsymbol{\theta}^*_n} = t_k + H_k + d(\mathbf{y}_l, \mathcal{C}_{l,1}(t_k, H_k)) + \sum_{i=1}^{n-1} d(\mathbf{y}_{\boldsymbol{\theta}^*_i}, \mathbf{y}_{\boldsymbol{\theta}^*_{i+1}})$$

and for $\boldsymbol{\theta}$

$$\tilde{\tau}_{\boldsymbol{\theta}_n} = t_k + H_k + d(\mathbf{y}_l, \mathbf{x}) + \sum_{i=1}^{n-1} d(\mathbf{y}_{\boldsymbol{\theta}_i}, \mathbf{y}_{\boldsymbol{\theta}_{i+1}}).$$

By the definition in (19), $\mathcal{C}_{l,1}(t_k, H_k))$ is on the shortest path from $\mathbf{x}_1(t_k)$ to $\mathbf{y}_l$, i.e., $\tilde{\tau}_{\boldsymbol{\theta}_n} > \tilde{\tau}_{\boldsymbol{\theta}^*_n}$. Again, with $\phi_i(t)$ being non-increasing, we have $\phi_{\boldsymbol{\theta}_n}(\tilde{\tau}_{\boldsymbol{\theta}_n}) \leq \phi_{\boldsymbol{\theta}^*_n}(\tilde{\tau}_{\boldsymbol{\theta}^*_n})$ for all n, which implies $J(v(\mathbf{x}), t_k, H_k) \leq J(\mathbf{v}^*, t_k, H_k)$.

We have thus proved the lemma statement that the optimal heading of the agent is one of the direct headings towards an active target. ■

Proof of Theorem 1: In the multi-agent mission, calculating the immediate reward and reward-to-go in (13) and (24) for each agent is equivalent to a one-agent mission limited to its own target subset $\mathcal{T}_{k,j}$. Therefore, the result follows directly from Lemma 2. ■

Proof of Theorem 2: We assume WLOG that $d(\mathbf{x}, \mathbf{y}_1) < d(\mathbf{x}, \mathbf{y}_2)$ so that at time t_k we have $H_k = d(\mathbf{x}, \mathbf{y}_1)$. This implies that target 1 is always an active target (the travel cost of target 1 at time $t_{k+1} = t_k + H_k$ is equal to 0). Recalling (19) and setting $\mathcal{C}_{2,1} = \mathcal{C}_{2,1}(t_k, H_k)$, we have $d(\mathbf{x}, \mathbf{y}_1) = d(\mathbf{x}, \mathcal{C}_{2,1}) = H_k$. This results in

$$d(\mathbf{x}, \mathbf{y}_2) = d(\mathbf{x}, \mathbf{y}_1) + d(\mathbf{y}_2, \mathcal{C}_{2,1}). \tag{38}$$

From Lemma 1, target 2 is an active target if and only if $\eta_2(\mathcal{C}_{2,1}, t_k + H_k) \leq \eta_1(\mathcal{C}_{2,1}, t_{k+1})$. Therefore, from (16), target 2 is an active target if and only if

$$\frac{d(\mathcal{C}_{2,1}, \mathbf{y}_2)}{\lambda_2 D_2^{-1}} \leq \frac{d(\mathcal{C}_{2,1}, \mathbf{y}_1)}{\lambda_1 D_1^{-1}}$$

which is rewritten as

$$\frac{\lambda_1}{D_1} d(\mathcal{C}_{2,1}, \mathbf{y}_2) \leq \frac{\lambda_2}{D_2} d(\mathcal{C}_{2,1}, \mathbf{y}_1).$$

We now consider two possible cases regarding target 2. First, assume target 2 is not an active target, i.e.,

$$\frac{\lambda_1}{D_1} d(\mathcal{C}_{2,1}, \mathbf{y}_2) > \frac{\lambda_2}{D_2} d(\mathcal{C}_{2,1}, \mathbf{y}_1). \tag{39}$$

Starting with the trivial inequality

$$0 > \frac{-\lambda_1}{D_1} \left[d(\mathcal{C}_{2,1}, \mathbf{y}_2) + d(\mathcal{C}_{2,1}, \mathbf{y}_1) \right]$$

add $\frac{\lambda_2}{D_2} \left[d(\mathcal{C}_{2,1}, \mathbf{y}_1) \right]$ to both sides and use (39) to get

$$\frac{\lambda_1}{D_1} \left[d(\mathcal{C}_{2,1}, \mathbf{y}_2) \right] > \frac{\lambda_2}{D_2} \left[d(\mathcal{C}_{2,1}, \mathbf{y}_1) \right]$$
$$> \frac{-\lambda_1}{D_1} \left[d(\mathcal{C}_{2,1}, \mathbf{y}_2) \right] + \left(\frac{\lambda_2}{D_2} - \frac{\lambda_1}{D_1} \right) d(\mathcal{C}_{2,1}, \mathbf{y}_1) \Big].$$

Adding the positive quantity of $\frac{\lambda_2}{D_2} \left[d(\mathcal{C}_{2,1}, \mathbf{y}_2) \right]$ to both sides and invoking the triangle inequality

$$\left(\frac{\lambda_1}{D_1} + \frac{\lambda_2}{D_2} \right) \left[d(\mathcal{C}_{2,1}, \mathbf{y}_2) \right] > \left(\frac{\lambda_2}{D_2} - \frac{\lambda_1}{D_1} \right) \left[d(\mathcal{C}_{2,1}, \mathbf{y}_2) \right]$$
$$+ \left(\frac{\lambda_2}{D_2} - \frac{\lambda_1}{D_1} \right) \left[d(\mathcal{C}_{2,1}, \mathbf{y}_1) \right] > \left(\frac{\lambda_2}{D_2} - \frac{\lambda_1}{D_1} \right) d(\mathbf{y}_1, \mathbf{y}_2).$$

Rearranging the last inequality and using (38) results in

$$\frac{\lambda_1}{D_1} \left[d(\mathbf{x}, \mathbf{y}_2) + d(\mathbf{y}_2, \mathbf{y}_1) \right] + \frac{\lambda_2}{D_2} d(\mathbf{x}, \mathbf{y}_2) > \frac{\lambda_1}{D_1} d(\mathbf{x}, \mathbf{y}_1)$$
$$+ \frac{\lambda_2}{D_2} \left[d(\mathbf{x}, \mathbf{y}_1) + d(\mathbf{y}_2, \mathbf{y}_1) \right] \tag{40}$$

which is the same as (31) implying that path $\theta_1 = (1, 2)$ is optimal, i.e., the CRH controller finds the optimal path.

Next, assume that target 2 is also an active target along with target 1. Let u_1 and u_2 be the headings for target 1 and 2, respectively, i.e., $\mathbf{x}_1(t_{k+1}, u_1) = \mathbf{y}_1$ and $\mathbf{x}_1(t_{k+1}, u_2) = \mathcal{C}_{2,1}$, The objective function of the CRH controller under u_1 and u_2 is

$$\begin{aligned} J(u_1, t_k, H_k) &= J_{\mathbf{I}}(u_1, t_k, H_k) + J_{\mathbf{A}}(u_1, t_k, H_k) \\ &= \lambda_1\phi_1(t_{k+1}) + \lambda_2\phi_2(t_{k+1} + d(\mathbf{y}_1, \mathbf{y}_2)) \\ J(u_2, t_k, H_k) &= J_{\mathbf{I}}(u_2, t_k, H_k) + J_{\mathbf{A}}(u_2, t_k, H_k) \\ &= 0 + \big[\lambda_2\phi_2\big(t_{k+1} + d(\mathcal{C}_{2,1}, \mathbf{y}_2)\big) \\ &\quad + \lambda_1\phi_1\big(t_{k+1} + d(\mathcal{C}_{2,1}, \mathbf{y}_2) + d(\mathbf{y}_1, \mathbf{y}_2)\big). \end{aligned}$$

Note that in order to evaluate the objective function for u_2 we find a tour starting at point $\mathcal{C}_{2,1}$ which goes to the target with minimum travel cost. However, for target 2 to be active at t_k it has to have the smallest travel cost at that point, which results in $J_{\mathbf{A}}(u_2, t_k, H_k)$ to be the reward of going to target 2 and then target 1. We can see that using the reward of each path from (29) and (30) we can write

$$J(u_1, t_k, H_k) = R_{(1,2)}, \qquad J(u_2, t_k, H_k) = R_{(2,1)}.$$

Thus, the objective function of the CRH controller under u_1 and u_2 is identical to the corresponding path rewards. Hence, the CRH controller selects the correct optimal heading at t_k. ■

REFERENCES

[1] J. S. Shamma, *Cooperative Control of Distributed Multi-Agent Systems*. Wiley Online Library, 2007.

[2] R. M. Murray, "Recent research in cooperative control of multivehicle systems," *J. Dynam. Syst., Measur., and Control*, vol. 129, no. 5, pp. 571–583, 2007.

[3] R. Murphey and P. M. Pardalos, *Cooperative Control and Optimization*, vol. 66 New York: Springer, 2002.

[4] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Trans. Autom. Control*, vol. 48, no. 6, pp. 988–1001, 2003.

[5] T. McLain, P. Chandler, S. Rasmussen, and M. Pachter, "Cooperative control of UAV rendezvous," *Proc. American Control Conf.*, pp. 2309–2314, 2001.

[6] C. Yao, X. C. Ding, and C. G. Cassandras, "Cooperative receding horizon control for multi-agent rendezvous problems in uncertain environments," in *Proc. 49th IEEE Conf. Decision and Control (CDC)*, pp. 4511–4516, Dec. 2010.

[7] C. G. Cassandras, X. Lin, and X. Ding, "An optimal control approach to the multi-agent persistent monitoring problem," *IEEE Trans. Autom. Control*, vol. 58, pp. 947–961, Apr. 2013.

[8] 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, 2004.

[9] M. Zhong and C. G. Cassandras, "Distributed coverage control and data collection with mobile sensor networks," *IEEE Trans. Autom. Control*, vol. 56, no. 10, pp. 2445–2455, 2011.

[10] D. Panagou, M. Turpin, and V. Kumar, "Decentralized goal assignment and trajectory generation in multi-robot networks: A multiple lyapunov functions approach," in *Proc. 2014 IEEE Int. Conf. Robotics and Automation (ICRA)*, pp. 6757–6762, May 2014.

[11] W. Ren and R. Beard, *Distributed Consensus in Multi-Vehicle Cooperative Control: Theory and Applications*. New YorkSpringer, 2008.

[12] M. Zhong and C. G. Cassandras, "Asynchronous distributed optimization with event-driven communication," *IEEE Trans. Autom. Control*, vol. 55, no. 12, pp. 2735–2750, 2010.

[13] R. Arvind, G. Vipin, S. Dharmashankar, B. Lorenz, and S. Tariq, 3D conflict resolution of multiple aircraft via dynamic optimization. *Guidance, Navigation, and Control and Co-located Conf.*, Amer. Inst. Aeronaut. and Astronaut., 2003.

[14] I. Harmati and K. Skrzypczyk, "Robot team coordination for target tracking using fuzzy logic controller in game theoretic framework," *Robot. Auton. Syst.*, vol. 57, no. 1, pp. 75–86, 2009.

[15] E. Frazzoli, Z. H. Mao, J. H. Oh, and E. Feron, "Resolution of conflicts involving many aircraft via semidefinite programming," *J. Guid., Control, and Dynamics*, vol. 24, no. 1, pp. 79–86, 2001.

[16] W. Ren, R. W. Beard, and T. W. McLain, "Coordination variables and consensus building in multiple vehicle systems," *Coop. Control*, vol. 309, pp. 171–188, 2005.

[17] W. Ren, "Cooperative control design strategies with local interactions," in *Proc. IEEE Int. Conf. Networking, Sensing and Control, 2006*, pp. 451–456, 2006.

[18] W. Li and C. G. Cassandras, "Centralized and distributed cooperative receding horizon control of autonomous vehicle missions," *Mathemat. Comput. Modell.*, vol. 43, no. 9, pp. 1208–1228, 2006.

[19] D. Cruz, J. McClintock, B. Perteet, O. A. A. Orqueda, C. Yuan, and R. Fierro, "Decentralized cooperative control–A multivehicle platform for research in networked embedded systems," *IEEE Control Syst.*, vol. 27, no. 3, pp. 58–78, 2007.

[20] N. P. Salz, "Anon—A theory for traveling salesman problem," *Oper. Res.*, vol. S 14, 1966.

[21] D. L. Applegate, R. E. Bixby, V. Chvatal, and W. J. Cook, *The Traveling Salesman Problem: A Computational Study*. Princeton, NJ: Princeton University Press, 2011.

[22] G. Laporte, "The vehicle routing problem: An overview of exact and approximate algorithms," *Eur. J. Oper. Res.*, vol. 59, no. 3, pp. 345–358, 1992.

[23] A. Ekici and A. Retharekar, "Multiple agents maximum collection problem with time dependent rewards," *Comput. and Ind. Eng.*, vol. 64, no. 4, pp. 1009–1018, 2013.

[24] H. Tang, E. Miller-Hooks, and R. Tomastik, "Scheduling technicians for planned maintenance of geographically distributed equipment," *Transport. Res. Pt. E: Logist. and Transport. Rev.*, vol. 43, no. 5, pp. 591–609, 2007.

[25] V. Pillac, M. Gendreau, C. Guret, and A. L. Medaglia, "A review of dynamic vehicle routing problems," *Eur. J. Oper. Res.*, vol. 225, no. 1, pp. 1–11, 2013.

[26] R. Lahyani, M. Khemakhem, and F. Semet, "Rich vehicle routing problems: From a taxonomy to a definition," *Eur. J. Oper. Res.*, vol. 241, no. 1, pp. 1–14, 2015.

[27] F. Bullo, E. Frazzoli, M. Pavone, K. Savla, and S. Smith, "Dynamic vehicle routing for robotic systems," *Proc. IEEE*, vol. 99, pp. 1482–1504, Sep. 2011.

[28] J. S. Bellingham, M. Tillerson, M. Alighanbari, and J. P. How, "Cooperative path planning for multiple UAVs in dynamic and uncertain environments," in *Proc. IEEE Conf. Decision and Control (CDC)*, pp. 2816–2822 vol. 3, 10–13 Dec. 2002.

[29] M. G. Earl and R. D'Andrea, "A decomposition approach to multi-vehicle cooperative control," *Robot. Auton. Syst.*, vol. 55, pp. 276–291, 2007.

[30] D. V. Dimarogonas and K. H. Johansson, "Event-triggered cooperative control," in *Eur. Control Conf.*, pp. 3015–3020, 2009.

[31] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[32] W. Li and C. G. Cassandras, "A cooperative receding horizon controller for multivehicle uncertain environments," *IEEE Trans. Autom. Control*, vol. 51, no. 2, 2006.

[33] C. Poli, G. Chini, G. Oddi, and A. Pietrabissa, "Receding horizon multi-vehicle routing for emergency scenarios," in *Mediterranean Conference on Control and Automation (MED)*, Palermo, Italy, , University of Palermo. Jun. 16–19, 2014.

[34] Y. Khazaeni and C. G. Cassandras, "A new event-driven cooperative receding horizon controller for multi-agent systems in uncertain environments," in *Proc. 53rd IEEE Conf. Decision and Control (CDC)*, Dec. 2014, Los Angeles, CA.

[35] J. J. Schneider, T. Bukur, and A. Krause, "Traveling salesman problem with clustering," *J. Statist. Phys.*, vol. 141, no. 5, pp. 767–784, 2010.

[36] Y. Khazaeni and C. G. Cassandras, "A new event-driven cooperative receding horizon controller for multi-agent systems in uncertain environments," *arXiv:1403.3434v2[cs.SY]*.

Yasaman Khazaeni (M'11) received the B.S. degrees in electrical engineering and petroleum engineering from Sharif University of Technology, Tehran, Iran in 2005 and the M.S. degree in petroleum engineering from West Virginia University, Morgantown, in 2010, and the Ph.D. degree in systems engineering from Boston University in 2016.

From 2010 to 2011, she was a Research Associate with West Virginia University, where she was involved in managing projects on the artificial intelligence application in reservoir engineering. From 2011 to 2016, she was a research assistant at CODES Lab in Boston university where her research was focused on Cooperative Control and Stochastic Optimization problems. She is currently a Research Staff Member in the Cognitive User Experience Lab, IBM Research, Cambridge, MA. Her research experiences are in the area of control theory and mathematical modeling, event-driven control, stochastic systems, and artificial intelligence.

Dr. Khazaeni is a member of Phi Kappa Phi and a recipient of several awards, including the 2008 and 2009 West Virginia University distinguished graduate student, a second prize from the Nico van Wingen SPE annual Scholarship, a 2011 Boston University Dean's Fellowship, and a 2014 NSF Travel Award.

Christos G. Cassandras (F'96) received the B.S. degree from Yale University, New Haven, CT, in 1977, the M.S.E.E. degree from Stanford University, Stanford, CA, in 1978, and the M.S. and Ph.D. degrees from Harvard University, Cambridge, MA, in 1979 and 1982, respectively.

He was with ITP Boston, Inc., Cambridge, MA, from 1982 to 1984, where he was involved in the design of automated manufacturing systems. From 1984 to 1996, he was a faculty member with the Department of Electrical and Computer Engineering, University of Massachusetts Amherst. He is currently a Distinguished Professor of Engineering with Boston University, Brookline, MA, the Head of the Division of Systems Engineering, and a Professor of Electrical and Computer Engineering. He specializes in the areas of discrete event and hybrid systems, cooperative control, stochastic optimization, and computer simulation, with applications to computer and sensor ne tworks, manufacturing systems, and transportation systems. He has authored over 380 refereed papers in these areas, and five books.

Dr. Cassandras was the recipient of several awards, including the 1991 Lilly Fellowship, the 1999 Harold Chestnut Prize (IFAC Best Control Engineering Textbook), the 2006 Distinguished Member Award of the IEEE Control Systems Society, the 2011 IEEE Control Systems Technology Award, the 2011 and 2014 prize for the IBM/IEEE Smarter Planet Challenge competition, a 2012 Kern Fellowship, and the 2014 Engineering Distinguished Scholar Award from Boston University, as well as several honorary professorships. He was Editor-in-Chief of the IEEE TRANSACTIONS ON AUTOMATIC CONTROL from 1998 to 2009. He serves on several editorial boards, has been a Guest Editor for various journals, and was President of the IEEE Control Systems Society in 2012. He is a member of Phi Beta Kappa and Tau Beta Pi, and a Fellow of the International Federation of Automatic Control (IFAC).