Control barrier Function with Spliting Coefficients for Multi-robot System

Xiaoshan Lin and Roberto Tron

***Abstract*— This paper presents a control methodology for multi-robot system to enforce completeness of tasks before deadline and enable decentralized coordination among agents. The approach is based on time-varying zeroing control barrier function (ZCBF). We construct a time-varying safety set within which a robot can potentially satisfy a deadline. Enforcing at least one robot in the safety set, i.e. designing controller to render the safety set forward invariant, guarantees the satisfaction of the task before deadline. We propose a consensus-like splitting protocol for the system and use dynamic coefficients to "split" the inequality constraint to each agent. Robots will either satisfy the safety constraint to finish the task or discard the constraint, depending on the value of their coefficients. The construction of such splitting protocol for multi-robot system allows for decentralized coordination while increasing local flexibility for robots.**

I. Introduction

The collaboration of multiple robots enables them to complete tasks efficiently and be qualified for complicated tasks that single robot can not deal with. In recent decades, multi-robot system has been adopted to perform tasks such as swarm, formation control [1], surveillance [2], [3], localization and mapping [4], [5], etc.

In some cases multi-robot system must complete the tasks before a deadline. Most of the current research on multi-robot system with deadline constraints focus on developing task assignment and scheduling algorithm [6]–[8]. Such problems with time constraints can be regarded as safety problems, and the solution is to find a control strategy which keeps the system safe. The safety of a system can be identified with set invariance, where any trajectory starting within a safety set will never leave the set [9]. Another methodology termed *Barrier Certificate* was introduced in the last decade for safety verification of hybrid systems and nonlinear systems [10], [11], and was then extended to stochastic safety verification [12]. However, the above research was only targeted at systems without control inputs. Motivated by Control Lyapunov Function(CLF) and barrier certificate, Peter Wieland et al. introduced Control Barrier Functions(CBF) [13]. Here they designed the control input to ensure safety for the system. Then a method to guarantee both stability and safety by merging CLF and CBF was presented [14]. The limitation of these "Lyapunov-like" methods is that, while they guarantee the safety the condition is overly restrictive so that every subset of the safety set is rendered invariant [15]. To enforce the safety set invariant but not its subset, refined versions of control barrier functions were introduce: reciprocal control barrier functions [16] and zeroing control barrier functions [17].

[18] uses time-varying CBF to enforce satisfaction of task with deadline expressed as Signal Temporal Logic (STL) formulas.

In this paper, we consider a group of robots required to complete some tasks with strict deadlines in an unknown environment, without any centralized coordination. In an unknown environment, enforcing as many robots as possible to complete the same task may guarantee the satisfaction of that task but it is rather inefficient and energy-consuming. On the other hand, assigning only one robot for a task is energy-efficient, but it can not guarantee satisfaction of the deadline constraint because the robot may be blocked by complex obstacles. This motivates us to propose the multi-robot control barrier function with splitting coefficients. First, for each task we define a time-varying safety set within which robots can always satisfy the deadline. Second, we construct time-varying ZCBF with spliting coefficients for each robots, which will be discussed with details in Section II and III. Third, we design control inputs for each robot that ensure the exsistence of time-varying ZCBF which renders the safety set invariant, and thus guarantees the satisfaction of the deadline. Different cost functions are considered while computing the control input, which is the solution of a constrained optimization problem [19]. In addition, with the proposed splitting protocol the robots can easily collaborate and improve efficiency.

We begin in Section II by introducing the prelimineries and problem formulation. Control approach and solutions to the problem are presented in Section III. Results of simulation are discussed in Section IV. Conclusions are given in Section V.

II. Preliminary and Problem Formulation

A. Signal Temporal Logic

Signal temporal logic is based on predicate μ which is a boolean obtained by evaluating a continous signal $h:\mathbb{R}^n \to \mathbb{R}$ as follows:

$$\mu = \begin{cases} True & if\, h(\mathbf{x}) \geq 0 \\ False & if\, h(\mathbf{x}) < 0 \end{cases} \tag{1}$$

An STL formula ϕ is defined as

$$\phi ::= \mu \mid \neg\phi \mid \phi \wedge \psi \mid \phi \vee \psi \mid \phi\, U_{[a,b]}\, \psi,$$

where ϕ, ψ are STL formulas, $a, b \in \mathbb{R}_{\geq 0}$ and $a \leq b$.

$(\mathbf{x}, t) \models F_{[a,b]}\, \phi \Leftrightarrow \exists t_1 \in [t+a, t+b]\ \ s.t.\ \ (\mathbf{x}, t_1) \models \phi$

Xiaoshan Lin is with the Department of Mechanical Engineering, Boston University, Boston, MA 02215, USA; xslin@bu.edu

Roberto Tron is with the Department of Mechanical Engineering and the Division of Systems Engineering, Boston University, Boston, MA 02215, USA; tron@bu.edu

B. Zeoring Control Barrier Function

We present the definition of zeroing control barrier function [17] here to give readers a basic knowledge of ZCBF.

Definition 1. A continuous function $\alpha : (-b,a) \to (-\infty,+\infty)$ for some $a,b>0$ is said to belong to extended class $\mathcal{K}$ if it is strictly increasing and $\alpha(0)=0$.

Definition 2. For $\varepsilon \leq 0$, define the family of closed set $\mathcal{C}_\varepsilon$ as

$$\mathcal{C}_\varepsilon = \{x \mid x \in \mathbb{R}^n, h(x) \leq -\varepsilon\}$$
$$\partial C_\varepsilon = \{x \mid x \in \mathbb{R}^n, h(x) = -\varepsilon\}$$
$$Int(\mathcal{C}_\varepsilon) = \{x \mid x \in \mathbb{R}^n, h(x) > -\varepsilon\}$$

Definition 3. Consider a dynamics system

$$\dot{x} = f(x) + g(x)u \tag{2}$$

with f and g locally Lipschitz continuous, $x \in \mathbb{R}$ and $u \in U \subset \mathbb{R}^n$. Given a set $\mathcal{C} \subset \mathbb{R}$ defined by (1)-(3) for a continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$, the function h is called a zeroing control barrier function defined on set $\mathcal{D}$ with $\mathcal{C} \subseteq \mathcal{D} \subset \mathbb{R}^n$, if there exist a class $\mathcal{K}$ function α such that

$$\sup_{u \in U}[L_f h(x) + L_g h(x)u + \alpha(h(x))] \geq 0, \forall x \in \mathcal{D} \tag{3}$$

where $L_f h(x) = \frac{\partial h(x)}{\partial x} f(x)$, $L_f h(x) = \frac{\partial h(x)}{\partial x} g(x)u$.

C. Problem Formulation

Consider a multi-robot system of N ($N>1$) robots, and the states of each robot $i (1 \leq i \leq N)$ evolves according to the affine dynamics $\dot{x_i} = f(x_i) + g(x_i)u_i$. In this case, state x_i represents the position of robot i, i.e. $[X_i\ Y_i]^T$. The robots are working together on M ($M \geq 1$) tasks in an unknown environment without centralized coordination. Communication is available between every two robots. Task m ($1 \leq m \leq M$) is satisfied if at least one robot arrives at task region $\mathcal{C}_m$ before its deadline t_m, where $\mathcal{C}_m = \{(x,y) \mid (x-x_m)^2 + (y-y_m)^2 < r_m^2\}$. Then Task m can be expressed as a STL formula ϕ_m using disjunction operation:

$$\phi_m := \psi_{1,m} \vee \psi_{2,m} \vee \ldots \vee \psi_{N,m} \tag{4}$$

$$\psi_{i,m} := F_{[0,t_m]} \|x_i(t) - [x_m\ y_m]^T\| < r_m, i \in [1,N] \tag{5}$$

where we denote by (x_m, y_m), and r_m the center of task region $\mathcal{C}_m$ and radius of $\mathcal{C}_m$ respectively. Then the problem we consider in this paper is given as follows.

Problem 1. Derive control law u_i for each robot i such that ϕ_m is satisfied for $\forall m \in [1,M]$.

III. Control Approach and Solution

In this section, we propose a control approach to solve Problem 1. First we derive the control approach for single task ($M=1$), and then extend the solution to deal with multiple tasks ($M>1$).

For each robot i, we define $h_{\mathcal{C}_m}(x_i,t)$ to be the difference between the deadline and the shortest possible time in which an agent can reach $\mathcal{C}_m$ from robot x_i at time t, assuming a first order integrator dynamics.

$$h_{\mathcal{C}_m}(x_i,t) = (t_m - t) - d_{\mathcal{C}_m}(x_i(t))/v_{max} \tag{6}$$

$$d_{\mathcal{C}_m}(x_i(t)) = \|x_i(t) - [x_m\ y_m]^T\| - r_m$$

where v_{max} is the maximum speed of robot i, and $d_{\mathcal{C}_m}$ represents the distance between x and the boundary of task region $\mathcal{C}_m$. $\{x \mid h_{\mathcal{C}_m}(x,t) \geq 0\}$ represents the set of states at time t from which the robot can potentially satisfies the deadline associated to $\mathcal{C}_m$, i.e. safety set for robot i to satisfy task m before its deadline. By definition of $h_{\mathcal{C}_m}$, we know that a robot i can never satisfy the task t before its deadline t_m as long as $h_{\mathcal{C}_m}(x_i,t) < 0$ for some $t \in [0,t_m]$. Therefore the STL formula in (5) is equivalent to

$$h_{\mathcal{C}_m}(x_i,t) \geq 0,\ \forall t \in [0,t_m] \tag{7}$$

Then (4) is equivalent to having at least one robot satisfying (6). Such constraint can easily be satisfied by using ZCBF in (5). In an environment without obstacles, apparently the most efficient way is to enfoce the closest robot to $\mathcal{C}_m$ to satisfy (6) while the other robots will discard this task. Such approach can not guarantee the satisfaction of deadline in an unknown environment as the robot might be blocked by obstacles. In that case the safest approach is to enforce every available robots to satisfy $h_{\mathcal{C}_m}(x_i,t) \geq 0$. However, this

To enable collaboration among robots, for each task m we introduce the *splitting coefficients* $\alpha_{m,i} (\alpha_{m,i} \geq 0)$ such that $\sum_i \alpha_{m,i} = C$, where C is some constant number and we set $C=1$ for simplification. These splitting coefficients are used to split the constraint $h_{\mathcal{C}_m} > 0$ into individual constraints:

$$h_{\mathcal{C}_m,i} = \alpha_{m,i} h_{\mathcal{C}_m}. \tag{8}$$

A. Splitting Protocol

The splitting coefficient $\alpha_{i,m}$ we discuss here can be regarded as "responsibility" that robot i takes to complete task m before deadline t_m. Using constant splitting coefficients α_i results in a fixed assignment of responsibilities among robots. For example, fixing $\alpha_{m,1} = 1, \alpha_{m,i} = 0 (i \neq 1)$, would assign *robot* 1 to be responsible for satisfying task m before its deadline, while the others would always ignore this task.

Here we propose a splitting protocol to deal with dynamical splitting coefficients. Bounded splitting weights $0 \leq W_{m,i} \leq W_{max}$ are introduced to manipulate the dynamical splitting coefficients $\alpha_{m,i}$. These weights can be intuitively regarded as "potential" of each agent to finish the task; splitting coefficients of robots with higher weights will be increased and vice versa. This allows the robots to collaborate: robots with lower potential to complete the task before deadline will decrease their splitting coefficients and may finally discard the task when the coefficients become zero, while robots with higher potential will increase their splitting coefficients and continue their task. For instance, we can set $W_{m,i} = h_{\mathcal{C}_m,i}$. We assume that any robot maintain efficient communication with the others and that they have simultaneous access to each other's weight. With this protocol, a robot which is blocked by complex obstacles or is too far from the task region will share this information by decreasing its own weight. The splitting coefficients $\alpha_{m,i}$ are updated according

to the following consensus-like protocol [20]:

$$\gamma_{m,i} = \begin{cases} 1, & \text{if } 0 < \alpha_{m,i} < 1. \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

$$\dot{\alpha}_{m,i} = \beta(\sum_{j \neq i}(W_i - W_j)\gamma_{m,i}) \tag{10}$$

β is a constant scalar used for adjusting the $\dot{\alpha}_{m,i}$ to prevent it from being too large or too small. This splitting protocol has two main properties:

1) The sum of derivatives of the splitting coefficients $\sum_i \dot{\alpha}_{m,i} = 0$. This guarantees $\sum_i \alpha_{m,i} = 1$ (when $\sum_i \alpha_{m,i}$ is initialized as 1).
2) This protocol implies that if a robot i has lower weight than the others in average, its splitting coefficients $\alpha_{m,i}$ will reduce. In the opposite if robot i has higher weight than average, its splitting coefficient $\alpha_{m,i}$ will increase. Splitting coefficient will stop when it reaches 0 or 1.

Here we only discuss the case where robots collaborating on a single task m. We assign the splitting coefficients to the robots at the initial stage to make sure the splitting coefficients satisfy $\sum_i \alpha_{m,i} = 1$. For those robots locating outside of the safety set (i.e. with initial positions in $h_{\mathcal{C}_m} < 0$), the splitting coefficients are set as 0 since they can never finish task m before the deadline t_m even moving at their fastest speed. For robots with initial location within the safety set, splitting coefficients are initialized using the following method:

$$\alpha_{m,i} = \begin{cases} \dfrac{h_{\mathcal{C}_m}(x_i)}{\sum\limits_{x_j \in \mathcal{S}} h_{\mathcal{C}_m}(x_j)} & if\ x_i \in \mathcal{S} \\ 0 & if\ x_i \notin \mathcal{S} \end{cases} \tag{11}$$

where $\mathcal{S} = \{x_i \mid h_{\mathcal{C}_m}(x_i) > 0, 1 \leq i \leq N\}$. Such initialization method ensures that the initial splitting coefficients $\alpha_{m,i}$ is proportional to $h_{\mathcal{C}_m}(x_i)$. Intuitively, robots with higher $h_{\mathcal{C}_m}$ (i.e. higher potential to complete the task) will be initialized with higher splitting coefficients and vice versa.

From the definition, for each task m at least one of the $\alpha_{m,i}(i \in [1,N])$ must be strictly positive. Therefore enforcing $h_{\mathcal{C}_m,i} > 0$ for all robot i with positive splitting coefficients will ensure satisfaction of the task. While robot i with $\alpha_i = 0$ will discard such constraint, enabling the multi-robot system to collaborate effectively. In Section 2.3, we will present a detailed introduction of the dynamic splitting protocol for manipulating the splitting coefficients. So in our CBF-based method, the control input:

$$u \in U \text{ subject to } \dot{h}_{\mathcal{C}_m,i} + \alpha(h_{\mathcal{C}_m,i}) \geq 0 \tag{12}$$

B. Control Cost

While designing control inputs for robots to ensure safety and complete tasks before deadline, additional control objectives will also be taken into consideration. We choose appropriate cost functions J_i for robots and find a control input that minimizes J_i to accomplish specific control objectives:

1) $J_i = \|u\|^2$

The objective can account for an energy cost associated to the control input,

2) $J_i = -\dot{h}_{\mathcal{C}_m,i}$

Make sure robot i satisfies the constraint as fast as possible.

3) $J_i = -\dot{H}$

If the robots has targets in its sensor range, we will define a utility objective of the form $H(x) = \int_\Omega r(x,p)f(p)dp$, where $f(p)$ is a distribution representing the targets (Dirac's deltas can be used for discrete targets), $r(x,p)$ is a sensor footprint function (e.g. the sensitivity as a function of distance) . Objectives of this form are common in distributed coverage [21]. The term $\dot{H}$ is then added to the objective of J_i so as to maximize robots' ability of detecting targets.

Note that different constraints can be combined together with priority to achieve trade-offs.

$$J_i(u_i) = p_1 u_i^T u_i + p_2(-\dot{h}_{\mathcal{C}_m,i}) + p_3(-\dot{H}) \tag{13}$$

where $p_1, p_2 and p_3$ are priority for the above control cost. Then the control input u_i for robot i at time t is computed by minimizing cost function J_i while satisfying the constraint of the ZCBF.

$$\min_{u_i \in U} J_i(u_i) \quad \text{subject to} \quad \dot{h}_{\mathcal{C}_m,i} + \alpha(h_{\mathcal{C}_m,i}) \geq 0 \tag{14}$$

C. Control Objectives

Apart from

1) Connectivity Maintenance

The maximum distance between every two robots to maintain reliable information exchange is denoted as R_0. Equivalently, we requires all the robots to stay within a circle $\{(x,y) \mid (x-x_0)^2 + (y-y_0)^2 < r_0^2\}$. We define $h_0(x) = r_0^2 - (x-x_0)^2 - (y-y_0)^2$. To guarantee connectivity maintenance we need to enforce

$$\dot{h}_0(x_i) + \alpha(h_0(x_i)) > 0 \tag{15}$$

2) Obstacle Avoidance

In most cases collision avoidance is a control objective that can not be ignored. Assume each robot has onboard sensors to detect obstacle k. In order to allow for collision avoidance, we define

$$h_{\mathcal{D},i}(x_i,t) = d_{\mathcal{D}}(x_i,t) \tag{16}$$

where $d_{\mathcal{D}}(x_i,t)$ denotes the shortest distance from the obstalce and is given by onboard sensors. By enforcing $h_{\mathcal{D}}(x_i,t)$ is a ZCBF, we can guarantee collision avoidance ($h_{\mathcal{D}}(x_i,t) > 0$). Then we need to solve the following constrainted optimization problem to get the optimal control law $u_i(t)$.

$$\begin{aligned} \min_{u_i \in U} \quad & J_i(u_i, x_i) \\ \text{s.t.} \quad & \dot{h}_0 + \alpha(h_{\mathcal{C}_m,i}) \geq 0, \\ & \dot{h}_{\mathcal{C}_m,i} + \alpha(h_{\mathcal{C}_m,i}) \geq 0, \\ & \dot{h}_{\mathcal{D},i} + \alpha(h_{\mathcal{D},i}) \geq 0, \\ & \dot{x}_i = f(x_i) + g(x_i)u_i, \\ & \|\dot{x}_i\| \leq v_{max} \end{aligned} \tag{17}$$

Assume the robot motion with respect to the global cartesian coordinate frame is described by

$$\dot{x}_i = u_i \tag{18}$$

This assumption is always feasible when we do not take full-body dynamics and kinematics into consideration. The resulting control law can be regarded as a high-level controller. A lower-level controller can be implemented to transfer control law u_i into input signals for motors or velocity of wheels, but that is beyond consideration in this paper. With this assumption, the above optimization problem is a quadratic programming problem similar to [16].

We next discuss how to transfer equation (18) into a quadratic programming problem.

$$\begin{aligned} \min_{u_i \in U} \quad & \frac{1}{2} {u_i}^T P u_i + q^T u_i \\ \text{s.t.} \quad & G u_i \leq h \end{aligned} \tag{19}$$

First we consider the control cost in equation (13). $p_1 u_i^T u_i$ can be rewritten as $\frac{1}{2}{u_i}^T P u_i$, where $P = diag(2p_1, 2p_1)$. $p_2(-\dot{h}_{\mathcal{C}_m,i}) = p_2 \alpha_{m,i}(1 + \frac{x_i^T - [x_m \ y_m]}{v_{max}\, d_{\mathcal{C}_m}(x_i)} u_i)$, and it can be rewritten as $q_1^T u_i$, where $q_1 = -p_2 \alpha_{m,i} \frac{x_i - [x_m \ y_m]^T}{v_{max}\, d_{\mathcal{C}_m}(x_i)}$. $p_3(-\dot{H}(x_i)) = -p_3 \frac{dH}{dx}\dot{x}_i = -p_3 \frac{dH}{dx} u_i$, and we rewrite it as $q_2^T u_i$, where $q_2 = -p_3 \frac{dH}{dx}^T$.

Then we consider the constraints in equation (18). The class $\mathcal{K}$ function α we use here could be, for instance, $\alpha(r) = r^2$. $\dot{h}_0 + \alpha(h_{\mathcal{C}_m,i}) \geq 0 \Leftrightarrow -2(x^T - [x_0 \ y_0])u_i + h_0^2(x_i) \geq 0$, which can be rewritten as $G_1 u_i \leq h_1$, where $G_1 = 2(x^T - [x_0 \ y_0]), h_1 = h_0^2(x_i)$. Constraint $\dot{h}_{\mathcal{C}_m,i} + \alpha(h_{\mathcal{C}_m,i}) \geq 0 \Leftrightarrow (-1 - \frac{x_i^T - [x_m \ y_m]}{v_{max}\, d_{\mathcal{C}_m}(x_i)} u_i) + h_{\mathcal{C}_m,i}^2 \geq 0$, can be written as $G_2 u_i \leq h_2$, where $G_2 = \frac{x_i^T - [x_m \ y_m]}{v_{max}\, d_{\mathcal{C}_m}(x_i)}, h_2 = \frac{h_{\mathcal{C}_m,i}^2(x_i)}{\alpha_{m,i}} - 1$. Constraint $\dot{h}_{\mathcal{D},i} + \alpha(h_{\mathcal{D},i}) \geq 0 \Leftrightarrow \frac{\partial h_{\mathcal{D},i}}{\partial x_i} u_i + \frac{\partial h_{\mathcal{D},i}}{\partial t} + h_{\mathcal{D},i}^2 \geq 0$, can be written as $G_3 u_i \leq h_3$, where $G_3 = \frac{\partial h_{\mathcal{D},i}}{\partial x_i}, h_3 = \frac{\partial h_{\mathcal{D},i}}{\partial t} + h_{\mathcal{D},i}^2$.

Here the optimization problem in (18) is formulated as a quadratic programming problem in (20), where $P = diag(2p_1, 2p_1), q = q_1 + q_2, G = [G_1^T \ G_2^T \ G_3^T]^T, h = [h_1 \ h_2 \ h_3]^T$. The algorithm for single task is given as follows.

In the previous subsections we discuss the case with a single task m. Here we extend the control approach to deal with multiple tasks.

IV. Simulation and Result

We consider a system of $N = 5$ robots with dynamics $\dot{x}_i = u_i$. Each robot is modeled as a circle with radius $r = 15$. We set the initial position of the robot as $x_1 = [30 \ 350]^T$, $x_2 = [180 \ 350]^T$, $x_3 = [120 \ 250]^T$, $x_4 = [400 \ 300]^T$, $x_5 = [400 \ 180]^T$. We require that at least one robot should arrive at the task region $\mathcal{C} = \{[X_i \ Y_i]^T \mid (X_i - 250)^2 + (Y_i - 250)^2 < 50^2\}$ before the deadline $t = 15s$. The maximum velocity of the robots is $v_{max} = 15m/s$. The weights for different control objectives are set as $w_1 = 0.5, w_2 = 3, w_3 = 0$.

Scenario 1:

Scenario 2:

Scenario 3:

Algorithm 1: Control Algorithm for Single Task

Input : $((x_m, y_m), r_m, t_m)$ - task; $((x_0, y_0), r_0)$ - maximum communication range; (p_1, p_2, p_3) - priority; v_{max} - maximum velocity

Output: control laws and state trajectories of robots

Initialize $\alpha_{m,i}$ according to equation (12);

Active group $\mathcal{A} \leftarrow$ *robot*1, 2, ..., N;

while *task m is not satisfied* **do**

  for $i \in \mathcal{A}$ **do**

    ; `// use parallel computing`

    if $h_{\mathcal{C}_m}(x_i, t) < 0$ **then**

      remove i from $\mathcal{A}$

    else

      compute u_i by solving equation (19);

    end

  end

  for $i \in \mathcal{A}$ **do**

    ; `// use parallel computing`

    update *state* x_i according to equation (18);

  end

  for $i \in \mathcal{A}$ **do**

    ; `// use parallel computing`

    update *splitting coefficients* $\alpha_{m,i}$ according to equation (10);

  end

end

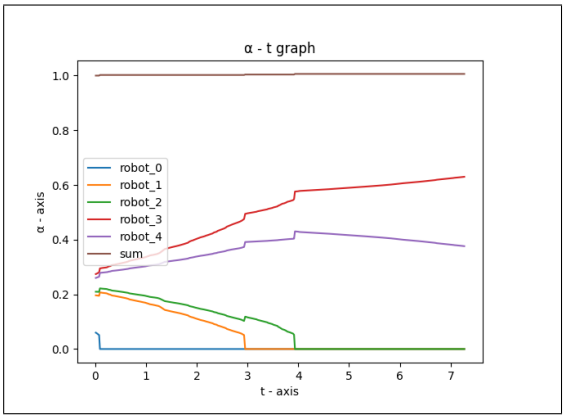

Fig. 1. Splitting coefficients of the robots

The initial positions of the robot is shown in Fig 1. The outer circle represents the boundary of the time-varying safety set $h_{\mathcal{C}_m}(x_i, t) > 0$. The inner circle represents the boundary of the constant task region $\mathcal{C}_m$.The resulting paths of the robots is presented in Fig 2. Fig 3. illustrates how the splitting coefficients change when the multi-robot system is working on the shared task. From figure 3 we can see that the sum of all the splitting coefficients is 1. This simulation verifies that the our splitting protocol design is correct and also demonstrate its effectiveness in multi-robot collaboration.

Another simulation in which we consider obstacle avoid-

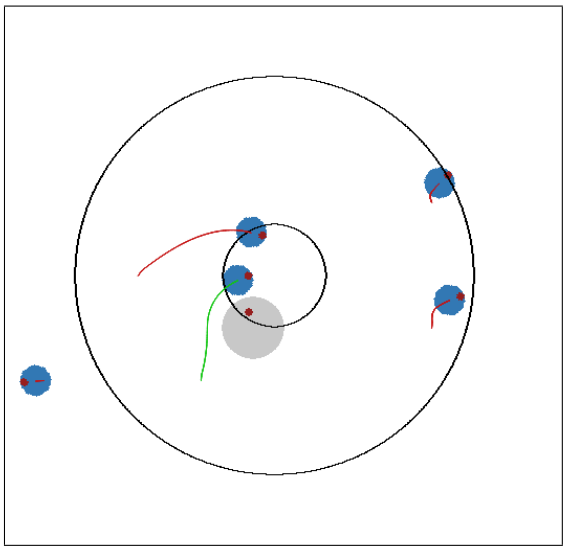

Fig. 2. Obstacle avoidance

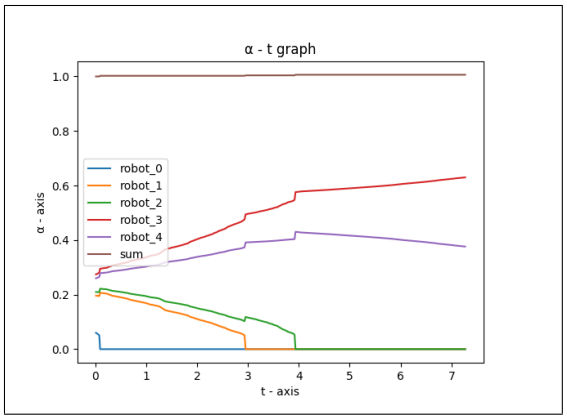

Fig. 3. Splitting coefficients of the robots

ance problems is presented here(see Fig.4 and Fig.5, the grey circle is a static obstacle). In fact, any robots can also be considered as a moving obstacle. By constructing $h_{x_{i,j}} = (X_j - X_i)^2 + (Y_j - Y_i)^2 - R_0^2$ and enforcing $h_{x_{i,j}} > 0$ we can actually achieve collision avoidance among robots. However, since we adopt constant weights in equation (19), we can not guarantee the trade-off between control objectives of $\dot{h}_{\mathcal{C}_m,i}$) and $\dot{h}_{\mathcal{D},i}$ is always reasonable.

V. CONCLUSIONS AND FUTURE WORKS

One of the merits of this approach is that instead of relying on complicated task allocation algorithm, by simply designing a control input that satisfy a inequality constraint we can make sure the deadline will be satisfied. This method also allows distributed control using information obtained by individual robot and enables collaboration among robots without any central coordination.

In this paper we investigate the problem of enforcing completeness of tasks with deadlines in an unknown environment. The innovation of this paper is that we design a consensus-like splitting protocol along with splitting coefficients and dynamical weights, which have been used to split the constraints to individual robot to enforce the tasks while at the same time, enable robots to collaborate on tasks. Adopting control barrier functionto force satisfaction of deadline is also an innovation. Since most of the research on such problem is focused on complicated task allocation algorithm, our method simply transform such problem into a optimization question, specifically quadratic program which is usually easy to be solved.

REFERENCES

[1] Feng Xiao, Long Wang, Jie Chen, and Yanping Gao. Finite-time formation control for multi-agent systems. *Automatica*, 45(11):2605–2611, 2009.

[2] Mazda Ahmadi and Peter Stone. A multi-robot system for continuous area sweeping tasks. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 1724–1729. IEEE, 2006.

[3] Andreas Kolling and Stefano Carpin. Multi-robot surveillance: an improved algorithm for the graph-clear problem. In *2008 IEEE International Conference on Robotics and Automation*, pages 2360–2365. IEEE, 2008.

[4] Andrew Howard. Multi-robot simultaneous localization and mapping using particle filters. *The International Journal of Robotics Research*, 25(12):1243–1256, 2006.

[5] Philipp Koch, Stefan May, Michael Schmidpeter, Markus Kühn, Christian Pfitzner, Christian Merkl, Rainer Koch, Martin Fees, Jon Martin, Daniel Ammon, et al. Multi-robot localization and mapping based on signed distance functions. *Journal of Intelligent & Robotic Systems*, 83(3-4):409–428, 2016.

[6] Lingzhi Luo, Nilanjan Chakraborty, and Katia Sycara. Distributed algorithm design for multi-robot task assignment with deadlines for tasks. In *2013 IEEE International Conference on Robotics and Automation*, pages 3007–3013. IEEE, 2013.

[7] Jose Guerrero, Gabriel Oliver, and Oscar Valero. Multi-robot coalitions formation with deadlines: complexity analysis and solutions. *PloS one*, 12(1):e0170659, 2017.

[8] Lingzhi Luo, Nilanjan Chakraborty, and Katia Sycara. Distributed algorithms for multirobot task assignment with task deadline constraints. *IEEE Transactions on Automation Science and Engineering*, 12(3):876–888, 2015.

[9] Franco Blanchini. Set invariance in control. *Automatica*, 35(11):1747–1767, 1999.

[10] Stephen Prajna and Ali Jadbabaie. Safety verification of hybrid systems using barrier certificates. In *International Workshop on Hybrid Systems: Computation and Control*, pages 477–492. Springer, 2004.

[11] Stephen Prajna. Barrier certificates for nonlinear model validation. *Automatica*, 42(1):117–126, 2006.

[12] Stephen Prajna, Ali Jadbabaie, and George J Pappas. Stochastic safety verification using barrier certificates. In *2004 43rd IEEE Conference on Decision and Control (CDC)(IEEE Cat. No. 04CH37601)*, volume 1, pages 929–934. IEEE, 2004.

[13] Peter Wieland and Frank Allgöwer. Constructive safety using control barrier functions. *IFAC Proceedings Volumes*, 40(12):462–467, 2007.

[14] Muhammad Zakiyullah Romdlony and Bayu Jayawardhana. Uniting control lyapunov and control barrier functions. In *53rd IEEE Conference on Decision and Control*, pages 2293–2298. IEEE, 2014.

[15] Aaron D Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. *arXiv preprint arXiv:1903.11199*, 2019.

[16] Aaron D Ames, Jessy W Grizzle, and Paulo Tabuada. Control barrier function based quadratic programs with application to adaptive cruise control. In *53rd IEEE Conference on Decision and Control*, pages 6271–6278. IEEE, 2014.

[17] Xiangru Xu, Paulo Tabuada, Jessy W Grizzle, and Aaron D Ames. Robustness of control barrier functions for safety critical control. *IFAC-PapersOnLine*, 48(27):54–61, 2015.

[18] Lars Lindemann and Dimos V Dimarogonas. Control barrier functions for signal temporal logic tasks. *IEEE control systems letters*, 3(1):96–101, 2018.

[19] Gennaro Notomista and Magnus Egerstedt. Constraint-driven coordinated control of multi-robot systems. In *2019 American Control Conference (ACC)*, pages 1990–1996. IEEE, 2019.

[20] Reza Olfati-Saber, J Alex Fax, and Richard M Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[21] Francesco Bullo, Jorge Cortes, and Sonia Martinez. *Distributed control of robotic networks: a mathematical approach to motion coordination algorithms*, volume 27. Princeton University Press, 2009.