2017 IEEE International Conference on Robotics and Automation (ICRA)
Singapore, May 29 - June 3, 2017

Distributed Multi-Robot Coordination for Dynamic Perimeter Surveillance in Uncertain Environments

Alexander Jahn[1], Reza Javanmard Alitappeh[2], David Saldaña[3],
Luciano C. A. Pimenta[1], Andre G. Santos[2], Mario F. M. Campos[3]

Abstract— **In this work, multiple robots circulate around the boundary of a desired region in order to create a virtual fence. The aim of the this fence is to avoid internal or external agents crossing through the delimited area. In this paper, we propose a distributed technique that allows a team of robots to plan the deformation of the boundary shape in order to escort the safe region from one place to a goal. Our proposal is composed of two parts. First, we present a distributed planning method for the dynamic boundary. We model the resulting plan as a twice differentiable function. Second, we use the obtained function to guide the robot team, where every member uses only local information for the controller. The robots distribute themselves along the time-varying perimeter and patrol around it. We show in simulation how the robots behave in partially/totally unknown environments with static obstacles.**

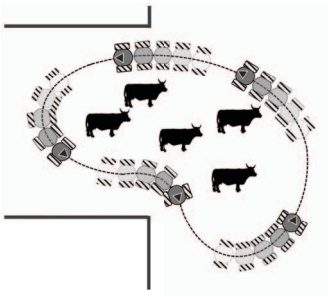


Fig. 1. Five robots surround a region of interest (*crocked egg*) with a group of cows. The robots move counterclockwise to create a virtual fence.

I. Introduction

This work extends the solution of the dynamic perimeter surveillance problem previously defined in our earlier work [1] by a decentralized and also uncertain scenario. As defined in [1], this problem consists of two sub-problems: i) planning the motion of a virtual dynamic abstraction (shape) from a given initial configuration to a final configuration in an environment populated with static obstacles; and ii) coordinating a team of N homogeneous robots to keep circulating the perimeter of the abstraction as it moves and deforms accordingly to the planned trajectory. Efficient solutions are of great interest in a scenario in which a group of agents (animals or humans for example) should be escorted from a given point to another one in the environment under the guard of a group of robots.

An example of this type of problem is illustrated in Fig. 1, where 5 robots are traversing the perimeter of a shape to create a virtual fence in an animal herding task. In [2] the virtual fence is a simple square of constant area and if an animal leaves the designated shape, an acoustic signal tries to steer it back. Clearly a deformable shape, as proposed in this work, widens the possible applications.

Other possible applications are (1) Region protection: where a virtual, dynamic fence is created by a team of robots to protect another group of robots that are performing a task (like collecting data); (2) Escorting: moving and protecting the internal region from external agents; (3) Guiding: leading one or more agents without letting them to get out from the region of interest, *i.e.* herding animals. In all of these applications on-line planning is necessary in order to deal with real world issues such as dynamic environments. Centralized solutions as proposed in our previous work [1] are not applicable in most real world problems. Thus, in this work we consider a decentralized approach in the presence of uncertainty in the environment.

The problem of coordinating multiple agents to patrol a certain area autonomously has been treated previously in robotics literature. Some works focus on adapting to certain shapes but not patrolling the boundary [3]. In perimeter surveillance, some works focus on the circulation around a static closed curve [4]. The work of Franchi et al. [5] focuses specifically on patrolling static circular shapes in a distributed manner, where the agents achieve consensus to be equally distributed along the circle and to move with the same linear velocity. An extended version for 3D environments is shown in [6]. The study of surrounding an escaping target is studied in [7]. Furthermore, in a recent study a comprehensive review of multi-robot target detection and tracking was done by Robin and Lacroix [8]. In monitoring tasks, [9] proposes a velocity controller for multiple robots to supervise an environment. In this task, the robots move along the closed curve to reduce a field value over the environment.

As previously mentioned, in this work we deal simultane-

[1] A. Jahn and L. C. A. Pimenta are with the Graduate Program in Electrical Engineering (PPGEE), Universidade Federal de Minas Gerais. L. C. A. Pimenta is a Member of the National Institute of Science and Technology for Cooperative Autonomous Systems Applied to Security and Environment. E-mails: jahn.alexander@gmail.com, lucpim@cpdee.ufmg.br

[2] R. Javanmard and A. Santos are with the Departamento de Informática, Universidade Federal de Viçosa. E-mails: {rezajavanmard64@gmail.com, andreufv@gmail.com}

[3] D. Saldaña, and M. F. M. Campos are with the Computer Vision and Robotics Laboratory (VeRLab), Computer Science Department, Universidade Federal de Minas Gerais. E-mails: {saldana, mario}@dcc.ufmg.br

*The authors gratefully acknowledge the support of the Brazilian agencies CAPES, CNPq and FAPEMIG.

ously with the problem of planning the perimeter evolution and also the problem of controlling the motion of the individual robots to execute the perimeter surveillance.

Some of the main contributions in comparison to [1]:

- A novel on-line decentralized trajectory planning which combines DMA-RRT [10] and DIMT-RRT [11] that assumes only a *partially or an unknown environment* that can be locally sensed by the robots.
- Application of a *decentralized* controller, that assumes only local information.
- Evaluation of the system performance in simulations with localization noise.

The paper is structured as follows: We formally state the problem in Section II. The proposed decentralized RRT-like algorithm for trajectory planning is explained in Section III. In Section IV, we report our decentralized motion controller. We present the experimental evaluation in Section V, and finally, our conclusions are discussed in Section VI.

II. Problem Setup

Imagine the goal is to escort a group of slow, reactive agents (which might be animals or other autonomous vehicles) through an environment $\mathcal{E} \subset \mathbb{R}^2$ which contains static obstacles $\mathcal{O} \subset \mathcal{E}$ which might be partially known or completely unknown to the team of robots. The *region of interest* (also called abstraction or shape) $\Omega \subset \mathcal{E}$ is bounded by its perimeter $\partial\Omega$, the choice for the shape of this region depends on the application and is out of the scope of this paper. Possible application is to optimize the livestock grazing on the croplands in the case of animal agents. As shown in [12] robots can actively influence the natural flocking behavior.

We model the boundary of the region as a parametric elliptical-shaped curve in order to determine its possible configurations based on n parameters:

Definition 1: A *parametric shape* $\Phi : [0, 2\pi] \times S \times SE(2) \rightarrow \mathcal{E}$ defines a closed curve in $\mathbb{R}^2$, parameterized by the angle $\phi \in [0, 2\pi)$, and a set $S \times SE(2)$ defining the configuration of the parametric shape (henceforth referred to as *shape*). The vector of parameters of the shape, denoted by $\mathbf{q}$, is composed of two parts: S, which contains properties of the shape i.e. size, radius, horizontal and/or vertical size; and $SE(2)$ or $\mathbb{R}^2 \times S^1$, which indicates the translation and rotation of the shape in the $\mathbb{R}^2$ workspace.

We summarize the parameter vector of the shape as $\mathbf{q} = [\mathbf{q}_C^T, \mathbf{q}_T^T]^T$. The sets $\mathbf{q}_C \in S$ and $\mathbf{q}_T \in \mathbb{R}^2 \times S^1$ show the aforementioned decomposed configuration of the shape. For instance, a shape in the form of an ellipse with fixed orientation aligned with the reference frame may be described as

$$\Phi(\phi, \mathbf{q}) = \begin{bmatrix} q_1 \cos(\phi) \\ q_2 \sin(\phi) \end{bmatrix} + \begin{bmatrix} q_3 \\ q_4 \end{bmatrix}, \tag{1}$$

for all $\phi \in [0, 2\pi)$, where the vector $\mathbf{q}_C = [q_1, q_2]^T$ denotes the semi-axes of the ellipse, centered at $\mathbf{q}_T = [q_3, q_4]^T$. The decomposition of the shape configuration in $\mathbf{q}_C$, $\mathbf{q}_T$ helps us to distinguish between the properties of the shape with respect to a local frame attached to it, called body frame, and its motion with respect to the global reference frame (see Fig. 2) respectively.

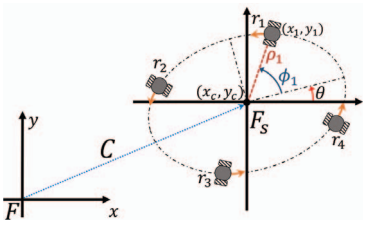


Fig. 2. Representation of an ellipse in the local frame, rotated by θ.

New: We represent the free configuration space for the shape abstraction as $\mathcal{Q}_{free} = \mathcal{Q} \backslash \mathcal{QO}$, where $\mathcal{Q}$ is the configuration space and $\mathcal{QO}$ is the obstacle set in the configuration space. In order to move the shape in the environment, a collision free trajectory is defined as a function $\tau : [t_0, t_f] \rightarrow \mathcal{Q}_{free}$ that maps a finite time interval $[t_0, t_f]$ into $\mathcal{Q}_{free}$ starting at $\mathbf{q}_0$ and ending at the final configuration $\mathbf{q}_f$.

In this work, different from [1], we assume the obstacle set $\mathcal{QO}$ to be unknown or partially known. Thus we have a dynamic $\mathcal{Q}_{free}$. The shape dynamics along the trajectory is defined as follows:

Definition 2: The *polar boundary function* $\gamma : [0, 2\pi] \times \mathbb{R} \rightarrow \mathcal{E}$ describes how the shape Φ changes its configuration $\mathbf{q}$ along time $t \in [t_0, t_f]$, and satisfies the equations:

$$\begin{aligned} \gamma(\phi, t) &= \Phi(\phi, \tau(t)), \\ \rho &= g(\phi, \mathbf{q}_C), \end{aligned} \tag{2}$$

where ρ is the radius coordinate related to the polar coordinates with respect to the body frame, since the shape is star shape, and g is a differentiable function. It should be mentioned that $\tau(t)$ defines the trajectory of the complete configuration $[\mathbf{q}_C^T, \mathbf{q}_T^T]^T$. According to this definition, the shape considering the body frame in polar coordinates for an ellipse is defined as:

$$\rho(\phi, \mathbf{q}_C) = \frac{q_1 q_2}{\sqrt{(q_2 \cos(\phi))^2 + (q_1 \sin(\phi))^2}}, \tag{3}$$

where $\mathbf{q}_C = [q_1 \ q_2]^T$ is the semi-axes of the ellipse in the two-dimensional space.

We can finally state the problems to be solved.

*Problem 1 (**Dynamic Shape Motion Planning**):* Compute a trajectory function $\tau(t)$ in order to move the shape Φ, from an initial configuration $\mathbf{q}_s$ to a final configuration $\mathbf{q}_f$. This trajectory must be continuously differentiable and move along the free, dynamic configuration space $\mathcal{Q}_{free}$.

*Problem 2 (**Robots Motion Control on Dynamic Shape**):* Given a boundary function $\gamma(t)$, control the robots in order to make them circulate along the dynamic boundary in a uniform way.

III. Motion Planning for a Dynamic Shape

As multiple robots are involved, each robot needs an individual (Algorithm 2) and an interaction component (Al-

gorithm 3) in the same spirit as in [10], but with the modification that a changing environment is considered, in this work the team of robots is cooperatively solving the *same* path planning problem, and instead of CL-RRT (Closed Loop Rapidly Exploring Random Trees [13]) a modified version (Algorithm 1) of the efficient DIMT-RRT algorithm (Double-Integrator Minimum Time RRT) from [11] is applied such that only nodes which might improve the committed plan are added to the tree. A sampling based technique is used because the configuration space is high dimensional even with a simple shape like an ellipse. If the application requires further freedom in the deformation of the region of interest the problem becomes increasingly complex, which makes other approaches computationally inefficient [14].

We assume the desired abstraction evolves according to the following dynamics:

$$\ddot{\mathbf{q}} = \mathbf{u}, \quad \mathbf{u} \in \mathcal{U}, \tag{4}$$

where $\mathbf{u}$ is an acceleration input from the compact space $\mathcal{U} \subset \mathbb{R}^n$. We define the state vector as $\mathbf{x} = [\mathbf{q}^T, \dot{\mathbf{q}}^T]^T \in \mathcal{X}$, where $\mathcal{X} \in \mathbb{R}^{2n}$ is the state space. Thus, the trajectory function $\tau(t)$ is generated by the following motion equation:

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)), \quad \mathbf{x}(t_0) = \mathbf{x}_0, \quad \mathbf{x}(t_f) = \mathbf{x}_f \tag{5}$$

where, $\mathbf{x}_0 = [\mathbf{q}_0^T, \dot{\mathbf{q}}_0^T]^T$ is the initial state, $\mathbf{x}_f = [\mathbf{q}_f^T, \dot{\mathbf{q}}_f^T]^T$ is the final state, and f is a continuous function, $\mathbf{x}(t) \in \mathcal{X}$ in time $t \in \mathbb{R}_{\geq 0}$. We denote the tree which contains the corresponding RRT nodes by $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ and $\mathcal{E}$ are the node and edge sets respectively. A single vertex v consists of: the current state $\mathbf{x}_v$; and a quadratic function τ_v that defines the trajectory from the parent of v to the current node v within a fixed time interval Δt.

The individual component proposed in Algorithm 2 describes the main loop of each of the robots. It is assumed that each robot has a certain range to sense changed obstacles. Further, as the communication is locally limited to the adjacent neighbors, it is assumed that each robot passes important small messages, like an environment change, quickly throughout the network (the robots abstract the sensor measurements). We assume each robot has 2 neighbors so that we define a ring communication topology (see Fig. 2). In contrast to the individual component in [10], in this work all team members are trying to solve the same navigation problem. The token owner receives the bids for the auction during the current interval and broadcasts the winner, which allows the respective robot to broadcast its solution τ^* to the team. If a previously unknown obstacle blocks the currently committed plan τ as detected in line 5, then a predefined safety action is applied, like maximum braking in line 6. It is important to note that this action is taken by a virtual shape, with the intention to maintain the boundary function within the free configuration space and continuously differentiable, this characteristic is important for the proof of convergence of the controller which is described later.

In line 21 the robot explores the configuration space by growing its tree (Algorithm 1). Where the method τ.`next_node()` returns the next node from the committed trajectory τ, this leads to periodical resetting of the tree on each of the agents to limit the data usage and simplify data management.

Algorithm 1 is based on the results of [11] and chosen due to its probabilistically completeness and the ability to handle double integrator dynamics. As evaluated in [11] the algorithm converges much quicker to the goal state, than the kinodynamic RRT approach utilized in our former work [1]. It starts with the initialization of the set of nodes $\mathcal{V}$ and edges $\mathcal{E}$ whenever the root $\mathbf{x}_{\text{init}}$ is reset in line 1. It chooses in line 4 a free configuration from $\mathcal{Q}_{free}$ or the goal-state $\mathbf{x}_f$ with a finite chance (5% is a common value [14]). In line 5 the closest neighbor is chosen based on the $DIMT$-function proposed in [11], which is defined as the Double Integrator Minimum Time necessary to connect a node on the tree with the sampled configuration, while maintaining limited input commands and configuration space. The `sat` function generates an expanded configuration $\mathbf{x}_{\text{new}}$, which is Δt away from the tree-node $\mathbf{x}_{\text{nearest}}$ while respecting the limitations of $\mathcal{X}$ and $\mathcal{U}$. For detailed information on the specific equations necessary for the `sat` and distance function $DIMT$ we would like to refer to [11]. The output of line 7 is the constant acceleration input a_{params} which connects $\mathbf{x}_{\text{nearest}}$ with $\mathbf{x}_{\text{new}}$ according to the dynamics in Eq. (4). The new node needs to be collision free as checked in line 8 and needs to have the potential to improve the committed plan τ as checked in line 11, where $cost(\mathbf{x}_{\text{new}})$ is defined as the time it takes to reach $\mathbf{x}_{\text{new}}$ from $\mathbf{x}_{\text{init}}$. Only then the node $\mathbf{x}_{\text{new}}$ is added to the tree structure. Once the goal or the random sample is reached at one of the iterations or if the expansions are blocked or cannot lead to improvements over an existing plan, the loop stops and returns to Algorithm 2.

Lastly, Algorithm 3 manages interaction between the agents. Four different kind of messages need to be handled. The first is τ_{new}, which corresponds to a an updated committed plan, the plan was either improved by one of the other robots or one robot sensed a blocking obstacle that forces the team to do an emergency stop. The second kind of message is the id of the `winner`, which gets the right to adjust the team-plan. The third message is the `bid`, which informs the current token holder that an improved solution is ready. The last message is an updated definition of the obstacles. The output of the combination of the three iterative processes is a continuously differentiable function $\tau(t)$, which has the following form:

$$\tau(t) = \begin{cases} \tau_1(t) & \text{if } t_0 \leq t < t_1 \\ \vdots & \\ \tau_m(t) & \text{if } t_{m-1} \leq t < t_m, \end{cases}$$

where each function $\tau_i(t)$, $i = 1, 2..., m$ is the second-order polynomial computed by `sat` in line 7 of Algorithm 1,

$$\tau_i(t) = \begin{bmatrix} a_1^i t^2/2 + b_1^i t + c_1^i \\ \vdots \\ a_n^i t^2/2 + b_n^i t + c_n^i \end{bmatrix}.$$

The coefficients a_j^i, b_j^i, c_j^i, $j = 1, 2..., n$ are computed for

Algorithm 1: *Modified DIMT – RRT*

1 **if** *new* $\boldsymbol{x}_{init}$ **then**
2 $V \leftarrow \mathbf{x}_{\text{init}}$
3 $E \leftarrow \emptyset$
4 Sample point $\mathbf{x}_{\text{rand}}$ from the free space $\mathcal{Q}_{free}$ or $\mathbf{x}_f$
5 $\mathbf{x}_{\text{nearest}} \leftarrow \arg\min_{v \in V}(DIMT(v, \mathbf{x}_{\text{rand}}))$
6 **while** *(True)* **do**
7 $\mathbf{x}_{\text{new}}, a_{\text{params}} \leftarrow sat(\mathbf{x}_{\text{nearest}}, \mathbf{x}_{\text{rand}}, \mathcal{X}, \mathcal{U}, \Delta t)$
8 **if** $\boldsymbol{x}_{new}$ *is in area of unavoidable collision with* $\mathcal{O}$ **then**
9 break
10 $t_{goal} \leftarrow DIMT(\mathbf{x}_{\text{new}}, \mathbf{x}_f)$
11 **if** $t_{goal} + cost(\boldsymbol{x}_{new}) > cost(\tau)$ **then**
12 break
13 $V \leftarrow V \cup \mathbf{x}_{\text{new}}$
14 $E \leftarrow E \cup a_{\text{params}}$
15 **if** $\boldsymbol{x}_{new}$ *is close to* $\boldsymbol{x}_f$ **then**
16 return Trajectory τ^* from $\mathbf{x}_{\text{init}}$ to $\mathbf{x}_{\text{new}}$
17 **if** $\boldsymbol{x}_{new}$ *reached* $\boldsymbol{x}_{rand}$ **then**
18 break
19 $\mathbf{x}_{\text{nearest}} \leftarrow \mathbf{x}_{\text{new}}$

Algorithm 2:
Extended DMA – RRT : Individual component

1 Initialize with $\mathbf{x}_0$
2 `HaveToken` $\leftarrow$ *false* except for one random robot
3 **while** *(Agent is active)* **do**
4 **if** *Changed obstacle detected* **then**
5 recheck_plan(τ,$\mathcal{O}$.update())
if τ *got invalid* **then**
6 Broadcast `e-stop trajectory` as τ_{new}
7 Broadcast updated $\mathcal{O}$ as $\mathcal{O}_{\text{new}}$
8 **if** τ^* *available// a solution returned from line 21* **then**
9 **if** *HaveToken* **then**
10 Broadcast improved τ^* as τ_{new}
11 **else**
12 `bid` $\leftarrow cost(\tau) - cost(\tau^*)$
13 Broadcast `bid`
14 **else**
15 `bid` $\leftarrow \varnothing$
16 Broadcast `bid`
17 **if** *HaveToken and bids are available* **then**
18 `winner` $\leftarrow$ robot with best `bid`
19 Broadcast `winner`
20 `HaveToken` $\leftarrow$ *False*
21 $\tau^* \leftarrow$ Grow DIMT-RRT-like (Algorithm 1) with root at τ.`next_node`()
22 Execute τ

Algorithm 3:
Extended DMA – RRT : Interaction component

1 **while** *(Agent is active)* **do**
2 Listen for messages
3 **if** *Received* τ_{new} **then**
4 $\tau \leftarrow \tau_{\text{new}}$
5 **if** *robot is* `winner` **then**
6 `HaveToken` $\leftarrow$ *True*
7 **if** *Received* `bid` *message* **then**
8 Update sender's `bid`
9 **if** *Received* $\mathcal{O}_{new}$ **then**
10 $\mathcal{O} \leftarrow \mathcal{O}_{\text{new}}$

the ith time-slice with $t \in [t_{i-1}, t_i]$ and $t_i - t_{i-1} = \Delta t$, considering that the accelerations in $\mathbf{u}$ are constant during that time interval Δt. As t_m is our final time, we have $t_f = t_m$. Therefore, the boundary function γ is well defined by inserting the calculated trajectory τ in Eq. (2). It has a continuous differentiable motion, because the n parameters of γ are defined for m sequential time-intervals by quadratic equations for the whole mission time $t \in [t_0, t_f]$.

In the next section, a decentralized controller is proposed that uses this trajectory of the abstraction in order to make the team of robots circulate it.

IV. Decentralized Motion Controller

Given the boundary function $\gamma(t, \phi)$, which defines the perimeter of the deformable shape in the environment $\mathcal{E} \subset \mathbb{R}^2$, now we need to address the problem of controlling – in a decentralized way – the team of n robots with locations $\mathcal{P} = \{\mathbf{p}_1, \cdots, \mathbf{p}_n\}$, $\mathbf{p} \in \mathbb{R}^2$ to circulate along the perimeter.

We assume a ring network topology, where each robot may communicate only with its two adjacent neighbors. Therefore, robot R_i communicates with R_{i-1} and R_{i+1}. In Fig. 2, we exemplify a boundary with four robots, where the neighbors of robot 3 are $\{2, 4\}$. We also illustrate the global frame (F), and the body frame, which is attached to the shape (F_s). We show the location of robot R_i with respect to the global frame by:

$$\mathbf{p}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}.$$

As explained before, the trajectory, $\tau(t)$, of the shape is obtained by applying an RRT-like decentralized algorithm which was described in Section III. We then have the motion of the shape considering the motion of the body frame parameters in $\mathbf{q}_C$ and the translation and rotation expressed in $\mathbf{q}_T$. Let $C = [x_c, y_c]^T$ be the position of the origin of the body frame. Let also $R(\theta)$ be the rotation matrix associated with rotation of the body frame by an angle θ (see Fig. 2). Now, a desired robot position on the boundary can be described in the global frame as:

$$\mathbf{p}_i^* = \begin{bmatrix} x_i^* \\ y_i^* \end{bmatrix} = R(\theta)\rho_i^* \begin{bmatrix} \cos(\phi_i^*) \\ \sin(\phi_i^*) \end{bmatrix} + C, \tag{6}$$

where ρ_i^* is the desired polar radius given by $g(\phi_i^*, \mathbf{q}_C)$ according to Eq. (2) and ϕ_i^* is the desired polar angle. Next, we show our decentralized controller.

A. Robot Controller

We assume the holonomic fully actuated model for each robot,

$$\dot{\mathbf{p}}_i = \begin{bmatrix} \dot{x}_i \\ \dot{y}_i \end{bmatrix} = \mathbf{w}_i, \tag{7}$$

where $\mathbf{w}_i$ is the control input of the ith robot. Based on the controllers proposed in [5], [15], we propose the control law:

$$\begin{aligned} \mathbf{w}_i = \dot{R}(\theta)\rho_i \begin{bmatrix} \cos(\phi_i) \\ \sin(\phi_i) \end{bmatrix} + R(\theta)\dot{\rho}_i \begin{bmatrix} \cos(\phi_i) \\ \sin(\phi_i) \end{bmatrix} \\ + R(\theta)\rho_i \begin{bmatrix} -\sin(\phi_i) \\ \cos(\phi_i) \end{bmatrix} \dot{\phi}_i + \dot{C}, \end{aligned} \tag{8}$$

where

$$\dot{\rho}_i = \dot{\rho}^*(\phi_i, \mathbf{q}_C) + K_\rho(\rho^*(\phi_i, \mathbf{q}_C) - \rho_i) \tag{9}$$

$$\dot{\phi}_i = \omega + K_\phi(\phi_i^* - \phi_i), \tag{10}$$

and K_ρ, $K_\phi > 0$ are constants gains, and ω is the desired velocity to move around the shape in steady state. The function $\rho_i^*(\phi_i, \mathbf{q}_C)$ is given by $g(\phi_i, \mathbf{q}_C)$, thus:

$$\dot{\rho}_i^*(\phi_i, \mathbf{q}_C) = \frac{\partial g}{\partial \phi_i}\dot{\phi}_i + \frac{\partial g}{\partial \mathbf{q}_C} \cdot \dot{\mathbf{q}}_C,$$

where $\dot{\phi}_i$ is given by Eq. (10) and $\dot{\mathbf{q}}_C$ along with $\dot{R}(\theta)$ and $\dot{C}$ are determined by the output of the RRT algorithm. In (10), the desired angle ϕ_i^* is given by $\phi_i^* = \bar{\phi}_i$, where $\bar{\phi}_i$ is the mean between the angle of the two immediate neighbors R_{i-1} and R_{i+1} as used in [15]. This acts as an implicit collision avoidance and allows each robot to regularly visit each point on the perimeter. If one or more robots fail, the network needs to be rewired automatically to maintain a ring topology.

B. Convergence analysis

Given the closed loop induced by (8) we can guarantee that:

$$\lim_{t\to\infty} \mathbf{p}_i(t) \in \gamma(t, \phi),$$

which indicates the convergence of the robot to the perimeter. The first and the last terms in (8) are feed-forward terms which allow us to think of the motion as happening in the body frame. By using (9) we can write:

$$\dot{e}_i(t) + K_\rho e_i(t) = 0,$$

where $e_i = \rho_i^* - \rho_i$. Therefore, the agent converges to the perimeter as $t \to \infty$.

Also, given Eq. (10), the robot will be uniformly distributed along the boundary as:

$$|d\phi_i^{i+1}| = |d\phi_i^{i-1}| = \frac{2\pi}{N} \text{ as } t \to \infty,$$

where N is the number of robots and $d\phi_i^j$ is the angle difference between agents j and i. The proof of this angle convergence can be found in [15].

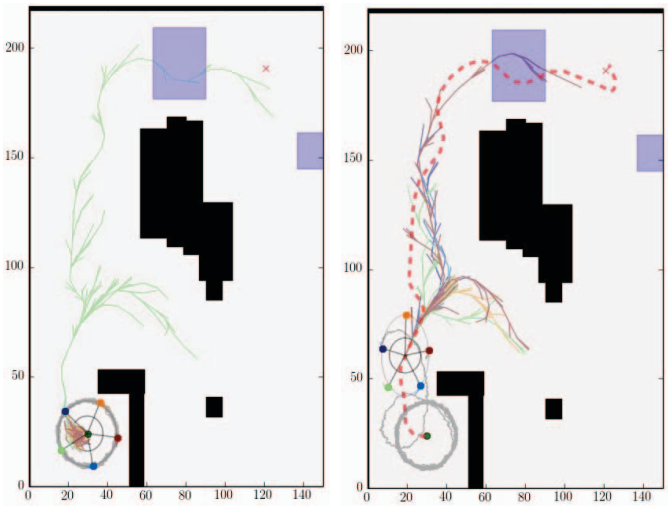
V. Experiments and Results

In our experiment we consider an ellipsoidal shape, a group of 5 holonomic robots with limited velocity, and an environment with partially known obstacles. The ellipse has a fixed area and can vary in its width to height proportion. The simulation uses the Runge-Kutta method (time-step 0.05s) to integrate the robot dynamics. Each robot is given the same time period (1s) during each interval for planning before the auction takes place. The position information of the adjacent robots is available at each time-step. To study the robustness of the controller proposed in Section IV noise affects the localization of each agent. The adapted noise-model is taken from [16]. The maximum position error is 7.5m, the average error is 3m, and the distribution is normal. We study three types of errors during our experiments.

$$\mathbf{e}_\rho^i = \frac{1}{2}(\rho_i - \rho_i^*)^2, \mathbf{e}_{\delta\rho}^i = \frac{(\rho_i - \rho_i^*)}{\rho_i^*}, \mathbf{e}_\phi^i = \bar{\phi}_i - \phi_i. \tag{11}$$

Error $\mathbf{e}_\rho^i$ is the actual radius compared to the desired one, $\mathbf{e}_{\delta\rho}^i$ is the relative error of the radius variable, and $\mathbf{e}_\phi^i$ tends to zero when the robots are evenly distributed. We want to show that all errors tend to zero as the robots converge to the boundary and are uniformly distributed.

Fig. 3 demonstrates (i) the trajectory of a robot over the execution of the algorithm (gray solid line) (ii) the RRT trees of each robot (colored straight lines with the root close to the center of the shape), and (iii) the currently committed trajectory τ (red dashed line). Note that two neighbor robots will maintain an equal angular distance. In the figure we show a line connecting the robot's current position on the curve to the center of the shape. Fig. 4 (a) shows how the

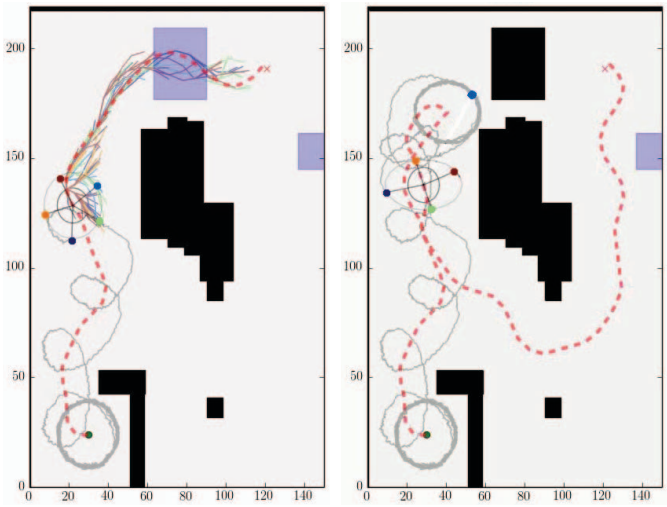


(a) $t = 40s$ (b) $t = 50s$

Fig. 3. The shape starts as a circle and squeezes into an ellipse to escape the fenced start area. The black obstacles are known to the team of agents, the two blue ones are unknown. The goal is in the top right corner. The trajectory of robot 0 is shown as the gray continuous line. Each robot's search tree is plotted the same color as the agent.

team progresses, it senses an obstacle as it gets into range. In Fig. 4 (b) the trajectory was re-planned and additionally

one of the robots had a malfunction, but as the ring-topology was rewired, the remaining team converges again to a new uniform distribution along the boundary without a change in the parameters of the controller. Fig. 5 shows how the



(a) $t = 65s$ (b) $t = 115s$

Fig. 4. Dynamic perimeter surveillance for five robots. In (a) the committed trajectory (red dashed line) was improved during the execution. In (b) two events happened: (i) the top obstacle was sensed, emergency stop, re-planning a new trajectory and started to follow it, and (ii) one of the robots had a malfunction, but as the topology changed accordingly.

errors $\mathbf{e}^i_\phi$, $\mathbf{e}^i_{\delta\rho}$, and $\mathbf{e}^i_\rho$ tend to go to zero during the first 20 seconds. The video of this simulation can be found in [Video link: `https://youtu.be/LSiS_p3NGqE`].

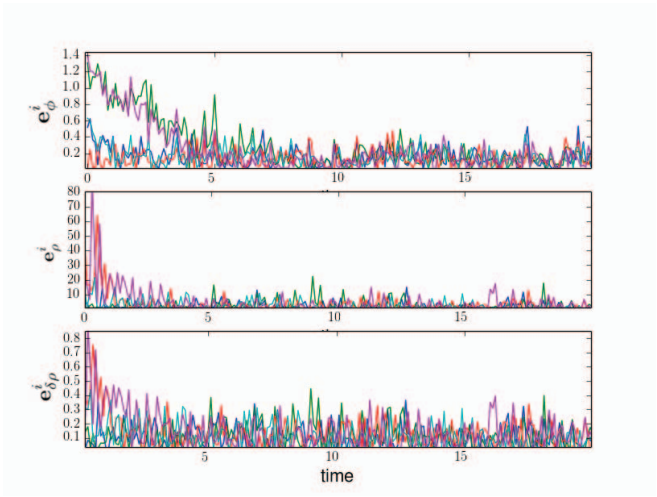


Fig. 5. Simulation results for the first 20 seconds with noisy localization. At the beginning the robots have to distribute themselves along the boundary and converge to it.

VI. Conclusions

In this paper we proposed a distributed approach to move a deformable shape with a team of robots attached to it over an environment populated with partially known or completely unknown obstacles that can be sensed by the robots. The problem is decomposed in two parts. Each robot computes the trajectory for the shape between the start and end point in the environment. The best plan gets accepted as the committed plan in an auction-like, decentralized procedure. During the execution the robots continue to explore the configuration space to find improved plans. Second we control the robots to circulate around the perimeter of the shape in such a way that they keep an equal distance in angle while only communicating with their immediate neighbors. We validated our proposal by a simulation for an elliptical shape with noise in the localization.

The current limitations of this work are: consideration of static obstacles, consideration of a specific class of shapes (ellipses or more generally star-shapes).

References

[1] D. Saldana, R. J. Alitappeh, L. C. A. Pimenta, R. Assuncao, and M. F. M. Campos, "Dynamic perimeter surveillance with a team of robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 5289–5294.

[2] Z. Butler, P. Corke, R. Peterson, and D. Rus, "From Robots to Animals: Virtual Fences for Controlling Cattle," *The International Journal of Robotics Research*, vol. 25, no. 5-6, pp. 485–508, 2006.

[3] C. C. Cheah, S. P. Hou, and J. J. E. Slotine, "Region-based shape control for a swarm of robots," *Automatica*, vol. 45, no. 10, pp. 2406–2411, 2009.

[4] L. C. A. Pimenta, G. A. Pereira, M. M. Gonçalves, N. Michael, M. Turpin, and V. Kumar, "Decentralized controllers for perimeter surveillance with teams of aerial robots," *Advanced Robotics*, vol. 27, no. 9, pp. 697–709, 2013.

[5] A. Franchi, P. Stegagno, M. Di Rocco, and G. Oriolo, "Distributed target localization and encirclement with a multi-robot system," in *7th IFAC Sym. on Intelligent Autonomous Vehicles*, 2010, pp. 151–156.

[6] A. Franchi, P. Stegagno, and G. Oriolo, "Decentralized multi-robot encirclement of a 3d target with guaranteed collision avoidance," *Autonomous Robots*, vol. 40, no. 2, pp. 245–265, 2015.

[7] G. Antonelli, F. Arrichiello, and S. Chiaverini, "The entrapment/escorting mission," *IEEE Robotics Automation Magazine*, vol. 15, no. 1, pp. 22–29, March 2008.

[8] C. Robin and S. Lacroix, "Multi-robot target detection and tracking: taxonomy and survey," *Autonomous Robots*, vol. 40, no. 4, pp. 729–760, 2016.

[9] S. L. Smith, M. Schwager, and D. Rus, "Persistent robotic tasks: Monitoring and sweeping in changing environments," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 410–426, April 2012.

[10] V. R. Desaraju and J. P. How, "Decentralized path planning for multi-agent teams with complex constraints," *Autonomous Robots*, vol. 32, no. 4, pp. 385–403, may 2012.

[11] T. Kunz and M. Stilman, "Probabilistically complete kinodynamic planning for robot manipulators with acceleration limits," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2014, pp. 3713–3719.

[12] R. Vaughan, N. Sumpter, J. Henderson, A. Frost, and S. Cameron, "Robot control of animal flocks," in *Proceedings of the 1998 IEEE ISIC and CIRA*. IEEE, pp. 277–282.

[13] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How, "Real-time motion planning with applications to autonomous urban driving," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 5, pp. 1105–1118, sep 2009.

[14] S. M. Lavalle, "Rapidly-exploring random trees: A new tool for path planning," Tech. Rep., 1998.

[15] M. M. Goncalves, L. C. A. Pimenta, and G. A. S. Pereira, "Coverage of curves in 3d with swarms of nonholonomic aerial robots," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 10 367 – 10 372, 2011.

[16] N. Bulusu, J. Heidemann, and D. Estrin, "GPS-less low-cost outdoor localization for very small devices," *IEEE Personal Communications*, vol. 7, no. 5, pp. 28–34, 2000.