Multi-Robot Path Planning with Maintenance of Generalized Connectivity

Yoann Solana, Michele Furci, Juan Cortés, Antonio Franchi

Preprint version, final version at http://ieeexplore.ieee.org/ 2017 IEEE 1st Int. Symp. on Multi-Robot and Multi-Agent Systems, Los Angeles, CA, USA

Multi-Robot Path Planning with Maintenance of Generalized Connectivity

Yoann Solana[1], Michele Furci[1], Juan Cortés[1], and Antonio Franchi[1]

***Abstract*— This paper addresses the problem of generating a path for a fleet of robots navigating in a cluttered environment, while maintaining the so called generalized connectivity. The main challenge in the management of a group of robots is to ensure the coordination between them, taking into account limitations in communication range and sensors, possible obstacles, inter-robot avoidance and other constraints. The Generalized Connectivity Maintenance (GCM) theory already provides a way to represent and consider the aforementioned constraints, but previous works only find solutions via locally-steering functions that do not provide global and optimal solutions. In this work, we merge the GCM theory with randomized path-planning approaches, and local path optimization techniques to derive a tool that can provide global, good-quality paths. The proposed approach has been intensively tested and verified by mean of numerical simulations.**

I. Introduction

Groups of robots have a high potential to achieve common tasks which are too complex for a single robot, with promising applications for exploration [1], [2], formation control [3], [4], transportation [5]–[7], assembly/construction [8], [9] and many more. The coordination of robots to achieve a task offers more flexibility, robustness and scalability because tasks can be reallocated between the robots if the resources are not sufficient. However coordinating a group of robots involves different problems such as maintaining a fleet formation, maintaining a good communication between robots, take into account sensors limitations, obstacles, occlusions and more.

Many problems related to multi-robot systems have been addressed through graph theory approaches [10]. For many practically relevant applications, one of the fundamental properties to be respected is graph connectivity. Connectivity is for example needed in order to let each robot communicate to a base station through multi-hops, or to ensure the convergence of several distributed computation and formation control methods [10] [11]. Graph theory provides a simple and accurate framework to describe the connectivity of a group of robots. In particular, tools such as the *Laplacian* matrix and its second smallest eigenvalue λ_2 are often used when dealing with multiple robots coordination [10]. In [12] for example, λ_2 is used in a static way, the connectivity graph remaining the same during the mission. In [13], the Generalized Connectivity Maintenance (GCM) is described, where it is shown how λ_2 can encode additional features such as collision avoidance, and how it can be used in a flexible way, that is to say the topology of the fleet can change during the mission by creating or deleting links without breaking the connectivity of the fleet.

One of the main problem of the aforementioned approaches, as of many other approaches in the literature, is that a local controller is used to drive the robots, which can result in non-optimal trajectories or can be trapped in local minima. Hence it arises the need of a global and possibly optimal method to generate paths for a fleet of robot to maintain the connectivity along the path.

This paper presents a new approach combining the GCM concept and motion planning algorithms. The goal is to compute paths for a fleet of robots that guarantee a good connectivity level all along them, together with other constraints or quality criteria, and avoiding local minima traps. Since a fleet of robots involves many degrees of freedom, our approach builds on sampling-based algorithms [14], which are able to deal with such highly-dimensional systems. This family of algorithms has been recently extended to consider a cost function associated with the configurations of the robot system [15], [16], aiming to compute not only feasible paths, but good-quality solutions. In this work, we apply one of the T-RRT algorithms [16], using a configuration cost function derived from GCM theory.

Algorithmic variants of cost-based planners have been proposed to compute (near-)optimal solutions when a cost function is defined to evaluate the quality of the paths [17]–[19]. Our preliminary tests with T-RRT* and AT-RRT [19] tend to show slow convergence toward the optimal solution due to the high-dimensionality of the constrained problems we address. Therefore, we prefer a different strategy, which relaxes the global optimality guarantee for the sake of computational efficiency.

Most often, a post-processing stage is applied to locally improve the paths provided by sampling-based planners [20]. The main goals are to reduce the path length and remove jerky motions. Here, we extend this idea to the optimization of a general cost function along the path. For this, we propose a variant of the Nudged Elastic Band (NEB) method [21], which was originally proposed to compute minimum energy paths of atomic/molecular systems. In a final stage, the optimized path is converted into a time-parameterized trajectory using a simple trajectory generation method.

The general approach is demonstrated in simulations, where a fleet of quadrotors has to reach a goal configuration in a cluttered environment, while maintaining the connectiv-

[1]LAAS-CNRS, Université de Toulouse, CNRS, Toulouse, France. yoann.solana@laas.fr michele.furci@laas.fr juan.cortes@laas.fr antonio.franchi@laas.fr

This work has been funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 644271 AEROARMS, and by the ANR JCJC MuRoPhen research project.

ity, i.e., keeping a certain level of communication, keeping a preferred inter-distance between robots, not colliding with obstacles, and other constraints described below.

II. Problem Definition

The motion planning problem addressed in this paper is the generation of a coordinated trajectory for a fleet of N_r robots to reach a final configuration q_f from an initial configuration q_i moving across a given 3D environment with obstacles. The particularity of the addressed problem is that the formation must be guaranteed with a minimum level of connectivity all along the coordinated trajectory, following the GCM theory described in Section IV. Furthermore, paths involving high level of multi-robot connectivity are preferred. Each robot is assumed to have configuration space defined by $\mathbb{R}^3 \times \mathbb{S}^1$, which models the robot 3D position and orientation about the vertical (gravity force) axis. This choice is general enough to encompass both ground mobile robots and flying robots, such as, e.g., quadrotors, for which position and orientation about the vertical axis are known to constitute a feedback linearizing (or flat) output [22], and can thus be controlled (and planned) independently.

Notation: The indexes $i, j \in \{1, \ldots, N_r\}$ are used to indicate the i-th and j-th robot, respectively. We denote with $p_i = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}^T \in \mathbb{R}^3$ the position of the i-th robot, and with $\psi_i \in \mathbb{S}^1$ its yaw angle. The non-negative number $d_{ij} = \left\| p_i - p_j \right\|$ denotes the distance between the i-th and j-th robots. As customary, we assume the environment to be populated by a finite number of obstacles, the index $k \in \mathbb{N}$ is used to define the k-th obstacle. We define d_{ijk} as the distance between the k-th obstacle and the segment connecting robot i with robot j, and with d_{ik} the distance between the k-th obstacle and the i-th robot. The path for the i-th robot, intended as sequence of robot configurations, is denoted with P_i. The path for the fleet is denoted with $P = [P_1, \cdots, P_{N_r}]$.

III. Methods Overview

Our approach to plan coordinated motions of a fleet of robots maintaining connectivity is based on three main components: Generalized Connectivity theory [13], cost-based motion planning (the T-RRT algorithm [16]), and path optimization (the NEB algorithm [21]). A synthesis of the interactions between these components is provided in Fig. 1.

The general idea is that the T-RRT algorithm takes as input the initial and final configurations, and explores the configuration space trying to find a good-quality path between them. For this, the algorithm iteratively constructs trees of random configurations q connected by local paths. For each fleet configuration, the GCM theory provides a cost $c(q)$ which depends on the generalized connectivity level of that configuration. This cost is used in the T-RRT framework to favor the explorations of low-cost regions of the configuration space, representing high degree of connectivity. The T-RRT returns a coordinated path P for the fleet. Although the solution provided by T-RRT is a good-quality path, it is sub-optimal, and possibly rough/jerky. Therefore, the NEB algorithm is used to improve the geometry and the cost of

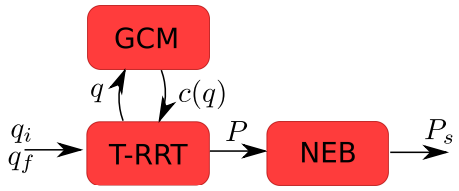


Fig. 1: Interconnections between the methods and algorithms used in this work.

the path P obtaining the final path P_s. The final path can later be converted into a time trajectory, using for example piecewise continuous functions to connect the configurations of the path. In the following sections, each method used is described with more details.

IV. The Generalized Connectivity

The GCM theory is based on some fundamental concepts of algebraic graph theory. Consider a system made of N_r agents: the presence of an interaction link among a pair of agents (i, j) is modeled by setting the corresponding elements $A_{ij} = A_{ji} = \{0, 1\}$ in the adjacency matrix $A \in \mathbb{R}^{N_r \times N_r}$, with $A_{ij} = A_{ji} = 0$ if no information can be exchanged at all, and $A_{ij} = A_{ji} = 1$ otherwise. The matrix A defines the *interaction graph*. To this graph one can then associate another matrix, the Laplacian matrix $L = [\text{diag}\left(\sum_{i=1}^{N_r} A_{i1}, \ldots, \sum_{i=1}^{N_r} A_{iN_r}\right\}) - A] \in \mathbb{R}^{N_r \times N_r}$, which is always positive semidefinite, i.e., all the eigenvalues are real and nonnegative. Furthermore the smallest eigenvalue is always zero, i.e., we have $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_{N_r}$. A well known result in algebraic graph theory is that the graph is connected if and only if $\lambda_2 > 0$. The reader can refer to [10] for more details,

As shown in [13], this model can be easily extended to explicitly consider more sophisticated *agent sensing/communication models* representing the actual (physical) ability to exchange mutual information because of the agent relative state. We recall here the main concepts briefly. Instead of discrete values $(0, 1)$ for the communication model, one can consider smooth scalar functions γ_{ij}. Once the functions γ_{ij} have been chosen, one can exploit them as *weights* on the inter-agent links, i.e., by setting $A_{ij} = \gamma_{ij}$. In this way the value of λ_2 becomes a (smooth) measure of the graph connectivity and, in particular, a (smooth) function of the system state (e.g., of the agent and obstacle relative positions). In [13] was proposed to augment the previous definition of the weights $A_{ij} = \gamma_{ij}$ as follows:

$$A_{ij} = \alpha_{ij} \beta_{ij} \gamma_{ij}. \tag{1}$$

The weight $\beta_{ij} \geq 0$ is meant to account for additional inter-agent *soft requirements* that should be preferably realized by the individual pair (i, j) (e.g., for formation control purposes, $\beta_{ij}(d_{ij})$ could have a unique maximum at some desired inter-distance $d_{ij} = d_0$ and $\beta_{ij}(d_{ij}) \to 0$ as d_{ij} deviates too much from d_0). Failure in complying with β_{ij} will lead to a disconnected edge (i, j) and to a corresponding decrease of λ_2, but will not (in general) result in a global loss of connectivity. The weight $\alpha_{ij} \geq 0$ is meant to represent *hard*

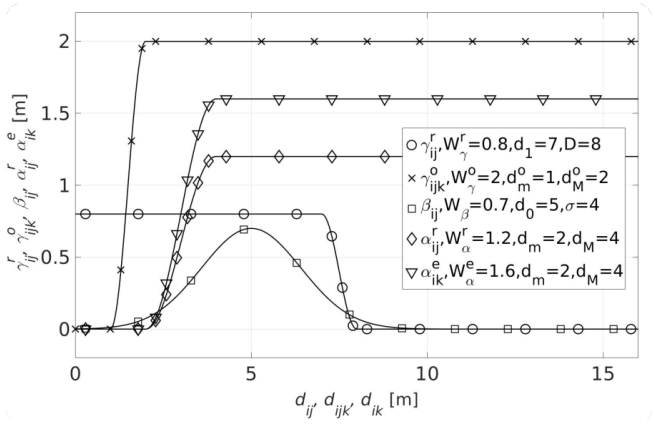


Fig. 2: Weight functions examples.

requirements that must be necessarily satisfied by agents i or j with some desired accuracy. A straightforward example is obstacle or inter-agent collision avoidance: whatever the task, collisions with obstacles or other agents must be mandatorily avoided. Failure in complying with a hard requirement will result in a null i-th row (and i-th column) in the adjacency matrix A, necessarily leading to a disconnected graph ($\lambda_2 \to 0$). Ensuring graph connectivity ($\lambda_2 > 0$) at all times will then automatically enforce fulfillment of all the mandatory behaviors encoded within α_{ij}.

The weight model for the specific problem in exam is chosen to encode the following requirements:

1) the distance d_{ij} is lower than a maximum range D,
2) the line of sight between robots i and j is not occluded,
3) robots i and j keep a preferred inter-distance $d_0 < D$,
4) every robot must avoid collision with other robots by having an inter-distance greater than d_m
5) every robot must avoid collision with obstacles.

The weights for requirements 1, 2, 3, 4 are respectively γ^a_{ij}, γ^b_{ij}, β_{ij} and α^*_{ij} as defined in [13]. In addition to that we define the weight for requirement 5 as:

$$\alpha^e_{ik}(d_{ik}) = \begin{cases} 0 & d_{ik} \leq d_m \\ \frac{k^e_\alpha}{2}\left(1-\cos\left(\mu\left(d_{ik}-d_m\right)\right)\right) & d_m < d_{ik} < d_M \\ W^e_\alpha & d_{ik} \geq d_M \end{cases} \tag{2}$$

with $k^e_\alpha \in \mathbb{R} > 0$ a constant weight for the function. Some example of weights functions can be seen in Figure 2.

The reader can refer to [13] to see how the functions $\gamma_{ij}, \beta_{ij}, \alpha_{ij}$, the Laplacian matrix and λ_2 can be computed from the proposed weights.

V. Cost-based motion planning: T-RRT

The T-RRT algorithm [16] extends the RRT framework [23] for handling more general problems involving continuous cost spaces. T-RRT is applicable to path planning for complex mobile systems in cluttered 3D workspaces, when some additional cost criterion needs to be considered during the search process in order to compute low-cost solution paths. The configuration trees constructed by the algorithm efficiently explores valleys and saddle regions of the cost space, escaping from local minima traps thanks to a self-adaptive mechanism. Details on the T-RRT algorithm can be found in previous work [16], [19].

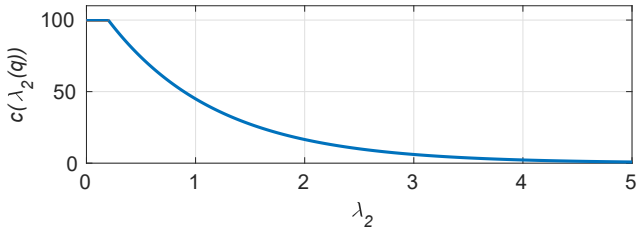


Fig. 3: Cost function for T-RRT with $c_{max} = 100$, $\lambda_2^{min} = 0.2$, $\xi = 1$.

A. The cost function

In the present context, the cost function used within the T-RRT algorithm has been defined aiming to favor multi-robot configurations with a high level of connectivity, i.e., having a high λ_2 value as defined in the GCM theory (see Section IV). The cost is a function of the connectivity eigenvalue λ_2, that is, because of the weight function definition, function of the configuration of the fleet q.

The cost of a configuration is defined as:

$$c(\lambda_2(q)) = \begin{cases} c_{max} & \text{if } \lambda_2(q) < \lambda_2^{min} \\ c_{max} e^{\xi\left(-\lambda_2(q)+\lambda_2^{min}\right)} & otherwise \end{cases} \tag{3}$$

with $c_{max} \in \mathbb{R}$ the maximum value of the cost function, λ_2^{min} the minimum acceptable level of connectivity and $\xi \in \mathbb{R}$ is tuned to modify the rate of the exponential function. An example of cost function with $c_{max} = 100$, $\lambda_2^{min} = 0.2$, $\xi = 1$ can be seen in Figure 3.

B. GCM shooting function

A preliminary analysis of the performance of our planner revealed that many configurations were rejected due to a violation of the generalized connectivity or to collisions between robots. Therefore, we developed a simple but effective configuration sampling function aiming to increase the success rate for tree extensions. This function performs random configuration sampling in such a way that connectivity requirements 1 and 4 are satisfied. To keep uniformity and to avoid favoring a particular type of configuration, the order to sample individual robot configurations is randomly sampled at each iteration. Then, the position of the first robot is randomly sampled in the environment. The position of the next robot in the list is sampled inside two spheres centered at the position of the previous robot. The radii of the spheres are the maximum range D and the minimum allowed distance d_m, as described in Section IV. The process is repeated until the configuration of all the robots is sampled.

VI. Path refinement: NEB

The paths produced by the T-RRT algorithm can be intricate and can locally cross high-cost regions. The application of a smoothing and local cost optimization algorithm helps to improve the quality of the solutions. For this, we use a

variant of the Nudged Elastic Band (NEB) method [21], which was designed to find paths of minimal energy for atomic/molecular systems. This method consists in iteratively deforming a straight-line path connecting an initial and final states by applying forces to a set of intermediate states in order to deform the path towards lower cost region while keeping the smoothness. At each iteration, a state is chosen and moved according to the computed forces. In short, the forces applied to intermediate states are $\mathbf{F_1}$, $\mathbf{F_2}$, and $\mathbf{F_3}$, where $\mathbf{F_1}$ is a force proportional to the cost gradient, thus moving the current state towards lower cost region, $\mathbf{F_2}$ is a force to balance the current state between the previous and next ones, and $\mathbf{F_3}$ is a force to locally straighten the path. The reader is invited to read [21] for more information about the original method.

The following subsections explain how the original NEB method has been modified to better fit our needs. The following notation is used. P_i represents the path of the robot i and $\mathbf{p}_i^c$ the current position on P_i. The positions $\mathbf{p}_i^p$ and $\mathbf{p}_i^n$ are the previous and next position w.r.t. $\mathbf{p}_i^c$ on P_i. The local tangent to $\mathbf{p}_i^c$ is $\tau_c = \frac{\mathbf{p}_i^n - \mathbf{p}_i^p}{|\mathbf{p}_i^n - \mathbf{p}_i^p|}$. A vector with subscript $\mathbf{V}_{||}$ means that only the parallel component to τ_c is considered, while $\mathbf{V}_{\perp}$ means that only the perpendicular component to τ_c is considered.

A. *Modified* $\mathbf{F}_1$

The original $\mathbf{F}_1 = -\nabla c(\mathbf{p}_i^c)|_{\perp}$ uses a cost gradient to move the current point $\mathbf{p}_i^c$. Due to complexity in computing analytically or numerically the cost gradient in our case, a local search using a discrete set of directions is considered. Since $\mathbf{F}_1$ (Fig.4a) lies in the red plane perpendicular to τ_c, a discrete number of vectors $\mathbf{d}_s$ lying on that plane are considered. The length $\overline{\delta}$ and the angle between them $\overline{\beta}$ can be set as parameters. The cost at each new configuration driven by $\mathbf{d}_s$ is computed, and the vector $\mathbf{d}_s$ with the lowest cost is kept. If no cost is lower than the current cost, then $\mathbf{d}_s$ in a null vector. Finally, the force is defined as:

$$\mathbf{F}_1 = K_1 \mathbf{d}_s \tag{4}$$

where K_1 is a weight set by the user.

B. *Modified* $\mathbf{F}_2$

The effect of the original springs force $\mathbf{F}_2$ can be too high, possibly moving the current point $\mathbf{p}_i^c$ too far from the expected result $\mathbf{p}_i^{c_{std}}$ (Fig.4b in green). To correct this problem, $\mathbf{F_2}$ is bounded in amplitude by a weight K_s to ensure that the resulting current point $\mathbf{p}_i^{c_{gcm}}$ does not lead to a worst local path (Fig.4b in red).

$$K_{is} = \begin{cases} \frac{min(d_{cn||}, d_{pc||})}{d_{pn}} + \frac{1}{2} & \text{if} \quad \|\mathbf{F_{is}}\| > d_{pn} \\ 1 & \text{otherwise} \end{cases} \tag{5}$$

Where d_{cn}, d_{pc}, d_{pn} are distances between $\mathbf{p}_i^c, \mathbf{p}_i^p, \mathbf{p}_i^n$ points.

The modified spring force becomes:

$$\mathbf{F_2} = K_2 K_{is} \mathbf{F}_{is||} \tag{6}$$

where the raw spring force is $\mathbf{F_{is}} = (\mathbf{p_i^p} - \mathbf{p_i^c}) + (\mathbf{p_i^n} - \mathbf{p_i^c})$ and K_2 is a weight set by the user.

C. *Modified* $\mathbf{F}_3$

The third component $\mathbf{F}_3$ is aimed to avoid acute angle ϕ between $(\mathbf{p}_i^p - \mathbf{p}_i^c)$ and $(\mathbf{p}_i^c - \mathbf{p}_i^n)$. It is defined as $\mathbf{F}_3 = f(\phi)\mathbf{F}_{is\perp}$, with $f(\phi)$ a factor limiting the effect of $\mathbf{F}_{is\perp}$.

In the original method, since the initial path is a straight line, the angle ϕ is in $[-\pi/2; \pi/2]$. We are in a more general case, where the angle can be in $]-\pi; \pi]$. Therefore, the range of $f(\phi)$ was increased to $]-\pi; \pi]$ and the effect of the force near to $\phi = 0$ was increased by replacing the input argument of $f(\phi)$ by $\phi_f(\phi) = \pi \sin(\frac{\phi}{2})$.

The modified straightening force becomes:

$$\mathbf{F_3} = K_3 f(\phi_f(\phi)) \mathbf{F}_{is\perp} \tag{7}$$

where K_3 is a weight set by the user, and

$$f(\phi_f(\phi)) = \frac{1}{2} + \frac{1}{2} \cos\left(\pi \cos\left(\frac{\pi \sin\left(\frac{\phi}{2}\right)}{2}\right)\right), \; -\pi < \phi < \pi. \tag{8}$$

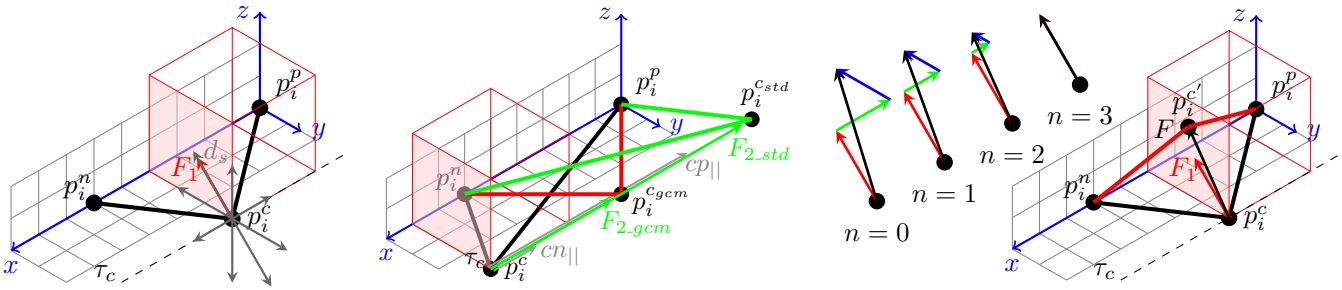
D. *Computation of the total force* $\mathbf{F}$

If $\mathbf{F}_2$ and $\mathbf{F}_3$ are too high compared with $\mathbf{F}_1$, $\mathbf{p}_i^c$ may move in regions with higher cost. A proposed algorithm (Algorithm 1) solves the problem decreasing the magnitude of $\mathbf{F}_2$ and $\mathbf{F}_3$ until a stop condition is reached (Fig. 4c). The first step (line 2) computes the total force $\mathbf{F}$. Then (line 3) $\mathbf{F}$ is applied to $\mathbf{p}_i^c$ to generate a new position $\mathbf{p}_i^{c'}$ and its cost $c_{c'}$ is computed. If the new cost $c_{c'}$ is lower than the current one (c_c) the algorithm stops and returns the total force $\mathbf{F}$. Otherwise, the magnitude of $\mathbf{F}_2$ and $\mathbf{F}_3$ are decreased (line 8) until the cost of the new configuration is lower than c_c and the number of maximum iterations n_{max} is not reached (Fig. 4c). If the maximum number of iterations is reached and the cost remains higher than c_c, then $\mathbf{F}$ is null. $\mathbf{F}$ is directly used to moved the state from $\mathbf{p}_i^c$ to $\mathbf{p}_i^{c'}$ (i.e. $\mathbf{F}$ defined the displacement vector).

Algorithm 1: Compute the total force in the NEB method

Data: $\mathbf{F}_1$, $\mathbf{F}_2$, $\mathbf{F}_3$, n_{max}, $\mathbf{p}_i^c$
Result: $\mathbf{F}$ the total force to apply on $\mathbf{p}_i^c$.

1 $c_c = cost(\mathbf{p}_i^c)$;
2 $\mathbf{F} = \sum_{i=1}^{3} \mathbf{F}_i$;
3 $\mathbf{p}_i^{c'} = \mathbf{p}_i^c + \mathbf{F}$;
4 $c_{c'} = cost(\mathbf{p}_i^{c'})$;
5 **if** $c_{c'} > c_c$ **then**
6 $\quad \mathbf{F}_{p2} = \mathbf{F}_2 / n_{max}$; $\mathbf{F}_{p3} = \mathbf{F}_3 / n_{max}$; $n = 0$;
7 $\quad$ **while** $c_{c'} > c_c$ *or* $n < n_{max}$ **do**
8 $\quad\quad \mathbf{F}_2 = \mathbf{F}_2 - \mathbf{F}_{p2}$; $\mathbf{F}_3 = \mathbf{F}_3 - \mathbf{F}_{p3}$;
9 $\quad\quad \mathbf{F} = \sum_{i=1}^{3} \mathbf{F}_i$;
10 $\quad\quad \mathbf{p}_i^{c'} = \mathbf{p}_i^c + \mathbf{F}$; $c_{c'} = cost(\mathbf{p}_i^{c'})$; $n = n + 1$;
11 $\quad$ **if** $c_{c'} > c_c$ **then**
12 $\quad\quad \mathbf{F} = \mathbf{0}$;



(a) Computation of $\mathbf{F}_1$: d_s with a length $\overline{\delta} = 1$ rotates with a step-size of $\overline{\beta} = \frac{\pi}{4}$ on the red plane.

(b) Computation of $\mathbf{F}_2$: The green (resp. red) line is the processed trajectory with the standard (1) (resp. adapted (2)) method.

(c) Illustration for the computation of $\mathbf{F}$ with $n_{max} = 3$ iterations. The red path, obtained for $n = 2$, is passing through the processed point $\mathbf{p}_i^{c'}$.

Fig. 4: Illustration of the adapted NEB method to improve path quality. The red cube C eases the 3D perception and it is defined from the current point $\mathbf{p}_i^c$ and the local tangent $\tau_\mathbf{c}$. The force $\mathbf{F}_1$ is on the red face of C. $\mathbf{F}_2$, $\mathbf{F}_3$ are on two edges of C. The black line is the initial path.

VII. Simulations and Results

In the following, we present the result of simulations using the proposed method. We simulate a fleet of 4 quadrotor UAVs that have to fly between two given configuration in cluttered scenarios while maintaining the connectivity along the whole trajectory. We propose two simulation scenarios: a small scenario for academic purpose, used to evaluate the performances of the algorithms and to tune the parameters, and a more realistic and large scenario for demonstration.

A. Academic Scenario and Parameter Settings

A relatively small but challenging scenario (illustrated in Fig. 5) was used to analyze the performance of the algorithms. The robots can fly inside a volume of $12 \times 5 \times 2m^3$ cluttered with obstacles. Obstacles sizes and positions were chosen to offer multiple solution paths through narrow and wide passages.

The algorithms were implemented in the Move3D software package [24]. Parameter setting for the algorithms are described next. Because of the randomized nature of $T-RRT$, ten simulations were run for each test to analyze the average performance of the planner. To ensure homogeneity, all the simulations were done on the same computer (i5-430M, 2.27GHz, 4GB DDR3 1066 MHz).

1) GCM settings: The parameters used for the GCM weights are presented in Table I.

2) T-RRT cost function: The parameters in the T-RRT cost function (eq. 3) are: $c_{max} = 100$, $\lambda_2^{min} = 0.2$, $\xi = 1$.

3) NEB settings: The following parameters were used for the NEB method: $K_1 = 1$, $K_2 = 1.5$, $K_3 = 0.5$, $\overline{\beta} = \pi/4$, $\overline{\delta} = 0.5$.

k_γ^a	D	d_1	k_γ^b	d_{min}^o	d_{max}^o	k_β	d_0	σ
1	2.5	2.4	1	0.05	0.3	1	2	2
k_α	d_{min_xy}	d_{max_xy}		d_{min_z}	d_{max_z}	k_α^e	d_m	d_M
1	0.5	0.7		0.8	1	1	0.25	0.5

TABLE I: Parameters for GCM weight functions.

B. Empirical performance alanysis

Results of a comparison between raw paths (generated by T-RRT), the smoothed paths after 10 and 100 iterations of NEB, and paths obtained with AT-RRT (any-time variant of T-RRT with asymptotic optimality guarantees) are presented in Table II and Figure 5. AT-RRT was run with a double stop condition, either a maximum of 10^6 iterations or 10^4 nodes in the tree. One can observe that 10 iterations of the NEB algorithm significantly reduce the cost of the path. Adding more iterations further reduces the cost and the jerkiness of the path, but the gain $g_{\#it} = \frac{\delta_{cost}}{t_{comp}}$ is less significant: $g_{10} = 47.47\ 10^{-3}$ for NEB_{10} and $g_{100} = 6.04\ 10^{-3}$ for NEB_{100}. Such a decrease of efficiency is due to a kind of saturation of the forces applied during the smoothing when the number of iterations increased. The lowest cost path is provided by AT-RRT, but the computing time is very high.

Figure 6 shows a comparison of the cost variation along the path for the different methods. One can clearly see that the smoothing method decreases cost variations significantly after 10 iterations of the NEB algorithms, being the gain lower for 100 iterations. Note also that, although the average cost is lower for AT-RRT than for T-RRT+NEB, the cost variation along the path is much more noisy for AT-RRT. This could be improved by applying NEB to the raw path of AT-RRT, at the expense of additional computing time.

Overall, these results show that, in the present context, the cost-based planning + local optimization strategy proposed in this paper is an effective approach to compute good-quality paths.

We also performed test to evaluate the efficiency of the GCM sampling technique. Table III shows a summary of the

	Raw	$+NEB_{10}$	$+NEB_{100}$	AT-RRT
Average Cost	11.5	9.25	8.64	7.58379
Comp. Time (s)	93.0	+47.4	+473.8	50956.9

TABLE II: Comparison between raw path from T-RRT, adding the NEB smoothing and the path computed by AT-RRT.

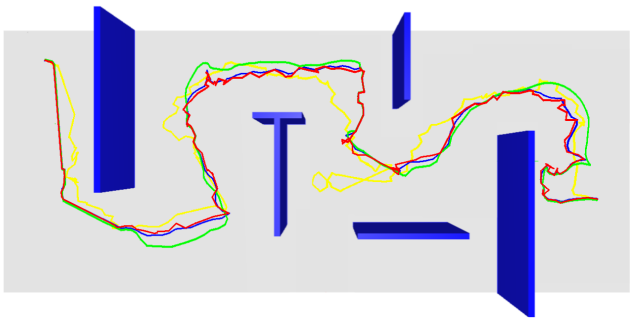


Fig. 5: Comparison of the solution paths for the simple scenario. Paths are shown for one of the four robot. The red line is the raw T-RRT path. The blue one was obtained by T-RRT and NEB smoothing with 10 iterations. The green one was obtained by T-RRT and NEB smoothing with 100 iterations. The yellow one was generated by AT-RRT (10^6 iterations).

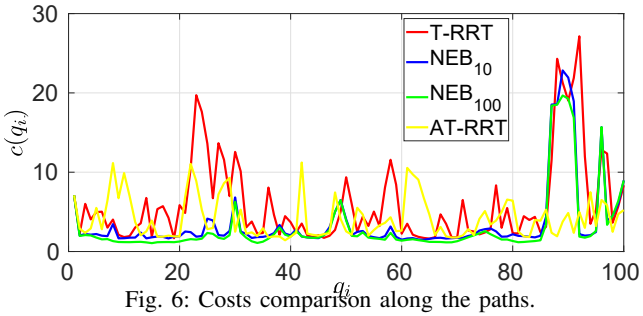


Fig. 6: Costs comparison along the paths.

	Avg. comp. Time (s)	Acceptance Rate (Avg %)
GCM SF	93	11
Standard SF	2560	0.4

TABLE III: Comparison between standard and GCM shooting functions.

results. Table III shows a very low rate of nodes accepted for the extension of the tree, 0.4%, using the standard method. Using the proposed GCM shooting function, the acceptance rate increased to 11%. This significantly speeds-up the growth of the exploration tree. Thanks to this sampling strategy, the average computing time to compute multi-robot paths goes down from more than 40 minutes to less than 2 minutes.

C. Realistic Scenario

The workspace consists of a $20 \times 20 \times 5m^3$ forest-like environment including many obstacles (trees, rocks, ...). The goal is to move the robots from one corner to the opposite corner of the map, while maintaining a good generalized connectivity of the fleet (fig 7). We can notice 3 macro areas in the scenario: on the bottom left, where the formation starts, it's a relatively empty area over a plateau; on the right another big plateau is cluttered of obstacles (rocks and trees) which offer very narrow passages; between the two plateau areas, a dry river offers a lower passage below the leafs (the big green rectangles). For a clearer comprehension, an attached video provides higher quality rendering and a FPV camera. The obstacles generate different types of problem for the planner

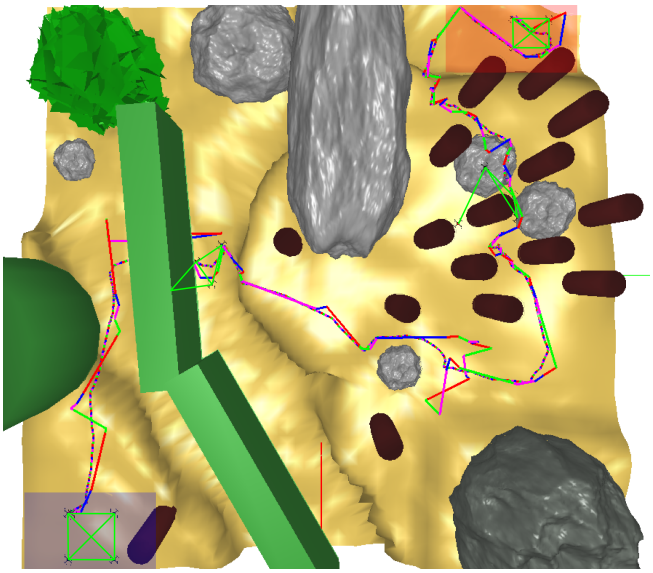


Fig. 7: The demonstrator scenario. The initial and final goal are the formations in the blue and red areas respectively. The jerky path is the T-RRT solution for one quadrotor. The smooth path is the result of NEB algorithm and the simulated dynamics of the quadrotor.

to maintain the connectivity. Sometimes the formation is able to keep the full connected graph (larger spaces), sometimes the graph can be not fully connected but still maintain the generalized connectivity as in the forest area, where some trees have to be circumnavigated by some robots in the formation, loosing the line of sight. Typical application for this kind of scenario include search and rescue missions, wildlife monitoring, fire or natural disaster monitoring.

The planner returns a trajectory in around 1 hour (including optimization) with the same computer used for the previous simulation. The large computing time is due to the very complex scenario, where in some areas very few configuration were feasible, and also to a lack of optimization of the code. We estimate at least a ten fold reduction in the computation time with an optimized implementation.

For a more realistic rendering of the solution, the path obtained by the T-RRT and NEB algorithms was converted into piece-wise constant velocity trajectories and given to a simulator implementing the dynamics and the low level control law for the fleet of four quadrotors.

Figure 7 shows the scenario, the raw and smoothed trajectory for one of the robots and some screenshots of the formation of the fleet along the generated path. We can notice how in some configuration, for example in the forest, the fleet doesn't guarantee a fully connected graph, but the generalized connectivity is maintained anyway.

Figure 8 presents the comparison of the cost along the trajectory for the T-RRT solution and the smoothed trajectory with NEB method for this scenario. The overall average cost of NEB is lower than the cost of plain T-RRT. Note that the high cost toward the end of the path cannot be avoided because robots need to navigate among the trees, which implies local disconnections.

Finally, Figure 9 shows the level of connectivity between pairs of robots along the trajectory, i.e. the values of the

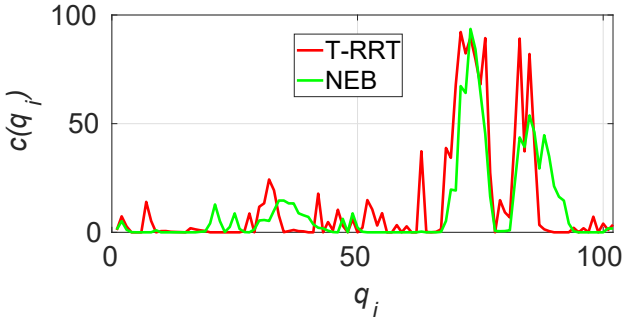


Fig. 8: The comparison of the cost with T-RRT only and smoothed with NEB for the realistic scenario.

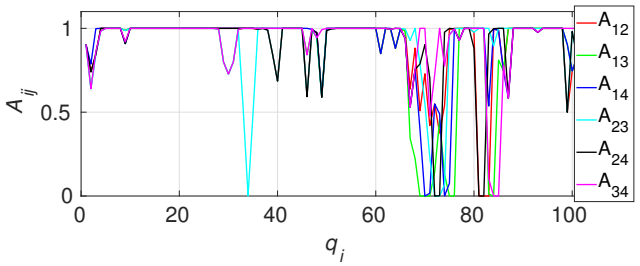


Fig. 9: Values of the upper triangular adjacency matrix. Values of A_{ij} close to zero indicate that two robots i, j are not connected.

upper triangular adjacency matrix. A value of zero indicates that the connectivity between two agents is lost (for example for lost line of sight) but he does not imply the formation not to be connected. In can be seen how in the second part of the trajectory, i.e. approaching the narrow passages of the forest some connections between robots get lost, mainly due to loss of line of sight.

A video of the simulation of this scenario can be found on the attached media.

VIII. Conclusion and Future Works

We have presented a new tool for planning paths for a fleet of robots to navigate in a cluttered environment while respecting the so called generalized connectivity constraints, which include obstacle avoidance, occlusion avoidance, maximum communication range and inter-robot avoidance. A previous approach based on GCM theory was limited by its local properties, which can lead to failure to find a solution because of local minima traps. The proposed approach solves this limitation. Our approach combines GCM, T-RRT and NEB methods. The combination of T-RRT and GCM allowed us to design a global planner that generates suitable paths for the fleet of robot respecting all the requirements for the connectivity, while the NEB algorithm is used to refine the path with the goal of obtaining smooth and low cost paths.

Simulations presented in this paper for a fleet of quadrotors in a cluttered environment show the strength of this approach on a particular use case. The method however is very general and can be applied to different heterogeneous robots and with different connectivity requirements.

Results presented in this paper have been obtained using a preliminary implementation of the proposed algorithms. Future work to improve their efficiency and generality will involve: an improvement of the cost function, the addition of more complex requirements that takes into account the orientation of the robots (for example if considered on-board sensors with a field of view), improvement of the GCM shooting function. Finally, we expect to perform experiments on a real fleet of robots using onboard sensors for mutual localization [25], [26] and estimation of the pairwise connectivity.

References

[1] W. Burgard, M. Moors, C. Stachniss, and F. Schneider, "Coordinated multi-robot exploration," *IEEE Trans. on Robotics and Automation*, vol. 21, no. 3, pp. 376–386, 2005.

[2] T. Nestmeyer, P. Robuffo Giordano, H. H. Bülthoff, and A. Franchi, "Decentralized simultaneous multi-target exploration using a connected network of multiple robots," *Autonomous Robots*, vol. 41, no. 4, pp. 989–1011, 2017.

[3] A. Franchi and P. Robuffo Giordano, "Online leader selection for improved collective tracking and formation maintenance," *IEEE Trans. on Control of Network Systems*, 2016.

[4] 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, 2016.

[5] I. Maza, K. Kondak, M. Bernard, and A. Ollero, "Multi-UAV cooperation and control for load transportation and deployment," *Journal of Intelligent & Robotics Systems*, vol. 57, no. 1-4, pp. 417–449, 2010.

[6] M. Manubens, D. Devaurs, L. Ros, and J. Cortés, "Motion planning for 6-D manipulation with aerial towed-cable systems," in *2013 Robotics: Science and Systems*, Berlin, Germany, May 2013.

[7] G. Gioioso, A. Franchi, G. Salvietti, S. Scheggi, and D. Prattichizzo, "The Flying Hand: a formation of UAVs for cooperative aerial tele-manipulation," in *2014 IEEE Int. Conf. on Robotics and Automation*, Hong Kong, China, May. 2014, pp. 4335–4341.

[8] F. Augugliaro, S. Lupashin, M. Hamer, C. Male, M. Hehn, M. W. Mueller, J. S. Willmann, F. Gramazio, M. Kohler, and R. D'Andrea, "The flight assembled architecture installation: Cooperative construction with flying machines," *IEEE Control Systems Magazine*, vol. 34, no. 4, pp. 46–64, 2014.

[9] N. Staub, M. Mohammadi, D. Bicego, D. Prattichizzo, and A. Franchi, "Towards robotic MAGMaS: Multiple aerial-ground manipulator systems," in *2017 IEEE Int. Conf. on Robotics and Automation*, Singapore, May 2017.

[10] M. Mesbahi and M. Egerstedt, *Graph Theoretic Methods in Multiagent Networks*, 1st ed., ser. Princeton Series in Applied Mathematics. Princeton University Press, 2010.

[11] D. Tardioli, A. Mosteo, L. Riazuelo, J. Villarroel, and L. Montano, "Enforcing network connectivity in robot team missions," *The International Journal of Robotics Research*, vol. 29, pp. 460–480, 2010.

[12] M. Ji and M. Egerstedt, "Distributed coordination control of multiagent systems while preserving connectedness," *IEEE Transactions on Robotics*, vol. 23, August 2007.

[13] P. Robuffo Giordano, A. Franchi, C. Secchi, and H. H. Bülthoff, "A passivity-based decentralized strategy for generalized connectivity maintenance," *The International Journal of Robotics Research*, vol. 32, no. 3, pp. 299–323, 2013.

[14] S. M. Lavalle, *Planning algorithms*. Cambridge university press, 2006.

[15] A. Ettlin and H. Bleuler, "Randomised rough-terrain robot motion planning," *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 5798–5803, 2006.

[16] L. Jaillet, J. Cortés, and T. Siméon, "Sampling-based path planning on configuration-space costmaps," *IEEE Transactions on Robotics*, vol. 26, no. 4, pp. 635–646, 2010.

[17] S. Karaman and E. Frazzoli, "Sampling-based Algorithms for Optimal Motion Planning," *International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[18] Y. Li, Z. Littlefield, and K. E. Bekris, "Asymptotically optimal sampling-based kinodynamic planning," *International Journal of Robotics Research*, vol. 35, pp. 528–564, 2016.

[19] D. Devaurs, T. Siméon, and J. Cortés, "Optimal path planning in complex cost spaces with sampling-based algorithms," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 415–424, 2016.

[20] R. Geraerts and M. H. Overmars, "Creating high-quality paths for motion planning," *The international journal of robotics research*, 2007.

[21] H. Jónsson, G. Mills, and K. W. Jacobsen, "Nudged elastic band method for finding minimum energy paths of transitions," in *CLassical and Quantum Dynamics in Condensed Phase Simulations*, 1998, pp. 385–404.

[22] V. Mistler, A. Benallegue, and N. K. M'Sirdi, "Exact linearization and noninteracting control of a 4 rotors helicopter via dynamic feedback," in *10th IEEE Int. Symp. on Robots and Human Interactive Communications*, Bordeaux, Paris, France, Sep. 2001, pp. 586–593.

[23] S. M. LaValle and J. J. Kuffner, "Rapidly-exploring random trees : Progress and prospects," in *Algorithmic and Computational Robotics: New Directions*, B. Donald, K. Lynch, and D. Rus, Eds. Boston: A.K. Peters, 2001, pp. 293–308.

[24] "Move3d motion planning," https://www.openrobots.org/wiki/move3d.

[25] A. Franchi, G. Oriolo, and P. Stegagno, "Mutual localization in multi-robot systems using anonymous relative measurements," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1302–1322, 2013.

[26] P. Stegagno, M. Cognetti, G. Oriolo, H. H. Bülthoff, and A. Franchi, "Ground and aerial mutual localization using anonymous relative-bearing measurements," *IEEE Trans. on Robotics*, vol. 32, no. 5, pp. 1133–1151, 2016.