2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Congress Center Hamburg
Sept 28 - Oct 2, 2015. Hamburg, Germany

# Multi-robot navigation in formation via sequential convex programming

Javier Alonso-Mora, Stuart Baker and Daniela Rus*

*Abstract*— **This paper presents a method for navigating a team of robots in formation in 2D and 3D environments with static and dynamic obstacles. The method is local and computes the optimal parameters for the formation within a neighborhood of the robots, allowing for reconfigurations, when required, by considering a set of target formations. The method consists of first computing the largest collision-free convex polytope in a neighborhood of the robots, followed by a constrained optimization via sequential convex programming where the optimal parameters for the formation are obtained. The robots navigate towards the target collision-free formation with individual local planners that account for their dynamics. The approach is efficient and scalable with the number of robots and performed well in simulations with a large team of quadrators and in experiments with two mobile manipulators carrying a rigid object.**

## I. Introduction

Muti-robot teams can be employed for various tasks, such as surveillance, inspection, or automated factories. In these scenarios, robots may be required to navigate in formation, for example for maintaining a communication network, for collaboratively handling of an object, for surveilling an area or to improve navigation.

Multi-robot navigation in formation has received extensive attention in the past. In this work we leverage efficient optimization techniques, namely quadratic programming, semidefinite programming and (non-linear) sequential quadratic programming, each one employed at different stages of our method for local motion planning in formation. These techniques provide good computational efficiency, local guarantees and generality.

Given a set of target formation shapes, our method optimizes the parameters (such as position, orientation and size) of the multi-robot formation in a neighborhood of the robots. The method guarantees that the team of robots remains collision-free and makes progress towards the goal. A simplified global planner, only waypoints for the formation center are required, can use this method to navigate the group of robots from an initial location to a final location. But a human may also provide the global path for the robots, or a desired velocity for the formation, and the robots will adapt their configuration automatically.

### A. Contribution

The main contribution of this paper is a scalable and efficient method for navigation of a team of robots while reconfiguring their formation to avoid collisions with static and moving obstacles. The method applies to robots navigating in 2D and 3D workspaces and is composed of:

*1) Locally optimal group formation:* computed within a neighborhood of the robots and given by a centralized sequential convex optimization with avoidance constraints.

*2) Local planning for individual robots:* local collision-free motion that directs the robot toward the optimal formation and respects the dynamic constraints of the individual robot is obtained through a distributed convex optimization.

### B. Related works

Extensive work exists for real-time navigation of multiple robots in formation. These techniques include using a set of reactive behaviors [1], potential fields [2], navigation functions [3] and decentralized feedback laws with graph theory [4]. These have been mostly shown in 2D environments and may require extensive tuning for the particular formation and environment. In contrast, our method automatically optimizes for the formation parameters natively in 2D and 3D environments. On the down side, our method is currently centralized, and does not directly model the agent dynamics in the optimization, although it includes them in the individual local planners.

Given a formation, extensive work (see short review [5]) also exists in maintaining it while respecting the agent's dynamics. Approaches include: leader follower [6], Lyapunov functions [7] and model predictive control [8]. Since our approach computes the optimal formation parameters, it could potentially be combined with these approaches.

Convex optimization frameworks for navigating in formation include semidefinite programming [9] which considers only 2D circular obstacles, distributed quadratic optimization [10] without global coordination and limited adaptation of the formation, and second order cone programming [11] which triangulates the free 2D space to compute the optimal motion in formation.

Off-line non-convex optimizations include a mixed integer approach [12] and a discretized linear temporal logic approach [13]. They provide global guarantees but scale poorly with the number of robots. The second one is further limited in the definition of the formation. In contrast, we aim at on-line computation, albeit local, by solving a non-linear program via sequential convex programming. This technique (SCP) has recently been employed [14] to compute collision-free trajectories for multiple UAVs, but without considering formations.

Similar to our work on pattern formation for animation display [15], although obstacles were not considered there,

* The authors are at CSAIL-MIT, 32 Vassar St, 02139 Cambridge MA, USA jalonsom,spbaker,rus@mit.edu

*This work was supported in part by SMARTS N00014-09-1051, the Boeing Company and the MIT-Singapore Alliance on Research and Technology under the Future of Urban Mobility. We are grateful for their support.

we decouple the problem here in two steps: finding an optimal formation and individually navigating towards it. An approach proven to scale well with the number of robots.

### C. Organization

Sec. II includes an overview of the method and definitions. Sec. III describes the non-linear optimization to compute the optimal formation; followed by Sec. IV describing the individual collision avoidance to transition between formations. In Sec. V an extension for transportation of an object by multiple manipulators is then introduced. Sec. VI presents experimental results and Sec. VII concludes this paper.

## II. Preliminaries

### A. Definitions

For each robot $i \in \mathcal{I} = \{1, \dots, n\} \subset \mathbb{N}$, its position at time $t$ is denoted by $\mathbf{p}_i(t) \in \mathbb{R}^3$. For simplicity of exposition, we consider all robots to have the same dynamic model and cylindrical non-rotating shape of radius $r$ and height $2h$ in the vertical dimension. Denote their volume by $\mathcal{V} \subset \mathbb{R}^3$. *For an alternative description of the robots, refer to the extension for mobile manipulators in Sec. V.*

The collision-free workspace is given by $\mathcal{W} = \mathbb{R}^3 \setminus (\mathcal{O} + \mathcal{V})$, where the set of static obstacles $\mathcal{O} \subset \mathbb{R}^3$ is dilated by the robot's volume $\mathcal{V}$ and is given by a list of polytopes (arbitrarily many, such as occupied voxel cells).

Moving obstacles $j \in \mathcal{J} = \{1, \dots, n_{DO}\} \subset \mathbb{N}$ are considered, of arbitrary shape $\mathcal{D}_j \subset \mathbb{R}^3$ and predicted trajectories of position $\mathbf{p}_j^{DO}(t) \in \mathbb{R}^3$ at time $t$. We denote by $\mathcal{D}_j(t)$ the volume occupied by the dynamic obstacle $j$ at time $t$.

### B. Algorithm overview

A global planner is considered, which provides at all times a desired goal position for the formation. This can be given by a human operator or a standard sampling based approach, and is outside of the scope of this work. Denote by $\mathbf{g}(t) \in \mathbb{R}^3$ the goal position for the centroid of the formation at time $t$.

The proposed method is a local planner that computes a target formation and the required motion of the robots for a given time horizon $\tau > 0$, which must be longer than the required time to stop. Denote the current time by $t_o$ and $t_f = t_o + \tau$.

Our method consists of the following steps.

- Compute the largest convex polytope $\mathcal{P}$ in free space such that the current positions are within, $\mathbf{p}_i(t_o) \in \mathcal{P} \subset \mathcal{W}$, $\forall i \in \mathcal{I}$, and that is directed towards the goal $\mathbf{g}(t_f)$. This is described in Sec. II-C.
- Compute the optimal formation $\mathcal{F}(t_f)$ contained within $\mathcal{P}$ and minimizing the distance between the formation's centroid and the goal $\mathbf{g}(t_f)$. The parameters of the formation are optimized subject to a set of constraints via a centralized sequential convex optimization described in Sec III. In this computation the robot's dynamics are ignored.
- In a faster loop, described in Sec. IV, the robots are optimally assigned to target positions of the formation $\mathcal{F}(t_f)$ and move towards them employing a low level local planner that generates collision-free inputs that respect the robot's dynamics. In particular, we build on a distributed convex optimization [16], extended to account for static obstacles in a seamless way.
- If no feasible formation exists, the robots navigate independently towards the goal.

### C. Directed largest convex region in free-space

A fast iterative method, IRIS, to compute the largest convex polytope in free space was recently presented by [17]. The method consists of two recurrent steps: finding the separating hyperplanes between an ellipsoid $E$ and the dilated obstacles $\mathcal{O}+\mathcal{V}$ via a quadratic optimization; and computing the largest ellipsoid $E$ within the convex polytope $\mathcal{P}$, union of hyperplanes, via a semi-definite program.

We extend the method by

- considering a set of points *In*, potentially $\{\mathbf{p}_1, ...\mathbf{p}_n\}$, to be contained within $\mathcal{P}$. The iterative algorithm breaks at convergence or when $In \nsubseteq \mathcal{P}$,
- growing the region $\mathcal{P}$ towards a desired point $\mathbf{g}_{dir}$. This is achieved by initializing $E$ to be the minimal ellipsoid such that $\{In, \mathbf{g}_{dir}\} \subset E$. The point $\mathbf{g}_{dir}$ is typically set to $\mathbf{g}(t_f)$, and must also be contained within $\mathcal{P}$. If no solution exists, we modify the point via a linear search between $\mathbf{g}_{dir}$ and the centroid of the points in *In*.

Denote by $\mathcal{P}_{In}^{\mathbf{g}_{dir}}(\mathcal{W})$ the resulting convex polytope, which satisfies $In \subset \mathcal{P}_{In}^{\mathbf{g}_{dir}}(\mathcal{W}) \subset \mathcal{W}$.

### D. Definition of the formation

*For an alternative description of the formation, refer to the extension for mobile manipulators in Sec. V.*

We consider a pre-defined set of $f \in \mathbb{N}$ default formations, such as square, line or T. Denote by $\mathcal{F}_0^i$, $1 \leq i \leq f$, one such default formation. Formation $\mathcal{F}_0^i$ is given by a set of robot positions $\{\mathbf{r}_{0,1}^i, \dots, \mathbf{r}_{0,n}^i\}$ and a set of vertices $\{\mathbf{f}_{0,1}^i, \dots, \mathbf{f}_{0,n_i}^i\}$ relative to the center of rotation (typically the centroid) of the formation. The set of vertices represents the convex hull of the robot's positions in the formation, thus reducing the complexity for formations with a large number of robots. See Fig. 1(a) for an example. Note that the use of a convex region in free space already implies that the collision avoidance is performed for the convex hull of the formation.

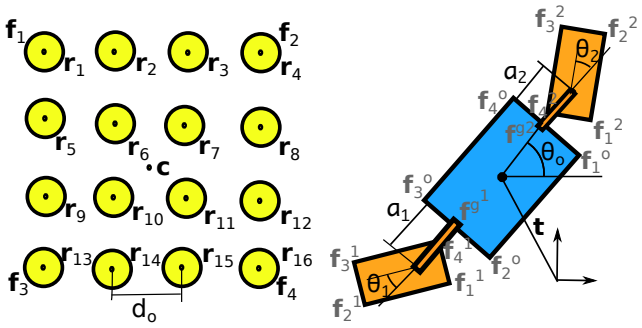
Further denote by $d_0^i$ the minimum distance between any given pair of robots in the default formation $\mathcal{F}_0^i$. Default formations can be defined by a human designer or automatically computed for optimal representation of a target shape [18].

A formation is then defined by an isomorphic transformation, which includes an expansion $s \in \mathbb{R}_+$, a translation $\mathbf{t} \in \mathbb{R}^3$ and a rotation represented by a unit quaternion $\mathbf{q} \in SO(3)$, its conjugate denoted by $\bar{\mathbf{q}}$. The vertices and robot positions of the resulting formation $\mathcal{F}^i$ are given by

$$\begin{aligned} \mathbf{r}_j^i &= \mathbf{t} + s\,\mathrm{rot}(\mathbf{q}, \mathbf{r}_{0,j}^i), \quad \forall j \in [1, n], \\ \mathbf{f}_j^i &= \mathbf{t} + s\,\mathrm{rot}(\mathbf{q}, \mathbf{f}_{0,j}^i), \quad \forall j \in [1, n_i], \end{aligned} \tag{1}$$

where the rotation in $SO(3)$ is given by

$$\begin{bmatrix} 0 & \mathrm{rot}(\mathbf{q}, \mathbf{f}_{0,j}^i) \end{bmatrix}^T = \mathbf{q} \times \begin{bmatrix} 0 & \mathbf{f}_{0,j}^i \end{bmatrix}^T \times \bar{\mathbf{q}}. \tag{2}$$




(a) Multi-agent formation (b) Mobile manipulators Sec. V

Fig. 1. Formation definitions. (a) For multi-agent navigation with robot positions and convex hull vertices. (b) For mobile manipulators of Sec. V.

In the exposition of the method we rely on this definition for the formation, but the method is more general and can be applied to alternative definitions, as shown in Sec. V for the case of several manipulators transporting a rigid object.

## III. Local Formation Optimization

In this section the constrained non-linear optimization to compute the optimal parameters (translation, expansion and rotation) of the formation is described. For a given default pattern $i \leq f$, the vector of optimization variables is denoted by $\mathbf{x}_i = [\mathbf{t}, s, \mathbf{q}] \in \mathbb{R}^8$.

*For an alternative formulation of the non-linear optimization, refer to the extension for mobile manipulators in Sec. V.*

### A. Collision-free convex region

First, the collision free space in position-time $\mathbb{R}^3 \times [0, \tau] \subset \mathbb{R}^4$ is obtained, accounting for static and dynamic obstacles.

For the given time horizon $\tau$ consider the union of static and dynamic obstacles

$$\bar{\mathcal{O}} = (\mathcal{O} + \mathcal{V}) \times [0, \tau] \cup \bigcup_{\substack{t \in [0,\tau] \\ j \in \mathcal{J}}} (\mathcal{D}_j(t_o + t) + \mathcal{V}) \times t \subset \mathbb{R}^4, \quad (3)$$

where $\mathcal{D}_j(t_o + t) + \mathcal{V}$ represents the volume occupied by dynamic obstacle $j$ at time $t_o + t$ dilated by the robots' volume $\mathcal{V}$. The position-time workspace is then

$$\bar{\mathcal{W}} = \mathbb{R}^3 \times [0, \tau] \setminus \bar{\mathcal{O}} \subset \mathbb{R}^4. \quad (4)$$

Following Sec II-C, we compute the convex polytopes

- $\mathcal{P}_{f_o \to g} := \mathcal{P}^{[\mathbf{g}(t_f), \tau]}_{[\mathbf{p}_1(t_o), \ldots, \mathbf{p}_n(t_o)] \times 0}(\bar{\mathcal{W}})$, which contains the robots at their current positions and is directed towards the formation's goal
- $\mathcal{P}_{o \to g} := \mathcal{P}^{[\mathbf{g}(t_f), \tau]}_{[\sum_{i \in \mathcal{I}} \mathbf{p}_i(t_o)/n, 0]}(\bar{\mathcal{W}})$, which contains the centroid of the robots' current positions and is directed towards the goal.

A representative example of these regions (projected in $\mathbb{R}^2$) is shown in Fig. 2. We consider the convex polytope $\mathcal{P} = \mathcal{P}_{f_o \to g} \cap \mathcal{P}_{o \to g}$, which guarantees that the transition to the new formation would be collision-free ($\mathcal{P} \subset \mathcal{P}_{f_o \to g}$ ) and is likely to make progress in future iterations ($\mathcal{P} \subset \mathcal{P}_{o \to g}$). If $\mathcal{P} = \emptyset$ an alternative convex region is selected as described in the forthcoming Sec. III-C.

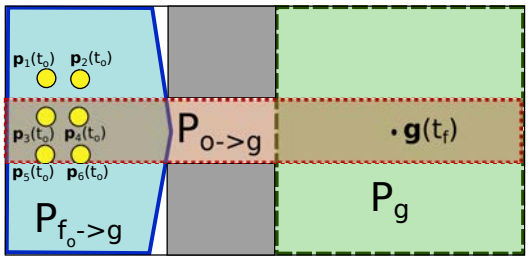



Fig. 2. Example of convex regions computed with the method described in Sec II-C for a scenario with six yellow disk robots and two (grey) squared obstacles. The convex regions are: $\mathcal{P}_{f_o \to g}$ in blue solid border, $\mathcal{P}_{o \to g}$ in red dotted border and $\mathcal{P}_g$ (defined in Sec. III-C) in green dashed border.

We rely on a representation of the collision-free convex polytope $\mathcal{P}$ given by its equivalent set of linear constraints

$$\mathcal{P} = \{\mathbf{z} \in \mathbb{R}^4 \mid A\mathbf{z} \leq \mathbf{b}, \text{ for } A \in \mathbb{R}^{n_l \times 4}, \mathbf{b} \in \mathbb{R}^{n_l}\}, \quad (5)$$

where $n_l$ denotes the number of faces of $\mathcal{P}$.

### B. Non-linear optimization

*1) Optimization cost:* We minimize a weighted sum of the deviation with respect to the formation's goal $\mathbf{g}(t_f)$, a desired size $\bar{s}$ and rotation $\bar{\mathbf{q}}$

$$C(\mathbf{x}_i) = w_t ||\mathbf{t} - \mathbf{g}(t_f)||^2 + w_s ||s - \bar{s}||^2 + w_q ||\mathbf{q} - \bar{\mathbf{q}}||^2 + c_i, \quad (6)$$

where $w_t$, $w_s$, $w_q$ are design weights, and $c_i$ is the predefined cost for formation $i$. The Jacobian is then given by

$$\partial C(\mathbf{x}) / \partial \mathbf{x} = 2[w_t(\mathbf{t} - \mathbf{g}(t_f)),\ w_s(s - \bar{s}),\ w_q(\mathbf{q} - \bar{\mathbf{q}})]. \quad (7)$$

*2) Constraints:* Constraints are introduced for the formation to be within the convex polytope ($C_1$), for satisfying a minimum inter-robot distance ($C_2$) and for imposing $\mathbf{q}$ is a unit quaternion ($C_3$). For the derivations, recall Sec. II-D.

$$\begin{aligned} C_1^j &= \{A(\mathbf{t} + s\,\mathrm{rot}(\mathbf{q}, \mathbf{f}^i_{0,j})) \leq \mathbf{b}\} \\ C_2 &= \{s\ d_0^i \geq 2\max(r, h)\} \\ C_3 &= \{||\mathbf{q}||^2 = 1\} \end{aligned} \quad (8)$$

where $C_1 = \bigcup_{j=1}^{n_i} C_1^j$ contains a constraint for each vertex $\mathbf{f}^i_{0,j}$ of the convex hull of the formation $\mathcal{F}_0^i$. The single constraint $C_2$ guarantees inter-robot collision avoidance since $d_0^i$ is the minimum inter-robot distance for the default formation, recall Sec. II-D, and the transformation applied to the formation is isomorphic.

The Jacobians are then given by

$$\begin{aligned} \partial C_1^j / \partial \mathbf{x} &= [A,\ A\,\mathrm{rot}(\mathbf{q}, \mathbf{f}^i_{0,j}),\ s\,A\,\partial \mathrm{rot}(\mathbf{q}, \mathbf{f}^i_{0,j}) / \partial \mathbf{q})] \\ \partial C_2 / \partial s &= d_0^i \\ \partial C_3 / \partial \mathbf{q} &= 2\mathbf{q} \end{aligned} \quad (9)$$

where $\partial \mathrm{rot}(\mathbf{q}, \mathbf{f}^i_{0,j}) / \partial \mathbf{q}$ is the Jacobian of Eq. (2)[1].

For planar formations, the additional constraints $\mathbf{q} \cdot [0, 1, 0, 0]^T = 0$ and $\mathbf{q} \cdot [0, 0, 1, 0]^T = 0$ may also be imposed to ensure rotation only occurs in the horizontal plane.

[1]We employ the implementation within the Drake toolbox, available in RobotLocomotion/drake/util/quatRotateVec.m

*3) Non-linear program:* For the default formation $i \leq f$ the parameters of the optimal transformation are found by solving the non-linear optimization given by

$$\begin{aligned} \mathbf{x}_i^* = \quad & \arg\min_{\mathbf{x}_i} C(\mathbf{x}_i) \\ s.t. \quad & \text{constraints } C_1,\, C_2,\, C_3. \end{aligned} \tag{10}$$

We employ the non-linear solver SNOPT [19], which internally executes a sparse Sequential Quadratic Program. The initial point is set to $\mathbf{x}_i^0 = [\mathbf{g}(t_f), 2\max(r,h)/d_0^i, \mathbf{q}_{ini}]$, where $\mathbf{q}_{ini} = \text{unitVector}(\bar{\mathbf{q}} + 0.1\,\text{rand}(1,4))$. The small random values are added to the desired quaternion to avoid singularities.

The optimal formation $\mathcal{F}$, with index $i^*$ is then given by

$$i^* = \arg\min_i C(\mathbf{x}_i^*). \tag{11}$$

*4) Remarks:* This particular definition of the formation and its associated non-linear optimization are given as an example, the framework is general and can therefore be applied to other problem instances, such as the one described in Sec V for manipulation of rigid objects.

We acknowledge that, even if the formation may have arbitrary shape, with this formulation collision avoidance is conservatively defined as for the convex hull of the formation. This fact is derived from the constraint $C_1$.

If no valid formation exists or no progress towards the goal is made, additional optimization searches are performed as discussed in the forthcoming Sec. III-C.

### C. Search iterations if no valid formation is found

If no valid formation exists within $\mathcal{P}$, or no progress is made towards the goal, the alternative collision-free convex polytope $\mathcal{P}_g := \mathcal{P}_{[\mathbf{g}(t_f),\tau]}(\bar{\mathcal{W}})$ is defined, which represents the largest convex polytope containing the formation's goal. A representative example is shown in Fig. 2.

Our method consists of a maximum of three additional search steps, solving the previous non-linear optimization and finalizing as soon as a valid formation that progresses towards the goal is found. The additional search steps are

i. Find $[\mathcal{F}, \tau] \subset \mathcal{P}_{f_o \to g}$
ii. Find $[\mathcal{F}, \tau] \subset \mathcal{P}_{o \to g}$
iii. Find $[\mathcal{F}, \tau] \subset \mathcal{P}_g$

If a valid target formation $\mathcal{F}$ is found in the original optimization or in the additional step i, the transition is guaranteed to be collision-free for holonomic agents. This is from the convexity of the polytope $\mathcal{P}_{f_o \to g}$ containing the current position of the robots. This is not the case for formations found in the additional steps ii or iii.

If a valid formation is only found in steps ii or iii the robots will be driven towards it with their individual local planners, as described in the forthcoming Sec. IV, but the formation might be broken during the transition. Nonetheless, this gives further flexibility to the method and the robots navigate in formation whenever possible.

## IV. Collision avoidance for individual robots

The result of the computation of Sec. III is a target formation $\mathcal{F}$ and its associated set of target robot positions $\{\mathbf{r}_{0,1}, \ldots, \mathbf{r}_{0,n}\}$ computed with Eq. (1).

In this section, we describe the local planner that links the centralized formation optimization with the physical robots. At each step of the execution, at higher frequency than that of computing a new formation, the following steps are executed.

### A. Goal assignment

Robots are optimally assigned to the goal positions with the objective of minimizing the largest travelled distance. For computational efficiency the optimal assignment $\sigma : \mathcal{I} \to \mathcal{I}$ is approximated by

$$\min_\sigma \quad \sum_{i \in \mathcal{I}} ||\mathbf{p}_i - \mathbf{r}_{\sigma(i)}||^2 \tag{12}$$

Following [15] this assignment can be centrally computed with the optimal Hungarian algorithm [20], used in this work, or a suboptimal auction algorithm [21], which scales well with the number of robots.

### B. Collision avoidance

To compute a collision-free local motion towards the goal, we build on the recent work on distributed reciprocal velocity obstacles with motion constraints for aerial vehicles [16]. This approach is able to adapt to changes in the environment and moving obstacles in real-time and respects the dynamics of the robot. Nonetheless, our method is agnostic to the local planner employed and the reader may choose a different one.

The idea behind the method is as follows. A set of candidate local trajectories is defined, where each local trajectory is given by a controller (of sufficient continuity) towards a straight-line reference trajectory of velocity $\mathbf{u} \in \mathbb{R}^3$.

In each time-step of the local planner, and given the current position and velocity of the neighboring agents, an optimal reference velocity $\bar{\mathbf{u}} \in \mathbb{R}^3$ is obtained by solving a convex optimization in $\mathbb{R}^3$. Its associated local trajectory is collision-free, satisfies the motion constraints and minimizes the deviation from the preferred velocity $\mathbf{u}^*$ towards the target position $\mathbf{r}_{\sigma(i)}$. The time horizon of the local planner $\tau_c > 0$, must be longer than the required time to stop.

We employ the convex optimization defined by [16] with identical motion constraints and constraints for avoidance of other agents, but we extend the method towards environments with complex static obstacles. In particular, we add a new constraint to guarantee that the motion of the robot is within the workspace $\mathcal{W}$ using the convex polytope computation described in Sec. II-C.

The largest convex polytope $\mathcal{P}_{\mathbf{p}_i}^{\mathbf{r}_{\sigma(i)}}(\mathbb{R}^3 \setminus (\mathcal{O} + \mathcal{V}_\varepsilon)) \subset \mathbb{R}^3$ is computed, such that the initial position is contained therein and the region is directed towards the robot's goal in the new formation. The static obstacles are dilated by the robot volume further enlarged by a value $\varepsilon$, as defined by [16]. The resulting region is then converted to an equivalent region in reference velocity $\mathbf{u}$ space,

$$\hat{\mathcal{P}}(\mathbf{p}, \varepsilon) := (\mathcal{P}_{\mathbf{p}_i}^{\mathbf{r}_{\sigma(i)}}(\mathbb{R}^3 \setminus (\mathcal{O} + \mathcal{V}_\varepsilon)) - \mathbf{p}_i)/\tau \tag{13}$$

This polytope is a set of linear constraints for static obstacle avoidance that can be included in the distributed convex optimization defined in [16]. Additional linear constraints for avoidance of moving obstacles were obtained from linearized pair-wise velocity obstacles. A collision-free reference velocity $\bar{\mathbf{u}}$ is then computed and transformed into a local motion for the robot respecting its dynamic constraints.

### C. Global planning towards the new formation

If for robot $i$ its target position $\mathbf{r}_{\sigma(i)}$ is within line of sight (this is the case if $\mathcal{F} \subset \mathcal{P}_{f_o \to g}$) the collision avoidance algorithm of Sec. IV-B can drive the robot towards it.

If the target position $\mathbf{r}_{\sigma(i)}$ is *not* within line of sight (this can only occur in the additional search steps ii and iii), a deadlock can potentially occur, due to the local nature of the collision avoidance algorithm. In this case, deadlocks can be avoided by first computing an approximate global path from $\mathbf{p}_i$ to $\mathbf{r}_{\sigma(i)}$ and then driving along it with the presented local collision avoidance algorithm. For efficiency, we use the approximate global planner described in [22].

## V. Extension for mobile manipulators

In the previous sections we described a method to navigate a team of robots in formation. In this section we show an extension where two mobile manipulators collaboratively carry a rigid object in a dynamic environment.

To achieve this goal, the robot shape, formation definition and optimization equations are modified, but the derivations follow the same line of thought of the previous sections.

### A. Robot and formation definition

The formation is defined by two rectangular mobile manipulators, each equipped with a robotic arm and grasping a rigid object at given points, see Fig. 1(b). The position $\mathbf{t} \in \mathbb{R}^2$ and orientation $\theta_o \in \mathbb{R}$ of the object can vary and each manipulator $i = \{1, 2\}$ can rotate around the center of its arm by an angle $\theta_i \in [\theta_{min}, \theta_{max}] \subset \mathbb{R}$ relative to the grasping direction and change the arm length $a_i \in [a_{min}, a_{max}] \subset \mathbb{R}$.

Denote the vertices of the object relative to its center by $\{\mathbf{f}^o_{0,1}, \ldots, \mathbf{f}^o_{0,n_o}\}$ and the vertices of manipulator $i$ relative to the center of its arm by $\{\mathbf{f}^i_{0,1}, \ldots, \mathbf{f}^i_{0,n_i}\}$. Denote the grasping positions on the object, relative to its center, by $\mathbf{f}^{g_i}_0$ and the arm grasps along the direction to the center of the object.

The manipulator and object vertices are then given by

$$\begin{aligned} \mathbf{f}^o_j &= [\mathbf{t}, 0]^T + R_{\theta_0} \mathbf{f}^o_{0,j}, && \forall j \in [1, n_o] \\ \mathbf{f}^i_j &= [\mathbf{t}, 0]^T + R_{\theta_0}((1 + a_i)\mathbf{f}^{g_i}_0 + R_{\theta_i} \mathbf{f}^i_{0,j}), && \forall j \in [1, n_i], \end{aligned} \tag{14}$$

where $R_\theta$ is the rotation matrix $[\cos(\theta), -\sin(\theta), 0; \sin(\theta), \cos(\theta), 0; 0, 0, 1]$.

### B. Collision-free convex region

Recalling Sec. III-A, the position-time collision-free workspace is given by

$$\begin{aligned} \bar{\mathcal{O}} = \mathcal{O} \times [0, \tau] \cup \bigcup_{\substack{t \in [0,\tau] \\ j \in \mathcal{J}}} (\mathcal{D}_j(t_o + t)) \times t \subset \mathbb{R}^4, \\ \bar{\mathcal{W}} = \mathbb{R}^3 \times [0, \tau] \setminus \bar{\mathcal{O}} \subset \mathbb{R}^4. \end{aligned} \tag{15}$$

where, now, the static and dynamic obstacles are *not* dilated.

The collision-free convex polytope containing the robots at their current state and directed towards the goal is

$$\mathcal{P}_{f_o \to g} := \mathcal{P}^{[\mathbf{g}(t_f), \tau]}_{\cup_{i=0:3}[\mathbf{f}^i_1(t_o), \ldots, \mathbf{f}^i_{n_i}(t_o)] \times 0}(\bar{\mathcal{W}}), \tag{16}$$

### C. Non-linear optimization

Denote by $\mathbf{x} = [\mathbf{t}, \theta_o, a_1, a_2, \theta_1, \theta_2] \in \mathbb{R}^7$ the vector of optimization variables. Recalling Eq. (14) and the representation of the collision-free polytope $\mathcal{P}_{f_o \to g}$ by a set of linear constraints, as in Eq. (5), the optimization is

$$\begin{aligned} \mathbf{x}^* = \quad & \arg\min_{\mathbf{x}} \ ||\mathbf{t} - \mathbf{g}(t_f)||^2 \\ s.t. \quad & A\mathbf{f}^i_j(\mathbf{x}) \leq \mathbf{b} && \forall j, \ \forall i \in \{0, 1, 2\} \\ & \theta_{min} \leq \theta_i \leq \theta_{max} && \forall i \in \{1, 2\} \\ & a_{min} \leq a_i \leq a_{max} && \forall i \in \{1, 2\} \end{aligned} \tag{17}$$

where additional cost terms can be added. The derivatives of the constraints are computed as in Eq. (9) and left out in the interest of space.

## VI. Experimental results

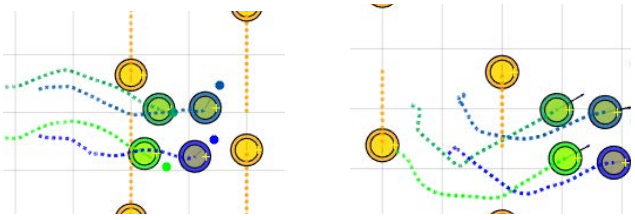
In this section we present simulations with teams of quadrotor UAVs and experiments with mobile manipulators. A video illustrating the results accompanies this paper and can be also found at https://youtu.be/MNvh03xYDIs. For the UAVs we employ the same dynamical model and controller of [16], which was verified with real quadrotors.

We use SNOPT [19] to solve the non-linear program, a goal-directed version of IRIS [17] to compute the largest convex regions and the Drake toolbox from MIT to handle quaternions, constraints and interface with SNOPT.

### A. Multiple aerial vehicles in formation

In our simulations a time horizon $\tau = 4$ s is considered for the experiments with 4 robots and of $\tau = 10$ s for the experiments with 16 robots, due to the large size of the formation and the scenario. In all cases a new formation is computed every 2 s. The individual collision avoidance planners run at 5 Hz. The quadrotors move at speeds between 0.5 m/s and 1.5 m/s.

Fig. 3 shows the trajectories of four quadrotors passing through two lanes of dynamic obstacles. The dynamic obstacles in the left lane move downwards at 0.4 m/s and the ones in the right move upwards with the same speed. Two



(a) Snapshot at 15s, paths: 0-15s (b) Snapshot at 25s, paths: 15-25s

Fig. 3. Top view. Four robots (green-blue) navigate a 8 x 15 $m^2$ environment with two lanes of dynamic obstacles (orange). The four robots locally reconfigure the formation moving towards the right side.

default formations are considered, square (which is preferred) and diamond. The goal for the formation follows a constant velocity trajectory along the middle horizontal line and the team successfully adapts the parameters of the formation to remain collision-free and pass in-between the obstacles. In this case we imposed that the formation remains on the horizontal plane for illustrative purposes.

Fig. 4 presents results for 3D navigation in the scenario of Fig. 3 with four quadrotors for different values of the speed of the dynamic obstacles and of the formation. We employed a fixed time horizon of $\tau = 4$ s and observe that most of the time the target formation for the robots is within $\mathcal{P}_{f_o->g}$. But, at high speeds, in order to quickly progress towards the goal, the robots temporally break the formation and select a target one within $\mathcal{P}_{o->g}$ or $\mathcal{P}_g$ (note that in this simulated 8 m x 15 m environment, for $\tau = 4$ s and for a speed of 2 m/s the goal for the formation is at a distance of 8 m, far ahead of the two lanes of dynamic obstacles.). We believe that the results could be improved with an adaptive time horizon depending on the speed or a simple global planner.

In this scenario, when the target speed of the formation is at least half of that of the dynamic obstacles, our framework successfully drives the robots towards the goal while avoiding collisions, only marginal ones are appreciated. Collisions arise when the speed of the dynamic obstacles is much higher (about double) than that of the formation. This is due to the local planning horizon and the robots being unable to escape on time due to their lower speed.

Fig. 5 shows snapshots and trajectories of four quadrotors tracking a circular trajectory while locally avoiding three static obstacles and a dynamic obstacle. Three default formations are considered: square (1st preference), diamond (2nd preference) and line. The optimal parameters are computed with the non-linear optimization allowing rotation in 3D (flat horizontal orientation preferred) and reconfiguration.

The four quadrotors start from the horizontal square and slightly tilt it (11 s) to avoid the incoming dynamic obstacle. To fully clear it while avoiding the obstacle in the lower corner, they shortly switch to a vertical line, and then back

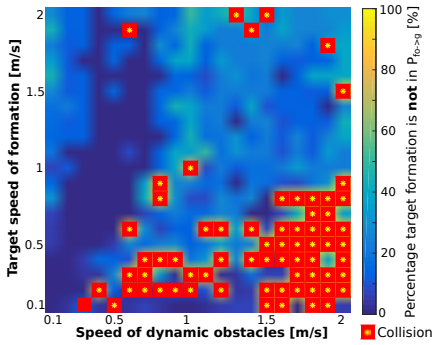



Fig. 4. Percentage of time in which the target formation, to make progress towards the goal, is within $\mathcal{P}_{o->g}$ or $\mathcal{P}_g$, but not in $\mathcal{P}_{f_o->g}$. Here, lower values are better, since they correspond to the percentage of time for which the formation might be broken. Results for 3D navigation in the scenario of Fig. 3 with four quadrotors and varying speed of the dynamic obstacles and the formation. Fixed time horizon of $\tau = 4$ s is used in all experiments.

to the preferred square formation (20 s). To pass through the next narrow opening they switch back to the line formation (30 s) and then to the preferred square, tilted to avoid the dynamic obstacle (37 s). Once the obstacles are cleared they return to the preferred horizontal square formation (45 s).

Fig. 6 shows the paths of 16 quadrotors moving along a corridor of three different widths. Three default formations are considered: 4x4x1 defined by four vertices (preferred), 4x2x2 defined by eight vertices and 8x2x1 defined by four vertices. At each time step the method computes the optimal parameters for each of the three and select the best one. Between times 75 s and 110 s the method successfully rotates the formation by $90^o$ for it to be collision free (the default formations were horizontal, which is also preferred in the cost function).

In Table I we provide computational times for our implementation in Matlab using a 2.6 GHz i7 laptop. The time given for the non-linear optimization is for a single formation, and in our experiments we employed two to three different formations, such as square, line or diamond. Thanks to the abstraction of a formation by the vertices of its convex hull, see Sec. II-D, the computation time of the non-linear optimization is independent of the number of robots. The approach shows close to real time performance, below 1s, even in our Matlab implementation and we believe that great speed gains would be achieved with a C++ implementation. For instance, our C++ implementation of the collision avoidance is about 40 times faster, below 2 ms per robot.

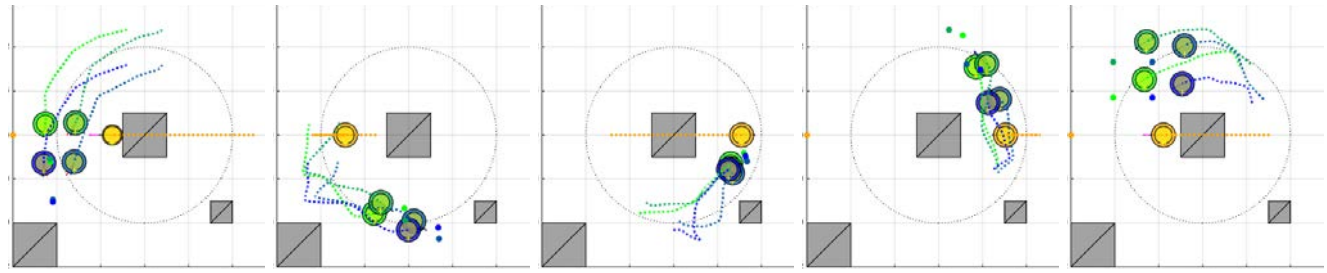
### B. Collaborative transportation with mobile manipulators

In Fig. 7 we show four snapshots of a representative avoidance maneuver with two mobile manipulators carrying a rigid object, as described in Sec. V. For each snapshot the current (blue) and target (green) formation given by the optimization are displayed. The two robots successfully adapt their formation, rotating as required, to avoid both the dynamic obstacle (red) and static (grey) obstacles in this 8m x 6m scenario. Slices at $t_o$ and $t_o + \tau$ of the convex region computed in position-time space are also shown for illustrative purposes.

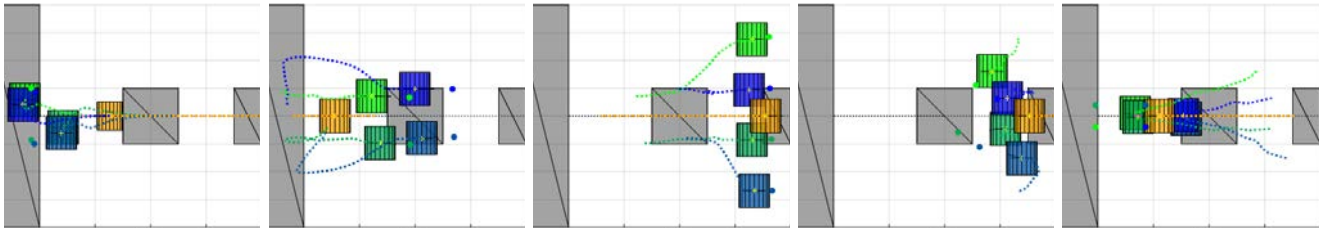
In Fig. 8 we show four snapshots of an experiment where the two mobile manipulators successfully carry the rigid object to the goal position behind the orange boxes while locally avoiding collisions with the human. In all of our experiments, executed with external tracking, the robots successfully adapted their formation to avoid collisions. This assumes that the human cooperates, otherwise, collisions

TABLE I
COMPUTATIONAL TIME [MS] FOR A MATLAB IMPLEMENTATION

| Compute | Min | Mean | Max | Standard deviation |
|---|---|---|---|---|
| Convex region | 31.8 | 82.8 | 221.4 | 72.1 |
| NL Optimization | 93.7 | 226.4 | 522.7 | 64.1 |
| Goal Assignment | 0.5 | 0.8 | 5.0 | 0.1 |
| Collision avoidance | 29.3 | 63.4 | 113.9 | 24.2 |

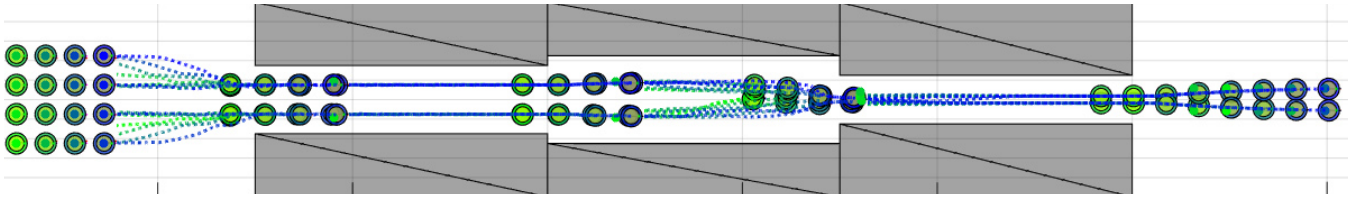


(a) Top view. From left to right, snapshots at 11 s, 20 s, 30 s, 37 s and 45 s, and paths of the robots in-between.

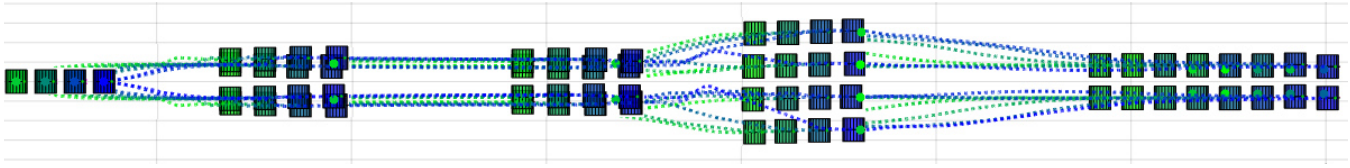


(b) Side view. From left to right, snapshots at 11 s, 20 s, 30 s, 37 s and 45 s, and paths of the robots in-between.

Fig. 5. Four quadrotors (green-blue cylinders) navigate in a 12 x 12 x 6 $m^3$ scenario with three static obstacles (grey) and a dynamic obstacle (yellow). The four quadrotors track a circular motion (black dots in top view) and locally reconfigure the formation to avoid collisions and make progress.



(a) Top view with robot paths and snapshots at time 0 s, 25 s, 50 s, 75 s, 110 s. Robots move from left to right.



(b) Side view with robot paths and snapshots at time 0 s, 25 s, 50 s, 75 s, 110 s. Robots move from left to right.

Fig. 6. 16 quadrotors navigate along a 70 x 10 x 10 $m^3$ corridor, with obstacles shown in grey. The quadrotors locally adapt the formation to remain collision free. The following formations are observed: 4x4x1 - 4x2x2 - 4x4x1 (vertical) - 8x2x1 (vertical), finally transitioning towards horizontal 8x2x1.

may still occur if the human moves faster than the robots or traps them against an obstacle.

### C. Discussion

The presented method shows good performance, successfully computing the optimal parameters for the multi-robot formation and allowing for reconfiguration, with real-time performance. Thus providing collision-free navigation among static and dynamic obstacles. At least in part, the computational efficiency is achieved by (a) not including the agent dynamics in the formation optimization but handling them in the individual local planners (this works well for robots with fast dynamics); and (b) considering the convex hull of the formation. In our experiments, the computation of the largest collision-free convex polytope following Sec. II-C showed very good results, but no guarantees exist that the best volume will be obtained. Searching several regions might prove advantageous. Furthermore, we do not allow for splitting and merging of formations.

## VII. Conclusion

In this paper we showed that navigation of teams of robots in formation among arbitrary static and dynamic obstacles can be achieved via a constrained non-linear optimization. By first computing the largest collision-free convex polytope and then optimizing the formation parameters, low computational cost is achieved together with good navigation results. In several simulations we showed successful navigation in formation where robots may reconfigure the formation as required to avoid collisions and make progress.

Furthermore, the approach can be adapted to other formation definitions and applications, as shown in the extension for mobile manipulators carrying a rigid object.

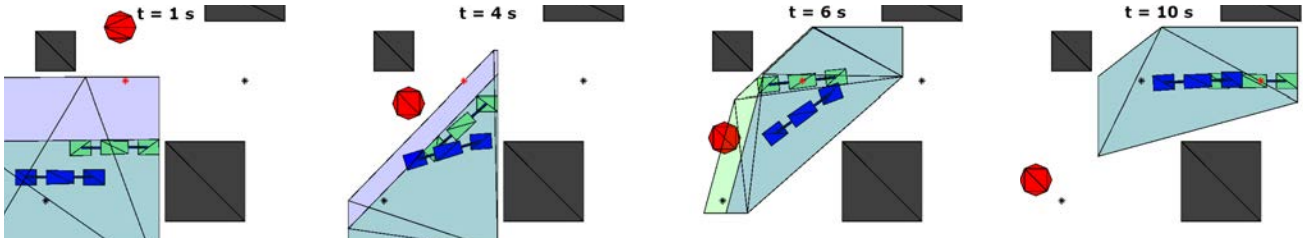



Fig. 7. Four consecutive snapshots of a 10 s avoidance maneuver where two mobile manipulators collaboratively carry a rigid object and navigate to two goals (crosses) while avoiding collisions with static (grey) and dynamic (red hexagon) obstacles. The current state of the two manipulators and the rigid object is displayed in blue and the target one (given by the optimization) in green. Two slices of the convex polytope are shown, purple for the current time $t_o$ and light green for time $t_o + \tau$ (the intersection is the larger light blue region). The dynamic obstacle is shown at time $t_o$ and it is moving at constant speed downwards. As displayed, the red dynamic obstacle may intersect with the light green slice of the convex polytope (at $t_f$), but not with the purple one (at $t_o$). The manipulators successfully navigate the rigid object through the two set points avoiding collisions.


Fig. 8. Four consecutive snapshots of an avoidance maneuver where two mobile manipulators collaboratively carry a rigid object and navigate it to the goal while adjusting their formation to avoid collisions with the orange boxes and the human. Robots and human are tracked by overhead cameras.

In future works, the observed limitations of the framework, Sec. VI-C, can be addressed. Further, a multi-step extension of our method could potentially be used for global planning.

## ACKNOWLEDGMENTS

The authors are grateful to Robin Deits and Hongkai Dai from MIT for their help with IRIS, Drake and SNOPT.

## REFERENCES

[1] T. Balch and R. C. Arkin, "Behavior-based formation control for multirobot teams," *IEEE Trans. on Robotics and Automation*, vol. 14, no. 6, pp. 926–939, 1998.

[2] T. Balch and M. Hybinette, "Social potentials for scalable multi-robot formations," in *IEEE Int. Conf. Robotics and Automation*, pp. 73–80 vol.1, 2000.

[3] N. Michael, M. M. Zavlanos, V. Kumar, and G. J. Pappas, "Distributed multi-robot task assignment and formation control," in *of the IEEE Int. Conf. on Robotics and Automation*, 2008.

[4] J. P. Desai, J. P. Ostrowski, and V. Kumar, "Modeling and control of formations of nonholonomic mobile robots," *IEEE Trans. on Robotics and Automation*, vol. 17, no. 6, pp. 905–908, 2001.

[5] Y. Q. Chen and Z. Wang, "Formation control: a review and a new consideration," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2005.

[6] W. Ren and N. Sorensen, "Distributed coordination architecture for multi-robot formation control," *Robotics and Autonomous Systems*, vol. 56, pp. 324–333, Apr. 2008.

[7] P. Ogren, M. Egerstedt, and X. Hu, "A control Lyapunov function approach to multi-agent coordination," in *IEEE Conf. on Decision and Control*, 2001.

[8] W. B. Dunbar and R. M. Murray, "Model predictive control of coordinated multi-vehicle formations," in *IEEE Conf. on Decision and Control*, 2002.

[9] J. Derenick, J. Spletzer, and V. Kumar, "A semidefinite programming framework for controlling multi-robot systems in dynamic environments," in *IEEE Conf. on Decision and Control*, 2010.

[10] J. Alonso-Mora, R. A. Knepper, R. Siegwart, and D. Rus, "Local Motion Planning for Collaborative Multi-Robot Manipulation of Deformable Objects," in *IEEE Int. Conf. Robotics and Automation*, 2015.

[11] J. C. Derenick and J. R. Spletzer, "Convex Optimization Strategies for Coordinating Large-Scale Robot Formations," *IEEE Trans. on Robotics*, vol. 23, no. 6, pp. 1252–1259.

[12] A. Kushleyev, D. Mellinger, and V. Kumar, "Towards a swarm of agile micro quadrotors," in *Robotics: Science and Systems*, 2012.

[13] I. Saha, R. Ramaithitima, V. Kumar, G. J. Pappas, and S. A. Seshia, "Automated Composition of Motion Primitives for Multi-Robot Systems from Safe LTL Specifications," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2014.

[14] F. Augugliaro, A. P. Schoellig, and R. D'Andrea, "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 1917–1922, 2012.

[15] J. Alonso-Mora, A. Breitenmoser, M. Rufli, R. Siegwart, and P. Beardsley, "Image and Animation Display with Multiple Robots," *Int. Journal of Robotics Research*, vol. 31, pp. 753–773, May 2012.

[16] J. Alonso-Mora, T. Naegeli, R. Siegwart, and P. Beardsley, "Collision avoidance for aerial vehicles in multi-agent scenarios," *Auton. Robot.*, Jan. 2015.

[17] R. Deits and R. Tedrake, "Computing large convex regions of obstacle-free space through semidefinite programming," *Workshop on the Algorithmic Fundamentals of Robotics*, 2014.

[18] J. Alonso-Mora, M. Schoch, A. Breitenmoser, R. Siegwart, and P. Beardsley, "Object and animation display with multiple aerial vehicles," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2012.

[19] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM journal on optimization*, vol. 12, no. 4, pp. 979–1006, 2002.

[20] H. W. Kuhn, "The hungarian method for the assignment problem," in *Naval Research Logistics*, pp. 83–97, 1955.

[21] D. P. Bertsekas, "The auction algorithm: a distributed relaxation method for the assignment problem," *Ann. Oper. Res.*, vol. 14, pp. 105–123, June 1988.

[22] J. Yu and S. M. LaValle, "Fast, near-optimal computation for multi-robot path planning on graphs," in *AAAI Conf. on Artificial Intelligence, late breaking papers*, 2013.