Article

The International Journal of Robotics Research
2014, Vol 33(1) 98–112
© The Author(s) 2013
Reprints and permissions:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364913515307
ijr.sagepub.com

CAPT: Concurrent assignment and planning of trajectories for multiple robots

Matthew Turpin, Nathan Michael and Vijay Kumar

Abstract

In this paper, we consider the problem of concurrent assignment and planning of trajectories (which we denote CAPT*) for a team of robots. This problem involves simultaneously addressing two challenges: (1) the combinatorially complex problem of finding a suitable assignment of robots to goal locations, and (2) the generation of collision-free, time parameterized trajectories for every robot. We consider the* CAPT *problem for unlabeled (interchangeable) robots and propose algorithmic solutions to two variations of the* CAPT *problem. The first algorithm,* C-CAPT*, is a provably correct, complete, centralized algorithm which guarantees collision-free optimal solutions to the* CAPT *problem in an obstacle-free environment. To achieve these strong claims,* C-CAPT *exploits the synergy obtained by combining the two subproblems of assignment and trajectory generation to provide computationally tractable solutions for large numbers of robots. We then propose a decentralized solution to the* CAPT *problem through* D-CAPT*, a decentralized algorithm that provides suboptimal results compared to* C-CAPT*. We illustrate the algorithms and resulting performance through simulation and experimentation.*

Keywords

Multi-robot trajectory planning, interchangeable robots, task assignment, micro aerial vehicles

1. Introduction

Many multi-robot applications require the assignment of robots to tasks at different destinations and the generation of vehicle plans to enable navigation to these destinations. We are particularly interested in the use of robots for first response and law enforcement in urban neighborhoods where it may be necessary to acquire georegistered information from sensors like cameras and microphones positioned at different points of interest. Because the points of interest depend on the particular event and can change quickly, it is meaningful to consider teams of robots rapidly responding to such requests of information. Other representative examples of such applications include automated storage and product retrieval in warehouse applications (Enright and Wurman, 2011) and automated structure construction in which robots are commanded to fetch and place parts (Werfel et al., 2007; Lindsey et al., 2012) or when the robots themselves are used as structural elements of a larger construction (TEMP, n.d.; Oung and D'Andrea, 2011).

In all of these applications, the deployment of a team of robots requires solving the task assignment or goal assignment problem in which each goal is assigned to a robot, and the generation of collision-free trajectories that guide the robots to the assigned goal locations. However, the computational complexity of a decoupled strategy in which the tasks are first assigned followed by the solution to the trajectory planning problem can be prohibitive in terms of computational complexity. On the other hand, if the robots are considered to be homogeneous (in other words, it does not matter who goes where, or who picks up what part)[1], it is possible to couple the two subproblems. We call this the concurrent assignment and planning of trajectories (CAPT). Our goal in this paper is derive solutions to the CAPT problem and show that the resulting computational complexity is manageable.

By definition, the CAPT problem seeks an assignment of N unlabeled (permutation invariant) robots to M desired goal locations. To explore the complexity of this problem, consider the case when $N = M$. Each possible assignment of robots to goals can be represented by an $N \times N$ permutation matrix and the space of all possible assignments is isomorphic to $\mathbb{S}^N$, or the group of all permutations. To completely enumerate all possible assignments requires $N!$ evaluations and is infeasible for any system with more than just a few robots. Fortunately, the task assignment problem occurs naturally in a number of disciplines including distributed computing, operations research, and robotics.

GRASP Laboratory, University of Pennsylvania, Philadelphia, USA

Corresponding author:
Matthew Turpin, GRASP Laboratory, Levine Hall 4th Floor, University of Pennsylvania, 3330 Walnut Street, Philadelphia, PA 19104-6228, USA.
Email: mturpin@seas.upenn.edu

As such, there has been significant study into generating optimal solutions and useful suboptimal solutions. While there are multiple classes of the task assignment problem, we will restrict our focus to the linear assignment problem of minimizing the sum of individual costs for robot–goal pairs. The well-known Hungarian Algorithm (Kuhn, 1955; Munkres, 1957) solves the linear assignment problem with computational complexity bounded polynomially in the number of robots, $\mathcal{O}(N^3)$, and is the most efficient known method to optimally solve the linear task assignment problem. There are also a number of relaxations of the linear assignment problem to find near optimal solutions. For example Rendl (1988) uses a heuristic to minimize the sum of Euclidean distance traveled for a simplified Euclidean linear assignment problem with complexity bound $\mathcal{O}(N^{\frac{5}{6}})$.

We must also consider the problem of trajectory planning for a multi-robot team with pre-assigned goal locations. For a single robot, graph search techniques like A* and D* are extremely powerful for finding collision-free paths through an environment. However, to consider multiple robots simultaneously and provide collision avoidance between robots requires enlarging the search space, growing exponentially with the number of robots (Erdmann and Lozano-Perez, 1986). There are also planners which decouple the problem into path planning and the creation of velocity profiles for the generated paths (Kant and Zucker, 1986). Probabilistic planners such as Rapidly-Exploring Random Trees are more successful in high dimensional spaces, but simply cannot handle the exponential dimensionality of the search space for large teams of robots (LaValle, 2006). Subdimensional expansion (Wagner and Choset, 2011; Wagner et al., 2012) has been introduced to reduce this exponential growth by only increasing the dimensionality of the search space when pairs of robots are in close proximity. Subdimensional expansion, while powerful, cannot provide an upper complexity bound as it is quite possible that there are time intervals in which robots are highly coupled, causing the search space to grow impractically large.

There have been a number of studies into solving the CAPT problem in the past and we will now outline some of results of this effort. A typical strategy used to generate trajectories which solve the CAPT problem is to find the assignment which minimizes the sum of Euclidean distance traveled (Ji et al., 2006; Smith and Bullo, 2007). We address some of the shortcomings of this approach in Section 3.1. Liu and Shell (2011) present a hybrid method for the Euclidean assignment algorithm for very large numbers of robots with both a global and local approach. Alonso-Mora et al. (2012) find a suboptimal assignment which seeks to minimize distance squared, but do not generate trajectories with collision avoidance guarantees. There are suboptimal approaches which solve the CAPT problem such as in Kloder and Hutchinson (2006), which presents a centralized algorithm to create collision-free paths and solve the assignment problem for two-dimensional robots.

One approach to minimizing complexity is to consider decentralized solutions to the CAPT problem. The trajectory planning problem is naturally decoupled when robots are far away from each other. Thus, it makes sense to have robots independently plan trajectories to assigned goals, but reevaluate their assignments and planned trajectories as they approach each other. In this context, it is useful to introduce the abstraction of a communication or sensing range, h, outside which robots ignore their neighbors allowing decentralized planning and execution. When a robot comes within h of a neighbor, it must coordinate its actions with its neighbor requiring a partially centralized solution. Indeed it is possible to define reactive controllers using artificial potential functions (Zavlanos and Pappas, 2007) that modify trajectories locally when robots come close to each other while pursuing their assigned goals. In addition, there is literature on controlling groups of robots to goal destinations (Molnár and Starke, 2001) or goal sets (Chaimowicz et al., 2005). In general, these gradient descent approaches lead to unpredictable trajectories, may take a long time to converge, and do not always guarantee complete assignment. We will be interested in *complete* solutions to D-CAPT, the decentralized solution of the problem of concurrent assignment and planning trajectories. While D-CAPT does not, in general, guarantee optimal solutions, we will show that every local modification of trajectories is associated with a reassignment of goals that improves the quality of the resulting solution. By considering only local interactions, D-CAPT has substantially reduced computational requirements that allows the algorithm to scale to larger systems than a centralized approach can accommodate and suffers a minor decrease in optimality.

We begin with the presentation of preliminaries in Section 2. Section 3 introduces C-CAPT, a centralized optimal solution to the CAPT problem in obstacle-free environments that is computationally tractable for many hundreds of agents and goals. Section 4 demonstrates the flexibility of C-CAPT by outlining two simple extensions to solve the vehicle routing problem and incorporate the dynamics of a quadrotor UAV. We then introduce D-CAPT, the distributed suboptimal version of C-CAPT in Section 5. Section 6 presents numerical simulations and analysis of C-CAPT and D-CAPT.

2. Preliminaries

We consider N robots with radius R navigating from initial locations to M desired goal locations in an n-dimensional Euclidean space. We define the set of integers between 1 and positive integer Z as: $\mathcal{I}_Z \equiv \{1, 2, \ldots, Z\}$. For example, if there are three goal locations, $M = 3$ and $\mathcal{I}_M = \{1, 2, 3\}$.

The location of the ith robot is specified by $\mathbf{x}_i \in \mathbb{R}^n$, $i \in \mathcal{I}_N$, and similarly, the jth goal location is specified by $\mathbf{g}_j \in \mathbb{R}^n, j \in \mathcal{I}_M$. We then define the Nn-dimensional system state vector, $\mathbf{X} \in \mathbb{R}^{Nn}$:

$$\mathbf{X}(t) = [\mathbf{x}_1(t)^{\mathrm{T}}, \mathbf{x}_2(t)^{\mathrm{T}}, \ldots, \mathbf{x}_N(t)^{\mathrm{T}}]^{\mathrm{T}}$$

and similarly define the system goal state vector $\mathbf{G} \in \mathbb{R}^{Mn}$:

$$\mathbf{G} = [\mathbf{g}_1{}^{\mathrm{T}}, \mathbf{g}_2{}^{\mathrm{T}}, \ldots, \mathbf{g}_M{}^{\mathrm{T}}]^{\mathrm{T}}$$

We define the *assignment matrix* $\phi \in \mathbb{R}^{N\times M}$, which assigns agents to goals:

$$\phi_{i,j} = \begin{cases} 1 & \text{if robot } i \text{ is assigned to goal } j \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

We require either all goals to be assigned or all robots to be assigned which results in:

$$\begin{aligned} \phi^{\mathrm{T}}\phi &= I_M \qquad \text{if } N \geq M \\ \phi\phi^{\mathrm{T}} &= I_N \qquad \text{if } N \leq M \end{aligned} \quad (2)$$

where I_Z is the $Z \times Z$ identity matrix. We will also make use of the expanded assignment matrix $\Phi \equiv \phi \otimes I_n$, where $\otimes$ signifies the Kronecker product.

The CAPT problem seeks to find Nn-dimensional trajectories:

$$\gamma(t) : [t_0, t_f] \rightarrow \mathbf{X}(t)$$

where t_0 and t_f are the initial and final times respectively. Initial robot positions $\mathbf{x}_i(t_0)\ \forall i \in \mathcal{I}_N$ provide initial conditions for the trajectories:

$$\gamma(t_0) = \mathbf{X}(t_0) \quad (3)$$

We define the assignment such that it provides terminal conditions such that goal locations are occupied by robots:

$$\Phi^{\mathrm{T}}\gamma(t_f) = \mathbf{G}$$

However, in the case that $M > N$, these assignments are not unique. To find a unique assignment, we premultiply this condition by Φ:

$$\Phi\Phi^{\mathrm{T}}\gamma(t_f) = \Phi\mathbf{G}$$

and solve for $\gamma(t_f)$:

$$\gamma(t_f) = (\Phi\Phi^{\mathrm{T}})^{-1}\,\Phi\mathbf{G}$$

Using (2), we see that because $M > N$, $\Phi\Phi^{\mathrm{T}} = I_{Nn}$. Therefore, the assignment of robots to goals supplies the following trajectory terminal conditions:

$$\begin{aligned} \Phi^{\mathrm{T}}\gamma(t_f) &= \mathbf{G} \qquad \text{if } N \geq M \\ \gamma(t_f) &= \Phi\mathbf{G} \qquad \text{if } N < M \end{aligned} \quad (4)$$

We define clearance δ as the minimum space between any pair of robots at any time during the trajectory:

$$\delta(t) = \inf_{i \neq j \in \mathcal{I}_N, t \in [t_0, t_f]} ||\mathbf{x}_i(t) - \mathbf{x}_j(t)|| - 2R$$

With the exception of Section 4, the robots are assumed to have simple first order dynamics:

$$\begin{aligned} \dot{\mathbf{x}}_i &= \mathbf{u}_i \\ ||\mathbf{u}_i||_2 &\leq v_{max} \end{aligned} \quad (5)$$

To ensure collision avoidance for all robots, we require the clearance to always be greater than zero:

$$\delta(t) > 0 \quad t \in [t_0, t_f] \quad (6)$$

We define $\mathcal{K}$ as the convex hull of initial locations and goal locations with the Minkowski sum of a ball of radius R:

$$\mathcal{K} \equiv \mathrm{conv}\left(\{\mathbf{x}_i(t_0) \,\middle|\, i \in \mathcal{I}_N\} \cup \{\mathbf{g}_j \,\middle|\, j \in \mathcal{I}_M\}\right) \oplus \mathcal{B}_R \quad (7)$$

See Figure 1 for a two-dimensional pictorial example of $\mathcal{K}$. Section 3 presents C-CAPT, which requires an obstacle-free environment such that $\mathcal{K}$ is free of obstacles.

3. C-CAPT: A centralized obstacle-free CAPT solution

In this section, we will explicitly define the requirements of a centralized CAPT solution and the necessary assumptions for an obstacle-free environment. We will then study a minimum distance solution to the assignment problem in Section 3.1. Section 3.2 modifies this cost function and evaluates the solution generated by a minimum velocity squared trajectory. Finally, Section 3.3 presents equations which generate a solution to the CAPT problem in an obstacle-free environment.

Problem definition: C-CAPT seeks to find optimal trajectories $\gamma^\star(t)$ which minimize a specified cost functional:

$$\begin{aligned} \gamma^\star(t) = \underset{\gamma(t)}{\mathrm{argmin}} \quad & \int_{t_0}^{t_f} L(\gamma(t))\, dt \\ \text{subject to} \quad & \text{equation}(1) : \text{Valid assignment} \\ & \text{equation}(2) : \text{Full Resource utilization} \\ & \text{equation}(3) : \text{Initial conditions} \\ & \text{equation}(4) : \text{Terminal conditions} \\ & \text{equation}(5) : \text{Robot capabilities} \\ & \text{equation}(6) : \text{Collision avoidance} \end{aligned} \quad (8)$$

Assumptions: We explicitly state the assumptions required for C-CAPT:

(A1) All robots are homogeneous and interchangeable with no preference of goal location.
(A2) Each robot is a set of points confined to $\mathcal{B}_R$, a ball with radius R.
(A3) The region $\mathcal{K}$ as defined in (7) is obstacle-free.

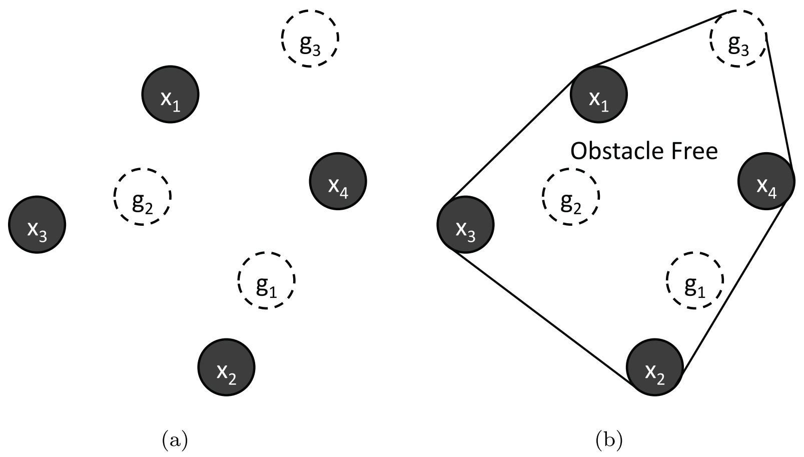


Fig. 1. For the locations of goals and agents in $\mathbb{R}^2$ in (a), $\mathcal{K}$ is designated by the closed area in (b).

(A4) The initial and goal locations are spaced Δ apart:

$$\begin{aligned} ||\mathbf{x}_i(t_0) - \mathbf{x}_j(t_0)|| &> \Delta \quad \forall\, i \neq j \in \mathcal{I}_N \\ ||\mathbf{g}_i - \mathbf{g}_j|| &> \Delta \quad \forall\, i \neq j \in \mathcal{I}_M \end{aligned} \tag{9}$$

where Δ will be defined later. Clearly $\Delta > 2R$ to ensure collision avoidance (6) at the boundary conditions. Additionally we require:

$$||\mathbf{x}_i(t_0) - \mathbf{g}_j|| > \Delta \ \ \forall i \in \mathcal{I}_N, j \in \mathcal{I}_M \quad \text{if} \quad N > M$$

(A5) Robots are fully actuated differentially flat systems, do not have any actuation error, and always have perfect state knowledge.

In Sections 3.1 and 3.2, we propose different cost functions $L(\gamma)$ in (8) which both seek to simultaneously find an assignment matrix ϕ and collision-free trajectories $\gamma(t)$ for all robots in the system such that the boundary conditions (3, 4) are satisfied.

3.1. Minimum sum of distances trajectories

The first cost function analyzed is commonly applied to the location assignment problem and seeks to minimize the sum of distances traveled by all agents:

$$\begin{aligned} &\underset{\phi,\gamma(t)}{\text{minimize}} && \sum_{i=1}^{N} \int_{t_0}^{t_f} \sqrt{\dot{\mathbf{x}}_i(t)^{\mathrm{T}} \dot{\mathbf{x}}_i(t)}\,dt \\ &\text{subject to} && (1), (2), (3), (4), (5), (6) \end{aligned}$$

If we temporarily ignore clearance requirements, it is clear that the solution reduces to positive progress on straight line paths for $t \in [t_0, t_f]$. Thus, this problem reduces to:

$$\begin{aligned} &\underset{\phi}{\text{minimize}} && \sum_{j=1}^{M} \sum_{i=1}^{N} \phi_{i,j} \left|\left|\mathbf{x}_i(t_0) - \mathbf{g}_j\right|\right|_2 \\ &\text{subject to} && (1), (2), (3), (4), (5) \end{aligned} \tag{10}$$

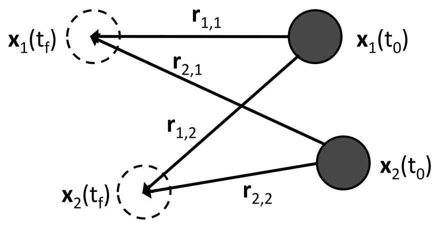


Fig. 2. For agents in n-dimensional Euclidean space, if paths intersect and are not collinear as is the case for straight lines along $\mathbf{r}_{1,2}$ and $\mathbf{r}_{2,1}$, we can see that using the triangle inequality, switching assignments will always lead to non-intersecting paths.

This is the well-known transportation assignment problem. We show below in Theorem 3.1 that the assignment from this optimization guarantees paths that almost never intersect in any $n > 1$-dimensional Euclidean space.

Theorem 3.1. *The optimal assignment ϕ using the minimum sum of distance optimization in* (10) *results in non-intersecting paths with the exception of the special case when a pair of robots have collinear start and goal locations.*

Proof. Assume the paths of agents i and j intersect but do not all fall on a line. These paths necessarily exist in a plane and therefore can be reduced to an equivalent $n = 2$ problem. Since these paths intersect, we know that switching the assignment will always reduce the sum of distances by using the triangle inequality and the given ϕ is not optimal for (10). Therefore, the minimum assignment found in (10) will never result in intersecting paths. □

Unfortunately, we can show with the simple example depicted in Figure 3 that agents with finite extent using the assignment from (10) are not guaranteed collision-free trajectories, rendering these trajectories useless for real robots with physical extent. This minimum distance assignment can be used to find trajectories which avoid collisions, but

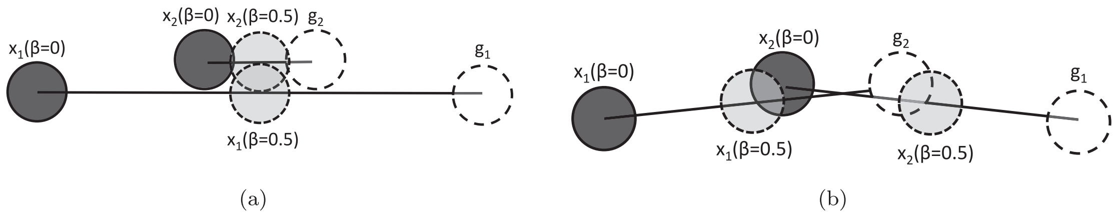


Fig. 3. For the example with two agents in (a) we can see that the minimum sum of distances paths (calculated by (10)) never intersect. However, having intersection-free paths does not guarantee collision-free trajectories for agents with finite size. In this case, merely switching goal assignments, as shown in (b), does ensure collision-free trajectories. It should be noted that minimizing the sum of distance traveled squared arrives at the collision-free assignment in (b).

are suboptimal, may require enlargement of the region $\mathcal{K}$, and are more difficult to compute than those which will be presented in Section 3.2.

3.2. Minimum velocity squared trajectories

The second method we propose is to minimize the sum of the integral of velocity squared traveled by all agents:

$$\begin{aligned} \underset{\phi,\gamma(t)}{\text{minimize}} \quad & \sum_{i=1}^{N} \int_{t_0}^{t_f} \dot{\mathbf{x}}_i(t)^{\mathrm{T}} \dot{\mathbf{x}}_i(t) dt \\ \text{subject to} \quad & (1), (2), (3), (4), (5), (6) \end{aligned}$$

which is equivalent to:

$$\begin{aligned} \underset{\phi,\gamma(t)}{\text{minimize}} \quad & \int_{t_0}^{t_f} \dot{X}(t)^{\mathrm{T}} \dot{X}(t)\, dt \\ \text{subject to} \quad & (1), (2), (3), (4), (5), (6) \end{aligned} \tag{11}$$

We propose C-CAPT as the solution to this problem and detail its development in the remainder of this section.

To clarify how the optimization in Section 3.2 differs from that in Section 3.1, consider moving a contiguous block of a number of books each with identical width to another contiguous block, but moved one book over and ignoring collisions. One solution is to move the first book to the last position, where another is to move each book one position over. Both schemes result in the same sum of distance traveled, however moving each book one unit over results in a lower sum of distances squared as a result of distance squared being a strictly convex cost function. Notice that in the many smaller moves solution, one book will not cross another. To relate this simple example to the CAPT problem, we note that all of the books can be simultaneously shifted to their new location without collision.

We will temporarily relax (11) to ignore the clearance requirements in (6):

$$\begin{aligned} \underset{\phi,\gamma(t)}{\text{minimize}} \quad & \int_{t_0}^{t_f} \dot{X}(t)^{\mathrm{T}} \dot{X}(t)\, dt \\ \text{subject to} \quad & (1), (2), (3), (4), (5) \end{aligned} \tag{12}$$

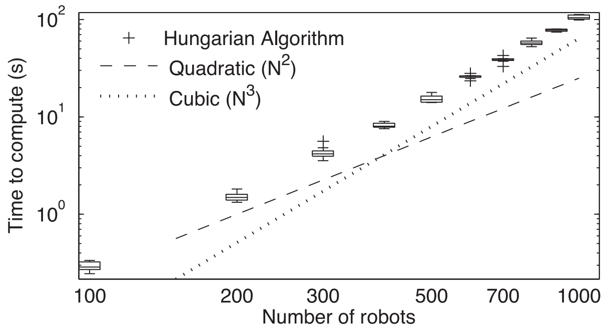


Fig. 4. Solution times for the Hungarian Algorithm to solve (13) in MATLAB on a standard laptop computer. The box-plot is generated based on 10 trials for each value of N. Note that the algorithm runtime trends toward N^3 growth. Boxes represent the 25th and 75th percentiles, whiskers denote the 99% confidence interval and outliers are marked using “+”.

The solution to (12) will consist of straight line trajectories which satisfy the boundary conditions while minimizing the sum of distance traveled squared.

3.2.1. Optimal assignment.

We first consider the assignment problem and create a distance squared matrix $D \in \mathbb{R}^{N \times M}$:

$$D_{i,j} = ||\mathbf{x}_i(t_0) - \mathbf{g}_j||^2 \qquad i \in \mathcal{I}_N, j \in \mathcal{I}_M$$

We then solve for the optimal distance squared assignment matrix $\phi^\star$:

$$\phi^\star = \underset{\phi}{\text{argmin}} \quad \sum_{i=1}^{N} \sum_{j=1}^{M} \phi_{i,j} D_{i,j} \tag{13}$$

We note that $\phi^\star$ satisfies (1) and (2).

This is a linear assignment problem and therefore an optimal assignment solving algorithm such as the Hungarian Algorithm can be used to solve for $\phi^\star$. In practice, we see solve times of under 10 seconds for 400 robot–goal pairs in MATLAB on a standard laptop computer. Figure 4 displays run times that trend toward cubic growth in N.

3.2.2. Trajectory generation. The termination time t_f can be computed as follows:

$$t_f = \underset{i}{\text{maximize}} \quad \frac{\left\| \mathbf{x}_i(t_0) - \sum_{j=1}^{M} \phi^*_{i,j}\mathbf{g}_j \right\|_2}{v_{max}} \tag{14}$$

Define the polynomial function of time:

$$\beta(t) \equiv \sum_{i=0}^{k} \alpha_i t^i \in [0, 1]$$

such that $\beta(t_0) = 0$ and $\beta(t_f) = 1$.

A straight forward application of the calculus of variations shows that the trajectories which minimize the integral of velocity squared are those with constant velocity and satisfy the boundary conditions:

$$\gamma^\star(t) = (1-\beta(t))\,\mathbf{X}(t_0) + \beta(t)\,(\Phi\mathbf{G} + (I_{Nn} - \Phi\Phi^{\mathrm{T}})\,\mathbf{X}(t_0)) \tag{15}$$

where β is defined by the first order polynomial:

$$\alpha_0 = \frac{-t_0}{t_f - t_0}, \quad \alpha_1 = \frac{1}{t_f - t_0}, \quad \alpha_2 = 0 \ \ldots \ \alpha_k = 0$$

It is clear that at $t = t_0$, $\gamma^\star(t_0) = \mathbf{X}(t_0)$, and therefore (15) satisfies the initial conditions in (3).

In equation (15), the term, $I_{Nn} - \Phi\Phi^{\mathrm{T}}$, selects all unassigned robots and ensures that these robots remain at their original location. If $M \geq N$, using (2), this term will disappear as expected, resulting from the fact that all robots will be assigned.

We also verify that $\gamma^\star(t)$ satisfies the final boundary conditions specified in (4):

$$\gamma^\star(t_f) = \Phi\mathbf{G} + (I_{Nn} - \Phi\Phi^{\mathrm{T}})\,\mathbf{X}(t_0)$$

If $M \geq N$, then using (2), $\Phi\Phi^{\mathrm{T}} = I_{Nn}$ and $\gamma^\star(t_f) = \Phi\mathbf{G}$. If instead $N \geq M$, we can premultiply (15) by Φ^{T}:

$$\Phi^{\mathrm{T}}\gamma^\star(t_f) = \Phi^{\mathrm{T}}\Phi\mathbf{G} + (\Phi^{\mathrm{T}}I_{Nn} - \Phi^{\mathrm{T}}\Phi\Phi^{\mathrm{T}})\,\mathbf{X}(t_0)$$

We also know from (2) that $\Phi^{\mathrm{T}}\Phi = I_{Mn}$ to thus verify that $\Phi^{\mathrm{T}}\gamma^\star(t_f) = \mathbf{G}$. Therefore, the trajectories defined in (15) satisfy the terminal conditions in (4).

The definition of the termination time t_f in (14) guarantees that all robots satisfy their actuation bounds in (5).

3.2.3. Collision avoidance. We have shown that (13) and (15) generate the solution to the relaxed problem without considering collisions in (12). However, we will utilize the properties of the CAPT problem to demonstrate in Theorem 3.3 that if $\Delta > 2\sqrt{2}R$, these equations also provide the solution to the full collision avoidance problem in (11).

For notational convenience, we define:

$$\mathbf{r}_{i,j} \equiv \mathbf{x}_j(t_f) - \mathbf{x}_i(t_0) \qquad \mathbf{u}_{ij} \equiv \mathbf{x}_j(t_0) - \mathbf{x}_i(t_0)$$
$$\mathbf{w}_{ij} \equiv \mathbf{x}_j(t_f) - \mathbf{x}_i(t_f)$$

We first prove a lemma related to the geometry of the optimal solutions.

Lemma 3.2. *The optimal solutions to* (12) *satisfy:*

$$\mathbf{w}_{i,j}{}^{\mathrm{T}}\mathbf{u}_{i,j} \geq 0 \ \forall i,j \in \mathcal{I}_N \tag{16}$$

Proof. We globally minimized the sum of integrated velocity squared in (12) such that switching goal states of agent i with agent j will not decrease the sum of distance squared, or:

$$||\mathbf{r}_{i,i}||^2 + ||\mathbf{r}_{j,j}||^2 \leq ||\mathbf{r}_{i,j}||^2 + ||\mathbf{r}_{j,i}||^2 \ \forall i,j \in \mathcal{I}_N \tag{17}$$

We then substitute:

$$\begin{aligned} ||\mathbf{r}_{i,j}||^2 &= \mathbf{r}_{i,j}{}^{\mathrm{T}}\mathbf{r}_{i,j} \\ &= \mathbf{x}_j(t_f)^{\mathrm{T}}\mathbf{x}_j(t_f) - 2\mathbf{x}_i(t_0)^{\mathrm{T}}\mathbf{x}_j(t_f) + \mathbf{x}_i(t_0)^{\mathrm{T}}\mathbf{x}_i(t_0) \end{aligned}$$

into (17) and simplify:

$$(\mathbf{x}_j(t_f) - \mathbf{x}_i(t_f))^{\mathrm{T}}(\mathbf{x}_j(t_0) - \mathbf{x}_i(t_0)) \geq 0 \ \forall i,j \in \mathcal{I}_N$$

or

$$\mathbf{w}_{i,j}{}^{\mathrm{T}}\mathbf{u}_{i,j} \geq 0 \ \forall i,j \in \mathcal{I}_N \tag{18}$$

□

Theorem 3.3. *If* $\Delta > 2\sqrt{2}R$*, trajectories in* (15) *will satisfy* (6) *and be collision-free.*

Proof. The location of robot i following the trajectory specified in (15) is:

$$\mathbf{x}_i(t) = (1 - \beta)\,\mathbf{x}_i(t_0) + \beta\mathbf{x}_i(t_f)$$

Therefore the distance between robots i and j is:

$$\begin{aligned} ||\mathbf{x}_j(t) - \mathbf{x}_i(t)|| = ||&(1-\beta)\,\mathbf{x}_j(t_0) + \beta\mathbf{x}_j(t_f) \\ &- (1-\beta)\,\mathbf{x}_i(t_0) - \beta\mathbf{x}_i(t_f)|| \end{aligned}$$

and the distance squared between these robots is:

$$\begin{aligned} ||\mathbf{x}_j - \mathbf{x}_i||^2 &= ||(1-\beta)\,(\mathbf{x}_j(t_0) - \mathbf{x}_i(t_0)) + \beta(\mathbf{x}_j(t_f) - \mathbf{x}_i(t_f))||^2 \\ &= ||\mathbf{u}_{ij} + \beta(\mathbf{w}_{ij} - \mathbf{u}_{ij})\,||^2 \end{aligned}$$

We can simplify this result:

$$||\mathbf{x}_j - \mathbf{x}_i||^2 = (\mathbf{u}_{ij} + \beta(\mathbf{w}_{ij} - \mathbf{u}_{ij}))^{\mathrm{T}}(\mathbf{u}_{ij} + \beta(\mathbf{w}_{ij} - \mathbf{u}_{ij})) \tag{19}$$

Define for notational convenience:

$$a \equiv \mathbf{u}_{ij}{}^{\mathrm{T}}\mathbf{u}_{ij} \qquad b \equiv \mathbf{w}_{ij}{}^{\mathrm{T}}\mathbf{u}_{ij} \qquad c \equiv \mathbf{w}_{ij}{}^{\mathrm{T}}\mathbf{w}_{ij}$$

Equation (19) simplifies to:

$$||\mathbf{x}_i - \mathbf{x}_j||^2 = a - 2\beta(a - b) + \beta^2(a - 2b + c) \tag{20}$$

We can find the value of β which minimizes the distance squared (and therefore the distance) between agents i and j ($i \neq j$):

$$\beta^\star_{i,j} = \underset{\beta}{\text{argmin}} \ ||\mathbf{x}_i - \mathbf{x}_j|| = \frac{a - b}{a - 2b + c} \tag{21}$$

If $\beta^{\star}_{i,j}$ is outside the range [0, 1], the agents are at a minimum at either the start or end of the trajectory and the agents will not collide due to (A4). If, however, $\beta^{\star}_{i,j} \in [0, 1]$, the minimum distance between robots i and j over the course of their planned trajectories can be found by combining (20) and (21). The minimum distance squared is:

$$||\mathbf{x}_i - \mathbf{x}_j||^2_{\min} = \frac{ac - b^2}{a - 2b + c} \qquad (22)$$

We will now seek to find starting conditions which minimize this minimum distance. From their definitions, we see that a, b, and c are all independent except that $b < c$ and $b < a$ are guaranteed from the fact that $\beta \in [0, 1]$. It is clear that this minimum–minimum distance decreases as b decreases. However, as shown in Lemma 3.2, $b \geq 0$ and the minimum possible distance between robots occurs when $b = 0$:

$$||\mathbf{x}_i - \mathbf{x}_j||^2_{\min} \geq \frac{ac}{a + c}$$

Assumption (A4) specifies:

$$||\mathbf{x}_i(t_0) - \mathbf{x}_j(t_0)|| > \Delta \qquad ||\mathbf{x}_i(t_f) - \mathbf{x}_j(t_f)|| > \Delta$$

These are equivalent to:

$$\sqrt{\mathbf{u}_{ij}{}^{\mathrm{T}}\mathbf{u}_{ij}} > \Delta \qquad \sqrt{\mathbf{w}_{ij}{}^{\mathrm{T}}\mathbf{w}_{ij}} > \Delta$$

From the assumption $\Delta > 2\sqrt{2}R$, these inequalities are equivalent to:

$$a > 8R^2 \qquad c > 8R^2$$

Using these inequalities, it is straightforward to show:

$$||\mathbf{x}_i - \mathbf{x}_j||^2_{\min} \geq \frac{ac}{a + c} > 4R^2$$

And therefore:

$$||\mathbf{x}_i - \mathbf{x}_j||_{\min} > 2R$$

Thus a robot with radius R can never intersect another robot with radius R and collision avoidance is guaranteed. □

3.3. C-CAPT *Definition*

We define C-CAPT as the solution to (11). Equation (23) reiterates (13) and (15), which we have shown provides collision-free trajectories that satisfy all requirements for the obstacle-free CAPT problem.

$$\begin{aligned} \phi^{\star} &= \underset{\phi}{\operatorname{argmin}} \sum_{i=1}^{N}\sum_{j=1}^{M} \phi_{i,j} D_{i,j} \\ \gamma^{\star}(t) &= (1 - \beta(t))\,\mathbf{X}(t_0) \\ &\quad + \beta(t)\,(\Phi^{\star}\mathbf{G} + (I_{Nn} - \Phi^{\star}\Phi^{\star\mathrm{T}})\,\mathbf{X}(t_0)) \end{aligned} \qquad (23)$$

Algorithm 1 C-CAPT with $M > N$

```
while $M > N$ do
    C-CAPT
    remove goals assigned by $\phi^{\star}$
C-CAPT
```

3.4. C-CAPT *with fewer robots than goals*

In the event that $M > N$, or there are more goal locations to visit than robots to assign, there are many possible solutions that may be considered to solve the CAPT problem. One potential algorithm design is to use C-CAPT to find the set of trajectories which safely navigate the robots to the goals which minimize the sum of distance traveled squared and consider the problem solved. This is appropriate when the mission goal is for each robot to be occupying one of the goal locations.

In the case that the operator wants the robots to inspect each of the goal locations, the problem could alternately be cast as an instance of the vehicle routing problem (VRP). There is a substantial body of literature (see, for example, Bertsimas and Van Ryzin (1991), Laporte (1992), and Bullo et al. (2009)) that address the vehicle routing problem on the Euclidean plane, but these works typically ignore the collision avoidance constraint. In the context of CAPT, robots may be reassigned to a new goal after its original goal has been visited. We present Algorithm 1, a simple solution to the VRP that guarantees all goals will be visited while ensuring collision avoidance and dynamic constraints by iteratively solving C-CAPT for assignments and trajectories until there are no remaining unreached goal locations. See Figure 14 for an example with 6 robots and 20 goal locations.

While the paths generated by Algorithm 1 provide a solution to the VRP, the solution is not necessarily optimal. Each iteration of C-CAPT guarantees local optimality in the sense of (11). For $M \approx N$, Algorithm 1 generates solutions very close to the optimal solution of the vehicle routing problem. As $M \gg N$, the benefits from solving the CAPT problem diminish and solutions from Algorithm 1 appear similar to a greedy solution of the traveling salesman problem.

4. C-CAPT Using a team of quadrotors

The past few years have seen quadrotors emerging as the unmanned micro-aerial vehicle platform of choice. Indeed in our previous work, Turpin et al. (2012), we have utilized quadrotors due to their simple construction, vertical take off and landing abilities, and high agility. As a result of this simple design, the dynamics of the quadrotor necessitate $\mathbb{C}^3$ smooth trajectories for accurate tracking. It is well known that minimizing the square of the norm of the snap (the fourth derivative of position) of the trajectory yields reference trajectories and inputs that are excellent for quadrotors

(Mellinger and Kumar, 2011). Fortunately, this only results in a simple modification to the cost function in (11):

$$\begin{aligned} \underset{\phi,\gamma(t)}{\text{minimize}} \quad & \int_{t_0}^{t_f} \overset{\cdots\cdot}{X}(t)^{\mathrm{T}} \overset{\cdots\cdot}{X}(t)\, dt \\ \text{subject to} \quad & (1), (2), (3), (4), (6), \end{aligned} \tag{24}$$

A straight forward application of optimal control theory shows that each of the individual robot trajectories returned from this optimization will be those that minimize snap along straight line paths from the initial position to the goal location. It also follows that these minimum snap trajectories are seventh order polynomial functions of time. Because we choose homogeneous boundary conditions (zero velocity, acceleration, and jerk at the initial and final conditions) the minimum snap trajectory has the same path as before[2] but reparameterized by $\beta(t)$ with $k = 7$.

Because the boundary conditions are homogeneous, the integral of snap squared for a given trajectory will be exactly a constant factor times the integral of velocity squared in the original problem statement (23). This factor is constant for all assignments and therefore, the optimal assignment is independent of $\beta(t)$, and is the same as that obtained by solving the original problem.

Aerodynamic interactions of the quadrotors require additional considerations for safety of the team. A robot flying below another robot is adversely affected by the downwash from the higher robot and will be unable to track trajectories satisfactorily. For this reason, we use an ellipsoidal model of the quadrotor to ensure that the separation in the vertical direction is larger than the separation in the horizontal direction:

$$\sqrt{\left(\frac{x_i(t)-x_j(t)}{1}\right)^2 + \left(\frac{y_i(t)-y_j(t)}{1}\right)^2 + \left(\frac{z_i(t)-z_j(t)}{4}\right)^2} > 2R \quad \forall\, i, j, t$$

This condition means that for a robot to fly directly above another robot, the separation in the vertical direction must be at least $8R$. Using a change of variables where $\hat{z} = z/4$, we can use the original boundary constraints (3), (4), and minimum clearance conditions (6). The returned trajectories must then be transformed back into the original space to guarantee the required ellipsoidal constraints are satisfied.

The robots used for experimentation are shown in Figure 5. A bank of motion capture cameras provides feedback as shown in Figure 6. More details on the robot dynamics and the control software are available in Michael et al. (2010) and Mellinger and Kumar (2011).

We have conducted numerous experiments using 4 to 16 quadrotors. In this paper we present representative results from experiments in which a team of eight robots was tasked to visit a sequence of randomly generated goal locations. Figure 7 shows the sequence of waypoints visited. All experiments resulted in safe execution and completion of the task. Minimum clearance between robots for the entire experiment is shown in Figure 8. It is clear from this plot that all robots were never in or near a collision state with any other robot.

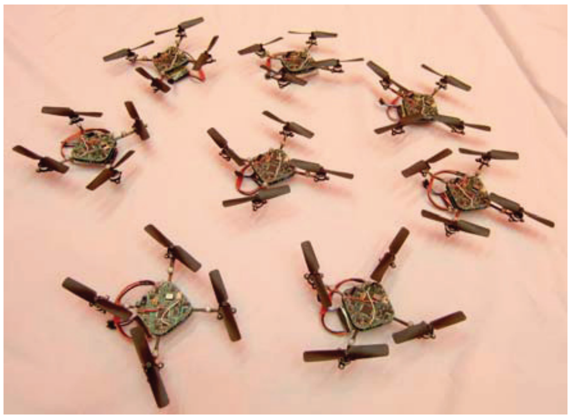


Fig. 5. Team of eight KMel Robotics NanoQuad quadrotors used in experimentation. More information about these robots is available at www.kmelrobotics.com.

Fig. 6. Eight robots flying in an indoor laboratory with feedback from motion capture cameras.

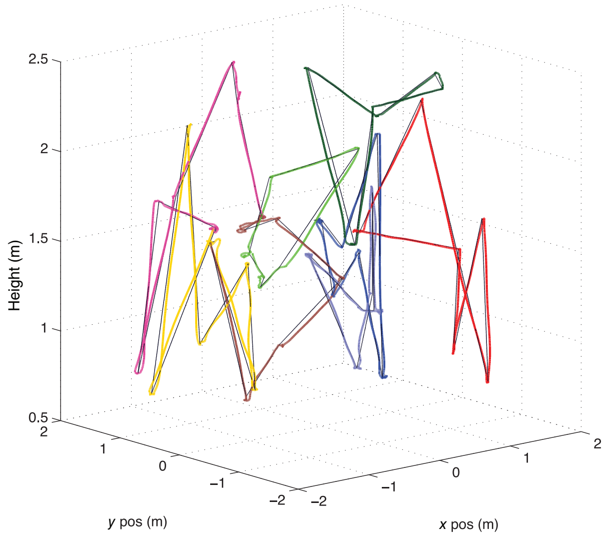


Fig. 7. The three-dimensional trajectories of the team of eight quadrotor robots are shown here. The thin black lines show the desired trajectories and each colored line shows the actual paths followed of a particular robot.

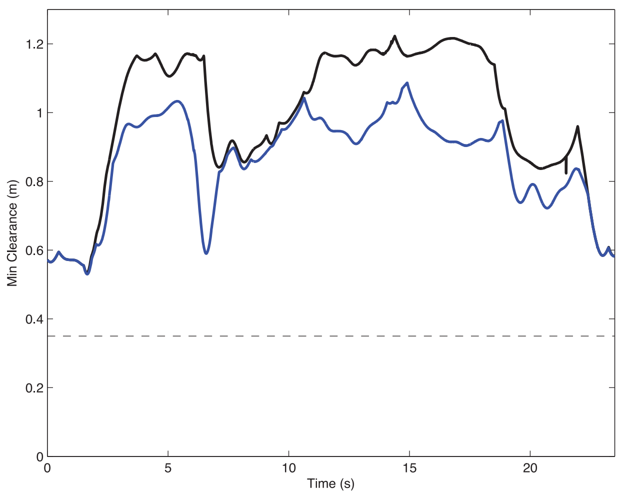


Fig. 8. Plot of the minimum clearance between any two robots while tracking their trajectories. The upper black line represents the actual separation. The blue line represents the transformed parabolic minimum clearance and the dotted line shows the actual diameter of the robots. Notice that collision avoidance is clear from the fact that the clearance is always larger than a robot's diameter.

5. D-CAPT: A decentralized algorithm for obstacle-free CAPT

In this section, we exploit our knowledge of properties of the centralized minimum velocity squared solution proposed in Section 3.2 to formulate a computationally tractable, online, decentralized algorithm which will guarantee collision-free paths for all robots. The decentralized method takes inspiration from (18) and is based on reassignment of goal locations to arrive at a locally optimal solution.

We first introduce a suitably defined communication or sensing range, h. When a robot comes within h of a neighbor, it must coordinate its actions with its neighbor requiring a partially centralized solution. h must be large enough so that robots can ignore neighbors outside this distance. And yet h must be small enough so that robots can estimate positions of neighbors within this distance and communicate with their neighbors. While this serves as a convenient abstraction to solve our problems, it is also realistic of small robots communicating over 2.4 GHz in urban environments with small, narrow-field-of-view cameras.

For very large systems, solving the Hungarian Algorithm begins to be prohibitively expensive as seen in Figure 4. This, coupled with the fact that interactions are inherently local, means that the decentralized algorithm with artificially reduced robot interaction sets may be preferred even when considering a fully connected network with centralized coordinator to improve computational performance at the cost of optimality.

We require (A1)–(A5) as well as two additional assumptions (A6)–(A7):

Algorithm 2 $\mathbf{x}_i(t) = \text{D-CAPT}(\mathbf{x}_i(t_0), \mathbf{f}_i)$

```
1:  compute trajectory using (26)
2:  $\mathcal{U}_i = \mathcal{C}_i(t_0)$
3:  $t_{prev} \leftarrow t_0$
4:  while $t < t_f$ do
5:      $t_c \leftarrow t$
6:
7:      for $j \in \mathcal{C}_i(t_c)$ do
8:          if $j \notin \mathcal{C}_i(t_{\text{prev}})$ then
9:              $\mathcal{U}_i = \mathcal{U}_i \bigcup j$
10:     $\mathcal{U}_i = \mathcal{U}_i \bigcap \mathcal{C}_i(t_c)$
11:
12:     for $j \in \mathcal{U}_i$ do
13:         request $\mathbf{x}_j(t_c)$ and $\mathbf{f}_j$ from agent $j$
14:         if $\mathbf{u}_{i,j}{}^{\mathrm{T}}\mathbf{w}_{i,j} < 0$ then
15:             send to robot $j$ to change goal location to $\mathbf{f}_i$
16:             $\mathbf{f}_i \leftarrow \mathbf{f}_j$
17:             $\mathcal{U}_i = \mathcal{C}_i(t_c)$
18:             recompute trajectory using (26)
19:         $\mathcal{U}_i = \mathcal{U}_i \setminus j$
20:
21:     if robot $j$ sends $\mathbf{f}_j$ as the new robot $i$ goal then
22:         $\mathbf{f}_i \leftarrow \mathbf{f}_j$
23:         $\mathcal{U}_i = \mathcal{C}_i(t_c)$
24:         recompute trajectory using (26)
25:         $\mathcal{U}_i = \mathcal{U}_i \setminus j$
26:     $t_{prev} \leftarrow t_c$
```

(A6) All robots are capable of exchanging information about their current state and their assigned goals to other robots closer than distance h. Of course h must be greater than Δ. If communication or sensing take significant time, we must ensure $h >> \Delta$.

(A7) $N = M$, where each goal location is initially assigned to exactly one robot.

As a result of robots communicating with neighboring robots within the communications range h, a moving robot can constantly be encountering new neighbors, and therefore learn new information about its neighbors, and by extension, information from its neighbors' neighbors and so on. The key feature of this algorithm is for every message sent, the system is locally minimizing a modified version of the cost functional in (11). Before we present the decentralized algorithm, Algorithm 2, we first introduce some new notation.

The robot–goal assignment ϕ is no longer known by any one robot, but rather distributed knowledge of the system of robots. Therefore we define $\mathbf{f}_i$ as the goal currently assigned to robot i.

We now define the proximity set $\mathcal{C}_i(t)$ as a list of all robots within the communications range of agent i at time t:

$$\mathcal{C}_i(t) = \{j \mid \|\mathbf{x}_j(t) - \mathbf{x}_i(t)\| \leq h, j \neq i\} \subset \mathcal{I}_N \tag{25}$$

We define the update list $\mathcal{U}_i(t) \subset \mathcal{C}_i(t)$ as the list of robots to which robot i will attempt to send new information.

We will use t_c to denote the current time of computation such that $t_0 \leq t_c < t_f$. We also modify the definitions of $\mathbf{u}_{i,j}$ and $\mathbf{r}_{i,j}$:

$$\mathbf{r}_{i,j} \equiv \mathbf{x}_j(t_f) - \mathbf{x}_i(t_c) \qquad \mathbf{u}_{i,j} \equiv \mathbf{x}_j(t_c) - \mathbf{x}_i(t_c)$$

In D-CAPT presented in Algorithm 2, the ith robot locally minimizes the contribution to (11) from every pair of i and j that satisfy (25):

$$\underset{\mathbf{f}_i, \mathbf{f}_j, \gamma(t)}{\text{minimize}} \quad \int_{t_c}^{t_f} \dot{x}_i(t)^{\mathrm{T}} \dot{x}_i(t)\, dt + \int_{t_c}^{t_f} \dot{x}_j(t)^{\mathrm{T}} \dot{x}_j(t)\, dt$$

The trajectory for the remaining time $t \in [t_c, t_f]$ is computed in a similar fashion to the centralized version (15):

$$\mathbf{x}_i(t) = \left(1 - \frac{t - t_c}{t_f - t_c}\right) \mathbf{x}_i(t_c) + \left(\frac{t - t_c}{t_f - t_c}\right) \mathbf{f}_i \tag{26}$$

Note that Algorithm 2 ensures only new information is transmitted to the robots in $\mathcal{C}_i(t)$ without making unnecessary communications. Using similar reasoning to Lemma 3.2, we will show in Lemma 5.1 that Algorithm 2 converges to a locally optimal solution to (11).

Lemma 5.1. *A change in goal locations between any pair of robots i and j in Algorithm 2 results in a decrease of the sum of distance remaining squared.*

Proof. We show that when a pair of robots exchange goal locations, the sum of distance squared remaining for those two agents decreases. All other robots are unaffected by the trade so the total sum of distance remaining squared decreases for the whole system.

After two robots trade their assigned goals, we have:

$$\mathbf{u}_{i,j}{}^{\mathrm{T}} \mathbf{w}_{i,j} > 0$$

Using algebra similar to that in Lemma 3.2, we find that:

$$||\mathbf{r}_{i,i}||^2 + ||\mathbf{r}_{j,j}||^2 < ||\mathbf{r}_{i,j}||^2 + ||\mathbf{r}_{j,i}||^2$$

In other words, the sum of remaining distance squared has decreased from the value before the reassignment and the re-planning. $\square$

Theorem 5.2. *Algorithm 2 results in each goal being occupied by one robot without any collisions.*

Proof. Define the cost-to-go function V:

$$V = \sum_{i=1}^{M} ||\mathbf{x}_i - \mathbf{f}_i||^2 \tag{27}$$

For an arbitrarily small constant, ϵ, we can use Lemma 5.1 and the trajectories defined in (26) to see that V is strictly decreasing:

$$V(t + \epsilon) < V(t)$$

Further, by (26), the final value of the cost-to-go function will be zero:

$$V(t_f) = 0.$$

From Theorem 3.3, we know all trajectories will be free of collisions if all assumptions are met for each pairwise interaction. $\square$

This pairwise algorithm is valid when the distance constraints in (A4) are met at the start of a pairwise interaction. However, there are pathological inputs that break these initial conditions. Appendix 7 addresses this in greater detail and provides two potential solutions. We note here that these conditions are extremely unlikely to arise in practice.

6. Simulation results

In this section, we simulate C-CAPT and D-CAPT on a large variety of boundary conditions to study their performance. For each trial, we randomly generate starting and goal locations that satisfy (A4).

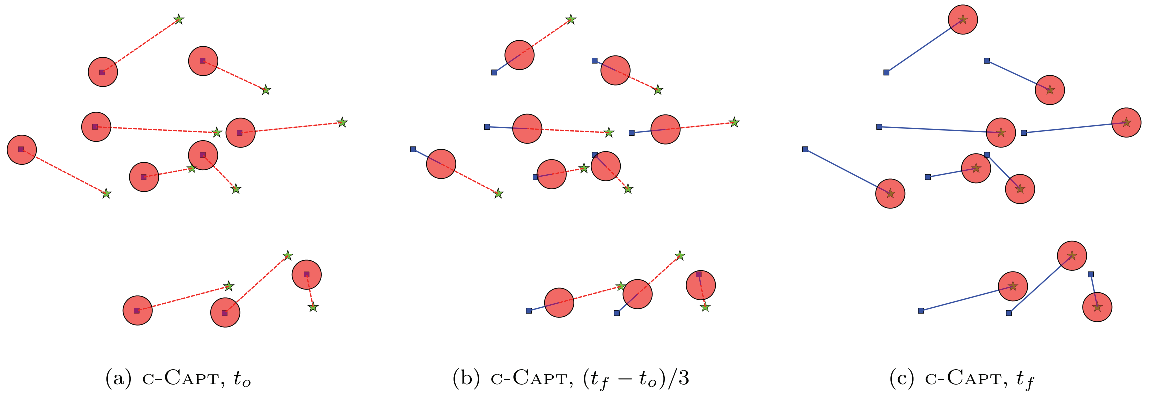
The first set of simulations we present is shown in Figure 9 compares C-CAPT and D-CAPT on the same set of initial locations for 10 robots and 10 goal locations. In this example, $h = 1.1\Delta$, very near the lower limit ($h > \Delta$). In general, h should be much larger than this in an experimental system to account for real world uncertainties.

Shown in Figure 10 is the verification that the energy function defined in (27) is indeed strictly decrescent by using a D-CAPT simulation with $N = M = 100$ in three dimensions with $h = 1.5\Delta$. It can be easily seen that $\dot{V} < 0$ and $V(t_f) = 0$.

In the third set of simulation runs we explore the scaling of D-CAPT as the number of robots increases. In Figure 11, we vary $N = M$ from 2 to 50 and present the result for 100 trials with $\frac{h}{\Delta} >> 1$ in three dimensions. We also note that the number of reassignments increases approximately linearly in N. We can also see that the total number of communications grows approximately as N^2. We compare the sum of distance traveled squared to the optimal value from C-CAPT. It should be noted that even with a fully connected network, optimality is not guaranteed with more than two robots. To see this, we show a simple example with three robots in Figure 12.

The next set of simulations in Figure 13 vary communications ranges used by D-CAPT for $N = M = 20$. The communications range is varied from the minimum value of $\frac{h}{\Delta} = 1$ through large values ($\frac{h}{\Delta} >> 1$). Note that the number of messages sent decreases as the communications range decreases at the cost of becoming quite suboptimal. Additionally, the minimum clearance δ between robots decreases with smaller communications ranges as one might expect, but always satisfy the clearance requirement in (6).

Finally, we demonstrate in Figure 14 C-CAPT with fewer agents than goals as outlined in Section 3.4. In this



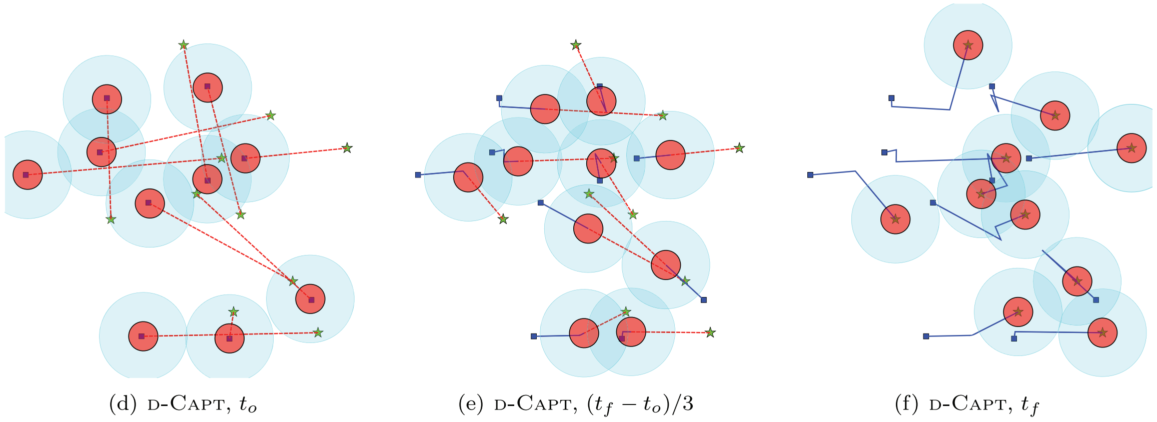


Fig. 9. We compare c-Capt and d-Capt simulations of 10 robots seeking 10 goal locations. In all figures, blue squares represent starting locations, green stars represent goal locations, small red circles represent the robots, red dotted lines indicate the planned trajectory, blue solid lines indicate the path already traversed, and cyan larger circles represent the communication range.

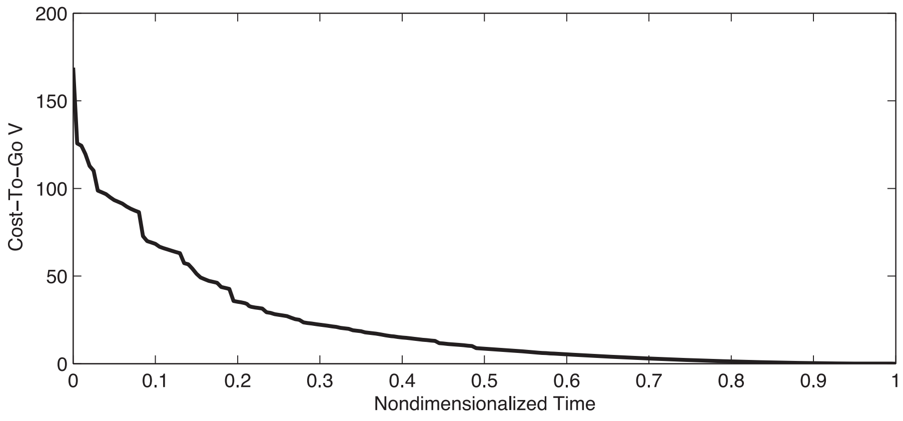
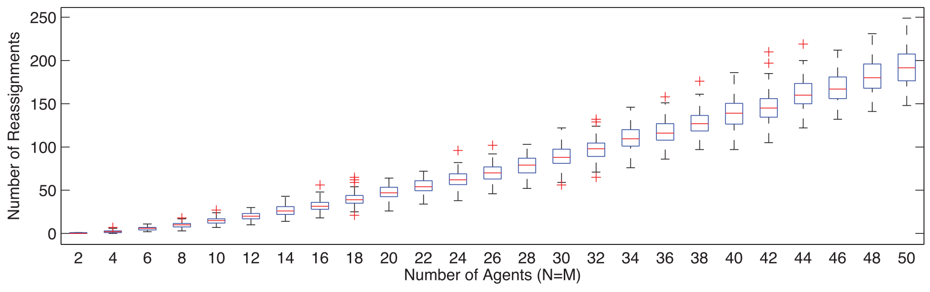


Fig. 10. Visualization of decay of the cost-to-go function V in (27) for d-Capt with $N = M = 100$, and $n = 3$. The instantaneous drops are a result of reassignments and the continuous decay is a result of trajectory tracking.

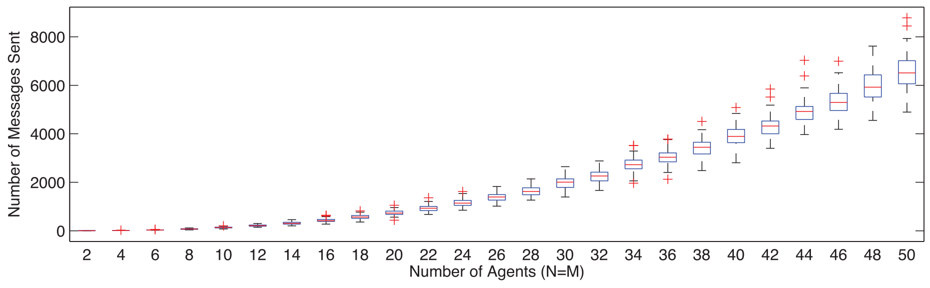
example, a brute force computation of the solution is completely infeasible and we provide a feasible collision-free trajectories using Capt.

7. Conclusion

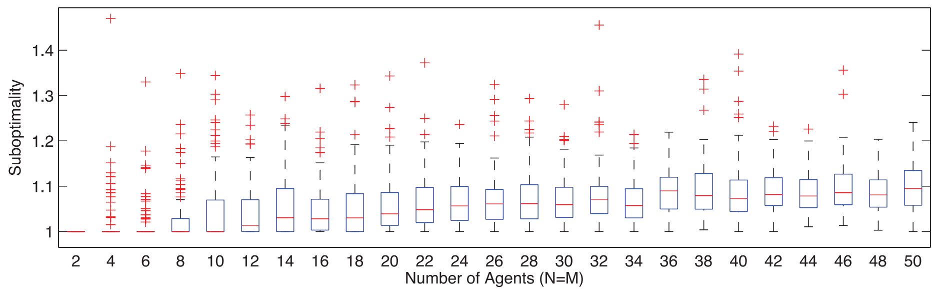
In this paper, we discuss the problem of concurrent assignment and planning trajectories for multi-robot systems. We first develop c-Capt, a centralized solution to the problem of assigning goals and planning trajectories that minimize a cost functional based on the square of velocity along the trajectory and show that the resulting trajectories are globally optimal and safe. This is modified to be suitable for use on highly dynamic quadrotor micro aerial vehicle robots. The algorithm is then applied to a team of eight robots, which successfully arrive at their goal locations while



(a)

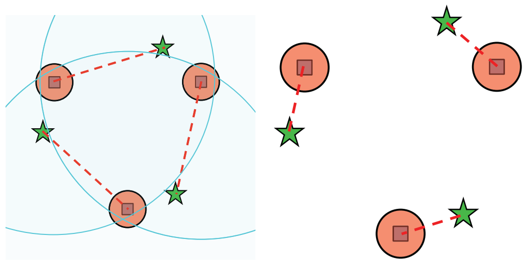


(b)



(c)

Fig. 11. Box plots of D-CAPT statistics on 100 simulated three-dimensional random configurations and initial assignments for $M = N$ robots and goals. (c) shows the distance traveled squared divided by the optimal value returned from C-CAPT. The "+" marks designate statistical outliers.



(a) (b)

Fig. 12. Simple case when D-CAPT does not find optimal solution for a fully connected team of $n > 2$ robots. For each pair of robots the scenario in (a), exchanging goals will not reduce the sum of distance squared. However, a permutation to modify all assignments to that in (b) does lower the sum of distance squared.

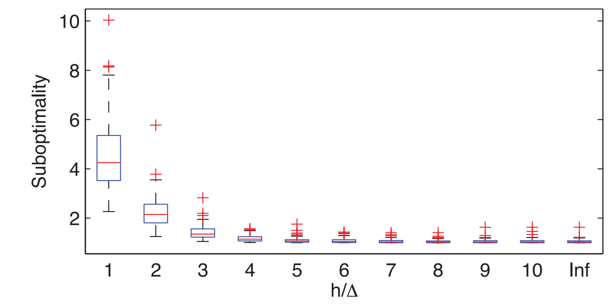
avoiding collisions. We then develop D-CAPT, a decentralized algorithm that relies on the exchange of information between robots within communication range. The algorithm requires local reassignment and re-planning across a pair of robots when maximum distance traveled can be reduced while simultaneously increasing minimum clearance. We show this algorithm yields suboptimal assignments but safe trajectories.

Funding

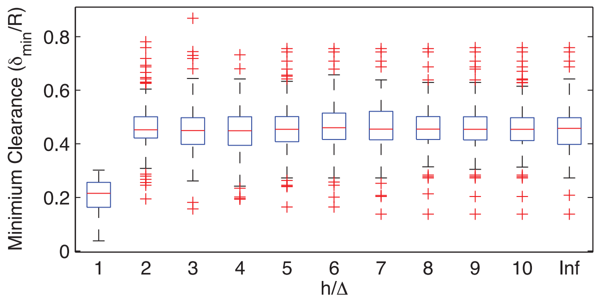
This work was supported by the Office of Naval Research (grant numbers N00014-07-1-0829, N00014-08-1-0696, and N00014-09-1-1051), the National Science Foundation (grant number CCF-1138847), the Army Research Laboratory (grant number W911NF-08-2-0004). Matthew Turpin was supported by NSF Fellowship Grant DGE-0822.

Notes

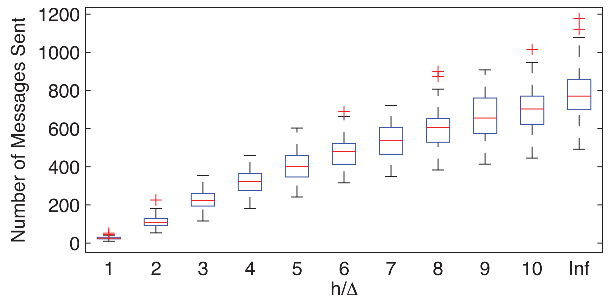
1. The assumption of homogeneity requires robots to be identical and not have any special capabilities that would lead to robots of a certain type being preferred for certain goals/tasks over others. However, even in the heterogeneous case, it may be useful to consider the abstraction of a team of N robots with



(a)



(b)



(c)

Fig. 13. Box plots of D-CAPT statistics on 100 simulated three-dimensional random configurations and initial assignments with varying communications distance h with $N = M = 20$. (c) shows that as the communications range decreases, the number of messages sent using D-CAPT drastically decreases at the expense of clearance in (b) and optimality in (a). Note that despite these small communications distances, the clearance requirement in (6) is never violated. "+" marks designate statistical outliers.

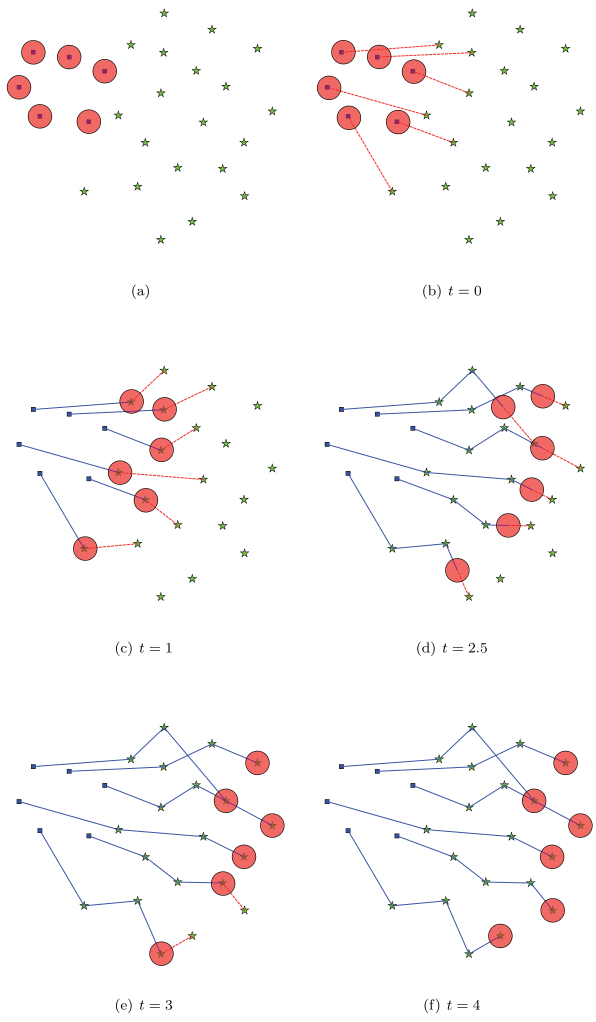


Fig. 14. We display C-CAPT iteratively evaluating all goal locations when there are more goals than robots. (a) shows the initial conditions with 6 robots and 20 goal locations. (b) shows the first assignment of robots to goals and once this is completed. (c) shows another optimal assignment. (d) displays the midpoint of the third segment for all robots. (e) shows the case where there are now more remaining unvisited goals than robots and we can see that only some of the robots have assigned goals. (f) shows the final state of system.

m different types in which the robots of the same type are interchangeable.

2. This analysis can be easily applied to any robotic system. For example, a mobile ground robot which can be modeled as a second order system would require a third order polynomial $\beta(t)$ with $k = 3$.

References

Alonso-Mora J, Breitenmoser A, Rufli M, et al. (2012) Image and animation display with multiple mobile robots. *The International Journal of Robotics Research* 31(6): 753–773.

Bertsimas DJ and Van Ryzin G (1991) A stochastic and dynamic vehicle routing problem in the Euclidean plane. *Operations Research* 39(4): 601–615.

Bullo F, Cortés J and Martínez S (2009) *Distributed Control of Robotic Networks* (*Applied Mathematics Series*). Princeton, NJ: Princeton University Press.

Chaimowicz L, Michael N and Kumar V (2005) Controlling swarms of robots using interpolated implicit functions. In: *Proceedings of the IEEE International conference on robotics and automation*, Barcelona, 18–22 April 2005, pp. 2487–2492.

Enright J and Wurman P (2011) Optimization and coordinated autonomy in mobile fulfillment systems. In: *Proceedings of AAAI conference on artificial intelligence*, San Francisco, CA, 7–11 August 2011, Vol. 1, p. 2.

Erdmann M and Lozano-Perez T (1986) On multiple moving objects. In: *IEEE Transactions on Robotics and Automation* 3: 1419–1424.

Ji M, Azuma S and Egerstedt MB (2006) Role-assignment in multi-agent coordination. *International Journal of Assistive Robotics and Mechatronics* 7(1): 32–40.

Kant K and Zucker SW (1986) Toward efficient trajectory planning: The path-velocity decomposition. *International Journal of Robotics Research* 5(3): 72–89.

Kloder S and Hutchinson S (2006) Path planning for permutation-invariant multirobot formations. *IEEE Transactions on Robotics* 22(4): 650–665.

Kuhn HW (1955) The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly* 2(1–2): 83–97.

Laporte G (1992) The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research* 59(3): 345–358.

LaValle SM (2006) *Planning Algorithms*. Cambridge: Cambridge University Press.

Lindsey Q, Mellinger D and Kumar V (2012) Construction with quadrotor teams. *Autonomous Robots* 33: 323–336.

Liu L and Shell DA (2011) Multi-level partitioning and distribution of the assignment problem for large-scale multi-robot task allocation. In: *Proceedings of 2011 robotics: Science and systems*, Los Angeles, CA, 27–30 June 2011, pp. 185–192.

Mellinger D and Kumar V (2011) Minimum snap trajectory generation and control for quadrotors. In: *Proceedings of the IEEE International conference on robotics and automation*, Shangai, China, 9–13 May 2011, pp. 2520–2525.

Michael N, Mellinger D, Lindsey Q, et al. (2010) The GRASP multiple micro UAV testbed. *IEEE Robotics and Automation Magazine* 17(3): 56–65.

Molnár P and Starke J (2001) Control of distributed autonomous robotic systems using principles of pattern formation in nature and pedestrian behavior. *IEEE Transactions on Systems, Man, and Cybernetics—Part B, Cybernetics* 31(3): 433–435.

Munkres J (1957) Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics* 5(1): 32–38.

Oung R and D'Andrea R (2011) The distributed flight array. *Mechatronics* 21(6): 908–917.

Rendl F (1988) On the Euclidean assignment problem. *Journal of Computational and Applied Mathematics* 23(3): 257–265.

Smith SL and Bullo F (2007) Target assignment for robotic networks: Asymptotic performance under limited communication. In: *Proceedings of the American control conference*, New York, 11–13 July 2007, pp. 1155–1160.

TEMP (n.d.) Tactically expandable maritime platform (TEMP). Available at: http://www.darpa.mil/Our_Work/TTO/Programs/Tactically_Expandable_Maritime_Platform_(TEMP).aspx.

Turpin M, Michael N and Kumar V (2012) Trajectory design and control for aggressive formation flight with quadrotors. *Autonomous Robots* 33(1-2): 143–156.

Wagner G and Choset H (2011) M*: A complete multirobot path planning algorithm with performance bounds. In: *Proceedings of the IEEE/RSJ International conference on intelligent robots and systems*, San Francisco, CA, 25–30 September 2011, pp. 3260–3267.

Wagner G, Kang M and Choset H (2012) Probabilistic path planning for multiple robots with subdimensional expansion. In: *Proceedings of the IEEE International conference on robotics and automation*, Minneapolis, MN, 14–18 May 2012, pp. 2886–2892.

Werfel J, Ingber D and Nagpal R (2007) Collective construction of environmentally-adaptive structures. In: *Proceedings of the IEEE/RSJ International conference on intelligent robots and systems*, San Diego, CA, 29 October–2 November 2007, pp. 2345–2352.

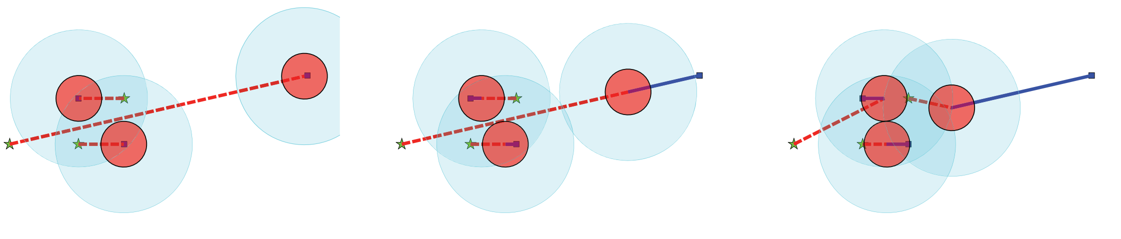
Zavlanos MM and Pappas GJ (2007) Potential fields for maintaining connectivity of mobile networks. *IEEE Transactions on Robotics* 23(4): 812–816.

Appendix A: Pathological case for D-CAPT and resolution

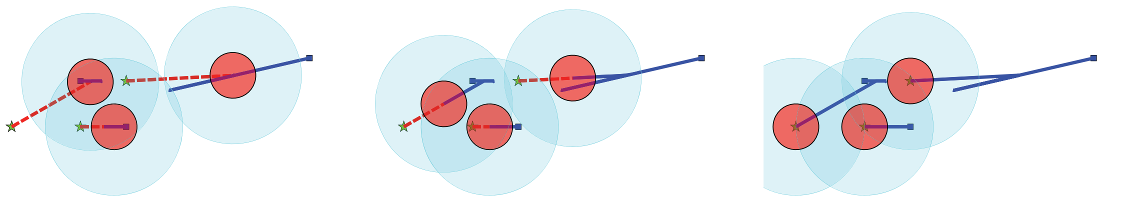
In D-CAPT presented in Section 5, there is a class of initial conditions that can break the guarantee of collision avoidance for some pairs of robots. This pathological case arises when multiple unconnected networks of robots merge and the initial conditions in equation (3) are not satisfied. The most basic example of this case can be visualized in the three robot example in Figure 15.

One potential solution is for all agents in the network to reverse directions long enough to ensure that the new trajectories are collision-free as displayed in Figs. 15(d)–(f). This however requires all agents in the connected network to coordinate this reverse. Another solution is to control affected robots to spread out enough while remaining in $\mathcal{K}$ until equation (3) is satisfied and then resuming Algorithm 2 until convergence.

Fortunately, the conditions for this scenario to arise are extremely rare. Over hundreds of millions of simulated robot–robot communications using random starting locations, goals, and initial assignments, this scenario has never presented itself.



(a) (b) (c)



(d) (e) (f)

Fig. 15. (a) displays initial conditions which will result in a failure of D-CAPT. (b) shows progress along the path, but (c) shows the time step immediately after the robots have converged to a connected graph and we see that the left most robots both instantaneously make progress toward their goals. (d) shows the robots reversing along their paths until Figure (e) and (f) show the algorithm successfully navigating all robots to goals.