Trajectory Planning and Assignment in Multirobot Systems

Matthew Turpin, Nathan Michael, and Vijay Kumar

Abstract In this paper, we consider the problem of tasking large numbers of homogenous robots to move to a set of specified goal locations, addressing both the assignment and trajectory planning subproblems concurrently. This is related to the standard linear Euclidean assignment problem except that the solution to the trajectory generation subproblem must result in time-parameterized trajectories and guarantee collision avoidance. We begin with a centralized approach and derive an optimal centralized solution and study the computational complexity. The main contribution of this paper, however, is a decentralized algorithm with limited communication between neighbors that guarantees collision-avoidance and overcomes the computational challenges of the centralized method at the cost of suboptimal solutions. We demonstrate the performance of the algorithm as the number of robots is increased to tens of robots and the resulting increase in communication across neighbors required for safe execution.

1 Introduction

We consider a system with N permutation invariant robots seeking M desired goal locations in an n-dimensional Euclidean space where $N \geq M$. This paper seeks to find a computationally tractable algorithm to plan collision-free agent trajectories such that each goal location is occupied by a robot at a desired termination time. The agents' homogeneity adds an additional degree of freedom to the trajectory planning problem as compared to a system with assigned goal locations. Some applications of these homogeneous robotic systems include object manipulation [3, 10], localization [4], and satellite

Matthew Turpin, Nathan Michael, and Vijay Kumar
GRASP Laboratory, University of Pennsylvania, Philadelphia, PA
e-mail: {mturpin,nmichael,kumar}@seas.upenn.edu

formation control [1]. In all these applications finding safe trajectories is critical. Indeed, for our work on aerial robots operating in close proximity [14], finding collision-free trajectories is very important since even near misses can result in aerodynamic interactions which may cause catastrophic collisions.

Since we require each goal location to be occupied by a single agent at the final time we must explicitly assign goals to robots. Fortunately, the assignment problem occurs naturally in a number of disciplines including distributed computing, operations research, as well as the problems already mentioned and there has been significant study into solutions of the assignment problem. How the cost function of this assignment is formulated determines whether it is a linear assignment or quadratic assignment problem.

For $N = M$, each possible assignment of robots to goals can be represented by an $N \times N$ permutation matrix. The space of all possible assignments for the N agent, N goal single agent to single goal is isomorphic to S^N, or the group of all permutations of every agent. To completely enumerate all possibilities requires $N!$ operations.

In general, the process of matching goals to robots is a linear assignment problem if the total cost to be minimized is the sum of individual transition costs. The usual assignment strategy for multi-robot systems is solving the linear assignment problem minimizing the sum of distance traveled [6, 13] and will be considered in detail in Sect. 3.1. The well-known Hungarian Algorithm [8] can solve the linear assignment problem in polynomial time.

Others have relaxed the linear assignment problem to find near optimal solutions. [12] uses a heuristic to minimize the sum of Euclidean distance traveled for a very simplified Euclidean linear assignment problem that grows in $\mathcal{O}(N^{\frac{5}{6}})$. [9] presents a hybrid method for the Euclidean assignment algorithm for very large numbers of robots with both a global and local approach.There has also been work in potential field algorithms [16] to solve the linear assignment problem. Other potential field methods for controlling groups of robots to goal destinations [11] or goal sets [2] have been developed. In general, these gradient descent approaches lead to unpredictable trajectories, may take very a long time to converge, and some do not guarantee all goals are satisfied. In general, decentralized auction-based algorithms can be more expensive than a centralized solution using the Hungarian algorithm when taking into consideration the cost of communications and may require a centralized auctioneer.

The quadratic assignment problem extends the notion of the linear assignment problem by adding the notion of a flow to each assignment. Exact solutions to the quadratic assignment problems are NP-hard but there are numerous suboptimal assignment algorithms for the quadratic assignment problem using Tabu search, simulated annealing, genetic algorithms, iterated locally greedy search, and many more [5].

Solutions to this problem are especially difficult when collisions have to be avoided. A centralized algorithm to create collision-free paths to solve the assignment problem for 2-dimensional robots is reduced to the numerical solution of roots of a complex polynomial in [7].

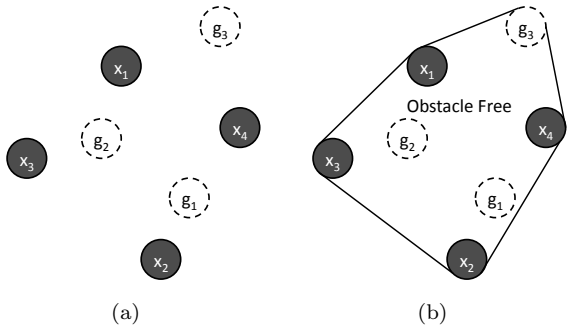


Fig. 1 For the locations of goals and agents in $\mathbb{R}^2$ in Fig. 1(a), the necessary obstacle free area $\mathcal{K}$ is designated by the closed area in Fig. 1(b)

Graph search techniques like A* or D* that are extremely powerful for finding collision-free paths do not scale well as the dimensionality of the configuration space grows with the number of robots. The M* algorithm [15] circumvents this difficulty by only resorting to searches in the joint space when pairs of robots are in close proximity. Our approach is similar in spirit to this work. However, we consider the continuous planning problem of interchangeable agents, and our decentralized algorithm functions online.

2 Preliminaries

We define the sets $\mathcal{N} = \{1, 2, \ldots, N\}$ and $\mathcal{M} = \{1, 2, \ldots, M\}$. The location of the i^{th} robot is having radius R is specified by $\mathbf{x}_i \in \mathbb{R}^n$:

$$\mathbf{x}_i(t) = \big[x_1(t), x_2(t), \ldots, x_n(t)\big]^{\mathrm{T}}, \ \forall i \in \mathcal{N}$$

Similarly, the i^{th} goal location is specified by $\mathbf{g}_i \in \mathbb{R}^n$:

$$\mathbf{g}_i = \big[g_1, g_2, \ldots, g_n\big]^{\mathrm{T}}, \ \forall i \in \mathcal{M}$$

The minimum convex operating space that must be obstacle free to solve this problem with straight line paths is defined by $\mathcal{K} \subset \mathbb{R}^n$, the Minkowski sum of the convex hull of initial locations and goal locations with ball of radius R:

$$\mathcal{K} \equiv \mathrm{conv}(\{\mathbf{x}_i(t_0) \big| i \in \mathcal{N}\} \cup \{\mathbf{g}_j \big| j \in \mathcal{M}\}) \oplus \mathcal{B}_R \tag{1}$$

See Fig. 1 for a 2-dimensional pictorial example of $\mathcal{K}$.

Similar to a permutation matrix, we define a binary relation to assign goals to agents.

$$\phi : \mathcal{M} \to \mathcal{N}$$

Accordingly we let ϕ be a $N \times M$ *assignment matrix* so that $\phi_{i,j} = 1$ if and only if agent i is assigned to goal location j. Note that ϕ has the following properties:

$$\begin{aligned} \phi_{i,j} &\in \{0,1\} \quad \forall i \in \mathcal{N}, j \in \mathcal{M} \\ \mathbf{1}_N{}^{\mathrm{T}}\phi &= \mathbf{1}_M{}^{\mathrm{T}} \end{aligned} \tag{2}$$

From this we know:

$$\phi^{\mathrm{T}}\phi = I_M$$

where I_M is the $M \times M$ identity matrix. We will use the expanded assignment matrix $\Phi \equiv \phi \otimes I_n$ where $\otimes$ signifies the Kronecker product.

We then define the Nn-dimensional state vector, $\mathbf{X} \in \mathbb{R}^{Nn}$:

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

where $\mathbf{x}_i(t) \in \mathcal{K}\ \forall i \in \mathcal{N}$. We similarly define the stacked 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 distinguish between paths and trajectories where paths are time-independent curves connecting current and final locations. This is in contrast to trajectories, which are time-parameterized planned locations of the agents. The goal of this paper is to find Nn-dimensional trajectories:

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

where t_0 and t_f are the initial and final times respectively.

The initial position $\mathbf{x}_i(t_0)\ \forall i \in \mathcal{N}$ and the goal locations $\mathbf{g}_j = \sum_{i=1}^{N} \phi_{i,j}\mathbf{x}_i(t_f)$ have been specified so the boundary conditions are:

$$\begin{aligned} \gamma(t_0) &= \mathbf{X}(t_0) \\ \Phi^{\mathrm{T}}\gamma(t_f) &= \mathbf{G} \end{aligned} \tag{3}$$

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

$$\delta(t) = \underset{i,j \in \mathcal{N}, i \neq j}{\text{minimize}} \quad ||\mathbf{x}_i(t) - \mathbf{x}_j(t)|| - 2R$$

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

$$\delta(t) > 0 \quad t \in [t_0, t_f] \tag{4}$$

Problem Definition

The concurrent trajectory planning and goal assignment problem involves finding a trajectory $\gamma^\star(t)$ that minimizes a cost functional:

$$\begin{aligned} \gamma^\star(t) = \arg\min_{\gamma(t)} \quad & \int_{t_0}^{t_f} L(\gamma, \dot{\gamma}, t)dt \\ \text{subject to} \quad & (2), (3), (4) \end{aligned} \tag{5}$$

Assumptions

It is useful to reiterate the assumptions (A1-A3) we have made thus far and introduce four new ones (A4-A7):

(A1) All agents are homogeneous and interchangeable with no preference of goal destination.
(A2) Each agent is a set of points confined to $\mathcal{B}_R$, a ball with radius R.
(A3) There are no obstacles in $\mathcal{K}$ as defined in (1).
(A4) All agents are kinematic ($\dot{x} = u$) with no actuation error and a perfect estimate of the state.
(A5) The initial and goal locations are spaced Δ apart. In other words, $||x_i(t_0) - x_j(t_0)|| > \Delta$, and $||g_i - g_j|| > \Delta$, $\forall\ i \neq j \in \{1, \ldots, N\}$ where Δ will be defined later. Clearly $\Delta > 2R$ to ensure collision avoidance.
(A6) All robots are capable of exchanging information about their current state and their assigned goals to other robots closer than distance $h > \Delta$. Robots with non-negligible communications time should ensure $h >> \Delta$.
(A7) Each goal location is initially assigned to exactly one robot.

Assumptions (A1-A5) are required for the centralized algorithm that is discussed in the next section. (A6-A7) deal with the decentralized case presented in Sect. 4.

3 Centralized Algorithm

We propose two different centralized cost functions for (5) which both seek to simultaneously find an assignment matrix ϕ and collision free trajectories $\gamma(t)$ for all agents in the system such that each goal state is occupied by an agent at $t = t_f$. The centralized optimization only needs to be computed initially at $t = t_0$ as the solution for ϕ will not change over the interval $[t_0, t_f]$.

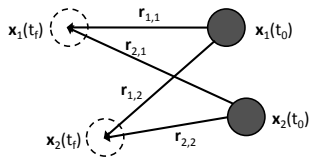


Fig. 2 For agents in n-dimensional Euclidean space ($n > 2$), if the paths collide as is the case for straight lines along $\mathbf{r}_{1,2}$ and $\mathbf{r}_{2,1}$ do, we can see that using the triangle inequality, switching assignments will always lead to collision free paths.

3.1 The Minimum Sum of Distances Trajectory

The first cost function we analyze is the current standard practice for the location assignment problem of minimizing the sum of distances traveled by all agents:

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

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:

$$\begin{aligned} \underset{\phi}{\text{minimize}} \quad & \sum_{j=1}^{M} \left|\left| \sum_{i=1}^{N} \phi_{i,j} \mathbf{x}_i(t_0) - \mathbf{g}_j \right|\right| \\ \text{subject to} \quad & (2), (3) \end{aligned} \tag{6}$$

This is the well-known transportation assignment problem. We show below in Theorem 1 that the assignment from this optimization guarantees paths that never intersect in any n-dimensional Euclidean space ($n > 2$).

Theorem 1. *The optimal assignment ϕ using the minimum sum of distance optimization in* (6) *results in paths that do not intersect except for the special case when a pair of agents have start and goal locations in a one dimensional space.*

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 we can reduce these two paths 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 (6). Therefore, the minimum assignment found in (6) will never result in intersecting paths.

Unfortunately, we can show with a simple example that agents with finite extent using the assignment from (6) are not guaranteed collision free

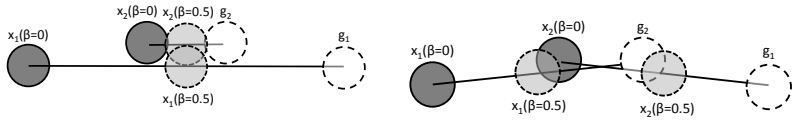


Fig. 3 For the example with two agents in Fig. 3(a) we can see that the minimum sum of distances paths (calculated by (6)) never intersect. However, having intersection-free paths does not guarantee collision-free trajectories for agents with finite size. In this case, merely switching goal assignments as shown in Fig. 3(b) does ensure collision-free trajectories.

trajectories in Fig. 3, rendering this optimization useless for real robots with physical extent.

3.2 The Minimum Velocity Trajectory

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

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

which is equivalent to:

$$\begin{aligned} \underset{\phi,\gamma(t)}{\text{minimize}} \quad & \int_{t_0}^{t_f} \dot{X}(t)^{\mathrm{T}} \dot{X}(t) dt \\ \text{subject to} \quad & (2), (3), (4) \end{aligned} \tag{7}$$

The paths returned from (7) will be identical to minimizing the sum of distance traveled squared.

If it is unclear how the optimization in Sect. 3.2 differs from Sect. 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 crossing another is unnecessary.

If we temporarily ignore the clearance requirements in (4), (7) returns an assignment matrix ϕ and trajectories:

$$\gamma^\star(t) = \left(1 - \frac{t-t_0}{t_f-t_0}\right)\mathbf{X}(t_0) + \left(\frac{t-t_0}{t_f-t_0}\right)\left(\Phi\mathbf{G} + (I_{Nn} - \Phi\Phi^{\mathrm{T}})\mathbf{X}(t_0)\right) \quad (8)$$

It is clear that at $t = t_0$, (8) satisfies $\gamma^\star(t_0) = \mathbf{X}(t_0)$. To verify the final boundary conditions, we can premultiply (8) by Φ^{T}:

$$\Phi^{\mathrm{T}}\gamma^\star(t) = \left(1 - \frac{t-t_0}{t_f-t_0}\right)\Phi^{\mathrm{T}}\mathbf{X}(t_c) + \frac{t-t_0}{t_f-t_0}\mathbf{G}$$

to verify $\Phi^{\mathrm{T}}\gamma^\star(t_f) = \mathbf{G}$.

Now that we know that the trajectory will take the form in (8), we can formulate (7) as a linear assignment problem and can use an optimal assignment solving algorithm such as the Hungarian Algorithm or a suboptimal algorithm. For hundreds of robots, as shown in [5], the Hungarian Algorithm can be used to compute the exact solution of the assignment problem in under a minute on a modest computer. We now demonstrate that if we take $\Delta > 2\sqrt{2}R$, the solution to (8) will be collision free.

For notational convenience, we define:

$$\begin{aligned} \mathbf{r}_{i,j} &\equiv \mathbf{x}_j(t_f) - \mathbf{x}_i(t_0) & \mathbf{u}_{i,j} &\equiv \mathbf{x}_j(t_0) - \mathbf{x}_i(t_0) \\ \mathbf{w}_{i,j} &\equiv \mathbf{x}_j(t_f) - \mathbf{x}_i(t_f) & \beta(t) &\equiv \frac{t-t_0}{t_f-t_0} \in [0,1] \end{aligned}$$

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

Lemma 1. *The optimal solutions to* (7) *satisfy:*

$$\mathbf{w}_{i,j}{}^{T}\mathbf{u}_{i,j} \geq 0 \ \forall i,j \in \mathcal{N} \quad (9)$$

Proof. Since we globally minimized the sum of integrated velocity squared in (7), we know that switching goal states of agent i with agent j will not decrease the sum of distance squared, or:

$$||\mathbf{r}_{i,i}||^2 + ||\mathbf{r}_{j,j}||^2 \leq ||\mathbf{r}_{i,j}||^2 + ||\mathbf{r}_{j,i}||^2 \ \forall i,j \in \mathcal{N} \quad (10)$$

We then substitute:

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

into (10) and simplify:

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

or

$$\mathbf{w}_{i,j}{}^{\mathrm{T}}\mathbf{u}_{i,j} \geq 0 \ \forall i,j \in \mathcal{N} \quad (11)$$

Theorem 2. *If $\Delta > 2\sqrt{2}R$, trajectories in* (8) *will satisfy* (4) *and be collision free.*

Proof. The analytic solution for when agents i and j ($i \neq j$) will be closest is:

$$\beta^{\star}_{i,j} = \frac{{\mathbf{u}_{i,j}}^{\mathrm{T}}(\mathbf{u}_{i,j} - \mathbf{w}_{i,j})}{(\mathbf{u}_{i,j} - \mathbf{w}_{i,j})^{\mathrm{T}}(\mathbf{u}_{i,j} - \mathbf{w}_{i,j})}$$

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 (A5). If however, $\beta^{\star}_{i,j} \in [0, 1]$, the minimum distance agent i will be from agent j is:

$$||\mathbf{x}_i - \mathbf{x}_j||_{\min} = \sqrt{{\mathbf{u}_{i,j}}^{\mathrm{T}}\mathbf{u}_{i,j} - \frac{({\mathbf{u}_{i,j}}^{\mathrm{T}}(\mathbf{u}_{i,j} - \mathbf{w}_{i,j}))^2}{(\mathbf{u}_{i,j} - \mathbf{w}_{i,j})^{\mathrm{T}}(\mathbf{u}_{i,j} - \mathbf{w}_{i,j})}} \tag{12}$$

As shown in Lemma 1, the assignment returned from (7) guarantees ${\mathbf{u}_{i,j}}^{\mathrm{T}}\mathbf{w}_{i,j} > 0$ for all pairs of robots. Using this fact, we can see from (12) that the minimum distance possible between two robots will occur when ${\mathbf{u}_{i,j}}^{\mathrm{T}}\mathbf{w}_{i,j} = 0$. Further if $||\mathbf{u}_{i,j}||$ and $||\mathbf{w}_{i,j}||$ are each allowed to be as small as possible which we called Δ in (A5), the minimum distance encountered using the assignment returned from (7) is:

$$||\mathbf{x}_i - \mathbf{x}_j||_{\min} = \frac{\Delta}{\sqrt{2}}$$

Since $\Delta > 2\sqrt{2}R$, we can rearrange and substitute to find the smallest distance between agents:

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

Thus the robots can never be in the ball of another robot and collision avoidance is guaranteed.

It should be noted that minimizing the sum of distance traveled squared arrives at the collision free assignment in Fig. 3(b).

4 Decentralized Algorithm

In this section, we exploit our knowledge of the centralized solution proposed in Sect. 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 (11) and is based on reassignment of goal locations to arrive at a locally-optimal solution.

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 (7). Before we present the decentralized algorithm, Algorithm 1, we first introduce some new notation.

As there is no centralized bookkeeper, ϕ is no longer known to any one robot. Therefore we define $\mathbf{f}_i$ as the goal currently assigned to robot i, if it exists, such that:

$$\sum_{i=1}^{N} \phi_{i,j}\mathbf{f}_i = \mathbf{g}_j$$

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:

$$\mathcal{C}_i(t) = \{j \mid \|\mathbf{x}_j(t) - \mathbf{x}_i(t)\| \leq h\} \subset \mathcal{N} \tag{13}$$

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 the decentralized algorithm (Algorithm 1), the i^{th} robot locally minimizes the contribution to (7) from every pair of i and j that satisfy (13):

$$\underset{\mathbf{f}_i, \mathbf{f}_j, \gamma(t)}{\text{minimize}} \quad \int_{t_c}^{t_f} \dot{x}_i(t)^{\mathrm{T}} \dot{x}_i(t) dt + \int_{t_c}^{t_f} \dot{x}_j(t)^{\mathrm{T}} \dot{x}_j(t) dt \tag{14}$$

The trajectory for the remaining time $[t_c, t_f]$ is computed in a similar fashion to the centralized version (8):

$$\begin{aligned} \mathbf{x}_i(t) &= \left(1 - \frac{t - t_c}{t_f - t_c}\right)\mathbf{x}_i(t_c) + \left(\frac{t - t_c}{t_f - t_c}\right)\mathbf{f}_i \quad && \text{if } \mathbf{f}_i \text{ exists} \\ \mathbf{x}_i(t) &= \mathbf{x}_i(t_c) && \text{otherwise} \end{aligned} \tag{15}$$

We do not actively control robots without assigned goal states and as such are unconcerned with their final locations, only that the vehicles do not collide.

Note that Algorithm 1 ensures only new information is transmitted to the robots in $\mathcal{C}_i(t)$ without making unnecessary communications. Using similar reasoning to Lemma 1, we will show in Lemma 2 that Algorithm 1 converges to a locally optimal solution to (7).

Lemma 2. *A change in goal locations between any pair of robots i and j in Algorithm 1 results in a decrease of the sum of distance remaining squared.*

Algorithm 1 Goal Assignment of Agent i

compute trajectory using (15)
initialize $\mathcal{U}_i = \mathcal{C}_i(t_0)$
while $t < t_f$ **do**
  $t_c \leftarrow t$
  for $j \in \mathcal{U}_i$ **do**
    request $\mathbf{x}_j(t_c)$ and $\mathbf{f}_j$ from agent j
    if $\mathbf{f}_i$ exists AND $\mathbf{f}_j$ does not exist AND $||\mathbf{x}_i - \mathbf{f}_i|| > ||\mathbf{x}_j - \mathbf{f}_i||$ **then**
      reassign $\mathbf{f}_i$ to $\mathbf{f}_j$
      set $\mathcal{U}_i = \mathcal{C}_i(t_c)$ and recompute trajectory using (15)
    else if $\mathbf{f}_i$ does not exist AND $\mathbf{f}_j$ exists AND $||\mathbf{x}_i - \mathbf{f}_j|| < ||\mathbf{x}_j - \mathbf{f}_j||$ **then**
      reassign $\mathbf{f}_j$ to $\mathbf{f}_i$
      set $\mathcal{U}_i = \mathcal{C}_i(t_c)$ and recompute trajectory using (15)
    else if both $\mathbf{f}_i$ and $\mathbf{f}_j$ exist AND ${\mathbf{u}_{i,j}}^{\mathrm{T}}\mathbf{w}_{i,j} < 0$ **then**
      exchange goal states $\mathbf{f}_i$ and $\mathbf{f}_j$
      set $\mathcal{U}_i = \mathcal{C}_i(t_c)$ and recompute trajectory using (15)
    remove j from $\mathcal{U}_i$
  if agent j requests a change of $\mathbf{f}_i$ **then**
    update $\mathbf{f}_i$
    set $\mathcal{U}_i = \mathcal{C}_i(t_c)$ and recompute trajectory using (15)
    remove j from $\mathcal{U}_i$
  if agent j is added to $\mathcal{C}_i(t_c)$ **then**
    add agent j to $\mathcal{U}_i$
  if agent j is removed from $\mathcal{C}_i(t_c)$ **then**
    ensure $j \notin \mathcal{U}_i$

Proof. We show that when a pair of robots exchange goals locations, the sum of distance 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.

Case 1: Both robots have assigned goals.
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 1, 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.

Case 2: Either $\mathbf{f}_i$ or $\mathbf{f}_j$ don't exist. After reassignment of a goal from agent j to agent i:

$$ ||\mathbf{x}_i(t_c) - \mathbf{f}_i|| < ||\mathbf{x}_j(t_c) - \mathbf{f}_i|| \quad \rightarrow \quad ||\mathbf{x}_i(t_c) - \mathbf{f}_i||^2 < ||\mathbf{x}_j(t_c) - \mathbf{f}_i||^2 $$

After reassignment of a goal from agent i to agent j:

$$||\mathbf{x}_j(t_c) - \mathbf{f}_j|| < ||\mathbf{x}_i(t_c) - \mathbf{f}_j|| \quad \rightarrow \quad ||\mathbf{x}_j(t_c) - \mathbf{f}_j||^2 < ||\mathbf{x}_i(t_c) - \mathbf{f}_j||^2$$

Thereby showing that the sum of distance remaining squared has decreased from the original assignment.

Theorem 3. *Algorithm 1 results in each goal being occupied by one robot without any collisions.*

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

$$V = \sum_{i=1}^{M} ||\mathbf{x}_i - \mathbf{f}_i||^2 \tag{16}$$

For an arbitrarily small constant, ϵ, we can use Lemma 2 and the trajectories defined in (15) to see that V is strictly decreasing:

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

Further, by (15), the final value of the cost-to-go function will be zero:

$$V(t_f) = 0.$$

From Theorem 2, we know all trajectories will be free of collisions.

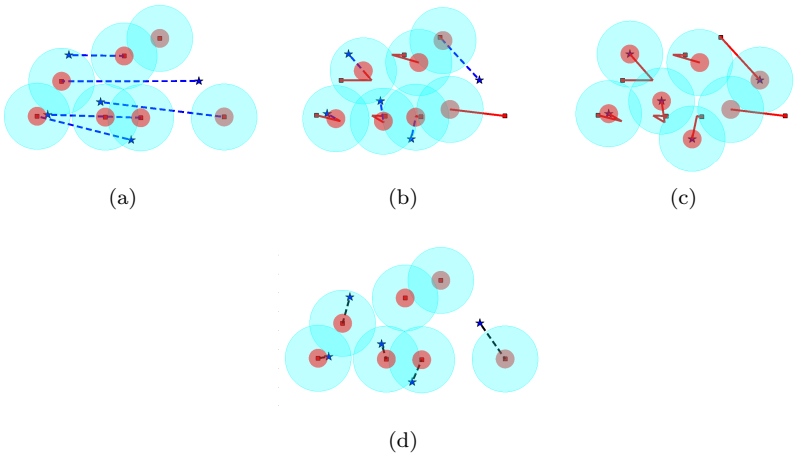
It should be noted that for the decentralized algorithm, there exist initial conditions which, resulting from the meeting of disconnected groups of robots, could potentially result in a collision. However, there were no instances of this pathological failure mode in millions of randomly generated simulations. Instead, artificial construction was required to experience the failure. Due to the extremely low likelihood of occurrence in a real world system, we defer detailed analysis of this failure case to a future work which will present our solution to this failure modality.

5 Simulation Results

We simulate our algorithm on a large variety of boundary conditions to study its performance. For each trial, we randomly generate starting and goal locations that satisfy (A5).

The first simulation is shown in Fig. 4 and demonstrates the functioning of the on-line decentralized algorithm for $N = 7$ robots $M = 5$ goal destinations in a 2-dimensional space with a small communications range $h = 1.2\Delta$.

The second result shown in Figure 5 is to verify that the energy function defined in (16) is indeed strictly decrescent by using a simulation with $N =$



(a) (b) (c)

(d)

Fig. 4 Figs. 4(a)-4(c) show snapshots of a representative 2-dimensional simulation with $N = 7$, $M = 5$ and limited communications range. In Figs. 4(a)-4(c), dotted lines represent expected trajectory, solid lines represent the path followed, stars are goal locations, boxes are initial locations, and communications range h is denoted by the translucent area. For comparison, Fig. 4(d) shows the optimal solution to the centralized problem with dotted lines.

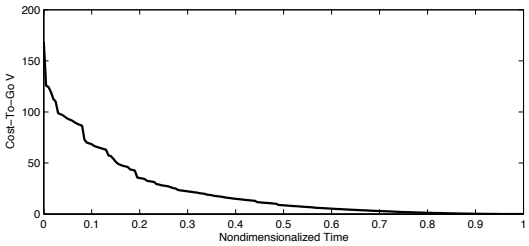
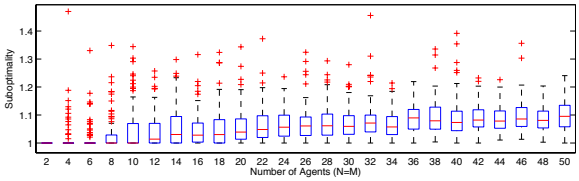


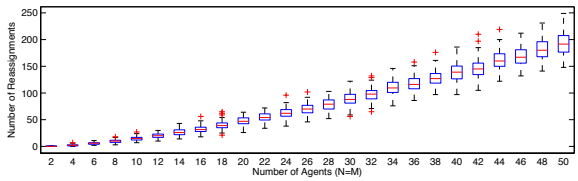
Fig. 5 Visualization of decay of the cost-to-go function V in (16) for $N = 100, M = 50$, and $n = 3$. The instantaneous drops are a result of reassignments and the continuous decay is a result of trajectory tracking.

$M = 100$ in three dimensions with $h = 1.5\Delta$. It can be easily seen that $\dot{V} < 0$ and $V(t_f) = 0$.

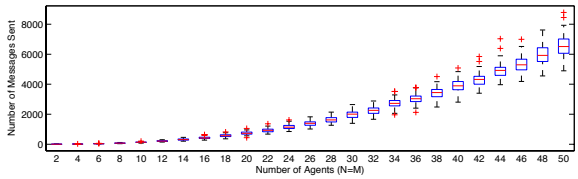
In the third set of simulation runs we explore the scaling of the decentralized algorithm with the number of robots. In Fig. 6, we vary $N = M$ from 2 to 50 and present the result for 100 trials with $\frac{h}{\Delta} >> 1$ in three dimensions ($n = 3$). We compare the sum of distance traveled squared to the optimal value returned from (7) using the Hungarian Algorithm. We also note that the number of reassignments increases approximately linearly in N. We can see that the total number of communications grows almost exactly as N^2.



(a)



(b)



(c)

Fig. 6 Box plots for the properties of the decentralized algorithm for 100 simulated random configurations and assignments for N agents and $M = N$ goals. Figure 6(a) shows the distance traveled squared divided by the optimal value returned from the centralized solution in Sect. 3.2. The "+" marks designate statistical outliers. These plots are for $n = 3$.

The final set of simulations can be seen in Fig. 7 where varied communications ranges are used 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, we see that minimum clearance δ between robots decreases with smaller communications ranges as one might expect, but never violates the clearance requirement in (4).

6 Conclusion and Future Work

In this paper, we have addressed the problem of concurrently assigning goals and planning trajectories to the goals for a team of N robots with $M \leq N$ destinations. The centralized assignment and planning problem is well known

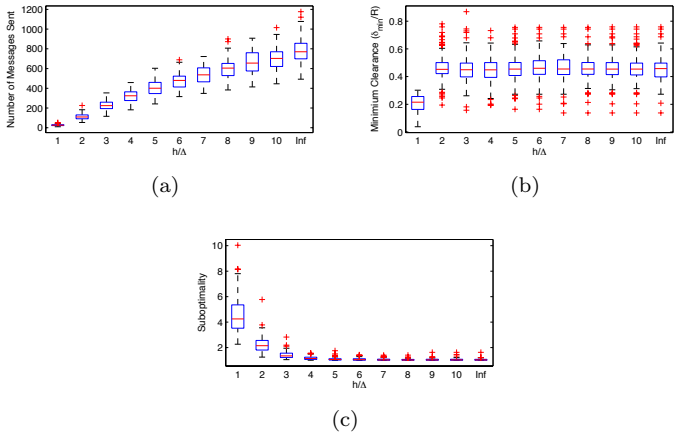


Fig. 7 Box plots for the properties of the decentralized algorithm for 100 simulated random configurations with varying communications distances with N=M=20. Figure 7(a) clearly shows that as the communications range decreases, the number of messages sent in the decentralized algorithm drastically decreases at the expense of clearance in Fig. 7(b) as well as the optimality of the solution in 7(c). The clearance requirement in (4) is never violated. "+" marks designate statistical outliers.

in the literature. However, the same problem applied to robots with finite size and requirements of collision-free trajectories introduces challenges. We first develop 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. The second contribution of the paper is a decentralized algorithm that relies on the robots exchanging information about their current state and their intended goal when they are 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. The performance of the algorithm improves with the density of the robots requiring more reassignments but with a net cost that is closer to the globally optimal cost. The computational complexity of Algorithm 1 scales very well when applied to large swarms where the number of communications is proportional to ratio of the communications radius h times the spatial density of robots. From simulations, it appears that the number of reassignments necessary scale linearly with N and the number of messages exchanged quadratically with N.

Our current work addresses incorporating the dynamics of robots and extending the cost functional L to include higher order time derivatives of

trajectories. We are also interested in adapting the decentralized algorithm to consider robot failures.

References

1. Beard, R., Lawton, J., Hadaegh, F.: A coordination architecture for spacecraft formation control. Control Syst. Technol., IEEE Transactions on **9**(6), 777–790 (2001)
2. Chaimowicz, L., Michael, N., Kumar, V.: Controlling swarms of robots using interpolated implicit functions. In: Proc. of the IEEE Int. Conf. on Robotics and Automation, pp. 2487–2492. IEEE, Barcelona (2005)
3. Das, A., Fierro, R., Kumar, V., Ostrowski, J., Spletzer, J., Taylor, C.: A vision-based formation control framework. Robotics and Automation, IEEE Transactions on **18**(5), 813–825 (2002)
4. Fox, D., Burgard, W., Kruppa, H., Thrun, S.: A probabilistic approach to collaborative multi-robot localization. Autonomous Robots **8**(3), 325–344 (2000)
5. Gerkey, B., Matarić, M.: A formal analysis and taxonomy of task allocation in multi-robot systems. The International Journal of Robotics Research **23**(9), 939–954 (2004)
6. Ji, M., Azuma, S., Egerstedt, M.: Role-assignment in multi-agent coordination. Int. Journal of Assistive Robotics and Mechatronics **7**(1), 32–40 (2006)
7. Kloder, S., Hutchinson, S.: Path planning for permutation-invariant multirobot formations. Robotics, IEEE Transactions on **22**(4), 650–665 (2006)
8. Kuhn, H.: The hungarian method for the assignment problem. Naval Research Logistics Quarterly **2**(1-2), 83–97 (1955)
9. Liu, L., Shell, D.: Multi-level partitioning and distribution of the assignment problem for large-scale multi-robot task allocation. In: In Proc. of Robotics: Science and Systems. Los Angeles, CA (2011)
10. Mataric, M., Nilsson, M., Simsarin, K.: Cooperative multi-robot box-pushing. In: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 556–561. Pittsburgh, PA (1995)
11. Molnár, P., Starke, J.: Control of distributed autonomous robotic systems using principles of pattern formation in nature and pedestrian behavior. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on **31**(3), 433–435 (2001)
12. Rendl, F.: On the euclidean assignment problem. Journal of Computational and Applied Mathematics **23**(3), 257–265 (1988)
13. Smith, S., Bullo, F.: Target assignment for robotic networks: Asymptotic performance under limited communication. In: Proc. of the American Control Conference, pp. 1155–1160. IEEE, New York (2007)
14. Turpin, M., Michael, N., Kumar, V.: Trajectory design and control for aggressive formation flight with quadrotors. In: Proc. of the Intl. Sym. on Robotics Research. Flagstaff, AZ. (2011)
15. Wagner, G., Choset, H.: M*: A complete multirobot path planning algorithm with performance bounds. In: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 3260–3267. San Francisco, CA (2011)
16. Zavlanos, M., Pappas, G.: Potential fields for maintaining connectivity of mobile networks. Robotics, IEEE Transactions on **23**(4), 812–816 (2007)