2012 American Control Conference
Fairmont Queen Elizabeth, Montréal, Canada
June 27-June 29, 2012

Feasibility Checks and Control Laws for Reconfigurations of Spacecraft Clusters

Nima Moshtagh*, Amir Ali Ahmadi†, Mehran Mesbahi§

***Abstract*— A multi-spacecraft system consists of wirelessly-connected spacecraft that share resources in Earth orbit or deep space. One of the enabling technologies of such a fractionated space architecture is cluster flight, which provides the cluster with the capability to perform a cluster scatter and regather maneuvers to rapidly evade debris-like threats or to reconfigure for scientific missions. In this work, we study the cluster reconfiguration problem. First, we propose an algorithm based on semidefinite programming to check for the feasibility of a desired configuration. Next, we provide a control law for cluster reconfiguration that requires only relative state information and the adjacency matrix of the underlying network topology. Stability analysis and simulation results are provided.**

I. Introduction

Under the concept of fractionated space systems [3], a cluster of wirelessly-interconnected modules share and utilize resources found elsewhere in the cluster. Such an architecture enhances the adaptability and survivability of space systems. Collision-safe multibody cluster flight and efficient relative navigation are enabling technologies for fractionated space architecture.

We distinguish between our notion of *cluster flight* and the more commonly discussed concept of *formation flight*. Unlike formation flight, spacecraft clusters do not generally require precise maintenance of the relative positions of the spacecraft. Thus, as long as the relative distances do not exceed the ranges supported by the cross-links and collision avoidance is ensured, the relative drift of the spacecraft (due to orbital disturbances) is perfectly acceptable [3].

The main problem studied in this work is the relative navigation and reconfiguration of cluster flying spacecraft. In such scenario, initial and final configurations are specified using the state-dependent graph associated with each configuration. Typically the desired final graph is determined based on mission objectives and sensing and communication constraints among the modules. In large clusters, the cross-link requirements might impose contradictory constraints on the network, and result in infeasible network topologies. Thus, the feasibility of any given desired configuration must be checked before it is used for cluster reconfiguration. This motivates the study of *graph realization*, where the objective is to determine whether the desired network topology corresponds to a set of feasible states.

We propose a semidefinite programming based algorithm for performing this feasibility check. The approach is to reformulate the problem as a rank minimization problem with linear and semidefinite constraints, and then apply the well-known nuclear norm relaxation for rank minimization. This is done in Section III. From a computational perspective, the attractiveness of our approach stems from the fact that semidefinite programs (SDPs) can be efficiently solved e.g. by using interior point algorithms. These algorithms have been implemented in several software packages such as SeDuMi [21]. For more background on SDPs, the interested reader is referred to [2].

Given a feasible final graph, the next stage is to generate collision-free trajectories that efficiently reconfigure the cluster. In formation control literature [15], typically the initial and final states of the formation are specified in an inertial frame, and the objective is for each agent to reach its final destination while minimizing fuel consumption and avoiding collision with other agents. In Section IV, a control law is designed that reconfigures the network of mobile agents to a desired graph using only adjacency information (relative navigation). A stability analysis of the control law is also provided.

Before presenting our main results, we formally define the problems of interest in Section II.

II. Problem Statement

The problems we consider here concern designing collision-free trajectories for a cluster of mobile agents during a reconfiguration mission. It is assumed that only relative state information can be used for control and reconfiguration.

Consider a network of n agents. The state (position) of agent i is represented as a point in the agent's configuration space X_i (e.g. $\mathbb{R}^3$). The state space of all agents, X, is defined as $X = X_1 \times X_2 \times \ldots \times X_n$. The network configuration is denoted by $\mathbf{x} = \{x_1, \ldots, x_n\} \in X$. The trajectory of agent i is represented as mapping $x_i : [0, T] \to X_i$, which evolves according to the simplified system dynamics,

$$\dot{x}_i(t) = u_i(t)$$

where u_i is the control input.

Let δ_l be the minimum safe distance between any two agents. The *collision-free* configuration space is now defined as

$$\Omega := \{\mathbf{x} \in X \mid \|x_i - x_j\| \geq \delta_l,\ \forall (i,j)\} \ .$$

Let $\mathcal{V} = \{1, \ldots, n\}$ be the set of n agents, and let $\mathcal{E}$ be the set of sensing edges defined by

$$\mathcal{E} = \{(i,j) \in \mathcal{V} \times \mathcal{V} \mid \delta_l \leq \|x_i - x_j\| \leq \delta_u\} \ , \qquad (1)$$

* Lockheed Martin, ATC, nima.moshtagh@lmco.com
† Massachusetts Institute of Technology, a_a_a@mit.edu
§ University of Washington, mesbahi@aa.washington.edu

978-1-4577-1094-0/12/$26.00 ©2012 AACC

where δ_u is the sensing range and δ_l is the collision range.[1] A state-dependent graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is characterized by the node set $\mathcal{V}$, and the edge set $\mathcal{E}$.

The adjacency matrix corresponding to $\mathcal{G}$ is defined as

$$\mathcal{A} : \Omega \to \mathcal{M} \subset \{0,1\}^{n \times n} \ ,$$

a mapping from the collision-free configuration space Ω to $\mathcal{M}$, the set of $n \times n$ symmetric matrices where each entry is either 0 or 1 and the diagonal terms equal 0.

During a reconfiguration mission, suppose the initial and the desired final configurations are specified in terms of adjacency matrices $A_{initial}$, and A_{final} respectively. The first problem is to determine whether the given final adjacency matrix A_{final} corresponds to a feasible configuration.

Problem 2.1 (Graph Realization): *Suppose graph $\mathcal{G}_o$ is specified using its adjacency matrix A_o, and two parameters δ_l and δ_u that respectively define the lower and upper bounds on the edge constraints (see (1)). A* realization *of graph $\mathcal{G}_o$ is an embedding (set of positions) $\mathbf{x} \in \Omega$ such that $\mathcal{A}(\mathbf{x}) = A_o$. Find a realization of $\mathcal{G}_o$ that is embedded in $\mathbb{R}^k$ for $k \in \{1,2,3\}$.*

This problem is is studied in Section III. After we determine that the given desired graph (adjacency matrix) has a feasible realization, we would like to find a set of collision-free trajectories for network reconfiguration. Here is a formal definition of the reconfiguration problem:

Problem 2.2 (Cluster Reconfiguration): *Given $A_{initial} = \mathcal{A}(\mathbf{x}_0)$ and A_{final}, we wish to find the trajectories $\mathbf{x}(t)$ so that $\mathcal{A}(\mathbf{x}(0)) = A_{initial}$ and $\mathcal{A}(\mathbf{x}(T)) = A_{final}$ for some $T > 0$, and $\mathbf{x}(t) \in \Omega$ for all $0 \le t \le T$.*

This problem is studied in Section IV.

III. Graph Realization Problem

Graph realization problem is studied extensively in many contexts, from molecular conformation (where one is interested in determining the spatial structure of molecules from a set of geometric constraints) to wireless sensor networks (where one is interested in determining the sensor locations from connectivity constraints) [6].

In the above examples, the distances between the pairs of nodes are given, and the problem is to find an embedding (a set of positions) that satisfies the distance constraints. More formally, the *embedding* of graph $\mathcal{G}$ in Euclidean space $\mathbb{R}^k$ (for a given dimension $k \ge 1$) is equivalent to a set of positions $x_i \in \mathbb{R}^k$ such that the Euclidean distance between the pair x_i and x_j is equal to the distance d_{ij}, i.e.

$$\|x_i - x_j\| = d_{ij} \ .$$

Euclidean embedding problem can be formulated as a global optimization problem [16]. The optimal value of the optimization

$$\min_{x_1,\ldots,x_n \in \mathbb{R}^k} \sum_{(i,j)\in\mathcal{E}} |\|x_i - x_j\| - d_{ij}| \tag{2}$$

[1]For simplicity, it is assumed that the sensing and collision ranges are the same for all agents. The extension to the case where they are different for different pairs of agents is straight forward.

is zero, if and only if $x_1, \ldots, x_n$ are the true node locations. However, Saxe showed [19] that for a fixed dimension k, the Euclidean embedding problem is NP-hard.

There are a number of relaxations of the optimization problem (2) including those based on semidefinite programming (SDP) [6], [11], [1], second order cone programming (SOCP) [22], [7], and more recently sum-of-squares (SOS) methods [16].

However, Schoenberg [20] showed in 1935, that existence of an embedding, given a set of distances, is equivalent to existence of a semidefinite matrix in terms of the positions. Such an exact characterization has many implications that we will use in the work.

The following theorem statement is adopted from [17].

Theorem 3.1: *The distances d_{ij} can be embedded in a Euclidean space if and only if the $n \times n$ matrix*

$$D = \begin{bmatrix} 0 & d_{12}^2 & d_{13}^3 & \ldots & d_{1n}^2 \\ d_{21}^2 & 0 & d_{23}^2 & \ldots & d_{2n}^n \\ \vdots & \vdots & \vdots & \ldots & \vdots \\ d_{n1}^2 & d_{n2}^2 & d_{n3}^2 & \ldots & 0 \end{bmatrix} \tag{3}$$

is negative semidefinite on the subspace orthogonal to the vector $\mathbf{1}_n = [1, \ldots, 1]^T$.

The Euclidean distance matrix (EDM) (3) is related to the position matrix $X = [x_1 \ldots x_n] \in \mathbb{R}^{k \times n}$. Since

$$d_{ij}^2 = \|x_i - x_j\|^2 = \langle x_i, x_i \rangle + \langle x_j, x_j \rangle - 2\langle x_i, x_j \rangle \ ,$$

we have

$$D = \text{diag}(Q)\mathbf{1}_n^T + \mathbf{1}_n \text{diag}(Q)^T - 2Q = f(Q) \ , \tag{4}$$

where Q is the matrix of inner products (sometimes called the Gram matrix):

$$Q = X^T X = \begin{bmatrix} \langle x_1, x_1 \rangle & \ldots & \langle x_1, x_n \rangle \\ \vdots & \ddots & \vdots \\ \langle x_n, x_1 \rangle & \ldots & \langle x_n, x_n \rangle \end{bmatrix} \ . \tag{5}$$

Q is positive semidefinite by construction, and its rank is equal to k, the dimension of the embedding space. Therefore, we are interested in finding the position vectors $x_1, \ldots, x_n \in \mathbb{R}^k$ with the smallest k, such that the distances between the points satisfy the distance constraints. This problem is called the *low-dimensional embedding problem*. Low-dimensional embedding problem can be formulated as the following optimization problem:

$$\min_{Q} \quad \text{rank}(Q) \tag{6}$$

$$\text{subject to} \quad f(Q) = D \tag{7}$$

$$Q \succcurlyeq 0 \tag{8}$$

where $f(Q)$ is defined in (4). The constraints are linear matrix inequalities (LMIs) which form a convex feasible set, but the objective function is not a convex function. Although matrix rank minimization problem (RMP) is NP-hard, several relaxations have been proposed that solve RMP approximately [13], [8], [9].

In particular, it is shown by Recht *et. al* [18] that minimizing the nuclear norm $\|\cdot\|_*$ of a matrix can provide a good approximation for minimizing its rank. Note that $rank(Q)$ is equal to the number of nonzero singular values of Q, or equivalently the cardinality of the vector of singular values $s = [\sigma_1, \ldots, \sigma_k]^T$. One can approximate the cardinality of s with its l_1-norm, $\|s\|_1$, which is the sum of singular values of matrix Q, and equals the nuclear norm of Q. Since Q is symmetric, its eigenvalues are the same as its singular values, and since Q is positive semidefinite, its eigenvalues are non-negative. Therefore,

$$\|Q\|_* = \|s\|_1 = \sum_{i=1}^{k} \sigma_i(Q) = \sum_{i=1}^{k} \lambda_i(Q) = \text{trace}(Q) \ . \quad (9)$$

Just as l_1-minimization is the "tightest" convex relaxation of the NP-hard cardinality-minimization problem, nuclear-norm minimization is the "tightest" convex relaxation of the NP-hard rank minimization problem [4], [8]. In [5], it is proven that nuclear-norm minimization succeeds nearly as soon as recovery is possible by any method whatsoever.

Thus, instead of solving optimization (6), we solve the approximate problem:

$$\begin{aligned} \min_{Q} \quad & \text{trace}(Q) \\ \text{subject to} \quad & (7), (8) \end{aligned} \quad (10)$$

which is a convex optimization problem, in fact an SDP.

In order to extract position vectors $x_i \in \mathbb{R}^k$ from the solution matrix Q, one can use matrix factorizations methods such as general eigenvalue decomposition or Cholesky factorization.

A. Constrained Graph Realization

Now we can present a solution to *Graph Realization* Problem presented in Section II. Suppose graph $\mathcal{G}$ is given in the form of its adjacency matrix A. The objective is to find an embedding $x_1, \ldots, x_n$ such that

$$\delta_l \leq \|x_i - x_j\| \leq \delta_u \ . \quad (11)$$

In the definition of the Euclidean embedding problem, the distance constraint $\|x_i - x_j\| = d_{ij}$ can be relaxed by the interval constraint (11). The upper-bound constraint $\|x_i - x_j\| \leq \delta_u$ is a convex constraint, however, the lower-bound constraint $\delta_l \leq \|x_i - x_j\|$ is not convex.

Working with the square of distances d_{ij}^2 allows us to write constraint (11) as a linear constraint on the (i,j)-th element of EDM D:

$$\Delta_l \leq D_{ij} \leq \Delta_u \ , \forall (i,j) \ s.t. \ A_{ij} = 1 \quad (12)$$

where $\Delta_l = \delta_l^2$ and $\Delta_u = \delta_u^2$. Similarly, when $A_{ij} = 0$, one can write the following constraint:

$$\Delta_u \leq D_{ij} \ , \qquad \forall (i,j) \ s.t. \ A_{ij} = 0, \ i \neq j \ . \quad (13)$$

Thus, we formulate the "graph realization with interval constraints" as the following optimization:

$$\begin{aligned} \min_{Q,D} \quad & \text{trace}(Q) \\ \text{subject to} \quad & (7), (8), (12), (13) \end{aligned} \quad (14)$$

Optimization problem (14) is a convex problem since constraints (12) and (13) are linear constraints on decision variables D_{ij}. A factorization of solution Q is then computed to extract the positions $x_1, \ldots, x_n$.

Remark 3.2: The solution to the relaxed optimization problem (14) is not always a realization with an embedding in the smallest dimension due to approximating $rank(Q)$ with $trace(Q)$ in the optimization. In other words, when for an adjacency matrix A the solution to (14) is embedded in $\mathbb{R}^k$, this does not necessarily imply that k is the smallest possible dimension for which A is realizable.

IV. Cluster Reconfiguration

In this Section we present a solution to Problem 2.2. Before presenting our solution, let us define the weighted adjacency matrix of a graph. The weighted adjacency matrix of a graph is given by

$$\mathcal{A}_w : \Omega \to \mathcal{M}_{[0\ 1]}$$

a mapping from Ω (the collision-free configuration space) to $M_{[0\ 1]}$ (the set of $n \times n$ symmetric matrices with each entry within the interval $[0\ 1]$), and it is defined by

$$a_{ij} = [A_w]_{ij} = \begin{cases} \sigma_\omega(\delta_u - d_{ij}) & i \neq j \\ 0 & i = j \end{cases} \quad (15)$$

where $\sigma_\omega : \mathbb{R} \to [0\ 1]$ is the sigmoid function

$$\sigma_\omega(z) = \frac{1}{1 + e^{-\omega z}} \ ,$$

as illustrated in Figure 1.

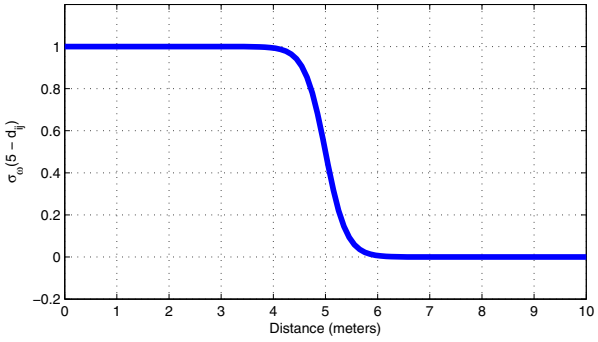


Fig. 1. Sigmoid function $\sigma_\omega(\delta - d)$ with $\omega = 5$, and $\delta = 5$.

A. Adjacency-Based Reconfiguration

Now we present our result on adjacency-based reconfiguration of spacecraft.

Proposition 4.1: *Consider a system of n agents with dynamics $\dot{x}_i = u_i \in \mathbb{R}^3$. Let $A_d \in \mathcal{M}$ denote a given adjacency matrix. If the state-dependent graph $\mathcal{G}$ remains connected, then by applying the control law*

$$u_i = 2\kappa\omega \sum_{j=1}^{n} a_{ij}\big(1 - a_{ij}\big)\big([A_d]_{ij} - a_{ij}\big)(x_j - x_i)/d_{ij} \quad (16)$$

where $\kappa > 0$, the agents converge to one of the following configurations:

a) $\{\mathbf{x} \mid A(\mathbf{x}) = A_d\}$;
b) $\{\mathbf{x} \mid x_i = x_j,\ \forall i, j = 1, \ldots, n\}$,
where $\mathbf{x} \in \mathbb{R}^{3n}$ *denotes the stack of all position vectors.*

Proof: Consider the following Lyapunov function as the distance between the given desired adjacency A_d and the weighted adjacency $A_w(\mathbf{x}(t))$ corresponding to configuration $\mathbf{x}(t)$:

$$V(\mathbf{x}) = \|A_d - \mathcal{A}_w(\mathbf{x})\|_F^2 = \sum_{i=1}^{n}\sum_{j=1}^{n}([A_d]_{ij} - a_{ij})^2 \ . \quad (17)$$

Note that $V(\mathbf{x}) \geq 0$ for all $\mathbf{x}$, and $V(\mathbf{x}) = 0$ if and only if $\mathcal{A}_w(\mathbf{x}) = A_d$. We define the control input of agent i as

$$\begin{aligned} u_i &= -\kappa\nabla_{x_i}V(\mathbf{x}) = -\kappa\sum_{j=1}^{n}\frac{\partial V(\mathbf{x})}{\partial d_{ij}}\frac{\partial d_{ij}}{\partial x_i} \\ &= 2\kappa\omega\sum_{j=1}^{n}a_{ij}(1-a_{ij})\big([A_d]_{ij} - a_{ij}\big)\mathbf{r}_{ij} \quad (18) \end{aligned}$$

where a_{ij} is given by (15), and $\mathbf{r}_{ij} = (x_j - x_i)/d_{ij}$ is the unit-norm bearing vector for the pair (i, j).

Let us define

$$w_{ij} = \frac{2\omega a_{ij}(1-a_{ij})}{d_{ij}}\big([A_d]_{ij} - a_{ij}\big) \ , \quad (19)$$

Thus, the control input becomes

$$u_i = -\kappa\sum_{j=1}^{n} w_{ij}(x_i - x_j) \ , \quad (20)$$

which is similar to the consensus-based input developed in [14] in the context of flocking and motion coordination. Assume an arbitrary orientation for the edges of graph $\mathcal{G}$. Consider the $n \times e$ incidence matrix, B, of this oriented *complete graph* with n vertices and $e = n(n-1)/2$ edges. Then equation (20) can be written as

$$\dot{\mathbf{x}} = \mathbf{u} = -\kappa\nabla_{\mathbf{x}}V = -\kappa\bar{B}\bar{W}(\mathbf{x})\bar{B}^T\mathbf{x} \quad (21)$$

where $\bar{B} = B \otimes I_3$, and $\bar{W} = W \otimes I_3$ with

$$W(\mathbf{x}) = \text{diag}\{w_{ij} \mid (i, j) \in \mathcal{E}\} \ ,$$

being a diagonal $e \times e$ matrix. ($\otimes$ denotes the Kronecker product and I_3 is the three dimensional identity matrix).

The time derivative of the Lyapunov function along the trajectories of the system becomes

$$\dot{V} = \nabla_{\mathbf{x}}V^T\dot{\mathbf{x}} = -\frac{1}{\kappa}\dot{\mathbf{x}}^T\dot{\mathbf{x}} \leq 0 \ .$$

Application of LaSalle's invariance principal over the set $\Gamma_c = \{\mathbf{x} \mid V(\mathbf{x}) < c\}$ reveals that all trajectories starting in Γ_c converge to the largest invariant set within the set $\{\mathbf{x} \mid \dot{V} = 0\}$. This set is characterized by the set of states that satisfy $\bar{B}\bar{W}(\mathbf{x})\bar{B}^T\mathbf{x} = \mathbf{0}_{3n\times 1}$, which happens if $\mathbf{x} \in null(\bar{B}^T)$, or if $W(\mathbf{x}) \equiv \mathbf{0}_{e\times e}$.

The solution set $null(\bar{B}^T) = span(\mathbf{1}_n \otimes I_3)$ corresponds to the configuration that all agents occupy the same position: $\{\mathbf{x} \mid x_i = x_j,\ \forall i, j = 1, \ldots, n\}$.

The other set of equilibrium points correspond to the set

$$\{\mathbf{x} \mid W(\mathbf{x}) = \mathbf{0}_{e\times e}\} = \{\mathbf{x} \mid A(\mathbf{x}) = A_d\},$$

where the state-dependent graph $\mathcal{G}$ has the desired adjacency matrix.

■

The set of equilibrium points where all agents are at the same position can be excluded by using a collision avoidance term in the control input. For stability analysis in the presence of collision-avoidance term, an approach similar to the recent work of Lee and Mesbahi [12] can be used. This is the subject of an ongoing work.

B. Collision Avoidance

The result of Section IV shows how network reconfiguration can be performed using only adjacency information. Though during the reconfiguration, agents could get too close to each other and violate the collision avoidance restrictions. In reality, any formation/cluster control requires collision avoidance, and collision avoidance cannot be done without range.

An inter-agent potential function [10] is defined to ensure collision avoidance during the reconfiguration. The potential function $f(d_{ij})$ is a symmetric function of the distance d_{ij} between agents i and j and is defined as follows

$$f(d_{ij}) = \begin{cases} \log d_{ij} + \frac{\delta_l}{d_{ij}} & d_{ij} < \delta_l \\ f_0 & d_{ij} \geq \delta_l \end{cases} \quad (22)$$

where $f_0 = \log(\delta_l) + 1$ is constant. The control law from this artificial potential function results in simple steering behaviors known as *separation*, which regulates the distance between the agents within the range $(0, \delta_l)$.

The total collision function of agent i is then given by

$$f_i(\mathbf{x}) = \sum_{j\in\mathcal{N}_i} f(d_{ij})$$

The total control inputs for reconfiguration now includes the additional collision avoidance term

$$\begin{aligned} u_i &= u_i^{\text{reconfig}} + u_i^{\text{collision}} \\ &= -\kappa\nabla_{x_i}V(\mathbf{x}) - \alpha\nabla_{x_i}f_i(\mathbf{x}) \quad (23) \end{aligned}$$

where u_i^{reconfig} is the reconfiguration input given by (16).

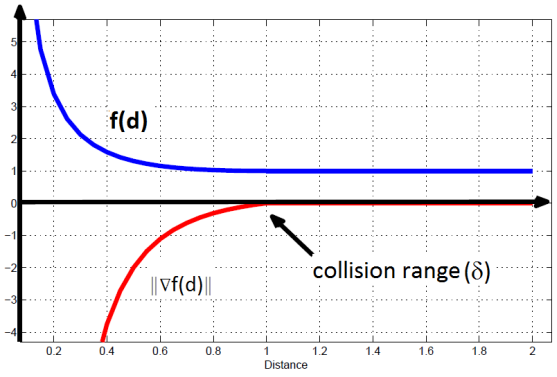
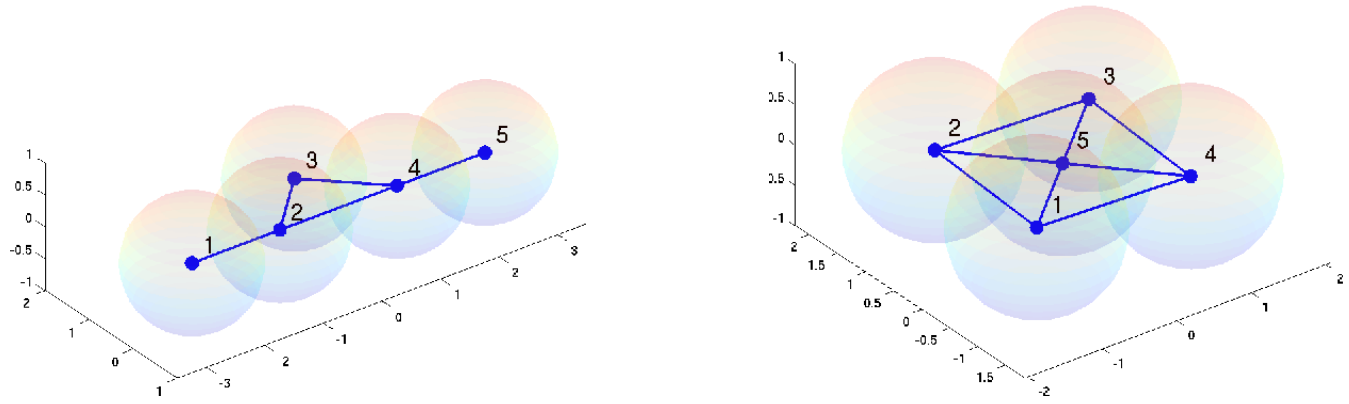


Fig. 2. Collision avoidance potential function, and the norm of its gradient.



(a) Initial Configuration

(b) Final Configuration

Fig. 3. The initial and final configurations of a network of 5 agents.

V. Simulations

Consider a cluster of $n = 5$ spacecraft in their initial configuration as shown in Figure 3. The spheres around each agent represent the collision-avoidance region with radius $\delta_l = 1$ unit. Suppose the desired final configuration of the cluster is specified in terms of adjacency matrix A_d of the final graph $\mathcal{G}_d(\mathbf{x})$:

$$A_d = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{bmatrix} \tag{24}$$

where constraint (11) must be satisfied with $\delta_l = 1$ unit and $\delta_u = 2.5$ units for all $A_{ij} = 1$.

In order to determine whether A_d corresponds to a feasible state-dependent graph (as defined in Definition 2.1), we solve the semidefinite program in (14). A feasible solution is given by the pair (Q, D):

$$Q = \begin{bmatrix} 1.56 & 0.0 & -1.56 & 0.0 & 0.0 \\ 0.0 & 1.56 & 0.0 & -1.56 & 0.0 \\ -1.56 & 0.0 & 1.56 & 0.0 & 0.0 \\ 0.0 & -1.56 & 0.0 & 1.56 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}.$$

and

$$D = \begin{bmatrix} 0 & 3.1250 & 6.2500 & 3.1250 & 1.5625 \\ 3.1250 & 0 & 3.1250 & 6.2500 & 1.5625 \\ 6.2500 & 3.1250 & 0 & 3.1250 & 1.5625 \\ 3.1250 & 6.2500 & 3.1250 & 0 & 1.5625 \\ 1.5625 & 1.5625 & 1.5625 & 1.5625 & 0 \end{bmatrix}. \tag{25}$$

It is easy to see that $rank(Q) = 2$. Therefore, A_d can be embedded in $\mathbb{R}^2$ and corresponds to a feasible graph.

Remark 5.1: Cholesky factorization of $Q = X^T X$ provides us with the position matrix X:

$$X = \begin{bmatrix} 1.25 & 0.0 & -1.25 & 0.0 & 0.0 \\ 0.0 & 1.25 & 0.0 & -1.25 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}.$$

To reconfigure the initial cluster, as given by Figure 3(a), to a desired configuration corresponding to A_d given by (24), we apply control input (23) to all spacecraft. The trajectories of all spacecraft are shown in Figure 5, where each spacecraft is located at its final position. The final positions correspond to the desired positions as in Figure 3(b). During the reconfiguration, collisions among the agents are avoided. Figure 4 shows that the value of the Lyapunov function $V(\mathbf{x})$ monotonically decreases as the cluster converges to the final configuration.

VI. Summary & Conclusions

Reconfiguration of a network of mobile agents (ground robots, UAVs, UUVs, spacecraft, etc.) is a challenging problem and of interest to NASA and DoD programs. Some of the enabling technologies are cluster flight, relative navigation, and distributed path planning. In this work, we studied the

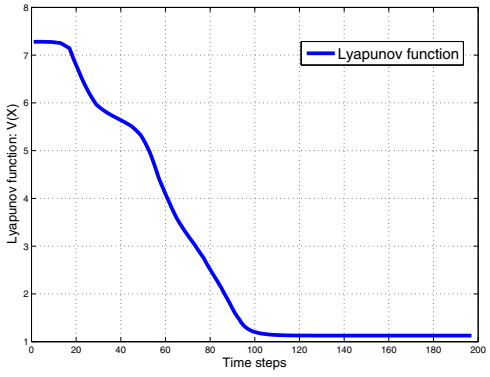


Fig. 4. Value of Lyapunov function $V(\mathbf{x})$ monotonically decreases.

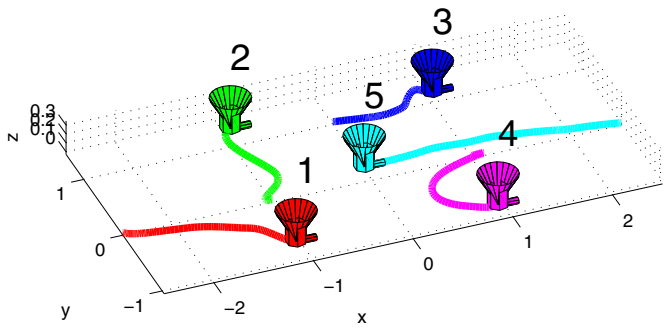


Fig. 5. Trajectories of 5 spacecraft during the reconfiguration manoeuvre.

problem of network reconfiguration using graph information only, which is the requirement for relative navigation of the agents. In order to develop a graph-based reconfiguration algorithm, we designed a control law for each mobile agent that only needed relative information (ID of the neighbors, and the distances to them). We showed the convergence to the desired equilibrium (desired graph), and verified the correctness of the control law using simulations.

We also developed a feasibility check that allowed us to see whether the desired final configuration (specified in terms of the adjacency matrix of a graph) is a feasible configuration. We formulated such graph realization problem as a search for low-dimensional embedding of the graph in the Euclidean space. Future work involves extending the realization problem to clusters of spacecraft with orientation constraints as well as distance constraints.

VII. ACKNOWLEDGMENTS

This work was supported by NASA-JPL under contract NNX10CA82C, while the first author was with Scientific Systems Company (SSCI). The authors would like to thank Dr. Behcet Açikmese (JPL), and Dr. Raman Mehra (SSCI) for their technical support.

REFERENCES

[1] P. Biswas, T.C. Liang, K.C. Toh, T.C. Wang, and Y. Ye. Semidefinite programming approaches for sensor network localization with noisy distance measurements. *IEEE Transactions on Automation Science and Engineering*, 3:4:360–371, 2006.

[2] S. Boyd and L. Vanderberghe. *Convex Optimization*. Cambridge University Press, 2004.

[3] O. Brown and P. Eremenko. The value proposition for fractionated space architectures. *AIAA*, 2006.

[4] E. J. Candes and Y. Plan. Matrix completion with noise. *Proceedings of the IEEE*, 2010.

[5] E. J. Candes and T. Tao. The power of convex relaxation: Near optimal matrix completion. Technical report, 2009.

[6] A. Man cho So. *Semidefinite programming approach to the graph realization problem: Theory, applications and extensions*. PhD thesis, Stanford University, 2007.

[7] L. Doherty, K. S. J. Pister, and L. El Ghaoui. Convex position estimation in wireless sensor networks. *Proc. 20th IEEE Infocom*, 3:1655–1663, 2001.

[8] M. Fazel. *Matrix Rank Minimization with Applications*. PhD thesis, Stanford University, 2002.

[9] M. Fazel, H. Hindi, , and S. Boyd. Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices. In *Proc. American Control Conference*, Denver, Colorado, June 2003.

[10] H. G.Tanner, A. Jadbabaie, and G. J. Pappas. Flocking in fixed and switching networks. *IEEE Transactions on Automatic Control*, 52(5):863 – 868, 2007.

[11] N. Krislock, V. Piccialli, and H. Wolkowicz. Robust semidefinite programming approaches for sensor network localization with anchors. *CORR*, May 2006.

[12] U. Lee and M. Mesbahi. Constrained consensus via logarithmic barrier functions. In *Proceeding of Conference on Decision and Control*, Orlando, Florida, Dec. 2011.

[13] M. Mesbahi and G. P. Papavassilopoulos. On the rank minimization problem over a positive semidefinite linear matrix inequality. *IEEE Transactions on Automatic Control*, 42(2):239–243, 1997.

[14] N. Moshtagh and A. Jadbabaie. Distributed geodesic control laws for flocking of nonholonomic agents. *IEEE Transactions of Automatic Control*, 52:681–686, 2007.

[15] N. Moshtagh, M. Mesbahi, and R. K. Mehra. Topology control of dynamic networks in the presence of local and global constraints. In *IEEE International Conference on Robotics and Automation*, Anchorage, Alaska, May 2010.

[16] J. Nie. Sum of squares method for sensor network localization. *Comput. Optim. Appl.*, 43:151–179, June 2009.

[17] P. A. Parrilo. Algebraic techniques and semidefinite optimization. MIT 6.256, Lecture 2, 2010.

[18] B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010.

[19] J. Saxe. Embeddability of weighted graphs in k-space is strongly np-hard. In *Proceedings of 17th Allerton Conference in Communications, Control, and Computing*, pages 480–489, Monticello, IL, 1979.

[20] I. J. Schoenberg. Remarks to maurice frechet's article "sur la definition axiomatique d'une classe d'espace distancies vectoriellement applicable sur l'espace de hilbert. *Ann. of Math.*, 36(3):724–732, 1935.

[21] J.F. Strum. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *optimization methods and software*, 11(1):545–581, 1999.

[22] P. Tseng. Second-order cone programming relaxation of sensor network localization. *SIAM Journal on Optimization*, August, 2005.