Multi-Robot Formation Shape Control Using Convex Optimization and Bottleneck Assignment

Jae D. Jeon and Beom H. Lee
Seoul National University (Department of Electrical and Computer Engineering), Seoul, Republic of Korea
Email: innocent88@snu.ac.kr, bhlee@snu.ac.kr

Abstract—This paper considers formation shape control of multiple robots while minimizing the maximum distance of robot covered. Previous studies have shown that the shape of formation is set of similarity transformations of that and solved the problem by using convex optimization. However, not only the similarity transformation but also permutation transformation preserves the shape of formation. Therefore, this work extends the earlier researches by considering the permutation factor. A threshold algorithm is employed to approach the problem finding optimal permutation of final formation, known as bottleneck assignment problem. This paper suggests an algorithm which alternately iterate the convex optimization and the bottleneck assignment algorithm until reaching convergence. The effectiveness of the proposed algorithm is presented by simulation in comparison with the previous work [1].

Index Terms—formation shape control, convex optimization, bottleneck assignment, threshold algorithm

I. INTRODUCTION

It is important to achieve a specific group formation to perform given tasks efficiently in multi-robot systems. For instance, multi-robots should compose the desired formation, which maximizes the performance of the system, when they explore unknown environment. It can be attained by not overlapping the sensing ranges of them. In another example, multi-robots should construct the formation which maximizes the net force acting on the object when they transport it. Moreover, reference [2] showed that the efficiency of traveling depended on the multi-robot formation. Thus, the formation has a great impact on the performance of multi-robot systems.

Multi-robot formation is characterized by location, orientation, scale, and shape. The first three characteristics are allowed to control independently without affecting the other characteristics. For example, the location of the formation is only changed if all robots are translated in the same distance and direction. The shape, however, cannot be controlled solely. In fact, there are some cases where the location, orientation, and scale comparisons between two formations are impossible and meaningless if the shape of them is different from each other. It is essential, therefore, to compose the formation having the desired shape when controlling multi-robot formation [3].

For that reason, multi-robot formation shape control has been studied variously. A specified desired formation shape was achieved through bidirection, gradient-based interagent distance control laws in [4], a synchronous controller in [5], bearing rigidity in [6], and a multiplicative potential energy function in [7]. In addition, Huang et al. [8] proposed control laws which find optimal scale of formation using a concept of degrees of similarity. Similarly, optimal factor of location, orientation, and scale could have obtained by using a convex optimization algorithm presented in [1], [9].

However, many previous studies, including convex optimization approaches, have considered only similarity transformation although the shape of formation is invariant with respect to permutation transformation as well as similarity transformation. Thus, we propose a multi-robot formation shape control algorithm which deals with both transformations in this paper. Because it is impossible to consider both transformations concurrently, we take into account each transformation by turns.

The paper is organized as follows. In chapter II and III, convex optimization-based multi-robot formation control algorithm and bottleneck assignment algorithms are presented, respectively. An algorithm combining the convex optimization and the bottleneck assignment algorithms is suggested in chapter IV. In chapter V, simulation results are discussed. And some conclusions are presented in chapter VI.

II. MULTI-ROBOT FORMATION SHAPE CONTROL USING CONVEX OPTIMIZATION

A. Multi-Robot Formation Shape

In this section, transformation preserving the shape of the multi-robot formation is confined to the similarity transformation, which includes translation, rotation, and scaling. That is because it has convexity property which guarantees that any local optimal solution is global optimal solution.

According to [10], the shape of a formation is all the geometrical information that remains when location, scale, and rotational effects are filtered out. Reference [1] presented a concise mathematical representation for the shape of the formation based on the concept of [10] and formulated the convex optimization problem for coordinating robot formations.
Let \( s_i \in \mathbb{R}^2 \) be the position of \( i \) th robot relative to local frame \( F \). A formation \( S \) of \( m \) robots can be represented as \( m \times 2 \) matrix
\[
S = [s_1 \ s_2 \ \cdots \ s_m]^T. \tag{1}
\]
Without loss of generality, assume that \( s_1 \) is the origin of frame \( F \).

Let \( U \) denote a formation which can be obtained by some similarity transformation. Then formation \( S \) and \( U \) are under the equivalent relation in that they have an equivalent shape. Therefore, the shape of the formation \( S \) is defined as the set of all formations which can be induced by the similarity transformation. That is equivalence class of the formation \( S \)
\[
[S] = \{aSR + 1d^T | a \in \mathbb{R}^+ \text{, } R \in SO(2), d \in \mathbb{R}^2\} \tag{2}
\]
where \( a \) is the scale, \( R \) is the rotation matrix, and \( d \) is the transition vector.

If some robot formation \( U \) is equivalent with the formation \( S \), it can be expressed as \( U \sim S \) or \( U \in [S] \) and means that
\[
U = aSR + 1d^T. \tag{3}
\]
for some \( a \), \( R \), and \( d \).

If we suppose that \( d = u_i \), equation (3) is
\[
u_i = a \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} s_i, \quad i = 2,3,\ldots,m. \tag{4}
\]
where \( \theta \) is the orientation of the formation.

If the scale and orientation of the formation is, respectively, defined as
\[
a = \frac{\|u_1 - u_2\|}{\|s_1\|}, \tag{5}
\]
\[
\theta = \arctan \frac{u_{2,y} - u_{1,y} - \arctan \frac{s_{2,y}}{s_{2,x}}}{u_{2,x} - u_{1,x}}, \tag{6}
\]
equation (4) can be transformed into
\[
\|u_i - u_{i+1}\| = R_i R_i^T (u_{i+1} - u_i), \quad i = 3,\ldots,m. \tag{7}
\]
where
\[
R_i = \begin{bmatrix} s_{i,x}^{-} & -s_{i,y}^{-} \\ s_{i,y}^{-} & s_{i,x}^{-} \end{bmatrix} \in \mathbb{R}^{2\times2}. \tag{8}
\]

Furthermore, equation (7) can be denoted in matrix form
\[
Au = 0, \tag{9}
\]
where \( A \in \mathbb{R}^{(2m-2)\times2m} \), \( u = [u_1^T \ u_2^T \ \cdots \ u_m^T]^T \in \mathbb{R}^{2m} \).

\section*{B. Convex Optimization Problem Formulation}

Let \( P = [p_1 \ p_2 \ \cdots \ p_m]^T \) denote the initial formation of \( m \) robots in world frame \( W \). We suppose that a formation, \( S \), having the desired shape is given. The goal of the multi-robot formation control is to find a new formation \( Q = [q_1 q_2 \ \cdots q_m]^T \) in \( W \), in which the maximum distance between the respective positions in \( P \) and \( Q \) are minimized. This problem can be expressed as a convex optimization problem
\[
\begin{aligned}
\text{minimize} \\
\quad \max_{i=1,\ldots,m} \|q_i - p_i\|
\text{subject to} \quad Aq = 0,
\end{aligned} \tag{10}
\]
where \( q = [q_1^T \ q_2^T \ \cdots \ q_m^T]^T \in \mathbb{R}^{2m} \).

The epigraph form of the problem (10) is the second-order cone program (SOCP)
\[
\begin{aligned}
\text{minimize} & \quad t \\
\text{subject to} & \quad \|y_i - p_i\| \leq t, \quad i = 1,2,\ldots,m, \\
& \quad Aq = 0.
\end{aligned} \tag{11}
\]

\section*{III. MULTI-ROBOT FORMATION SHAPE CONTROL USING BOTTLENECK ASSIGNMENT ALGORITHMS}

\subsection*{A. Threshold Algorithm}

In this section, we suppose that the initial formation \( P \) and the final formation \( Q \) are given. The cost coefficient \( c_{ij} \) is the distance between the final position \( q_i \) and the initial position \( p_j \). In other words, the cost matrix is \( m \times m \) matrix
\[
C = \begin{bmatrix} |q_1 - p_1| & |q_1 - p_2| & \cdots & |q_1 - p_m| \\ |q_2 - p_1| & |q_2 - p_2| & \cdots & |q_2 - p_m| \\ \vdots & \vdots & \ddots & \vdots \\ |q_m - p_1| & |q_m - p_2| & \cdots & |q_m - p_m| \end{bmatrix}. \tag{12}
\]

The objective is to assign the final destinations to robots such that the maximum distance covered is minimized. If we describe permutations, which preserve the shape of the formation, by the corresponding permutation matrices \( \Pi = (\pi_{ij}) \), the problem can be expressed as
\[
\begin{aligned}
\text{minimize} & \quad \max_{i=1,\ldots,m} \sum_{j=1}^m c_{ij} \pi_{ij} \\
\text{subject to} & \quad \sum_{j=1}^m \pi_{ij} = 1, \quad i = 1,2,\ldots,m, \\
& \quad \sum_{i=1}^m \pi_{ij} = 1, \quad j = 1,2,\ldots,m, \\
& \quad \pi_{ij} \in \{0,1\}, \quad i, j = 1,2,\ldots,m.
\end{aligned} \tag{13}
\]

To solve the above problem, a bipartite graph \( G = (U,V;E) \) with vertex sets \( U = \{q_1,q_2,\ldots,q_m\} \) and \( V = \{p_1,p_2,\ldots,p_m\} \) and edge set \( E \) is utilized. Every edge \([i,j]\) has a length \( c_{ij} \). The problem (13) is considered as finding a maximum matching in \( G \) such that the maximum length of an edge in this matching is as small as possible, bottleneck min-cost maximum matching problem [11].
One of the most well-known algorithms for the bottleneck assignment problem is using a threshold method [12]. A threshold algorithm consists of two steps. In the first step, a threshold cost $c'$ is picked and a threshold matrix $\tilde{C}$,

$$\tilde{C}_q = \begin{cases} c_q, & \text{if } c_q \leq c', \\ \infty, & \text{otherwise}, \end{cases}$$

is updated. In the second step, a bipartite graph $G$ with edges $[i, j] \in E$ if and only if $\tilde{C}_{ij} < \infty$ is constructed. Afterward, it is checked whether the bipartite graph $G$ has a perfect matching with finite total cost through Hopcroft-Karp algorithm [13]. The threshold algorithm alternates two steps until finding the smallest value $c'$ which guarantees the perfect matching of $G$. Table I provides pseudo code for the threshold algorithm in detail.

| Algorithm, Threshold Algorithm | \begin{verbatim}
Input: m×m cost matrix C
Output: maximum length $z$, modified cost matrix $\tilde{C}$
1. $c_{\min} \triangleq \min_i \{c_{ij}\}$, \hspace{1em} $c_{\max} \triangleq \max_i \{c_{ij}\}$;
2. if $c_{\min} = c_{\max}$ then $c' \triangleq c_{\min}$;
3. else
4. while $\tilde{C} = \{c_q \mid c_{\min} < c_q < c_{\max}\} \neq \emptyset$
5. \hspace{1em} $c' \triangleq \min\{c \in \tilde{C} \mid \{c_q \mid c_{\min} \leq c_q \leq c_{\max}\} \neq \emptyset\}$;
6. Feasibility_check($c'$, $c_{\min}$, $c_{\max}$);
7. endwhile;
8. if Feasibility_check has not yet been executed for $c_{\min}$ then
9. \hspace{1em} Feasibility_check($c_{\min}$, $c_{\max}$, $c_{\min}$);
10. endif;
11. $z \triangleq c_{\min}$ : $\tilde{C} = C$;
12. for each $c_q \in C$ do
13. \hspace{1em} if $c_q > z$ then $\tilde{C}_q \triangleq \infty$;
14. \hspace{1em} endif;
15. Procedure Feasibility_check($c'$, $c_{\min}$, $c_{\max}$)
16. Define the current graph $G(c')$;
17. if $G(c')$ contains a perfect matching then
18. \hspace{1em} $c_{\min} \triangleq c'$;
19. else
20. \hspace{1em} $c_{\max} \triangleq c'$;
21. endif;
22. endif;
\end{verbatim} |

| TABLE I. THE THRESHOLD ALGORITHM |

B. Hungarian Algorithm

In previous section, the modified cost matrix $\tilde{C}$ which contains a perfect matching with finite total cost is gained. Because it is possible to have more than one perfect matching in the bipartite graph $G$ having $\tilde{C}$, an additional criterion is necessary to select one of them.

In this section, we decide to minimize the sum of the corresponding costs which is selected in the matching. It can be described by the following problem

$$\text{minimize} \sum_{i=1}^{n} \sum_{j=1}^{n} \tilde{C}_{ij} x_{ij}$$

subject to $\sum_{i=1}^{n} x_{ij} = 1, \hspace{1em} i=1,2,\cdots,m, \hspace{1em} \sum_{j=1}^{n} x_{ij} = 1, \hspace{1em} j=1,2,\cdots,m, \hspace{1em} x_{ij} \in \{0,1\}, \hspace{1em} i, j=1,2,\cdots,m.$

The Hungarian algorithm [14], which belongs to primal-dual algorithms, is one of the most famous algorithms in the linear sum assignment problem (15). It also use the concept of bipartite graph $G=(U,V; E)$.

In this paper, the Hungarian algorithm executed $O(n^3)$ times [15] is implemented. This algorithm consists of three steps. In the first step, called preprocessing, a feasible dual solution and a partial primal solution satisfying the complementary slackness conditions is determined. In the next step, an augmenting tree rooted at any unsatisfied vertex in $U$ is obtained. In the final step, the solution is updated by interchanging unassigned and assigned edges along the augmenting path. The second step and final step is alternatively iterated until all vertexes in $U$ are assigned.

IV. MULTI-ROBOT FORMATION SHAPE CONTROL USING BOTH ALGORITHMS

In this chapter, multi-robot formation control algorithm considering similarity transformation and permutation transformation is suggested. When the initial formation and the desired shape is given, a problem finding optimal location, orientation, scale, and permutation of the final formation having the desired shape can be represented as

$$\text{minimize} \max_{i,j} \| y_i - p_i \| \pi_{ij}$$

subject to $A(\Pi^T \otimes I_2)q = 0,$

$$\sum_{j=1}^{n} \pi_{ij} = 1, \hspace{1em} i=1,2,\cdots,m, \hspace{1em} \sum_{i=1}^{n} \pi_{ij} = 1, \hspace{1em} j=1,2,\cdots,m, \hspace{1em} \pi_{ij} \in \{0,1\}, \hspace{1em} i, j=1,2,\cdots,m.$$
Afterward, the convex optimization algorithm is executed again with the updated initial formation. And then the bottleneck assignment algorithm is alternately iterated until the solution converges, where Π is identity matrix. Table II presents pseudo code of this algorithm.

### Algorithm

**Input:** initial formation P, 2(n-2)x2m matrix A of desired shape

**Output:** final formation Q, minmax distance d

1. Π = 0
2. while Π ≠ I do
    3. Q = Convex_Optimization(P, A); // solve the problem (11)
    4. C = ||q_i - p_j||;
    5. {d, C} = Threshold_Algorithm(C);
    6. Π = Hungarian_Algorithm(C); // solve the problem (15)
    7. P = ΠP;
3. endwhile

**TABLE II. THE PROPOSED ALGORITHM**

| Algorithm                  | Input: initial formation P, 2(n-2)x2m matrix A of desired shape | Output: final formation Q, minmax distance d |
|----------------------------|------------------------------------------------------------------|__________________________________________|
| **Input:** initial formation P, 2(n-2)x2m matrix A of desired shape |                                                                  | final formation Q, minmax distance d |
| **Output:** final formation Q, minmax distance d |                                                                  |                                          |

Afterward, the convex optimization algorithm is executed again with the updated initial formation. And then the bottleneck assignment algorithm is alternately iterated until the solution converges, where Π is identity matrix. Table II presents pseudo code of this algorithm.

**Algorithm.** Alternative iteration algorithm

**Input:** initial formation P, 2(n-2)x2m matrix A of desired shape

**Output:** final formation Q, minmax distance d

1. Π = 0
2. while Π ≠ I do
    3. Q = Convex_Optimization(P, A); // solve the problem (11)
    4. C = ||q_i - p_j||;
    5. {d, C} = Threshold_Algorithm(C);
    6. Π = Hungarian_Algorithm(C); // solve the problem (15)
    7. P = ΠP;
3. endwhile

**TABLE III. SIMULATION RESULTS COMPARISON**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>The Algorithm [1]</th>
<th>The Proposed Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distance</td>
<td>Total distance</td>
<td>2232.3m</td>
</tr>
<tr>
<td></td>
<td>Max. distance</td>
<td>58.50m</td>
</tr>
<tr>
<td>Time</td>
<td>Computation time</td>
<td>0.83s</td>
</tr>
<tr>
<td></td>
<td>Travel time</td>
<td>41.78s</td>
</tr>
<tr>
<td></td>
<td>Completion time</td>
<td>42.61s</td>
</tr>
<tr>
<td>Parameters</td>
<td>α</td>
<td>0.82m</td>
</tr>
<tr>
<td></td>
<td>θ</td>
<td>-3.43°</td>
</tr>
<tr>
<td></td>
<td>d</td>
<td>(42.00m, 43.91m)</td>
</tr>
</tbody>
</table>

**V. SIMULATION**

The algorithm was implemented in the Matlab 2012a and simulated on a PC equipped with Intel Core i7-3770 3.40GHz CPU and 8GB memory. In addition, CVX [16], developed by Michael Grant, Stephen Boyd, and Yinyu Ye, is utilized to solve the convex optimization problem. As shown in Fig. 1, the simulation environment is a two-dimensional space of 100m by 100m. There are 56 Pioneer 3DX robots, whose maximum speed is 1.4m/s. The location of all robots is randomly selected. Fig. 2 shows the desired formation shape emulated the letters ‘ICSII.’

The performance of the algorithm using both convex optimization and bottleneck assignment is compared with that of the algorithm [1] using only convex optimization. Fig. 3 and Fig. 4 show the simulation results of the algorithm [1] and the proposed algorithm, respectively. In addition, the results are summarized in Table III. We assume that robots travel at their maximum speed.

Although computation time of the proposed algorithm is longer than that of the algorithm [1], completion time of the suggested algorithm is about 12 seconds shorter than that of [1]. That is because maximum distance of the algorithm is nearly 46 percent less than that of [1]. As a result, travel time of robots is reduced with same ratio. Parameters of Table III represent the characteristics of the final formation of the equation (3).

**VI. CONCLUSION**

In this paper, the algorithm which combines the convex optimization and bottleneck assignment algorithms is
presented to control the formation shape of the multirobot system while minimizing the maximum distance of robot traveled. The convex optimization algorithm finds the optimal location, orientation, and scale of the formation and the threshold algorithm seeks the optimal permutation of the final formation under given conditions. The simulation results show that desired formation is constructed faster when the proposed algorithm is employed.

However, this algorithm cannot guarantee the optimality of solution. Even though each algorithm assures the optimality of the corresponding solution, it is damaged in the process of iteration. Therefore, it remains an open question to obtain the final formation having optimal location, orientation, scale, and permutation simultaneously. In addition, the robot position composing the final formation is only studied in this paper. Thus, control laws should be researched in the future.

ACKNOWLEDGMENT

This work was supported in part by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP) (No.2013R1A2A1A05005547), in part by the Brain Korea 21 Project, in part by ASRI, and in part by the Industrial Foundation Technology Development Program of MOTIE/KEIT [Development of CIRT(Collective Intelligence Robot Technologies)], and in part by STX Scholarship Foundation.

REFERENCES


