Optimal role and position assignment in multi-robot freely reachable formations

Alejandro R. Mosteo a,b,1 Eduardo Montijano b,c Danilo Tardioli a,b

a Centro Universitario de la Defensa, ctra de Huesca s/n, 50090 Zaragoza, Spain
b Instituto de Investigación en Ingeniería de Aragón, c/ Mariano Esquillor 1, 50018 Zaragoza, Spain
c Departamento de Ingeniería Informática y de Sistemas, Universidad de Zaragoza, Spain

Abstract

Many multi-robot problems require the achievement of formations as part of the overall mission. This work considers a scenario in which unlabeled homogeneous robots must adopt a given formation pattern buildable anywhere in the environment. This involves finding the relative pose of the formation in regard to the initial robot positions, understood as a translation and a rotation; and the optimal assignment of the role of each robot within the formation. This paper provides an optimal solution for the combined parameters of translation, rotation and assignment that minimizes total displacement. To achieve this objective we first formally prove that the three decision variables are separable. Since computing the optimal assignment without accounting for the rotation is a computationally expensive problem, we propose an algorithm that efficiently computes the optimal roles together with the rotation. The algorithm is provably correct and finds the optimal solution in finite time. A distributed implementation is also discussed. Simulation results characterize the complexity of our solution and demonstrate its effectiveness.

Key words: multi-robot formations, task allocation, pose optimization, formation roles.

1 Introduction

The problem of a team of robots reaching and maintaining a specific formation finds application in many multi-robot tasks such as cooperative manipulation [1,21], or environmental surveillance [28]. Typically, this problem is addressed by defining pairwise desired values between the robots, and then designing control laws that drive them towards these values [27]. In some cases the robots use relative positions [7], bearing angles [11] or inter-robot distances [26]. Other approaches include obstacle avoidance [2] and vision sensors [9,22]. A common assumption in all these approaches is a predefined assignment of the role of each robot in the formation, i.e., each robot has a known “label” within the formation, which defines the pairwise desired values with its neighbors based on it. Depending on the initial conditions, this can result in large displacements of all the robots or, in some cases, even convergence to undesired equilibria, as observed in some distance-based solutions [19].

A possible way to overcome this limitation is to include as a part of the problem the role assignment of each robot within the formation, i.e., given a specific pattern decide which robot should occupy which position on it. When taking this route, many solutions in the literature make the strong assumption that final coordinates for the formation positions are known at the time of role allocation: this particular optimization problem is known as the optimal or linear assignment problem [25], where the objective is, given a set of robots and a set of targets, to find a bipartite matching that minimizes some cost function, e.g., total displacement. This well-known classical problem can be centrally solved using, for example, the Hungarian [20] or simplex [8] methods. Since this problem naturally appears in many robotic contexts, it has been exhaustively analyzed from a general multi-robot task allocation perspective [13,18,29] and, more specifically, in formation-oriented works aiming at relevant bottlenecks in the role allocation like communications and scalability [3,24,33,35].

Lifting the assumptions in the previous examples, known roles and known final positions, gives rise to a more general formulation, which our work studies. There are prac-
tical advantages to this approach, that involves considering the computation of the target positions and the role assignments together in the formation problem. The joint computation can lead to overall smaller displacements of the robots, reducing, e.g., total fuel consumption, which is useful in any domain where the formation is not fixed within the global frame [32]. In addition, the solution of the joint problem implicitly satisfies proximity to the final formation, which in a distance-based context can increase the chances of convergence to the desired equilibrium. On the other hand, the joint problem poses a non-linear optimization problem with a non-trivial solution, because all the variables are intrinsically related. To overcome these issues, partial optimal solutions are computed in [15], provided one of the variables is fixed. These results are used afterwards to propose different suboptimal iterative algorithms for the remaining variables. The solution in [34] guarantees a consistent control law while using a market-based algorithm to assign formation roles to agents. This work makes no claims on the optimality of the assignment, only ensuring its soundness in regard to the absence of inconsistent allocations. Another solution is presented in [16], where optimality is sacrificed by using a two-step approach in which the formation translation and rotation are found by consensus and role allocations are corrected during an on-line phase. Inversely, [10] starts from a given assignment for a previous formation to find with convex optimization the optimal translation, rotation, and scale for a new formation. The previous assignment is kept for the formation transition and hence the solution will be typically suboptimal unless precisely the same assignment were optimal for both formations.

The main contribution of this paper is an exact and provably correct solution able to find all the optimal parameters in finite time for the problem of simultaneous role assignment and formation placement. In order to do so, we first demonstrate that the variables to optimize (translation, rotation, and role assignment) can be computed separately. Then, since the computation of the roles results in a quadratic assignment problem, computationally intractable, our second contribution is an algorithm that exploits complex-number properties of our model to efficiently compute the optimal roles and rotation simultaneously. Our third contribution is a distributed implementation of such algorithm. Taken together, these results provide a complete and efficient solution to the problem being addressed. A preliminary version of this paper was presented in [23]. This work extends our previous results to account for rotations in the formation, which considerably increase the complexity of the problem.

Our work provides a solution prior to the motion phase and relies on properties of the squared $L^2$ norm. Hence, it is optimal for holonomous robots and can be useful to other motion models for which this norm is a good approximation. Finally, any problem that requires an initial matching of roles can also benefit by using it as a generic allocation algorithm, such as control laws or motion plans that optimize the action [17,24,30] during the motion phase.

The rest of the paper is organized as follows: Section 2 formally defines the problem. A mathematical decomposition to find the optimal solution is described in Section 3. An efficient algorithm using a recursive search is described in Section 4. Section 5 provides a distributed implementation of the algorithm. Empirical evaluation with simulations is provided in Section 6. Finally, the conclusions of the work are given in Section 7.

2 Problem Definition

Let us consider a team of $N$ homogeneous robots, $\mathcal{V} = \{1,\ldots,N\}$, moving on the plane. The position of robot $i$ is denoted by $p_i \in \mathbb{R}^2$, and $p = [p_1^T,\ldots,p_N^T]^T \in \mathbb{R}^{2N \times 1}$ denotes the concatenation of all the positions.

The objective of the team is to self-organize, distributing themselves in a known specific formation pattern, which can be described by a set of points, $b_j$, and $b = [b_1^T,\ldots,b_N^T]^T$. Without loss of generality, the pattern is defined in such a way that

$$b = \sum_{k=1}^N b_k = 0_2,$$  \hspace{1cm} (1)

which can be accomplished simply by shifting the coordinates of the different $b_j$ by their original centroid. We assume that there are no restrictions about the role of the different robots in the formation, or about where it is achieved in the environment. However, we assume that a limited amount of energy is available, and thus the formation should be achieved minimizing the total distance covered by the whole team of robots in order to reduce the total fuel consumption.

In this regard, denote by $q = [q_1^T,\ldots,q_N^T]^T$ a particular realization of the pattern, expressed as a function of $b$ with the application of a translation, $q$, and a rotation, $\psi$, so that

$$q = 1_N \otimes \psi + (1_N \otimes R_\psi) b$$  \hspace{1cm} (2)

with

$$R_\psi = \begin{bmatrix} \cos \psi & -\sin \psi \\ \sin \psi & \cos \psi \end{bmatrix}$$

a rotation matrix, $1_N$ the identity matrix of dimension $N$, $1_N$ a vector with all its components equal to one and the operator $\otimes$ used to describe the Kronecker product.

In addition, in order to identify which robot takes which role within the formation, we define the set of assignment variables $x_{ij}, i, j \in \{1,\ldots,N\}$. As it is standard in assignment problems, $x_{ij} = 1$ denotes that robot $i$ assumes the role $j$ within the formation, whereas otherwise $x_{ij} = 0$. Since each robot can only be assigned one
role and there cannot be two robots with the same role,
\[ \sum_{j=1}^{N} x_{ij} = \sum_{i=1}^{N} x_{ij} = 1 \quad \text{and} \quad x_{ij} \in \{0, 1\} \quad (3) \]
for all \( i, j \in \{1, \ldots, N\} \). The assignment variables can also be arranged in a matrix, \( \mathbf{X} = [x_{ij}] \). When all the constraints in (3) are satisfied, the matrix \( \mathbf{X} \) is a permutation and, as such, \( \mathbf{X}^{-1} = \mathbf{X}^T \).

Finally, given a formation, we consider that the energy (cost) required to move robot \( r \)th to the \( j \)th position in the formation is given by the squared distance the robots need to move to reach that point, i.e., \( \| \mathbf{p}_r - \mathbf{q}_j \|_2^2 \). Therefore, the goal of the paper is to find the translation, \( \mathbf{q} \), rotation, \( \psi \), and role assignment, \( \mathbf{X} \), that result in the minimum total displacement of the whole team of robots. This can be presented as an optimization problem of the form
\[
\text{minimize} \quad \mathbf{x}, \mathbf{q}, \psi \sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij} \| \mathbf{p}_i - \mathbf{q}_j \|_2^2, \\
\text{subject to} \quad (3). \tag{4}
\]

This formulation differs from traditional task assignment problems in that, due to the new decision distance the robots need to move to reach that point, i.e., \( \| \mathbf{p}_i - \mathbf{q}_j \|_2^2 \). This sets the problem apart from the regular instantaneous optimal assignment problem [13], requiring a new analysis for an optimal and efficient algorithm. It also differs from typical formation control strategies in that, due to the new decision distance the robots need to move to reach that point, i.e., \( \| \mathbf{p}_i - \mathbf{q}_j \|_2^2 \). This can be presented as an optimization problem of the form
\[
\text{minimize} \quad \mathbf{x}, \mathbf{q}, \psi \sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij} \| \mathbf{p}_i - \mathbf{q}_j \|_2^2, \\
\text{subject to} \quad (3). \tag{4}
\]

This formulation differs from traditional task assignment problems in that, due to the new decision distance the robots need to move to reach that point, i.e., \( \| \mathbf{p}_i - \mathbf{q}_j \|_2^2 \). This sets the problem apart from the regular instantaneous optimal assignment problem [13], requiring a new analysis for an optimal and efficient algorithm. It also differs from typical formation control strategies in that, due to the new decision distance the robots need to move to reach that point, i.e., \( \| \mathbf{p}_i - \mathbf{q}_j \|_2^2 \). This can be presented as an optimization problem of the form
\[
\text{minimize} \quad \mathbf{x}, \mathbf{q}, \psi \sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij} \| \mathbf{p}_i - \mathbf{q}_j \|_2^2, \\
\text{subject to} \quad (3). \tag{4}
\]

Remark 2.1 The cost function (4) implicitly assumes that the agents can move from their positions, \( \mathbf{p}_i \), to any possible point in the environment following a straight line, i.e., single integrator dynamics. For more complex motion models, if the cost for the motion can be expressed as a linear function of the distance, \( \gamma \| \mathbf{p}_i - \mathbf{q}_j \|_2^2 \), with \( \gamma \) any positive constant, the solution discussed in this paper can still be applied with optimality guarantees. Finally, for those dynamics which do not allow such form, our algorithm can still be used as an initial solution with relatively low cost that can be refined in a second stage with any of the several existing planning algorithms for these dynamics, e.g., [17, 30].

3 Fully Decoupled Equivalent Problem

This section presents a decomposition of the problem such that the three optimization variables in eq. (4) can be solved independently. This is a case of constrained optimization over \( \mathbf{X} \) and unconstrained optimization over \( \mathbf{q} \) and \( \psi \) (with the implicit understanding that \( \psi \in [0 \ldots 2\pi] \)). Let us denote by \( f(\mathbf{q}, \psi, \mathbf{X}) \) the cost function to minimize; this function can be expressed in vectorial form by
\[
f(\mathbf{q}, \psi, \mathbf{X}) = \mathbf{p}^T \mathbf{p} + \mathbf{q}^T \mathbf{q} - 2 \mathbf{p}^T (\mathbf{X} \otimes \mathbf{I}_2) \mathbf{q}. \tag{5}
\]

Our first result demonstrates that the optimal translation of the formation is always the same, independently of the assignment or the rotation.

**Theorem 3.1** For any rotation \( \psi \in [0, 2\pi] \) and any assignment matrix, \( \mathbf{X} \), that satisfies the constraints in (3), the optimal translation of the formation, \( \mathbf{q} \), remains constant and equal to the centroid of the robot positions,
\[
\mathbf{q} = \frac{\sum_{i=1}^{N} \mathbf{p}_i}{N} = \bar{\mathbf{p}}. \tag{6}
\]

**Proof:** Let us consider a generic assignment matrix, \( \mathbf{X} \) such that it satisfies all the constraints associated to the assignment variables and a particular rotation \( \psi \). Substituting \( \mathbf{q} \) by (2),
\[
f(\mathbf{q}, \psi, \mathbf{X}) = \mathbf{p}^T \mathbf{p} + N \mathbf{q}^T \mathbf{q} + 2b^T (\mathbf{I}_N \otimes \mathbf{R}_\psi) \mathbf{q} + \mathbf{b}^T \mathbf{b} - 2 \left( \sum_{i=1}^{N} \mathbf{p}_i^T \right) \mathbf{q} - 2p^T (\mathbf{X} \otimes \mathbf{R}_\psi) \mathbf{b}. \tag{7}
\]
Since \( \mathbf{p} \) and \( \mathbf{b} \) are constant values, independent of the optimization variables, the terms \( \mathbf{p}^T \mathbf{p} \) and \( \mathbf{b}^T \mathbf{b} \) can be removed from the cost. Additionally, using (1),
\[
2 \mathbf{b}^T (\mathbf{I}_N \otimes \mathbf{R}_\psi) \mathbf{q} + q_\bar{q} = 2q^T \mathbf{R}_\psi \sum_{k=1}^{N} \mathbf{b}_k = 0. \tag{8}
\]
Therefore, the minimization of \( f(\mathbf{q}, \psi, \mathbf{X}) \) is equivalent to the minimization of
\[
g(\mathbf{q}, \psi, \mathbf{X}) = N \mathbf{q}^T \mathbf{q} - 2 \left( \sum_{i=1}^{N} \mathbf{p}_i^T \right) \mathbf{q} - 2p^T (\mathbf{X} \otimes \mathbf{R}_\psi) \mathbf{b}. \tag{9}
\]
The partial derivative of (9) with respect to \( \mathbf{q} \) is equal to
\[
\frac{\partial g(\mathbf{q}, \psi, \mathbf{X})}{\partial \mathbf{q}} = 2N \mathbf{q} - 2 \sum_{i=1}^{N} \mathbf{p}_i, \tag{10}
\]
which does not depend on \( \mathbf{X} \) or \( \psi \). Since \( \mathbf{q} \) is a free variable, we can find the optimal value making the derivative zero, which yields (6). \( \square \)

The theorem shows the independence of \( \mathbf{q} \) from the other two optimization variables. The rotation and assignment parameters are, however, interconnected in a way that precludes such isolated optimization. Nonetheless, it is still possible to decouple both variables working with (9). Let us define the variables \( \alpha_{ij} \) and \( \beta_{ij} \),
\[
\alpha_{ij} = \mathbf{p}_i^T \mathbf{b}_j, \quad \beta_{ij} = \mathbf{p}_i^T \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \mathbf{b}_j \tag{11}
\]
computed from the robot positions and the desired formation pattern.
Theorem 3.2 The optimal assignment, $X^*$, for the problem (4) is the same as the one that solves the following optimization problem

$$\text{maximize}_{X} \left(\sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij} \alpha_{ij}\right)^2 + \left(\sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij} \beta_{ij}\right)^2 \quad (12)$$

subject to (3).

Proof: Doing some algebraic manipulation in the last term of the right hand side of expression (9)

$$p^T(X \otimes R_\psi)b = \sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij} (p_i^T R_\psi b_j) \quad (13)$$

$$= \sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij} (\alpha_{ij} \cos \psi + \beta_{ij} \sin \psi) \quad (14)$$

$$= \sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij} M_{ij} \cos(\psi - \phi_{ij}) \quad (15)$$

where $\alpha_{ij}$ and $\beta_{ij}$ are given in (11) and

$$M_{ij} = \sqrt{\alpha_{ij}^2 + \beta_{ij}^2}, \quad \phi_{ij} = \arctan(\beta_{ij}, \alpha_{ij}). \quad (16)$$

Noting that (15) represents a sum of sinusoids of a same ‘frequency’, namely the unity, for any value of $X$, the outcome is necessarily a new sinusoid of the same frequency and modulus, $M(X)$, with phase, $\phi(X)$, dependent on the $M_{ij}$ and $\phi_{ij}$ selected by $X$:

$$2 \sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij} M_{ij} \cos(\psi - \phi_{ij}) = M(X) \cos(\psi - \phi(X)). \quad (17)$$

This way, recalling that this term is subtracting in (9), the original optimal assignment is equivalent to the assignment that maximizes $M(X)$.

A simple way to compute the sum of multiple trigonometric functions of the same frequency is by considering their complex notation applying Euler’s formula. With this representation we just need to sum the real and the imaginary parts separately, and then transform back the result into the trigonometric form. At this point is when eq. (14) comes in handy, because it provides the complex representation of (15), with the real part of the complex defined by $\alpha_{ij}$ and the imaginary part by $\beta_{ij}$. Therefore,

$$M(X) = \left(\sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij} \alpha_{ij}\right)^2 + \left(\sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij} \beta_{ij}\right)^2, \quad (18)$$

which shows that (12) finds the optimal assignment.

Consequently, the objective is to find the assignment that maximizes the modulus of the complex number given by the sum of subsets of $\alpha_{ij}$ and $\beta_{ij}$, which is precisely (12).

The previous theorem demonstrates that the optimal assignment can be computed independently from the rotation angle, $\psi$. Once $X^*$ is available, the optimal rotation angle is the one that makes the cosine equal to one, i.e., $\psi^* = \phi(X^*)$, where

$$\psi^* = \phi(X^*) = -\arctan\left(\frac{\left(\sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij} \beta_{ij}\right)^2}{\left(\sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij} \alpha_{ij}\right)^2}\right). \quad (19)$$

Remark 3.1 Although this decomposition shows the independence of the three decision variables, in practice the optimal assignment computation is not trivial. Unfortunately, (12) represents an NP-Hard quadratic assignment problem [6], for which an efficient optimal solution is unknown.

In the next section we propose an efficient solution using a recursive search approach that optimizes both the rotation and the assignment simultaneously.

4 Optimal efficient solution

The presented formulation exhibits properties that enable the design of another optimal exact approach, at the cost of solving several regular assignment problem instances. This is achieved without depending on any discretization parameter.

4.1 Algorithm explanation

The algorithm requires as inputs the robots and formation positions, $p_i, b_j$, to exactly solve particular instances of the problem for a given rotation. The matrices $\alpha, \beta$ from eq. (11) are used to obtain the optimal rotation for a particular solution $X$. An initial solution is computed in order to obtain an initial bound using the Hungarian or simplex algorithms, for example for initial rotation $\psi = 0$.

Next, increasingly small rotation sub-ranges are evaluated in a recursive refinement fashion until the full rotation space is explored, $[0, 2\pi]$. Consider a particular sub-range, $[\psi_L, \psi_R]$, such that $\psi_R - \psi_L < \pi$. The algorithm computes the optimal assignment for the two angle extrema, namely $X_L$ and $X_R$. The two solutions have two costs, $c_L$ and $c_R$, which are optimal for those two particular rotations. Conversely, given one assignment, the optimal rotation and cost for that assignment can be easily computed. These costs are denoted $M_L$ and $M_R$, and correspond to the amplitude in (18) for a particular assignment.

If the optimal assignment is the same for both extrema, $X_L = X_R$, then it also holds that $M_L = M_R$. Moreover, for any angle within the range, we will prove that the optimal assignment is the same, which means that there is no further need to sub-divide that interval. If this value improves the best known cost, then it becomes the new bound and the optimal assignment is also updated.

©2017. This manuscript version is made available under the CC-BY-NC-ND 4.0 license.

http://creativecommons.org/licenses/by-nc-nd/4.0/

Final version available at [https://doi.org/10.1016/j.automatica.2017.03.040](https://doi.org/10.1016/j.automatica.2017.03.040)
If the optimal assignment is different, $X_L \neq X_R$, we compute the cosine with fixed period equal to $2\pi$ that goes through the points $(\psi_L, c_L)$ and $(\psi_R, c_R)$ [4]. We let $M_{LR}\cos(\phi - \phi_{LR})$ be such cosine, with

$$M_{LR} = \sqrt{c_L^2 + c_R^2 - 2c_Lc_R \cos(\psi_R - \psi_L)},$$  \hspace{1cm} (20)

$$\phi_{LR} = \frac{\pi}{2} + \arctan\left(\frac{c_R \sin(\psi_L - c_L \sin(\psi_R)}{c_R \cos(\psi_L - c_L \cos(\psi_R)}\right).$$  \hspace{1cm} (21)

If $M_{LR}$ is larger than the best known cost, then the interval $[\psi_L, \psi_R]$ is divided into two sub-intervals and the procedure is repeated for both of them. Otherwise the interval can be pruned out.

Once the solution space has been explored, the optimal rotation $\psi^*$ is obtained with (19) from the optimal assignment $X^*$ and the algorithm is done.

This solution is detailed in Algorithm 1. The procedure “SOLVEFOR” stands for an instance of the Hungarian or simplex for the given rotation, and “FITCOSINE” refers to (20)-(21). The algorithm is presented in recursive form (lines 2-4, 19-20) for simplicity and the reason for the initial trisection (lines 2-4) is to ensure that $\psi_R - \psi_L < \pi$ for every interval.

**Algorithm 1 Optimal $X^*$ and $\psi^*$**

**Inputs:** p, b, α, β

// Obtain initial bound using Hungarian and (16)
1: $X^*, M^* \leftarrow$ SOLVEFOR($\psi = 0$)

// Start search
2: CHECKRANGE(0, 2\pi/3)
3: CHECKRANGE(2\pi/3, 4\pi/3)
4: CHECKRANGE(4\pi/3, 2\pi)

5: $\psi^* \leftarrow$ OBTAINFROM($X^*$) using (19)
6: return $X^*, \psi^*$

7: procedure CHECKRANGE($\psi_L, \psi_R$)
8: \{$c_L, M_L, X_L\} \leftarrow$ SOLVEFOR($\psi = \psi_L$)
9: \{$c_R, M_R, X_R\} \leftarrow$ SOLVEFOR($\psi = \psi_R$)
10: $M^* \leftarrow$ MAX($M^*, M_L, M_R$)

// Update bound
11: if $X_L = X_R$ then
12: if $M_L = M^*$ then
13: $X^* \leftarrow X_L$
14: end if
15: else
16: \{$\phi_{LR}, M_{LR}\} \leftarrow$ FITCOSINE($\psi_L, \psi_R, c_L, c_R$)
17: if $M_{LR} > M^*$ AND $\phi_{LR} \in [\psi_L, \psi_R]$ then
18: $\psi_C \leftarrow (\psi_L + \psi_R)/2$
19: CHECKRANGE($\psi_L, \psi_C$)
20: CHECKRANGE($\psi_C, \psi_R$)
21: end if
22: end if
23: end procedure

4.2 Implementation considerations

Some efficiency improvements have been omitted to streamline the presentation. More precisely, the solutions at lines 8, 9 are sometimes redundant, since the solution for a particular $\psi$ is evaluated more than once. This deficiency is fixed by moving the computation to the calling point (lines 19, 20). Also, the intrinsically parallel nature of the algorithm suggests the generation of more than two subranges for each refinement if more computational units are available.

While a recursive implementation is conceptually simpler, a priority queue with pending ranges can be used instead in an iterative implementation. By ordering these ranges by their potential best bound, $M_{LR}$, we found noticeable improvements in the algorithm efficiency.

4.3 Optimality and termination

We formally demonstrate now that Algorithm 1 finds the optimal solution in finite time. The outline of this demonstration is first to show that the cosine given in (20) and (21) represents an upper bound of the cost in the interval $[\psi_L, \psi_R]$. This allows us to discard any interval for which the optimal assignment at both extrema is the same. Then we show that intervals are not divided an infinite number of times, meaning that the algorithm will find the optimal solution in a finite amount of time.

**Proposition 4.1** Let $(\psi_L, c_L)$ and $(\psi_R, c_R)$ be such that $\psi_L < \psi_R$, $\psi_R - \psi_L < \pi$ and $c_L$ and $c_R$ are the best cost values obtained at $\psi_L$ and $\psi_R$ respectively via the optimal assignments at those orientations. Then, the values of the cosine $M_{LR}\cos(\phi - \phi_{LR})$, with (20) and (21), in the interval $[\psi_L, \psi_R]$, represent an upper bound for the cosine of the optimal assignments in those rotations.

**Proof:** We prove this result showing that any cosine fitted with some value above $M_{LR}\cos(\phi - \phi_{LR})$ in $(\psi_L, \psi_R)$ is not feasible, in the sense that the value of such cosine in $\psi_L$ (or $\psi_R$) is strictly greater than $c_L$ (or $c_R$), which cannot be possible due to the optimality of $c_L$ (or $c_R$).

Choose any other cosine, with period equal to $2\pi$, such that there is at least some $\psi \in (\psi_L, \psi_R)$ for which such cosine has a greater value than the cosine defined using (20) and (21). Having the same period, the two cosines will intersect at two angles, $\psi_1$ and $\psi_2 = \psi_1 + \pi$. Since these are continuous functions, one of the cosines will be greater than the other in the interval $(\psi_1, \psi_2)$ and the other one in the intervals $(0, \psi_1)$ and $(\psi_2, 2\pi)$. If one of such intersections is within $[\psi_L, \psi_R]$ then the new cosine will also have a greater value than $c_L$ at $\psi_L$ or than $c_R$ at $\psi_R$, which cannot occur. Otherwise $[\psi_L, \psi_R] \subset [\psi_1, \psi_2]$. In such case, by continuity of the cosines, if one point is over the cosine defined by (20) and (21), then all the points will be above, again violating the optimality of $c_L$ and $c_R$ at $\psi_L$ and $\psi_R$. Therefore, we conclude that the cosine defined by (20) and (21) provides an upper bound for the cost of the optimization problem in the interval $[\psi_L, \psi_R]$. ☐

The next corollary is a direct implication of the previous proposition:

**Corollary 4.1** Given two rotations, $\psi_L < \psi_R$, with $X(\psi_L) = X(\psi_R)$ and $\psi_R - \psi_L < \pi$, then $\forall \psi_M \in [\psi_L \cdots \psi_R] \Rightarrow X(\psi_M) = X(\psi_L)$. 

©2017. This manuscript version is made available under the CC-BY-NC-ND 4.0 license.

http://creativecommons.org/licenses/by-nc-nd/4.0/

Final version available at: https://doi.org/10.1016/j.automatica.2017.03.040
Remark 4.1 These two results show that the algorithm will always find the optimal solution \( X^* \). When we prune an interval because both extreme have the same assignment, we can assert that there will not be a different optimal assignment in between them. Also, when the maximum of the bound cosine is outside the interval, the maximum value of the problem is achieved at the maximum between \( c_L \) and \( c_R \), and since the optimal assignment at this value is already computed, there is no need for further exploration of such interval. Therefore, whenever an interval is pruned the optimal solution is not contained in that interval.

The second result shows that the algorithm is not dividing an infinite number of times a given interval.

Proposition 4.2 Algorithm 1 terminates in finite time.

Proof: Assume in the worst case that the exploration of intervals is performed up to the point in which the interval contains only two optimal assignments, the ones obtained for \( \psi_L \) and \( \psi_R \). If the interval is divided in two again, one of the two halves will be discarded, as necessarily both extremas will have the same optimal assignment. Regarding the other half, the cost obtained at point \((\psi_L + \psi_R)/2\) will be strictly smaller than the value of the upper bound cosine at that point, as otherwise the two optimal assignments would have to be, as proved, the same.

We need to show now that this division of the interval in two does not occur infinitely. For simplicity in the notation, we perform this analysis considering the interval between \( \psi_L \) and \( \psi_R \) (instead of \( \psi_L \) or \( \psi_R \) and \((\psi_L + \psi_R)/2\) and we denote \( \psi_L = r - \psi_L \). We analyze now how the value of \( M_{LR} \) varies by reducing \( c_L \). A symmetric analysis can be applied fixing \( c_L \) and decreasing \( c_R \), reaching the same conclusions. The partial derivatives of (20) and (21) with respect to \( c_L \) are

\[
\frac{\partial M_{LR}}{\partial c_L} = \frac{2c_L - 2c_R \cos(\psi_{LR})}{\sin^2(\psi_{LR}) M_{LR}}, \quad (22)
\]

\[
\frac{\partial \phi_{LR}}{\partial c_L} = -c_R \sin(\psi_{LR}) < 0. \quad (23)
\]

Since the denominator of (22) is always positive, we only need to analyze what happens with the numerator. If \( c_L = c_R \cos(\psi_{LR}) \), then \( M_{LR} = c_R \), and thus, \( \phi_{LR} = \psi_R \). For smaller values of \( c_L \), (22) becomes negative, which means that at this point \( M_{LR} \) increases. However, noting that (23) is always negative, the point where \( M_{LR} \) is achieved, \( \phi_{LR} \) must be strictly greater than \( \psi_R \) and then the interval will be discarded because \( \phi_{LR} \notin [\psi_L, \psi_R] \).

On the other hand, if \( c_L > c_R \cos(\psi_{LR}) \) then eq. (22) is positive, which means that in this case \( M_{LR} \) is an increasing function with respect to \( c_L \). Therefore, for smaller values of \( c_L \), such that they are greater than \( c_R \cos(\psi_{LR}) \), \( M_{LR} \) will be decreased. In this case, due to the continuity of the optimal value over \( \psi \), as \( \psi_{LR} \) tends to zero, the values \( c_L \) and \( c_R \) approach to each other.

Thus,

\[
\lim_{\psi_{LR} \to 0} \frac{\sqrt{2c_L^2 - 2c_R^2 \cos \psi_{LR}}}{\sin \psi_{LR}} = c_L.
\]

We finally need to consider two scenarios, the first one when \( c_L < M_L \) and the second one when \( c_L = M_L \), with \( M_L \) being the amplitude for the cosine given by the optimal assignment at \( \psi_L \). In the first case, since \( M_{LR} \) converges to \( c_L \), there exists some \( \epsilon > 0 \) such that, for an interval of length \( \epsilon \), \( M_{LR} < M_L \leq M^* \), meaning that \( M_{LR} \leq M^* \) and the interval is not split anymore.

In the second case, when \( c_L = M_L \), denote \( \psi_M \in (\psi_L, \psi_R) \) as the angle in which the optimal assignment changes from the one given in \( \psi_L \) to the one given in \( \psi_R \). There exists a finite number of divisions by two such that \( \psi_M \) will not be contained in the left interval (the one starting at \( \psi_L \)). When that happens both extremas of the left interval will have the same optimal assignment, thus terminating the exploration of this interval, and the right interval will correspond to the first case, showing as well a finite time termination.

Remark 4.2 The recursive essence of the algorithm enables that every subrange check be made by a different computational unit. Hence, the simplest optimization is to parallelize subrange checking with as many computational units as possible.

5 Distributed Implementation of the Algorithm

In this section we outline the foundation for a fully distributed solution in case of limited communications [14]. Assume the team of robots is arranged in an undirected connected graph, \( G \), where the nodes represent the robots, the edges represent direct communication links, and \( \mathcal{N}_i \) is the set of neighbors of the robot \( i \). The goal is to use distributed consensus algorithms to address the different problems involved in the computation of the optimal solution. Initially, robots know the formation \( q \), in a distributed fashion. This computation can be done by means of any existing distributed averaging algorithm, e.g., [31], running a standard linear iteration of the form

\[
\bar{q}(n+1) = (W \otimes I_d)\bar{q}(n),
\]

with \( W \) a doubly stochastic weight matrix compatible with the communication graph and \( \bar{q}(n) = (\bar{q}_1(n), \ldots, \bar{q}_N(n))^T \) the vector with the estimation at iteration \( n \) that all the robots have about the centroid of their positions, initialized by \( \bar{q}_i(0) = p_i \). It is well known [31] that the previous iteration converges to:

\[
\lim_{n \to \infty} \bar{q}(n) = 1_N \otimes \left( \frac{1}{N} \sum_{i \in \mathcal{V}} p_i \right).
\]

©2017. This manuscript version is made available under the CC-BY-NC-ND 4.0 license. http://creativecommons.org/licenses/by-nc-nd/4.0/ Final version available at https://doi.org/10.1016/j.automatica.2017.03.040
Algorithm 1 can also be solved using only local communications. Note that the remaining issue to be solved in a distributed fashion is the computation of the optimal assignment for a particular rotation, \( \psi \), and its associated cost, \( c \); i.e., distributively compute \( \text{SolveFor}(\psi) \).

To do so, we propose to use a distributed simplex like the one described in [5]. Given a particular rotation, \( \psi \), denote \( c_{ij} = M_{ij} \cos(\psi - \phi_{ij}) \), computed from (11), (16).

The distributed simplex algorithm defines a partition \( \mathcal{P} = \{ \mathcal{P}^1, \ldots, \mathcal{P}^N \} \) of the full problem columns so that robot \( i \) initially only knows the columns relative to its own costs for every task \( c_{ij} \) for all \( j \), plus a set of artificial columns that form its initial lex-feasible basis, \( \mathcal{B}_M \), constructed using the big-M method.

Robots periodically broadcast to neighbors the columns forming their current optimal basis, \( \mathcal{B}^i \), with their associated costs. After a communication round each robot solves to optimality, using as starting basis its current one, the subset simplex formed by the columns it is aware of: its permanent set \( \mathcal{P}^i \), its current basis, \( \mathcal{B}^i \), and the bases received in the last round, \( \mathcal{B}^N \). By virtue of [5], eventually all robots arrive at an optimal, same basis for the complete problem, in this case the optimal assignment for (17).

Now, given two angles, \( \psi_L \) and \( \psi_R \), and their associated optimal costs, \( c_L \) and \( c_R \), computed using the previous procedure, the upper-bound cosine in (20), (21), can be locally found by each robot. Therefore, all the robots can locally decide the next interval to call in \( \text{CheckRange}() \), which will be the same for all of them under the assumption of sharing the same reference frame. By Proposition 4.1 and Proposition 4.2 we can assert that eventually all the robots will reach the optimal assignment and rotation.

Algorithm 2 shows a high-level implementation of the whole approach. The lines in the algorithm beginning with “||” account for the parts of the algorithm that are run concurrently, whereas the lines with cited references account for distributed computations, following the descriptions above.

**Algorithm 2** Distributed Optimal \( q^* \), \( X^* \) and \( \psi^* \) as run by robot \( i \)

**Inputs:** \( p, b, W \). Common frame (e.g., [12])

1. \( || \) \( q^* \leftarrow \text{Find robots’ centroid} \) (e.g., [31]).
2. \( || \) \( X^*, \psi^* \leftarrow \text{Algorithm 1 using DistSolveFor instead of SolveFor} \)
3. \( \text{return} \ q^*, X^*, \psi^* \)

4. **procedure** DistSolveFor(\( \psi \))
5. Prepare local data \( c_{ij}, \forall j, \psi \) with (11), (16).
6. **Solve Assignment** (17) (e.g., [5]).
7. **return** \( c, M, X \)
8. **end procedure**

An alternative implementation without the global frame requirement is also possible. Since the assignments computed in Algorithm 1 consider a fixed value of \( \psi \), if \( \bar{q} \) is precomputed by the robots, instead of (17) in Line 6 they could distributively solve a standard task assignment based on the total distance. In this case only the distance to the goal is required, and this can be computed independently of each robot’s frame. The limitation of this approach compared to the previous one is that it requires to be run in two steps, one to compute \( \bar{q} \), using e.g., [12], and another one to run the distributed version of Algorithm 1.

### 6 Simulations

In this section we present empirical evaluations. Firstly, an example instance is analyzed in detail to highlight the concepts behind the algorithm. Then, exhaustive simulations show the typical performance of the algorithm as a function of the instance size.

#### 6.1 Illustrative example

We use a small problem instance to enhance clarity of presentation. This case is geometrically depicted in Fig. 1, where the robots, desired formation and the found final target positions can be seen, as well as the optimal rotation and assignment.

![Figure 1. A solution example with four robots.](https://example.com/figure1.png)

Given that the optimal translation has been shown to be computable in a first separate step, all figures that follow have this translation already applied. Fig. 2 shows the costs associated to each robot for some possible assignments and all rotations. The sinusoids predicted by eq. (14) are clearly visible. The most/least convenient rotation for each robot is at the valley/peak of its sinusoid. Since there are as many possible allocations as permutations, for \( N=4 \) there are 24 possible assignments. The final cost of the solution is the sum of the sinusoid values at the optimal rotation abscissa. The sum of individual robot sinusoids in each subplot in Fig. 2 is another sinusoid representing the cost of the whole assignment. These sinusoids (for all 24 assignments) are shown in Fig. 3. All sinusoids have the same period and offset (given by the constant costs in (7)), meaning that the largest amplitude determines the optimal assignment, independently of the optimal rotation,
which can be found later at the valley of its sinusoid. Consequently, the lower frontier of all assignments in Fig. 3 gives the optimal assignment and cost for each rotation angle.

A step-by-step insight into the algorithm operation is in Fig. 4, which depicts the pruning progress for this particular instance. The numbered cosines identify the recursion depth at which each range has been examined by the refinement process. A cosine corresponds to an actual assignment when the solutions found at its bound-aries match (blue lines) or to a potential bound when they differ (red/green lines).

Initial potential bounds (labeled with ‘1’) are clearly optimistic (besides being unfeasible, as the cost cannot be negative). At the next recursion level, ‘2’, the optimal solution is already found (‘x’ marker) albeit this is not known until the complete rotation range is explored. At depth ‘3’ the rotation space is exhausted and the algorithm is done. In this particular example 13 sub-ranges were considered (corresponding to each plotted cosine in Fig. 4) and 8 Hungarian problem instances were solved (one for each angle-split operation).

6.2 Simulation results

Batch simulations were run to characterize the typical execution cost of the algorithm. Since the costlier operation is the solving of Hungarian instances, which has complexity \(O(N^3)\), the total cost of instances solved was one of the selected metrics. The number of robots ranged from 4 to 128, with the limiting factor being the memory necessary to store the matrices rather than computing time. Robots were always randomly placed, whereas three formation scenarios were considered (Fig. 5): random configurations which do not contain symmetries, an abreast formation that contains two symmetries, and a worst-case (from the point of view of computational cost) circular configuration which, having as many symmetries/optimal solutions as targets, forces the finding of all of them during the search. A hundred identically seeded instances were run for each combination of team size and scenario.

Figure 6 (a) shows the amount of unique Hungarian problem instances solved. The results show that the search effort, measured in subranges evaluated, increases in the worst case of maximum symmetries roughly linearly with the number of robots/tasks. Let us recall that splitting a range (algorithm line 17) requires the solving of one Hungarian instance at \(\psi\).

Fig. 6 (b) shows the count of leaf nodes having a unique solution found during the search. These leaves provide a locally optimal assignment for a subrange and thus an exact bound for the search, corresponding to line 13 of the algorithm. It is clear that for the circular formation at least as many leaves as symmetric solutions have to
be found. However, the small number of leaves reached during the search in the other cases is striking. The implication is that the bounds found by the algorithm are in practice very tight, which explains its quasi-linear performance in terms of number of nodes to be explored. Consequently, even if formations with many symmetries are sought, it may be worth to slightly noisify them to avoid the worst-case performance of the algorithm.

Fig. 6 (c) shows the maximum recursion depth reached during algorithm execution, where a logarithmic trend is observed. Since each recursion level splits its subranges in two, creating a new Hungarian instance with cost $O(N^3)$, we arrive at a complexity of $O(2 \log_2 N^2) = O(N^4)$, consistent with the previous figures complexity of $N$. Hence, the algorithm adds a worst-case linear cost over the solving of a single optimal assignment problem.

7 Conclusion

This article has studied the problem of assigning formation roles to robots when the formation is not constrained by translation or rotation, by minimizing the sum of squared distances between each robot and its target position. By defining the formation restrictions with a null centroid, we have been able to demonstrate that the three decision parameters can be separated. The centroid can be computed with a standard averaging process, whereas the assignment can be obtained by solving a quadratic maximization problem. The rotation is directly computable from the assignment afterwards using our model. Since there are no known well-performing algorithms for such maximization problems, we have exploited the complex-number properties of our formula to design an efficient algorithm that simultaneously finds the assignment and rotation with a worst-case complexity of $O(N^4)$ according to extensive simulations. We have also described the distributed implementation of this algorithm in a graph-like network.

Acknowledgements

This research has been supported by the projects ROBOCHALLENGE (DPI2016-76676-R-AEI/FEDER-UE) and ALERTA (CUD2016-17).

References


Figure 6. Simulation results. (a) Amount of Hungarian problem instances solved. A linear trend is shown for reference. The performance is roughly linear in the worst-case circular formation. (b) Number of exact bounds found during search. Only the worst-case scenario with maximal symmetries exhibits the linear trend. (c) Maximum recursivity depth reached during algorithm execution. A logarithmic trend is shown for reference.


