Towards Collective Manipulation Without Inter-Agent Communication

Siamak G. Faal
Worcester Polytechnic Institute
100 Institute Rd
Worcester, MA
sghorbanifaal@wpi.edu

Shadi Tasdighi Kalat
Worcester Polytechnic Institute
100 Institute Rd
Worcester, MA
stasdighikalat@wpi.edu

Cagdas D. Onal
Worcester Polytechnic Institute
100 Institute Rd
Worcester, MA
cdonal@wpi.edu

ABSTRACT
This article details the formulation of a decentralized controller for collective manipulation that does not require any communication between agents involved in the task. First, a centralized controller based on the complete system Jacobian is discussed as a benchmark. Then, the centralized controller is reformulated to obtain the algorithm for the proposed decentralized control approach. Both, centralized and decentralized controllers utilize Moore-Penrose pseudoinverse to distribute a control action through the agents of the group. The convergence and stability of both controllers are discussed in detail. Moreover, robustness and effectiveness of the proposed controllers are investigated through simulating numerous scenarios, formations and populations of the agents. We show that, as the population of the group increases, the results of the decentralized controller approach to its centralized counterpart with significantly lower computational cost.

CCS Concepts
• Computer systems organization → Robotic control;
  Robotic autonomy;
• Computing methodologies → Multi-agent systems;
  Cooperation and coordination;
  Control methods;

Keywords
Collective manipulation; Decentralized controller; Swarm robotics

1. INTRODUCTION
Collective manipulation is a solution for manipulating relatively large and heavy objects by utilizing agents that are incapable of fulfilling the task individually. This behavior is frequently observed in different insect colonies during the foraging process. The large prey retrieval of Eciton Burchelli, also known as army ants, is an impressive example of collective manipulation in nature [1]. Collective manipulation can be employed in a wide range of applications and has been a research focus in swarm and multi-robot studies. Applications of collective manipulation include: carrying and assembling parts for automated on-site construction; object manipulation and assembly in factories; search and rescue operations in disaster relief actions; and debris collection. Although utilizing a group of simple agents, compared to a single well-instrumented agent, increases manipulation dexterity, reliability, and robustness, it introduces new challenges on team formation, organization, and control.

Collective manipulation is achievable through two fundamental control strategies: centralized control [2, 3, 4] and decentralized control [5, 6, 7, 8, 9, 10, 11]. The centralized control approaches mostly focus on group formations and manipulation is performed by keeping form- or force-closure around the manipulated object. While centralized approaches can guarantee a form of optimality, they suffer from intense internal communication between agents of the group. Combined with the large population of the group, the required level of communication make it impractical to utilize centralized control approaches in many real life scenarios [12].

Figure 1: Snapshot of three $\Delta \rho$ robots carrying a four-leaf clover.
2. MODELING

This section covers the derivation of the differential equations of motion (DEM) for the system under consideration. Although a reformulation of the proposed controller may make it applicable to non-rigid and/or 3-D systems, for the sake of brevity, the manipulated object is considered to be a rigid body that is constrained to planar motion. Throughout this manuscript, the vector and coordinate frame notations are adopted form [16]. Based on these notations, a transformation $A$ from coordinate frame $\{i\}$ to coordinate frame $\{j\}$ is designated by $\hat{J}_A$. Similarly, $\hat{v}$ represents the vector $v$ that defined in coordinate frame $\{i\}$. A free body diagram of the overall system is illustrated in Fig. 2. Note that the coordinate frame $\{O\}$ is attached to the Center of Mass (CoM) of the object. The vector $\hat{O}_r$ defines the position of the applied force by the $i^{th}$ agent with respect to the CoM of the object. The position of the $i^{th}$ agent with respect to the CoM of the object is defined in coordinate frame $\{O\}$. Similar to $\hat{O}_r$, $\hat{f}_i$ defines the forces applied by the $i^{th}$ agent to the object in the same coordinate frame. The vector $\hat{O}_{e_d}$ is the linear error between the desired position and the object position. The total error vector, that includes linear and angular error values, is described by (1).

$$\hat{O}_e = \hat{O}_w R \times \hat{O} w e,$$  

(1)

where $\hat{O}_w R$ is the rotation matrix that maps the world coordinate frame $\{W\}$ to the object frame $\{O\}$. Since the object is constrained to planar motions, $\hat{O}_w R$ is equal to the elemental rotation about $z$ axis for $\theta_o$ radians, $R_z(\theta_o)$. The total error vector in world coordinate frame, $\hat{w} e$, is defined as:

$$\hat{w} e = \hat{W} e = \hat{W} \left[(x_d - x_o), \hat{\dot{y}} + (y_d - y_o), \hat{\dot{\theta}} + (\theta_d - \theta_o), \hat{k}\right].$$  

(2)

Equations (3) and (4) describe the DEM of an object with mass of $m_o$ and mass moment of inertia of $I_o$ written in the body fixed (non-inertial) reference frame $\{O\}$. The vectors $\hat{O} v_o$ and $\hat{O} f_i$ represent the velocity of the object and the forces applied by each agent, respectively. The position of the applied force by the $i^{th}$ agent with respect to the CoM of the object is defined by $\hat{O} r_i$. This position is also referred as the agent position throughout this article.

$$m_o (\hat{O} v_o + \hat{\theta}_o \times \hat{O} k \times \hat{O} v_o) = \sum_{i=1}^{N} \hat{O} f_i$$  

(3)

$$I_o \hat{\ddot{\theta}_o} = \sum_{i=1}^{N} \hat{O} r_i \times \hat{O} f_i$$  

(4)

where $N$ represents the total number of agents that are involved in the task. Unless noted otherwise, throughout the rest of this article, all the vectors are defined in coordinate frame $\{O\}$. Thus, for the sake of brevity, the superscript $O$ is dropped from vector names in the following sections.

3. CONTROLLER DESIGN

This section starts with formulation of a centralized controller that uses the full Jacobian of the system and Moore-Penrose pseudoinverse [17] method to distribute a control function output between the agents of the group. The discussion is followed by reformulating the proposed central-
ized controller and converting it into a decentralized controller. Since the decentralized controller does not require any knowledge on the group population and formation, it does not need any information exchange between the members of the group. Note that the terms “pseudoinverse” and “Moore-Penrose pseudoinverse” are used interchangeably throughout the text.

### 3.1 Centralized Controller

To derive the centralized controller, the first step is to define the full Jacobian of the system. As explained in (5), the full Jacobian is formulated by defining the transformation that maps all the forces applied by all the agents into a resultant force and moment that is applied to the CoM of the object.

\[
F = J \begin{bmatrix} f_1^T & f_2^T & \cdots & f_N^T \end{bmatrix}^T. \tag{5}
\]

In the above equation, \( F = [F_x \quad F_y \quad M_z]^T \) is the vector of the resultant forces along the \( x \) and \( y \) axes and the moment about the \( z \) axis of the object coordinate frame, respectively. The matrix \( J \) is the full Jacobian of the system:

\[
J = \begin{bmatrix}
1 & 0 & 1 & 0 & \cdots & 1 & 0
\end{bmatrix}
\begin{bmatrix}
0 & 0 & 1 & 0 & \cdots & 0 & 1
\end{bmatrix}
= \begin{bmatrix}
1 & 0 & 1 & 0 & \cdots & 1 & 0
0 & 0 & 1 & 0 & \cdots & 0 & 1
\end{bmatrix}, \tag{6}
\]

where \( r_{xi} \) and \( r_{yi} \) are the \( x \) and \( y \) components of the vector \( r_i \), respectively. Since the size of \( J \) is \( 3 \times 2N \), the overall system can either be under-actuated (\( N = 1 \)) or over-actuated (\( \{N \in \mathbb{Z}|N \geq 2\} \)). To hold the focus of the article on multirobot and swarm scenarios, it is assumed that \( N \geq 2 \). Note that one can also derive the Jacobian by writing the differentials of \( r_i \) vectors with respect to time as a vector-valued function and taking partial derivatives of this function with respect to \( \dot{x}_o, \dot{y}_o \) and \( \dot{\theta}_o \).

The control objective is to define all the \( f_i \) values such that the norm of the error approaches to zero in finite time. This can be achieved by equating \( F \) to output of a control function \( \phi(e) \). Considering the dynamic behavior of the system and to simplify the overall formulation, a Proportional-Derivative (PD) controller is considered to serve as the control function \( \phi(e) \). Note that any linear or nonlinear control function can be utilized as \( \phi(e) \) based on the complexity of the object’s dynamics. Substituting a PD controller for \( \phi(e) \) yields:

\[
F = \phi(e) = K_p e + K_d \dot{e}. \tag{7}
\]

The next step is to convert the values obtained for \( F \) to the forces applied by each agent. Since \( J \) is not a square matrix, there is no unique solution for the system of equation. Thus, Moore-Penrose pseudoinverse is utilized to compute the minimum Euclidean norm solution for \( f_i \).

Doing so, the final centralized controller is formulated as:

\[
f = J^+ \phi(e) = J^+ (K_p e + K_d \dot{e}), \tag{8}
\]

where \( J^+ \) is the Moore-Penrose pseudoinverse of the Jacobian matrix \( J \).

### 3.2 Decentralized Controller

The fundamental idea for the formulation of the decentralized controller is that each agent only relies on a virtual agent that is located at an arbitrary location around the object. It is assumed that each agent only knows a minimal set of information: its point of attachment to the object, \( r_i \), and the error vector \( e \) [5]. Although the virtual agent \( i \), associated with \( i^{th} \) agent, does not necessarily represents the effect of the rest of the group, it allows the members of the group to define the necessary control forces to fulfill the manipulation task. As a result, each agent will have a locally defined Jacobian \( J_i \). The locations of the virtual agents need to be defined such that the pseudoinverse of the local Jacobian matrices yield to a valid solution, as discussed in detail in the following section. A possible location for the virtual agent \( i \) is on the mirror position of the location of the \( i^{th} \) agent with respect to the CoM of the object. The local Jacobian, \( J_i \), for each agent is defined as:

\[
J_i = \begin{bmatrix}
1 & 0 & 1 & 0 & \cdots & 1 & 0
0 & 0 & 1 & 0 & \cdots & 0 & 1
\end{bmatrix} \begin{bmatrix}
r_{xi} & -r_{yi}
-\frac{r_{yi}}{r_{xi}} & -a_{yi}
-\frac{a_{yi}}{a_{xi}} & a_{yi}
\end{bmatrix}, \tag{9}
\]

where \( J_i \) is the Jacobian of the system assumed by agent \( i \). Parameters \( a_{xi} \) and \( a_{yi} \) are the \( x \) and \( y \) components of the position of the \( i^{th} \) virtual agent. After defining all the local Jacobian matrices, the control formulation follows the discussion in Section 3.1. By applying a PD control function as \( \phi(e) \), the final control law for each agent is defined as:

\[
f_i = J_i^+ \phi(e) = J_i^+ (K_p e + K_d \dot{e}), \tag{10}
\]

where \( J_i^+ \) is the Moore-Penrose pseudoinverse of the local Jacobian matrix \( J_i \). A simulation demonstrating the proposed decentralized collective manipulation algorithm is shown in Fig. 3.

### 3.3 Stability and convergence

For clarity, the proof of the convergence and stability of the centralized controller is discussed first. Based on (5), the
total force applied to the CoM of the object is equal to the multiplication of the full Jacobian of the system to the forces applied by each agent. As described in Section 3.1, the pseudoinverse of the full Jacobian matrix is used to distribute the output of the control function \( \phi(e) \) between the agents. Thus, substituting (5) and (8) into (3) and (4) yields:

\[
[m_o(\dot{v}_o + \dot{\theta}_o \cdot \hat{k} \times v_o)]_{\theta_o} = J [J^+ \phi(e)]. \tag{11}
\]

Since the rows of the matrix \( J \) are linearly independent, the multiplication of \( J \) by its pseudo-inverse results in an identity matrix \( (JJ^+ = I) \). Thus, equation (11) reduces to:

\[
[m_o(\dot{v}_o + \dot{\theta}_o \cdot \hat{k} \times v_o)]_{\theta_o} = \phi(e). \tag{12}
\]

Thus, the convergence and stability of the system directly related to the definition of the control function \( \phi(e) \). If \( \phi(e) \) guarantees the stability of the object dynamics (as defined in (3) and (4)), the centralized controller will also be stable and its response will be the same as the response of the system that is directly controlled by \( \phi(e) \).

A similar approach can be followed to analyze the stability and convergence of the decentralized controller. Rewriting (12) for the decentralized controller yields:

\[
[m_o(\dot{v}_o + \dot{\theta}_o \cdot \hat{k} \times v_o)]_{\theta_o} = J \begin{bmatrix} K_1 \\ K_2 \\ \vdots \\ K_N \end{bmatrix} \phi(e), \tag{13}
\]

where \( K_i \) is equal to the rows 1 to 2 and columns 1 to 3 of \( J_i^+ \) and it is equal to:

\[
K_i = \frac{1}{2R_i} \begin{bmatrix} k_{i11} & k_{i12} & k_{i13} \\ k_{i21} & k_{i22} & k_{i23} \end{bmatrix}, \tag{14}
\]

where:

\[
k_{i11} = R_i + a_{yi}^2 - r_{yi}^2, \quad k_{i12} = -(a_{xi} + r_{xi})(a_{yi} - r_{yi}), \\
k_{i13} = 2(a_{yi} - r_{yi}), \quad k_{i21} = -(a_{xi} - r_{xi})(a_{yi} + r_{yi}), \\
k_{i22} = R_i + a_{xi} - r_{xi}^2, \quad k_{i23} = -2(a_{xi} - r_{xi}), \quad \text{and} \\
R_i = (a_{xi} - r_{xi})^2 + (a_{yi} - r_{yi}).
\]

Since \( [K_1 \ K_2 \ \cdots \ K_N]^T \) is not the pseudoinverse of \( J \), the right hand side of (13) will not reduce to \( \phi(e) \). Thus, the outputs of the control function \( \phi(e) \) are mapped to the forces applied to the CoM of the object by a resultant trans-

**Table 1: System parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Dimension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( m_o )</td>
<td>1</td>
<td>([Kg])</td>
<td>Mass</td>
</tr>
<tr>
<td>( I_o )</td>
<td>1</td>
<td>([Kg \cdot m^2])</td>
<td>Mass moment of inertia</td>
</tr>
<tr>
<td>( R )</td>
<td>0.2</td>
<td>([m])</td>
<td>Radius of the object</td>
</tr>
<tr>
<td>( K_p )</td>
<td>10</td>
<td>([N/m])</td>
<td>Proportional gain</td>
</tr>
<tr>
<td>( K_d )</td>
<td>5</td>
<td>([N/m])</td>
<td>Derivative gain</td>
</tr>
</tbody>
</table>

formation \( T \). In other words:

\[
F = J \begin{bmatrix} K_1 \\ K_2 \\ \vdots \\ K_N \end{bmatrix} \phi(e) = T\phi(e). \tag{15}
\]

As long as the transformation matrix \( T \) preserves the convergence and stability of \( \phi(e) \), the decentralized controller will be stable and convergent. A necessary, but not sufficient, condition on matrix \( T \) to preserve the behavior of \( \phi(e) \) is that all the eigenvalues of \( T \) must be positive. Based on the formulation presented in Section 3.2, the \( a_t \) vectors can have any arbitrary value, as long as they generate a valid \( T \) matrix. Here, to reduce the complexity of the system, it is assumed that a virtual agent for each agent \( i \) is located at its mirror position with respect to the CoM of the object. Thus:

\[
\begin{bmatrix} \delta N \end{bmatrix} = \begin{bmatrix} 1 & 0 & -r_{yi}/(r_{xi}^2 + r_{yi}^2) \\ 0 & 1 & r_{xi}/(r_{xi}^2 + r_{yi}^2) \end{bmatrix}, \tag{16}
\]

Note that the \( K \) matrix for a system composed of only two agents which are located on the mirror position of each other will be equal to the pseudo-inverse of the complete Jacobian of that system. Thus, in this specific case, the decentralized controller will behave similarly to the centralized controller. The final step is to substitute the results obtained in (16) into (15) and perform the multiplication to calculate \( T \). The transformation matrix \( T \) for a group of \( N \) agents is equal to:

\[
T = \frac{1}{2} \begin{bmatrix} N & 0 & -\sum_{i=1}^{N} r_{yi}/(r_{xi}^2 + r_{yi}^2) \\ 0 & N & \sum_{i=1}^{N} r_{xi}/(r_{xi}^2 + r_{yi}^2) \end{bmatrix} - \sum_{i=1}^{N} r_{yi}\sum_{i=1}^{N} r_{xi} \begin{bmatrix} N \\ N \end{bmatrix}, \tag{17}
\]

and the corresponding eigenvalues of the matrix \( T \) are:

\[
\lambda = \begin{bmatrix} N & N + \delta & N - \delta \end{bmatrix}, \tag{18}
\]

where \( \delta \) is defined as:

\[
\delta = \sqrt{\sum_{i=1}^{N} r_{yi}^2 + \sum_{i=1}^{N} r_{xi}^2 + \sum_{i=1}^{N} r_{yi} + \sum_{i=1}^{N} r_{xi}^2 + \sum_{i=1}^{N} r_{yi}^2}, \tag{19}
\]

All the eigenvalues of \( T \) are positive if and only if \(-N < \delta < N \). Since the perimeter of the object is finite and bounded, and the physical agents have a finite and nonzero perimeter, the growth in population of the group causes the matrix \( T \) to approach to a scaled identity matrix. As a result, all
the eigenvalues of $T$ remain greater than zero and a stable mapping is obtained between $\phi(e)$ and $F$.

4. RESULTS

The system responses to the proposed controllers are simulated by numerical integration of the nonlinear DEM using the Dormand-Prince method available in MATLAB. The numerical values for system parameters that are used for simulations are depicted in Table 1. Figure 3 illustrates snapshots of a simulation with the decentralized controller and 4 agents that are located at $0^\circ$, $90^\circ$, $100^\circ$, $220^\circ$ with respect to $O_x$ axis around the object. The desired position is defined to be 1 for all $x$, $y$ and $\theta$ values that are illustrated by black dashed lines in the figure. The color mapping is used to depict the passage of time in seconds. As illustrated in this figure, the decentralized controller can successfully manipulate the object to the desired position and orientation.

Several formations of the agents around the object are simulated to show the reliability and robustness of the proposed decentralized controller. The results are also compared with the centralized controller. As discussed in Section 3.3, as long as there is no limitation on the forces produced by each agent, the centralized controller will behave independent of the formation and population of the group. Fig. 5 shows the simulation results for different formations of the agents around the object that are controlled by the decentralized controller. As depicted in Fig.4, the agents are located around the object with $18^\circ$ incremental angles between them. The trajectories of the object and its orientations for the same formations are illustrated in Fig.6. In both Fig.5 and Fig.6, the response and the trajectory of the centralized controller are illustrated with black dashed lines. The color map in these two figures defines the population of the agents, $N$. As observed in the figures, As the population of the agents grows, the response of the decentralized controller approaches to the centralized one. This effect is also observable in (17).

To further analyze the behavior of the proposed controller, a series of simulations conducted with completely random formation of the agents around the object. The simulations started with a group of 4 agents and continued to a group of 20 agents. For each group, total of 20 uniformly distributed random formations are considered. For each simulation, the settling time of the response is calculated. Fig.7 illustrates the results of these simulations. The central red mark of each box represents the median of each settling time for each group population, the edges of the box are the 25th and 75th percentiles and the whiskers are extended to the most extreme data points without including outliers. The outliers are illustrated by red plus markers. As shown in the figure, as the population of the group gets larger, the number of outliers reduces. Also the extreme data points approach to the 25th and 75th percentiles. This indicates that the reachability and robustness of the system increases by the growth of the population of the group. The settling time is defined to be the time where the norm of error vector remains less than or equal to 0.01 (which is equivalent to 0.57% of the initial norm of the error vector).

5. CONCLUSIONS

This paper described formulation of a centralized and a decentralized controllers that are utilizing the jacobian of the system to distribute the output of a control function between the agents that are performing a collective manipulation. The proposed decentralized controller does not require any information about the formation and the population of the group. As a result, agents do not need to exchange any
6. REFERENCES


