machines

MDPI

Article

Scheme of Operation for Multi-Robot Systems with Decision-Making Based on Markov Chains for Manipulation by Caged Objects

Daniel Arreguín-Jasso [1,†], Anand Sanchez-Orta [1,*,†] and Hussain Alazki [2,†]

[1] Robotics and Advanced Manufacturing Division, Center for Research and Advanced Studies (Cinvestav), Saltillo 25000, Mexico
[2] Department of Mechatronics, Faculty of Engineering, Autonomous University Carmen, Ciudad del Carmen 24180, Mexico
* Correspondence: anand.sanchez@cinvestav.mx
† These authors contributed equally to this work.

Abstract: This paper presents the design of a new control scheme for a group of omnidirectional robots in a multi-robot system operating in an environment with obstacles. The control scheme uses a decision agent based on discrete-time Markov chains and takes into account the state of the system, obstacle positions, and geometries to manipulate targets, providing robustness against measurement uncertainties. The decision process is dynamic, with state information updating at each time step and tasks being executed based on the hierarchy determined by quadratic hierarchical programming. The system's stability in the mean-square sense is analyzed through the study of a closed-loop stochastic system, and the effectiveness of the proposed control scheme is demonstrated through numerical simulations, including a comparative analysis with a finite-state machine decision agent.

Keywords: mobile manipulation; discrete-time Markov chain; multi-robot systems; hierarchical quadratic programming

Citation: Arreguín-Jasso, D.; Sanchez-Orta, A.; Alazki, H. Scheme of Operation for Multi-Robot Systems with Decision-Making Based on Markov Chains for Manipulation by Caged Objects. *Machines* **2023**, *11*, 442. https://doi.org/10.3390/machines11040442

Academic Editor: Dan Zhang

Received: 10 February 2023
Revised: 18 March 2023
Accepted: 28 March 2023
Published: 31 March 2023

1. Introduction

The demand for advanced technology has increased due to the growing popularity of multi-robot systems in industry and warehouses for logistics tasks. These systems are essential for performing tasks that cannot be handled by a single robot, and are more cost-effective and durable than specialised robots [1]. This makes them ideal for use in flexible manufacturing cells or automated warehouses [2–4]. However, the current challenge is to effectively manipulate objects with mobile robots in large warehouses. Non-inertial multi-robot systems offer a solution to this problem, as they can operate in a much larger workspace than inertial robots. Among mobile robots, holonomic robots have proven to be effective in performing manipulation tasks in various applications, such as material handling, warehouse management, and assembly line production. These robots have high flexibility and manoeuvrability as they can move in any direction and orientation. However, the design and control of such robots can be challenging, requiring careful consideration of the robot's dynamics and control algorithms to ensure safe and effective manipulation of objects [5]. Despite this advantage, there is still a need to enable them to manipulate a wider range of object types with specific end effectors [6,7]. Further research is needed to improve the efficiency of these systems and to overcome any challenges that may arise during their deployment.

Manipulation by caging refers to a method of controlling groups of robots to interact with and manipulate objects in the environment [8]. This method is based on the idea of surrounding and holding an object with the action of multiple robots. This approach offers several advantages over traditional manipulation methods, such as increased stability,

robustness to external disturbances, and the ability to manipulate objects with complex shapes and textures. To achieve this type of manipulation, it is often necessary to break it down into simpler behaviours. These behaviours may include establishing and maintaining contact between the manipulators and the object, moving the object to a desired position, and releasing the object. In this paper, we focus on a manipulation scheme that is capable of manipulating regular cylindrical objects rather than those with complex shapes. Although complex figures may require more sophisticated manipulation techniques, regular shaped objects are more common in warehouse environments and can be manipulated efficiently using simple grasping and manipulation strategies. In addition, the perception and localization required for the manipulation task are simplified because regular objects are easier to recognize and track in the environment. Therefore, our decision to focus on regular shaped objects is motivated by the practical considerations of warehouse applications. Nevertheless, the proposed manipulation scheme can be extended to handle more complex shapes by incorporating additional perception and planning capabilities.

Transition processes are useful for coordinating cooperative behaviour in a multi-robot system. Specifically, the transition process is a sequence of specific behaviours, whose fulfilment in the correct order contributes to the performance of a specific task autonomously. In a multi-robot system, the autonomy of the transition process is attained by integrating a decision agent that manages the transition between the behaviours executed by each robot during the process [9,10]. In Ref. [11], a solution is proposed to integrate a transition agent using flags to change the system behaviour. This behaviour consists of task sequences, moving groups of robots between work areas, but relies heavily on sensor precision due to error dependence [12–14]. Additionally, the described solution increases the complexity of working with groups of tasks. From here, the problem of the phase transition arises, which has been worked on in Ref. [15], where a kinematic controller capable of handling several conflicting tasks in a multi-robot system is proposed; this proposal considers the performance of both cooperative tasks as well as individual tasks. In Ref. [16], a task set is proposed for object handling and transportation. The method utilizes internal distances between robots to form relatively rigid formations, ignoring clamping methods. The transition process is managed by a state machine, the transition agent, which changes the state based on a metric determined by the designer. In consequence, the system's reliability depends on the operator's knowledge and the sensitivity of the sensors. Regarding the task transition planning, the authors in Ref. [17] focus on the task planning aspect of home service robotics, where sequences of actions are generated to complete tasks based on high-level and low-level actions. However, unlike classical planning methods, task planning in the home environment is fraught with uncertainty and change, which requires consideration of human–robot interactions. In addition, this paper discusses the challenges and current approaches to accomplish the task under uncertain and incomplete information. In Ref. [18], a dynamic control for mobile robots with an integrated manipulator is presented. This work proposes a robust controller that ensures finite-time convergence of the error in the presence of parametric uncertainties. The work integrates a finite-state machine (FSM) with transitions of predetermined lapses. This type of FSM works only under the premise that the predetermined transition time is greater than the convergence time of the system.

Our proposed scheme focuses on transitions as events in a probability space and utilizes measurements to adjust the probability measure of each event to formulate a stochastic control system. Stochastic systems are well-suited for scenarios where the phenomenon under investigation is influenced by unpredictable factors. They have been applied in the field of multi-robot systems, resulting in innovative navigation approaches in stochastic environments (such as wind currents and flexible patterns) [19,20]. In Ref. [21], the authors analyse network applications and information processing, where access to information depends heavily on human interaction. To enhance this, a medium access control based on stochastic models is proposed to improve the management of information flow. In Ref. [22], a Markov chain-based methodology is proposed to define system behaviour subject to random forces. Additionally, in

Ref. [23], the authors present a control design for stochastic perturbations using observer-based output feedback with a Markov chain and the invariant ellipsoid approach.

Following this line of research, it is necessary to find useful qualities to analytically evaluate a stochastic system. In our proposal, stability in the mean-square sense was adopted. This stability notion is a deeply studied quality of the behaviour of stochastic differential systems, which is used to demonstrate the desired properties of such systems [24,25]. In Ref. [26], the mean square lemma is analysed. It shows, through system causality, that having a bound on the mathematical expectation of the state, output, or stochastic phenomenon is sufficient to prove mean-square stability of the entire closed-loop system [27].

The main contribution of this paper is the development of a novel decision agent, represented by a discrete-time Markov chain (DTMC), which ensures the stability of a comprehensive manipulation scheme. This scheme uses information from the environment and a group of robots to execute a set of behaviours that achieve the goals of the manipulation task [28]. Furthermore, our proposal is robust to uncertainties that may occur during the manipulation, keeping the desired behaviour according only to the actual conditions of the task, which makes it more reliable and applicable to real-world scenarios. Additionally, a stability analysis of the manipulation scheme is presented, demonstrating that the mathematical distribution of the DTMC is dependent on the error of the robot task and the resulting closed-loop system is stable in a mean-square sense, thereby confirming the stability of the manipulation scheme.

The paper is organized as follows. Section 2 presents the methodology of the proposed manipulation scheme. Section 3 describes the equations of motion of the multi-robot system. Section 4 details the tasks that the multi-robot system can perform. Then, Section 5 shows how to deal with tasks simultaneously with hierarchical quadratic programming (HQP). Section 6 explains how the tasks are grouped in a space defined as "action phases". Section 7 describes the construction of a Markov chain with a state-measure-dependent distribution. Section 8 shows simulations that support the operation of the complete scheme, demonstrating that the proposed DTMC decision agent provides significant improvement in performance when compared with a FSM agent. Finally, the conclusions of the work and future work based on the results of this research are presented in Section 9.

2. Manipulation Scheme

The methodology involved in constructing the manipulation scheme consists of six process blocks: Robot, Environment, Tasks, Phases, DTMC, and finally HQP. The blocks are connected as shown in Figure 1. In the Robot block, the dynamic model of the multi-robot system is found, from which the state of all the agents in the system is obtained. In the environment block all the information of the workspace and the assignment of objectives of the scheme are defined, such as the position and size of the obstacles, the selection of the objective, and the desired position of the objective. The Tasks block obtains the values of the tasks available to the multi-robot system in the form of errors, whose minimization represents the fulfilment of the task. The tasks are computed with the information of the environment and robots, which includes both cooperative and individual tasks. In the Phases block, tasks are grouped into distinct phases, defining the necessary behaviour for the robot to perform the target manipulation process. The DTMC process block has a transition agent, represented by a DTMC, which selects a Phase based on the completion of the previous one, as it uses a state-measured probability distribution. Finally, the HQP block generates a control input to the multi-robot system that seeks the fulfillment of the tasks contained in the selected phase simultaneously.

In the context of modelling our decision agent, it is important to choose a suitable mathematical model that can maintain consistent performance despite potential uncertainties. While conventional DTMC and FSM are common mathematical models for discrete systems, modelling decision agents requires selecting a model that can effectively address robustness to unexpected events and consider its limitations. Our focus is on DTMC as it has been shown to be effective in handling uncertainties and randomness [23,29]. On

the other hand, FSM, which relies on deterministic rules, is limited in its ability to model probabilistic behaviour [15,18], and constructing an accurate FSM-based decision agent can be complex and requires specialized knowledge of the system. Despite this, the proposed manipulation scheme has the disadvantage of requiring a simultaneous transition of the behaviour of each of its agents. Therefore, without reconsidering a more specialised Markov chain, it is currently not possible to implement it in a decentralised manipulation scheme.

We implement this scheme in a simulation study by considering a group of Festo Robotinos. We chose these robots since they represent a versatile and powerful mobile robot platform suitable for a wide range of applications, including material handling, warehouse management, and research [30]. The Robotino's omniwheels provide smooth and accurate motion, enabling precise positioning for object manipulation.

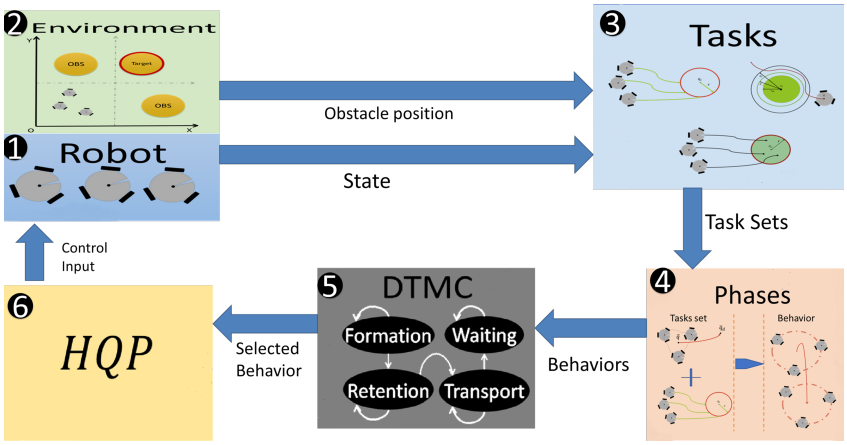

Figure 1. The Manipulation scheme involves six interconnected blocks: 1. Robots, representing the omnidirectional robots; 2. Environment, the operational space for the robots; 3. Tasks, the specific goals to be achieved by the robots; 4. Phases, the stages involved in the manipulation scheme; 5. DTMC, which utilizes a discrete-time Markov chain; and 6. HQP, which computes the control of the robots to carry out simultaneous tasks.

3. Dynamic Model

For implementing the scheme, the Robot block represents a group of n omnidirectional robots, with its dynamic model. The notation used in the scheme is presented in Table 1. The position and orientation of this group of robots is represented by $\xi = \begin{bmatrix} q_1^T & \cdots & q_n^T \end{bmatrix}^T$, where $q_i = \begin{bmatrix} x_i, y_i, \theta_i \end{bmatrix}^T$ is the position and orientation of the mobile frame p_i attached to the centre of the i-th robot, with respect to the inertial frame O. Let us define the rotation matrix $R_p^o(\theta)$ as

$$R_p^o(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

then, for a total representation of the system, it is defined

$$\mathcal{R}_p^o = \text{blkdiag}\left[R_p^o(\theta_1), \cdots, R_p^o(\theta_n) \right] \tag{2}$$

which is the block-diagonal matrix that relates the frame P to the frame O. To determine the velocities of each of the n robots in the P frame, we introduce the space $\eta = \begin{bmatrix} \eta_1^T, \cdots, \eta_n^T \end{bmatrix}^T \in \mathbb{R}^{3n}$ with $\eta_i \in \mathbb{R}^3$ defined as

$$\eta_i = R_p^o(\theta_i)^T \dot{q}_i = \begin{bmatrix} \dot{x}_i & \dot{y}_i & \omega_i \end{bmatrix} \tag{3}$$

where $R_p^o(\theta_i)^T$ is the transpose of the rotation matrix from frame P to O at orientation θ_i, and $\dot{q}_i$ is the velocity of the i-th robot in the O frame.

These robots have an omnidirectional configuration that is achieved with three holonomic wheels coupled equidistant around the main body [5]. In order to better understand the mathematical representation of the robot, the variables used in (4) are described in Table 1. The robot seen in Figure 2 is represented mathematically by

$$\begin{aligned} \bar{M}_i \dot{\eta}_i + h_i &= \tau_i \\ \bar{M}_i &= R_p^o(\theta_i) M_i R_o^p(\theta_i) + E^T M_{\phi i} E \\ h_i &= R_p^o(\theta_i) M_i \dot{R}_p^o(\theta_i) \eta_i \\ \tau_i &= E^T \tau_{\phi i} \end{aligned} \tag{4}$$

where the matrix E encodes the wheel configuration and represents the holonomic wheel projection as

$$E^{-1} = \begin{bmatrix} 0 & \frac{r_2\sqrt{3}}{3} & \frac{r_3\sqrt{3}}{3} \\ \frac{2r_1}{3} & \frac{r_2}{3} & \frac{r_3}{3} \\ \frac{r_1}{3b} & \frac{r_2}{3b} & \frac{r_3}{3b} \end{bmatrix} \tag{5}$$

where b is the distance between the robot centroid and the wheel, and r_i is the radius of the wheel i.

Table 1. Common variables used.

O	Inertial Frame
P	Referential in the robot C.o.M.
n	number of robots
ξ	robot extended state
q_i	i-robot state
x_i	i-robot position along x-axis
y_i	i-robot position along y-axis
θ_i	i-robot orientation along z-axis
$R_Q^P(\theta)$	Rotation Matrix from Frame O to P
$\mathcal{R}_Q^P(\theta)$	Extended Rotation Matrix
M_i	i-robot inertia Matrix
$M_{\phi i}$	i-robot wheels inertia Matrix
h_i	non-linear Vector from i-robot
τ_i	i-robot control input
$\tau_{\phi i}$	i-robot control input for each wheel
M	Extended inertia Matrix
h	non-linear extended vector from i-robot
e_Q	Q-task error
J_Q	Q-task Jacobian
$\dot{J}_Q$	Time derivative from J_Q
u_Q	Q-task auxiliary control
$\mathbb{S}$	Behaviour state space
$\mathbb{S}_j$	j-state of behaviour
α_j	j-state distribution

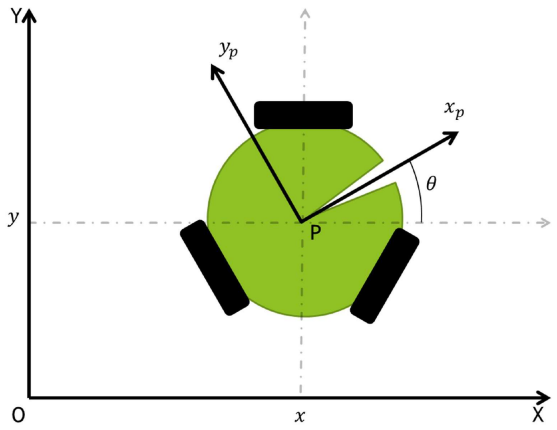

Figure 2. Omni-wheeled robot representation.

A compound system is defined to represent the dynamic behaviour of a group of robots,

$$\begin{aligned} M\dot{\eta} + h &= \tau \\ M &= \text{blkdiag}[\bar{M}_1, \cdots, \bar{M}_n] \\ h &= [h_i^T, \cdots, h_n^T]^T \\ \tau &= [\tau_i^T, \cdots, \tau_n^T]^T \end{aligned} \tag{6}$$

4. Multi-Robot Tasks

Manipulation by caging is a complex task that requires coordination and precision among various simpler tasks. To perform this type of manipulation, a set of individual tasks is designed and executed in a specific sequence. These tasks are often geometric or kinematic in nature and are used to guide and control the motion of objects towards a desired location. The success of manipulation by caging is dependent on the proper execution of these smaller tasks, which can include tasks such as grasping, positioning, and releasing. The combination of these individual tasks results in a smooth and accurate final outcome. In accordance with (6), the presence of obstacles in the workspace is taken into account to formulate the available tasks for the robot to perform the manipulation. These tasks are modelled through the application of inequality or equality constraints in a quadratic programming framework. Each task is characterized by an error function, the optimization of which serves to minimize the error and thereby indicate successful task completion.

4.1. Equality Tasks

We consider equality tasks to be those tasks that are active at all times. We refer to regulation and geometric formation tasks. Tasks can be assigned to individual agents or multiple agents working in coordination. The equality tasks used here are described below.

4.1.1. Geometric Shape Formation

A geometric shape formation task, such as maintaining a circular formation with vehicles, is demonstrated in Figure 3. In this particular case, the task limits the feasible displacements of each robot over the circumference.

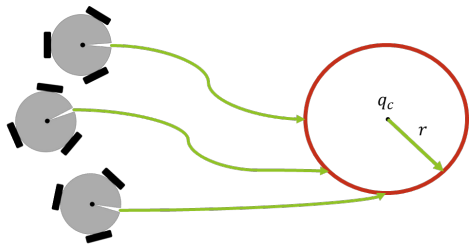

Figure 3. Geometric shape formation.

This is a local task function because each vehicle reaches the perimeter of a given circumference without the need to know the location of other team members. The error function of the task is defined as

$$e_{ci} = \frac{1}{2}(q_i - q_c)^T(q_i - q_c) - r^2 \in \mathbb{R}^3 \tag{7}$$

where r is the radius and $q_c = [x_c, y_c]^T \in \mathbb{R}^2$ is the centre of the circle. The circular task error function is described as $e_c = [e_{c1}, e_{c2}, ..., e_{cn}]^T \in \mathbb{R}^n$, where its time-derivative is $\dot{e}_c = J_c\dot{\xi}$. The task Jacobian $J_c = \frac{\partial e}{\partial \xi} \in \mathbb{R}^{n\times 3n}$ is calculated as:

$$J_c = \text{blkdiag}\Big[\Big[(q_1 - q_c)^T, 0\Big] \cdots \Big[(q_n - q_c)^T, 0\Big]\Big] \tag{8}$$

and the double time derivative is computed as $\ddot{e}_c = J_c\dot{\eta} + \dot{J}_c\eta$ where $\dot{J}_c$ is defined as:

$$\dot{J}_c = \text{blkdiag}\left[\left[\dot{q}_1^T, 0\right] \cdots \left[\dot{q}_n^T, 0\right]\right] \tag{9}$$

4.1.2. Regulation Task

Individual

In order to reach an individual target, as can be seen in Figure 4, the error function is defined as

$$e_{ri} = \xi - \xi_d \in \mathbb{R}^{3n} \tag{10}$$

where ξ_d is the desired state of the system, which also implies a desired position and orientation for each of the robots. The time derivative of (10) is $\dot{e}_{ri} = \dot{\xi} - \dot{\xi}_d$, with $\dot{\xi}_d = 0$ since it is a regulation task. Then, the task Jacobian becomes

$$J_{ri} = I_n \in \mathbb{R}^{n\times n} \tag{11}$$

Since J_{ri} is constant, its time derivative becomes $\dot{J}_{ri} = 0$.

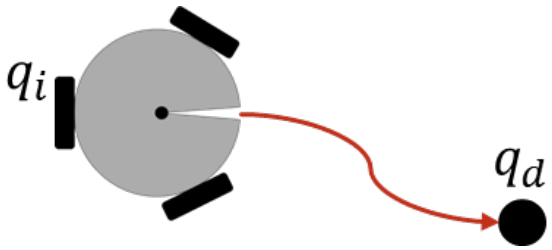

Figure 4. Individual regulation task.

Cooperative

This task is designed to reach a target as a group while the vehicles can keep a desired formation, see Figure 5. It consists in regulating the centroid of the formation towards a desired object. The centroid of the formation is $\bar{q} = \begin{bmatrix}\bar{x} & \bar{y}\end{bmatrix}^T \in \mathbb{R}^3$, where

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i, \quad \text{and} \quad \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{12}$$

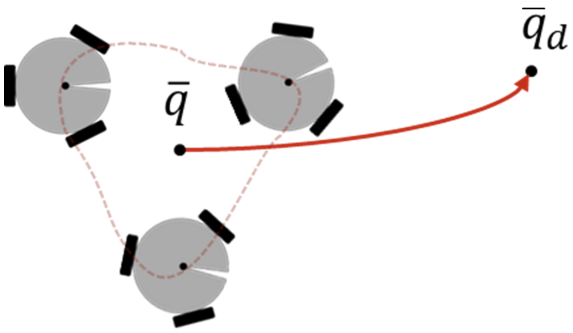

Figure 5. Cooperative regulation task.

Then, the error function becomes

$$e_r = \bar{q} - \bar{q}_d \in \mathbb{R}^2 \tag{13}$$

It is straightforward to adapt the task error for tracking purposes if the target is a time-varying function: $\dot{e}_r = J_r\dot{\xi} - \dot{\hat{q}}_d$, where the task Jacobian of n agents is given by

$$\begin{aligned} J_r &= \begin{bmatrix} J_{ti} & J_{t[i+1]} & \cdots & J_{tn} \end{bmatrix} \in \mathbb{R}^{2\times 3n} \\ J_{ti} &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \in \mathbb{R}^{2\times 3} \quad \forall \quad i \in \{1, ..., n\} \end{aligned} \tag{14}$$

Since J_r is constant, its time derivative becomes $\dot{J}_{tn} = 0$.

4.2. Inequality Tasks

The proposed function to control the inequality tasks is a combination of an inequality and an activation function. The activation function, $f_{ac}(\xi)$, determines if the task is activated, based on the distance between the i-th robot and the target (obstacle, robot, or formation), d_i. The activation function has a range of $[0, 1]$ and is defined as:

$$f_{ac}(\xi) = \frac{1}{2} + \frac{1}{2}\tanh(\gamma(d_i - d_f)) \tag{15}$$

where γ is a weight, d_f is the activation distance, and d_i is the distance between the ith robot and the target. The function has a smooth transition and is used for obstacle avoidance, collision avoidance, and geometric permissible region, where the value of γ is negative.

4.3. Obstacle Avoidance

Avoidance tasks are at the top of the hierarchy of task stacks. They are essential for proper navigation within a workspace populated by inert objects and robots. They prevent robots from being damaged by accidental encounters in their defined tasks. In addition, geometric formation tasks can be constructed with less processing, since it is not necessary to precisely calculate the position of each robot, but it is enough to define the centroid of the formation.

When the robot approaches the obstacles, the corresponding inequality constraint is activated, as shown in Figure 6. In particular, each obstacle is embedded within a spherical shell of radii r_m and d_M, with $r_m < d_M$. Thus, the obstacle avoidance task is defined as

$$e_{oa} = \begin{bmatrix} d_{i,j} - dM \\ d_{i,j+1} - dM \\ \vdots \\ di, k - dM \end{bmatrix} \in \mathbb{R}^k \tag{16}$$

where $d_{i,j} = \| C_i - C_j \|$ is the distance from a centre point of the i-th robot $C_i(\xi) \in \mathbb{R}^2$, to the nearest point over the j-th obstacle $C_j \in \mathbb{R}^2$. Assuming that C_j is constant, the time derivative of (16) becomes $\dot{e}_{oa} = J_{oa}\dot{\xi}$ where the task Jacobian is given by

$$J_{oa} = L_i^T J_{Ci}\dot{\xi} \in \mathbb{R}^k \tag{17}$$

where $J_{C_i} = \frac{\partial C_i}{\partial \xi} \in \mathbb{R}^{2\times n}$ is the linear velocity Jacobian and $L_i = \begin{bmatrix} l_{i,j} l_{i,j+1} ... l_{i,k} \end{bmatrix} \in \mathbb{R}^{2\times k}$ with

$$l_{i,j} = \frac{C_i - C_j}{\| C_i - C_j \|} \in \mathbb{R}^2 \tag{18}$$

also, the time derivative of (18) is:

$$\dot{l}_{i,j} = l_{i,j}^T J_{C_i}\dot{\xi}\frac{C_i - C_j}{\| C_i - C_j \|^2} - J_{C_i}\dot{\xi}\frac{\| C_i - C_j \|}{\| C_i - C_j \|^2} \in \mathbb{R}^2 \tag{19}$$

which is needed to construct $\dot{L}_i = [\dot{l}_{i,j}\dot{l}_{i,j+1}, \cdots, \dot{l}_{i,k}] \in \mathbb{R}^{2\times k}$ in order to compute the double time derivative of (16), $\ddot{e}_{oa} = J_{oa}\ddot{\xi} + \dot{J}_{oa}\dot{\xi}$ where the time derivative Jacobian becomes

$$\dot{J}_{oa} = L_i^T \dot{J}_{C_i} + \dot{L}_i^T J_{C_i} \tag{20}$$

since J_{Ci} is constant, its time derivative becomes $\dot{J}_{Ci} = 0$.

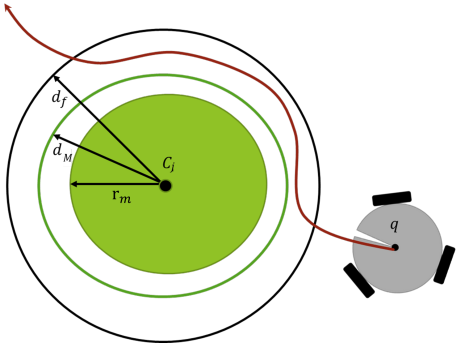

Figure 6. Obstacle avoidance task.

4.4. Collision Avoidance

Because the group of robots shares the same workspace, it is necessary to consider the evasion of the robots among themselves, as can be seen in Figure 7.

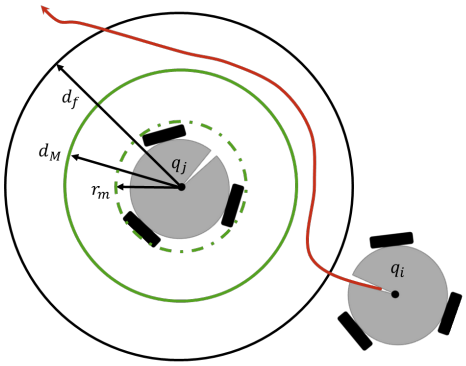

Figure 7. Collision avoidance task.

The collision avoidance task, e_{om}, is obtained in the same way as the obstacle avoidance, with the difference that C_j is considered as the position of the robot j, which implies that J_{C_i} becomes

$$J_{C_i} = \frac{\partial C_i}{\partial \xi} - \frac{\partial C_j}{\partial \xi} \in \mathbb{R}^{2\times 3n} \tag{21}$$

If $C_i = [x_i, y_i]$ and $C_j = [x_j, y_j]$ then J_{Ci} has the form

$$\begin{aligned} J_{Ci} &= \begin{bmatrix} J_{C1} \cdots & J_{Ck} & \cdots & J_{Cn} \end{bmatrix} \\ J_{Ck} &= \begin{cases} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} & : k = i \\ \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} & : k = j \\ 0_{2\times 3} & : \text{otherwise} \end{cases} \end{aligned} \tag{22}$$

Since $\dot{J}_{Ci}$ is constant, its time derivative becomes $\dot{J}_{Ci} = 0$.

4.5. Geometric Permissible Region

In this task, the allowable navigation area represents the interior of a circle, as shown in Figure 8, where the robots must remain. Then, the structure of the task error and its Jacobian are the same as in (7)–(9). However, we will consider it as an inequality task:

$$e_{pi} = e_c f_{ac}(\xi) \tag{23}$$

$$J_{pi} = J_c f_{ac}(\xi) \tag{24}$$

$$\dot{J}_{pi} = \dot{J}_c \tag{25}$$

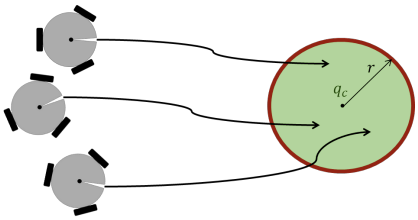

Figure 8. Geometric permissible region.

5. Hierarchical Quadratic Programming

HQP is capable of handling multiple hierarchical objectives while ensuring that kinematic and dynamic constraints are satisfied, and provides a computationally efficient solution by decomposing the problem into a hierarchy of sub-problems and solving them iteratively. This allows us to ensure that the robot's motion satisfies all the constraints, while dealing with multiple objectives at different levels of the hierarchy.

A group of the tasks described in Section 4 can be performed simultaneously by the multi-robot system in each of the phases of the scheme, so HQP is used to find a control input τ, applied in (6), which fulfils all the tasks included in a phase. The main characteristic of HQP is that lower priority tasks cannot affect higher priority tasks by solving a quadratic programming cascade with slack variables. The main idea is to find a control signal that obeys a set of p tasks to be executed simultaneously, as shown in the Algorithm 1.

Consider the double integrator system $\ddot{e} = u(t)$ and $u(t)$ as a PD control law $u(t) = -K_d\dot{e} - K_p e$, which is needed for HQP of the tasks defined in Section 4 [18]. The tasks are considered an error function of the form

$$e = f(\xi) - f^d \in \mathbb{R}^m \tag{26}$$

We assume that (26) is twice differentiable with respect to time

$$\begin{aligned} \dot{e} &= \dot{f} = \bar{J} R_p^o(\theta)\eta = J\eta \\ \ddot{e} &= J\dot{\eta} + \dot{J}\eta \end{aligned} \tag{27}$$

From (6) and (27), we obtain

$$\ddot{e} = Q\tau + \mu \tag{28}$$

where $Q = JM^{-1} \in \mathbb{R}^{m\times 3n}$ and $\mu = -Qh + \dot{J}\eta$ is the task's drift. The task-based inverse dynamic is obtained by solving for τ in (28) as follows:

$$\begin{aligned} \tau &= Q^{\#M}(u - \mu) \in \mathbb{R}^{3n} \\ Q^{\#M} &= MQ^T[QMQ^T]^{-1} = J^T[JM^{-1}J^T]^{-1} \in \mathbb{R}^{n\times m} \end{aligned} \tag{29}$$

$Q^{\#M}$ is the weighted generalized inversion of JM^{-1}, and u is the auxiliary vector of control inputs. To overcome possible conflicts among the tasks, the hierarchy between them is imposed such that (29) becomes:

$$\begin{aligned} \tau &= \sum_{i=1}^{p} \tau_i \in \mathbb{R}^{3n} \\ \tau_i &= \bar{Q}_i^{\#M} \left(u_i + \mu_i - Q_i \sum_{k=0}^{i-1} \tau_k \right) \in \mathbb{R}^{3n} \\ N_i &= N_{i-1} - \bar{Q}_i^{\#M} \bar{Q}_i \in \mathbb{R}^{3n \times 3n} \end{aligned} \tag{30}$$

where $\bar{Q}_i = Q_i N_{i-1} \in \mathbb{R}^{m_i \times 3n}$, $\bar{Q}_i^{\#M} = M\bar{Q}_i^T \left[\bar{Q}_i M \bar{Q}_i^T \right]^{-1} \in \mathbb{R}^{3n \times m_i}$, $\tau_0 = 0$ and $N_0 = I_n$. Notice that N_i belongs to the null space of $\bar{Q}_i$.

Algorithm 1 Task phase

Require: State of the group $(\xi, \dot{\xi})$, Task phase $\mathbb{S}$, δt
Ensure: Control input τ
Get $\mathcal{R}_p^o$ with (2)
$\eta = \mathcal{R}_p^o \dot{\xi}$
Get M, h with (6)
Indicate the order of the tasks: $\mathbb{S} = [e_{oa}, e_{ci}, \cdots]$
Get task Jacobians: $J_s = [J_{oa}, J_{ci}, \cdots]$
Get time derivative task Jacobians: $\dot{J}_s = [\dot{J}_{oa}, \dot{J}_{ci}, \cdots]$
Get τ with (30)
Get $\dot{\eta}$ with (6)

6. Action Phases

Each phase requires specific behaviour in the robots. Depending on the task, the robots can work as a team or individually. However, they are always reacting to their environment for obstacle and collision avoidance tasks.

The action phase $\mathbb{S}_i$ is a set of tasks introduced in the HQP block to be fulfilled in a defined hierarchy. Each Action phase considers the sets of task errors, task projector Jacobians, the time derivative from the projector Jacobians, and a set of auxiliary control inputs that asymptotically minimize each error of the Action Phase. The i phase of action is arranged in the group $\mathbb{S}$. For the purpose of this paper, four phases of action are considered, presented in Table 2,

$$\mathbb{S}_i = \left\{ \{ e_{ni}, J_{e_{ni}}, \dot{J}_{e_{ni}}, u_{ni} \} \in \mathbb{S} \right\} \tag{31}$$

The evaluation of whether tasks are fulfilled is measured through the error generated by each task. A task is considered accomplished when its error remains in a neighbourhood close to zero. Likewise, a phase is considered to be reached when the errors of all its tasks are fulfilled. It is considered a state composed of the errors generated in the phase,

$$e_{\mathbb{S}_i} = [e_{ni} \in \mathbb{S}_i] \tag{32}$$

This state exists in a normed space of the phase whose vector norm provides us with substantial information about the development of the task at each time step. A norm $\| e_{\mathbb{S}_i} \|_2$ is proposed in order to have an analytical perception of the behaviour of the vector $e_{\mathbb{S}_i}$ in a scalar way, even when the dimension of the vector changes between the existing phases in the totality of the process.

Table 2. Tasks to accomplish in each phase.

Phase	Task	Error	Hierarchy
Phase 1 Formation	Obstacle avoidance	e_{oa}	1
	Collision avoidance	e_{om}	2
	Geometric formation	e_{ci}	3
Phase 2 Grip	Obstacle avoidance	e_{oa}	1
	Collision avoidance (equidistant)	e_{om}	2
	Geometric formation	e_{ci}	3
Phase 3 Manipulation	Obstacle avoidance	e_{oa}	1
	Collision avoidance	e_{om}	2
	Permissible space	e_{pi}	3
	Cooperative regulation	e_r	4
Phase 4 Waiting	Obstacle avoidance	e_{oa}	1
	Collision avoidance	e_{om}	2
	Individual regulation	e_{ri}	3

7. Discrete-Time Markov Decision Agent

Definition 1 (Probabilistic space). *Let Ω be a space with a family of subsets $\mathbb{F}$ (in this paper it is considered as the set $\mathbb{S}$), such that any subset $A_i \in \mathbb{F}$ is closed under accounting complements, unions, and intersections. A family of subsets possessing these properties is known as a $\sigma-$algebra. If a probability measure P is defined on the σ-algebra $\mathbb{F}$, then the triple $(\Omega, \mathbb{F}, P)$ is called a Probability space and the sets in $\mathbb{F}$ are called random events.*

Definition 2 (Markov condition). *Consider a probability space from Definition 1 $(\Omega, \mathbb{F}, P)$. Let E be a finite or countable set of states and $\{X_n : \Omega \to E, n \in \mathbb{N}\}$ a sequence of random variables. The process is said to satisfy the Markov condition if for all integer $n \geq 0$ and all $j, i_n, \ldots, i_0 = X_0$ that belongs to E, it holds:*

$$\begin{aligned} \mathbb{P}(X_{n+1} = j \mid X_n = i_n, ..., X_0 = i_0) \\ = \mathbb{P}(X_{n+1} = j \mid X_n = i_n) \end{aligned} \tag{33}$$

Definition 3 (Distribution by state measure). *Let E be a countable set. Each $X \in E$ is called a state and I is called a state space. We say that a subset $\alpha = \{\alpha_X : X \in E\}$, where $\alpha_X \in [0, 1]$, is a measure on E $\forall$ $X \in E$. We work with a probability space $(\Omega, \mathbb{F}, P)$ from Definition 1, a random event Ξ that takes values of $\mathbb{F}$ and a random event $\{W : \Omega \to E\}$ with values in E. Then, it is said that α defines a distribution by state measure if it satisfies the condition:*

$$\mathbb{P}(\Xi = \mathbb{S}_X) = \mathbb{P}(\{\alpha_X : W(\alpha_t) = X\}) \tag{34}$$

Definition 4 (Mean-square stability). *Let $X = X_i : i \in \mathbb{N}$, if for every $\epsilon > 0$, there exist $\delta > 0$ such that $\mathbb{E}[X_i^2] < \delta$ for all $i \in \mathbb{N}$ whenever $\mathbb{E}[X_0^2] < \epsilon$ then it is said that X is stable in mean-square sense [31].*

The Markov chain is a discrete-time stochastic process in which a random variable that takes values of $\mathbb{S}$ changes over time. Its main characteristic is that it complies with the Definition 2 [22]. The measurement of the phase for the system comes from the error vector of (32). Likewise, the norm $\|e_{\mathbb{S}_i}\|_2$ is proposed as a decision agent to affect the transition matrix. A bounded exponential function is generated to provide information about the tasks in each phase and ensure that the task has been completed before each transition. This function depends on $\| e_{\mathbb{S}_i} \|_2$,

$$\alpha_i = f(e_{\mathbb{S}_i}) = 1 - e^{-\|e_{\mathbb{S}_i}\|_2} \tag{35}$$

Notice the non-negative nature of the vector norm and the exponential envelope of the function $f(e_{\mathbb{S}_i}) \rightarrow [0,1]$. A new transition matrix is defined, where the probability of task development is based on relevant state measurement information for each element,

$$P(f_i) = \frac{p_{Ti}\alpha_i}{\mu_T} \tag{36}$$

where $\mu_n = \sum_{i=1}^{m} p_{Ti}\alpha_i$, p_{Ti} is the nominal probability, m is the number of states in the system, and T is the current instant of the system. This obeys the requirement of a transition matrix where the sum of the elements of each row is 1. After finding α_i, it is balanced with the complement probability, forcing $P(\mathbb{S}) = 1$ with (36), this α_i of State 1 is considerably small, the measurement α_j of State 2 is the complement and therefore takes higher values.

Theorem 1. *Let a decision agent be represented by a probabilistic space* $(\Omega, \mathbb{S}, P(\alpha))$ *and fulfil the Markov condition with distribution by state measure as Definition 3. Then* $(\Omega, \mathbb{S}, P(\alpha))$ *is stable in mean-square sense according to Definition 4.*

Sketch of the Proof. Once the decision agent, represented by a Markov chain, selects a phase $\mathbb{S}_i \in \mathbb{S}$, we obtain an error e_{ni} and therefore the computation of a control u_{ni}, which is applied to the robot to minimize that error. Since α_i measures the phase error e_{ni}, it will change as the error converges to zero. Given the α_i from (35), there is a direct proportion relationship with e_{ni}, if the error takes values close to 0 then α in phase $\mathbb{S}_i$ is close to 0, ensuring the existence of δ. Since there are a limited number of transitions from the current state, according to Definition 3, ϵ exists in every moment and implies that ϵ is less than or equal to the number of possible transitions that exist in each phase. Then ϵ and δ from Definition 4 always exist, which implies that the proposed decision agent is stable in the least-squares sense. □

The manipulation scheme is considered a stochastic system when it operates in a closed loop since the random variable of the decision agent is being fed back. Therefore, it is important to demonstrate the stability of the system with stochastic tools. The stability presented in the decision agent ensures that the stability in the mean-squares sense is preserved in the rest of the processes thanks to the causality of the events in the whole manipulation scheme.

8. Simulation Results

To test the validity of the proposed manipulation scheme, which is described in Figure 9, numerical simulation results are presented. The simulator was programmed in Python3, on a computer equipped with an Intel® Core™ i7-8565U CPU @ 1.80 GHz and 8 GB RAM. The Markov chain's transition probability is set at a 0.01 second sampling time, affecting the duration of each phase. Each phase is expected to last 1500 steps with this transition probability design. This means that the probability of staying in phase $\mathbb{S}_i$ is $p_{ni} = 1499/1500 = 0.9993$, which implies that the initial transition probability is $p_{[n+1]i} = 1 - 0.9993 = 0.0007$. Wheeled robots with a mass of m = 1 kg, an inertia tensor of $I_z = 0.0083$ kgm, as well as an inertia tensor of the wheels of the wheeled robot of $I_\phi = 0.01$ kgm, and a radius of $r = 0.20$ m, are considered. There are obstacles at $C_1 = [5, 15]m, C_2 = [20, 0]m$ and $C_3 = [20, 15]m$ with a radius of $r_o = 3$ m.

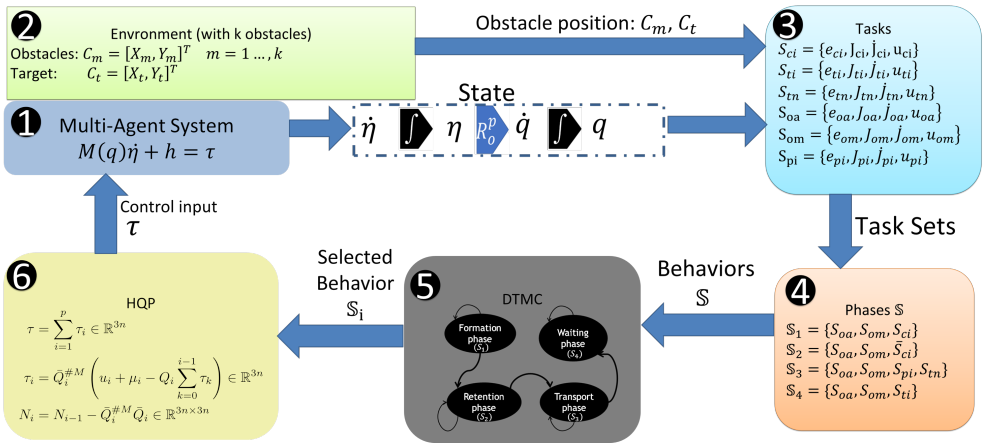

Figure 9. Process blocks of the Manipulation scheme. 1. Robots, given by (6); 2. Environment; 3. Tasks, given from (7) to (25); 4. Phases, presented in Table 2; 5. DTMC, whose probability is given by (36); and finally 6. HQP, given by (30).

8.1. Phase 1: Formation Phase

The robots are directed from their starting position to the border with the target (e_{ci}), where strict avoidance of the obstacles and collision between them is important (e_{oa}). Therefore the task Phase is designed as

$$\mathbb{S}_1 = \{e_{oa}, e_{om}(dm_c = 0.2m, d_s = 0.5m), eci\} \tag{37}$$

where dm_c is the radius of the robots and d_s is the security distance of the obstacles, which indicates that e_{oa} is more important than e_{ci}. The resulting trajectory is seen in Figure 10.

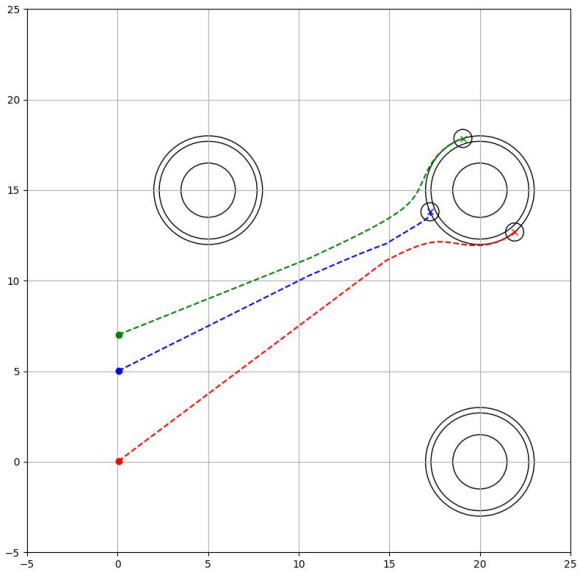

Figure 10. Formation phase.

8.2. Phase 2: Grip Phase

The robots must ensure that the object they are going to hold is kept in the centroid of the formation, therefore, they must avoid the target space without separating from it, while separating equally from each other. The task to accomplish is practically the same as the first phase, however, they need to take a greater distance to avoid collisions between the robots. Then

$$\mathbb{S}_2 = \{e_{oa}, e_{om}(dm_c = 2.5\sqrt{3}m, d_s = 1.1rm), eci\} \tag{38}$$

where $dm_c = 2.5\sqrt{3}$ is the distance needed in order to form a circumscribed triangle towards the objective as seen in Figure 11.

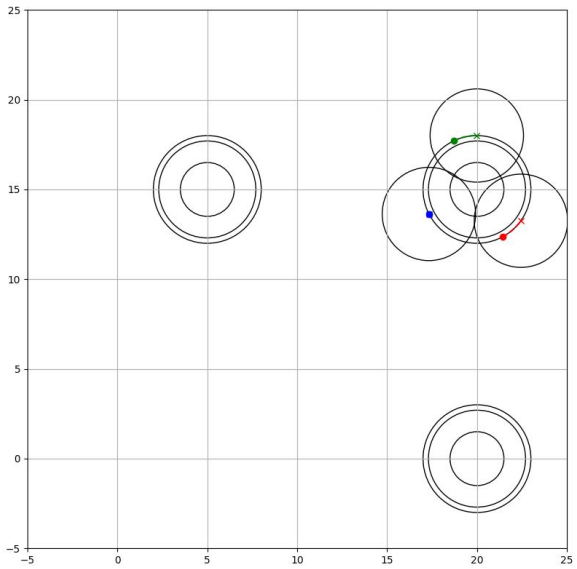

Figure 11. Grip Phase.

8.3. Phase 3: Manipulation Phase

Robots must maintain the desired permissible space in the formation while avoiding obstacles on the stage. The task is completed when they move the object, that is, the centroid of the formation to the desired position. Thus, the task phase becomes

$$\mathbb{S}_3 = \{e_{oa}, e_{om}(dm_c = 2.5\sqrt{3}m, d_s = 5m), e_{pi}, e_{tn}(q_d)\} \tag{39}$$

The behaviour can be seen in Figure 12.

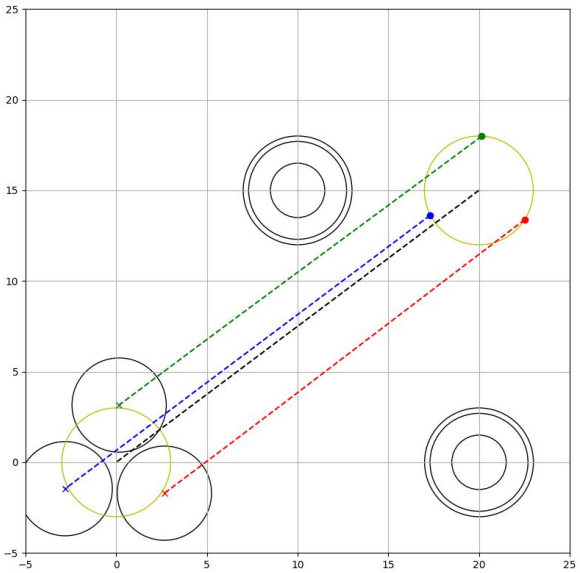

Figure 12. Manipulation Phase.

Notice that the formation can avoid also the target to move since it is logically considered an obstacle.

8.4. Phase 4: Waiting Phase

Once the robots have moved the target to the desired location, they separate from it and head towards a pre-defined obstacle-free position, in order to finish the job or wait for a new target. The task phase becomes

$$\mathbb{S}_4 = \{e_{oa}, e_{om}(dm_c = 0.2m, d_s = 0.5m), eti(\xi_d)\} \tag{40}$$

The behaviour can be seen in Figure 13.

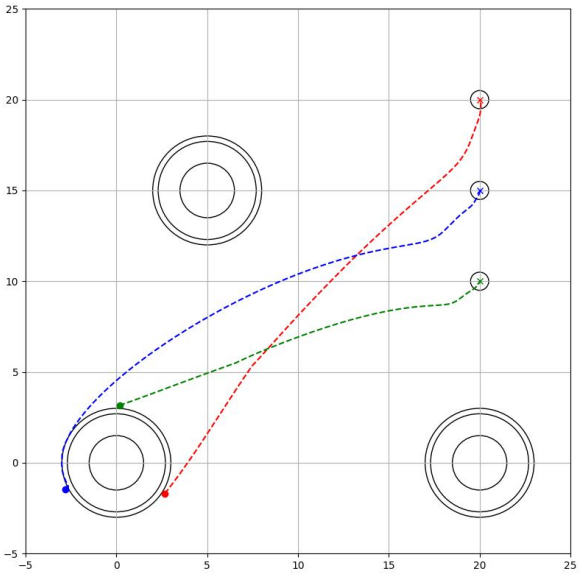

Figure 13. Waiting Phase.

Notice that in all the tasks one of the priorities is that the robots strictly avoid the obstacles, therefore it would be a task of greater importance in all the phases. Although the tasks of lower priority are projected into the null space of the stricter tasks, they find a close solution for each of the tasks.

8.5. Variance Value and Phase Measure

The closed-loop system becomes a differential stochastic system. Therefore, the stability analysis of the system has to consider tools applied in stochastic formulations.

The function α_i, defined in (35) and based on the error within phase $\mathbb{S}_i$, serves as the applied measure. The system exhibits robustness against uncertainty in α measurement, as evidenced by the recurrent activation of obstacle and collision avoidance tasks in Figure 14, without immediate transition until the error persists and the transition probability increases. This is in contrast to other schemes where the transition occurs as soon as a flag is crossed, leading to transition even in the presence of false positives arising from uncertainty in sensor measurement.

Due to the behaviour of the α distribution in each phase, the variance of each phase starts at zero and tends towards the next state until a transition occurs, as shown in Figure 16. The variance in a finite-state Markov chain is always bounded since there are a limited number of transitions from the current state. Thus, the validation of Theorem 1 is shown in Figure 16.

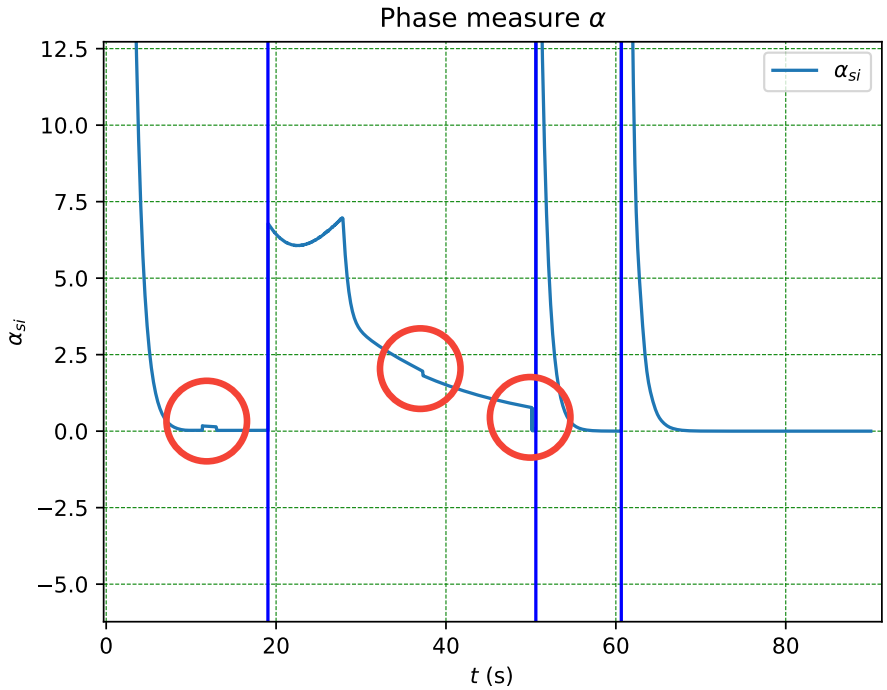

Figure 14. Close up of the Measure from Figure 15.

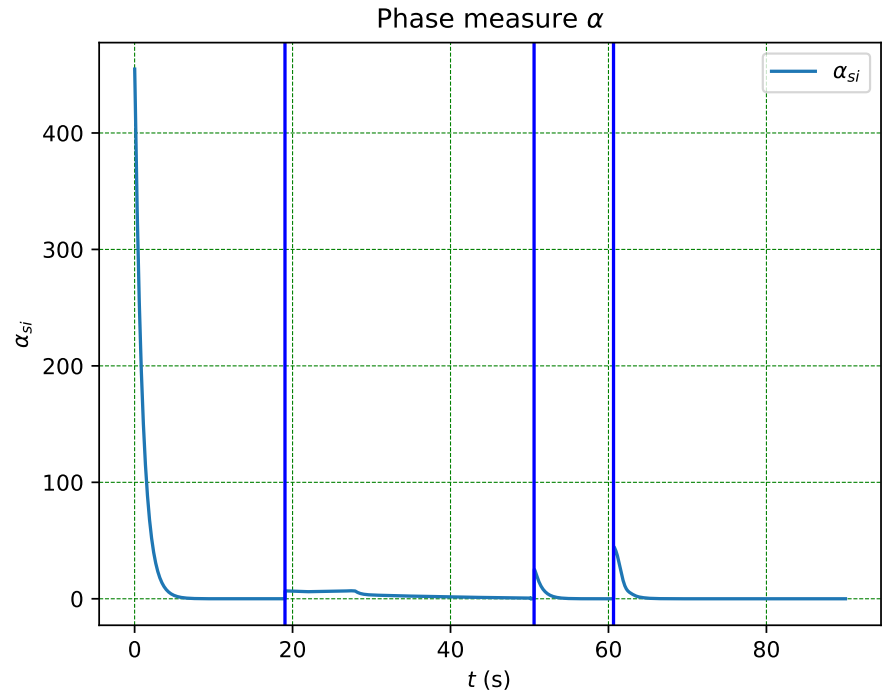

Figure 15. Measure given by each phase.

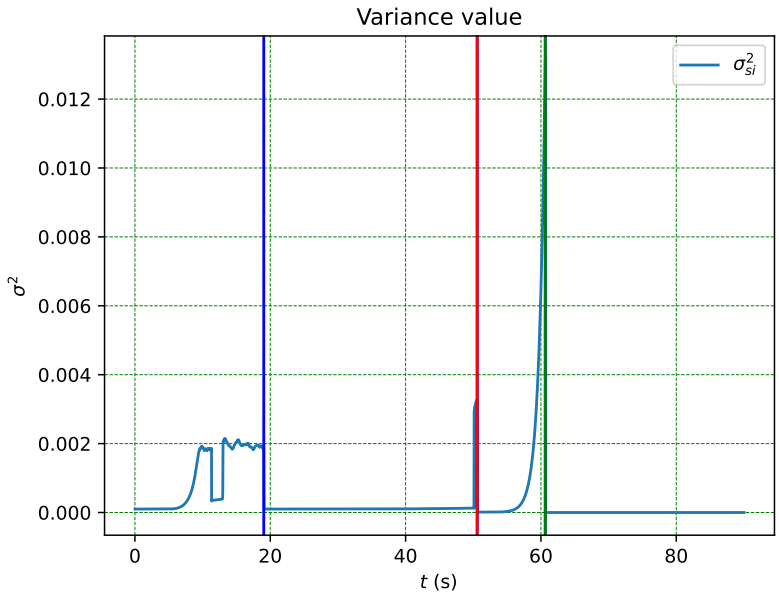

Figure 16. Variance given in each phase.

8.6. Comparison of DTMC and FSM in the Presence of Unexpected Failures

Figure 17 compares the DTMC and FSM decision agents for the same manipulation task in the case of unexpected failures that lead to the loss of partial state information and therefore the error in the measure of α of the system. The experience in the use of the system indicates that a flag error limit of $\alpha_{FSM} = 0.001$ is necessary for the correct functioning of the FSM while DTMC does not need such information. The failures occur between $t = 5$ and $t = 10$.

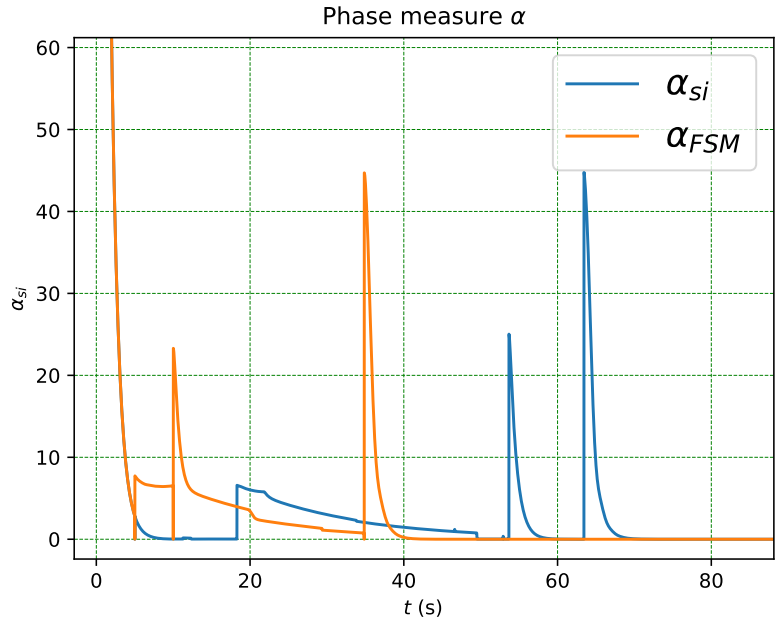

Figure 17. Comparison between DTMC and FSM decision agents.

The DTMC-based decision agent is able to recover from the failure and continues to make the correct decision, while the FSM-based decision agent is unable to recover and continues to make the incorrect decision. Note that at t = 5, due to the unexpected transition, the FSM decision agent can no longer satisfy the behaviour in the first phase, which implies that the following behaviours can no longer be satisfied. This highlights the advantage of using DTMC-based decision agents, as they are more robust and can handle unexpected failures better than FSM-based agents.

9. Conclusions

This paper presents a manipulation scheme for robots to transport objects in a workspace with obstacles. The novel decision agent proposed in this paper is the main contribution, as it guarantees the stability of the scheme and offers an effective and efficient solution for object manipulation in complex workspaces. The proposed decision agent is model-free, relying solely on errors of tasks in each phase, which are dependent on workspace information. Distribution by state measurement provides robustness against intermittent event measurement and unexpected behaviours, as demonstrated in simulation results comparing the DTMC agent with a FSM. The proposed approach reduces the required expertise during implementation, as the decision agent handles most of the complexity. The stability of the scheme was evaluated using mean-square analysis, with the variance being constrained at all times. The scheme demonstrated the ability to change the phase autonomously as the error approached zero, indicating its efficacy in ensuring stability. Future work includes developing auxiliary controls to enhance the robustness against parametric uncertainty and disturbances from contact forces.

Author Contributions: D.A.-J. proposed the manipulation scheme, theorem of the decision agent, contributed to the algorithm design, and generated the simulation results. A.S.-O. proposed the Multi-Robot system application, worked on the control law used in the scheme, and provided the contextual concepts for the theorem of the manipulation scheme. H.A. worked on the mean-square stability proof and definitions related to the proposed scheme and on the design of the Markov Chain decision Agent. All the authors discussed the proposed approach, and results, and reviewed and approved the paper. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data is available under request to the corresponding author.

Acknowledgments: The authors acknowledge the support from Conacyt-Mexico, CINVESTAV and UACAM.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

DTMC	Discrete-Time Markov Chain
HQP	Hierarchical Quadratic Programming
FSM	Finite-State Machine

References

1. Ebel, H.; Eberhard, P. Non-Prehensile Cooperative Object Transportation with Omnidirectional Mobile Robots: Organization, Control, Simulation, and Experimentation. In Proceedings of the 2021 International Symposium on Multi-Robot and Multi-Agent Systems (MRS), Cambridge, UK, 4–5 November 2021; pp. 1–10. [CrossRef]
2. Simões, M.A.C.; da Silva, R.M.; Nogueira, T. A Dataset Schema for Cooperative Learning from Demonstration in Multi-robot Systems. *J. Intell. Robot. Syst.* **2020**, *99*, 589–608. [CrossRef]
3. Lin, S.; Liu, A.; Wang, J.; Kong, X. A Review of Path-Planning Approaches for Multiple Mobile Robots. *Machines* **2022**, *10*, 773. [CrossRef]
4. Ordaz-Rivas, E.; Rodriguez-Liñan, A.; Torres-Treviño, L. Flock of Robots with Self-Cooperation for Prey-Predator Task. *J. Intell. Robot. Syst.* **2021**, *101*, 39. [CrossRef]
5. Saenz, J.; Bugarin, E.; Santibañez, V. Kinematic and Dynamic Modeling of a 4-Wheel Omnidirectional Mobile Robot Considering Actuator Dynamics. Congreso Internacional de Robótica y Computación. 2015. pp. 115–121. Available online: https://posgrado.lapaz.tecnm.mx/CIRC2015/CIRC2015.pdf (accessed on 18 March 2023).
6. Li, R.; Qiao, H. A Survey of Methods and Strategies for High-Precision Robotic Grasping and Assembly Tasks—Some New Trends. *IEEE/ASME Trans. Mechatronics* **2019**, *24*, 2718–2732. [CrossRef]
7. Juang, C.F.; Lu, C.H.; Huang, C.A. Navigation of Three Cooperative Object-Transportation Robots Using a Multistage Evolutionary Fuzzy Control Approach. *IEEE Trans. Cybern.* **2022**, *52*, 3606–3619. [CrossRef]

8. Dai, Y.; Kim, Y.G.; Lee, D.H.; Lee, S. Symmetric caging formation for convex polygon object transportation by multiple mobile robots. In Proceedings of the 2015 IEEE International Conference on Advanced Intelligent Mechatronics (AIM), Busan, Republic of Korea, 7–11 July 2015; pp. 595–600. ISSN: 2159-6255. [CrossRef]
9. Yazıcıoğlu, Y.; Bhat, R.; Aksaray, D. Distributed Planning for Serving Cooperative Tasks with Time Windows: A Game Theoretic Approach. *J. Intell. Robot. Syst.* **2021**, *103*, 27. [CrossRef]
10. Chandarana, M.; Hughes, D.; Lewis, M.; Sycara, K.; Scherer, S. Planning and Monitoring Multi-Job Type Swarm Search and Service Missions. *J. Intell. Robot. Syst.* **2021**, *101*, 44. [CrossRef]
11. Yang, H.; Sun, L.; Li, X. A Multi-Agent Navigation Controller for Sequential Tasks Performing. In Proceedings of the 2017 IEEE 7th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER), Honolulu, HI, USA, 31 July–4 August 2017; pp. 1680–1684. [CrossRef]
12. Wu, H.; Yan, W.; Xu, Z.; Li, S.; Cheng, T.; Zhou, X. Multimodal Prediction-Based Robot Abnormal Movement Identification Under Variable Time-length Experiences. *J. Intell. Robot. Syst.* **2021**, *104*, 8. [CrossRef]
13. Luo, S.; Wu, H.; Duan, S.; Lin, Y.; Rojas, J. Endowing Robots with Longer-term Autonomy by Recovering from External Disturbances in Manipulation Through Grounded Anomaly Classification and Recovery Policies. *J. Intell. Robot. Syst.* **2021**, *101*, 51. [CrossRef]
14. Shen, L.; Mao, P.; Fang, Q.; Wang, J. A Trajectory Tracking Approach for Aerial Manipulators Using Nonsingular Global Fast Terminal Sliding Mode and an RBF Neural Network. *Machines* **2022**, *10*, 1021. [CrossRef]
15. Pérez-Villeda, H.M.; Arechavaleta, G.; Morales-Díaz, A. Multi-vehicle coordination based on hierarchical quadratic programming. *Control. Eng. Pract.* **2020**, *94*, 104206. [CrossRef]
16. Koung, D.; Kermorgant, O.; Fantoni, I.; Belouaer, L. Cooperative Multi-Robot Object Transportation System Based on Hierarchical Quadratic Programming. *IEEE Robot. Autom. Lett.* **2021**, *6*, 6466–6472. [CrossRef]
17. Li, H.; Ding, X. Adaptive and intelligent robot task planning for home service: A review. *Eng. Appl. Artif. Intell.* **2023**, *117*, 105618. [CrossRef]
18. Obregón-Flores, J.; Arechavaleta, G.; Becerra, H.M.; Morales-Díaz, A. Predefined-Time Robust Hierarchical Inverse Dynamics on Torque-Controlled Redundant Manipulators. *IEEE Trans. Robot.* **2021**, *37*, 962–978. [CrossRef]
19. Alam, T.; Rahman, M.M.; Carrillo, P.; Bobadilla, L.; Rapp, B. Stochastic Multi-Robot Patrolling with Limited Visibility. *J. Intell. Robot. Syst.* **2020**, *97*, 411–429. [CrossRef]
20. Mirzaei, F.; Pouyan, A.A.; Biglari, M. Automatic Controller Code Generation for Swarm Robotics Using Probabilistic Timed Supervisory Control Theory (ptSCT). *J. Intell. Robot. Syst.* **2020**, *100*, 729–750. [CrossRef]
21. Nguyen, V.; Kim, O.T.T.; Pham, C.; Oo, T.Z.; Tran, N.H.; Hong, C.S.; Huh, E.N. A Survey on Adaptive Multi-Channel MAC Protocols in VANETs Using Markov Models. *IEEE Access* **2018**, *6*, 16493–16514. [CrossRef]
22. Meyn, S.P.; Tweedie, R.L. *Markov Chains and Stochastic Stability*; Springer London: London, UK, 1993. [CrossRef]
23. Poznyak, A.S.; Alazki, H.; Soliman, H.M. Invariant-set design of observer-based robust control for power systems under stochastic topology and parameters changes. *Int. J. Electr. Power Energy Syst.* **2021**, *131*, 107112. [CrossRef]
24. Yang, Y.; Xi, X.; Miao, S.; Wu, J. Event-triggered output feedback containment control for a class of stochastic nonlinear multi-agent systems. *Appl. Math. Comput.* **2022**, *418*, 126817. [CrossRef]
25. Han, Y.; Zhou, S. Novel Criteria of Stochastic Stability for Discrete-Time Markovian Jump Singular Systems via Supermartingale Approach. *IEEE Trans. Autom. Control* **2022**, *67*, 6940–6947. [CrossRef]
26. Qi, T.; Chen, J.; Su, W.; Fu, M. Control Under Stochastic Multiplicative Uncertainties: Part I, Fundamental Conditions of Stabilizability. *IEEE Trans. Autom. Control* **2017**, *62*, 1269–1284. [CrossRef]
27. Long, S.; Zhong, S. Mean-square exponential stability for a class of discrete-time nonlinear singular Markovian jump systems with time-varying delay. *J. Frankl. Inst.* **2014**, *351*, 4688–4723. [CrossRef]
28. Francelino, E.; Pereira, M.; Inoue, R.; Terra, M.; Siqueira, A.; Nogueira, S. Markov System with Self-Aligning Joint Constraint to Estimate Attitude and Joint Angles between Two Consecutive Segments. *J. Intell. Robot. Syst.* **2022**, *104*, 43. [CrossRef]
29. Cao, X.; Jain, P.; Goodrich, M.A. Adapted Metrics for Measuring Competency and Resilience for Autonomous Robot Systems in Discrete Time Markov Chains. In Proceedings of the 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Prague, Czech Republic, 9–12 October 2022; pp. 71–76. [CrossRef]
30. Ralph-Christoph, W.; Bellenberg, M.; Schwarzenberger, D. *Festo's Robotino User Guide*; Manual Order No.:KG 544305; FESTO: Esslingen, Germany, 2009.
31. Lang, A.; Petersson, A.; Thalhammer, A. Mean-square stability analysis of approximations of stochastic differential equations in infinite dimensions. *BIT Numer. Math.* **2017**, *57*, 963–990. [CrossRef]