GRID-BASED CYCLIC MULTI-ROBOT ALLOCATION FOR OBJECT CARRYING

A Thesis

Submitted to the Faculty

of

Purdue University

by

Jee Hwan Park

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science

August 2020

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF THESIS APPROVAL

Dr. Galen King, co-Chair

School of Mechanical Engineering

Dr. Byung-Cheol Min, co-Chair

Department of Computer and Information Technology

Dr. Nina Mahmoudian

School of Mechanical Engineering

Approved by:

Dr. Nicole Key

Head of the School Graduate Program

ACKNOWLEDGMENTS

This work would not have been possible without my advisors, Professor Byung-Cheol Min and Professor Galen king. I would like to express my deep and sincere gratitude to my advisors for supporting and helping me. It was a great privilege and honor to work and study under their guidance. I am extremely grateful for what they has offered me. I would also like to thank them for their friendship, empathy. I would like to appreciate my committee member, Professor Nina Mahmoudian for sparing her time to discuss, to provide a valuable feed backs about my research. I also would like to thank all of SMART lab members for their support. It was great honor to work with talented people to achieve meaningful and valuable results. Lastly, I would like thank my family for their love, support and encouragement from South Korea. It was a great privilege to study abroad and to connect with talented students and professors. Finally, my thanks go to all the people who have supported me to complete the research work directly or indirectly.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

SYMBOLS

g_x	horizontal grid spacing
g_y	vertical grid spacing
q_p	Boundary points of the object
p	set of boundary points
x_p	x position of particular boundary point
y_p	y position of particular boundary point
x_c	x position of center of mass
y_c	y position of center of mass
q_c	position of center of mass of an object
C	Planar shape
θ	orientation of C
set$A$	group of robots under the object
set$A$	group of robots surrounds the set$A$
R	Generated polygon by boundary of set A robots
N_{min}	Minimum number of robots required for transportation
$\triangle x_\theta$	Set of distance from x_c to x_p
$g_{x,\theta}$	horizontal grid spacing at θ
$g_{y,\theta}$	vertical grid spacing at θ
$\triangle q_p$	set of distance between one boundary point and another boundary point
$d_{p,\theta}$	set of distance that $\triangle q_p$ is equal to $g_{x,\theta}$
$\triangle y_\theta$	set of distance between center of mass and $d_{p,\theta}$
δx	horizontal safety margin
δy	vertical safety margin

x^g	horizontal axis of global coordinate
y^g	vertical axis of global coordinate
z^g	abscissa axis of global coordinate
x^b	horizontal axis of attached coordinate
y^b	vertical axis of attached coordinate
z^b	abscissa axis of attached coordinate

ABSTRACT

Park Jee Hwan MS, Purdue University, August 2020. Grid-based Cyclic Multi-robot Allocation for Object Carrying. Major Professors: Galen King, Byung-Cheol Min.

In this thesis, we are addressing new method of object transportation using multi-robot system. The new method of object transportation is called A grid-based cyclic robot allocation (GCRA) method which consists multiple spherical robots. The object is placed on top of group of spherical robots before the transportation. The rotation of the multiple spherical robots cause the displacement of the object and reach the goal location based on the direction and speed of the rotation of the robots. The GCRA method for spherical robots is proposed along with specific stability criterion, which designs the formation of the multi-robot system. The formation is created based on the customized grid which is to be modified based on the properties of the object. The shape and the center of gravity of the shape define the horizontal gap, g_x and vertical gap, g_y. All the possible locations of spherical robots is the cross points of grid which implies that g_x and g_y defines the distance between the robots and based on the boundary of the robots placed underneath the object, the condition of the stability is defined. It also identifies minimum number of robots required based on the arbitrary shape of an object for stable omni-directional translation of the object on a 2 dimensional space. The desired positions and formation of the robots is identified based goal position of the object. Under centralized system, position control is applied to drive the robots to the desired positions. The position control simultaneously makes the object mobile and maintain the stability of the object. Mathematical proof of the proposed method is shown verifying the stability of the transportation process with the assumptions of no slip between the robots and the object. 2 Dimensional Simulation results of robot allocation using GCRA for several arbitrary shapes certify the proposed method.

1. INTRODUCTION

This thesis introduces new method of object transportation using multi-robot system. A multi-robot system is defined by a system that consist more than one robot. Multi-robot system can be applied at various tasks such as, cooperation of multi-robot system with human for rescue purpose [1], applying multi-robot system for Mapping Environmental Variables of Greenhouses [2], multi-dimensional mapping using the multi-robot system and SLAM [3] and target navigation and detection using multi-robot system [4]. One of the scenarios where multi-robot system can be employed is object transportation. Different ways of object transportation are discovered and studied in the past such as Pushing [5] [6], Grasping methods [7] and Caging [8] [6]. A multi-robot system of object transportation can be effective and helpful in a variety of applications that can bring high economical impacts; e.g., Object transportation in rural area [9] hazardous material retrieval and disposal, warehouse robots used to carry an heavy objects [10]. The cooperative multi-robot system applied in the object transportation can make the transportation more time efficient and reduces the risk from handling heavy or hazardous objects.

Different types of robots are used for the object transportation such as using non-holonomic mobile robots to transport an object [11], using cooperative swarm robotics for the transportation [12] [13].

The new mechanism of transporting an object is to carry an arbitrary object using multiple spherical robots. Unlike the existing methods of transporting an object, it can transport heavier object with less effort on each robots, robots does not require any functionalities such as grasping arm but the function to rotation and new alignment of the robots is not required due to omni-directional movement.

The major study of this thesis is to compute the most efficient allocation of the robot based on the properties of the object to complete the transportation while ensuring the stability of the object.

1.1 Background

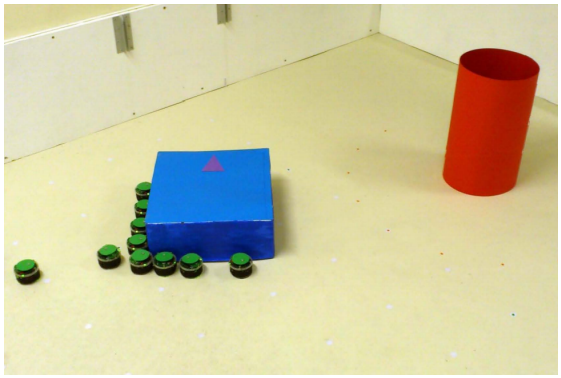
Object transportation using multi-robot systems has been a popular research topic due to its robustness and fault tolerant features. Over the years, significant improvements in individual robot capabilities for object manipulation and coordinating with others in a group have been observed for successful implementation of multi-robot object transportation in the field. As a result, traditional and popular methods of object transportation by multi-robot systems have remained confined to pushing [14], grasping [15], and caging [16] strategies. Different types of robot and control systems are used for each cases such as using omni-directional robots to transport an object using rotation control [17], using multiple quad-rotors cooperative object transportation without any peer communication [18]

As shown in the Fig. 1.2(a), pushing strategy uses the multi-robot system to generate push force on the object and causes the displacement. To successfully transport the object, robots should be able to generate enough force to overcome the external force applied on the object such as friction between the object and the ground. Grasping strategy includes multiple robots that has function that can physically grasp the object in order to transport the object. Like the Fig. 1.2(b), once the object is grasped, the robots starts to transport the object in desired path. Caging strategy encloses the object in certain formation of the robot to make sure that the object follows the planned path as shown in Fig. 1.2(c). It is similar to pushing strategy but the caging strategy maintain the enclosure during the transportation.

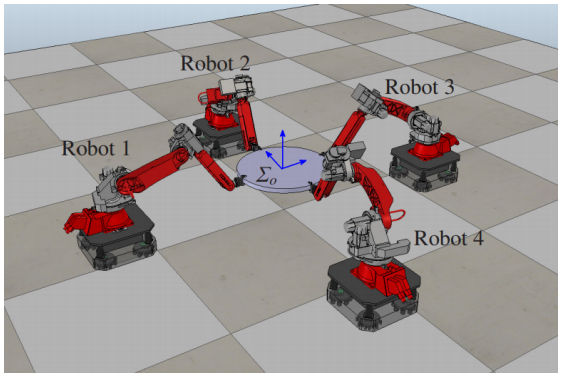
1.2 Problem statement

The traditional methods are already applied in various application such as usage of swarm robotics to push an object [5] and still lots of related studies are on-going. These strategies are effective and widely used. However, these strategies still have some limitations and possible improvable points as follows:

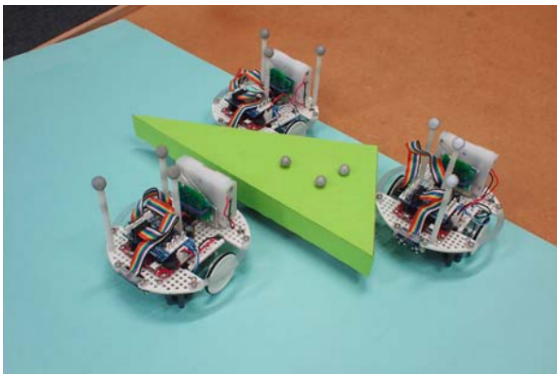
- Pushing strategies can be limited based on the weight of the object. Each robot should be able to produce enough force to overcome the external force applied on



(a) Pushing Strategy. The image was downloaded from [5].



(b) Grasping Strategy. The image was downloaded from [7].



(c) Caging Strategy. The image was downloaded from [8].

Fig. 1.1.: Traditional strategies of object transportation.

the object. This can be crucial once the multi-robot system tries to transport a heavy object.

- Grasping strategies limits the material of the object to be transported and each robot should have grasping function that can physically hold the object and the robot.

- Alignment of the robot has to be precise to translate the object in desired path. Both caging and pushing requires precise formation control and alignment control. To follow an arbitrary path with curves and sharp turns, the alignment and formation needs to be precisely controlled.

1.3 Research Question

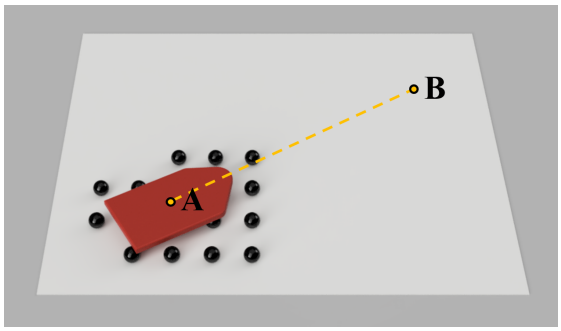
In this thesis, we are trying to address the listed limitations by introducing new method of object transportation. The new strategy is called object carrying method. object carrying method is a multi-robot system consisting multiple spherical robots [19] [20] with the functionalities of rolling in omni-direction. The multi-robot system will carry the object and transport it using the rotational movement. The spherical design of the robot brings agile and efficient translation. The strategy does not requires complex inter-agent coordination as long as the object is supported. With the assumption of no slip condition between the object and the robot, the as the robot travels in a speed of v, the rotation of the robot causes the object to travel in speed of $2v$. Due to the difference of the traveling speed, the robot falls behind while the object is traveling. This can cause the instability of the object. Thus, the allocation of the robot is the critical in this strategy.

An effective allocation of multi-robots can assure rapid transportation while maintaining the stability of the object. The stability is maintained as long as the center of the mass of the object is geometrically within the polygon that is formed by outer groups of robots. It also ensures even distribution of the weight of the object on each carrying spherical robots to minimize the individual effort during the transportation process. Unlike the other object transporting strategies, Object carrying strategy performs better with heavier objects. This is because the weight of the object is directly related to the friction between the object and the spherical robots. This friction allows less slip during the rotation of the robots. This mechanism allows the object to be transported to the desired point without requiring any special features for pushing or grasping.

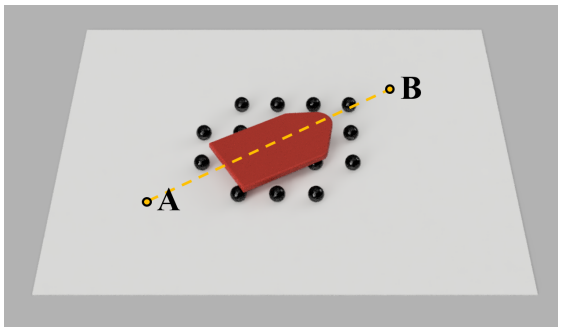
For the successful transportation using object carrying strategy, a fundamental problem is the allocation of robots. Computing the proper formation of the multi-robot system can ensure the stability of the object and avoid over-dependency of weight for robots while transporting. Thus, we present a Grid-based Cyclic Robot Allocation (GCRA) method in this paper. The robot allocation for object carrying strategy should consider the object's properties such as shape and the location of center of mass. By using these consistent data,

robots will be allocated. The allocation includes the horizontal and vertical gaps of the grid. The computed design of the grid and the desired path of the object can compute the minimum robots required to transport. Computed parameters of the grid will ensure the stability and the completion of the efficient transportation of the object.

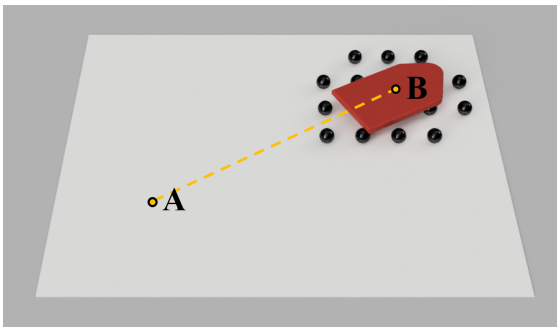
Therefore, the research question is *what is the most efficient allocation strategy to ensure the stability of the object during the transportation while using minimum number of robot required?* By solving this research question, the object carrying strategy can use GCRA method to transport the object maintaining the stability.



(a) Concept description for object carrying - Start position.



(b) Concept description for object carrying - Mid position.



(c) Concept description for object carrying - Final posiion.

Fig. 1.2.: Object carrying strategy using multi-robot system. The point A indicates the starting position of the object and the point B indicates the goal point of the object.

As shown in the Fig. 1.2, a multi-robot system consisting spherical robots allocates create a formation using GCRA methods. GCRA method generates the required properties of the grid consisting the information of horizontal gap, g_x and vertical gap, g_y. The

possible position of the robots are the cross points of the grid and according to the shape and the location of the object, GCRA method computes the required position of the robots. Once the robots are positioned, the object is placed on top of the robots and the rotation of the robots cause the transition of the object. The carrying method is a continuous system and the allocation of the robots continuously updates based on the object's position. Continuous allocation of the robots are required because since the object travels twice faster than that of the robots, fall out robots needs to re-position itself to maintain the required formation. This procedure will eventually make the object reach the desired goal position.

1.4 Assumptions

This thesis has following assumptions:

- The system is centralized and position of each robot and the object is known. Each robot has accessibility of the object and other robot's position.
- The object has a point contact and there is no slip between the object and the robots. This is to ensure that the rotation of the robots can directly impact the transition of the object. In addition, the point contact assumption ensures that action to change the heading direction of the robot does not cause any movement of the object.
- The object has a flat surface on the contact with robots. This ensures that the all the supporting robots has point contact on the object.

2. RELATED WORKS

Application for multi-robot systems has been extensively studied. Multi-Robot Systems (MRS) is defined as a system that includes more than one robots to achieve a task more efficiently. The application for MRS includes indoor mapping, wide range communication and object transportation. For object or payload transportation, MRS is widely used to achieve a task more efficiently using more than one robot. MRS is essential because it can accomplish the goal more quickly with less effort on each robot. Object transportation using MRS has been extensively studied in literature with focus on some traditional strategies. The strategies are pushing, grasping and caging strategies. This studies can be extended to the robust object transportation where the object's properties are unknown in advance [21]or transportation of an object by using formation control in dynamic environment [22]. Various mechanisms and approaches for traditional methods are studied and it is recently reviews by Tuci *et al.* [23]. It categorized the object carrying methods under grasping strategies since the robots align their forces and sustain the transport without losing physical contact with the object.

Some of the earliest work on multi-robot object carrying were proposed by Stilwell *et al.* [24] and Johnson *et al.* [25] where a group of ant-like robots transported palletized loads of unknown mass and center of geometry in a distributed system. Coordination between the robots was achieved based on sensing the forces applied on the object by the robots. Similar coordination in object carrying were later implemented in [26] and [27] with a leader-follower approach.

In this context, we emphasize the significance of GCRA proposed in this paper in Section **??**, that it does not require further computation or adjustments in robot allocation for object carrying in two dimension once the initial cyclic grid system is initialized. The proposed grid generation method is solely based on the properties of the object. The method is applicable for multi-robot system consisting group of spherical robots that does not require

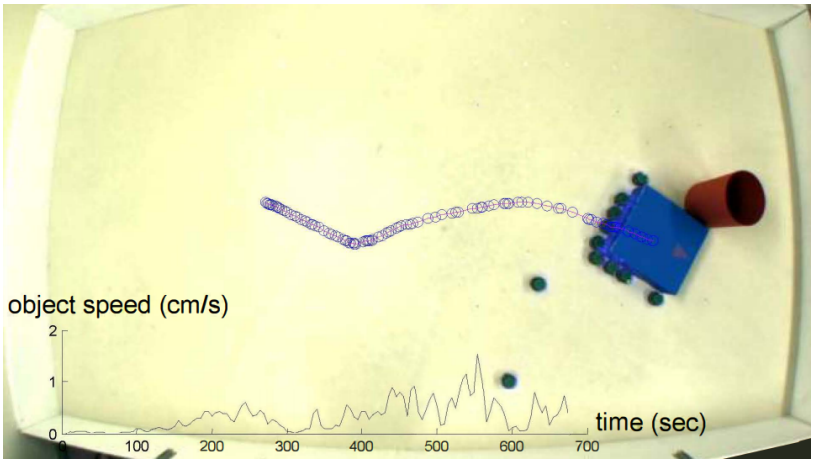
extra functionalities except rotation. The proposed grid generation method allows planar omni-directional translation as shown under grid analysis with proof of stability in Section 4.3.

2.1 Pushing strategy

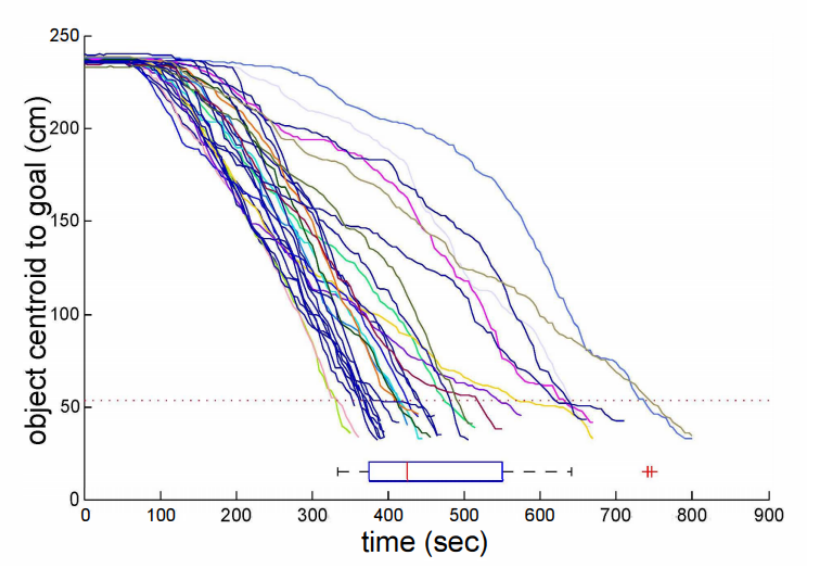
Pushing strategy is the methods of using multi-robots that are not attached to the object. Multi-robots exerts pushing forces on the object to overcome frictional force, gravitational force and other dynamical force during the transportation. The pushing strategy may appear to be simple but this strategy has challenge of alignment of the total force created by each robots. The object can be move on a inefficient trajectory unless the robots gently manage the external forces to stabilise the direction of transport. Most of the pushing strategies are based on homogeneous groups, where the controller of the robot is designed using a behaviour-based methodology [28].

One of the application using the pushing strategy is done by Gerkey and Matarić [6]. The application contains three robots in which one robot of the group acts as a watcher, and the other two robots act as a pusher. The watcher detects the object and the final goal point. The main task of the watcher is to lead the multi-robot system by communicating and providing the information regarding the direction towards the goal. Main function of pushers is to provide enough push force to push the object without sensing or observing any information related to goal position. The main advantage of the strategy is that it can avoid any obstacle on the desired path. On the other hands, alignment of the robot has to be precise to translate the object in desired path and pushers should be able to produce enough force to overcome external forces on the objects.

Another application is done by Chen and Gauci [5]. This uses swarm of mobile robots to push the object to the desired goal. By using the finite state machine, each robot relocates to push the object. Each robot has 8 proximity sensors to detect the object and one color sensor to detect the object and the goal. The core technology of this mechanism is to use well designed finite state machine to achieve the goal in decentralized system.



(a) Object centroid to the goal - Pushing strategy [5].



(b) Displacement tracking - Pushing strategy [5].

Fig. 2.1.: Result of pushing strategy using swarm robotics. The images were downloaded from [5].

As it can be seen from the Fig. 2.1, The multi-robot system successfully transported the object for all 30 trials. However, the variation of the time to reach the final destination is huge which implicates that the mechanism is not the most efficient to find the path to reach the goal. This maybe caused by inefficient alignment of the robots to generate the push force. Moreover, similar pushing mechanism could also applied in rod-type object transportation which requires detailed force analysis [29].

2.2 Grasping strategy

Grasping strategy transports object by a robot that is physically attached to an object. Therefore, grasping strategies can be accomplished using robots with the grasping mechanism such as robotic arms. One of the widely used mechanism along with grasping is lifting mechanism to perform a motion of lift and transport. Compared to other strategies, grasping strategy provides a better control of the object once it is grasped and lifted. However, re-positioning of the object using this mechanism is dependent on the weight of the object, graspable area of the object and the materials of the object which can change the coefficient of frictional.

Similar to the watcher and pusher strategy, a leader and follower approach is also applied by Wang and Schwager [7]. The application introduces a multi-robot system to transport an object using a group of four robots. The leading robot pulls the object and perceives the direction towards the goal and other three robots pushes the object to produce enough force. The control model requires information beforehand such as mass of the object, friction coefficients and total number of the robots forming the group. This is to compute the velocity and acceleration at the centre of mass of the object. Each robot has a gripper of single degree of freedom to grasp the object. Moreover, The study can be extended to estimation of the centroid of the object and use motion controller for collective transport by multi-robot systems [30].

As it can be seen from the Fig. 2.2, The robot requires various functions to grasp the object. The design for each robot is complicated and requires lots of data handling for each robot. The robot is customized to perform task with light weight object but if the study is extended to transport a heavy object, robot is required with higher functionalities which is not economical.

2.3 Caging strategy

Object transportation by caging strategy has similarities with pushing strategy. Caging strategy intentionally encloses the object by group of robots and ensures that the object

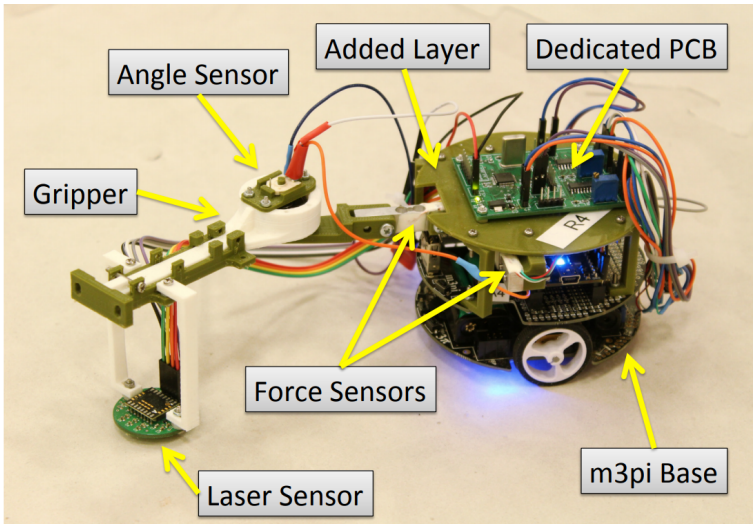


Fig. 2.2.: Custom-built manipulation robot used to grasp the robot which includes various sensors for the force feedback system. The image was downloaded from [7].

follows the planned trajectories. From the start of the transportation to the end, the object will be surrounded to avoid any desertion from the enclosure. To build a efficient caging strategy, the object's size and shape are important features to create multi-robot system that can complete the task using minimum number of robots required. The main challenge of the strategy is the allocation and positioning of robots.

One of the application using the caging strategy can be described in [31]. It uses three robots that can travel in omni-direction and applied force controller to the robotic system to transport an object. This study clearly shows the difference from watcher and pusher strategy [6]. The difference is that the leader robot only produces force to transport and object while the other robots holds the object by creating a cage. Robots detects the resultant force of the object and acts as a feedback of the controller. This allows following robots to maintain the shape of the cage or the formation. The limitation of the mechanism is that accuracy of following the arbitrary path is low especially with sharp turns.

Apart from centralized multi-robot system applied to the caging method, decentralized algorithms for object transportation using multi-robot system via caging is studied in [32].

As it can be seen from the Fig. 2.3, The object was successfully transported to the goal position in the condition of unbalanced ground. However, the success rate varies based on

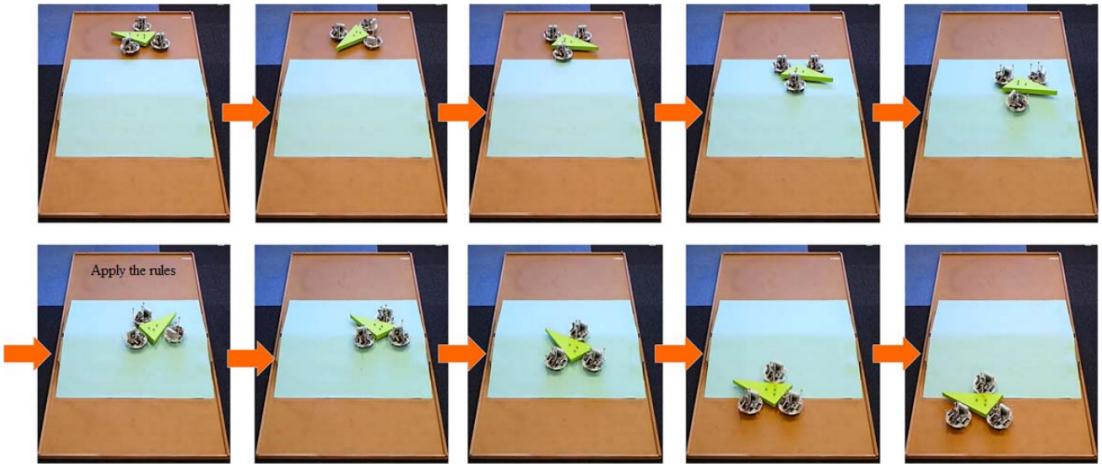


Fig. 2.3.: Custom-built manipulation robot used to grasp the object which includes various sensors for the force feedback system. The image was downloaded from [8].

the posture of the object. The success rate for certain orientation of the object is very low. This indicates that the orientation and the allocation of the robot while transporting needs to be perfect and maintained. Unsuccessful control for each robot to maintain the enclosure will cause the failure.

2.4 Carrying strategy

In the early work of Grid-based Cyclic Robot Allocation [33], The result stated that the grid-based robot allocation could transport the object. The design of the grid is solely based on the object's shape and center of mass of the object. The strategy was successfully tested with the Matlab simulation. As it can be seen from the Fig. 2.4, the simulation of the transportation was successful in terms of stability of the object and the achievement of the given task. However, the study considered the movement of the transportation in only one direction with single path. This does not fully ensures the success of the transportation and needs to be tested in higher dimension.

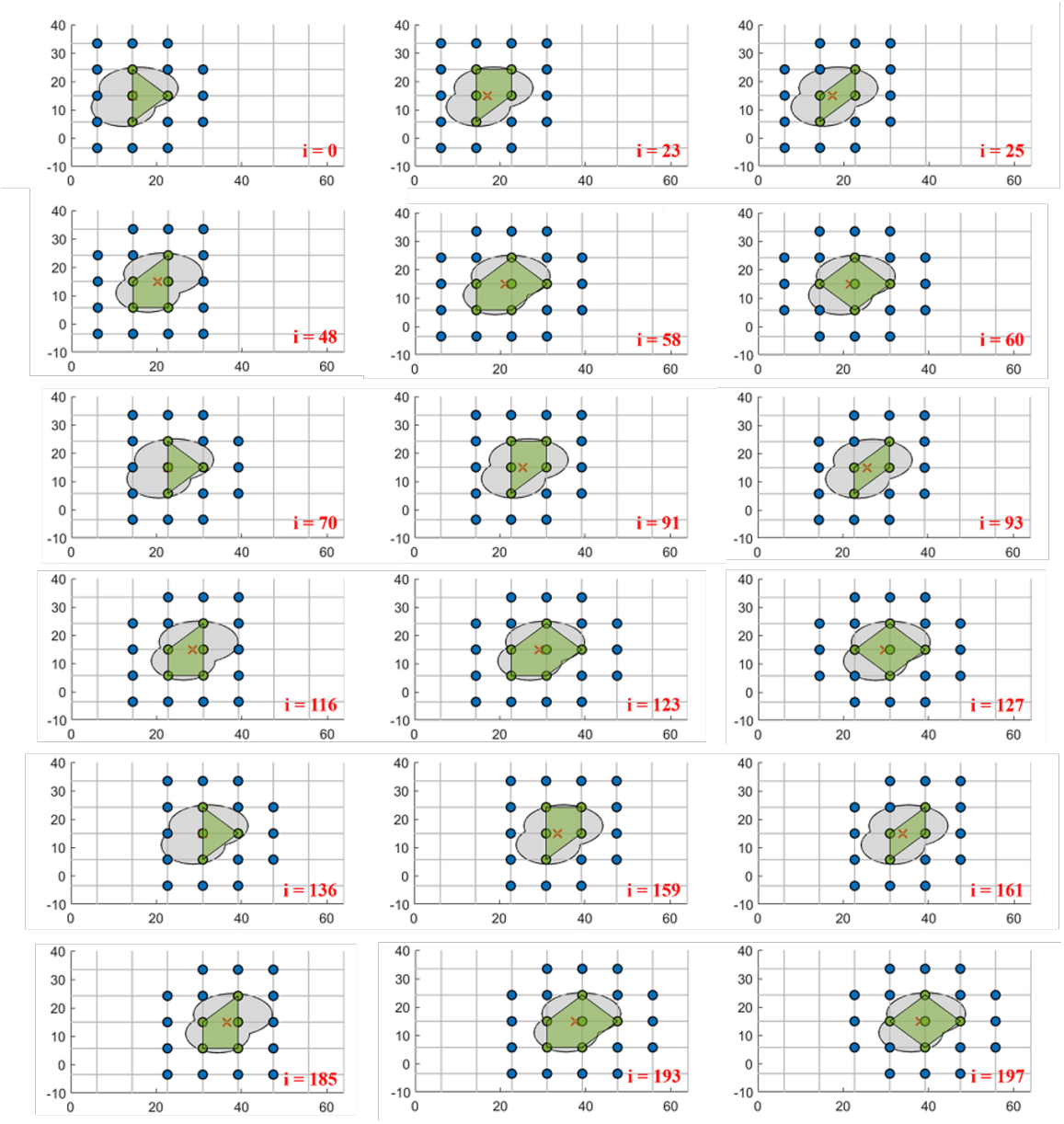


Fig. 2.4.: Result for Early GCRA method using an arbitrary shape 1. The direction of the transportation is in x-axis only.

3. PROBLEM STATEMENT

For Grid based Cyclic Robot Allocation (GCRA) method, the object is handled in two dimensional space. Let the object to be defined as an arbitrary planar shape C consisting of the set of m boundary points as shown in 3.1.

$$q_p = (x_p, y_p) \in \mathbb{R}^2, \qquad p \in \{1, 2, .., m\} \tag{3.1}$$

An arbitrary shape of the object is shown in the Fig. 3.1, the global coordinate is defined as $x^g y^g z^g$. With the defined boundary of the object, C, the center-of-mass is denoted as $q_c = (x_c, y_c)$. In this case, the center-of-mass of the object is assumed to align with the geometric center of the shape such that $q_c \notin C$. In addition, the mass of the object is equally distributed and the planar shape C is consistent in z^g such that q_c is always with in the boundary of C. A local coordinate frame $x^b y^b$ is attached on the object at q_c with the object orientation relative to $x^g y^g$ denoted as θ. The object is placed on top of the allocated robots and to create a no-slip condition between the object and the robot, the weight of the object assumed as high enough to generate frictional force.

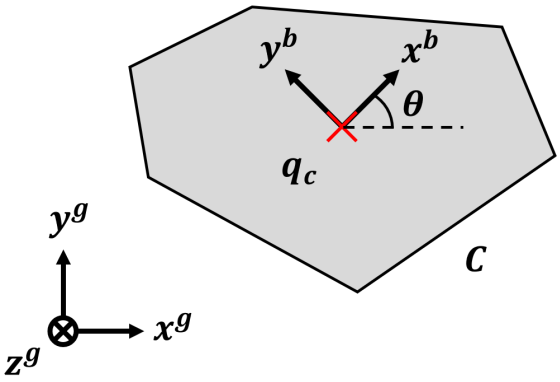
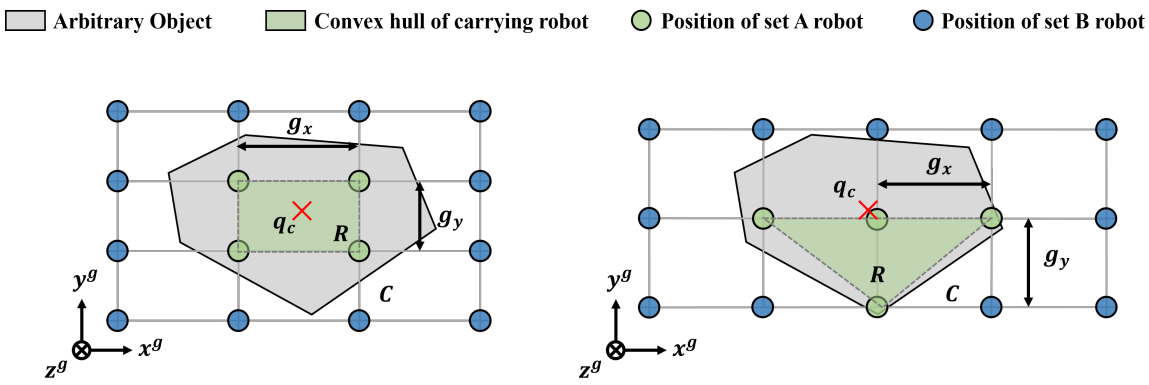


Fig. 3.1.: An arbitrary object Planar with the shape C with centroid q_c at orientation θ.

To transport the object C in omni-direction, robots are allocated as a grid formation. The grid is relative to $x^b y^b$ with spacing along the x and y axis defined as horizontal and vertical gap which is denoted by g_x and g_y. The only possible desired positions of N robots are the cross points of the generated grid. Position control and collision avoidance are applied on each robot to re-position itself. N robots are divided in set A or set B; set A represent the robots under the planar shape C, and set B represent the robots that are not directly involved in carrying but ensures the stability of the object in transition or change of the direction.

Stability of the object can be defined as consistency of the inclination of the object in z^g. This can be determined by set A such that the center of the mass, q_c of the object is within the outer boundary formed by the set A as shown in 4.3.



(b) Stable allocation of robots - inclusion of q_c within convex hull, R.

(c) Unstable allocation of robots - exclusion of q_c within convex hull, R.

Fig. 3.2.: Definition of stability of the object based on the grid characteristic, g_x and g_y.

The rotation of spherical robots of set A causes the translation of C and R. For each rotation of the robot, the displacement of the object is twice of that of the robot. We assume that the speed of the rotation of the robot is step input with constant velocity which, denoted by v_r. The rotation causes the transition of C in twice speed of that of the set A robots, v_r. Therefore, the relative velocity of R is half of that of C.

Due to the difference in the velocity of R and C, set A robots lose contact with C during the transition. At the same time, robots needs to cover the leading edge and maintain the

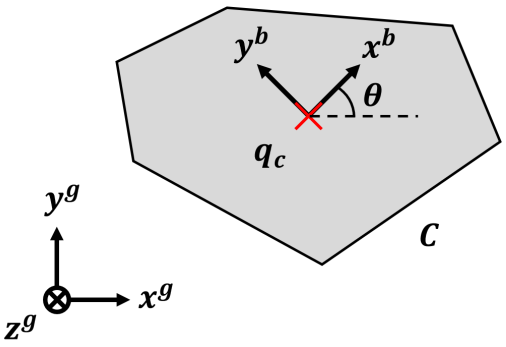
formation to ensure the stability. Breakdown of the position covering or inefficient g_x and g_y computation cause the falling of the object and leads to failure of the transportation. Therefore, the stability criterion of the object carrying using spherical robots is defined as follows.

Definition 1: Object C of uniform weight distribution being carried by N spherical robots forming a convex hull R is stable, if the geometric centroid q_c of object C is always bounded within the convex hull R.

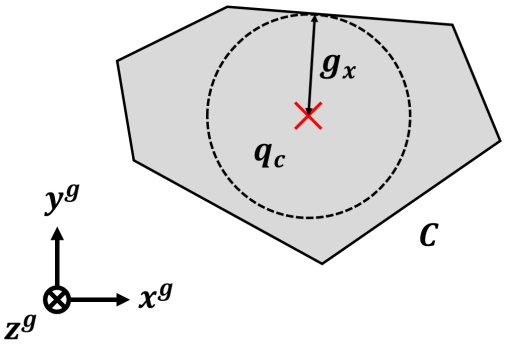
The objective is to determine the grid spacing parameters g_x and g_y for the grid formation based on the properties of the object, and the minimum number of robots required denoted as N_{min}, such that stable omni-directional transportation of the object defined as planar shape C with constant g_x and g_y is achieved based on the stability criterion defined in *Definition 1*.

4. PROPOSED SOLUTION

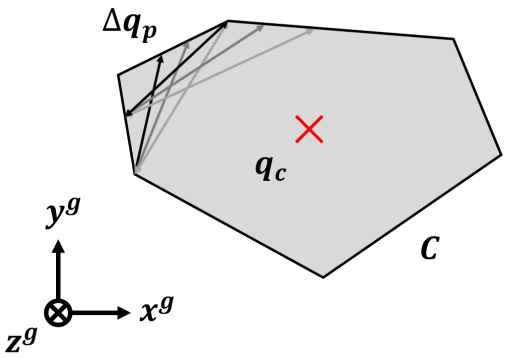
We design the proposed GCRA method for spherical robots for object carrying, by defining the unit grid size parameters g_x and g_y, where the resulting grid intersections under C denote allocated robots.



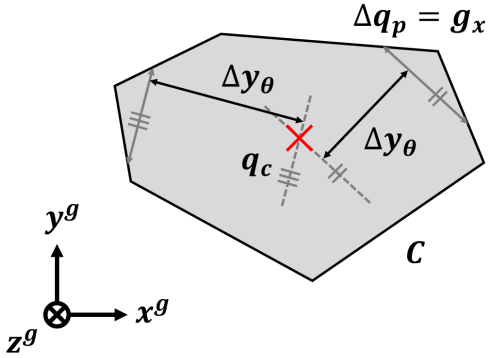
(a) Step1: Define the shape of the object, C and the center of the mass, q_c.



(b) Step2: Determine the shortest distance between the center of the mass, q_c and q_p and define it as g_x.



(c) Step3: Set up $\triangle x_\theta$ that is the distance between every q_p.



(d) Step4: Compare all the elements in the set $\triangle x_\theta$ with g_x to determine g_y.

Fig. 4.1.: Computation of grid properties based on C and q_c.

4.1 Grid generation system

Grid generation system is differed based on the shape of the object. With the defined object shape, C as shown in Fig. 4.1(a), the horizontal grid spacing g_x relative to the local frame $x^b y^b$ must be the minimum distance between the center of the mass, q_c and $q_p \in C$, $\forall p = \{1, 2, .., m\}$ for planar omni-directional object translation. This condition allows circle to be drawn with the radius of g_x and the center at the q_c as shown in Fig. 4.1(b). If the circle's boundary does not include every q_p, the grid generation follows the normal steps. However, if the boundary include every q_p, the grid generation follows the exceptional case.

For the first normal step, the set of lengths from q_c to $q_p \in C$, $\forall p = \{1, 2, .., m\}$ along the $x - axis$ for orientation θ from 0 to 2π is calculated as,

$$\triangle x_\theta = \{ \mid x_c - x_p \mid y_p = y_c, \forall (x_p, y_p) \in C, 0 \leq \theta < 2\pi \}. \tag{4.1}$$

With defined set $\triangle x_\theta$, The horizontal grid spacing $g_{x,\theta}$ is the minimum element of the set, which can be derived as,

$$g_{x,\theta} \leq \min(\triangle x_\theta). \tag{4.2}$$

With defined horizontal grid space, set of distances between q_p, $\forall p = \{1, 2, .., m\}$ needs to be computed first. This is defined as,

$$\triangle q_p = \{|q_p - q_{p+1}|, \forall (p) = \{1, 2, .., m\} \in C\}. \tag{4.3}$$

All elements in $\triangle q_p$ needs to be compared with the horizontal grid spacing to compute $d_{p,\theta}$. This is defined as,

$$d_{p,\theta} = \{\triangle q_p = g_{x,\theta}\}. \tag{4.4}$$

For each and every element of set $d_{pj,\theta}$, define the equation of the line as $a_n x^g + b_n y^g + c_n = 0$ where n denotes the number of elements in set $d_{p,\theta}$.

The set of lengths from q_c to $d_{p,\theta} \in C, \forall p = \{1, 2, .., m\}$ is calculated as,

$$\triangle y_\theta = \{\frac{\mid a_n x_p + b_n y_p + c_n \mid}{\sqrt{a_n b_n}}, \forall (n, p) \in C\} \tag{4.5}$$

With defined $\triangle y_\theta$, the vertical grid spacing, $g_{y,\theta}$ is computed as the minimum element of set $\triangle y_\theta$ defined in 4.5.

$$g_{y,\theta} \leq \min(\triangle y_\theta). \tag{4.6}$$

For the exceptional case where every elements of $\triangle\ x_\theta$ is equal, such as circular shape, the grid gaps are compared and the smaller gap is used for both horizontal and vertical grid spacing. This makes the required number of robot larger but ensures the stability of the object.

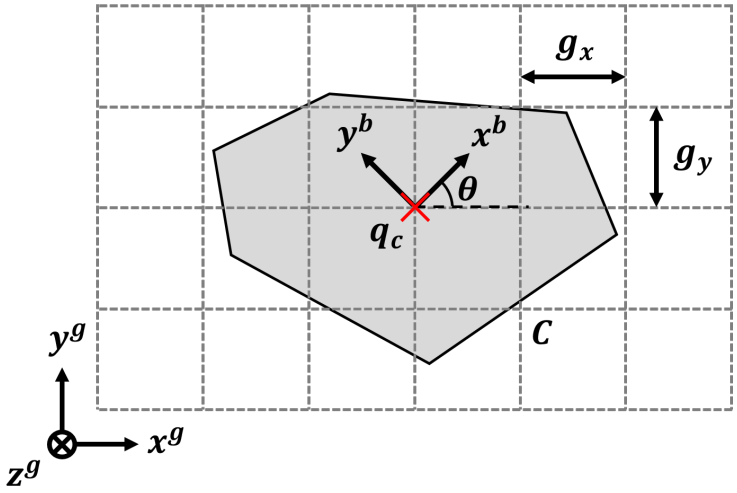


Fig. 4.2.: The final view of the generated grid for C shown in Fig. 3.1 with g_x and g_y.

We assume that the object is heavy enough to generate the frictional force and the transport process is slow enough to make no slip condition between each robot and the object surface. However, in reality we add an additional safety margin to each horizontal and vertical gaps, g_x and g_y to account for movement inertia of C and any slip that may occur during the transport process as,

$$G_x = \{g_{x,\theta}\},\ G_y = \{g_{y,\theta}\},\ \text{for } 0 \leq \theta < 2\pi \tag{4.7}$$

$$g_x = \min(G_x) - \delta x,\ g_y = \min(G_y) - \delta y \tag{4.8}$$

where δx and δy represent safety margins on the grid spacing design such that $g_x > \delta x \geq 0$ and $g_y > \delta y \geq 0$. The grid generation process is illustrated in Fig. 4.3.

With the computed grid properties, the grid is placed as shown in Fig. 4.2. At the starting position, the center of the mass, q_c is placed on any cross section of the grid. All The final result for the grid generation process is illustrated in Fig. 4.2

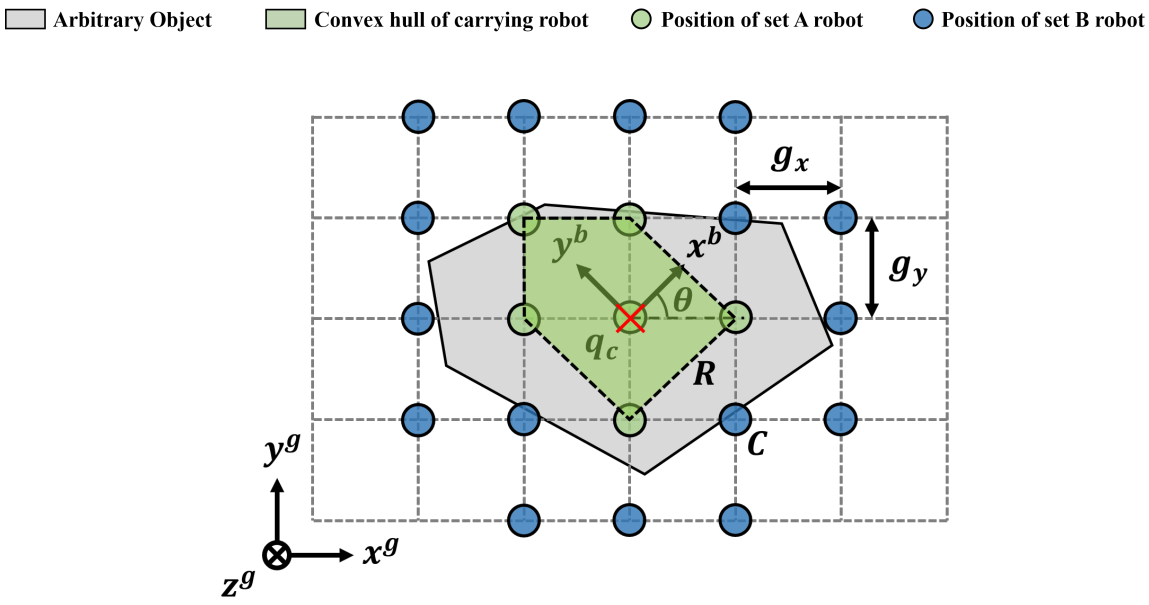


Fig. 4.3.: Final view of the grid generation process. including convex hull of carrying robot, R by set A robots and position of set B robots to satisfy the *Definition 2*.

4.2 Robot Allocation

The possible positions of the robots to ensure the stability of the object are all the grid intersections. The intersections under C is the allocation of the robots in set A. These

robots has direct relations of the speed to the speed of the translation. Because of this, a layer of robots is required on the planar movement direction to receive C as the object translates. To ensure the stability in omni-directional movement, extra layer of the robots is also allocated around set A. In anther words, apart from robots in set A, extra robots are allocated in every direction on the grid to surround the set A and these groups are set B.

Definition 2: At any given time all robots in A must have eight neighboring robots; two along $\pm g_x$, two along $\pm g_y$ and four at $\pm g_x$ and $\pm g_y$ diagonal locations from its current position on the grid. The layer of robots fulfilling this requirement and not in set A allows omni-directional translation of C and constitutes robots of set B.

Due to the difference in the transition velocity between robots and the object, the robots in set A falls to B and eventually be positioned in unnecessary position. To fulfill the *Definition 2*, these robots are re-allocated continuously around C. The re-allocation of the robot is mainly done by using position control with collision avoidance. The vision feedback is used for the control system to ensures the re-allocation of the robot. For carrying method, the most efficient method to transport the object is to use minimum number of robots to transport the object while ensuring the stability of the object. The ultimate goal of the Grid based Cyclic robot Allocation is to generate grid based on the object and define the minimum number of the robots. Therefore, the minimum number of robots required for stable omni-directional transport of C is defined as $N_{min} = |A \cup B|$. Fig. 4.3 illustrates robot allocation based on the set A or set B classification. The ultimate goal of the Grid based Cyclic robot Allocation is to generate

We identify robot dynamics and control system for self-organizing and re-allocation of the multi-robot at desired location to follow *Definition 2* as beyond the scope of the paper, but we conclude that a decentralized approach may be employed because of the cyclic nature of the proposed robot allocation method. At this stage, we focus on the robot allocation problem in two dimension that ensures stable omni-directional object transportation following *Definition 1*.

4.3 Stability Analysis

The successful grid spacing design ensures the transportation of the object efficiently using minimum number of robots while maintaining the stability of the object. The stability takes huge part in terms of success in transportation. In this section, we analytically investigate the stability of the proposed grid spacing design in transporting the planar shape C following the stability criterion presented in *Definition 1*.

Proposition: To ensure that the center of the mass of the obect, q_c is always bounded within the C, at any orientation θ of C, the minimum horizontal distance from q_c to $q_p \in C$, $\forall p = \{1, 2, .., m\}$ along the x^+ and along the x^- axis denoted as bg_x and ag_x respectively such that $a + b = \alpha$, q_c is always bounded within R, if $a \geq 0$ and $b \geq 0$.

Proof: The above setup yields the following conditions for stable object transportation along the x-axis:

$$\big(x_c + (\alpha - a)g_x\big) - x_c \geq 0, \tag{4.9}$$

$$x_c - (x_c - ag_x) \geq 0. \tag{4.10}$$

Based on our initial setup assumption that $q_c \notin C$ and $\delta x \geq 0$, the grid spacing parameter g_x is always $\mathbb{R}_{>0}$. Therefore, following Eq.4.9 we conclude that $\alpha \geq a$. Similarly, Eq. 4.10 suggests $a \geq 0$ and hence, $\alpha \geq 0$. With $\alpha \geq 0$ and $a \geq 0$, we conclude that $b \geq 0$. ■

The Eq.4.9 and Eq.4.10 is ensures the stability in x^g. The grid spacing parameter g_y is calculated based on g_x following Eq. (4.4) and (4.5). The stability of translation along the y-axis can be validated in a similar fashion. Since g_x and g_y are calculated based on the minimum distance to C from q_c for $0 \leq \theta < 2\pi$, the above proof holds true for omni-directional planar translation.

5. VALIDATION

In this chapter, the proposed solution is validated if it solves the stated problem in the previous chapter. To physically validate the proposed solution, multi-robot system consisting spherical robot is needed. The ideal option for the robot is Sphero. As shown in the Fig. 5.1,The robot consists multiple sensors such as accelerometer, gyroscope, compass, infrared receiver, infrared emitter, ambient light sensor and motor encoder. It is driven using two wheels and it is covered with polycarbonate shell. The robots are used in various purposes such as educational purpose for children for robotic-enhanced activities [34] and rehabilitation of children with autism in social and communication skills [35]. One of the reasons that the Sphero is suitable for the carrying method is ease of control. The control of the Sphero is not complicated and could be controlled using web browser for ease of control [36]. By using this control mechanism, Distributed Formation Control of Sphero robots are studied [37]. It is centralized system and uses camera to visualized the position of the robots.

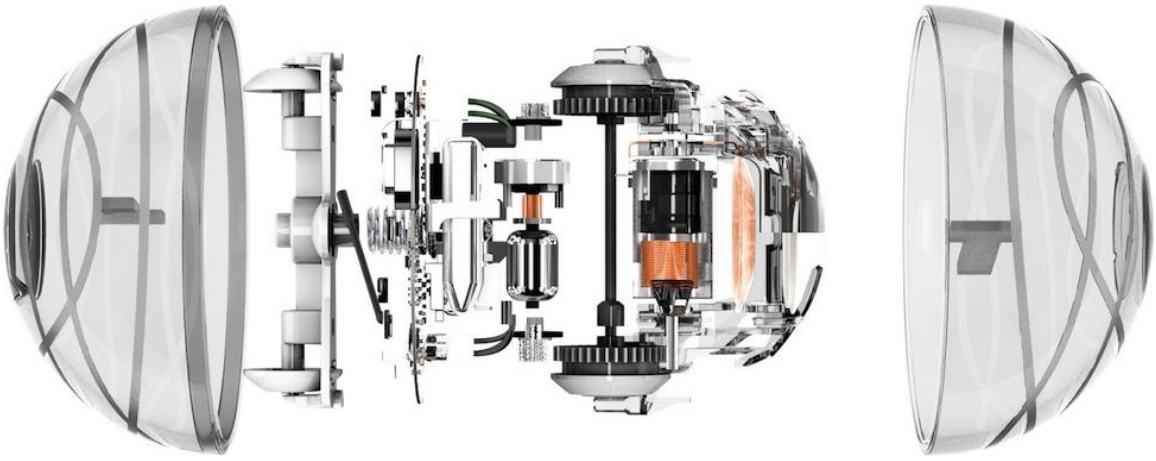


Fig. 5.1.: Spherical robot, Sphero consisting multi-sensor system with two wheel based driver. The image is downloaded from [38].

The material of the shell of the sphero is polycarbonate which has the maximum hardness of 396 MPa. This implies that maximum of 4038.08 kg/cm^2 can be sustained. Unlike the traditional strategies mentioned in Chapter 2, the limitation of the weight of the object is high. In addition, the well allocated robots distributes the weight of the object which highers the limitation of the weight. The mechanism of using multi-robot system is similar to that of swarm robotics. One of the advantage of this can be weight distribution and causes less impact on individual robots. Mechanism for load sharing using swarm robotics [39] can be applied in carrying method where multiple spherical robots carry the objects to minimize the stress on individual robots and maximize the limit of the weight of the object.

Moreover, internal sensors such as infrared receiver and emitter, accelerometer and gyroscope can be used to extend the multi-robot system to de-centralized system and make the transportation fully autonomous. Using sensor feedbacks, the motion controller for spherical robot can be developed with control moment [40]. Development of motion controller for a spherical robot can be used to allocate the robots in desired positions.

To validate the proposed solution, using multiple Spheros are not the most economical method. To visualize the robot allocation for object carrying, Matlab simulation is used. Total 6 different objects are generated and tested. Since the ultimate goal of the GCRA is to design the allocation of the robot that can move in omni direction, 4 different paths of the movement were applied and tested. The simulation is to validate if the object can be successfully transported to the desired position while maintaining the stability of the robot.

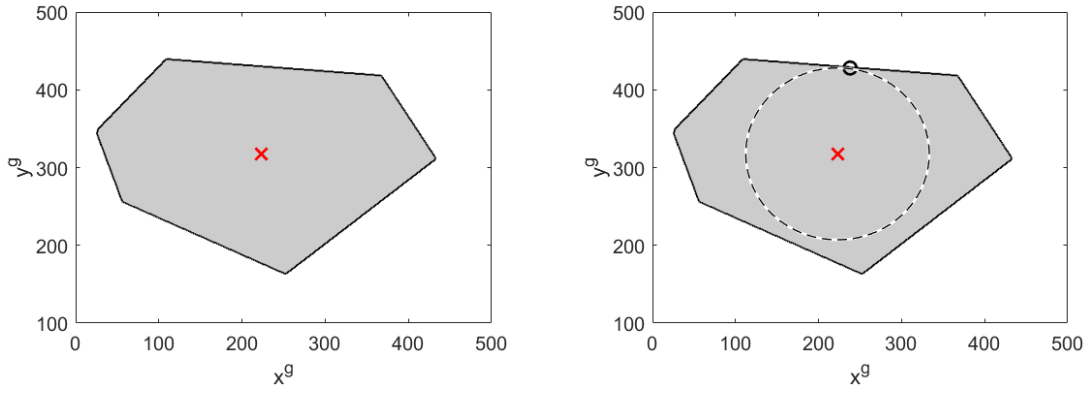
Both *Definition 1* and *Definition 2* are applied in the simulation. The results of the transportation is shown with the minimum distance from q_c to q_p $\forall p = \{1, 2, .., m\}$ and discussed in Section 5.2.

5.1 Setup

To validate the proposed concept of grid-based robot allocation, 4 different objects are used. To test the general grid generation method, the shape of objects are Triangular shape

($C1$), rectangular shape ($C2$)and arbitrary shape ($C3$) shown in Fig. 3.1 are used. To test the exceptional method, the circular shape ($C4$) was used. The objects are shown in Fig. .

The objects are represented with high resolution boundary points in C and known center of mass, q_c for each case. In this case, the center of mass is assumed to be equally positioned as that of the centroid of the object.



(a) Step1: Define the shape of the object, $C3$ and the center of the mass, q_c.

(b) Step2: Determine the shortest distance between the center of the mass, q_c and q_p and define it as g_x.

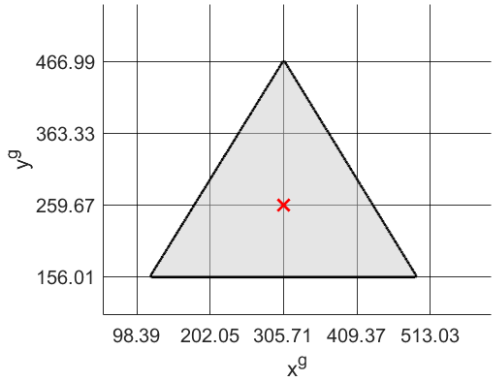
(c) Step3: Set of $d_{p,\theta}$ that is group of elements of $\triangle x_\theta$ that is equal to the g_x.

(d) Step4: For every elements of $d_{p,\theta}$, the shortest distance from the element to the q_c represents g_y.

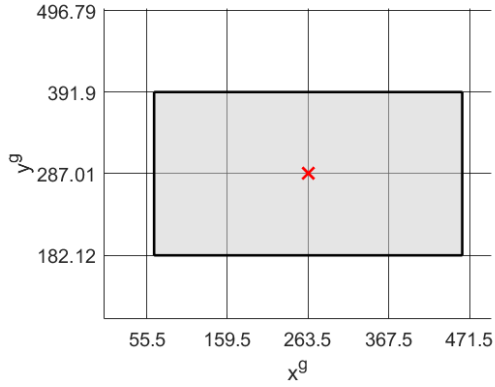
Fig. 5.2.: Steps of grid generation procedure for the object $C3$ using Matlab.

Using the proposed grid generation method, the grid parameters g_x and g_y are calculated and the intersections of the grids are assumed to host spherical robots either under set A or set B for each of the cases $C1$, $C2$, $C3$ and $C4$. To visualize the simulated result of the grid generation procedure, the Fig. 5.2 describes the proposed solution with the same

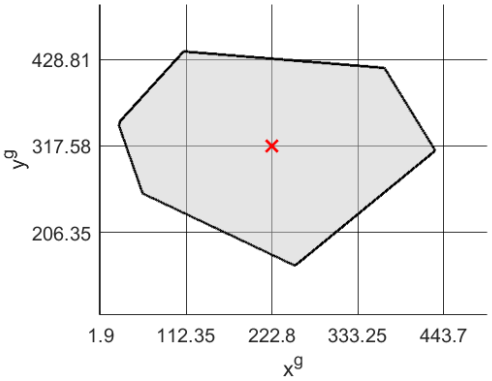
figure shown in Fig. 3.1. All the object $C1$, $C2$, $C3$ and $C4$ follows the same steps and the final grid including g_x and g_y for every object is shown in Fig. 5.3.



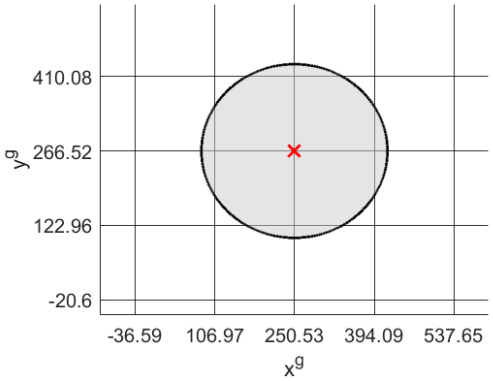
(a) $C1$: Triangular shape with q_c=(305.71, 259.67) and $g_x = 103.66$, $g_y = 103.66$.



(b) $C2$: Rectangular shape with q_c=(263.5, 287.01) and $g_x = 104.00$, $g_y = 104.88$.



(c) $C3$: Arbitrary shape with q_c=(222.8, 317.58) and $g_x = 110.45$, $g_y = 111.23$.



(d) $C4$: Circular shape with q_c=(250.53, 266.52) and $g_x = 143.56$, $g_y = 143.56$.

Fig. 5.3.: Generated grid based on the shape of the object $C1$, $C2$, $C3$ and $C4$ using Matlab.

We assume that for each of the cases, the object translates at a constant velocity $v_0 = 1$ $units/iteration$, slow enough such that no slipping occurs at the contact areas between each robot and the object. Since the object dynamics and inertia are not considered at this stage, the grid parameters g_x and g_y are calculated with $\delta x = 0$ and $\delta y = 0$ safety margins for the presented simulations.

The velocity of the object travels twice the speed of the R. Therefore, the relative velocity of the object is v_0 to he R. For ease of view, the simulation display the relative velocity, v_0.

For validation purposes, the minimum distance from q_c to $q_p \in C$, $\forall p = \{1, 2, .., m\}$ denoted as $s_{c,R}$ is measured at every iteration instance for each of the $C1$, $C2$, $C3$ and $C4$ cases. Whenever the stability of the object breaks, the simulation automatically stops.

5.2 Simulation

Based on the computed horizontal gap, g_x and vertical gap, g_y, the grid was generated as shown in Fig. 5.3. With the generated grid, the robots are positioned at the required position as shown in Fig. 5.5. The general proposed solution is applied for $C1$, $C2$, $C3$ and the exceptional proposed solution is applied for $C4$. In this chapter, the object transportation was simulated to validate if the stability of the object is satisfied. All the objects are following the same path $y_g = 200sin(x_g/200)$ where $0 \leq x_g \leq 400pi$ as shown in Fig. 5.4.

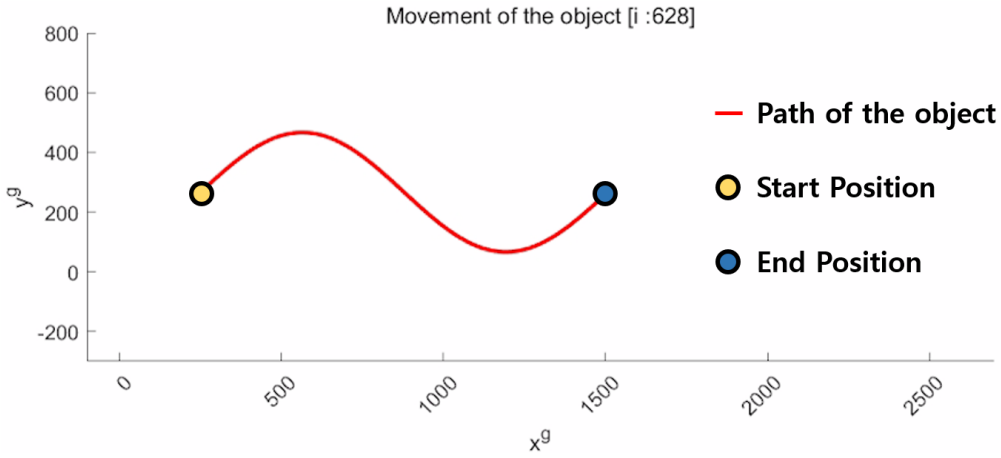
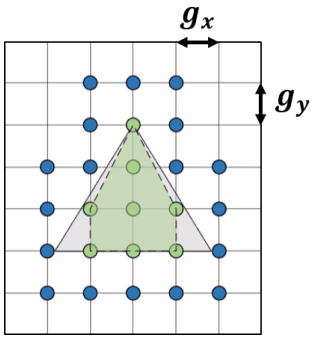


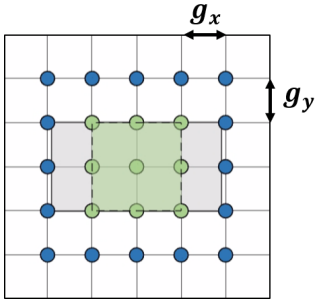
Fig. 5.4.: Path of the object's transportation. All objects follow the path, $y_g = 200sin(x_g/200)$.

Sinusoidal path was to demonstrate the transportation because it generates wide range of direction. This is to verify the robustness of the suggested solution. In the previous

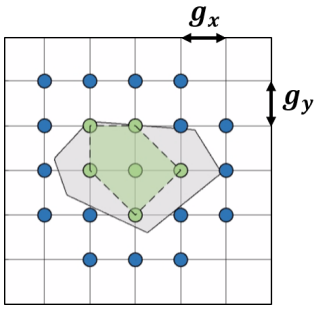
work of development of the GCRA [33], the circular shape $C4$ was able to be transported in 1-Dimensional path. However, To increase the degree of freedom of the movement, 2-Dimensional path was applied and found that the 4 can not travel in omni-direction using the general solution. Therefore, increasing the dimension of the desired path of the transportation enhanced the quality of the study.



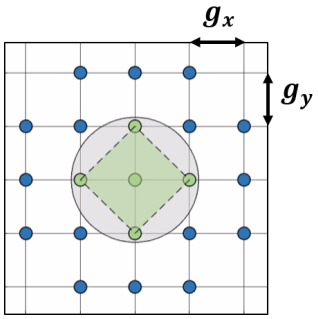
(a) Starting allocation for $C1$ with q_c=(305.71, 259.67) and $g_x = 103.66$, $g_y = 103.66$.



(b) Starting allocation for $C2$ with q_c=(263.5, 287.01) and $g_x = 104.00$, $g_y = 104.88$.



(c) Starting allocation for $C3$ with q_c=(222.8, 317.58) and $g_x = 110.45$, $g_y = 111.23$.



(d) Starting allocation for $C4$ with q_c=(250.53, 266.52) and $g_x = 143.56$, $g_y = 143.56$

Fig. 5.5.: Starting position of robots including set$A$ and set$B$.

5.2.1 General solution

The general solution is applied for $C1$, $C2$ and $C3$ where the $\forall(|q_p - q_c|) \neq g_x$ where $\forall p = \{1, 2, .., m\}$. This indicates that C is not the set of points that is equidistant,g_x from

Table 5.1.: Grid spacing parameters for validation cases

Case	g_x	g_y	N_{min}	**Solution Type**
$C1$: Triangular shape	103.66	103.66	26	General
$C2$: Rectangular shape	104.00	104.88	25	General
$C3$: Arbitrary shape	110.45	111.23	22	General
$C4$: Circular shape	143.56	143.56	21	Exceptional

q_c. The transportation was shown in 5 different screen capture and shown in every 126 iterations.

Object $C1$, $C2$ and $C3$ were successfully transported as shown in the Fig. 5.8 through Fig. 5.13. The stability at any iteration was maintained to be stable. However, using general solution to transport the object $C4$ fails at iteration 162 as shown in Fig. 5.14.

The result of the transportation can also be validated based on the Stability analysis stated at the Section 4.3. To validate if the transportation satisfy the *Definition 1*, Fig. 5.17(a) through Fig. 5.17(c) can be used to defined the stability during the transportation.

5.2.2 Exceptional solution

The general solution is applied for $C4$ where the $\forall(|q_p - q_c|) = g_x$ where $\forall p = \{1, 2, .., m\}$. This indicates that C is the set of points that is equidistant,g_x from q_c. The transportation was shown in 5 different screen capture and shown in every 126 iterations.

Unlike the first approach of using general solution as shown in Fig5.14 , Object $C4$ were successfully transported as shown in the Fig. 5.15 and Fig. 5.16. The stability at any iteration was maintained to be stable.

The result of the transportation can also be validated based on the Stability analysis stated at the Section 4.3. To validate if the transportation satisfy the *Definition 1*, Fig. 5.18(a) can be used to defined the stability during the transportation.

5.3 Discussion

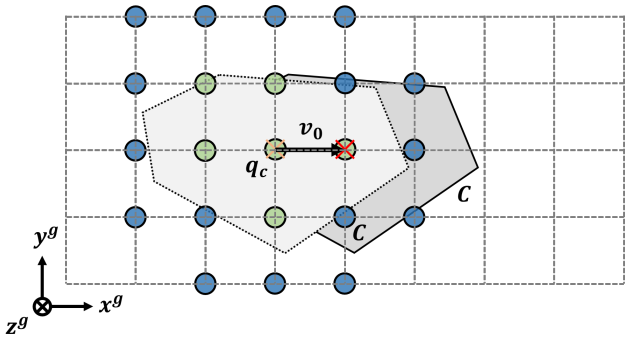
For general solution, $C1$ was able to be transported from start position to the end position as shown in Fig. 5.4. The snapshots of transportation is shown in Fig. 5.8 and Fig. 5.9. The maximum number of robots used during the transportation was 26, which is same as the number of robots used at the starting position. During the transportation in the sinusoidal path, definite pattern of the allocation was not found. This can be seen from the Fig. 5.17(a) where the repetition of the graph can not be found. From the Table.5.1, the object $C1$ has same g_x and g_y even though it followed the general solution. Therefore, it can be concluded that Exceptional solution is not the only solution that generates the same g_x and g_y.

In case of $C2$, the object was also transported successfully ensuring the stability of the object. This can be found in Fig. 5.10 and Fig. 5.11. Similar to case of $C1$, the maximum number of robots used during the transportation is same as the number of robots used at the starting position. However, unlike $C1$, definite pattern of the allocation was found and it can also be seen from the Fig. 5.17(b) where same pattern of the graph from $i = 0$ to $i = 314$ is repeated.

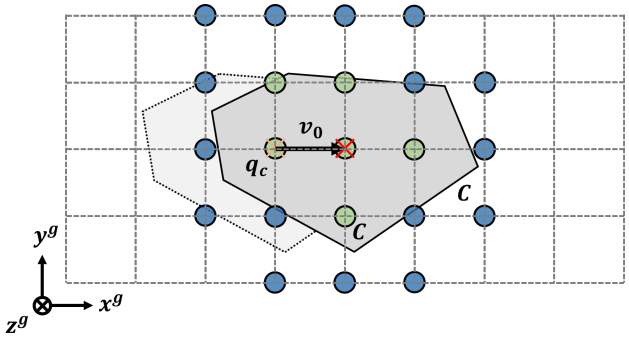
Similar to $C2$, the object $C3$ successfully transitioned as shown in Fig. 5.12 and Fig. 5.13. The maximum number of robots used while transporting was 30, which requires 8 more robots from the starting allocation. Like $C1$, definite pattern of the allocation was not found as shown in Fig. 5.17(c).

In case of $C4$, which is the only object that used exceptional solution, the transportation was done successfully while maintaining the stability of the object as shown in Fig. 5.15 and Fig. 5.16. The maximum number of robots used during the transportation was 21, which is the same number of robots at the starting allocation. It is hard to find the definite pattern of the allocation as shown in Fig. 5.18. By comparing the Fig. 5.18(a) and Fig. 5.18(b), it can be clearly seen that the $s_{c,R}$ drops below 0 and breaks the stability of the object when $i = 162$. When $C4$ was tested with the general solution, same number of robot was used but the stability was broken when $i = 162$, which can be seen at Fig. 5.14.

Therefore, it can be concluded that the number of the robot does not directly impact the stability of the object. However, the grid's properties are the parameters that is related to the stability of the object.



(a) Position 1 of transition of the object with position of set$A$ and B.

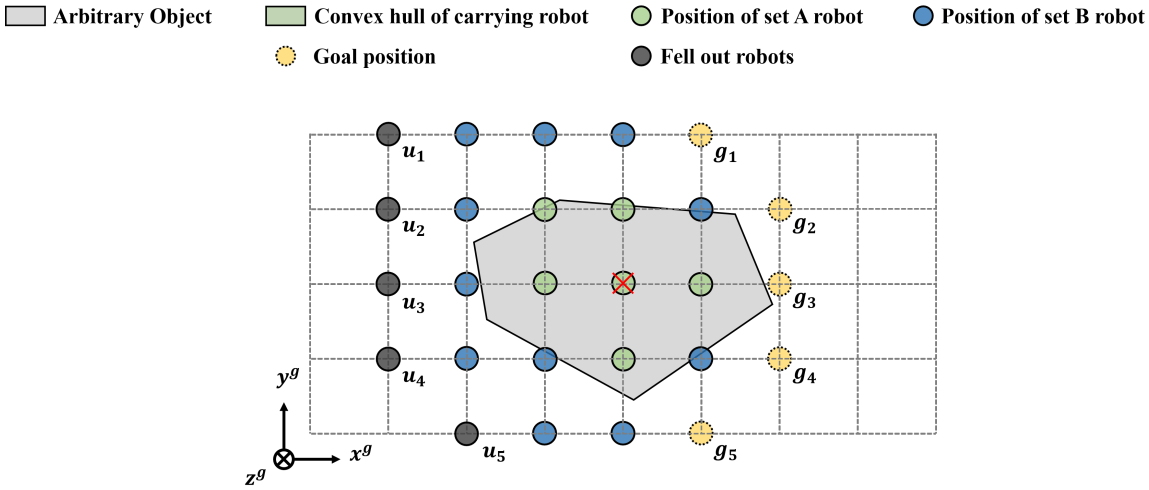


(b) Position 2 of transition of the object with new position of set$A$ and B.

Fig. 5.6.: Transition of the object with velocity of v_0.

As the results states, all the objects were transported successfully using properly assigned solutions. Moreover, from the Fig. 5.17, it can be found that the number of robot used has direct relationship with the $s_{c,R}$. This implies that if the access of the number spherical robots is limited, the starting allocation does not always ensures the sufficient number of robots which can affect the stability of the object. However, N_{min} from the Table.5.1 sufficiently satisfy the number of robots for set$A$. Therefore, the possibility of the instability due to insufficient number of robot can be solved by formation control or position control of the robots.

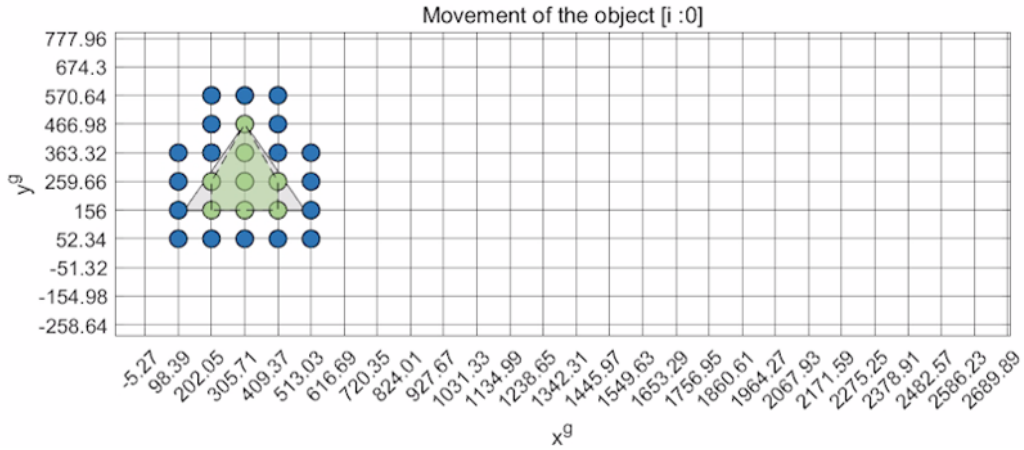
There are various ways to control the formation of the robot such as, using multi-layering system to control the formation [41] and using leader and follower communication system to control the multi-agent robots [42]. For the object carrying method using GCRA, the quick cover of the fell out robots due to difference in travel speed enhances the quality of the formation control. One of the possible options is to use Hungarian method to control the position of the robots to manage the formation. To show this, let's consider the transition of the object as shown in Fig. 5.6. Once the object is moved to the new location, there are fell out robots as shown in Fig. 5.7



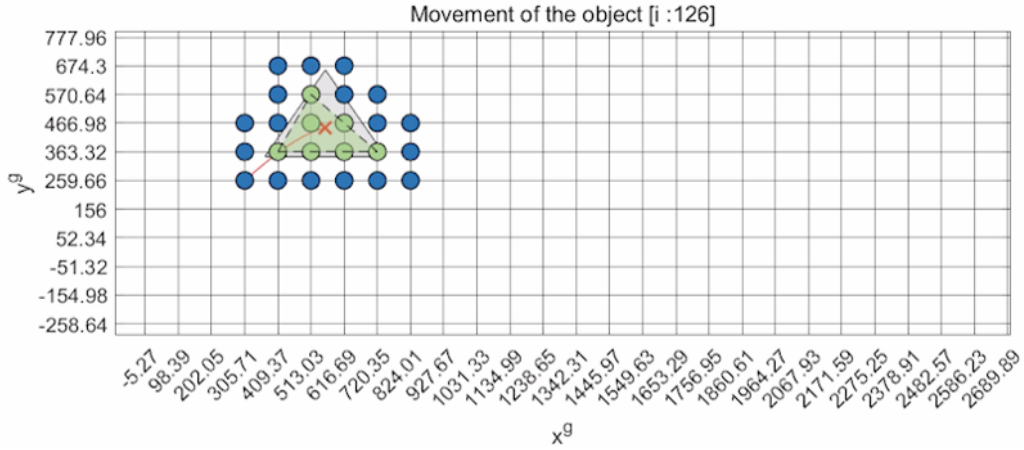
(a) After the transition of the object, required position of the robots to satisfy *Definition 2*.

Fig. 5.7.: Formation control to to ensure the stability of the object while transporting.

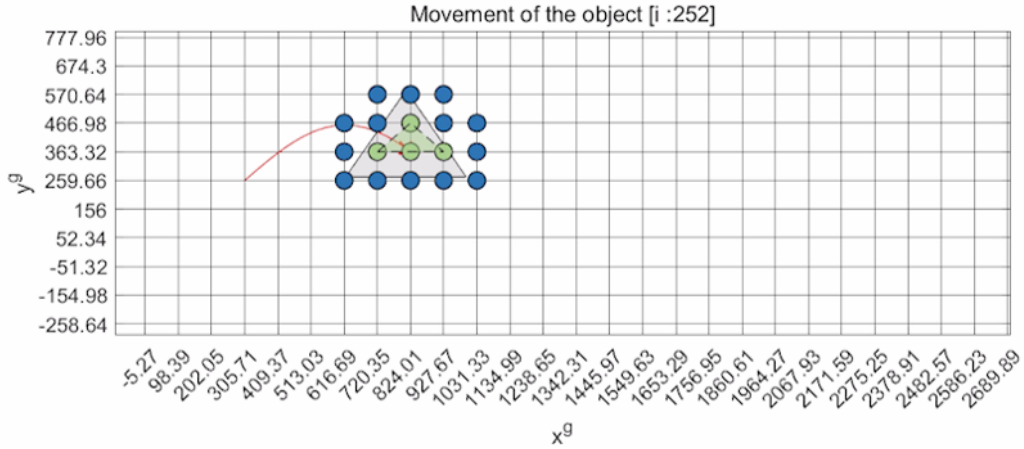
Once the object is located in new position, to satisfy the *Definition 2*, the set B robots are required to re-allocate. The goal of the new allocation of he robot can be computed as shown seen in Fig. 5.7. One of the possible solution to fulfil the goal positions is to use position control fell out robots to the goal positions. Another option is that all robots in set B can relocate to fulfil the goal positions. By moving all the robots in set B by one unit, and fulfil the position one by one will eventually maintain the required formaion. In the extended study, a control system to verify the efficient method of formation control will be created and tested.



(a) Transportation of $C1$: iteration = 0, stability of the object = Stable.

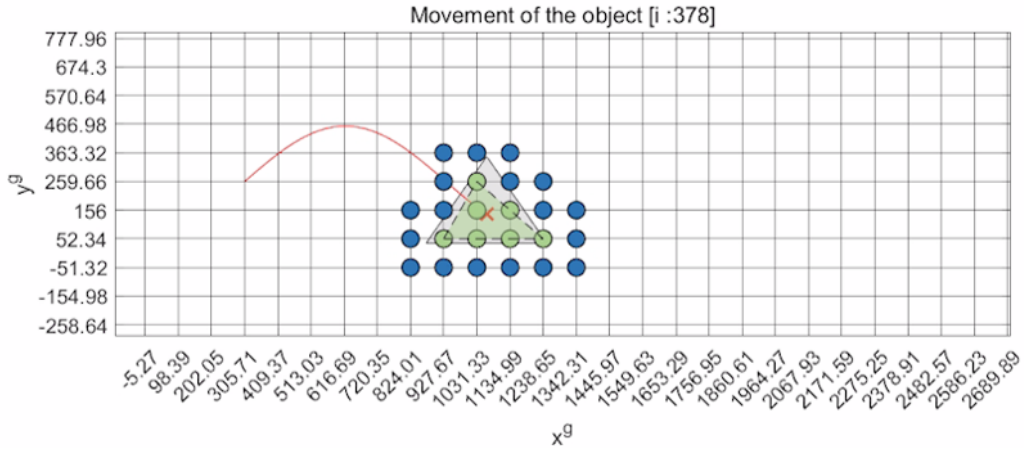


(b) Transportation of $C1$: iteration = 126, stability of the object = Stable.

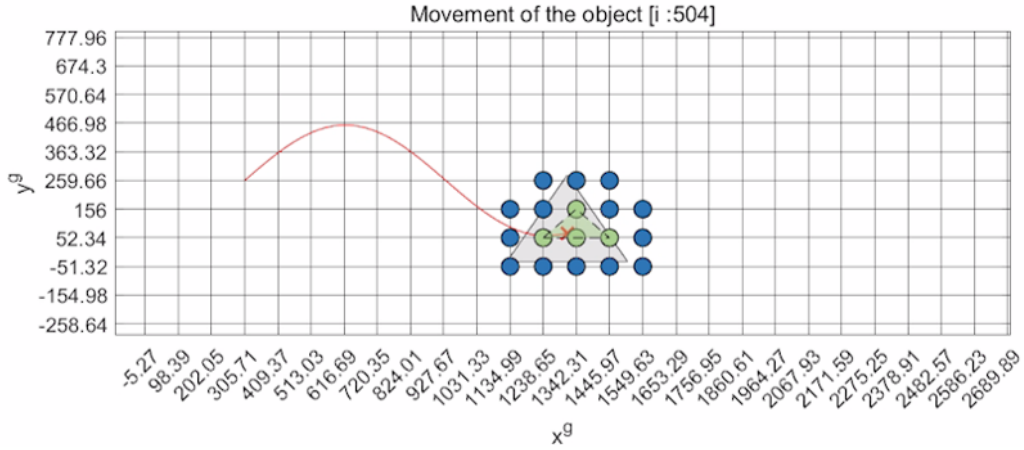


(c) Transportation of $C1$: iteration = 252, stability of the object = Stable.

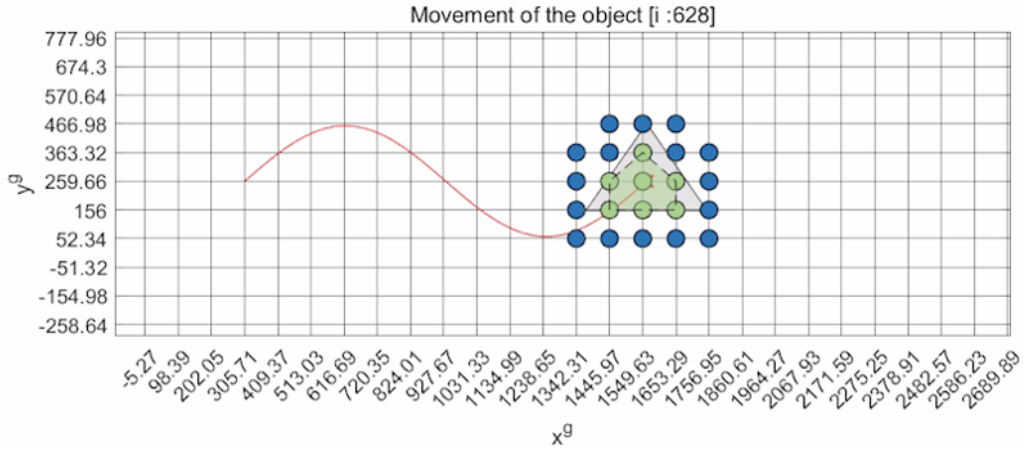
Fig. 5.8.: Transportation of $C1$. The iteration range is: $0 \leq i \leq 252$.



(a) Transportation of $C1$: iteration = 378, stability of the object = Stable.

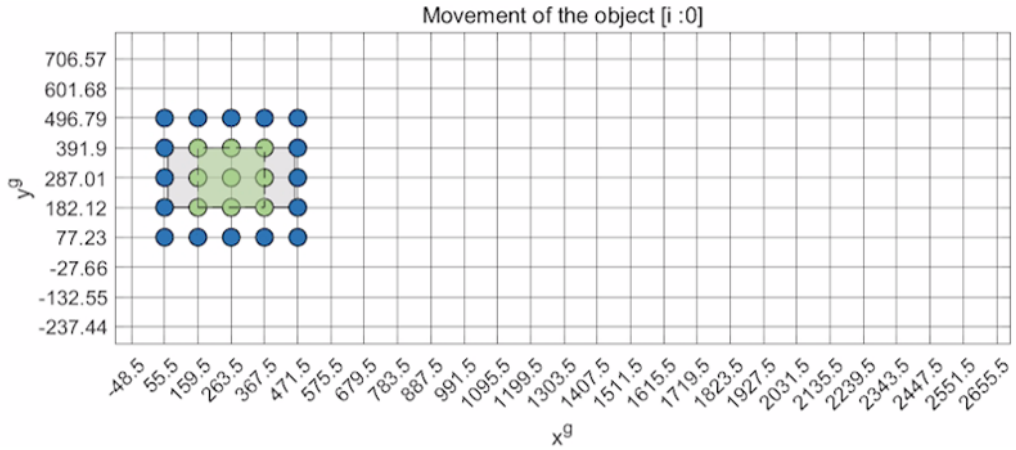


(b) Transportation of $C1$: iteration = 504, stability of the object = Stable.

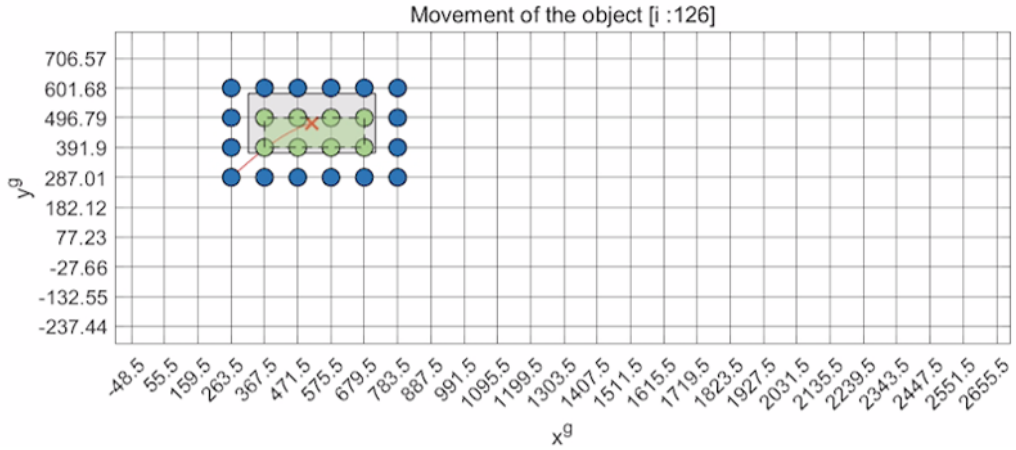


(c) Transportation of $C1$: iteration = 627, stability of the object = Stable.

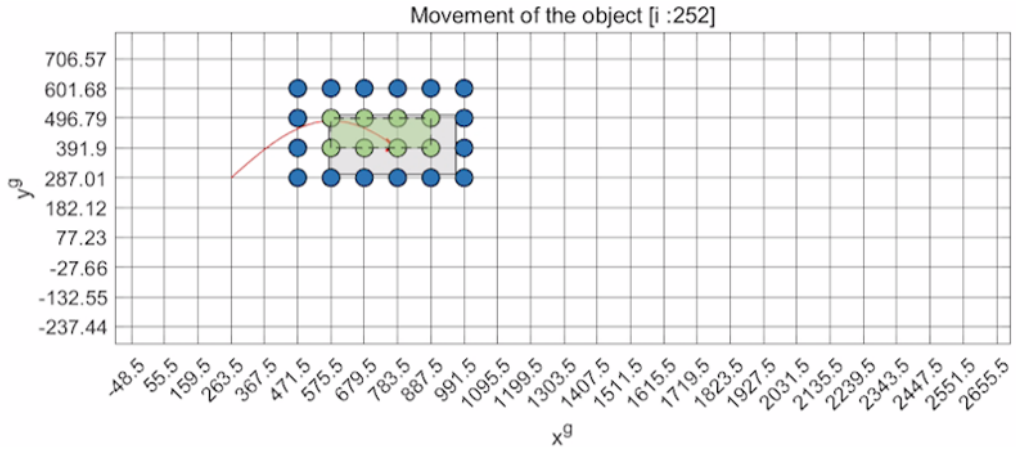
Fig. 5.9.: Transportation of $C1$. The iteration range is: $252 < i \leq 628$.



(a) Transportation of $C2$: iteration = 0, stability of the object = Stable.

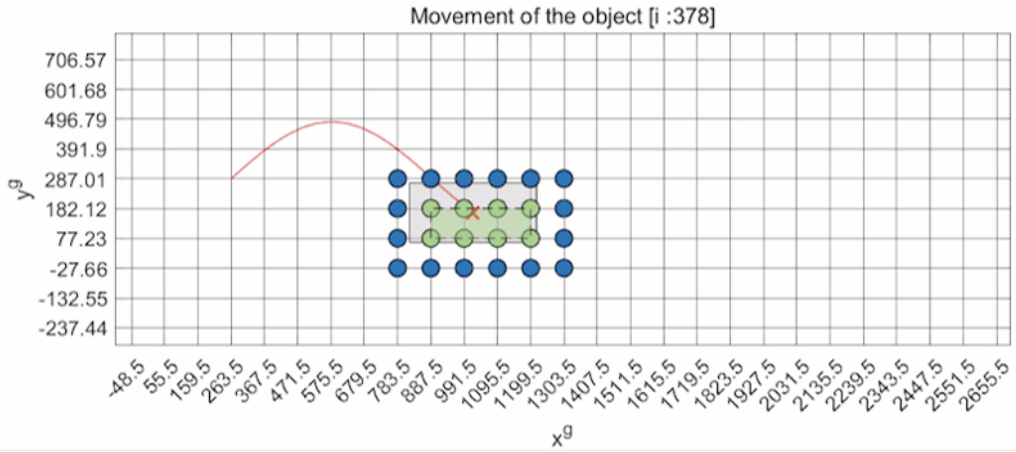


(b) Transportation of $C2$: iteration = 126, stability of the object = Stable.

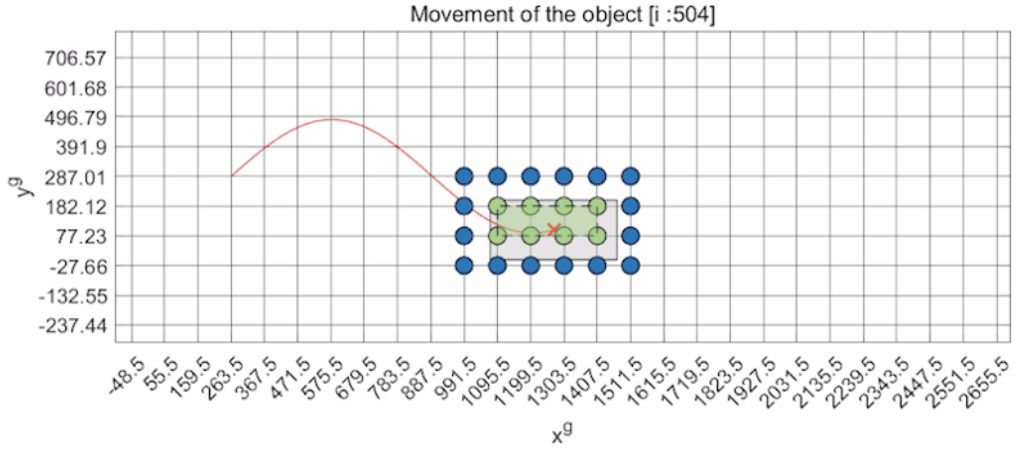


(c) Transportation of $C2$: iteration = 252, stability of the object = Stable.

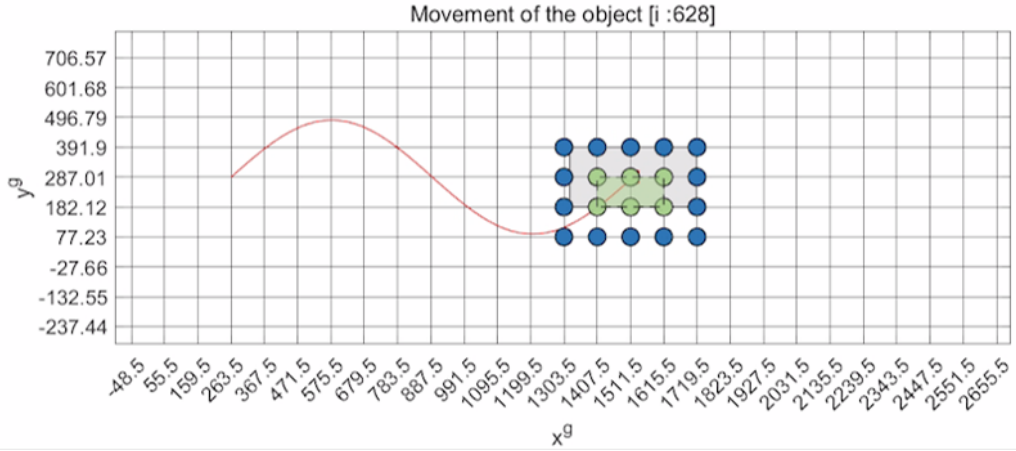
Fig. 5.10.: Transportation of $C2$. The iteration range is: $0 \leq i \leq 252$.



(a) Transportation of $C2$: iteration = 378, stability of the object = Stable.

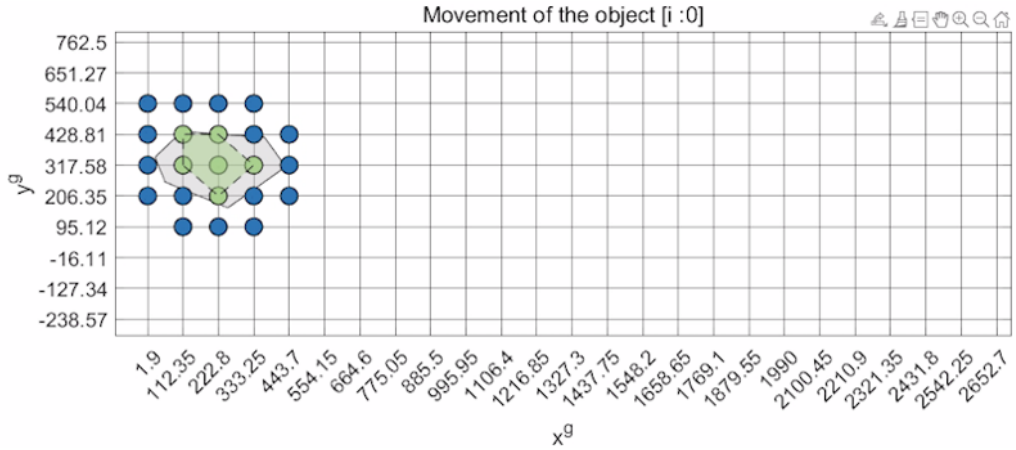


(b) Transportation of $C2$: iteration = 504, stability of the object = Stable.

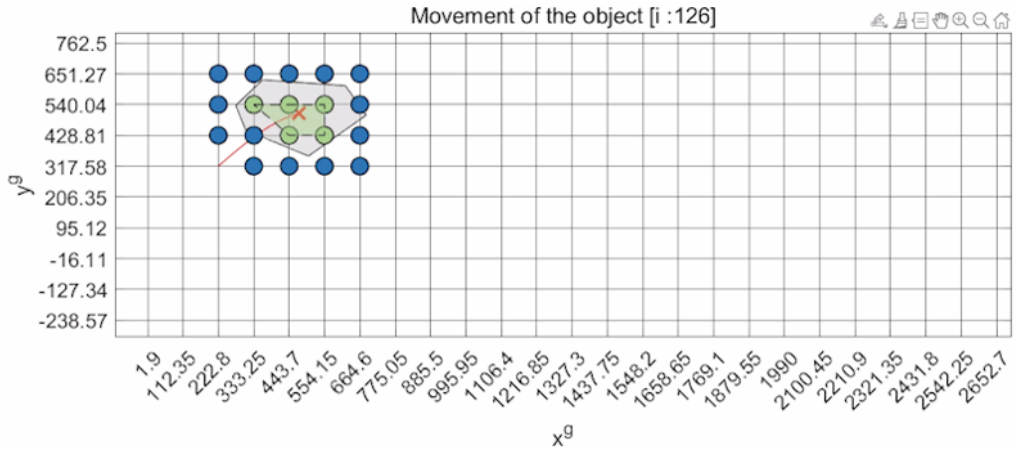


(c) Transportation of $C2$: iteration = 627, stability of the object = Stable.

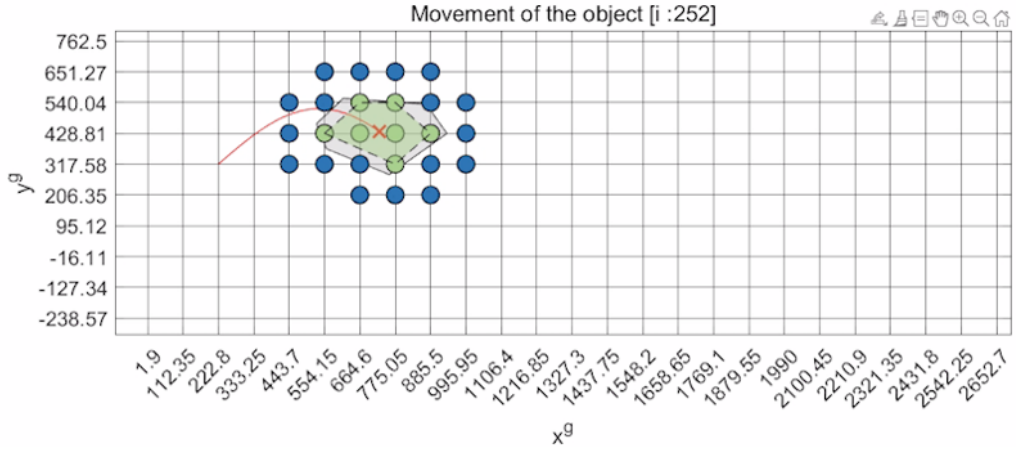
Fig. 5.11.: Transportation of $C2$. The iteration range is: $252 < i \leq 628$.



(a) Transportation of $C3$: iteration = 0, stability of the object = Stable.

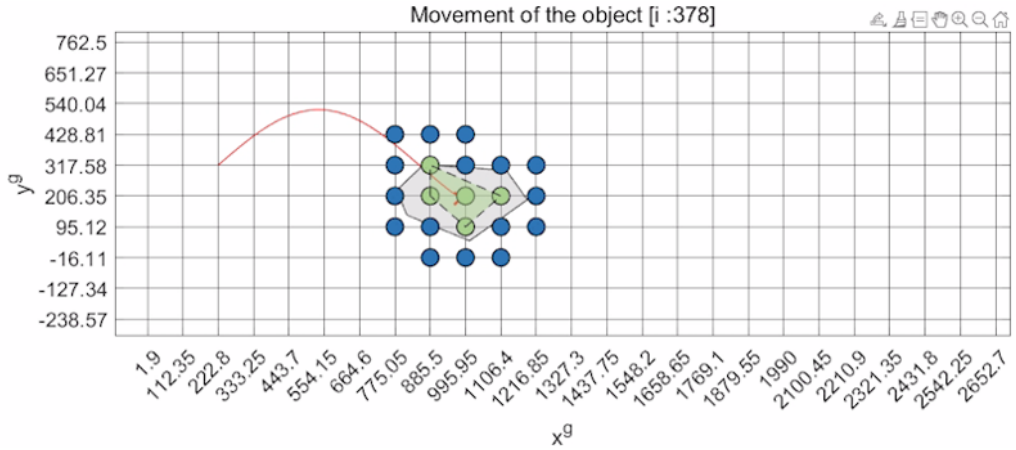


(b) Transportation of $C3$: iteration = 126, stability of the object = Stable.

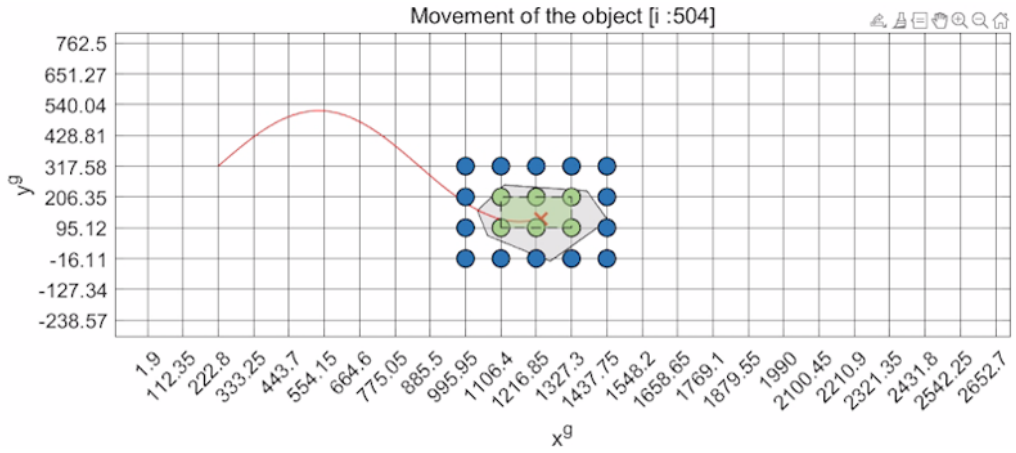


(c) Transportation of $C3$: iteration = 252, stability of the object = Stable.

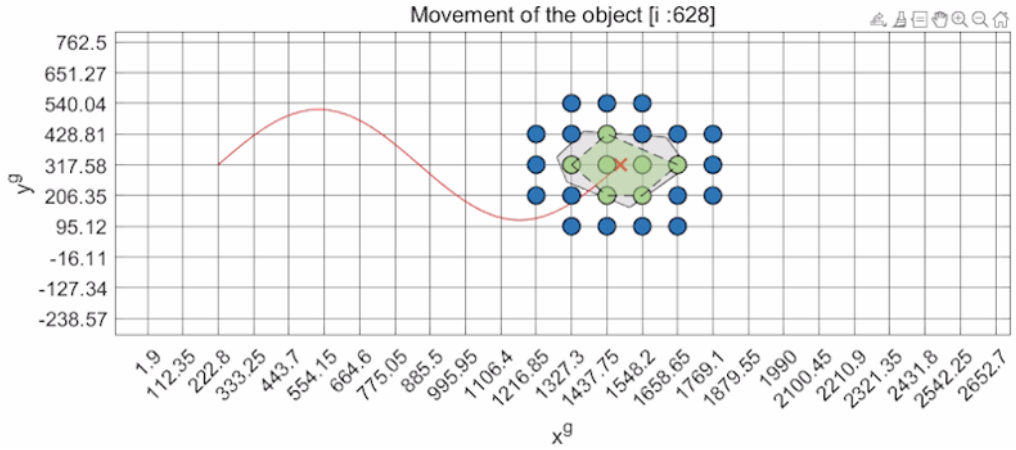
Fig. 5.12.: Transportation of $C3$. The iteration range is: $0 \leq i \leq 252$.



(a) Transportation of $C3$: iteration = 378, stability of the object = Stable.

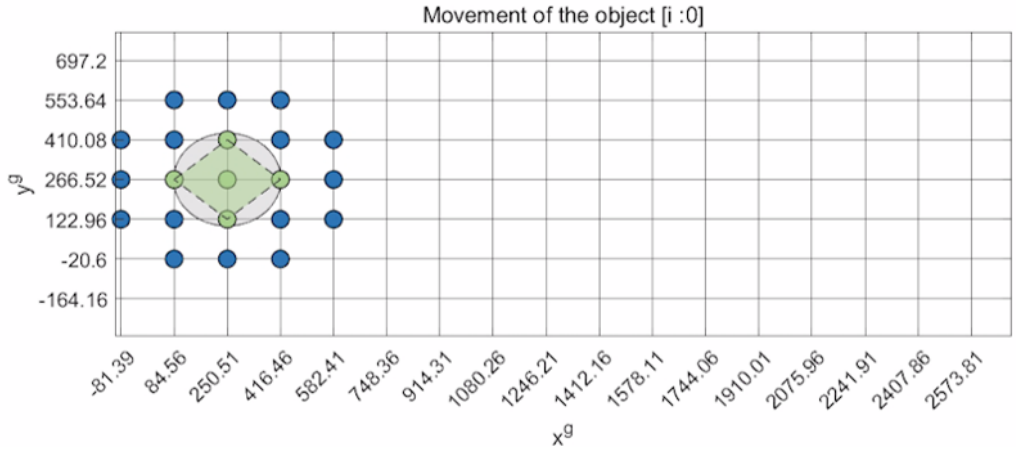


(b) Transportation of $C3$: iteration = 504, stability of the object = Stable.

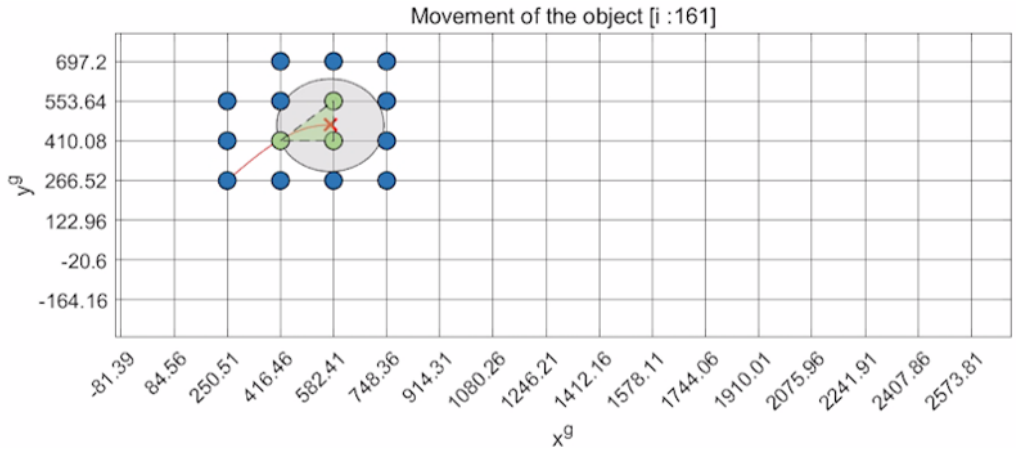


(c) Transportation of $C3$: iteration = 627, stability of the object = Stable.

Fig. 5.13.: Transportation of $C3$. The iteration range is: $252 < i \leq 628$.

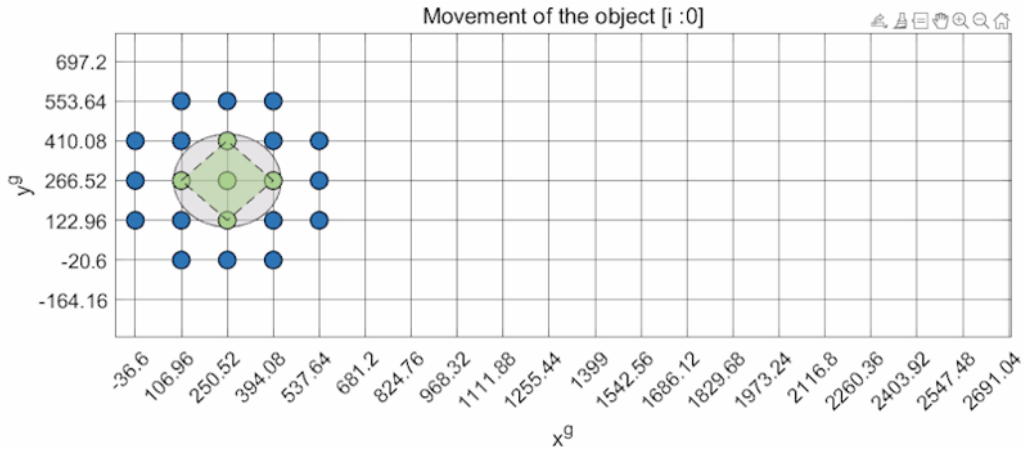


(a) Transportation of $C4$: iteration = 0, stability of the object = Stable.

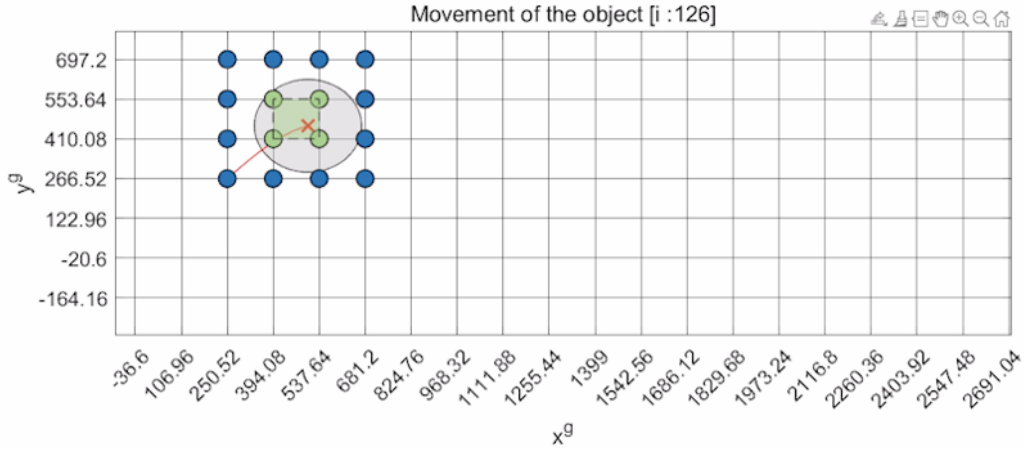


(b) Transportation of $C4$: iteration = 162, stability of the object = UnStable.

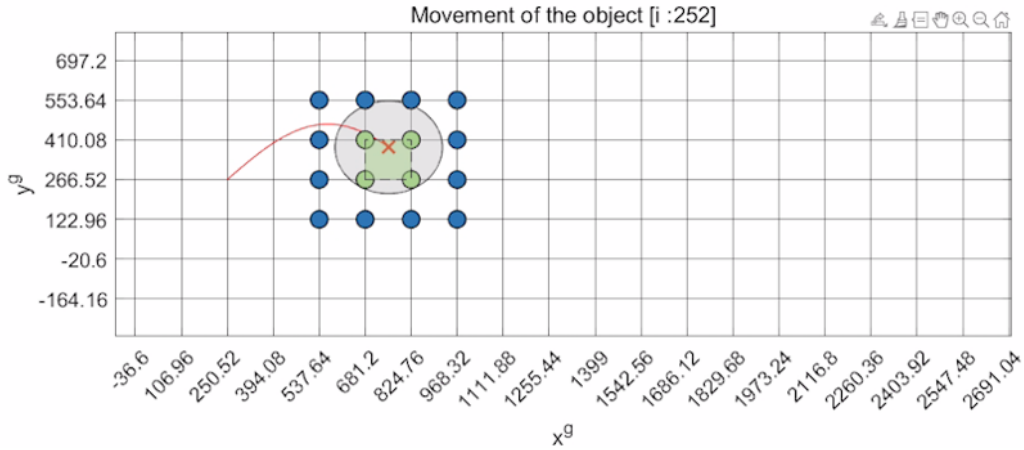
Fig. 5.14.: Transportation of $C4$ fails. The iteration range is: $0 < i \leq 162$.



(a) Transportation of $C4$: iteration = 0, stability of the object = Stable.

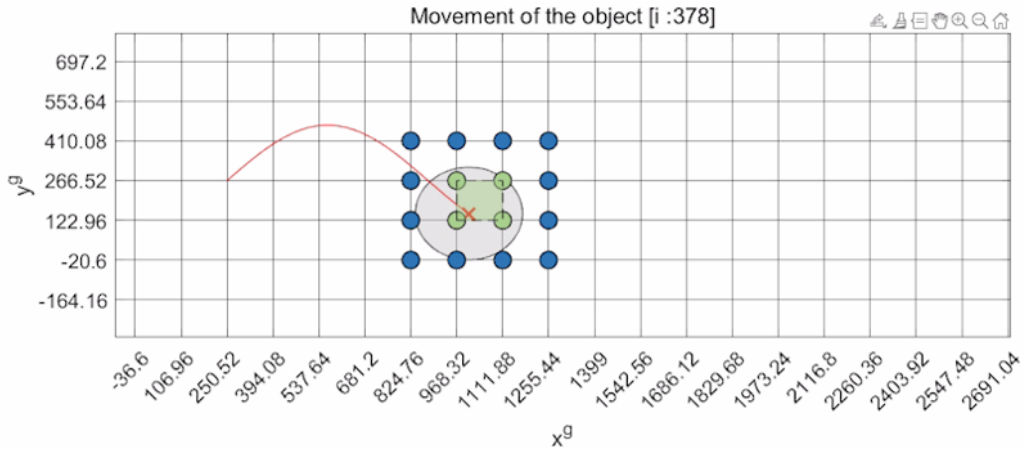


(b) Transportation of $C4$: iteration = 126, stability of the object = Stable.

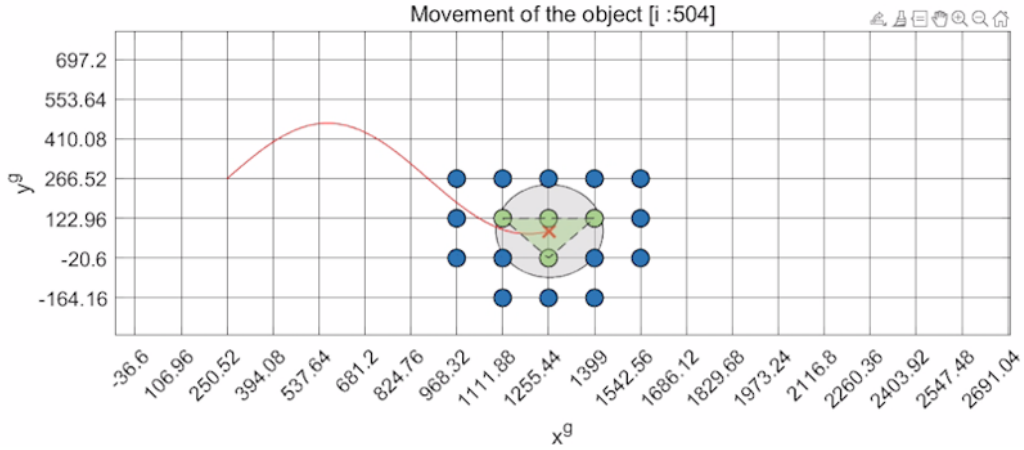


(c) Transportation of $C4$: iteration = 252, stability of the object = Stable.

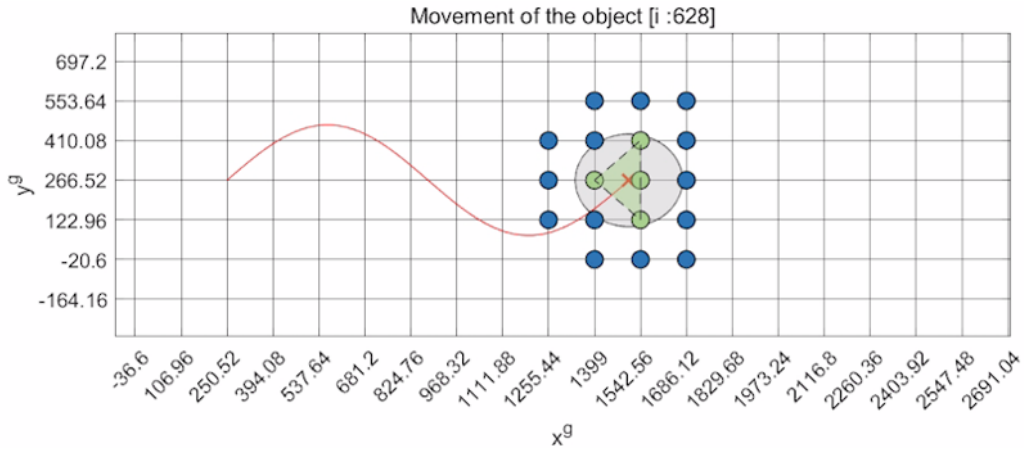
Fig. 5.15.: Transportation of $C4$. The iteration range is: $0 \leq i \leq 252$.



(a) Transportation of $C4$: iteration = 378, stability of the object = Stable.

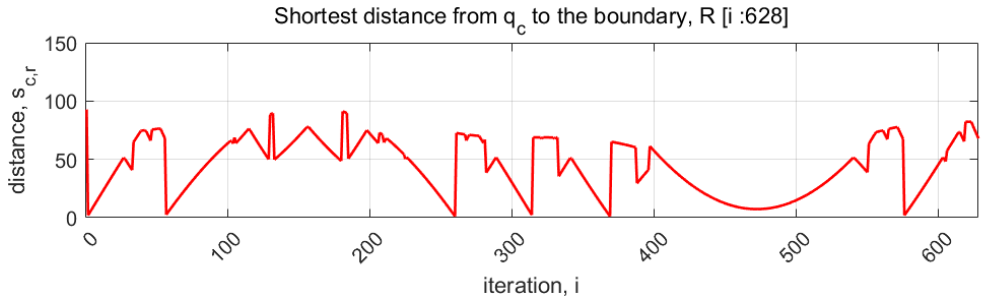


(b) Transportation of $C4$: iteration = 504, stability of the object = Stable.

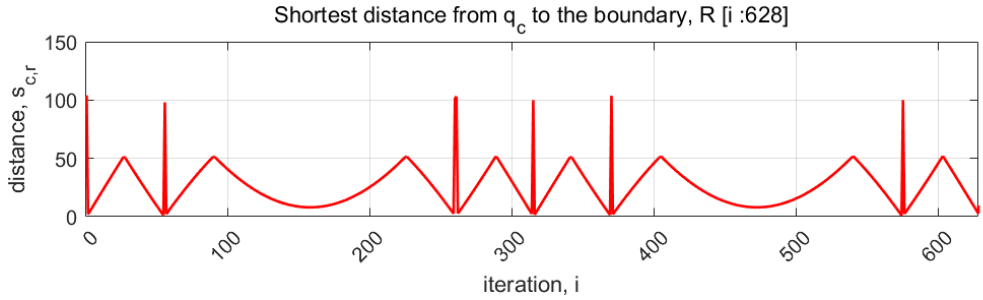


(c) Transportation of $C4$: iteration = 627, stability of the object = Stable.

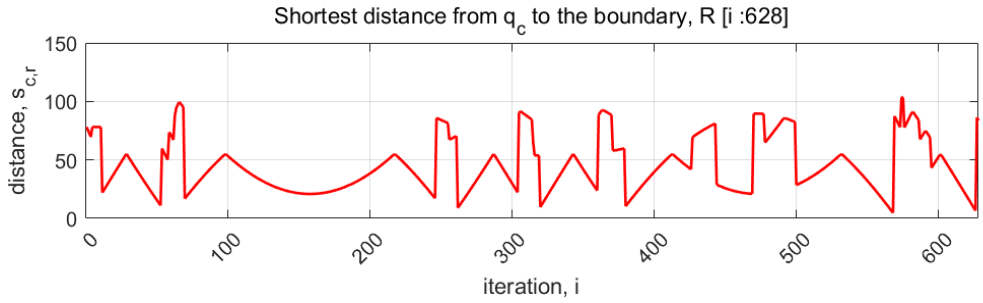
Fig. 5.16.: Transportation of $C4$. The iteration range is: $252 < i \leq 628$.



(a) Object $C1$: Shortest distance from q_c to R.

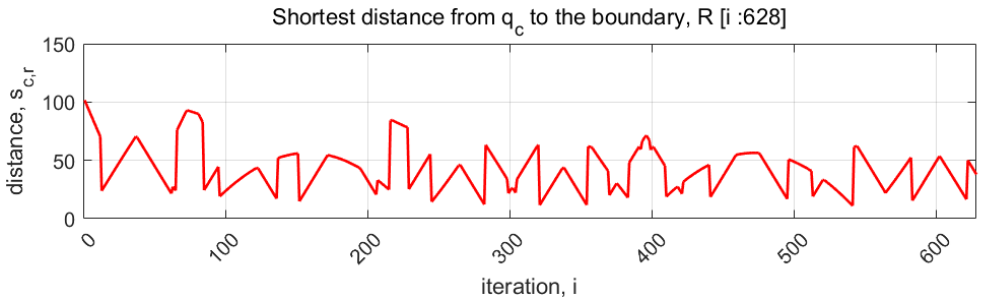


(b) Object $C2$: Shortest distance from q_c to R.

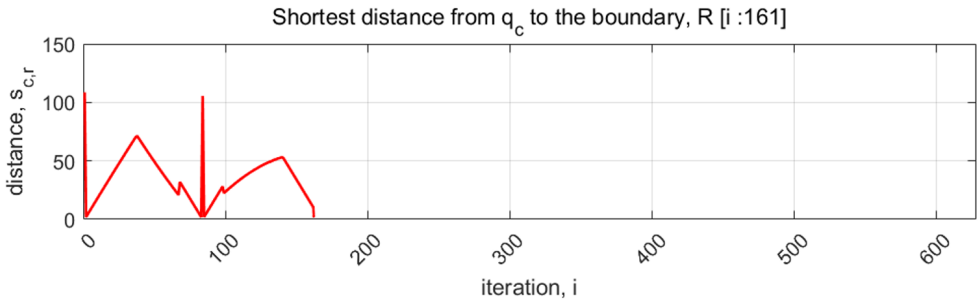


(c) Object $C3$: Shortest distance from q_c to R.

Fig. 5.17.: Successful Transportation: Shortest distance from q_c to R. The iteration range: $0 < i \leq 628$.



(a) Object $C4$ (Excpetional Solution): Shortest distance from q_c to R.



(b) Object $C4$ (General Solution): Shortest distance from q_c to R.

Fig. 5.18.: $C4$ Transportation: Comparison of shortest distance from q_c to R. The iteration range: $0 < i \leq 628$.

6. CONCLUSION AND FUTURE WORKS

6.1 Conclusion

In this thesis, innovative method of transporting an object is introduced and to efficiently transport the object, Grid-based Cyclic Robot Allocation (GCRA) method is introduced. To solve the problem of maintaining the stability of the object while carrying the object, general and exceptional solutions are proposed and validated under centralized system. The proposed solutions computes the grid properties such as horizontal spacing, g_x and vertical spacing, g_y. To ensure and to analyze the stability of the object, *Definition 1* and *Definition 2* were implemented and stability of the object carrying with the proposed robot allocation methodology is analytically proven. To validate the proposed solutions, 4 objects are generated and tested using the Matlab simulation. The transportation of all objects are described and shown in Fig. 5.8 through Fig. 5.16. To validate the stability of the object while transporting, distance between the center of the mass to the boundary of the objects are measured and plotted. Simulation results validate the proposed concept illustrating the object centroid always bounded by the convex hull of the allocated carrying robots. The proposed object carrying method using a multi-robot system of spherical robots requires further investigation on cost and feasibility analysis depending on robot design. However, referring to the long history of successful object transportation by log rollers, we believe it is a viable alternative to the current methods of grasping, pushing or caging.

6.2 Future Works

With successful multi-robot allocation, further work on developing multi-robot formation control will be studied and applied. The formation control system is to ensure the

stability of the object while transporting the object with limited access of the robots. By re-positioning the fell out robots to the required positions to control the formation can ensure the stability with minimum number of robots required. With the successful development of the control system, the validation can be extended to the real-life experiment or 3-dimensional simulation environment. Using computed N_{min} number of robots to transport the object is the ultimate goal of the study. To do so, static and dynamic modeling of the spherical robots and building a centralized system to apply the control system using GCRA to the spherical robot is the future work of the study.

REFERENCES

REFERENCES

[1] A. Khasawneh, H. Rogers, J. Bertrand, K. C. Madathil, and A. Gramopadhye, "Human adaptation to latency in teleoperated multi-robot human-agent search and rescue teams," *Automation in Construction*, vol. 99, pp. 265 – 277, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0926580517310890

[2] J. J. Roldán, P. Garcia-Aunon, M. Garzón, J. De León, J. Del Cerro, and A. Barrientos, "Heterogeneous multi-robot system for mapping environmental variables of greenhouses," *Sensors*, vol. 16, no. 7, 2016. [Online]. Available: https://www.mdpi.com/1424-8220/16/7/1018

[3] M. S. S. M. Koch, P., "Multi-robot localization and mapping based on signed distance functions," *Intell Robot Syst*, vol. 83, no. 7, 2016. [Online]. Available: https://doi.org/10.1007/s10846-016-0375-7

[4] L. Liu, C. Luo, and F. Shen, "Multi-agent formation control with target tracking and navigation," in *2017 IEEE International Conference on Information and Automation (ICIA)*, 2017, pp. 98–103.

[5] J. Chen, M. Gauci, and R. Groß, "A strategy for transporting tall objects with a swarm of miniature mobile robots," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 863–869.

[6] B. Gerkey and M. Matarić, "Pusher-watcher: An approach to fault-tolerant tightly-coupled robot coordination," vol. 1, p. 464–469, 2002.

[7] Z. Wang and M. Schwager, "Kinematic multi-robot manipulation with no communication using force feedback," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 427–432.

[8] W. Wan, B. Shi, Z. Wang, and R. Fukui, "Multirobot object transport via robust caging," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 1, pp. 270–280, 2020.

[9] T. Huntsberger, G. Rodriguez, and P. S. Schenker, *Robotics Challenges for Robotic and Human Mars Exploration*, pp. 340–346. [Online]. Available: https://ascelibrary.org/doi/abs/10.1061/40476%28299%2945

[10] H. Ma, X. Wu, Y. Gong, Y. Cui, and J. Song, "A task-grouped approach for the multi-robot task allocation of warehouse system," in *2015 International Conference on Computer Science and Mechanical Automation (CSMA)*, 2015, pp. 277–280.

[11] J. E. Inglett and E. J. Rodríguez-Seda, "Object transportation by cooperative robots," in *SoutheastCon 2017*, 2017, pp. 1–6.

[12] S. A. H. El-Ayat, Khaled Committe Member: Elkassas, "Cooperative transport in swarm robotics. multi object transportation," in *SoutheastCon 2017*, 2018, pp. 1–6.

[13] J. L. Farrugia and S. G. Fabri, "Swarm robotics for object transportation," in *2018 UKACC 12th International Conference on Control (CONTROL)*, 2018, pp. 353–358.

[14] J. Chen, M. Gauci, and R. Groß, "A strategy for transporting tall objects with a swarm of miniature mobile robots," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 863–869.

[15] J. Alonso-Mora, S. Baker, and D. Rus, "Multi-robot formation control and object transport in dynamic environments via constrained optimization," *The International Journal of Robotics Research*, vol. 36, no. 9, pp. 1000–1021, 2017.

[16] W. Wan, B. Shi, Z. Wang, and R. Fukui, "Multirobot object transport via robust caging," *IEEE transactions on systems, man, and cybernetics: systems*, no. 99, pp. 1–11, 2017.

[17] Z. Wang, G. Yang, X. Su, and M. Schwager, *OuijaBots: Omnidirectional Robots for Cooperative Object Transport with Rotation Control Using No Communication*. Cham: Springer International Publishing, 2018, pp. 117–131. [Online]. Available: https://doi.org/10.1007/978-3-319-73008-0_9

[18] Z. Wang, S. Singh, M. Pavone, and M. Schwager, "Cooperative object transport in 3d with multiple quadrotors using no peer communication," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 1064–1071.

[19] V. A. Joshi, R. N. Banavar, and R. Hippalgaonkar, "Design and analysis of a spherical mobile robot," *Mechanism and Machine Theory*, vol. 45, no. 2, pp. 130–136, 2010.

[20] V. Kaznov, F. Bruhn, P. Samuelsson, and L. Stenmark, "Ball robot," Jan. 17 2012, uS Patent 8,099,189.

[21] A. Tagliabue, M. Kamel, R. Siegwart, and J. Nieto, "Robust collaborative object transportation using multiple mavs," *The International Journal of Robotics Research*, vol. 38, no. 9, pp. 1020–1044, 2019. [Online]. Available: https://doi.org/10.1177/0278364919854131

[22] J. Alonso-Mora, S. Baker, and D. Rus, "Multi-robot formation control and object transport in dynamic environments via constrained optimization," *The International Journal of Robotics Research*, vol. 36, no. 9, pp. 1000–1021, 2017. [Online]. Available: https://doi.org/10.1177/0278364917719333

[23] E. Tuci, M. H. Alkilabi, and O. Akanyeti, "Cooperative object transport in multi-robot systems: A review of the state-of-the-art," *Frontiers in Robotics and AI*, vol. 5, p. 59, 2018.

[24] D. J. Stilwell and J. S. Bay, "Toward the development of a material transport system using swarms of ant-like robots," in *[1993] Proceedings IEEE International Conference on Robotics and Automation*. IEEE, 1993, pp. 766–771.

[25] P. J. Johnson and J. S. Bay, "Distributed control of simulated autonomous mobile robot collectives in payload transportation," *Autonomous Robots*, vol. 2, no. 1, pp. 43–63, 1995.

[26] G. A. Pereira, B. S. Pimentel, L. Chaimowicz, and M. F. Campos, "Coordination of multiple mobile robots in an object carrying task using implicit communication," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 1. IEEE, 2002, pp. 281–286.

[27] C. C. Loh and A. Traechtler, "Cooperative transportation of aload using nonholonomic mobile robots," *Procedia Engineering*, vol. 41, pp. 860–866, 2012.

[28] M. B. M. D. Manuele Brambilla, Eliseo Ferrante, "Swarm robotics: a review from the swarm engineering perspective," *Swarm Intelligence*, vol. 7, pp. 1–41, 2013.

[29] G. B. Chandran and M. Geetha, "A force couple approach for cooperative object pushing of rod object with multiple robots," in *2018 3rd IEEE International Conference on Recent Trends in Electronics, Information Communication Technology (RTEICT)*, 2018, pp. 1989–1995.

[30] G. Habibi, Z. Kingston, W. Xie, M. Jellins, and J. McLurkin, "Distributed centroid estimation and motion controllers for collective transport by multi-robot systems," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 1282–1288.

[31] ZhiDong Wang, Y. Hirata, and K. Kosuge, "Control a rigid caging formation for cooperative object transportation by multiple mobile robots," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, vol. 2, 2004, pp. 1580–1585 Vol.2.

[32] G. A. S. Pereira, M. F. M. Campos, and V. Kumar, "Decentralized algorithms for multi-robot manipulation via caging," *The International Journal of Robotics Research*, vol. 23, no. 7-8, pp. 783–795, 2004. [Online]. Available: https://doi.org/10.1177/0278364904045477

[33] J. H. Park, T. Mina, and B. Min, "Grid-based cyclic robot allocation for object carrying," in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, 2019, pp. 140–145.

[34] M. Ioannou and T. Bratitsis, "Teaching the notion of speed in kindergarten using the sphero sprk robot," in *2017 IEEE 17th International Conference on Advanced Learning Technologies (ICALT)*, 2017, pp. 311–312.

[35] S. Golestan, P. Soleiman, and H. Moradi, "Feasibility of using sphero in rehabilitation of children with autism in social and communication skills," in *2017 International Conference on Rehabilitation Robotics (ICORR)*, 2017, pp. 989–994.

[36] M. Kohana and S. Okamoto, "Sphero control system using a web browser," in *2014 17th International Conference on Network-Based Information Systems*, 2014, pp. 606–610.

[37] K. Fathian, T. H. Summers, and N. R. Gans, "Robust distributed formation control of agents with higher-order dynamics," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 495–500, 2018.

[38] S. Education, *Sphero robot components in 3D view*, 2010. [Online]. Available: https://sphero.com/products/sphero-sprk-plus

[39] R. I. B. Ashokkumar1, M. Danny Frazer, "Implementation of load sharing using swarm robotics," vol. 3, no. 3, pp. 114–120, 2016.

[40] J. Chen, P. Ye, H. Sun, and Q. Jia, "Design and motion control of a spherical robot with control moment gyroscope," in *2016 3rd International Conference on Systems and Informatics (ICSAI)*, 2016, pp. 114–120.

[41] D. Li, S. S. Ge, W. He, G. Ma, and L. Xie, "Multilayer formation control of multi-agent systems," *Automatica*, vol. 109, p. 108558, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109819304194

[42] T. Han, Z.-H. Guan, M. Chi, B. Hu, T. Li, and X.-H. Zhang, "Multi-formation control of nonlinear leader-following multi-agent systems," *ISA Transactions*, vol. 69, pp. 140 – 147, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S001905781730424X

APPENDICES

A. MATLAB 2-D GRID-BASED CYCLIC ROBOT ALLOCATION SIMULATION CODE

A.1 Main program

```
% This is the main Matlab file to simulate the object transportation
% using sphero. The spheros will carry the object and move to desired
% goal point.
% The object needs to be defined prior of running the file.
%
% Title : Grid-based Cyclic Robot Allocation for Object Carrying
% Writer: Jee Hwan Park

clear; clc; close all;
addpath('geom2d')
addpath('object')
warning('off','all')
warning

%% Define object properties:
I = imread('object5_circle.png'); % Define object
[obj_boundary, centroid] = ObjectDefine(I);

%% Compute the gap distance:
gap = GapDefine(obj_boundary, centroid);

% For exceptional solution:
% g_x = gap(1);
% g_y = gap(2);
% if (g_x < g_y)
```

```
%     gap(2) = g_x;
% else
%     gap(1) = g_y;
% end

%% Draw grid:
Drawgrid(obj_boundary, centroid, gap);

%% Simulating Transportion of the object:
% Followings are the path options. Please enable desired path (ordered)
% Path1: y = 2x
% Path2: y = -x
% Path3: y = 1
% Path4: x = 1
% Path5: y = sin(x)
path_choice = [0 0 0 0 1];

Simulation(obj_boundary, centroid, gap, path_choice);
```

A.2 Defining object

```
function [obj_boundary, centroid_send] = ObjectDefine (I)
% Defines objects by reading the image file
% The function returns following objects:
% 1. Object boundary
% 2. Centroid of the object
%

%% Reads the image:

BW = im2bw(I);

%% Compute the object boundary:
dim = size(BW);
```

```
col = round(dim(2)/2) - 90;
row = min(find(BW(:,col)));
obj_boundary = bwtraceboundary(BW,[row, col],'N'); % Return value

%% Get the geometry of the object (centroids)
[ geom, iner, cpmo ] = polygeom(obj_boundary(:,1), obj_boundary(:,2));
cent_x = geom(2); cent_y = geom(3);

%% Final reurn:
centroid_send = [cent_x, cent_y];

end
```

A.3 Defining horizontal and vertical spaces g_x and g_y

```
function [gap] = GapDefine(obj_boundary, centroid)
% Define vertical and horizontal gap g_x, g_y
% The function returns following objects:
% 1. gap = [g_x, g_y]
%
% * Returns shortest g_x and g_y
%

%% Global declaration:
obj_size = size(obj_boundary);
g_x = inf;
g_y = inf;
polyin = polyshape(obj_boundary(:,1), obj_boundary(:,2));
poly_size = size(polyin.Vertices(:,1));
obj_boundary_size= size(obj_boundary);
safety_gap = 1;
figure(3)
plot(obj_boundary(:,1), obj_boundary(:,2), '-k', 'LineWidth',2); hold on;
fill(obj_boundary(:,1), obj_boundary(:,2), [0.8 0.8 0.8]);
```

```
%% Compute g_x:
for i=1:obj_size(1)
    points = [obj_boundary(i,1),obj_boundary(i,2) ; centroid(1)
    ,centroid(2)];
    d = pdist(points,'euclidean');
    if d < g_x
        g_x = d;
        min_x_x = obj_boundary(i,1);
        min_x_y = obj_boundary(i,2);
    end
end

%% Compute g_y (precise but long):
for i=1:(obj_boundary_size(1)+round(obj_boundary_size(1)*0.2))
    if i > obj_boundary_size(1)
        i = i - obj_boundary_size(1);
    end
    p1 = [obj_boundary(i,1), obj_boundary(i,2)];
    for j=(i+1):(obj_boundary_size(1)+round(obj_boundary_size(1)*0.2))
        if j > obj_boundary_size
            j = j - obj_boundary_size(1);
        end
        p2 = [obj_boundary(j,1),obj_boundary(j,2)];
        d = round(pdist([p1; p2],'euclidean'));
        if d >= round(g_x)
            distance = point_to_line_distance([centroid(1), centroid(2)]
            , p1, p2);
            lineseg = [p1;p2];
            [in,out] = intersect(polyin,lineseg);
            check = size(out);
            %plot([p1(1),p2(1)] , [p1(2),p2(2)],'-k');
            if check(1) == 0
                if distance < g_y
```

```
                    g_y = distance;
                    min_y_x = p1;
                    min_y_y = p2;
                end
            end
            break
        end
    end
end
for i=1:(obj_boundary_size(1)+round(obj_boundary_size(1)*0.2))
    if i > obj_boundary_size(1)
        i = i - obj_boundary_size(1);
    end
    p1 = [obj_boundary(i,1), obj_boundary(i,2)];
    for j=(obj_boundary_size(1)-i):0-round(obj_boundary_size(1)*0.2)
        if j < 0
            j = obj_boundary_size(1) + j;
        end
        p2 = [obj_boundary(j,1),obj_boundary(j,2)];
        d = round(pdist([p1; p2],'euclidean'));
        if d >= round(g_x)
            distance = point_to_line_distance([centroid(1), centroid(2)]
            , p1, p2);
            lineseg = [p1;p2];
            [in,out] = intersect(polyin,lineseg);
            check = size(out);
            %plot([p1(1),p2(1)] , [p1(2),p2(2)],'-k');
            if check(1) == 0
                if distance < g_y

                    g_y = distance;
                    min_y_x = p1;
                    min_y_y = p2;
                end
```

```
            end
            break
        end
    end
end

%% Final return:
gap = [g_x-safety_gap, g_y-safety_gap];

%% Object plot:

plot(centroid(1), centroid(2),'rx','Linewidth',2, 'markersize',13);
%plot(min_x_x, min_x_y,'ko','Linewidth',2, 'markersize',10);
%plot([min_y_x(1),min_y_y(1)] , [min_y_x(2),min_y_y(2)],'-r', 'Linewidth'
,2);
%viscircles([centroid(1), centroid(2)],gap(1), 'Linewidth',1, 'LineStyle'
, '--', 'Color','k');
xlim([0 500]); ylim([100 500]);
end
```

A.4 Plotting the grid based on g_x and g_y

```
function Drawgrid(obj_boundary, centroid, gap)
% Draws object and grid lines based on g_x and g_y
% The function has no returns
%

%% Object plot:
figure(1)
xlimit = 500;
ylimit = 500;
xlim([0 xlimit]); ylim([100 ylimit]);
hold on;
fill(obj_boundary(:,1), obj_boundary(:,2), [0.8 0.8 0.8]);
```

```
plot(obj_boundary(:,1), obj_boundary(:,2), '-k', 'LineWidth'
,2);
alpha(.5)
plot(centroid(1), centroid(2),'rx','Linewidth',2,
'markersize',13);

%% Grid plot:
grid on;
% Grid properties:
ax = gca;
ax.GridColor = [0 0 0];
ax.GridAlpha = 0.8;

% Draw grid based on the computed gap:
temp_div = centroid(1)/gap(1);
start_x = centroid(1) - round(temp_div)*gap(1);
temp_div = centroid(2)/gap(2);
start_y = centroid(2) - round(temp_div)*gap(2);
xticks(round(start_x,2):round(gap(1),2):xlimit);
yticks(round(start_y,2):round(gap(2),2):ylimit);
set(gca,'FontSize',15)
xlabel('x^g')
ylabel('y^g')
```

A.5 Begin simulation with defined path

```
function Simulation (obj_boundary, centroid, gap, path_choice)
% Simulates the transportation of the object
% Set path and speed of the transportation:
% Assumption: 1. speed is uniform
%             2. ignore friction
%

%% Initialize the position of the object:
```

```
obj_boundary_Lpos = obj_boundary;
centroid_path = centroid;
speed  = 1;

%% Path1: y = 2x
if (path_choice(1) == 1)
    path_x = 0:speed:200;
    path_y = path_x * 2;
    path = [path_x.', path_y.'];
    [obj_boundary_Lpos centroid_path_addon] =
    SimulateTransport(obj_boundary_Lpos, centroid, centroid_path, gap, path);
    centroid_path = [centroid_path; centroid_path_addon];
end

%% Path2: y = -x
if (path_choice(2) == 1)
    path_x = 0:speed:100;
    path_y = path_x * -1;
    path = [path_x.', path_y.'];
    [obj_boundary_Lpos centroid_path_addon] =
    SimulateTransport(obj_boundary_Lpos, centroid, centroid_path, gap, path);
    centroid_path = [centroid_path; centroid_path_addon];
end

%% Path3: y = 1
if (path_choice(3) == 1)
    path_y = 0:speed*1:100;
    path_x = path_y * 0;
    path = [path_x.', path_y.'];
    [obj_boundary_Lpos centroid_path_addon] =
    SimulateTransport(obj_boundary_Lpos, centroid, centroid_path, gap, path);
    centroid_path = [centroid_path; centroid_path_addon];
end

%% Path4: x = 1
```

```
if (path_choice(4) == 1)
    path_x = 0:speed:200;
    path_y = path_x * 0;
    path = [path_x.', path_y.'];
    [obj_boundary_Lpos centroid_path_addon] =
    SimulateTransport(obj_boundary_Lpos, centroid, centroid_path, gap, path);
    centroid_path = [centroid_path; centroid_path_addon];
end

%% Path5: y = sin(x)
if (path_choice(5) == 1)
    path_x = 0:2:400*pi;
    path_y = 200*sin(path_x/200);
    path = [path_x.', path_y.'];
    [obj_boundary_Lpos centroid_path_addon] =
    SimulateTransport(obj_boundary_Lpos, centroid, centroid_path, gap, path);
    centroid_path = [centroid_path; centroid_path_addon];
end
```

A.6 Display the transportation, variation of the distance from q_c to R

```
function [obj_boundary_Lpos, centroid_path] = SimulateTransport
(obj_boundary, centroid, centroid_path_in, gap, path)
% Simulates the transportation of the object
% The function has no returns
% The simulation includes followings:
% 1. Object movement (o)
% 2. Centroid path (o)
% 3. Robot positions (o)
% 4. boundary of under robots (o)
%

%% Initialization of some variables:
% For object draw:
```

```
obj_boundary_Lpos = obj_boundary;
% polyin = polyshape(obj_boundary(:,1), obj_boundary(:,2));

iteration=0;
v = VideoWriter('ObjectTransportation C1new.avi');
v.FrameRate = 100;
v.Quality = 100;
open(v);

% For path draw:
centroid_path_x = centroid_path_in(end,1);
centroid_path_y = centroid_path_in(end,2);
path_x_size = size(path(:,1));
path_y_size = size(path(:,2));
path_x = path(:,1).';
path_y = path(:,2).';
limits = [2700 800];

% For gap draw:
temp_div = centroid(1)/gap(1);
start_x = centroid(1) - floor(temp_div)*gap(1);
temp_div = centroid(2)/gap(2);
start_y = centroid(2) - floor(temp_div)*gap(2);
grid_xticks = round(start_x,2)-3*round(gap(1),2):round(gap(1),2):limits(1);
grid_yticks = round(start_y,2)-3*round(gap(2),2):round(gap(2),2):limits(2);

% For robot position:
all_cross_pos = [];
size_grid_xticks = size(grid_xticks);
size_grid_yticks = size(grid_yticks);
for i=1:1:size_grid_xticks(2)
    for j=1:1:size_grid_yticks(2)
        all_cross_pos = [all_cross_pos; grid_xticks(i), grid_yticks(j)];
    end
end
```

```
%% Simulate the movement of the object with centroid:
% Define the iteration number:
if (path_x_size(1)>path_y_size(1)) no_i = path_x_size(1);
else no_i = path_y_size(1); end
max_no_robot = 0;
robot_xy_total=[];
dist_array = [];
iter_array = [];
for i_x = 1:1:no_i
    % Iteration settings:
    i_y = i_x;
    if (i_x >= path_x_size(1)) i_x = path_x_size(1); end
    if (i_y >= path_y_size(1)) i_y = path_y_size(1); end

    % Define new polygon for each iteration:
    polyin2 = polyshape(obj_boundary(:,1)+path_x(i_x), obj_boundary(:,2)
    +path_y(i_y));

    % Define new centroid for each iteration:
    [ geom, iner, cpmo ] = polygeom(obj_boundary(:,1)+path_x(i_x),
    obj_boundary(:,2)+path_y(i_y));
    centroid_x_new = geom(2);   centroid_y_new = geom(3);

    % Store the last/recent position of the object:
    obj_boundary_Lpos = [obj_boundary(:,1)+path_x(i_x), obj_boundary(:,2)
    +path_y(i_y)];

    % Define the path of the centroid of the object:
    centroid_path_x = [centroid_path_x, centroid_x_new];
    centroid_path_y = [centroid_path_y, centroid_y_new];
    % centroid_path = [centroid_path_x; centroid_path_y].';
    centroid_path_temp = [centroid_path_x; centroid_path_y].';
    centroid_path = [centroid_path_in; centroid_path_temp];
```

```
% Compute position and boundary of robots (Under):
in = inpolygon(all_cross_pos(:,1), all_cross_pos(:,2),
obj_boundary_Lpos(:,1), obj_boundary_Lpos(:,2));
no_in = sum(in(:) == 1);
all_cross_pos_x = all_cross_pos(:,1).';
all_cross_pos_y = all_cross_pos(:,2).';
robot_u_pos_x = all_cross_pos_x(in);
robot_u_pos_y = all_cross_pos_y(in);
robot_u_boundary = boundary(robot_u_pos_x.', robot_u_pos_y.', 0.1);
robot_u_polygon = polyshape(robot_u_pos_x(robot_u_boundary),
robot_u_pos_y(robot_u_boundary));
robot_u_xy = [all_cross_pos_x(in).', all_cross_pos_y(in).'];

% Compute position and boundary of robots (Surround):
robot_s_xy = [0,0];
add_value = round([gap(1),0; gap(1)*-1,0; 0,gap(2); 0,gap(2)*-1;
gap(1),gap(2); gap(1)*-1,gap(2); gap(1),gap(2)*-1; gap(1)*-1,gap(2)*-1]
, 2);
for (i=1:size(robot_u_xy,1))
    x = robot_u_xy(i,1);
    y = robot_u_xy(i,2);
    for (j=1:size(add_value,1))
        new_x = x+add_value(j,1);
        new_y = y+add_value(j,2);
        result = redundant_check(robot_u_xy, robot_s_xy, new_x, new_y);
        if result == 1
            robot_s_xy = [robot_s_xy; new_x, new_y];
        end
    end
end
robot_s_xy(1,:) = [];

% Compute all positions covered during the transportation:
robot_xy_total = [robot_xy_total; robot_s_xy; robot_u_xy];
robot_xy_total_table = table(robot_xy_total);
```

```
    robot_xy_total_table = unique(robot_xy_total_table);
    robot_xy_total = table2array(robot_xy_total_table);

    sum_of_no_robot = size(robot_s_xy,1) + size(robot_u_xy,1);
    if sum_of_no_robot > max_no_robot
        max_no_robot = sum_of_no_robot;
    end

    % Compute the distance between the cenroid and the created boundary
    [d,x_poly,y_poly] = p_poly_dist(centroid_x_new, centroid_y_new,
    robot_u_polygon.Vertices(:,1), robot_u_polygon.Vertices(:,2));
    dist_array = [dist_array, d];
    iter_array = [iter_array, iteration+1];

    % Draw followings:
    % 1. Object movement
    % 2. Centroid path
    % 3. Robot positions
    % 4. boundary of under robots

    figure(2)
    drawplots(polyin2, centroid_x_new, centroid_y_new, centroid_path,
    limits, grid_xticks, grid_yticks, all_cross_pos, in, robot_u_polygon,
    robot_s_xy, obj_boundary_Lpos, v, iteration);

    % Plotting distance from centroid to boundary
    figure(4)
    drawplots_dist(dist_array, iter_array, iteration, limits);

    frame = getframe(gcf);
    writeVideo(v, frame);
    iteration = iteration+1;
end
close(v);
end
```

```
%% 'drawplots' function to plot object, centroid, path of centroid:
function drawplots(polyin2, centroid_x_new, centroid_y_new, centroid_path,
limits, grid_xticks, grid_yticks, all_cross_pos, in, robot_u_polygon,
robot_s_pos, obj_boundary_Lpos, v, iteration)
    all_cross_pos_x = all_cross_pos(:,1).';
    all_cross_pos_y = all_cross_pos(:,2).';
    %plot(polyin2);
    fill(obj_boundary_Lpos(:,1), obj_boundary_Lpos(:,2), [0.8 0.8 0.8]);
    hold on;
    alpha(.5);

    % Draw grid:
    drawgrid(grid_xticks, grid_yticks);

    % Plot object:
    plot(centroid_x_new, centroid_y_new,'rx','Linewidth',2, 'markersize'
    ,13);

    % Plot path of centroid:
    line(centroid_path(:,1), centroid_path(:,2),'Color','red');

    % Plot surrounding robots:
    plot(robot_s_pos(:,1), robot_s_pos(:,2),'o','Linewidth',1,
    'MarkerFaceColor','#2E75B6', 'MarkerSize',13, 'MarkerEdgeColor','black');

    % Plot all possible pos of robots:
    plot(all_cross_pos_x(in), all_cross_pos_y(in),'o','Linewidth',1,
    'MarkerFaceColor','#A9D18E', 'MarkerSize',13, 'MarkerEdgeColor','black');
    % points inside
    % plot(all_cross_pos_x(~in), all_cross_pos_y(~in),'bo'); % points outside

    % Plot boundary of robots(under)
    plot(robot_u_polygon,'FaceColor','#A9D18E','FaceAlpha',0.5,
    'LineStyle', '--' );
```

```
    xlim([-100 limits(1)]); ylim([-300 limits(2)]);
    title(['Movement of the object [i :' num2str(iteration) ']']);
    set(gcf, 'Units', 'Normalized', 'OuterPosition', [0, 0.04, 0.6, 0.5]);
    set(gcf,'color','w');
    set(gca,'FontSize',15)
    xtickangle(45)
    xlabel('x^g');
    ylabel('y^g');
    drawnow;
    hold off;
end

%% 'drawplots_dist' function to plot object, centroid, path of centroid:
function drawplots_dist(dist_array, iter_array, iteration, limits)
    plot(iter_array, abs(dist_array), 'Linewidth', 2, 'Color','red');
    hold on;grid on;
    title(['Shortest distance from q_c to the boundary,
    R [i :' num2str(iteration) ']']);
    xlim([0 628]); ylim([0 150]);
    set(gcf, 'Units', 'Normalized', 'OuterPosition', [0, 0.5, 0.6, 0.36]);
    set(gcf,'color','w');
    set(gca,'FontSize',15)
    xtickangle(45)
    xlabel('iteration, i');
    ylabel('distance, s_{c,r}');
    hold off;
end

%% 'drawgrid' functon to draw grid:
function drawgrid(grid_xticks, grid_yticks)
    grid on;

    % Grid properties:
    ax = gca;
```

```
    ax.GridColor = [0 0 0];
    ax.GridAlpha = 0.6;

    % Draw grid based on the computed gap:
    xticks(grid_xticks);
    yticks(grid_yticks);
end

function[return_val] =  redundant_check(robot_u_xy, robot_s_pos, new_x, new_y)
    return_val = 1;
    for(i=1:size(robot_u_xy,1))
        compare_x = round(robot_u_xy(i,1),1);
        compare_y = round(robot_u_xy(i,2),1);
        if (round(new_x,1) == compare_x && round(new_y,1) == compare_y)
            return_val = 0;
        end
    end
    for(i=1:size(robot_s_pos,1))
        compare_x = round(robot_s_pos(i,1),1);
        compare_y = round(robot_s_pos(i,2),1);
        if (round(new_x,1) == compare_x && round(new_y,1) == compare_y)
            return_val = 0;
        end
    end
end
```