2018 UKACC 12th International Conference on Control (CONTROL)
Sheffield, UK, 5-7 Sept 2018

Swarm Robotics for Object Transportation

Jean Luc Farrugia
Department of Systems & Control Engineering
University of Malta
Msida, Malta, MSD2080
jeanlucfarrugia516@gmail.com

Simon G. Fabri
Department of Systems & Control Engineering
University of Malta
Msida, Malta, MSD2080
simon.fabri@um.edu.mt

***Abstract*— In this paper a system is proposed whereby a group of small and inexpensive LEGO robots cooperate autonomously to transport a relatively much larger object to a specific location. In achieving this objective, the robots were able to generate a formation, recognise and locate neighbouring robot positions, control their motion and coordinate as a team in order to move the object to a set target location. Cooperative transportation algorithms were implemented, tested and evaluated on a physical setup, including pushing, caging and grasping. Results demonstrate that any desired formation shape can be generated and maintained without distance or orientation constraints. Caging and grasping algorithms yielded an accurate delivery performance, exhibiting reliability across different scenarios.**

I. INTRODUCTION

Advances in communications and the coming of the Internet have enabled computational devices to connect and collaborate with each other, sharing information and resources across the globe. Such technologies have proven to be indispensable for academia, industry and other scenarios, giving rise to the Internet of Things and Industry 4.0 [1]. In light of this trend, a promising area of research is Cooperative and Swarm Robotics whereby a team of robots with limited individual resources communicate and organize themselves as a group to perform tasks that go beyond the abilities of the individual robots [2]. Using a group of robots instead of one yields several advantages, including increased efficiency, enhanced redundancy and fault tolerance, and reduced costs.

This paper describes a system whereby a swarm of small and inexpensive differential drive LEGO robots [3] cooperate autonomously to transport a relatively much larger object to a specific location. In this work, swarms of only seven robots or less were considered. A variety of cooperative transportation approaches in the literature were reviewed as part of this work. In [4] it is shown how lack of communication results in loss of control, especially if the group of robots is small. In [5] robots organise themselves at two sides of an object to be moved, and push in a perpendicular direction. In order to change direction of movement, the robots have to change formation and re-organise themselves. The system is not completely autonomous as the robots move towards a leader that is teleoperated by a human to guide them along a desired trajectory. In [6], holonomic Scarab robots 'cage' the object, creating a bounded moveable area for the object to be transported. Scarabs are equipped with laser range finders, Wi-Fi connectivity and make use of two overhead cameras and LED targets for self and neighbour localisation to maintain formation. In [7] a Kilobot swarm moves the object by grasping it. Circular rings were mounted along the edges of the object to emulate grippers. In [8] a simulated S-bot swarm is proposed which also grasps the object to be transported. Robot formation shapes generated are non-deterministic; with the most efficient for transport being a caging-like formation whereby the robots surround the object. Formation generation, which is a research domain in its own right, is also a crucial requirement for the achievement of cooperative transportation. Reference [9] proposes a behaviour-based approach which generates different formation shapes but is unable to switch formation on the fly. Leader and neighbour referenced formations proved to be more feasible than unit-centre referencing, which heavily taxed the communicaton system. In [10] an algorithm is proposed which can easily change formation, but requires the shape to be composed of equilateral triangles. In [11], robots must be contiguous in order to coordinate and generate a formation, imposing restrictions on the formation shape.

In this paper, an autonomous system was developed whereby one robot is designated as a leader of a team of robots, who leads the group so as to cooperatively move an object towards a target position set by the user. This approach was taken because a proper balance between leader-based and distributed control is known to yield positive results [12]. The final objective of cooperative transportation requires various intermediate steps for its achievement. The robots need to generate a formation, recognise and navigate towards the object to be transported, recognise and locate neighbouring robot positions, control their motion and coordinate as a team in order to reach the set target. Hence in this work, two novel control algorithms were developed; one based on rigid formations, and a second one based on leader imitation. The rigid formation algorithm can generate any shape bounded only by the range of the communication system used. Adaptations to an algorithm for visual target tracking [14] were also performed. Finally, experimental implementation and validation on a physical set up were carried out in a research domain where work is commonly implemented in simulation only. Three cooperative transportation strategies were investigated: pushing, caging and grasping, in target destinations situated at front, left and right of the swarm's initial position. The robots only relied on Wi-Fi or Bluetooth for communication, and local sensors were limited to a webcam, gyro and wheel encoders. Through this communication infrastructure, robots can still collaborate even if visually occluded by the object to be transported.

978-1-5386-2864-5/18/$31.00 ©2018 IEEE

The paper proceeds with a description of the rigid formation algorithm in Section II. This is followed by a description of the imitation-based control algorithm in Section III. Sections IV and V illustrate the target tracking algorithm used in the simulated and physical setups, and the experimental results obtained. Conclusions are presented in Section VI.

II. The Rigid Formation Algorithm

A novel rigid formation algorithm is introduced that is formulated on different principles to those in [5]-[11]. Consider the scenario in Fig.1 whereby a follower robot (yellow) is to move towards a target pose (x_{rel}, y_{rel} ,θ_{rel}) defined on the leader robot's local axes X" and Y", because moving the follower to this position with respect to the leader's current pose will yield some desired formation. The leader transmits its pose, including the orientation, over wireless communication to the follower. The latter can then compute its current target position relative to the leader, but in order to navigate to this target pose, the follower must convert the pose to coordinates in the global reference frame (X and Y). The target pose defined in the X" and Y" (red) axes must first be translated to the X' and Y' axes (blue) which can then be translated to the global axes. The target point (x_{rel}, y_{rel}) is rotated with identical coordinates on the X' and Y' axes by θ_{lead} degrees on the leader's current coordinates, which represent the origin of both the blue and red axes. The blue axes need then to be translated to the global X and Y axes by adding x_{lead} and $y_{lead,}$. Hence global frame coordinates are computed using (1):

$$\begin{bmatrix} x_g \\ y_g \\ \theta_g \end{bmatrix} = \begin{bmatrix} cos\,(\theta_{lead}) & -sin(\theta_{lead}) & 0 \\ sin(\theta_{lead}) & cos\,(\theta_{lead}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{rel} \\ y_{rel} \\ \theta_{rel} \end{bmatrix} + \begin{bmatrix} x_{lead} \\ y_{lead} \\ \theta_{lead} \end{bmatrix} \quad (1)$$

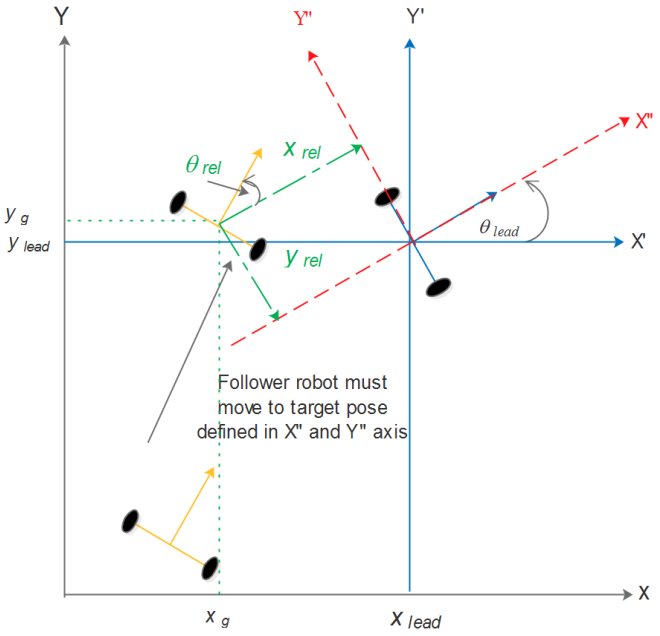

Fig. 1: Rigid formation control model.

where x_g, y_g , θ_g are the target coordinates and orientation in the global frame respectively. The follower robot can then use a point stabilisation or trajectory control algorithm to navigate to the target pose in the global coordinate frame. Hence it is suitable for different locomotion mechanisms. A point stabilisation algorithm from [13] was implemented. The proposed robust criterion was adhered to, guaranteeing stability [13]. This algorithm was also used for the pushing and caging strategies, exhibiting the following advantages: **Flexibility**: using relative poses for the follower robots which can be defined by the user in the leader's local axis, the swarm can form any shape, free from constraints such as having a formation based on equilateral triangles or from short distances between the neighbours. Follower robots can switch formation on the fly, by just changing target pose parameters (x_{rel}, y_{rel}, θ_{rel}) relative to a global leader or a neighbour. **Ease of use:** fundamental coordinate transformations are needed to define a desired formation relative to the leader, by configuring the target pose for each follower. For a large formation, the user may draw a desired shape and the relative positions for followers are calculated by the algorithm. **Scalability**: Even though Fig. 1 shows a scenario of two robots, the system scales well to moderately larger formations, whereby each follower robot can compute its target position relative to the local axes of a neighbour robot or a local leader in a hierarchy, to avoid communication overload on a single leader. **Efficiency:** Matrix calculations can exploit parallel processing architectures prevalent in modern computing systems. Additional details can be found in [14].

III. The Imitation Control Algorithm

In this section a novel robot control algorithm is introduced to replace the pose stabilisation algorithm [13] once the formation is established through the previous rigid formation algorithm and cater for a grasping strategy. The principle is to have the followers imitate the driving commands (translational velocity and angular velocity) adopted by the leader robot in moving an object towards a target destination. Since the imitation controller mimics the leader's motion, the target pose is not rotated by changes in the leader's heading. Hence the follower robots only apply a force on the object in the leader's direction, and do not attempt to move the object in opposing directions, which often results in deadlock. This approach is particularly useful in the following scenarios: when an object needs to be pushed or grasped by a group of robots, and the followers need to maintain a line in contact with the object, regardless of the leader's orientation; when only the leader robot, or a sub-set of the robots know where the target destination is; when follower robots have limited computational resources and navigational abilities, relying on a leader or neighbour to compute control actions. Consider the scenario in Fig 2: the leader robot (blue) is turning in a curve of radius r with translational velocity v_l and angular velocityω. Hence $v_l = r\omega$. The follower robot (yellow) is to imitate the leader's control actions, hence it will need to turn in an arc of the same curvature, but with a larger distance from the Instantaneous Centre of Curvature (ICC). Let r_{sep} be the separation distance between the robots, which is known a priori and to be maintained in formation. Hence in order for the

follower to imitate the leader's velocity whilst turning on the same arc (angular velocity is ω for both leader and followers) but further away from the ICC; by substituting v_l, the follower's translational velocity v_f is calculated by:

$$v_f = v_l \pm r_{sep}\omega \quad (2)$$

where '+' is used if the follower robot is on the outside of the curve as in Fig 2, and '-' if on the inside of the curve with respect to the leader. Further details can be found in [14].

IV. Simulated and Physical Setup

The system was tested both in simulation and on physical robots. In the simulation setup, an algorithm for visual target tracking using double exponential smoothing [15] was implemented to maintain a set distance to a neighbour in formation. In the physical experimentation setup, the same algorithm was used to navigate the robot towards the object to be transported. Adaptations were also made to allow the robots to track the targets at different set angles, thereby creating different formation shapes.

Matlab was used for simulating the developed algorithms with different parameters and configurations including: rigid formations, swarming and formations based on vision-based control adapted from [15]. The MRSim plugin [16] was used to create environments and configure swarms of robots, including the size of the group and the characteristics of the sensors of each individual robot. The plugin works in a step-by-step approach, whereby one step is the time taken for all robots to take a sample and process their control loop. In both simulation and the physical setup, independent software classes were implemented for the control algorithms and the low-level locomotion control of the robot. The decoupling of the controllers and low-level motion control facilitates code reuse of the control algorithms class with different types of locomotion hardware, e.g. legged robots, omni-wheeled robots and other systems. The control algorithm classes were developed in Java, and the same code was used in both the simulated and physical setup.

The physical and hardware setup consists of three Lego Mindstorms robots, including two Mindstorms EV3 and an NXT robot to form a swarm. Hence the swarm can be classified as heterogeneous. Each robot includes its embedded computer board, sensors and actuators. A remote computer for the user to interact with the swarm and a wireless router for inter-robot communication were used. The EV3 has a faster processor and more RAM than the NXT. In addition the EV3 also has built-in support for both Wi-Fi (through a USB dongle) and Bluetooth communication. The NXT only has built-in Bluetooth communication.

Fig. 3 illustrates this setup and Fig 4 illustrates the components of the leader robot. This robot contains an EV3 processor for executing control algorithms and two Raspberry Pi processor boards for executing sensor fusion and computer vision functions respectively. Only onboard sensors are used, including a webcam, two touch sensors and a gyro. Pose information is obtained by fusing information from odometry and the gyro, using an Extended Kalman Filter. No laser scanners, motion capture systems, omnidirectional or overhead cameras are used, keeping costs low. The LeJOS framework [17] and Raspbian OS[18] were installed on the Mindstorms, and Raspberry Pi boards respectively. Follower robots possessed a similar setup, but lacked a webcam. The NXT robot also lacked a gyro but benefitted from UMB Benchmark odometry calibration [19] performed beforehand as part of this work. Hence the swarm is considered heterogeneous, and algorithms developed must cater for heterogeneity. Sensor tuning, calibration, wear and tear, drift and manufacturing differences can adversely affect the system if it does not cater for diversity. Reference [20] discusses these factors in further detail.

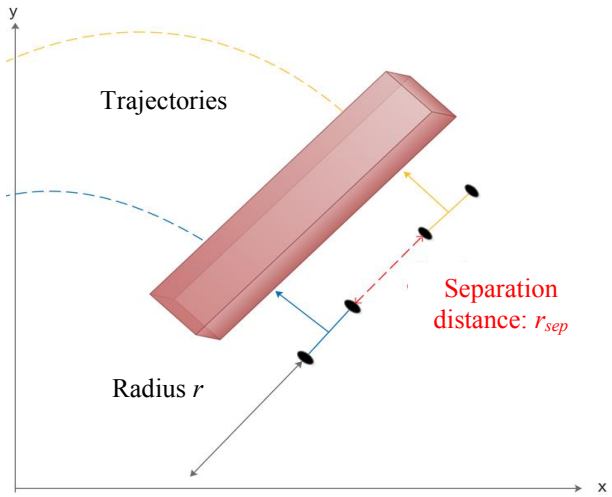

Fig. 2: Imitation control in cooperative transport.

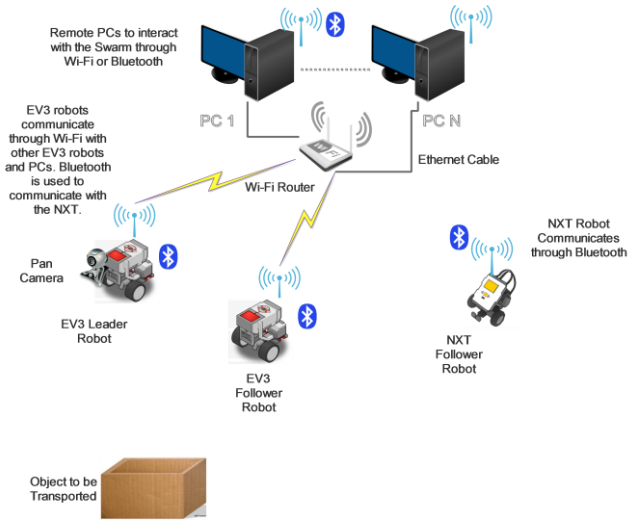

Fig. 3: Physical setup.

The webcam mounted on the leader was visually calibrated using curve fitting to develop a formula relating the size of the object to the relative distance [14]. Java was used for developing the control algorithms running on LEGO hardware, whilst C++ and Python were used for sensor fusion and computer vision respectively on the Raspberry Pi boards. Reference [14] describes the software architecture in further detail.

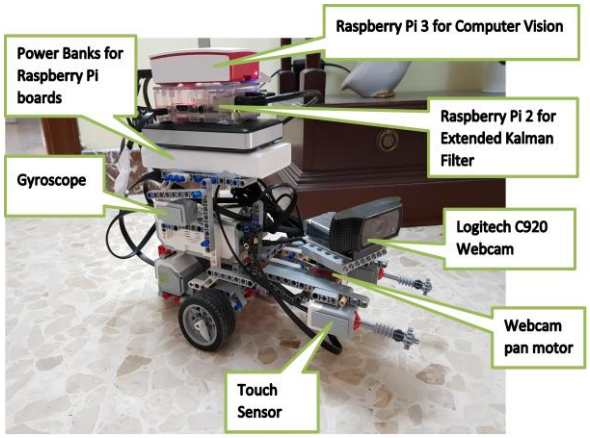

Fig. 4: EV3 leader robot.

V. Experimental Results

A. Simulation Results

Simulation experiments were performed to evaluate the rigid formation algorithm, formation generation through target tracking, and swarming. The rigid formation algorithm using pose stabilisation successfully guided the robots to converge into their target positions for different formation shapes. These shapes include line, wedge, square and triangular shapes. Fig. 5 plots the path taken by each robot to converge to a line formation. In this example the followers moved to target positions relative to the leader (blue robot). Similar results were obtained with the other shapes mentioned, and in curved figure of eight paths, such as in Fig 6. Neighbour-referenced target position schemes were also evaluated, which also resulted in the robots converging into formation, although convergence times were measured to be 34% slower than the leader-referenced approach in tests pertaining to large robot groups as depicted in Fig. 7. Through the modified target tracking algorithm, adapted from [15], the robots tracked a leader at different set angles. Hence by adjusting the value of the set target angle, different formation shapes could be created. A laser scanner sensor model was simulated to obtain the distance and orientation of the leader. Random noise was added to the readings to simulate sensor noise. Double exponential smoothing was used to filter out noise in readings [15]. Reference [14] discusses the outcomes of these tests in more detail.

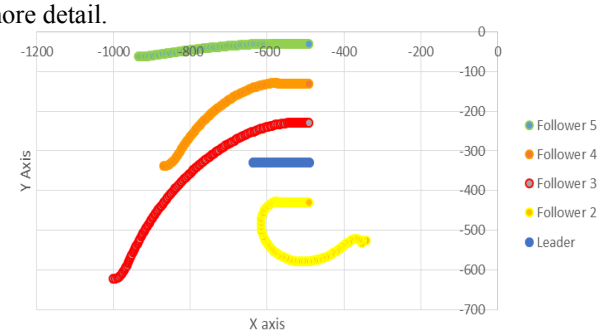

Fig. 5: Paths taken to converge into line formation.

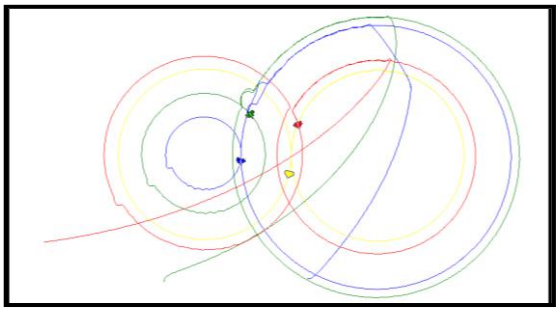

Fig. 6: Robots converging and maintaining square formation in a figure of eight path.

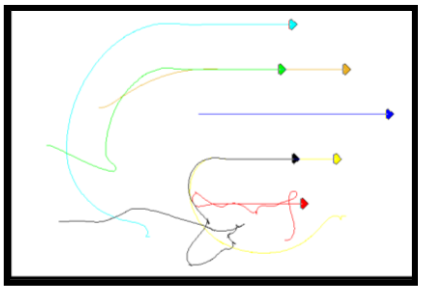

Fig. 7: Converging to large triangular formation using neighbour-referenced target positions.

B. Physical Setup Results

Experiments on the physical setup were performed to evaluate the rigid formation algorithm, swarming, target tracking and cooperative transportation of a relatively large object. The rigid formation algorithm was tested prior to evaluating the system's performance in cooperative transportation. The robots successfully generated different formation shapes including a triangle, square and line formations, obtaining similar results to the simulations. Further tests were carried out to evaluate the accuracy of the poses established, when the initial positions of the robots are located in different quadrants with respect to the target positions. These included tests whereby the robots moved in formation towards target destinations situated at the front, and in the right quadrant as illustrated in Fig. 8. The initial poses of the robots were as follows where the first two terms denote x, y coordinates in mm, and the third term denotes orientation θ in degrees: EV3 leader: (0, 0, 0); EV3 follower: (0, -400, 0); NXT follower: (0, -800, 0). Table I lists the desired target and the actual final positions achieved by the robots for four runs of the same test. The following global average percentage errors were obtained after computing the average of the values in the last two columns of Table 1: EV3 leader: -0.10%; EV3 follower: -2.15%; NXT follower: 2.16%. This is a very satisfactory performance especially considering that only local sensors were used and the gyro sensor was found to be unable to detect changes in heading when moving at slow velocities [14]. The right quadrant proved to be more challenging, due to the arc trajectory required. In both quadrants the EV3 leader obtained the most accurate results due to sensor fusion and better weight distribution than the other two robots. The Raspberry Pi power banks at the back, and the webcam at the

TABLE I. Position results in different target destinations.

	Desired Position *(x, y) (mm)*	Test Run 1	Test Run 2	Test Run 3	Test Run 4	Actual average position across the 4 test runs *(x, y) (mm)*	Average error as a % of desired *x* position	Average error as a % of desired *y* position
		Actual Position *(x, y) (mm)*	Actual Position *(x, y) (mm)*	Actual Position *(x, y) (mm)*	Actual Position *(x, y)(mm)*			
Robot	***Front Quadrant Tests***							
EV3 Leader	(2500, 0)	(2490, 20)	(2506, 3)	(2499, 2)	(2497, 8)	(2498, 8.25)	-0.08	Not Applicable
EV3 Follower	(2200, 300)	(2210, 290)	(2205, 310)	(2212, 300)	(2193,302)	(2205, 300.5)	0.23	0.17
NXT Follower	(2200, -300)	(2212, -330)	(2025, -276)	(2027, -280)	(2210, -320)	(2118.5, -301.5)	-3.70	-0.5
Robot	***Right Quadrant Tests***							
EV3 Leader	(2500, -2800)	(2500, -2795)	(2495, -2800)	(2499, -2802)	(2498, -2788)	(2498, -2796.25)	-0.08	-0.13
EV3 Follower	(-1500, -2400)	(-1240, -2430)	(-1510, -2460)	(-1120, -2452)	(-1490, -2420)	(-1340, -2440.5)	-10.67	1.69
NXT Follower	(-1500, -3200)	(-2020, -3180)	(-1530, -3150)	(-1600, -3210)	(-1652, -3194)	(-1700.5, -3183.5)	13.37	-0.52

front helped to achieve a weight distribution that was centered on the middle of its wheelbase.

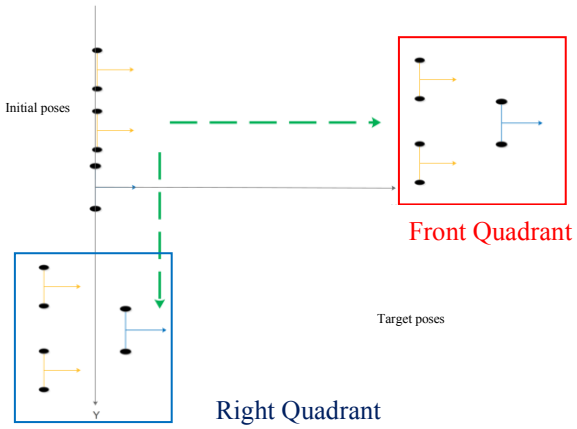

Fig. 8: Target poses for rigid formation tests

The next best accuracy was obtained by the EV3 follower which benefitted from sensor fusion but lacked the centred weight distribution, Raspberry Pi board and webcam required for computer vision. The NXT follower obtained better results in the *y* coordinate due to the narrower width of its tyres which are more likely to rotate consistently on fixed points at the tips than the flatter, wider EV3 tyres.

For the target tracking algorithm, a webcam was calibrated by measuring the size of the target object (flagged by an onboard purple LED light) detected by the image sensor over different distances, to provide the relative distance to the object. Curve fitting was used to determine a formula which relates the object size to the relative distance of the object from the webcam. The target tracking algorithm successfully guided the leader robot towards the object to be transported, until two tactile sensors became active when the robot made contact with the object.

Three cooperative transport strategies were next evaluated. This included pushing, caging and grasping of an irregular, rectangular shaped cardboard box of size 90cm by 32cm. Rectangular objects are more challenging to transport [5]. The centroid of the box was used as a reference point to measure performance accuracy of all three strategies. The pushing approach (illustrated in Fig. 9) proved to be effective in transporting the object to target destinations situated at the front of the initial starting poses, but resulted in the robots losing control of the object when target destinations were situated at the left or right. This occurred due to the large changes in heading demanded by the employed pose stabilisation algorithm [13]. The pushing strategy imposes mobility restrictions such that the robots must constantly apply a force which is perpendicular to the object's surface, else control is lost. This strategy achieved an average percentage error of –0.335% for destination targets situated 2800mm from the start position. The caging approach (illustrated in Fig. 10) was based on 'conditional closure' as only three robots were available, and 'object closure' requires at least four robots [12]. This approach was successful in moving the object to target destinations situated at the left, right and front of the robots' initial positions. An average percentage error of -1.98% was obtained over distances of 2800mm at the front and 3795mm at the sides. The grasping approach (illustrated in Fig. 11) was also successful in moving the object to target destinations situated at the left, right and front of the robots' initial positions. Robots were placed inside 'grip rings' to emulate a gripper as per [7]. An average percentage error of -0.42% was obtained in identical test conditions used for the caging approach. The grasping approach achieved more accurate results due to the more 'harmonious' and synchronised motion produced by the imitation algorithm, whereas in the caging approach each robot can be moving and applying a force in a different direction to move to the current target pose computed by the rigid formation algorithm. Thus the object is moved in the direction of the global resultant force, and a higher probability of wheel slippage is incurred. The grasping approach requires only two robots to perform the task, instead of three or four robots as required by caging. On the other hand the caging algorithm proved to be robust to faults and is suitable in unstructured environments whereby the object to be transported cannot be grasped. Both grasping and caging are free from motion restraints imposed by the pushing strategy.

This freedom of movement is beneficial for navigating in dynamic real-world environments. In all three approaches, not all robots need to have knowledge of the final target location.

Fig. 9: The pushing approach

Fig. 10: The caging approach

Fig. 11: The grasping approach

VI. Conclusion

In this paper, novel control algorithms for multiple robots to cooperatively transport a relatively larger object to a target destination are proposed, simulated and experimentally evaluated. A novel rigid formation algorithm is implemented which can generate any shape bounded only by the range of the communication system used. This algorithm is abstracted from the navigation algorithm, allowing it to be used with different locomotion methods. Advantages include ease of use, scalability, efficiency and flexibility whereby robots can also switch formation on the fly. Another algorithm, the imitation control algorithm, is proposed whereby followers imitate the leader's movement only and apply a force on the object in the leader's direction, and do not attempt to move the object in opposing directions, which often results in deadlock. The robots are also able to perform target tracking using double exponential smoothing to maintain a set angle to the target in addition to distance. The robots are able to generate a formation, recognise and locate neighbouring robot positions, control their motion and coordinate as a team in order to move the object to a set target location. Simulation tests confirm that using the rigid formation algorithm, robots are able to generate any desired shape, and maintain it whilst the swarm is in motion, using a pose stabilisation algorithm of choice. Moreover robots can also change formation positions on the fly. Robots converge into formation using either leader-referenced or neighbour-referenced target positions, with the latter offering a more distributed approach at the expense of longer convergence times. Similar results were obtained on a physical setup consisting of different Mindstorms platforms and other inexpensive parts, demonstrating that the system is effective in a heterogeneous environment. The pushing, caging and grasping cooperative transportation strategies were implemented and evaluated. The caging and grasping approaches provide increased freedom of movement over the pushing strategy. Grasping yielded slightly better results due to more synchronous motion.

Future improvements include integrating inertial measurement units on board the robots, to investigate the efficacy of the system over longer distances. A colour coding scheme for robot identification can be introduced to implement vision-based formation control on the physical setup.

References

[1] Cisco, "Cisco #InternetOfEverything," 2016. [Online]. Available: http://ioeassessment.cisco.com/. [Accessed: 09-Dec-2016].

[2] I. Navarro and F. Matia, "An Introduction to Swarm Robotics," *ISRN Robot.*, vol. 2013, p. 10, 2013.

[3] LEGO, "mindstorms," 2017. [Online]. Available: https://www.lego.com/en-us/mindstorms.

[4] M. J. Mataric, N. Martin, and K. T. Simsarian, "Cooperative Multi-Robot Box-Pushing," in *IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, 1995.

[5] J. Chen, M. Gauci, W. Li, A. Kolling, and R. Groß, "Occlusion-Based Cooperative Transport with a Swarm of Miniature Mobile Robots," *IEEE Trans. Robot.*, vol. 31, no. 2, pp. 307–321, 2015.

[6] J. Fink, M. Ani Hsieh, and V. Kumar, "Multi-robot manipulation via caging in environments with obstacles," in *Proceedings - IEEE Int. Conference on Robotics and Automation*, 2008, pp. 1471–1476.

[7] A. C. Michael Rubenstein Justin Werfel, Golnaz Habibi, James McLurkin, Radhika Nagpal, "Collective Transport of Complex Objects by Simple Robots: Theory and Experiments," *12th International Conference on Autonomous Agents and Multiagent Systems*. Saint Paul, Minnesota, USA, 2013.

[8] R. Groß and M. Dorigo, "Towards group transport by swarms of robots," *Int. J. Bio-Inspired Comput.*, vol. 1, no. 1/2, p. 1, 2009.

[9] T. Balch and R. C. Arkin, "Behaviour-based Formation Control for Multi-robot Teams," *IEEE Trans. Robot. Autom.*, vol. 14, no. 6, pp. 926–939, 1998.

[10] T. Kato, K. Watanabe, S. Maeyama, and M. K. Habib, "A Forming Algorithm and its position estimation for Triangle-based Robot Formation," *Int. J. Mechatronics Manuf. Syst.*, vol. 6, no. 1, pp. 38–56, 2013.

[11] M. Rubenstein, A. Cornejo, and R. Nagpal, "Programmable self-assembly in a thousand-robot swarm," *Sci. Mag.*, vol. 345, no. 6198, 2014.

[12] S. G. Ponnambalam and M. Yogeswaran, "An Extensive Review of Research in Swarm Robotics," *World Congress on Nature & Biologically Inspired Computing (NaBIC 2009)*. IEEE, 2009.

[13] R. Siegwart, I. R. Nourbakhsh, D. Scaramuzza, and R. C. Arkin, *Introduction to Autonomous Mobile Robots*, 2nd ed. The MIT Press, 2011.

[14] J. L. Farrugia, "Swarm Robotics for Object Transportation," M.Sc. Dissertation University of Malta, Msida, 2018.

[15] F. Guerin, S. G. Fabri, and M. Bugeja, "Double Exponential Smoothing for predictive vision based target tracking of a wheeled mobile robot," in *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*, 2013.

[16] M. S. Couceiro, "MRSim - Multi-Robot Simulator v1.0," *Brno University of Technology*, 2012. [Online]. Available: http://home.isr.uc.pt/~micaelcouceiro/media/help/helpMRSim.htm. [Accessed: 01-Jan-2017].

[17] leJOS, "LEJOS Java for Lego Mindstorms," 2015. [Online]. Available: http://www.lejos.org/. [Accessed: 01-Jan-2017].

[18] A. Rosebrock, "Install guide: Raspberry Pi 3 + Raspbian Jessie + OpenCV 3," 2016. [Online]. Available: http://www.pyimagesearch.com/2016/04/18/install-guide-raspberry-pi-3-raspbian-jessie-opencv-3/. [Accessed: 01-Jan-2017].

[19] C. Jung and W. Chung, "Design of test tracks for odometry calibration of wheeled mobile robots," *Int. J. Adv. Robot. Syst.*, vol. 8, no. 4, pp. 1–9, 2011.

[20] B. Siciliano and O. Khatib, *Springer Handbook of Robotics*. Berlin: Springer, 2008.