Distributed Formation Control in Cluttered Environments*

Whye Leon Seng†, Jan Carlo Barca°, Y. Ahmet Sekercioglu†
†Department of Electrical and Computer Systems Engineering
°Clayton School of Information Technology
Monash University, Melbourne, Australia
whyeleon.seng@gmail.com, jan.barca@monash.edu, Ahmet.Sekercioglu@monash.edu

***Abstract*— A graph theory based control mechanism that enables groups of ground moving nonholonomic robots is proposed. The mechanism allows the robot to dynamically manage formation shapes and follow the leader through environments with obstacles. It improves upon a state of the art formation control algorithm where a formation can be maintained without the need of inter-robot communications. Obstacle avoidance is designed to be scalable and allows the robots to dynamically manage their formation according to the environment. The formation is also capable of rebuilding itself when individual robots within the formation fail. The algorithm has been tested on a nonholonomic multi-robot system, with results showing that the proposed algorithm enables a formation to complete an obstacle course and regenerate original formation shapes within 12 seconds with no collisions.**

I. INTRODUCTION

In recent years, there has been an increasing interest in automated control and coordination of robot systems, particularly in critical missions such as search and rescue where human lives are at stake. The robots deployed today, while technologically advanced, are expensive. Our solution to this problem is to introduce a distributed swarm-based Multi-Robot (MR) system. Every robot only has to be equipped with basic sensor equipment, significantly reducing the cost of deployment. With several of these robots, the swarm is capable of performing tasks beyond the capabilities of a single robot. It also allows the system to be scalable and possess self-repairing abilities should robots within the swarm fail, reducing the risk of mission failure [1].

Formation control is the key to mission success [1]. Shao et al. considers a one-leader constraint formation control [2], using adjacency and parameter matrices to define the inter-robot relationships and the formation shape. However, it does not allow the formation shape to be maintained as the formation moves. Formation control with both one-leader and two-leaders constraints was presented in [3], focusing on classifying control graphs based on the number of followers with one or two leaders, and the use of a transition matrix to change the formation structure. However, as in [2], they do not explain how one can select an ideal formation shape in environments that potentially include obstacles.

We have also looked into several obstacle avoidance techniques. Obstacles are regarded as virtual leaders in [4-5], and may lead to nonrigid formation structures that are undesirable. A behaviour-based formation control is proposed in [6], where a swarm of robots navigates about obstacles by rotating and scaling the entire formation, which can be time consuming and inefficient for nonholonomic robots. Kuppan et al. propose the election of a temporary Formation Leader (FL) to steer the formation away from obstacles [7], but this can complicate the control process when confronted with obstacles from multiple directions. A semi-rigid obstacle avoidance that allows Follower Robots (FR) to vary the angle constraint from their Local Leader (LL) is presented in [8]. However, it becomes challenging when the robots have multiple LLs and distance constraints to follow. Another technique utilizes potential fields to represent interactions between robots and obstacles with repulsive and attractive forces. [9-12]. However, the magnitude of the repulsive force is determined by the distances of obstacles, which may lead to robots slowing down excessively even when the obstacles are not necessarily blocking them.

The common problem with the techniques discussed above is that they have not been implemented on real robots [1-6, 9-12], where issues such as sensor noise and kinematics must be considered. Another problem is that the solutions to obstacle avoidance are presented as net force vectors in [10-12], indicating the instantaneous velocity that a robot should take. However, nonholonomic robots are not capable of switching from one state to another immediately. In our work, we have explicitly detailed the algorithm and its output in meaningful physical terms such as velocities and angular velocities, which can be included effortlessly in the control systems of existing robots. Our work also focused on having the robots maintain a desired formation when moving through environments with obstacles. This is achieved through techniques such as formation scaling and formation morphing. We did not, however, consider path planning for the formation in this paper. We have implemented the control mechanisms on our in-house developed eBug-II MR system [13] to properly evaluate the algorithm.

In Section II, the proposed formation control method is described. Section III introduces the implemented obstacle avoidance techniques, followed by a discussion of formation rebuilding in Section IV. The performance of the algorithm measured as obtained from the conducted experiments is discussed in Section V. Section VI concludes the paper as well as the possible future research work that can be done.

*This research was supported in part by Lise and Arnfinn Hejes Grant for Education and Research. We would like to thank Adv. Harald Røer for the way he administered the grant.

II. FORMATION CONTROL

The proposed formation control algorithm is an improvement on the work presented in [14], and employs graph theory based formation structures that branch out from a FL to the FRs, which each are assigned to one or two LLs according to their allocated Formation Positions (FP).

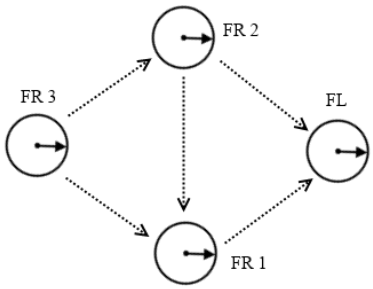

Figure 1. Inter-robot relationships represented with directed graphs, where the directed edges goes from the follower robots to their local leaders

The inter-robot relationships are described with directed graphs [2, 15-16], as shown in Fig. 1. If a path from the root (the FL) to every vertex (all the other robots) exists, it implies that all the robots are part of the formation. Every FR only needs to know the positions of the LLs relative to itself in order to maintain the formation. The formation shape is determined by the corresponding distance and angle constraints at every FP. To maintain the formation shape as it moves, we have used rigid graphs, which can be interpreted as networks of agents, connected to one another by rigid bars of length defined by the assigned distance constraints on each agent [17]. To further reduce the complexity of the network, formation structures have been designed to be minimally rigid, such that if any edge is removed from the graph, the formation will no longer be rigid [17].

A. Formation Position Assignment

The algorithm begins by assigning the robot that is the closest to a predefined destination as the FL. This positions the other robots behind the FL relative to the destination, allowing them to easily get to their assigned FPs as the FL starts moving [9].

Next, the remaining robots are sequentially assigned to FPs, starting with the first FP whose LL is the FL. Robots are assigned to FPs based on their character costs in terms of the Euclidean distance to a FP, d_{error} [18]. An improvement over the work by Chen is that we only require local information to calculate the cost, rather than using global coordinates, as seen in Fig. 2. These FPs are determined by predefined angle and distance constraints (which are dependent on the starting formation shape chosen by the user) relative to the corresponding LLs for the FPs. To begin, a FR first detects the distances and angles of its LLs relative to it. If neither LLs can be detected, i.e. beyond the detection range of the FR, R_{scan}, it will not be able to content for this FP. It then calculates the cost to the desired FP using (1-3). Note that the angles under the formation control section are all measured with respect to the opposite direction of where the formation is heading.

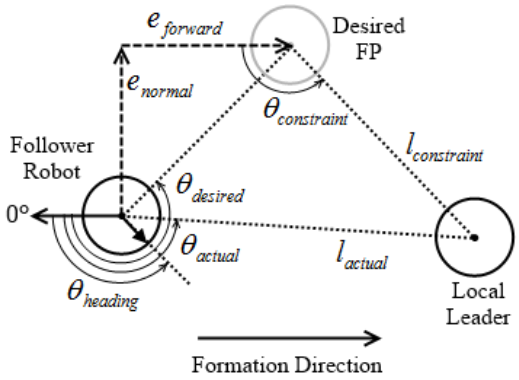

Figure 2. Reducing discrepancy between follower robot and its desired formation position with respect to the local leader

$$e_{forward} = l_{cstr} \cdot \cos(\theta_{cstr}) - l_{actual} \cdot \cos(\theta_{actual}) \quad (1)$$

$$e_{normal} = l_{cstr} \cdot \sin(\theta_{cstr}) - l_{actual} \cdot \sin(\theta_{actual}) \quad (2)$$

$$d_{error} = \sqrt{e_{forward}^2 + e_{normal}^2} \quad (3)$$

where:

$e_{forward}$	: distance error in direction of formation movement
e_{normal}	: distance error in direction perpendicular to formation movement
l_{cstr}	: required distance between robot and LL for this FP
l_{actual}	: actual distance between robot and LL for this FP
θ_{cstr}	: required angle from robot to LL for this FP
θ_{actual}	: actual angle from robot to LL for this FP
d_{error}	: Euclidean distance error from robot to desired FP

This process is performed by every robot that has not been assigned a FP and the FP is finally assigned to the robot with the lowest d_{error} for that FP. If a FP requires two LLs, the robots content for the FP with the average d_{error} for both LLs. The aforementioned steps are repeated for the subsequent FPs and are recorded in an adjacency matrix which describes the leader-follower relationships, and a parameter matrix which stores the assigned constraints as described in [2-3].

B. Control System

The following subsections describe the Velocity Controller (VC) and Angular Velocity Controller (AVC) which are responsible for reducing d_{error} between a FR and its desired FP.

1) Velocity Controller

A nonlinear controller (4) is used to provide rapid convergence towards the desired FP. It considers the heading angle of the FR to reduce unnecessary movements in the direction perpendicular to formation movement. The velocity constant, K_v should be adjusted to be lower than the formation speed to prevent the robot from overshooting its desired FP as the VC tries to reduce $e_{forward}$.

$$v = K_v \cdot \log_{10}(e_{forward}) \cdot \cos(\theta_{desired} - \theta_{heading}) \quad (4)$$

where:

v : velocity output

K_v : constant for velocity control

$e_{forward}$: distance error in direction of formation movement

$\theta_{desired}$: angle of FP measured from robot

$\theta_{heading}$: heading angle of robot

2) Angular Velocity Controller

The AVC (5-7) has two states. When the FR is at a distance, R_{dzone} away from its desired FP, the controller steers the robot directly towards the desired FP. However, as the robot enters the dead zone, the controller attempts to steer the robot in the direction that the formation is travelling in. This second state helps to reduce the oscillations that arise from the controller trying to overcorrect the heading angle of the robot when it is too close to the desired FP. R_{dzone} should only be slightly wider than the diameter of the robot as the main function of AVC is keeping the FR as close to the desired FP as possible to preserve the formation shape.

$$\alpha = \theta_{desired} - \theta_{heading} \tag{5}$$

$$\beta = 180° - \theta_{heading} \tag{6}$$

$$\omega = \begin{cases} K_\omega \cdot \alpha & , \quad d_{error} > R_{dzone} \\ K_\omega \cdot (\beta - (\beta - \alpha) \cdot (\frac{d_{error}}{R_{dzone}})) & , \quad d_{error} \le R_{dzone} \end{cases} \tag{7}$$

where:

$\theta_{desired}$: angle from robot to the desired FP

$\theta_{heading}$: heading angle of robot

ω : angular velocity output

K_ω : constant for angular velocity control

d_{error} : Euclidean distance from robot to its desired FP

R_{dzone} : radius of dead zone

III. Obstacle Avoidance

We first describe our obstacle avoidance technique which is based on potential fields, followed by the incorporation of formation scaling and morphing, wall following and escaping from local minima. Three new zones around a robot with the following radii, as well as the behaviour of the robot when obstacles are detected in these zones are defined:

R_{avoid} : robot starts avoiding obstacles

R_{wf} : robot takes more evasive measures such as wall-following

R_{stop} : robot stops entirely as the output of VC is 0

The radii of the three zones are dependent on the size of the robot and the formation velocity to ensure that the robots have sufficient distance to perform obstacle avoidance.

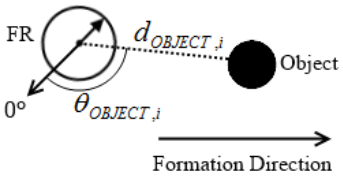

Figure 3. Measurements taken for obstacle avoidance

A. Potential Field Based Obstacle Avoidance

Obstacle avoidance is performed in two stages: reducing velocity and turning away, by considering the distance, d_{object} and the angle of the object, θ_{object} measured from the robot. Note that θ_{object} is measured with respect to the back of the robot (see Fig. 3), as obstacle avoidance is performed on an individual basis rather than collectively as a formation. An object is deemed blocking the robot if it is within R_{avoid} from the robot, and is in the forward 180° arc of the robot. Forward is defined as the i) front of the robot if the VC output is positive or ii) back of the robot if the VC output is negative. In the first stage, the robot only considers the closest object and reduces its velocity exponentially (8) as the object gets closer. However, this reduction in velocity can be excessive if the object is located nearer to the side of the robot rather than blocking it directly in the forward direction. Hence, a second multiplier is added to dampen the effect of the first multiplier (9).

$$v_{multp\_1} = 1 - \exp(-\frac{K_{rise}}{R_{avoid}} \cdot (d_{object} - R_{wf})) \tag{8}$$

$$v_{multp\_2} = \frac{\theta_{object} \cdot (1 - v_{multp\_1})}{90° \cdot v_{multp\_1}} + 1 \tag{9}$$

$$v_{final} = v \cdot max(0, v_{multp\_1} \cdot v_{multp\_2}) \tag{10}$$

where:

K_{rise} : exponential curve rising time constant

R_{avoid} : radius of obstacle avoidance zone

d_{object} : distance between robot and object

R_{wf} : radius of wall-following zone

θ_{object} : angle from robot to object

v_{final} : final velocity output

The next step is to steer the robot away from the object. When avoiding an object, the robot disregards the output of the AVC. Instead, θ_{object} is used to calculate a new angular velocity, $\omega_{avoid,i}$ for every object within R_{avoid} to turn the robot away as in (11). This is further improved upon by considering the $d_{object,i}$ in calculating $\omega_{avoid,i}$, which gradually turns the robot away more as the object approaches (12). The robot then calculates and turns away with an angular velocity of ω_{final} (13).

$$\omega_{avoid,i} = \begin{cases} 90° - \theta_{object,i} & , \quad 0° < \theta_{object,i} \le 180° \\ -90° - \theta_{object,i} & , \quad -180° < \theta_{object,i} \le 0° \end{cases} \tag{11}$$

$$\omega_{multp,i} = d_{object,i} \cdot \frac{\omega_{min} - \omega_{max}}{R_{avoid}} + \omega_{max} \tag{12}$$

$$\omega_{final} = \sum_{i=1}^{n} \omega_{multp,i} \cdot \omega_{avoid,i} \tag{13}$$

where:

$\theta_{object,i}$: angle from robot to object *i*

$d_{object,i}$: distance between robot and object *i*

ω_{min} : minimum angular velocity output

ω_{max} : maximum angular velocity output

R_{avoid} : radius of obstacle avoidance zone

ω_{final} : final angular velocity output

B. Formation Scaling

Formation scaling reduces the size of the formation when confronted with external objects, allowing the formation to maintain its shape for a longer period than otherwise. Scaling is triggered whenever a robot detects obstacles within R_{scan} and informs the entire formation. The distance constraints on the robots are continuously reduced by a factor of K_{scale} till a minimum value of K_{scale_min}, such that the robots do not get within R_{avoid} of other robots (14). When no obstacles are detected, the process is reversed (15). K_{scale} governs the rate at which the formation size scales, hence its value is empirically determined based on the environment layout.

$$l_{cstr}(t+1) = \begin{cases} l_{cstr}(t) \cdot K_{scale} & , \dfrac{l_{cstr}(t)}{K_{scale\_min}} > l_{cstr}(0) \\ l_{cstr}(0) \cdot K_{scale\_min} & , \dfrac{l_{cstr}(t)}{K_{scale\_min}} \le l_{cstr}(0) \end{cases} \quad (14)$$

$$l_{cstr}(t+1) = \begin{cases} \dfrac{l_{cstr}(t)}{K_{scale}} & , \dfrac{l_{cstr}(t)}{K_{scale}} < l_{cstr}(0) \\ l_{cstr}(0) & , \dfrac{l_{cstr}(t)}{K_{scale}} \ge l_{cstr}(0) \end{cases} \quad (15)$$

where:

$l_{cstr}(t)$: required distance between robot and its LL at time t

K_{scale} : rate at which formation size scales

K_{scale_min}: minimum formation scaling factor

C. Formation Morphing

Formation morphing is introduced to preserve the formation structure should formation scaling fail to maintain the formation shape. The use of character costs and character set matrix for every formation shape forms the basis of this technique [17]. Whenever a robot detects an object within R_{avoid}, it broadcasts a signal to trigger the consideration for formation morphing. The robots perform the sequence for assigning FPs (see Section II-A) multiple times for every predefined formation shapes. The formation shape with the lowest total d_{error} is selected and the corresponding FP constraints are assigned to the robots. A transition matrix is used to prevent any robots from being disconnected from the structure during formation morphing [3].

However, experiments show that the formation may alternate rapidly between two formation shapes which have similar d_{error}. Two mechanisms are introduced to prevent this. First, a morph timer is used to prevent formation morphing if it had previously morphed in the past T_{morph} seconds. The second mechanism employs hysteresis where the new formation shape is only accepted if the d_{error} of the current shape is greater than the d_{error} of the new shape by a factor of K_{morph}. T_{morph} and K_{morph} are determined empirically and is proportional to the size of the formation.

D. Wall Following

A robot may occasionally find itself in a situation where it is facing roughly 90° away from its desired FPs, and that there are obstacles in between the robot and the desired FP, as depicted in the Fig. 4.

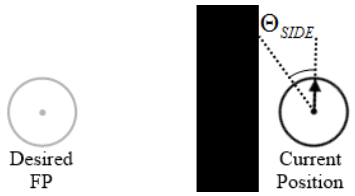

Figure 4. Situation where robot may be unable to move forward

While the AVC tries to turn the robot towards the desired FP, the obstacle would in return push the robot in the opposite direction. Coupled with the small output from the VC, the robot is unable to move to its FP. To overcome this, a wall following technique based on [19] has been implemented. When an obstacle is detected within R_{wf} and sits beyond a threshold angle, Θ_{side} (measured from the centre of the forward arc), the robot negates v_{final} (10) and instead travels in the forward direction at V_{lock}. However, should i) any obstacle angles falls within Θ_{side} or ii) the obstacle distance is within the safety margin of R_{stop} from the robot, wall following cannot be initiated to prevent potential collisions. In our experiments, V_{lock} is equivalent to the formation speed as we found it to be sufficiently slow for our robots to navigate around tight corners safely. Θ_{side} should be set such that the robot has a sufficiently wide opening to move straight ahead without colliding into the obstacles when they are initially located at R_{wf}.

E. Escaping Local Minima

A robot may also find itself trapped in local minima, which is when its velocity is below a threshold value, V_{trap} (close to 0) for a time, T_{trap}. T_{trap} is introduced as the robot's velocity can momentarily be less than V_{trap} when it is performing a zero radius turn towards its desired FP, rather than being trapped. Hence, T_{trap} is set to be slightly greater than the time the robot takes to perform a 180° turn. Fig. 5 explains how a robot first finds an angle, θ_{escape} that offers an open path to escape from the local minima. The robot then turns towards θ_{escape} and is forced to move at a positive velocity, which is the magnitude of its last calculated v_{final} (10) for a period, T_{escape}, after which it would assume that it is no longer trapped and resumes running the VC. T_{escape} is dependent on the layout of the environment. It should be long enough to prevent the robot from travelling back to the same local minimum but short enough so that it does not stray from the formation.

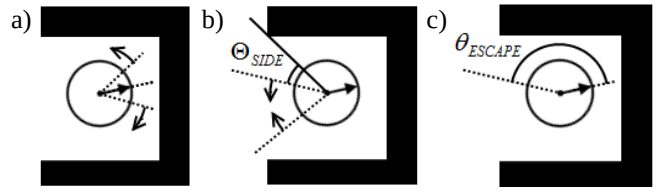

Figure 5. Finding θ_{escape}: (a) search begins from the centre of the front arc and gradually expands in both left and right directions by one resolution each iteration, (b) the aim is to find an angle where all obstacles within R_{avoid} are beyond Θ_{side}, and (c) this angle is set to be θ_{escape}.

IV. Formation Rebuilding

The formation is also able to rebuild itself should the number of operating robots changes. The term rebuild is defined as the ability for the formation to reform the last formed shape by considering the number of operating robots at any time instant. Rebuilding is done without affecting the whole formation to increase scalability. This is achieved by running the sequence for assigning FPs (see Section II-A) starting from a particular FP which is determined by one of the following scenarios. Firstly, when a new robot is added to the formation, it detects the closest neighbouring robot and its FP. The new robot then signals the formation to rebuild itself from that FP onwards. However, if the FP belongs to the FL, this new robot instead considers the next closest FP to prevent a change in leadership. The second case is met when a robot fails. If it is a LL, its FRs will be able to detect the failure and signal the formation to rebuild itself. The FP which the robot was previously assigned to is the point where the rebuilding starts.

V. Experiment Design and Results

The algorithm was implemented onto a group of three eBugs, which are ground moving nonholonomic robots designed by D'Ademo [13]. An eBug has a diameter of 120 mm and is equipped with two stepper motors powering its wheels. A base station is used for algorithm computation as the eBugs themselves do not have sufficient processing power yet. Each eBug is treated as a separate process by the base station for a decentralized approach and movement commands in terms of wheel speeds (translated from the calculated v_{final} and ω_{final}) are sent to the eBugs wirelessly over a ZigBee link. Laser range finders with a 360° field of view on the robots were emulated with the use of the BCH marker system and an overhead camera to track the coordinates of the robots. The FL was controlled through a funnel-shaped obstacle course (Fig. 6), with the FR following autonomously using the algorithm.

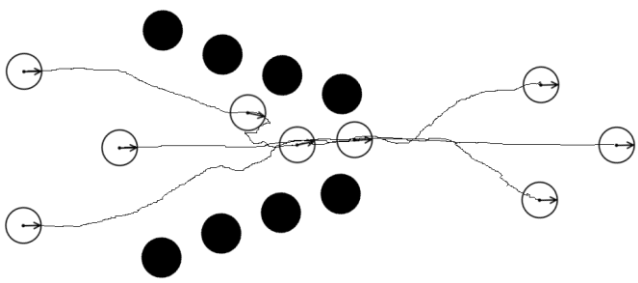

Figure 6. Funnel-shaped obstacle course

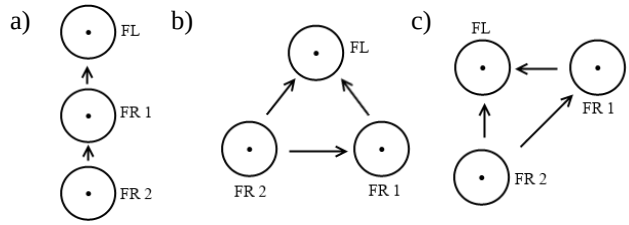

Figure 7. Formation shapes: (a) Line (b) Equilateral Triangle & (c) Right-Angled Triangle

Fig. 7 shows the different starting formation shapes that were considered. The algorithm is evaluated in terms of the i) average deviation from ideal FP and ii) time taken to complete the obstacle course and rebuild the starting shape. We have investigated the robustness of the algorithm towards i) path width and ii) formation velocity. These criteria are systematically varied until the formation breaks, which is when a FR fails to detect any of its LL within R_{scan}. The values used for implementation specific variables throughout the experiments are presented in Table 1. A video on the implementation of the algorithm can be seen in [20].

TABLE I. Parameters for Implementation Specific Variables

Variables	Value	Variables	Value
$l_{constraint}(0)$	360 mm	R_{STOP}	5 mm
K_V	30	K_{RISE}	10
K_ω	1.5	ω_{MIN}	1
R_{DZONE}	100 mm	ω_{MAX}	2.5
R_{AVOID}	100 mm	K_{SCALE}	0.999
R_{WF}	20 mm	K_{SCALE_MIN}	0.5
T_{MORPH}	0.5 s	V_{LOCK}	70 mm/s
K_{MORPH}	1.5	T_{TRAP}	1.5 s
Θ_{SIDE}	60°	T_{ESCAPE}	10 s

A. Robustness to Path Width

This experiment tested the ability of the formation go through tightly confined areas at a speed of 70 mm/s. Path width is defined as the narrowest point of the funnel. Starting at 240 mm, it is reduced at 20 mm intervals until the formation consistently breaks. Each path width is tested 30 times to ensure the validity of the results.

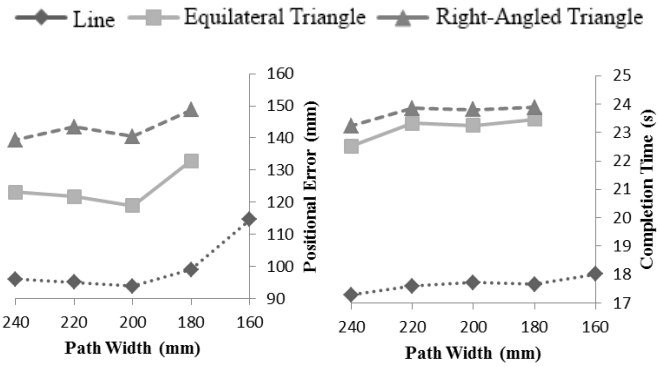

Figure 8. Performance of algorithm in tightly confined spaces

As the path width narrowed, the FRs were repelled by the obstacles and eventually morphed into a line formation to go through the funnel. This explains the superiority of the line formation over the other two as illustrated in Fig. 8, as the FRs had to travel extra distances to get to the new desired FPs after morphing. At the breaking point (160 mm and 180 mm correspondingly), the FRs could not morph quickly enough to keep up with the FL.

B. Robustness to Formation Speed

We have also tested the algorithm's ability to keep up with the FL at increasing movement speeds through a path width of 240 mm. This is vital as the algorithm is intended to be deployed onto different types of robots and missions, each with different operation speed. In our experiment, we started with a formation speed of 70 mm/s and slowly increased it at 20 mm/s intervals, with each interval tested 30 times.

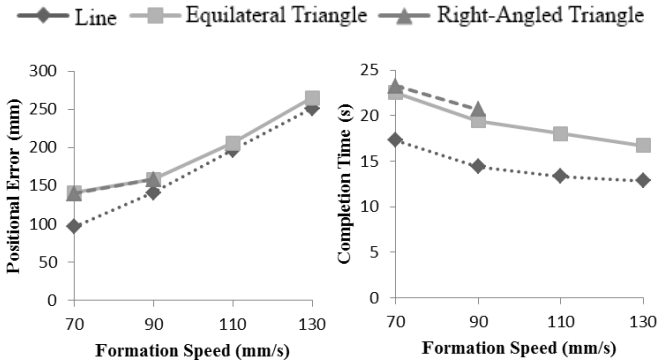

Figure 9. Performance of algorithm as formation speed increases

Fig. 9 shows how quickly the average deviation from ideal FP scaled up with speed. This is due to K_v being optimized for the default speed of 70 mm/s. The right-angled triangle formation was the first to break at a speed of 110 mm/s. Triangle formation fared better as its shape resembles the layout of the obstacle course, hence it was able to scale down in size first before morphing, significantly reducing the deviation from desired FPs at higher speeds. Line and triangle formations eventually broke at the speed of 150 mm/s as the VC was unable to cope with the large d_{error}.

VI. CONCLUSION

A novel adaptive formation control algorithm for wireless mobile robot networks has been created. The algorithm allows a MR network to preserve its formation in the presence of obstacles. The process of forming and maintaining different formation shapes throughout navigation has been thoroughly investigated. A robust obstacle avoidance approach has also been incorporated into the algorithm, allowing the formation to go through obstacle courses without any collision. Formation structure is consistently maintained via formation scaling and morphing. Results show that the FRs deviate from their ideal FP by as little as 96 mm, which is less than the diameter of our robots. Our future research will address i) determining the distance and angle constraints autonomously based on the environment and the number of robots, ii) making formation scaling and morphing more scalable and iii) implementing path planning for the formation as a whole, hence decreasing the chances of robots dropping out of the formation.

REFERENCES

[1] Barca, J C., Sekercioglu, Y. A.: Swarm robotics reviewed. Robotica, Available on CJO 2012, doi:10.1017/S026357471200032X, pp 1-32.

[2] Shao, J., Xie, G., Wang, L.: Leader-following formation control of multiple mobile vehicles. IET Control Theory & Applications 1(2), pp. 545-552 (2007)

[3] Desai, J. P., Ostrowski, J. P., Kumar, V.: Modeling and control of formations of nonholonomic mobile robots. IEEE Transactions on Robotics and Automation 17(6), pp. 905-908 (Dec 2001)

[4] Soorki, M. N., Talebi, H. A., Nikravesh, S. K. Y.: A robust dynamic leader-follower formation control with active obstacle avoidance. IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2011, pp. 1932-1937

[5] Soorki, M. N., Talebi, H. A., Nikravesh, S. K. Y.: A robust leader-obstacle formation control. IEEE International Conference on Control Applications (CCA), 2011, pp. 489-494

[6] Hou, S., Cheah, C., Slotine, J.: Dynamic region following formation control for a swarm of robots. IEEE International Conference on Robotics and Automation (ICRA), Kobe, 2009, pp. 1929-1934

[7] Kuppan, C. R. M, Singaperumal, M., Nagarajan, T.: Distributed planning and control of multirobot formations with navigation and obstacle avoidance. IEEE Recent Adcanves in Intelligent Computational Systems (RAICS), Bandar Sunway, 2011, pp. 621-626

[8] Brandao, A. S., Sarcinelli-Filho, M., Carelli, R., Bastos-Filho, T. F.: Decentralized control of leader-follower formations of mobile robots with obstacle avoidance. IEEE International Conference on Mechatronics (ICM), 2009. pp. 1-6

[9] Barca, J. C., Sekercioglu, Y. A.: Generating formations with a template based multi-robot system. Australasian Conference on Robotics and Automation, Melbourne, 2011

[10] Liang, Y., Lee, H.-H.: Decentralized formation control and obstacle avoidance for multiple robots with nonholonomic constraints. American Control Conference, Minneapolis, 2006

[11] Wang, J., Wu, X.-B., Xu, Z.-L.: Decentralized formation control and obstacles avoidance based on potential field method. International Conference on Machine Learning and Cybernetics, Dalian, 2006, pp. 803-808

[12] Jia, Q., Li, G.: Formation control and obstacle avoidance algorithm of multiple autonomous underwater vehicles (AUVs) based on potential function and behavior rules. IEEE International Conference on Automation and Logistics, Jinan, 2007, pp. 569-573

[13] D'Ademo, N., Li, W.H., Lui, D., Sekercioglu, Y. A., Drummond, T.: eBug - An open robotics platform for teaching and research. Australasian Conference on Robotics and Automation (ACRA), Melbourne, 2011

[14] Barca, J. C., Sekercioglu, Y. A., Ford, A.: Controlling formations of robots with graph theory. 12th International Conference on Intelligent Autonomous Systems (IAS), Jeju Island, pp. 1-8 (2012)

[15] Ogren, P., Leonard, N.E.: Obstacle avoidance in formation. IEEE International Conference on Robotics and Automation (ICRA) 2, pp. 2492-2497 (2003)

[16] Das, A.K., Fierro, R., Kumar, V., Ostrowski, J.P., Spletzer, J., Taylor, C.J.: A vision-based formation control framework. IEEE Transcations on Robotics and Automation 18(5), pp. 813-825 (2002)

[17] Anderson, B., Yu, C., Fidan, B., Hendrickx, J.: Rigid graph control architectures for autonomous formations. IEEE Control Systems 28(6), pp. 48-63 (2008)

[18] Chen, Y.-C., Wang, Y.-T.: Obstacle avoidance and role assignment algorithms for robot formation control. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2007, pp. 4201-4206

[19] Yang, F., Liu, F., Liu, S., Zhong, C.: Hybrid formation control of multiple mobile robots with obstacle avoidance. 8th World Congress on Intelligent Control and Automation (WCICA), 2010, pp. 1039-1044

[20] Seng, W. L.: Distributed Formation Control [Video file]. Retrieved from http://www.youtube.com/watch?v=BYdYBK0nLD4 (2013)