A Bearing-Only Pattern Formation Algorithm for Swarm Robotics

Nicholi Shiell[1,3] and Andrew Vardy[2,3]

[1] Department of Computer Science, Faculty of Science, Memorial University of Newfoundland, St. John's, Canada. nsm152@mun.ca
[2] Department of Computer Science, Faculty of Science and Department of Electrical and Computer Engineering, Faculty of Engineering and Applied Science, Memorial University of Newfoundland, St. John's, Canada. av@mun.ca
[3] http://bots.cs.mun.ca/

Abstract. *Pattern formation is a useful behaviour for a swarm of robots in order to maximize their efficiency at tasks such as surveying. Previous pattern formation algorithms have relied upon various combinations of measurements (bearing, distance, heading, unique identity) of swarm mates as inputs. The ability to measure distance, heading, and identity requires significant sensory and computational capabilities which may be beyond those of a swarm of simple robots. Furthermore, the use of unique identities reduces the scalability, flexibility and robustness of the algorithm. This paper introduces a decentralized pattern formation algorithm using bearing-only measurements to anonymous neighbours as input. Initial results indicate the proposed algorithm improves upon the performance, scalability, flexibility, and robustness when compared to a benchmark algorithm.*

Keywords: Bearing-only control, pattern formation, behaviour-based robotics, swarm robotics

1 Introduction

This paper introduces a decentralized behaviour-based pattern formation algorithm which uses a neighbour-referenced approach [1], and bearing-only measurements to nearby swarm mates as input. The bearings are measured with respect to a common reference direction (i.e North). The proposed algorithm differs from similar bearing-only techniques, for example [12], in two important ways; the definition of the desired formation, and the lack of statically defined reference neighbours. The pattern formation algorithm proposed in this paper will be referred to as the Dynamic Neighbour Selection (DNS) algorithm.

The DNS algorithm is intended for use with a swarm of simple robots with limited sensory and communication abilities. These limitations are imposed to keep the robots cheap and expendable, allowing the DNS algorithm to be used in dangerous applications.

Behaviour-based formation control techniques have a long history in the literature [1, 8, 10]. The techniques presented in these papers make use of bearing,

distance, and heading information about neighbours. Acquiring this information requires significant sensory capabilities. However, bearing can be measured using more limited sensors. Lately there has been a significant amount of work in the study of bearing-only pattern formation algorithms [2–4, 9, 12]. The DNS algorithm is intended to improve upon the scalability, flexibility, robustness, and performance of similar bearing-only algorithms. The cost for these improvements however, is a reduced set of possible formations.

The pattern formation algorithm presented in [12] shares many traits with the DNS algorithm. Both algorithms use bearing-only measurements as inputs, are intended for use with a swarm of simple robots, and are meant for a human operator to dictate formation parameters. For these reasons the algorithm from [12] will be used as a benchmark for evaluating the DNS algorithm. The evaluation will be based on the range of formations which can be constructed, and the scalability, flexibility, robustness, and performance of the algorithms. The pattern formation algorithm from [12] will be referred to as the Static Neighbour Selection (SNS) algorithm. Note that neither algorithm controls the spacing between adjacent robots. As a result robots will not be uniformly distributed along the formation's edges, and the relative lengths of edges may differ. For example, a square formation will actually result in the convex hull of a rectangle.

The performance of the DNS and SNS algorithms will be evaluated using a discrete time simulation. The simulated robots will be controlled by behaviour-based controllers incorporating pattern formation and simple obstacle avoidance. The controller implementing the DNS algorithm will be used in a second simulation meant to test the algorithm's performance under more realistic conditions in preparation for live trials on a group of BuPiGo robots [15]. The first set of simulations will be referred to as the evaluation simulations, and the second will be called the proof-of-concept simulations.

This work is the first step in the development of a behaviour-based solution to the sweep coverage problem [5]. In this problem robots are equipped with a payload sensor (for example an optical camera) in addition to other non payload sensors, and cooperate to collectively survey a given area. This is a task for which pattern formation is quite useful. The DNS algorithm was developed to construct formations which maximize sensor coverage (line or wedge), however, other formations were also found to be possible. In order to effectively cover an entire region the spacing between robots must be controlled. This can not be done with bearing-only data. Future work will explore how environmental cues can be used to determine when adjacent robots are "close enough". This type of indirect communication is known as stigmergy [6]. For example, to conduct optical surveys of the sea floor external lighting must be supplied by the robots. Adjacent robots could sense when they are "close enough" to a neighbour when the illuminated region of their payload camera's field of view has been maximized. This would have the added effect of constricting spacing when visibility worsens, and expanding spacing when it improves.

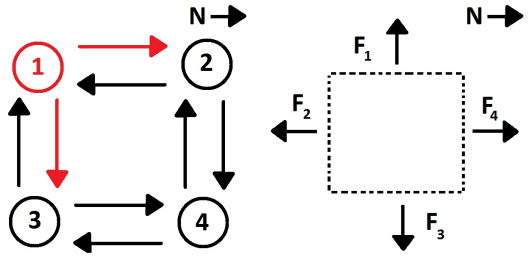
The structure of the paper follows. First the technique used to define formations in both the DNS, and SNS algorithms will be explained. The range

of formations, scalability, flexibility, and robustness of the two formation definitions will then be examined. Next the implementation of the algorithms as behaviours and their incorporation into a behaviour-based controller will be described. Next, a series of numerical simulations which evaluate the performance of the algorithms will be described, and their results discussed. Finally, the paper will conclude with a summary of the results, and description of future work.

2 Formation Definitions

2.1 Static Neighbour Selection

The SNS algorithm [12] defines the desired formation by specifying bearing constraints between a robot and a subset of its swarm mates. Each robot must be uniquely identifiable in order to converge to the desired formation. Figure 1 illustrates how these constraints are defined using the example of 4 robots forming a square. Each robot has a unique identification number (1 to 4), and a set of constraints (target ID and bearing). The bearing constraint is used to construct a unit vector $\boldsymbol{f}$ which the algorithm uses to calculate a control signal.



(a) SNS (b) DNS

Fig. 1. Defining a square formation using the SNS and DNS algorithms. Each arrow in the figures represents a value required by the formation definition. In (a) robot 1 has bearing constraints 0 and $-\pi/2$ with robots 2 and 3 respectively. Similar constraints exist for the remaining 3 robots. In (b) the formation definition is given by only 4 values ($\pi/2$, π, $-\pi/2$,0) regardless of the number of robots. All bearings are measured counter clockwise from North as indicated in the top right corner each figure.

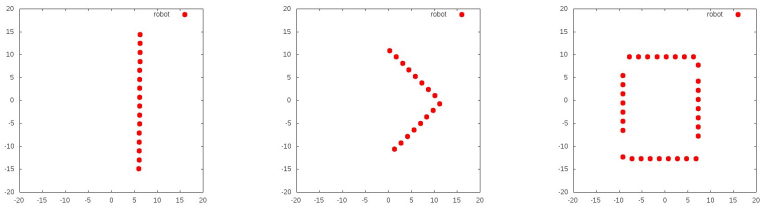
2.2 Dynamic Neighbour Selection

The DNS algorithm defines the desired formation by specifying a set of unit vectors, $\{\boldsymbol{F_i}\}$, perpendicular to the formation's edges. Note, for the remainder of the paper perpendicular refers to a 90 degree rotation counter clockwise (CCW). The swarm of robots is divided into teams and each team assigned one vector

from the set. During the operation of the algorithm the robots do not need to identify the individual or team ID of another robot. Figure 1 illustrates how a square formation is defined by the DNS algorithm.

2.3 Comparison of Formation Definitions

Variety of Formations. The SNS algorithm is able to form any parallel rigid formation [12]. This includes shapes with internal structure (e.g. filled polygons). Based on the formations tested in simulation, the DNS algorithm is limited to line segments and the convex hulls of polygons. Although limited, the formations available to the DNS algorithm are useful in the context of the sweep coverage problem and others [1, 13]. Example formations are shown in Figure 2.



(a) Line (b) Wedge (c) Square

Fig. 2. Examples of the formations studied in simulation. (a) The line maximizes the cumulative field of view (FoV) of sensors perpendicular to the formation. (b)The wedge formation has similar FoV benefits as the line with the added benefit of increased visual contact between robots. (c) The square formation has benefits outside the context of the sweep coverage problem such as perimeter keeping and containment.

Neither algorithm controls the scale of the formation. This is a result of having only bearing information to define the formation. However, with the inclusion of an obstacle avoidance behaviour, which treats other robots as obstacles, a minimum scale is maintained.

Scalability, Flexibility, and Robustness. The advantages of a swarm robotic system as identified by [7] are scalability, flexibility, and robustness. The DNS and SNS algorithms will be evaluated based on these attributes.

Scalability, as defined by [7], in a swarm robotic system means the same control algorithm can be used regardless of swarm size. Both DNS and SNS are decentralized algorithms, and so both scale in this sense. However, both algorithms were developed with human interaction in mind. In order for a human operator to manipulate the formation new information must be transmitted to the swarm (i.e sets of formation normals, or target IDs and bearing values for DNS and SNS respectively). In order for communication with the swarm to be scalable, it must be independent of group size [4]. Figure 1 shows how the

information require to define a formation in the SNS algorithm depends linearly on the group size. Therefore, communication with the swarm is not scalable when using the SNS algorithm. The definition of a formation in the DNS algorithm is independent of group size, and therefore communication with the swarm is scalable with respect to group size. However, communication would scale linearly with the complexity of the formation (ie. number of edges).

Flexibility of a swarm robotic system is the ability to handle changes to group size [7]. The SNS algorithm requires changes to the bearing constraints of neighbours of a lost or added swarm member. The DNS algorithm requires no changes to the information stored by the swarm when members are added or removed.

A source of robustness in swarm robotic systems comes from unit redundancy [7]. That is any member of the swarm can take on the role of another member of the swarm (assuming homogeneous robots). The DNS algorithm maintains unit redundancy by not requiring neighbours to be uniquely identifiable. The SNS algorithm requires robots to be uniquely identifiable and so lacks unit redundancy.

3 Methods

This section will describe the methods used to compare the DNS and SNS algorithms. The model of the robot used in the simulations will be described first. Next the behaviour-based controller used to implement the algorithms, as well as a definition of the behaviours will be given. Finally, the simulations used to evaluate the algorithms will be described.

3.1 Robot Sensors and Drive System

The robot model can be broken into two parts, the sensors, and the drive system. The sensor modalities of the robots are the same in both the evaluation and proof of concept simulations. However, the range, and effects of line of sight differ. Robots will be given their heading with respect to a common direction (compass measurement), the bearing to all visible robots, and the bearing to any robots in physical contact. The drive systems will differ between the simulations. In the evaluation simulations the robots are assumed to be holonomic, and able to change their velocities instantly. In preparation for future live trials the robots in the proof-of-concept simulations will use a differential drive system. A simple proportional control law will convert desired velocities into right and left wheel speeds.

3.2 Behaviour-Based Controller

The simulated robots are controlled by a behaviour-based controller composed of a set of behaviours, $\{\beta_1 ... \beta_n\}$. Each behaviour responds to sensor stimuli

with a desired velocity vector. The controller sums these vectors, Equation 1, to produce the final velocity command.

$$\boldsymbol{v_{cmd}} = \sum_i \boldsymbol{v_i} \tag{1}$$

Where $\boldsymbol{v_{cmd}}$ is the velocity command, and $\boldsymbol{v_i}$ is the i^{th} behaviour's velocity vector with magnitude bounded between [0,1]. Details of the behaviours' stimuli, and responses are given in the following sections.

Two different controllers were defined for the evaluation simulation, controller D (dynamic), and controller S (static). In addition to a pattern formation behaviour, a simple obstacle avoidance behaviour is included in each controller's behaviour set. The behaviour sets for Controller D and S are $\{\beta_{obst}, \beta_{DNS}\}$, and $\{\beta_{obst}, \beta_{SNS}, \beta_{SNS}\}$, respectively. Controller S contains two β_{SNS} behaviours, one for each constraint used to define the formation [12]. Controller D was also used by the proof of concept simulations.

Obstacle Avoidance Behaviour ($\boldsymbol{\beta_{obst}}$). The input for the obstacle avoidance behaviour is a vector, $\boldsymbol{r}$, in the direction of the detected obstacle. If there is no obstacle in physical contact then the response is $\boldsymbol{v_{obst}} = \boldsymbol{0}$. The behaviour response in the presents of an obstacle, $\boldsymbol{v_{obst}}$ is given by Equation 2,

$$\boldsymbol{v_{obst}} = \left(\gamma \boldsymbol{r}^{\perp} - \boldsymbol{r}\right) \tag{2}$$

where γ is a random uniformly distributed number between [0,1], and $\boldsymbol{r}^{\perp}$ is perpendicular to $\boldsymbol{r}$. The random vector was used to break robots apart which previously became stuck in cyclic behaviours. These stuck robots resulted in the DNS algorithm diverging from the desired formation .

SNS Algorithm Behaviour ($\boldsymbol{\beta_{sns}}$). The input for the SNS behaviour is a unit vector, $\boldsymbol{r}$, in the direction of the target robot specified by the bearing constraint. (Section 2.1). The behaviour response, $\boldsymbol{v_{sns}}$ is given by Equation 3,

$$\boldsymbol{v_{sns}} = (\boldsymbol{r} \cdot \boldsymbol{f}^{\perp})\boldsymbol{r}^{\perp} \tag{3}$$

where $\boldsymbol{f}^{\perp}$ is perpendicular to the target bearing, $\boldsymbol{f}$, associated with the target robot, and $\boldsymbol{r}^{\perp}$ is the vector perpendicular to input $\boldsymbol{r}$. This behaviour causes the robot to travel along a circular arc centred on the target robot until the target bearing is achieved.

DNS Algorithm Behaviour ($\boldsymbol{\beta_{dns}}$). The input for the DNS behaviour is the set of unit vectors, $\{\boldsymbol{r_i}\}$, encoding the bearings to all visible swarm mates. The field of view of the robot's bearing sensor is divided into three sensing regions; dead ahead, forward, and rear (See Figure 3). Note that the dead ahead region is a subset of the forward region. If a robot is detected in a region the corresponding boolean flag (B_{da}, B_{fw}, B_{re}) is set to true. The value of these flags determines

the response of the DNS behaviour. The response and its dependence on the boolean flags, is shown in Equation 4.

$$\boldsymbol{v_{dns}} = \begin{cases} -\boldsymbol{F} & B_{re} \cdot (!B_{fw}) \\ \boldsymbol{F}^{\perp} & B_{da} \\ (\boldsymbol{r} \cdot \boldsymbol{F})\boldsymbol{F} & B_{fw} \end{cases} \tag{4}$$

Where $\boldsymbol{r} \in \{\boldsymbol{r_i}\}$ is the bearing vector with the largest projection on to the formation normal, $\boldsymbol{F}$. The DNS behaviour causes the robot to move in the direction of $\boldsymbol{F}$ until one of two conditions are met. The first condition is if a swarm mate is detected in the dead ahead visual region. In this case the direction of travel is rotated by $\pi/2$ radians (CCW). The other condition is if the robot has no swarm mates ahead of it. In this case the behaviour responds by moving backward along $\boldsymbol{F}$. This backward motion is meant to stabilize the edge in the presence of noise.

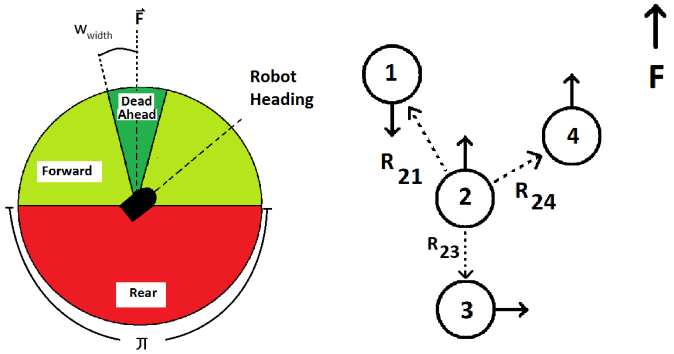


Fig. 3. (a) The sensing regions are defined with respect to the formation normal, $\boldsymbol{F}$, and not the orientation of the robot. The width of the dead ahead region is controlled by the ω_{width} parameter. (b) An illustrated example of the DNS behaviour responses of 4 robots with the same $\boldsymbol{F}$. The formation normal $\boldsymbol{F}$ is shown in the upper right corner. Robot 1 only senses swarm mates to the rear, and response with $-\boldsymbol{F}$. Robots 2 and 4 sense neighbours ahead (outside of the dead ahead region), and in their rear sensing region and response with $\boldsymbol{F}$. Lastly, robot 3 senses a neighbour in its dead ahead region, and so travels along $\boldsymbol{F}^{\perp}$. The vectors $\boldsymbol{R_{21}}, \boldsymbol{R_{23}}, \boldsymbol{R_{24}}$ encode the bearings measured by robot 2 of its neighbours. Vector $\boldsymbol{R_{21}}$, the bearing measured by robot 2 of robot 1, has the largest projection on the $\boldsymbol{F}$, and is used by robot 2 in the calculation of its DNS behaviour's response (see section 3.2).

The parameter, ω_{width}, controls the angular width of the dead ahead visual region (See Figure 3). Two different values of ω_{width} were used depending on the formation being constructed. A value of 18^o was used for line formations. This value should realistically depend on physical dimensions of the robot since

it is intend to help the robot find an empty space along its edge, however the above value was found to be adequate for the simulations. It was found that using the dead ahead visual zone for non-linear formations caused the algorithm to diverge from the desired formation.

3.3 Simulation Software

The performance of both algorithms was compared using a single integrator simulation written in C++ by the author (available for download from GitHub[4]). Performance was evaluated based on the average integrated path length of all robots in the swarm. The robots were modelled as hard disks with radius r_{robot} and mass m_{robot}. Collisions between robots were approximated using 2d kinematics. Each robot has direct control over its instantaneous velocity. Control was provided by either controller S or D. Once a time step, robots were updated with sensor data (Section 3.1). The simulation then waited for a velocity response from all robots, and then updated the robot positions. This loop was repeated until a maximum number of times steps was reached. The simulation parameters used are summarized in Table 1. The simulation assumed the robots were always visible to each other regardless of range or line of sight. To initialize the simulation robots were randomly distributed in a circle of radius r_{deploy}.

The proof of concept simulations for the DNS algorithm were conducted in V-REP [14]. A model of the BuPiGo robot [15], which will be used for live trails of the DNS algorithm, was implemented in V-REP. The BuPiGo is equipped with an omnidirectional camera which served as a proxy for the bearing-only sensor. The BuPiGo does not have an omnidirectional contact sensor, therefore one was added to the model for the purposes of the simulation. Simulated sensor data from V-REP was sent to the behaviour-based controller via an interface with ROS [11]. The V-REP simulation included nonholonomic constraints, as well as limited sensor visibility (line of sight and range), and limited sensor resolution. It was assumed the robots could identify each other using a blob detection algorithm on the images captured by the omnidirectional camera.

Table 1. Summary of parameters used during the evaluation simulations. Note ω_{width}, and r_{max} relate to behaviours and are described in Section 3.2

Parameter	Value	Description
Δt	0.05	Length of time step
maxTimeSteps	100000	Max number of time steps
r_{deploy}	50	Radius of initial deployment
ω_{width}	18^o or 0^o	Angular width of dead ahead visual region (Section 3.2)
r_{robot}	5.0	Robot radius
m_{robot}	1.0	Robot mass

[4] https://github.com/nicholishiell/DiskSimulation

4 Experimental Results

The performance of the algorithms was evaluated when constructing line, wedge, and square formations with various group sizes (8, 16, 20, and 32 robots). Each set of evaluation parameters (algorithm, formation, group size) was simulated 100 times and the metric values recorded for each run. The average metric values and standard deviation over all runs were calculated, and the results shown graphically in Figure 4.

4.1 Performance Metric

The performance metric, α, is the average integrated path length of robots in the swarm. A distance based metric was used to evaluate the performance of the algorithms as an analogy for energy use, which is a limiting factor in mobile robotic systems. The performance metric is defined in Equation 5,

$$\alpha = \frac{\sum_i d_i}{N} \tag{5}$$

Where d_i is the integrated path length of the i^{th} robot,and N is the total number of robots in the swarm. The integrated path length of each robot is defined by Equation 6.

$$d_i = \sum_j v_i(j) \Delta t \tag{6}$$

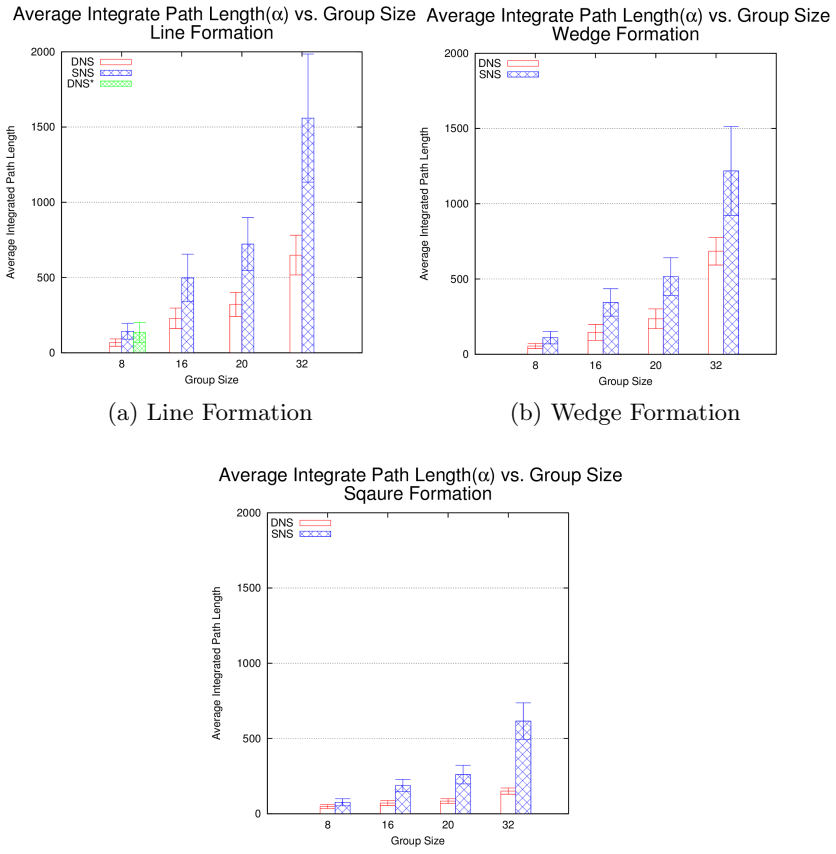
Where d_i is the integrated path length of the i^{th} robot, the sum is over all time steps j, the velocity of the i^{th} robot at time step j is $v_i(j)$, and Δt is the time step used in the simulation.

4.2 Simulation Results

The plots shown in Figure 4 summarize the results from the evaluation simulations. The plots show the average integrated path lengths (α) versus group size, for each formation type. The results indicate the algorithms differ in their dependence on initial deployment, and group size. A comparison of the DNS algorithm's performance across different formation types and group sizes shows the simulation data is internally consistent, and demonstrates the effects of the dead ahead sensing region used in line formation.

The standard deviation of α values associated with the SNS algorithm are relatively large compared to the DNS algorithm. This shows there was a strong variation in average integrated path lengths between runs with the same formation and group size. The only difference between these runs was in the initial positions of the robots. This indicates that the SNS algorithm depends more strongly on initial deployment than the DNS algorithm.

The performance of the algorithms, as measured by α, diverge quickly for group sizes larger than 8. The SNS algorithm shows a stronger dependence on group size than the DNS algorithm, and this trend can be seen in all formations



(a) Line Formation (b) Wedge Formation

(c) Square Formation

Fig. 4. Results of evaluation simulations. The bar in plot (a) labelled DNS* is the result of running the DNS algorithm for a line formation without the dead ahead region.

tested. Therefore, the performance of the DNS algorithm scales better with group size than the SNS algorithm.

Comparing results from the DNS algorithm for a line formation of 8 robots without dead ahead region, (DNS* in Figure 4(a)), a wedge of 16 robots, and a square of 32 robots, demonstrates the simulated data is internally consistent. The formation of a wedge of 16 robots involves forming 2 lines of 8. Similarly a square of 32 robots involves the construction of 4 lines of 8. The average α in these three cases should be similar. This similarity is seen in the plots shown in Figure 4. Differences in these values can be attributed to robots colliding, and avoiding each other.

Comparing the results of the DNS algorithm when constructing a line formation, with (DNS) and without (DNS*) dead ahead regions shows the regions

utility. Using the dead ahead region decreased α by a factor of approximately 2. Unfortunately, the use of this sensing region in more complex formations (wedge and square) caused the algorithm to diverge.

A video showing the proof of concept simulations in V-REP is available online[5]. The video shows the DNS algorithm constructing line, wedge, and square formations with a group size of 12 robots.

5 Conclusion

This paper introduced a decentralized bearing-only pattern formation algorithm, known as the Dynamic Neighbour Selection algorithm. The DNS algorithm was compared to a similar algorithm presented in [12] using a single integrator simulation, and by comparing the impact of their differing formation definitions. The DNS algorithm was further tested in a more realistic V-REP simulation. A cooperative behaviour-based controller was used in both simulations which integrated each pattern algorithm with a basic obstacle avoidance behaviour. The formation definition comparison shows the DNS algorithm improves upon the scalability, flexibility, and robustness of the SNS algorithm. Furthermore, the simulation shows the DNS algorithm to be more efficient than the SNS algorithm for group sizes greater than 8. The simulations also showed the DNS algorithm to be less dependent on group size, and initial deployment than the SNS algorithm.

Although the DNS algorithm has improved upon some aspects of bearing-only pattern formation algorithms, it still has limitations similar to all bearing-only algorithms. The relative lengths of polygon segments are not controlled, and the density of robots along a segment of the formation is not uniform. In the context of a solution to the sweep coverage problem these limitations are significant and will need to be addressed.

Before the DNS algorithm can be integrated into a behaviour-based solution to the sweep coverage problem a number of tasks remain. A theoretical proof of convergence and shape limitations of the DNS algorithms would strengthen the case for the robustness of the algorithm. Another task is the development of a behaviour to space the robots at "effective" intervals using environmental cues. Lastly, the control software developed in ROS for the V-REP simulation must be ported to the BuPiGo robots and live trials of the DNS algorithm conducted to confirm the results reported in this paper.

6 Acknowledgements

The authors of this paper would like to thank the anonymous reviewers for their helpful comments and insights.

[5] https://youtu.be/D7ZB4lNxJsE

References

1. Balch, T., Arkin, R.C.: Behavior-based formation control for multirobot teams. IEEE Transactions on Robotics and Automation 14(6), 926–939 (1998)
2. Bishop, A.N., Basiri, M.: Bearing-only triangular formation control on the plane and the sphere. In: Control & Automation (MED), 2010 18th Mediterranean Conference on. pp. 790–795. IEEE (2010)
3. Bishop, A.N., Shames, I., Anderson, B.: Stabilization of rigid formations with direction-only constraints. In: Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on. pp. 746–752. IEEE (2011)
4. Franchi, A., Giordano, P.R.: Decentralized control of parallel rigid formations with direction constraints and bearing measurements. In: Decision and Control (CDC), 2012 IEEE 51st Annual Conference on. pp. 5310–5317. IEEE (2012)
5. Gage, D.W.: Command control for many-robot systems. Tech. rep., DTIC Document (1992)
6. Grassé, P.P.: La reconstruction du nid et les coordinations interindividuelles chezbellicositermes natalensis etcubitermes sp. la théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs. Insectes sociaux 6(1), 41–80 (1959)
7. Martinoli, A.: Collective complexity out of individual simplicity. Artificial Life 7(3), 315–319 (2001)
8. Monteiro, S., Bicho, E.: A dynamical systems approach to behavior-based formation control. In: Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on. vol. 3, pp. 2606–2611. IEEE (2002)
9. Moshtagh, N., Michael, N., Jadbabaie, A., Daniilidis, K.: Vision-based, distributed control laws for motion coordination of nonholonomic robots. IEEE Transactions on Robotics 4(25), 851–860 (2009)
10. Reynolds, C.W.: Flocks, herds and schools: A distributed behavioral model. Computer Graphics 21(4), 25–34 (1987)
11. ROS: Robot Operating System, `http://www.ros.org`
12. Schoof, E., Chapman, A., Mesbahi, M.: Bearing-compass formation control: A human-swarm interaction perspective. In: American Control Conference (ACC), 2014. pp. 3881–3886. IEEE (2014)
13. Sousselier, T., Dreo, J., Sevaux, M.: Line formation algorithm in a swarm of reactive robots constrained by underwater environment. Expert Systems with Applications 42(12), 5117–5127 (2015)
14. V-REP: Virtual Robot Experimentation Platform, `http://www.coppeliarobotics.com/`
15. Vardy, A., Shiell, N.: Bupigo: An open and extensible platform for visually-guided swarm robots (2015)