Improved swarm formation and perimeter compression in self-healing swarms.

Neil Eliot[1,*], David Kendall[2], and Michael Brockway[2]

[1]*Northumbria University, Faculty of Engineering and Environment, Department of Computer and Information Sciences*
[2]*Hexham University, Faculty of Computer Science*
[*]*Corresponding author: Dr Neil Eliot, neil.eliot@northumbria.ac.uk*

***Abstract*—Perimeter Compression is a technique where by a void reducing effect can be added to a basic swarming algorithm. The effect is dependant upon perimeter identification and is controlled by applying three weighting factors to the existing swarming formulae. One to the cohesion calculation, one the repulsion calculation, and one to the repulsion field size.**

I. Introduction

When cohesion and repulsion field effects (sometimes referred to as potential fields [2], [9], [12], [22], [23], [17]) are used to create a swarming effect, the stable structures that develop are limited to either straight edges or partial lattices [8]. The maintenance of a well-structured swarm is crucial to effective deployment for applications such as reconnaissance or artificial pollination, where 'blind spots' are best eliminated [7], and containment, where the swarm is used to surround an object or region [5]. Over time swarms form regular shapes [19] and perimeters form of partial lattices that may contain so-called *anomalies*, such as concave 'dents' or convex 'peaks' [10]. These anomalies contribute to the disruption of an otherwise well-structured swarm. The key, therefore, is to ensure that these *anomalies* are dynamically removed from a swarm.

Perimeter compression is a technique that creates a 'pull' effect between perimeter agents. It is dependant upon perimeter agent identification as discussed by Eliot et. al. in [8], [9], [10] and discussed in Section IV.

The aim of this new algorithm is to reduce the spacing between perimeter-based agents. Figure 1) shows an agent and it fields. S_b is the sensor field. O_b is the obstacle field. C_b is the cohesion field and R_b is the repulsion field. The implementation involves introducing three controlling weights; p_{kc} which can be used to increases the magnitude of the cohesion vector. p_{kr} which can be used to modify the repulsion vector and p_c which can be used to alter the repulsion filed of agents. When trying to effect a perimeter compression the following assumptions are required.

Assumption 1: $p_c \leq 1$
Assumption 2: $p_{kr} \leq 1$
Assumption 3: $p_{kc} \geq 1$

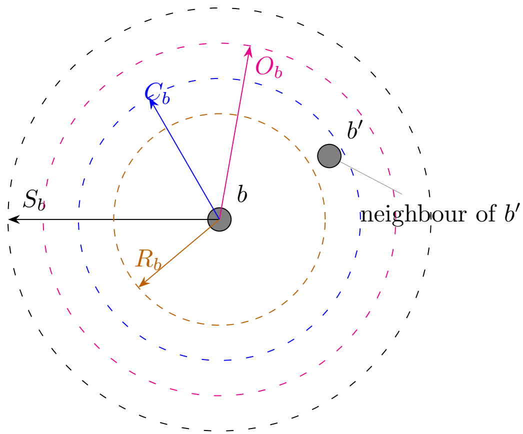


Fig. 1: Agent Fields

II. Related work

As far back as 1987 swarm theory has adopted the use of field effects/potential fields to coordinate agents [20] and this has continued since then in an attempt to improve the structure of a swarm, coordinate obstacle avoidance, and improve navigation [1], [2], [3], [4], [9], [12], [14], [22], [23]. Improvements to the basic structure of swarms has developed through the likes of a prototype framework for self-healing swarms that was developed by Dai et al. They considered how to manage agent failure in hostile environments [6]. This was similar to work by Vassev and Hinchey, who modelled swarm movement using the ASSL (Autonomic System Specification Language) [26]. This technique was employed by NASA (US National Aeronautics and Space Administration) for use in asteroid belt exploration as part of their ANTS (Autonomous Nano Technology Swarm) project. However, this work is focused towards failure of an agent's internal systems, rather than on the removal of anomalies in a swarm distribution. This need for formation control is also discussed by Speck and Bucci with respect to the diverse applications of swarms and the need to control a swarms structure [24].

In the context of swarm structure maintenance, Roach et al. focussed on the effects of sensor failure, and the impact that has on agent distribution [21]. Lee and Chong identified the issue of concave edges within swarms in an attempt to create regular lattice formations [16], and the main focus of their work is the dynamic restructuring of inter-agent formations. Ismail and Timmis demonstrated the use of *bio-inspired* healing using *granuloma formation*, a biological method for encapsulating an antigen [15]. They have also considered the effect failed agents can have on a swarm when traversing a terrain [25].

This paper proposes an alternative approach to agent coordination that can be used to induce a void reduction effect through perimeter compression. This is an extension of the work presented by Eliot et al. [10], Ismail and Timmis [15], [25], and on the work of McLurkin and Demaine on the detection of perimeter types [18]. However, perimeter type identification requires a communications infrastructure to allow the perimeter angle to be calculated. Communications within swarm formations limits swarm sizes and introduces performance problems [11]. The technique employed in this paper does not explicitly require the identification of the perimeter type as it would limit the size of the swarm[10], [16] and is therefore a reduced perimter detection algorithm to identify *any* perimeter.

III. Basic swarming model

In the Original work by Eliot et. al. the resultant vector of an agent was calculated using Equation 1. Where k_c, k_r, k_d, k_o are weighting factors for the summed vectors associated with each interaction. i.e. v_c, v_r, v_d, v_o for cohesion, repulsion, direction and object avoidance respectively.

$$v(b) = k_c v_c(b) + k_r v_r(b) + k_d v_d(b) + k_o v_o(b) \quad (1)$$

Equation 1 shows the movement vector as a linear combination of a cohesion vector v_c tending to move b towards its neighbours, a repulsion vector v_r tending to move b away from its neighbours, a direction vector v_d tending to move b towards a goal, and a vector v_o tending to steer it away from obstacles. $k_c, k_r, ...$ are the scalar coefficients of the the linear combination.

This paper does not consider goals or obstacles so we assume $k_d = k_o = 0$ and omit the third and fourth terms.

A. Cohesion

The cohesion component is calculated based on the proximity of neighbours. Where $n_c(b)$ is the set of neighbour agents for b (Eq. 2). The inclusion of an agent from a swarm (S) in by the agent's cohesion field (C_b).

$$n_c(b) = \{b' \in S \;:\; b' \neq b \wedge \|b' - b\| \leq C_b\} \quad (2)$$

The effect of an agent being within this set is that it will generate a vector that should 'encourage' agents to maintain their proximity. i.e. generate a cohesive swarm. The general weighted (k_c) formula for agents to maintain their proximity is to direct their motion towards the central point of all neighbouring agents as shown in Equation 3. Where $|n_c(b)|$ denotes the cardinality of $n_c(b)$. This formula includes the k_c quotient that allows the cohesion effect to be 'balanced' with respect to other vector influences as described in [8], [9], [10].

$$v_c(b) = \frac{1}{|n_c(b)|} \sum_{b' \in n_c(b)} (b' - b) \quad (3)$$

B. Repulsion

The repulsion component of an agent's movement is calculated from interaction with its neighbours $n_r(b)$ (Eq. 4) in a swarm ($\mathcal{S}$) that are within the agent's (b) repulsion field (R_b).

$$n_r(b) = \{b' \in \mathcal{S} : b \neq b' \wedge |b' - b| \leq R_b)\} \quad (4)$$

The repulsion is then calculated as the average of all the vectors created by the agent (b) to the neighbours (b') (Eg. 5) and its proximity ($\|b' - b\| - R_b$). Where $|n_r(b)|$ denotes the cardinality of $n_r(b)$.

$$v_r(b) = \frac{1}{|n_r(b)|} \sum_{b' \in n_r(b)} (\|b' - b\| - R_b\,)\,(\widehat{b' - b}) \quad (5)$$

Here, $\widehat{b' - b}$ denotes $b' - b$ normalized to unit length.

IV. Perimeter detection

For perimeter compression to be applied to a swarm the perimeter needs to be detected. A perimeter can be defined as a continuous 'surface' of agents that are not enclosed by other agents. These agents may form an outer (green) or inner (red) boundary (Fig. 2).

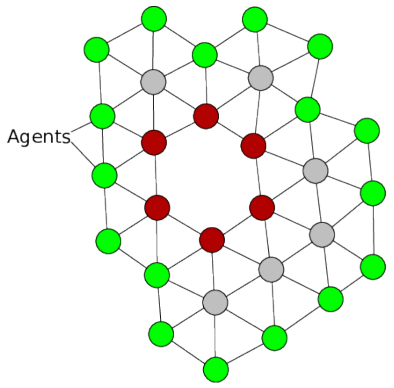


Fig. 2: Outer and inner swarm perimeters.

The detection process is achieved using a cyclic analysis of the agents that surround an agent (Fig. 3). Ghrist et al. discusses a similar technique using sweep angles [13] as does McLurkin et al [18].

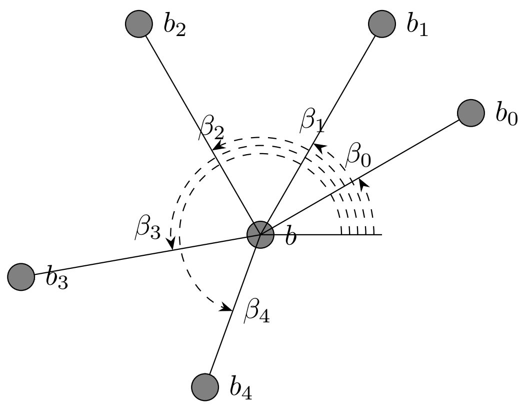


Fig. 3: Agent neighbours

The initial detection of the agents is based on the distance that each agent in the swarm is away from the current agent as described in Section III-A and shown in Equation 2. The perimeter detection set is based on the azimuth angle (β) of each neighbour agent as shown in Fig. 3.

Let $n_r(b) = \langle b_0, b_1, ...b_{n-1}\rangle$ be an enumeration of the neighbours of b in order of increasing azimuth angle. The *azimuth angle* with respect to b of b_i is the angle β_i which vector $\vec{bb_i} = b_i - b$ makes with the positive x axis: $\beta_i = \text{atan2}((b_i - b)_y, (b_i - b)_x)$.

An agent is considered to be on the perimeter of the swarm if it is not enclosed by the polygon defined in the sorted set $n_c(b)$ if it has a 'gap' as shown by agents b_4 and b_0 in Figure 3 (assuming $\|b_4 - b_0\| > C_b$). Also when the polygon defined by $n_c(b)$ is closed but two or more neighbour agents are compressed to the point that they are within range C_b but are 'behind' agent b. This condition can be detected based on any neighbour pair angle being greater than π as shown in Figure 4 with agent b_0 and b_3 assuming the neighbour agent pair are within range assuming $\|b_3 - b_0\| <= C_b$.

A gap can therefore be defined as any two consecutive neighbours $b' = b_i, b'' = b_{(i+1)\%n}$ [the *next* neighbour in the azimuth-ordered *circular list*] when *either*

- b', b'' are out of cohesion range (they are not neighbours of each other) *or*
- b', b'' subtend a reflex angle at b. That is, $\beta' - \beta$ (or $\beta' - \beta + 2\pi$ if the former is negative) is $> \; pi$.

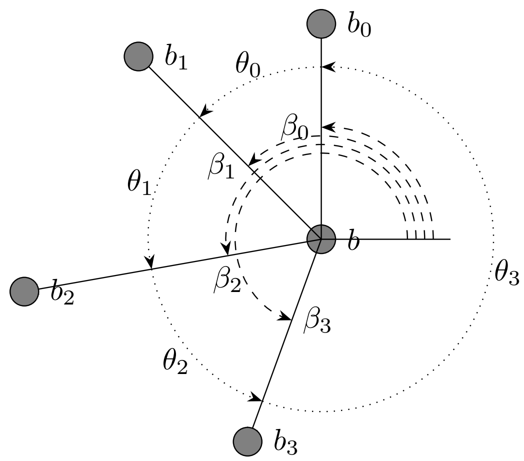


Fig. 4: Agent neighbour angles

V. Gap reduction model (to be moved below the next section when complete!)

An enhancement to the basic model that has proven useful in combination with the enhancements detailed, especially in reducing internal voids in the swarm, is to introduce another term on the right of Equation 8. This gap reducing model is based upon the model described by Eliot et al. in [10].

For instance (omitting obstacles, destinations for now),

$$v(b) = v_c(b) + v_r(b) + v_g(b) \tag{6}$$

where $v_g(b)$ is a vector which will, in the case that a gap has been identified in the perimeter test for b, will contribute a motion of b toward the midpoint of the gap.

Suppose that $n_r(b) = \langle b_0, b_1, ...b_{n-1}\rangle$ are the neighbours of b in azimuth angle order, and let the *first* gap found in the perimeter test be $b' = b_i$ to $b'' = b_{(i+1)\%n}$. Then k_g can be defines as Equation 7.

$$v_g(b) = k_g\left(\frac{b'+b''}{2} - b\right) = k_g\frac{((b'-b)+(b''-b))}{2} \quad (7)$$

k_g is a weighting for the gap-filling vector allowing the combination of it with the other motion vectors (cohesion, repulsion, ...) to be "tuned".

VI. New Inter-agent Models

In this section we will discuss the application of the new p_r, p_{kc} and p_{kr} parameters and the application of the gap reduction which can be used to introduce a stabilising property to the swarm perimeter.
The new model requires each agent to modify their inter-agent repulsion and cohesion vectors based upon their perimeter status and each neighbour's perimeter status. The basic perimeter control technique is shown in Equation 8). The cohesion and repulsion weighting factors (k_c, k_r, p_{kr}, p_{kc}, p_r) are integrated into the equations and 14 and 16 so equation 1 now reads, simply,

$$v(b) = v_c(b) + v_r(b) \quad (8)$$

A. p_r, p_{kr} and p_{kc}

The basic method of controlling the inter-agent relationships can be used to implement perimeter compression by reducing the repulsion field of a perimeter-perimeter agent relationship to allow agents to move more closely to each other before they are repelled. Likewise the cohesion can be manipulated to overcome some of the neighbouring inner-agent repulsion forces that may prevent the agents to move.

p_{kr} (Eq. 9), p_{kc} (Eq. 10), and p_r (Eq. 11) are three, 2x2-arrays.

$$p_{kr} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \quad (9)$$

$$p_{kc} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \quad (10)$$

$$p_r = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \quad (11)$$

These arrays (via the index) provide a scalar weighting to specific agents within the swarm based upon their perimeter status as shown is Equation 12 where $\mathsf{erf}(b,b')$ is the inter-perimeter-agent repulsion weighting for the repulsion radii, $\mathsf{ekc}(b,b')$ is the effective cohesion weighting, and $\mathsf{ekr}(b,b')$ is the effective repulsion weighting and where $\mathsf{prm}(b)$ returns the perimeter status of an agent which is then used as an index into the arrays: 1 or 0 according as b is/is not on the perimeter.

$$\begin{aligned} \mathsf{erf}(b,b') &= p_r[\mathsf{prm}(b),\mathsf{prm}(b')]R_b \\ \mathsf{ekc}(b,b') &= p_{kc}[\mathsf{prm}(b),\mathsf{prm}(b')]k_c(b) \\ \mathsf{ekr}(b,b') &= p_{kr}[\mathsf{prm}(b),\mathsf{prm}(b')]k_r(b) \end{aligned} \quad (12)$$

B. Agent repulsion (p_r, $\mathsf{erf}(b,b')$, $\mathsf{ekr}(b,b')$)

The repulsion component of the swarm model is now applied as shown in Equations 12, 13 and 14.

$$v_r(b) = \frac{1}{|n_r(b)|}\sum_{b'\in n_r(b)} \mathsf{ekr}(b,b')\,(\|b'-b\| - \mathsf{erf}\,(b,b')\,)\,(\widehat{b'-b}) \quad (13)$$

$$n_r(b) = \{b' \in \mathcal{S} : b \neq b' \wedge \|b'-b\| \leq \mathsf{erf}(b,b')\} \quad (14)$$

The introduction of the matrices allows for specific relationships to effect the movement of agents. Using a J matrix results no perimeter repulsion change and all agents use the same repulsion field and the repulsion effect is monolithic within the swarm. However if $P_r[1][1] < 0$ as shown in Equation 15) and both agents are perimeter-based i.e. $\mathsf{prm}(b)$ and $\mathsf{prm}(b')$ $(true, true)$ result in $\mathsf{erf}(b,b')$ returning 0.4 from the matrix. This effectively reduces the R_b field allowing neighbouring agents to move more closely together which effects a compression effect. This is then combined with a weighting effect from the p_{kr} matrix which is used by $\mathsf{ekr}(b,b')$. This function returns a weighting value for the overall inter-agent repulsion relationship. This can be used to reduce the effect of specific vectors in the reopulsion calculations. When this effect is combined with a perimeter-based cohesion (§ VI-C) increase a sustainable compression is possible as shown in figure 5.

$$p_r = \begin{pmatrix} 1 & 1 \\ 1 & 0.4 \end{pmatrix} \quad (15)$$

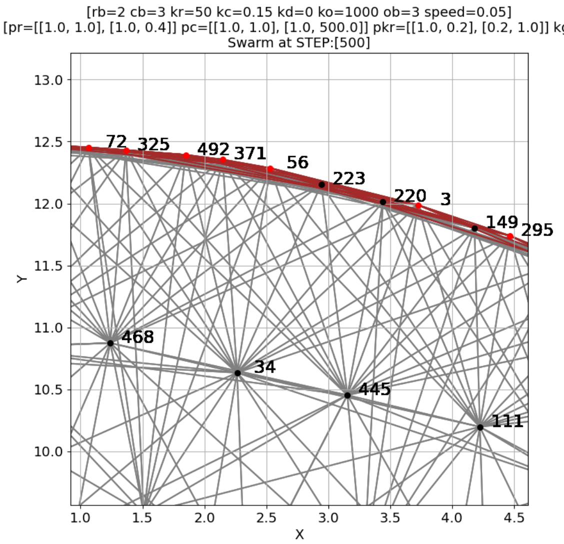


Fig. 5: Perimeter compression.

C. Inter-agent cohesion (p_{kc}, $\mathsf{ekc}(b, b')$)

The cohesion component of the swarming effect (p_{kc}) is applied when an agent (b) and its neighbour (b') are both within range (C_b). $\mathsf{ekc}(b, b')$ works in a similar manor to $\mathsf{ekr}(b, b')$. $\mathsf{ekc}(b, b')$ refrences the p_{kr} matrix to obtain a weighting that is based upon the two agents status. If the agents are not both perimeter-based then the agents cohesion vector is scaled by the scalar value that is referenced in the matrix (Eq. 12). e.g. If both agents are perimeter-based and the matrix is as shown in equation 17 then the effect of the additional cohesion-compression weighting will be increase the generated cohesion-vector for that individual relationship (Eq. 16) which causes the agent to have a tendency to move towards that neighbour.

$$v_c(b) = \frac{1}{|n_c(b)|} \sum_{b' \in n_c(b)} \mathsf{ekc}(b, b')\ (b' - b) \tag{16}$$

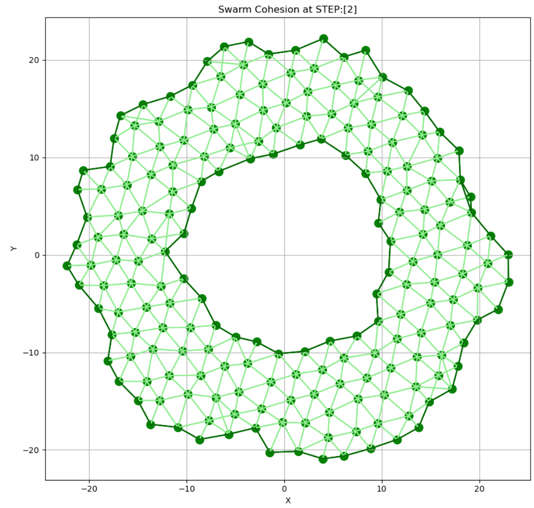
$$p_{kc} = \begin{pmatrix} 1 & 1 \\ 1 & 500.0 \end{pmatrix} \tag{17}$$

==== STILL TO BE WORKED ON =====
Figure 6 shows how the compression effect can remove a void from a swarm by surround an obstacle in a similar manner to the method described in [10].

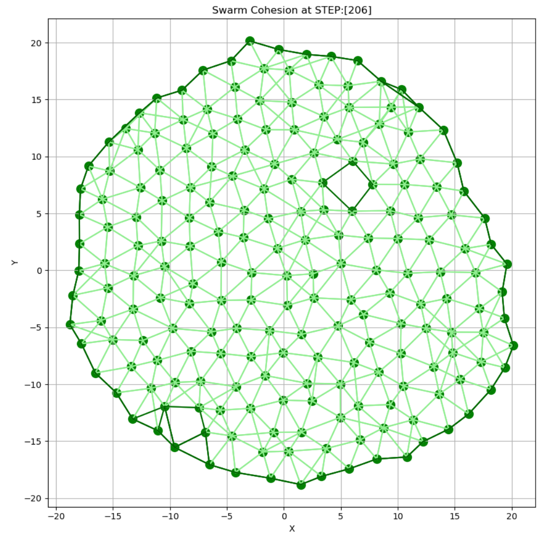
D. Inner compression model

$$p_{kr} = \begin{pmatrix} 1 & x \\ 1 & 1 \end{pmatrix} \tag{18}$$

$$p_{kc} = \begin{pmatrix} 1 & y \\ 1 & 1 \end{pmatrix} \tag{19}$$



(a) Void removal start



(b) Void removal finish

Fig. 6: Void removal through perimeter compression

1) Inner + Gap compression model:

E. Outer compression model

$$p_{kr} = \begin{pmatrix} 1 & x \\ x & 1 \end{pmatrix} \tag{20}$$

$$p_{kc} = \begin{pmatrix} 1 & y \\ y & 1 \end{pmatrix} \tag{21}$$

1) Outer + Gap compression model:

VII. Experimental Results

A. Baseline

For all the experiments the parameters used to create the basic swarming effect are shown in Table I. Where C_b is the cohesion field, k_c is the cohesion weighting, R_b is the repulsion field, k_r is the repulsion weighting and k_g is the weighting factor applied in the comparison of the gap reduction algorithm discussed in [10]. The swarm consists of 200 agents which are distributed with a void at the centre. These initial parameters create a hexagonal-based distribution of agents that stabilise as shown in Figure 7. This basic swarm is used as the initial state for all the experiments.

Swarming Variable	Value
C_b	3.00
k_c	0.15
R_b	2.00
k_r	50.00
k_g	25.00

TABLE I: Swarming effect parameters

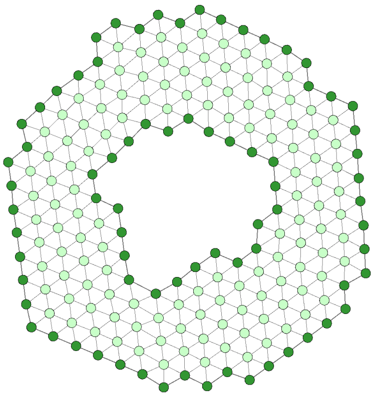


Fig. 7: Baseline swarm in stabilised configuration.

When the simulation is ran with no compression the changes are identified using a magnitude-based metric [9]. The resultant magnitudes generated are shown in figure 8. These states are used as the baseline for the experiments to measure the effects of the compression algorithm and compare the new algorithm to the existing void reduction algorithm.

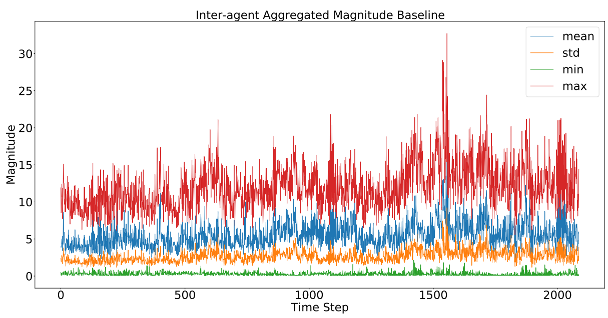


Fig. 8: Baseline swarm in stabilised configuration.

B. Compression Effects

The compression effect parameters are shown in tables II and III

Pr/Pc	10	20	30	40	50
0.1	0.1/10	0.1/20	0.1/30	0.1/40	0.1/50
0.2	0.2/10	0.2/20	0.2/30	0.2/40	0.2/50
0.3	0.3/10	0.3/20	0.3/30	0.3/40	0.3/50
0.4	0.4/10	0.4/20	0.4/30	0.4/40	0.4/50
0.5	0.5/10	0.5/20	0.5/30	0.5/40	0.5/50
0.6	0.6/10	0.6/20	0.6/30	0.6/40	0.6/50
0.7	0.7/10	0.7/20	0.7/30	0.7/40	0.7/50
0.8	0.8/10	0.8/20	0.8/30	0.8/40	0.8/50
0.9	0.9/10	0.9/20	0.9/30	0.9/40	0.9/50

TABLE II: Experiment parameters 1

Pr/Pc	60	70	80	90	100
0.1	0.1/60	0.1/70	0.1/80	0.1/90	0.1/100
0.2	0.2/60	0.2/70	0.2/80	0.2/90	0.2/100
0.3	0.3/60	0.3/70	0.3/80	0.3/90	0.3/100
0.4	0.4/60	0.4/70	0.4/80	0.4/90	0.4/100
0.5	0.5/60	0.5/70	0.5/80	0.5/90	0.5/100
0.6	0.6/60	0.6/70	0.6/80	0.6/90	0.6/100
0.7	0.7/60	0.7/70	0.7/80	0.7/90	0.7/100
0.8	0.8/60	0.8/70	0.8/80	0.8/90	0.8/100
0.9	0.9/60	0.9/70	0.9/80	0.9/90	0.9/100

TABLE III: Experiment parameters 2

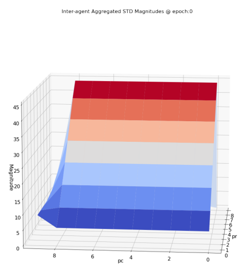


Fig. 9: Experiment Start

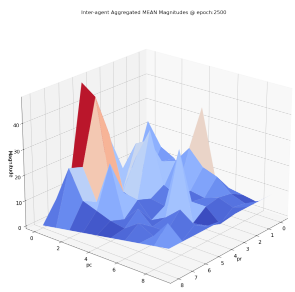


Fig. 10: Experiment Middle

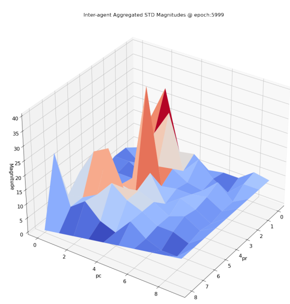


Fig. 11: Experiment End

The first area of comparison is the effect of the algorithms on the number of perimeter agents. The baseline swarm's agents oscillates but remain in a relatively stable state with a constant number of perimeter agents and the internal anomaly persists (Fig. 7). The maximum and minimum number of perimeter agents is shown in table IV.

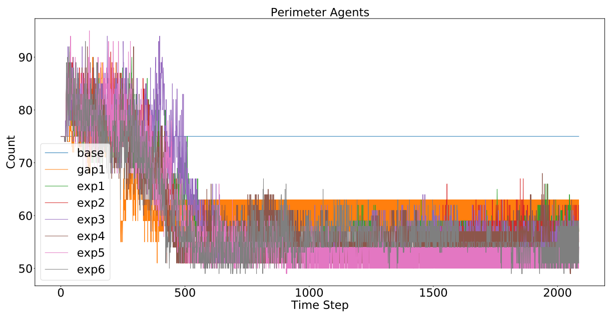


Fig. 12: Perimeter Count of baseline, gap reduction and perimeter compression.

Comp.	Base	Void	1	2	3	4	5	6
Max	75	90	90	90	94	92	95	93
Min	75	51	51	51	51	49	49	49
Mean	75	62	59	58	60	59	57	59
Std	0	6	9	10	10	8	10	8

TABLE IV: Perimeter agents

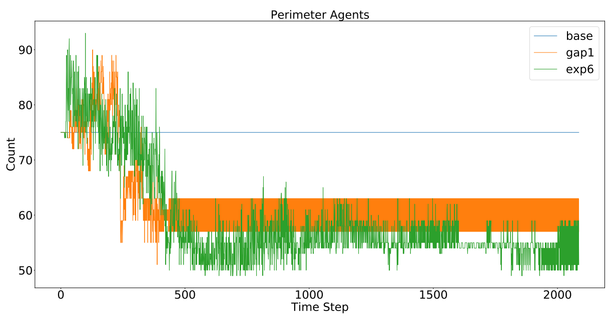


Fig. 13: Perimeter Count of baseline, gap reduction and Experiment 6.

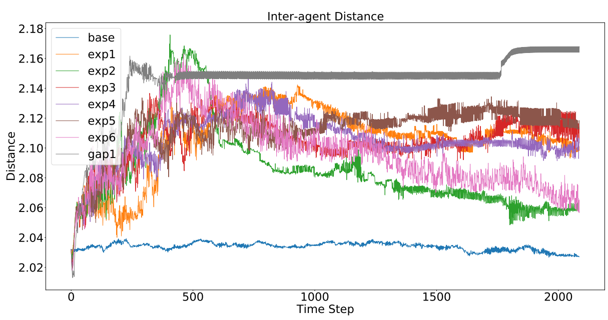


Fig. 14: Distance metric of baseline, gap reduction and perimeter compression.

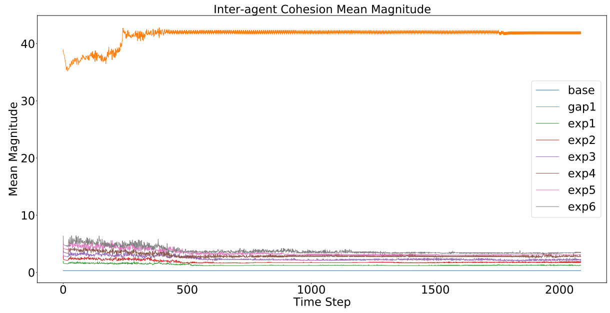


Fig. 15: Inter-agent Cohesion Mean.

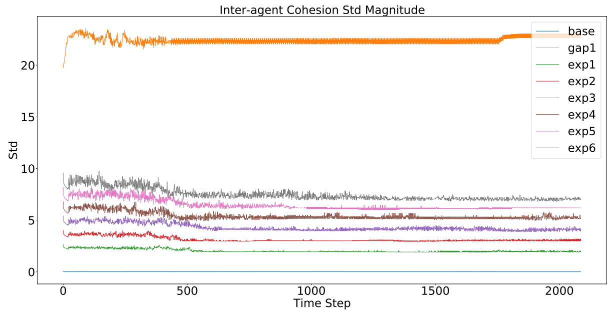


Fig. 16: Inter-agent Cohesion Std.

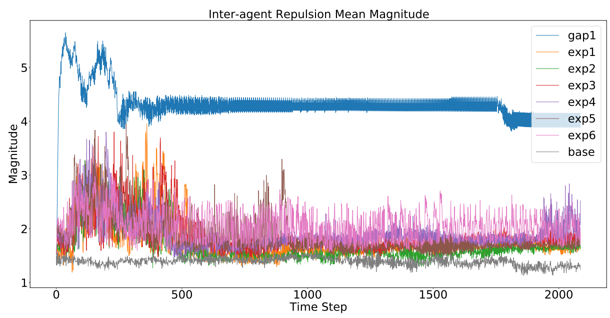


Fig. 17: Inter-agent Repulsion Mean.

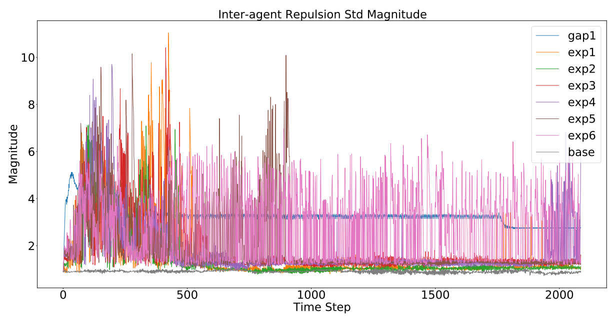


Fig. 18: Inter-agent Repulsion Std.

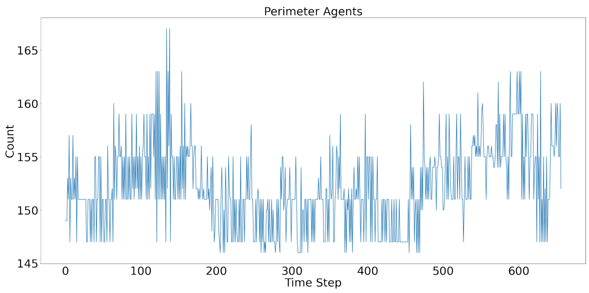


Fig. 19: Baseline swarm in stabilised configuration.

C. Gap compression

D. Perimeter compression

Compression 1

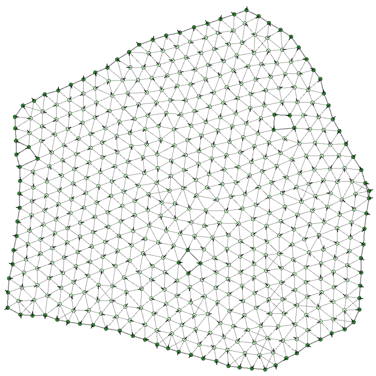


Fig. 20: Baseline swarm in with compression set 1 resultant configuration.

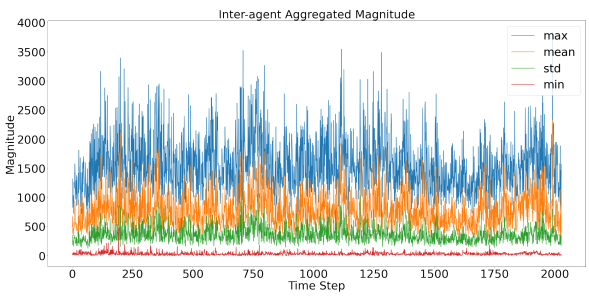


Fig. 21: Baseline swarm in with compression set 1.

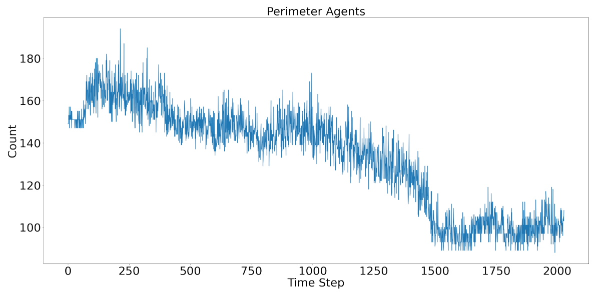


Fig. 22: Baseline swarm in with compression set 1.

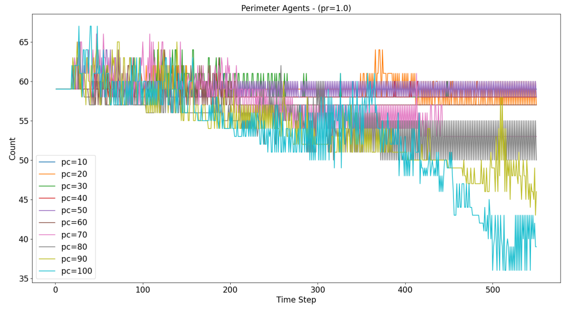


Fig. 23: Sample

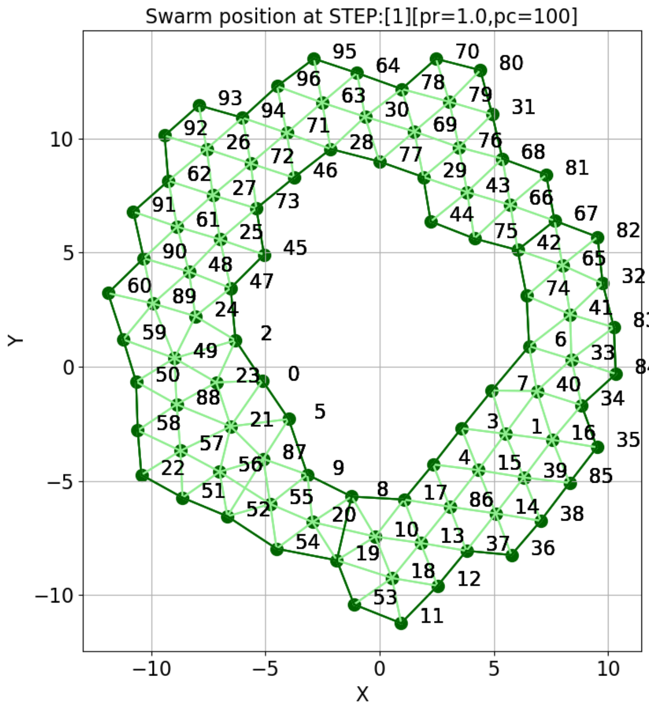


Fig. 24: Sample

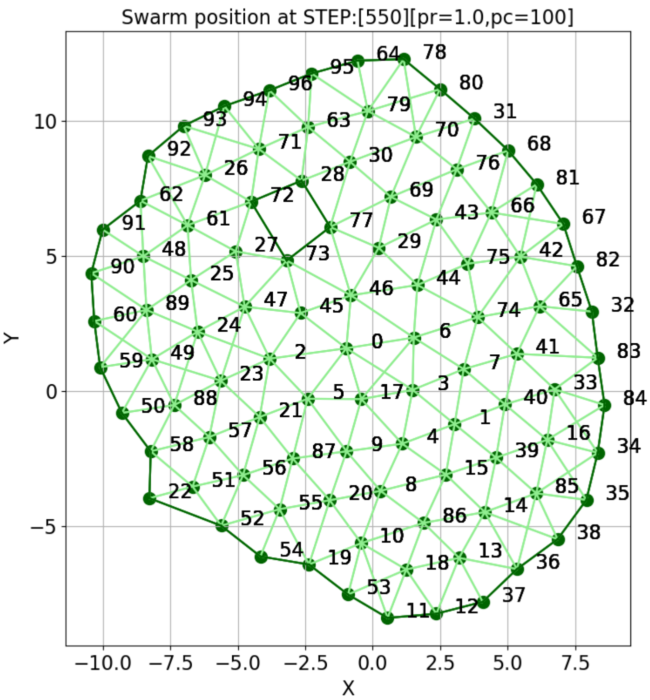


Fig. 25: Sample

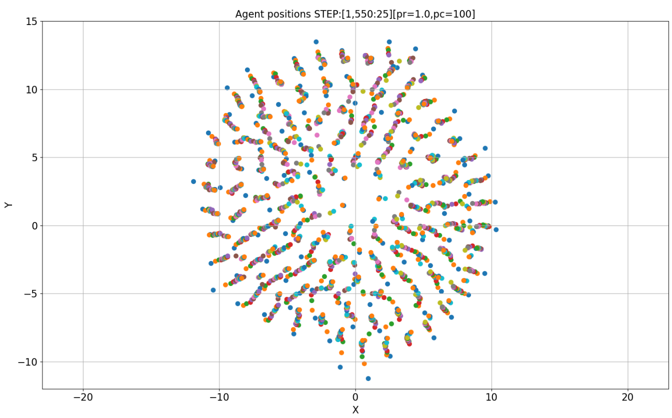


Fig. 26: Sample

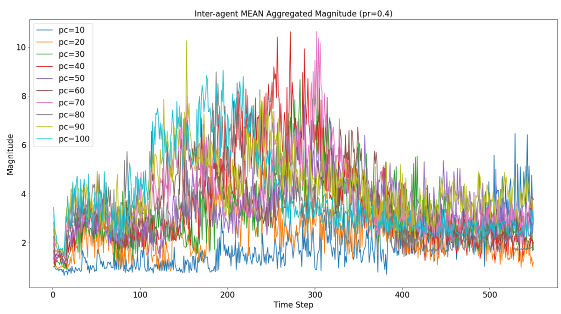


Fig. 27: Sample

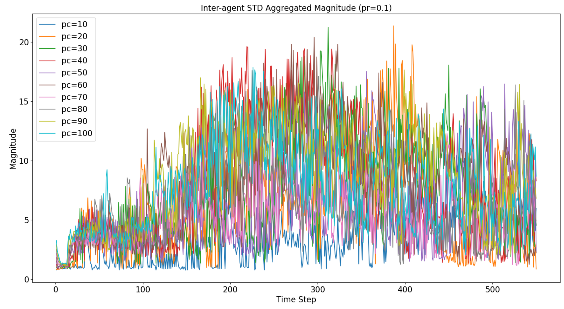


Fig. 28: Sample

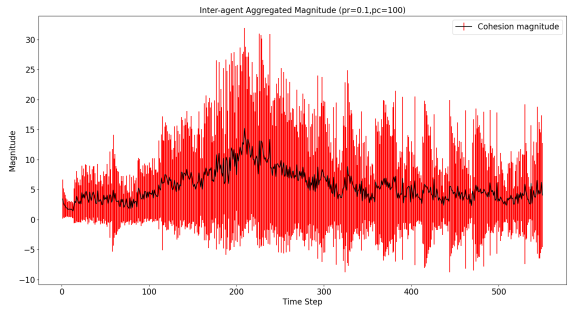


Fig. 29: Sample

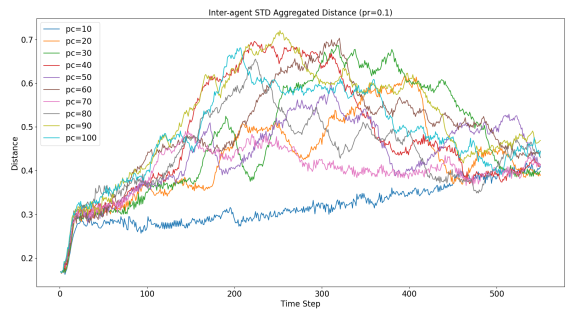


Fig. 30: Sample

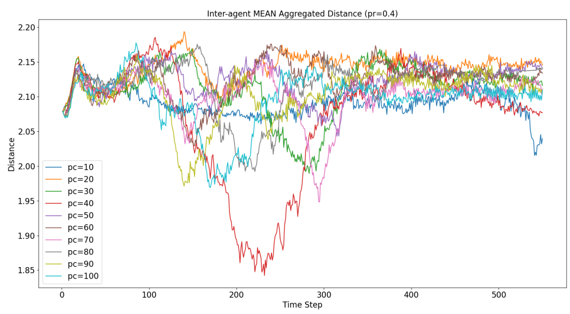


Fig. 31: Sample

Fig. 32: Sample

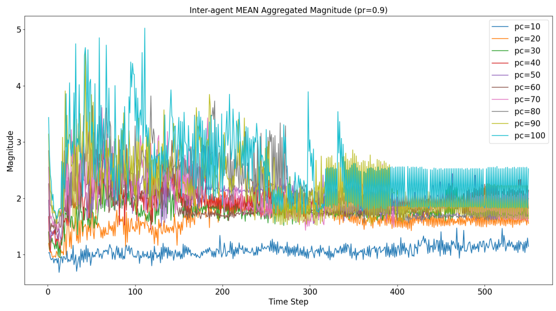


Fig. 33: MAG pr=0.9 mean

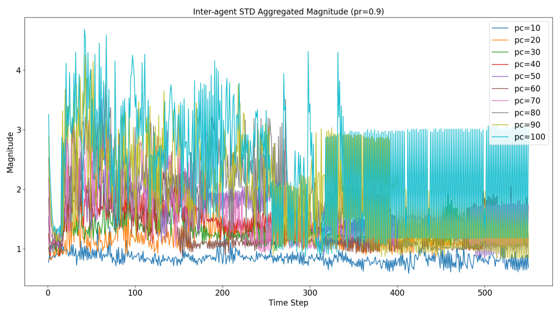


Fig. 34: MAG pr=0.9 std

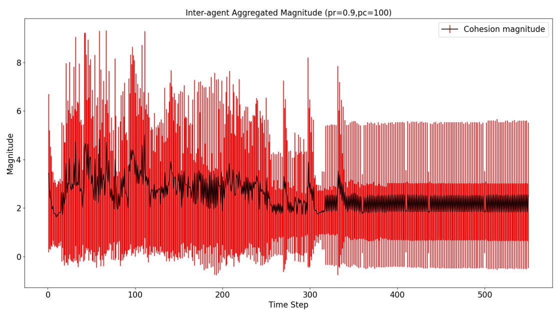


Fig. 35: MAG pr=0.9 pc=100 error

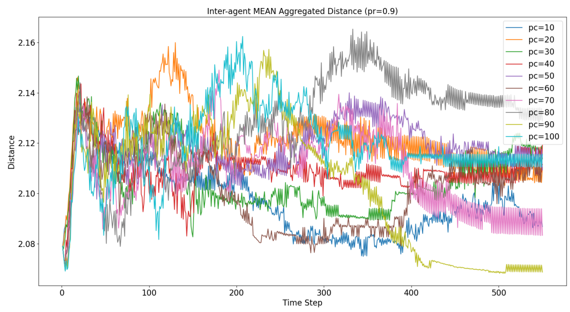


Fig. 36: DIST pr=0.9 mean

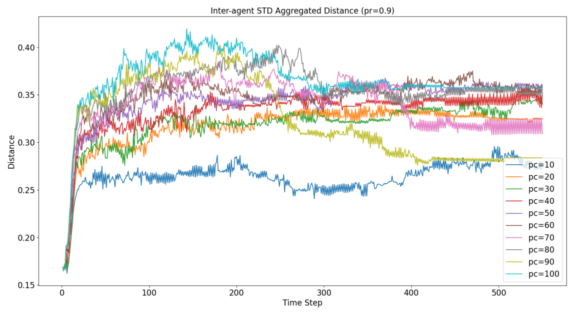


Fig. 37: DIST pr=0.9 std

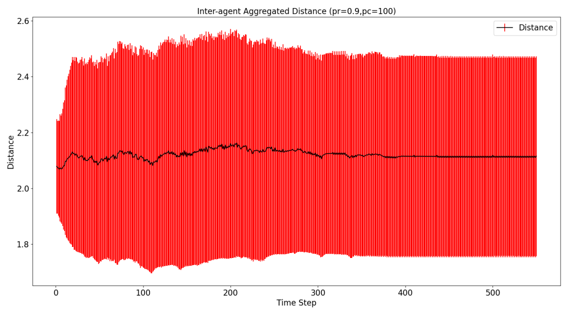


Fig. 38: DIST pr=0.9 pc=100 error

E. Comparison

VIII. Conclusions

From the initial simulations it is possible to show that the technique is able to successfully remove voids and surround an obstacle as shown in the video https://youtu.be/3eY1vvq0JWo.

IX. Future Work

References

[1] L. Barnes, W. Alvis, M. Fields, K. Valavanis, and W. Moreno. Heterogeneous swarm formation control using bivariate normal functions to generate potential fields. In *Distributed Intelligent Systems: Collective Intelligence and Its Applications, 2006. DIS 2006. IEEE Workshop on*, pages 85–94. IEEE, 2006.

[2] L. Barnes, W. Alvis, M. Fields, K. Valavanis, and W. Moreno. Swarm formation control with potential fields formed by bivariate normal functions. In *Control and Automation, 2006. MED'06. 14th Mediterranean Conference on*, pages 1–7. IEEE, 2006.

[3] L. Barnes, M. Fields, and K. Valavanis. Unmanned ground vehicle swarm formation control using potential fields. In *Control & Automation, 2007. MED'07. Mediterranean Conference on*, pages 1–8. IEEE, 2007.

[4] D. Bennet and C. McInnes. Verifiable control of a swarm of unmanned aerial vehicles. *Journal of Aerospace Engineering*, 223(7):939–953, 2009.

[5] Y. Cao, W. Ren, and M. Egerstedt. Distributed containment control with multiple stationary or dynamic leaders in fixed and switching directed networks. *Automatica*, 48(8):1586–1597, 2012.

[6] Y. Dai, M. Hinchey, M. Madhusoodan, J. Rash, and X. Zou. A prototype model for self-healing and self-reproduction in swarm robotics system. In *2006 2nd IEEE International Symposium on Dependable, Autonomic and Secure Computing*, pages 3–10, Sept 2006.

[7] K. Elamvazhuthi and S. Berman. Optimal control of stochastic coverage strategies for robotic swarms. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1822–1829. IEEE, 2015.

[8] N. Eliot. *Methods for the Efficient Deployment and Coordination of Swarm Robotic Systems*. University of Northumbria at Newcastle (United Kingdom), 2017.

[9] N. Eliot, D. Kendall, and M. Brockway. A new metric for the analysis of swarms using potential fields. *IEEE Access*, 6:63258–63267, 2018.

[10] N. Eliot, D. Kendall, A. Moon, M. Brockway, and M. Amos. Void reduction in self-healing swarms. In *Artificial Life Conference Proceedings*, pages 87–94. MIT Press, 2019.

[11] X. Fu, J. Pan, H. Wang, and X. Gao. A formation maintenance and reconstruction method of uav swarm based on distributed control. *Aerospace Science and Technology*, 104:105981, 2020.

[12] V. Gazi. Swarm aggregations using artificial potentials and sliding-mode control. *IEEE Transactions on Robotics*, 21(6):1208–1214, Dec 2005.

[13] R. Ghrist, D. Lipsky, S. Poduri, and G. Sukhatme. Surrounding nodes in coordinate-free networks. In *Algorithmic Foundation of Robotics VII*, pages 409–424. Springer, 2008.

[14] S. P. Hou and C. C. Cheah. Multiplicative potential energy function for swarm control. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 4363–4368, Oct 2009.

[15] A. R. Ismail and J. Timmis. Towards self-healing swarm robotic systems inspired by granuloma formation. In *Engineering of Complex Computer Systems (ICECCS), 2010 15th IEEE International Conference on*, pages 313–314. IEEE, 2010.

[16] G. Lee and N. Y. Chong. Self-configurable mobile robot swarms with hole repair capability. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 1403–1408, Sept 2008.

[17] X. Liang, X. Qu, N. Wang, Y. Li, and R. Zhang. Swarm control with collision avoidance for multiple underactuated surface vehicles. *Ocean Engineering*, 191:106516, 2019.

[18] J. McLurkin and E. D. Demaine. A distributed boundary detection algorithm for multi-robot systems. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 4791–4798. IEEE, 2009.

[19] T. Räz. On the application of the honeycomb conjecture to the bee's honeycomb. *Philosophia Mathematica*, page nkt022, 2013.

[20] C. W. Reynolds. Flocks, herds and schools: A distributed behavioral model. In *ACM SIGGRAPH computer graphics*, volume 21, pages 25–34. ACM, 1987.

[21] J. H. Roach, R. J. Marks, and B. B. Thompson. Recovery from sensor failure in an evolving multiobjective swarm. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(1):170–174, Jan 2015.

[22] F. E. Schneider and D. Wildermuth. A potential field based approach to multi robot formation navigation. In *Robotics, Intelligent Systems and Signal Processing, 2003. Proceedings. 2003 IEEE International Conference on*, volume 1, pages 680–685 vol.1, Oct 2003.

[23] J. Son, H. Ahn, and J. Cha. Lennard-jones potential field-based swarm systems for aggregation and obstacle avoidance. In *2017 17th International Conference on Control, Automation and Systems (ICCAS)*, pages 1068–1072, 2017.

[24] C. Speck and D. J. Bucci. Distributed uav swarm formation control via object-focused, multi-objective sarsa. In *2018 Annual American Control Conference (ACC)*, pages 6596–6601, 2018.

[25] J. Timmis, A. Ismail, J. Bjerknes, and A. Winfield. An immune-inspired swarm aggregation algorithm for self-healing swarm robotic systems. *Biosystems*, 146:60 – 76, 2016. Information Processing in Cells and Tissues.

[26] E. Vassev and M. Hinchey. Assl specification and code generation of self-healing behavior for nasa swarm-based systems. In *2009 Sixth IEEE Conference and Workshops on Engineering of Autonomic and Autonomous Systems*, pages 77–86, April 2009.