A Novel Relationship-based Approach to Swarm Control.

Neil Eliot[1,*], David Kendall , Michael Brockway , Paul Oman[1], and Ahmed Bouridane[1]

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

***Abstract*—Relationship-based coordination is a technique where by new emergent behaviours can be created to improve the structure of a swarm for a given application. The algorithm is dependant upon perimeter identification and is controlled by applying three weighting arrays to the existing swarming formulae. One to the cohesion calculation and two to the repulsion calculation. These simple changes allow for the likes of packed and expanded perimeters to emerge for a random swarm deployment.**

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 packing 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-A.

The aim of this new algorithm is to create a flexible relationship-based coordination technique that allows new emergent behaviours realised. Figure 1) shows an agent and it fields. S is the sensor field. O is the obstacle field. C is the cohesion field and R is the repulsion field. The implementation involves introducing three controlling arrays; k_c which can be used to increases the magnitude of the cohesion vector. k_r which can be used to modify the repulsion vector and R which can be used to alter the repulsion fields of agents.

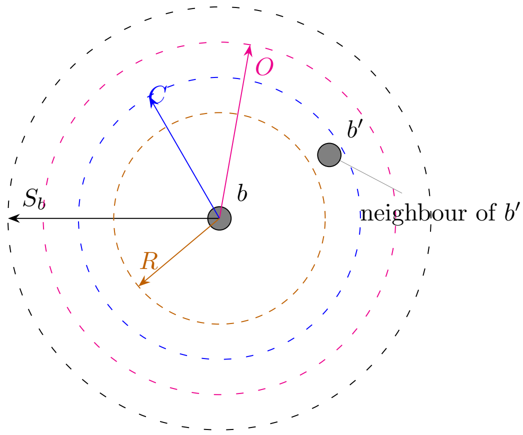

Fig. 1: Agent Fields

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, among other behaviours, a void reduction effect through perimeter packing. 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).

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

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 formula for agents to maintain their proximity is shown in Equation 1. Equation 3 shows the technique applied to accumulating the vectors that create the cohesive effect. $|n_c(b)|$ denotes the cardinality of $n_c(b)$. This is the component of the overall vector calculation that has the k_c quotient applied to it to allow 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).

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

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 ($\|b' - b\| - R$). Where $|n_r(b)|$ denotes the cardinality of $n_r(b)$. This vector is then scaled to 'balance' the effect with respect to other vector influences as shown is Equation 1 where k_c is applied.

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

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

IV. New Inter-agent Model

In this paper, we propose that the behaviour of an agent should be modified depending on whether or not it is on a *perimeter*. Figure 2 shows a simple swarm. Perimeter agents are highlighted in red. Perimeter-based agents can form part of an inner boundary or an outer boundary. The swarm can also contain non-perimeter agents which are shown in black.

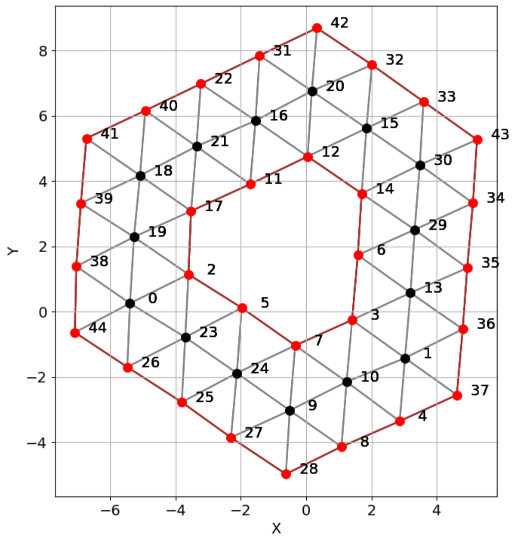

Fig. 2: Outer and inner swarm perimeters.

A. Perimeter detection

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].

When detecting a perimeter it is useful to define an ordering on an agent's cohesion neighbours. We choose to order the cohesion neighbours of an agent b by their *polar angle* (α) with respect to b (Fig. 3).

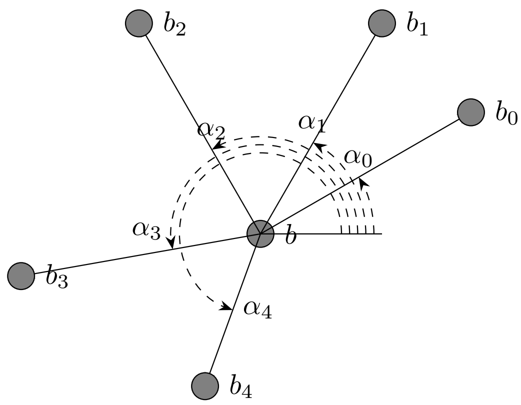

Fig. 3: Agent neighbours

The polar angle with respect to b of b', $\alpha(b, b')$, is the counter-clockwise angle that vector $\vec{bb'} = b' - b$ makes with the positive x axis shown in Figure 3 as α and described by Equation 6.

$$\alpha(b, b') = \mathsf{atan2}((b' - b)_y, (b' - b)_x) \tag{6}$$

A partial ordering of agents by polar angle with respect to a specific agent, b, is denoted $\leq_{\alpha_b}$, as defined in equation 7.

$$b' \leq_{\alpha_b} b'' \iff \alpha(b, b') \leq \alpha(b, b'') \tag{7}$$

We denote by $\langle b_0, b_1, .., b_{n-1}\rangle_{\leq_{\alpha_b}}$ a bijection from $\{0, .., n-1\} \rightarrow n_c(b)$ that is ordered by polar angle as shown in Figure 4 and more formally in Equation. 8.

$$\forall i, j \;:\; 0 \leq i, j, < n \cdot i \leq j \implies b_i \leq_{\alpha_b} b_j \tag{8}$$

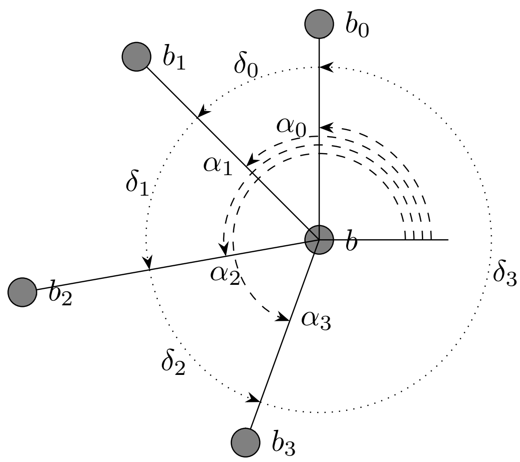

Fig. 4: Agent neighbour angles

An agent b is on a perimeter if it satisfies any one of three conditions:

1) consecutive neighbours are not within each other's cohesion field, or
2) consecutive neighbours subtend a reflex angle (as shown in Figure 4 as δ_3), or
3) the agent has too few neighbours.

A function, $\mathsf{prm}(b)$, specifies these conditions formally. Let b be the agent of interest and b', b'' any pair of consecutive neighbours of b in the angle-sorted list $\langle b_0, b_1, .., b_{n-1}\rangle_{\leq_{\alpha_b}}$, i.e. $b' = b_i, b'' = b_{(i+1)\%n}$ for some $i \in \{0, .., n-1\}$. Then $\mathsf{prm}(b)$ if any one of the following conditions is satisfied:

1) $b' \notin n_c(b'')$,
2) $\delta > \pi$, where $\delta = \alpha(b, b'') - \alpha(b, b')$ (or $\delta = \alpha(b, b'') - \alpha(b, b') + 2\pi$ if the former is negative), or
3) $|n_c(b)| < 3$.

B. R, k_r and k_c

In this section we will discuss the application of the new R, k_r and k_c arrays which are structured as a two dimensional array as shown below:

$$\begin{array}{cc} & \begin{array}{cc}\text{False} & \text{True}\end{array} \\ \begin{array}{c}\text{False} \\ \text{True}\end{array} & \begin{bmatrix} i-i & i-p \\ p-i & p-p \end{bmatrix} \end{array}$$

Where i represents an internal agent and p is a perimeter agent. If we consider Figure 2 then agents $18 \rightarrow 21$ would be internal to internal ($i-i$), $18 \rightarrow 39$ would be internal to perimeter ($i-p$), $39 \rightarrow 19$ would be perimeter to internal ($p-i$) and $41 \rightarrow 40$ would be perimeter to perimeter ($p-p$).

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 9 where the cohesion and repulsion arrays (k_c, k_r, R) are integrated into $v_c(b)$ and $v_r(b)$.

$$v(b) = v_c(b) + v_r(b) \tag{9}$$

1) Cohesion vector:

Cohesion neighbours are identified as described in Equation 2. The cohesion influence is then calculated as shown in Equation 10.

$$v_c(b) = \frac{1}{|n_c(b)|} \sum_{b' \in n_c(b)} k_c[p_b, p_{b'}](b' - b) \tag{10}$$

where $|n_c(b)|$ denotes the cardinality of $n_c(b)$, $p_b = \mathsf{prm}(b)$, $p_{b'} = \mathsf{prm}(b')$, and k_c is a 2x2 boolean-indexed array of constants that determine the weight of a component of the cohesion vector according to whether the interaction between b, b' is between non-perimeter agents, non-perimeter–perimeter, perimeter–non-perimeter, or perimeter–perimeter agents.

2) Repulsion vector:

The set of repellers of b are defined as Equation 11.

$$n_r(b) = \{b' \in \mathcal{S} : b \neq b' \wedge b' - b \leq R[p_b, p_{b'}]\} \tag{11}$$

where $p_b = \mathsf{prm}(b)$, $p_{b'} = \mathsf{prm}(b')$, and R is a 2x2 boolean-indexed array of constants that determine the radius of the *repulsion field* for agents in the swarm, according to whether the interaction between b, b' is between non-perimeter agents, non-perimeter–perimeter, perimeter–non-perimeter, or perimeter–perimeter agents.

Now $v_r(b)$ is defined by Equation 12

$$v_r(b) = \frac{1}{\|n_r(b)\|} \sum_{b' \in n_r(b)} k_r[p_b, p_{b'}] \left(1 - \frac{R[p_b, p_{b'}]}{b' - b}\right)(b' - b) \tag{12}$$

where $p_b = \mathsf{prm}(b)$, $p_{b'} = \mathsf{prm}(b')$, and k_r is a 2x2 boolean-indexed array of constants that determine the weight of a component of the repulsion vector according to whether the interaction between b, b' is between non-perimeter agents, non-perimeter–perimeter, perimeter–non-perimeter, or perimeter–perimeter agents.

C. Gap-filling

In addition to cohesion and repulsion vectors, a *gap-filling* vector can also be used to contribute to agent behaviour. Gap-filling vectors have proven useful in quickly reducing internal voids and in controlling the shape of the external perimeter.

A gap-filling vector for b contributes a motion of b towards the midpoint of a gap identified in the perimeter test for b.

Let $\langle b_0, b_1, .., b_{n-1} \rangle_{\leq \alpha_b}$ be the cohesion neighbours of b in polar angle order, and let $b' = b_i$ and $b'' = b_{(i+1)\%n}$ be the first pair of consecutive neighbours that satisfy either condition (1) or condition (2) of the perimeter function $\mathsf{prm}()$, then the gap-filling vector, $v_g(b)$, for agent b is defined in Equation 13.

$$v_g(b) = k_g \left(\frac{b' + b''}{2} - b\right) = k_g \frac{b' - b + b'' - b}{2} \tag{13}$$

If there is no such pair of consecutive neighbours then $v_g(b) = 0$.

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".

A stricter alternative to this is to choose the first consecutive neighbour pair b', b'' that satisfy condition (1), ignoring condition (2). This would then exclude any reflex angles that create a 'gap'.Again, $v_g(b)$ is defined by eq (13) if such a pair exists, or 0 otherwise.

D. Resultant vector

The resultant vector is simply the sum of the cohesion, repulsion and gap-filling vectors as shown in Equation 14 and a resulant swarm segment is shown in Figure 5

$$v(b) = v_c(b) + v_r(b) + v_g(b) \tag{14}$$

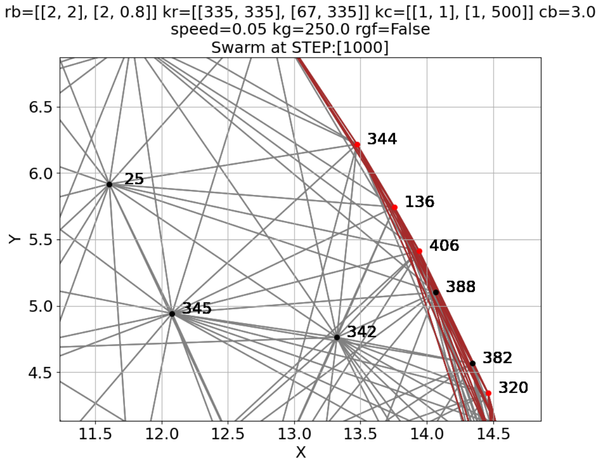

Fig. 5: Swarm Example.

E. Relationship-based swarm effects

The introduction of the arrays allows for specific relationships to effect the movement of agents. Using uniform arrays results a simple cohesion/repulsion based swarm with all agents exhibiting the same properties similar to the original model discussed in § III. However, modifying the arrays for specific relationships can induce emergent behaviours such as perimeter packing as discussed in § V-D.

1) Cohesion model:
When using Equation 10 one array is used, k_c. This array is used to scale the cohesion vector generated between an agent pair which is proportional to their distance apart, which will be within C as shown in Equation 4. Consider the array shown in Equation 15.

$$k_c = \begin{bmatrix} 1 & 1 \\ 1 & 500 \end{bmatrix} \tag{15}$$

For a given agent pair their perimeter status will be calculated and applied to the arrays. If both agents are perimeter based then the value selected would be $k_c[P_b, P_{b'}] \Rightarrow 500$. If the agent pair were perimeter $\rightarrow$ non-perimeter then the value selected would be $k_c[P_b, P_{b'}] \Rightarrow 1$. This configuration would cause inter-perimeter agents to tend to move towards each other more strongly than any other relationship.

2) Repulsion model:
When using Equation 12 two arrays are used k_r and R. k_r is used to scale the resultant repulsion vector that is generated. R is the radius of the repulsion field and is used to generate the proportion of the repulsion vector that is applied. Therefore consider the following two arrays (Eqs 16 and 17):

$$R = \begin{bmatrix} 2 & 2 \\ 2 & 0.8 \end{bmatrix} \tag{16}$$

$$k_r = \begin{bmatrix} 335 & 335 \\ 67 & 335 \end{bmatrix} \tag{17}$$

For a given agent pair their perimeter status will be calculated and applied to the arrays. If both agents are perimeter based then the values selected would be $R[P_b, P_{b'}] \Rightarrow 0.8$ and $k_r[P_b, P_{b'}] \Rightarrow 335$. If the agent pair were perimeter $\rightarrow$ non-perimeter then the values selected would be $R[P_b, P_{b'}] \Rightarrow 2$ and $k_r[P_b, P_{b'}] \Rightarrow 67$.

V. Experimental results

The new modelling method allows for a highly configurable swarm. Each configuration will have an impact on the the swarm's structure changes. This can be analysed in several different ways. This includes identifying the number of perimeter agents and internal agents, the magnitude of the interactions between agents and the distances between the agents. The experimental results show the effects on the swarm in terms of inter-agent distances, magnitude metric [9], and the effect on the perimeter.

A. Distance metric

The distance metric is used by many researchers as a method of examining the structure of a swarm [1], [2], [7], [12], [22]. However, due to the new model allowing the field effects and magnitudes to be varied the distance metric will need to be adapted to analyse the agents involved in specific relationships rather than globally, therefore S will be sub-divided into the three relationship categories of S_i, S_p, S_o. Where S_i are the internal-internal relationships, S_p are the perimeter-perimeter relationships and S_o are all the internal-perimeter or perimeter-internal relationships. The distance metric is based upon the mean of a set of agents distances from its neighbours and the standard deviation between those agent sets. The mean is calculated as shown in equation 18 where $\mu_d(S)$ is the mean. The standard deviation is calculated as shown in equation 19 where $\sigma_d(S)$ is the standard deviation. The mean distance value can be compared to the repulsion field to identify if a swarm is optimally distributed in that the mean value should be as close to the repulsion filed as possible. The standard deviation identifies the overall differences in the distances which can be caused by the swarm agents oscillating. A standard deviation of $\sigma_d(S) = 0$ would indicate that all the agents are equally spaced.

$$\mu_d(S) = \frac{\sum_{b \in S} \sum_{b' \in nbr(b)} |b' - b|}{\sum_{b \in S} |nbr(b)|} \tag{18}$$

$$\sigma_d(S) = \sqrt{\frac{\sum_{b \in S} \sum_{b' \in nbr(b)} \Big(|b' - b| - \mu_d(S)\Big)^2}{\sum_{b \in S} |nbr(b)|}} \tag{19}$$

Therefore the distance metric for the distribution of a set of agents is both $\mu_d(S)$ and $\sigma_d(S)$. This can be written informally as:

$$\psi_d(S) = \mu_d(S) \pm \sigma_d(S) \tag{20}$$

An example is shown in Figure 11.

B. Magnitude metric

The magnitude metric as defined by Eliot et al. [9] is based upon the relationship between agents and as such is independent of the resultant structure in terms of distances. i.e. agents can be different distances from each other but have the same relationship magnitude. The metric is based upon the mean of a set of agents relationships and the standard deviation between those relationships. The mean is calculated as shown in equation 21 where $\mu_p(S)$ is the mean. The standard deviation is calculated as shown in equation 22 where $\sigma_p(S)$ is the standard deviation. Due to the metric being based on inter-agent relationships the swarm can be analysed as a whole.

$$\mu_p(S) = \frac{\sum_{b \in S} P(b)}{\sum_{b \in S} |nbr(b)| + \sum_{b \in S} |n_r(b)|} \tag{21}$$

Std:

$$\sigma_p(S) = \sqrt{\frac{\sum_{b \in S} \Big(P(b) - \mu_p(S)\Big)^2}{\sum_{b \in S} |nbr(b)| + \sum_{b \in S} |n_r(b)|}} \tag{22}$$

The metric for the internal movement is the mean and standard deviation of the swarm's internal *cohesion/repulsion*. The pair $\mu_p(S)$, $\sigma_p(S)$ may therefore be written informally as:

$$\psi_p(S) = \mu_p(S) \pm \sigma_p(S) \tag{23}$$

C. Baseline

For all the experiments the parameters used to create the basic swarming effect are shown in Table I. Where C is the cohesion field, k_c is the cohesion weighting, R is the repulsion field, k_r is the repulsion weighting and k_g is the weighting applied in the gap reduction algorithm discussed in [10]. The swarm consists of 500 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 6. This basic swarm is used as the initial state for all the experiments.

Swarming Variable	Value
C	3.0
k_c	[[1.0,1.0][1.0,1.0]]
R	[[2.0,2.0][2.0,2.0]]
k_r	[[335,335][335,335]]
k_g	0.0

TABLE I: Swarming effect parameters

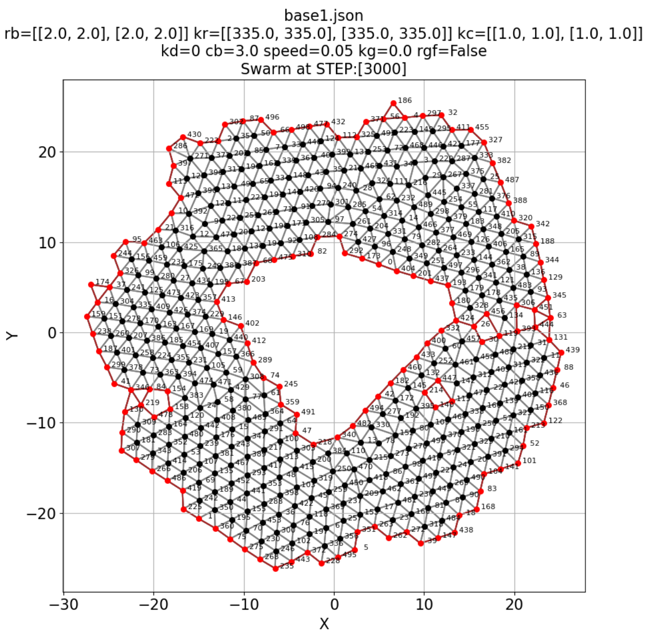

Fig. 6: Baseline swarm.

When the simulation is ran with no relationship differences i.e. all array values are equal, the changes are identified using a magnitude-based metric [9]. The resultant magnitudes generated are shown in figure 7. These states are used as the baseline for the experiments to measure the effects of changing the arrays.

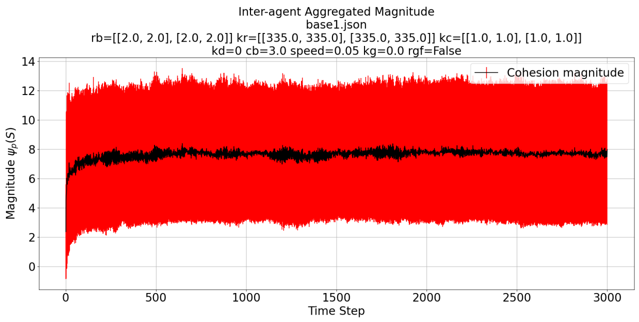

Fig. 7: Baseline swarm (Magnitude)).

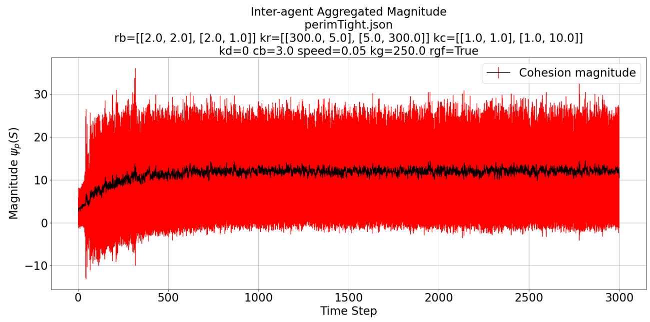

Fig. 10: Packed Perimeter (Magnitude).

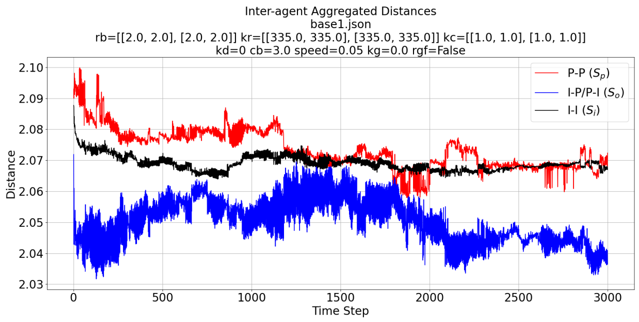

Fig. 8: Baseline swarm (Distance).

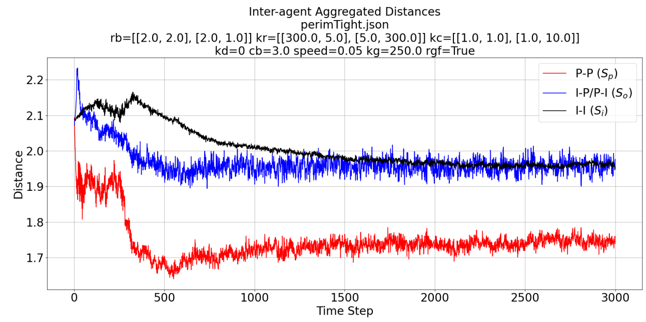

Fig. 11: Packed Perimeter (Distance).

D. Perimeter Packing

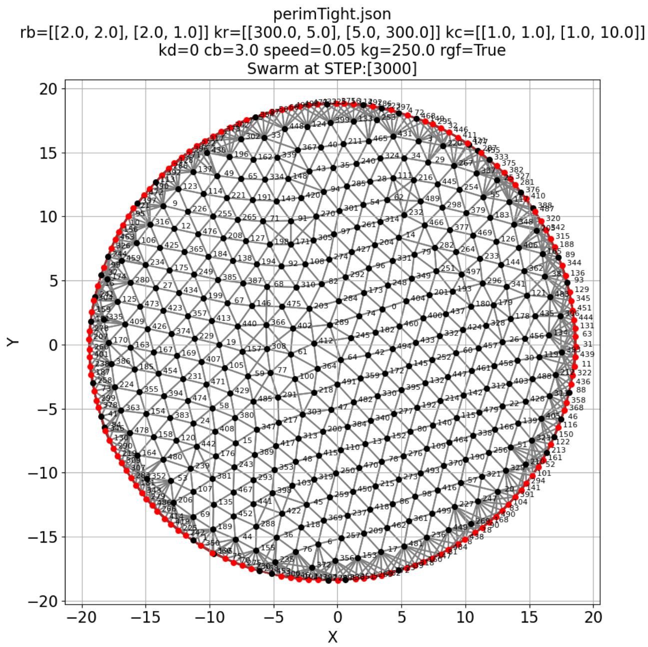

Fig. 9: Packed Perimeter.

E. Baseline 2

The swarming parameters are the same as baseline 1 as shown in table I.

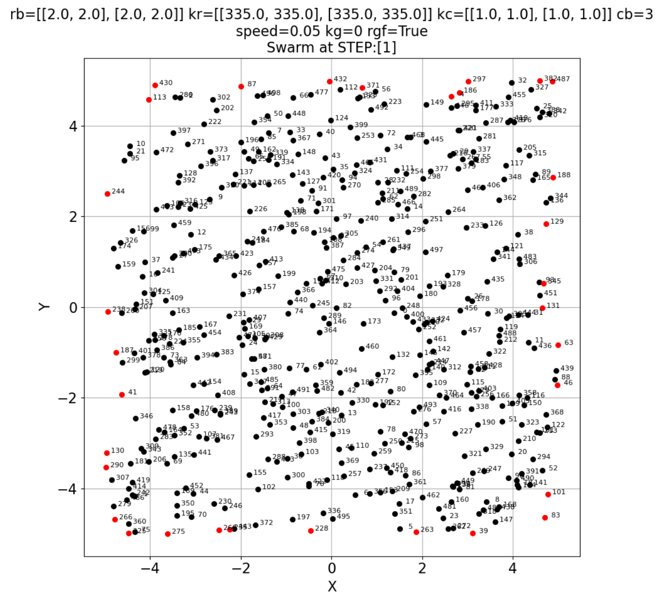

Fig. 12: Baseline 2 start.

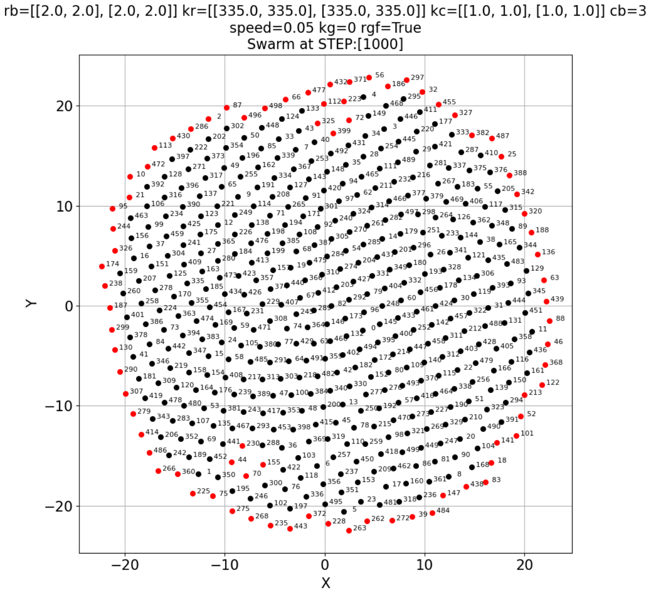

Fig. 13: Baseline 2 end.

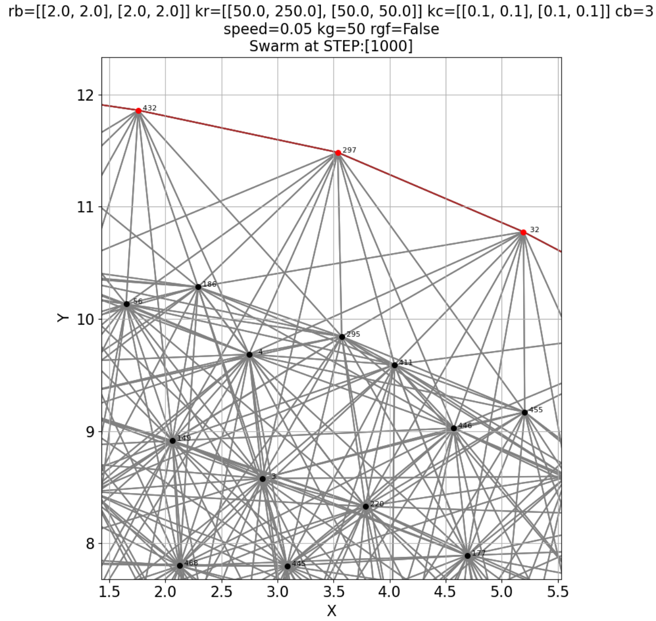

Fig. 16: Perimeter Expanded 2.

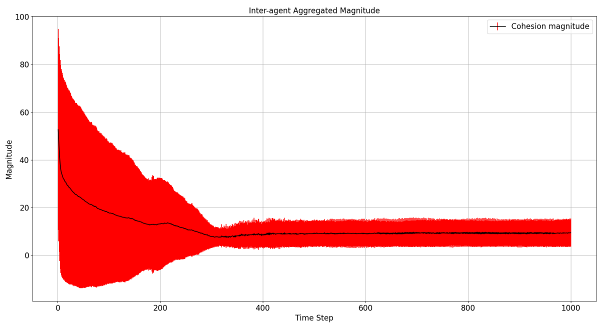

Fig. 14: Baseline 2 (Magnitude).

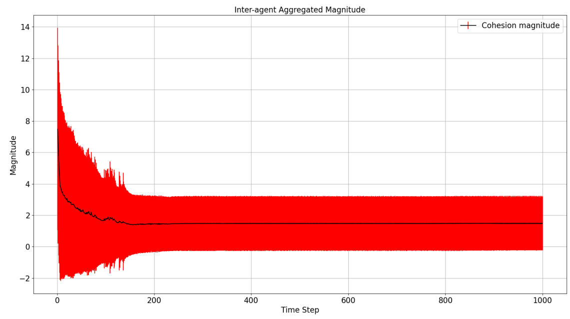

Fig. 17: Perimeter Expanded (Magnitude).

F. Perimeter Expansion

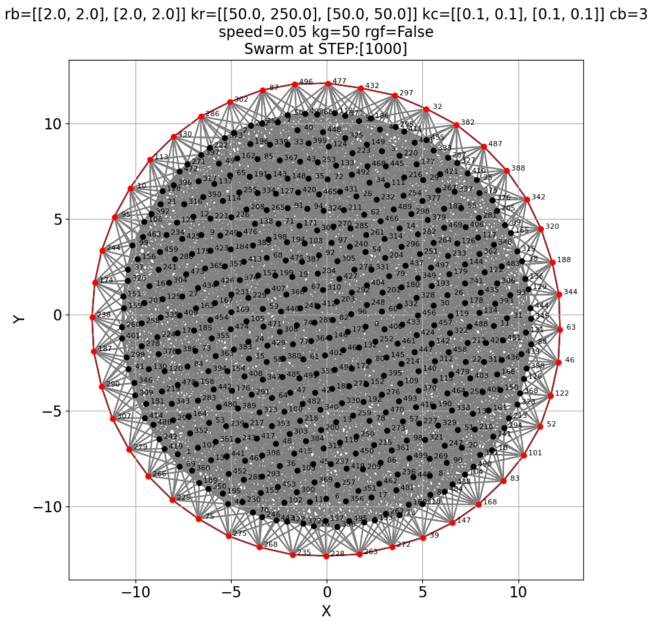

Fig. 15: Perimeter Expanded 1.

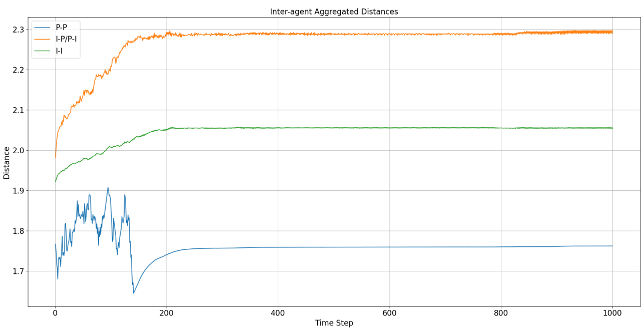

Fig. 18: Perimeter Expanded (Distance).

==== STILL TO BE WORKED ON FROM HERE =====

G. Inner packing model

$$p_{kr} = \begin{pmatrix} 1 & x \\ 1 & 1 \end{pmatrix} \tag{24}$$

$$p_{kc} = \begin{pmatrix} 1 & y \\ 1 & 1 \end{pmatrix} \tag{25}$$

1) Inner + Gap reduction model:

H. Outer packing model

$$p_{kr} = \begin{pmatrix} 1 & x \\ x & 1 \end{pmatrix} \tag{26}$$

$$p_{kc} = \begin{pmatrix} 1 & y \\ y & 1 \end{pmatrix} \tag{27}$$

1) Outer + Gap reduction model:

I. Compression Effects

J. Comparison

VI. 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.

VII. 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.