Perimeter Compression in self-healing swarms

Brockway M, Eliot N, Kendal D,
Hexham University
Cronkley Campus
Kiln Pit Hill
Department of Computer Science

February 4, 2021

Abstract

Perimeter Compression is a technique where by a void reducing effect can be added to a basic swarming algorithm. The affect is dependant upon perimeter identification and is controlled by applying two weighting factors to the existing swarming formulae. One to the cohesion calculation and the other to the repulsion calculation.

1 Introduction

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 [1, 2, 3].

The aim of the algorithm is to reduce the spacing between perimeter-based agents by reducing the repulsion field and increasing the cohesion affect on perimeter agents. Figure 1) shows an agent and it feilds. 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 two controlling weights; p_c (Perimeter Compression Cohesion) which increases the cohesion vector ($k_c \rightarrow p_c k_c$) and p_r (Perimeter Compression Repulsion) which reduces the size of the repulsion field ($k_r \rightarrow p_r k_r$) of the perimeter-based agents.

Assumption 1 $p_c >= 1$

Assumption 2 $p_r <= 1$

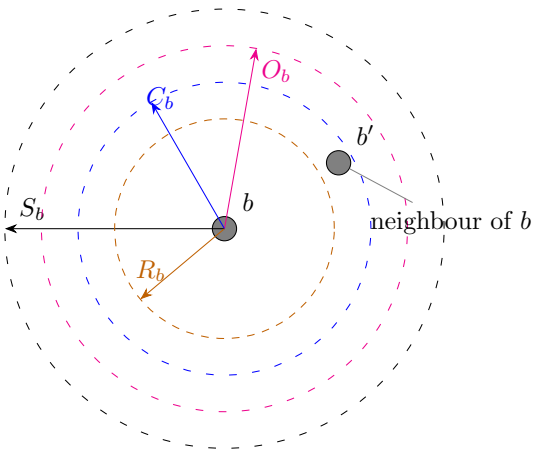

Figure 1: Agent Fields

2 Resultant Vector Calculation

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. The new algorithm requires each individual agent to have a variation to the vector generated inside each calculation based on the perimeter status of the agent and each neighbour. The equation has been simplified (Eq. 2) and the weighting factors have been transfered into the calculations along with the additional weighting factors that are applied to specific agents within the cohesion and repulsion vector calculations.

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

$$v(b) = v_c(b) + v_r(b) + v_d(b) + v_o(b) \tag{2}$$

The effect of these weighting factors can be seen in Figure 2. The metric used in producing the graph is based upon the inter-agent magnitudes [2].

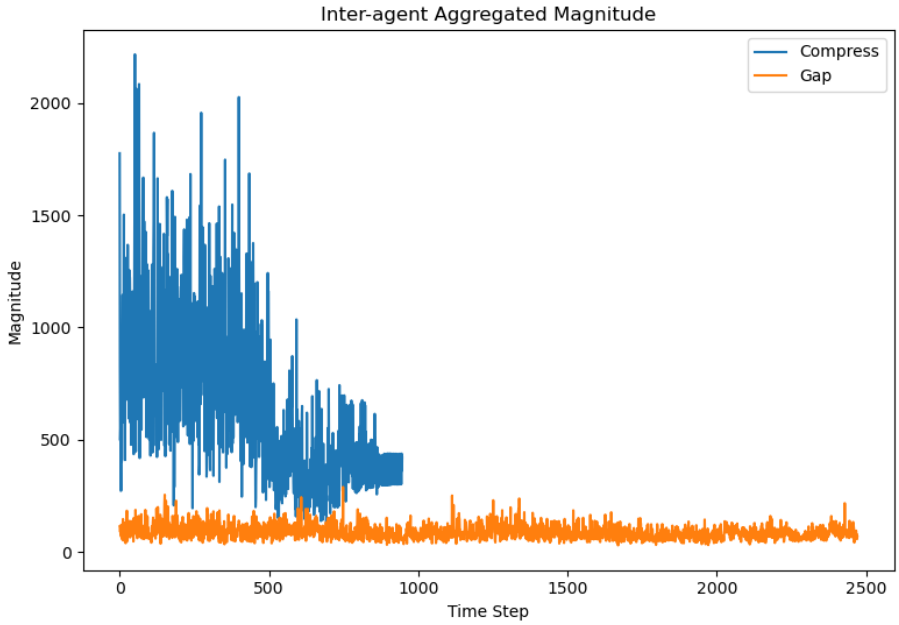

Figure 2: Comparison of magnitude change using gap reduction and perimeter compression

3 Repulsion

3.1 Repulsion neighbours

The repulsion component of an agent's movement is calculated from its interaction with its neighbours $n_r(b)$ (Eq. 3) that are within the agent's (b) repulsion field (R_b) or the adjusted (p_r) range $(\mathsf{erf}(b, b'))$ if the agents are both perimeter-based. Equation 4 shows the calculation of the effective field. As the agent's repulsion field is always within the cohesion field (Eq. 8), the repulsion neighbours can also be defined as a subset of the cohesion neighbours $n_c(b)$ (Eq. 5).

$$n_r(b) = \{b' \in \mathcal{S} : b \neq b' \wedge \|\vec{bb'}\| \leq \mathsf{erf}(b, b')\} \tag{3}$$

$$\mathsf{erf}(b, b') = \text{if } \mathsf{per}(b) \text{ and } \mathsf{per}(b') \text{ then } p_r R_b \text{ else } R_b \tag{4}$$

$$n_r(b) = \{b' \in n_c(b) \ : \ \|bb'\| <= \mathsf{erf}(b, b')\} \tag{5}$$

3.2 Repulsion compression

To reduce the repulsion effect the compression weighting (p_r) is applied to an agent's repulsion field if both itself and it's neighbour are perimeter agents. The function $\mathsf{per}()$ returns an agent's perimeter status. An agent is identified as a perimeter agent using the technique shown by Eliot et.al. in [3] which uses a cyclic-check of neighbour agent angles to identify "gaps" in the neighbours.

If the condition of both agents being a perimeter is met ($\mathsf{per}(b)$ and $\mathsf{per}(b')$) the repulsion field distance is multiplied by the compression factor (p_r) and the new field effect is used to generate a resultant-repulsion-vector (Eq. 7).

The effect of Equation 6 will be to reduce the repulsion of inter-perimeter-based agents allowing them to be closer together before a reduced repulsion-vector is applied.

Important: The repulsion-vector that is generated is based upon $p_r R_b$, the reduced repulsion field, and not the full R_b field. This is to scale the resultant-repulsion-vector as well as reducing the repulsion field.

$$v_r(b) = \frac{1}{|n_r(b)|} \sum_{b' \in n_r(b)} k_r \left(\|\vec{bb'}\| - \mathsf{erf}(b, b') \right) \widehat{bb'} \tag{6}$$

$$\mathsf{erf}(b, b') = \mathsf{if}\ \mathsf{per}(b)\ \mathsf{and}\ \mathsf{per}(b')\ \mathsf{then}\ p_r R_b\ \mathsf{else}\ R_b \tag{7}$$

4 Cohesion

4.1 Cohesion neighbours

The cohesion component of an agent is calculated in a similar way to the repulsion in that it is dependent upon the proximity of neighbours. Where $n_c(b)$ is the set of neighbour agents for b (Eq. 8). The inclusion of an agent from a swarm (S) in by the agent's cohesion field (C_b).

$$n_c(b) = \{b' \in S \ : \ b' \neq b \wedge \|bb'\| <= C_b\} \tag{8}$$

The affect 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 9. This formula includes the k_c quotient that allows the cohesion effect to be 'balanced' with respect to other vector influences as described in [1, 2, 3]

$$v_c(b) = \frac{1}{|n_c(b)|} \sum_{b' \in n_c(b)} k_c \vec{bb'} \tag{9}$$

4.2 Compression cohesion

The cohesion component of the compression effect (p_c) is applied when an agent (b) and its neighbour (b') are both perimeter-based. If the agents are not both perimeter-based then the agents vector is only scaled by k_c (Eq. 11). The effect of the additional cohesion-compression weighting is to increases the size of the generated cohesion-vector $\mathsf{efc}(b, b')$ (Eq. 10).

$$v_c(b) = \frac{1}{|n_c(b)|} \sum_{b' \in n_c(b)} \mathsf{ekc}(b, b')\, \vec{bb'} \tag{10}$$

$$\mathsf{ekc}(b, b') = \mathsf{if}\ \mathsf{per}(b)\ \mathsf{and}\ \mathsf{per}(b')\ \mathsf{then}\ \mathrm{p}_c k_c\ \mathsf{else}\ k_c \tag{11}$$

5 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.

References

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

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

[3] 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.