Perimeter Compression in self-healing swarms

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

January 27, 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 factors to the existing swarming formula. 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 the Alife Paper [3].

The aim of the algorithm is to reduce the spacing between perimeter-based agents by reducing the repulsion field (Figure. 1) and increasing the cohesion affect on perimeter agents. S_b is the sensor field. O_b is the obstacle field. C_b is the cohesion field. R_b is the repulsion field. The implementation involves introducing two controlling factors; k_{cpc} (Cohesion Perimeter Compression) which increases the cohesion vector ($C_b \rightarrow k_{cpc}C_b$) and k_{rpc} (Repulsion Perimeter Compression) which reduces the size of the repulsion field ($R_b \rightarrow k_{cpc}R_b$) on the inter-perimeter agents.

Assumption 1 $k_{cpc} >= 1$

Assumption 2 $k_{rpc} <= 1$

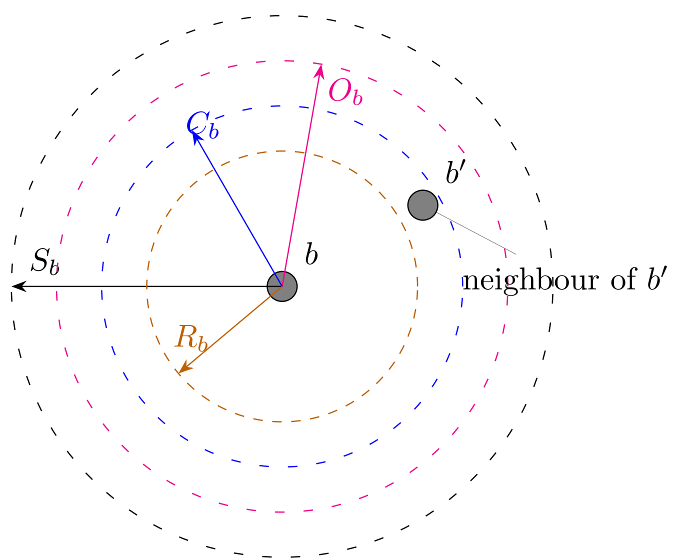

Figure 1: Agent Fields

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 transposed into the calculations along with additional weighting factors that are applied to specific agents within the cohesion and repulsion vector calculations (k_{cpc} - cohesion compression, k_{rpc} - repulsion compression).

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

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

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].

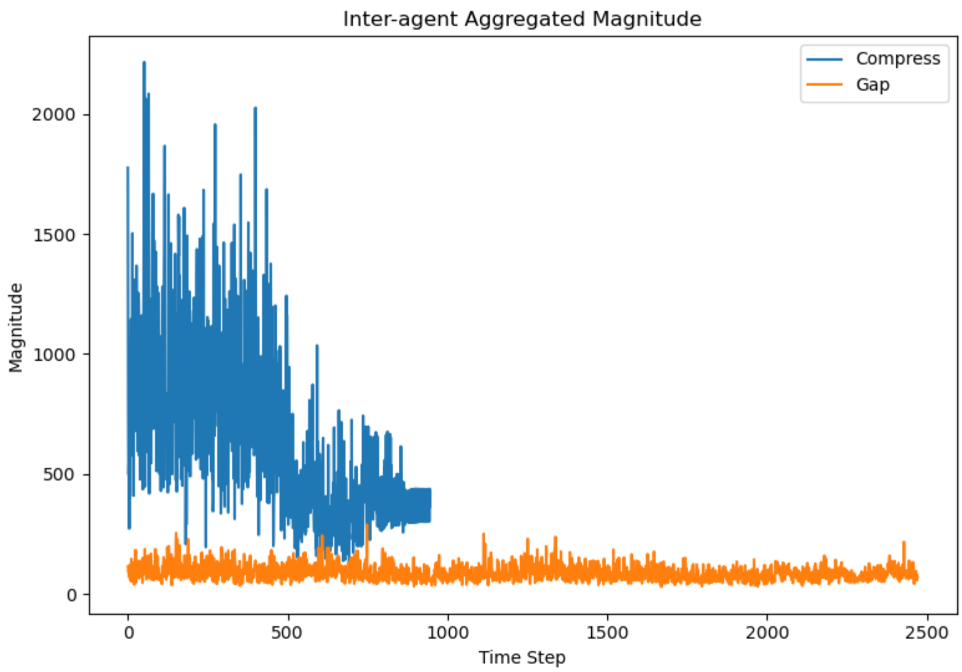

Figure 2: Compression Effect based on Magnitude change

3 Repulsion

The repulsion component of an agent's movement is calculated from its interaction with its neighbours $n_r(b)$ that are within the agent's (b) repulsion field (R_b) (Eq. 3) or from any agent in the swarm (S) that are within the agent's (b) repulsion field (R_b) (Eq. 4). The resultant set will be the same.

$$n_r(b) = \{b' \in S \ : \ b' \neq b \wedge \|bb'\| <= R_b\} \tag{3}$$

Given that the repulsion field is within the cohesion field (neighbours) then the repulsion set can also be identified as a subset of those neighbours where $n_r(b)$ (Eq. 9) is the set of all of the neighbours of b [1]:

$$n_r(b) = \{b' \in n_c(b) \ : \ \|bb'\| <= R_b\} \tag{4}$$

3.1 Compression repulsion

To reduce the repulsion effect, the control factor (k_{rpc}) is applied to an agent's repulsion field if both itself and it's neighbour are perimeter agents. Where 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]

Therefore if the condition of both agents being a perimeter is met ($\mathsf{per}(b) \wedge \mathsf{per}(b')$), where per() returns true if an agent is on the perimeter and or false if it is not, and b' is within repulsion field distance multiplied by the compression factor, or both agents are not both perimeter agents ($\neg(\mathsf{per}(b) \wedge \mathsf{per}(b'))$) and b' is within the 'normal' repulsion field (R_b) then an agent is part of the repulsion set (Eq. 5 or 6). Equation 6 being the least expensive.

$$\begin{aligned} n_r(b) = \{&b' \in S \ : \\ &((b' \neq b) \wedge \mathsf{per}(b) \wedge \mathsf{per}(b') \wedge \|bb'\| <= k_{rpc}R_b) \\ &\vee \\ &((b' \neq b) \wedge \neg(\mathsf{per}(b) \wedge \mathsf{per}(b')) \wedge \|bb'\| <= R_b) \\ &\} \end{aligned} \tag{5}$$

$$\begin{aligned} n_r(b) = \{&b' \in n_r(b) \ : \\ &(\mathsf{per}(b) \wedge \mathsf{per}(b') \wedge \|bb'\| <= k_{rpc}R_b) \\ &\vee \\ &(\neg(\mathsf{per}(b) \wedge \mathsf{per}(b')) \wedge \|bb'\| <= R_b) \\ &\} \end{aligned} \tag{6}$$

The effect of Equations 5 and 6 is that the perimeter-based agents will be closer together before a repulsion vector is generated.

Important: The repulsion vector that is generated is based upon R_b, the full repulsion field, and not the reduced field. This is to reduce potential agent collisions.

The final calculation for the repulsion having generated the set $n_r(b)$ is shown in Equation 7.

$$v_r(b) = \frac{1}{|n_r(b)|} \sum_{b' \in n_r(b)} \left(\mathsf{ekr}(b, b') \left(\|\vec{bb'}\| - R_b \right) \widehat{bb'} \right) \tag{7}$$

$$\mathsf{ekr}(b, b') = \mathsf{if}\ \mathsf{per}(b)\ \mathsf{and}\ \mathsf{per}(b')\ \mathsf{then}\ k_{cpc}k_r\ \mathsf{else}\ k_r \tag{8}$$

==
ISSUE
==
The methodology applied is to remove agents from the repulsion set if they are on the perimeter until they are at the altered range (Eq. 5,6). As such erf() isn't required as the calculation should still use the R_b field to generate a 'suitable' repulsion vector which will prevent a collision.
or
should erf() be left in and discussed as an alternative approach allowing greater flexibility in the application of the compression algorithm?
==

What do you think?
==

4 Cohesion

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 $n_c(b)$ (Eq. 9). The determining factor for an agent being affected is determined using the cohesion field (C_b).

$$n_c(b) = \{b' \in S \ : \ b' \neq b \wedge \|bb'\| <= C_b\} \tag{9}$$

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 formula for agents to maintain their proximity is to direct their motion towards the central point of all neighbouring agents as shown in Equation 10. This formula includes the k_c quotient that allows the cohesion effect to be 'balanced' with respect to other vector influences as described in Section 2

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

4.1 Compression cohesion

The compression effect on the cohesion of the perimeter agents is that any perimeter-perimeter relationships should be re-enforced using the cohesion compression quotient (k_{cpc}). This is achieved by identifying the relationship within the cohesion calculation.

$$v_c(b) = \frac{1}{|n_c(b)|} \sum_{b' \in n_c(b)} \mathsf{ekc}(b, b') \, \vec{bb'} \tag{11}$$

$$\mathsf{ekc}(b, b') = \mathsf{if} \ \mathsf{per}(b) \ \mathsf{and} \ \mathsf{per}(b') \ \mathsf{then} \ k_{cpc} k_c \ \mathsf{else} \ k_c \tag{12}$$

==
PROBLEM (WORKING ON IT NOW)
==
Are you happy with the notation to apply k_{cpc}
==

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.