Report for Vehicle Control System Design

Ruikai Cui u6919043

2019/9/27

Abstract

This article discusses a robust method to coordinate vehicles in 3-d space and shrink the swarm size to a specific number using a distributed decision-making strategy. The system structure and main ideas are analyzed in this article. The capabilities and limitations are experimentally determined and discussed.

1. Motivation

The background is that multiple vehicles glide in space and consume their dedicated battery to keep alive. Vehicles replenish their charge to full by passing "emery globes" in close proximity [1]. My design goal is keeping as many vehicles alive as possible for as long as possible (in both Stage B and C) and scheduling certain vehicles to run out of energy and vanish to shrink the swarm to a specific size (for Stage D). To steer to globes and charge, vehicles need to know the position of energy globes by either detection or *communication*. As only vehicles that are close enough to globes can detect them, position information needs to be shared using communication. Another problem introduced by charging is collision. Destinations might become unreachable when multiple vehicles are bound for the same destination [2], which requires a *coordination* strategy to be employed.

2. Design & Implementation

2.1. Overall Structure

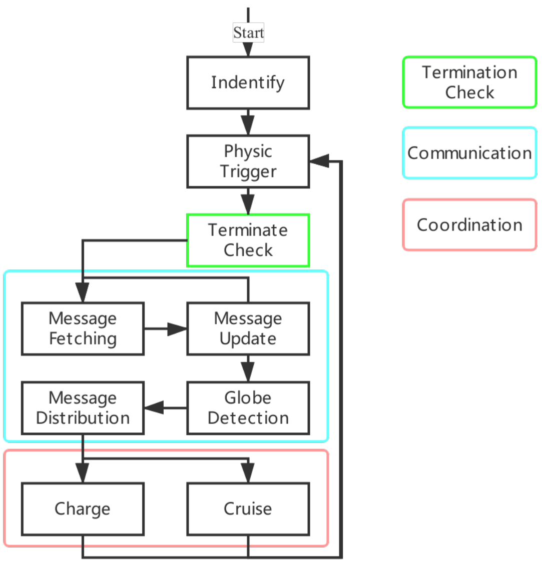


Figure 1: System Structure

The diagram in Figure 1 gives the system structure and the implementation outcome is illustrated in Figure 2. The goal is achieved by the combination of three components named *Termination Check*, *Communication* and *Coordination*.

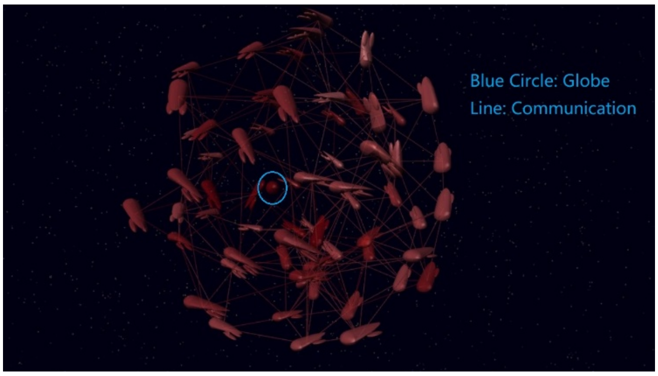


Figure 2: Implementation Outcome

2.2. Communication

As only vehicles near globes can use detection, communication is the main method for vehicles to acquire globe positions. Thus, the goal of the communication system is to efficiently share globe locations. The on-board asynchronous message passing system allows vehicles to broadcast messages to neighbours, which makes up the basis of communication. Other ingredients are information carried by messages and how to use received messages.

The globe position is an element contained by messages, and another element is a timestamp for message updating on individual vehicles. By these two elements and the message passing system, globe positions can be shared.

After receiving messages, individual vehicles need to figure out which message contains the most possible globe position. This is implemented by storing the latest one according to comparing timestamps made when vehicles find globes. The reason is that during a specific time span, the globe will move for a distance, and the shorter the time span is the smaller the distance will be. Therefore, timestamps indicate the most possible globe position. After acquiring the position and checking whether it has expired, vehicles will pass it forward to inform as many vehicles as possible.

2.3. Coordination

Coordination strategy is necessary because a position only allows one vehicle to reach at a moment, which means vehicles can only get charged sequentially. If multiple vehicles intend to charge simultaneously, then some need to wait for others to finish first. So, the goal of coordination is to avoid collision during charging.

The strategy is allowing a vehicle to charge when its energy is below a threshold, otherwise driving it to a point where it will not influence others to charge. The key is how to decide this point.

Two factors need to be considered when deciding the point. The first one is collision avoidance, which requires vehicles to fly as far as possible from globes to make room for vehicles that need to charge. The second factor is that the distance between two vehicles cannot be too large otherwise message passing will become unresponsive which may lead to collapse. Basing on factors above, coordinating vehicles on the surface of a sphere (globe as centre) would be a solution because it allows vehicles to communicate when cruising on the surface and make room in the middle for charging. This is implemented by setting destination (p') basing on current position(p) using formula (1),

$$p' = p_o + rR_2(\theta)R_1(\theta)R_3(\theta)\left(\frac{p_o - p}{||p_o - p||}\right) \quad (1)$$

where θ is the rotation angle, r is the radius of the sphere, p_o is position of a globe and R is rotation matrixes defined as:

$$R_1(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

$$R_2(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

$$R_3(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

For random globes environments (Stage C), the robustness is improved by driving vehicles to the nearest one to reduce energy consumption. Because this would consume the smallest amount of energy compared to drive to other globes. Therefore, it can reduce the possibility of vanish on the way or waiting and improve the possibility of successfully charging.

2.4. Robustness

Robustness on both Stage B and C was enhanced by methods below:

a. *Random Charing Threshold*: Noticing that setting charge threshold as a fixed number would result in most vehicles steering to globes at the same time when first charging and leads to vehicle vanish, different first-time charging threshold can ease this phenomenon because this method is similar to roughly queuing up vehicles to charge for the first time by randomly assigning them a float number as the order.

b. *Charing Frequency Reducing*: The capability of a globe is limited. To support more vehicles using a certain number of globes, the frequency of charge should be minimized. (1) The throttle will influence the discharge rate and in turns influence the frequency of charging. By decreasing cruise throttle, the frequency will be reduced. Therefore, the total charge amount in a certain duration can be reduced, and more vehicles can be supported by the same number of globes compared with a high cruise throttle method. (2) Reducing the charge threshold also helps to reduce the frequency. This is illustrated in formula (2).

$$N_c = \frac{Duration}{(Total\ Charge - Threshold)/Discharge\ Rate} N \quad (2)$$

N_c is the number of vehicles that need to charge and N is the number of vehicles gilding in space.

Some potential enhancement will be discussed in the rest of this section.

(1) *Priority*: Queuing up low battery vehicles and assign priority by charge level and distance will decrease collision because it forces vehicles to replenish charges sequentially. This was not implemented because of the complexity of coding.

(2) *Globe Position Predication*: The globe velocity can be detected by sensors. It can help vehicles to predicate the position of globes by the time of reaching. The limitation of this strategy is that globes also have accelerations which requires other unknown techniques to address.

(3) *Centre Coordinator*: Shared memory based or message-based coordinator can increase the precision of globe position, but this strategy is questionable in a physical deployment [3].

2.5. Shrink Strategy

Basing on the above structure, vehicles can be alive stably, so the shrink strategy can be developed under the assumption that no vehicles will vanish without scheduling. It contains two parts, which are a distributed information sharing method and agreement on action.

Information Sharing: Implementation of the information sharing method is augmenting the message by two additional elements which are a list of vehicle No. and a timestamp. The update principle is illustrated in Figure 3 and Figure 4 is the simplified version. This principle can guarantee that Message 2 always contains the latest list between M1 and M2.

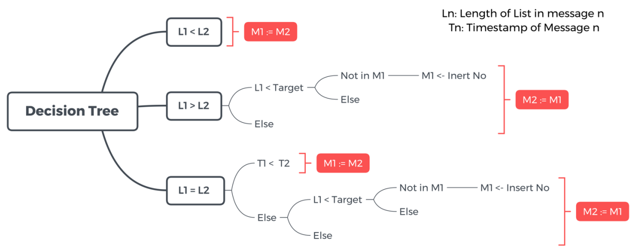


Figure 3: Decision Tree of update principle between two messages.

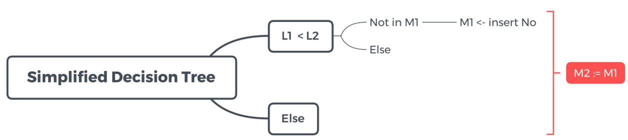


Figure 4: Simplified Decision Tree

The intuition is that every vehicle adds its No. with a timestamp to a list and send this message to others. If this list is not full then the receiver will add itself and send the message with a new timestamp again. If the list is full, only the latest one will be preserved and sent forward. After iteratively doing this, all vehicles will hold the same list eventually. Let the length of the list be the target number, an array of vehicles to be preserved is gotten.

Action agreement: The first agreement is an assumption about when all vehicles have fully exchanged messages. It is hard to check whether all vehicles have gotten the same message in a distributed system. Setting a certain point (e.g. 20s after starting) as agreement can address this problem in an approximate way. The second agreement is that vehicles whose No is not in the list terminate after the time set in agreement 1. By doing this, eventually, the number of existing vehicles will equal the length of the list.

A Secure but Complex Method: Raft consensus algorithm solves the problem of getting multiple servers to agree on a shared state [4]. Every participant would be one of the three roles (Follower, Candidate, Leader). The Leader can arrange others to do some operations. Therefore, this algorithm can be applied to shrink swarm size. Moreover, it is secure because it does not require the assumption that no vehicles vanish without scheduling. However, it is too difficult to be implemented.

3. Experiments

Some experiments were carried out to evaluate the implementation.

3.1. Experiment Environment

Configuration: Lab Computers Intel(R) Core (TM) i7-7700 CPU @ 3.60GHz
System: Ubuntu 19.04
IDE: GNAT Programming studio 2017
To reproduce experiments, parameters need to be adjusted according to Appendix.

3.2. Capability

Capability refers to the maximum and the minimum number of vehicles that the implementation can stably support. The result is that the capability (within 10 minutes) of vehicles is between 1 to 350 under single globe environment. The lower bound for random globes is 2 but the upper bound is quite different basing on different configurations. After running for ten minutes, the most excellent performance is keeping 440 vehicles alive.

Theoretically speaking, the maximum in Random Globes mode should be at least twice bigger than the maximum in Single Globe Mode because there are at least two globes in the first mode. However, the observed stable upper bound is significantly smaller than expected. The explanation is that globes randomly disappear, which may cause a scenario that many ships are waiting to charge in a globe and this globe disappears. So, they can only steer to other globes, which may lead to either vanish on the way or crowdedness in another globe.

3.3. Robustness

This section discusses robustness-enhance methods mentioned in Section 2.4.

a. Random Charing Threshold: Under the same configuration, the upper bound is 225 for a random first-time charge threshold and 210 for a fixed threshold. This result can support the analysis in section 2.4.a. However, the difference is not significant. An explanation is that random first-time charge threshold can only prevent vanish in the initial stage but cannot change the upper bound. Therefore, the stable number will eventually converge to the limitation of this configuration.

b. Charing Frequency Reducing:

Table 1: Effect of Cruise Speed and Charge Threshold.

Value	Threshold[1]	Speed[2]
0.1	304	286
0.2	359	303
0.3	346	324
0.4	322	323
0.5	271	326
0.6	213	304
0.7	200	249
0.8	142	240
0.9	65	216
1	5*	195

The result in Table 1 shows a pattern that the maximum will firstly increase and then decrease when the threshold or speed increases. The explanation is that vehicles do not need to preserve energy to deal with losing the globe position which allows setting the threshold as a small number to reduce charge frequency. And cruise speed will influence the speed of leaving the globe. If it is too small, charged vehicles may block others to charge. If the speed is high, energy consumption will be high and result in a high charge frequency. The reason for 5* is that vehicles will not leave the globe after charging and prevent others from charging. A conclusion derived from this is that a globe can charge at most 5 vehicles simultaneously.

The benefits of a small charge threshold are not only reducing charge frequency but also prevent crowdedness. Because such a threshold will result in a small waiting time tolerance, which means if there are too many vehicles waiting to charge using the same globe then some of them may vanish rather than keep waiting and block new comers.

3.4. Reliability

The Shrink Strategy was developed basing on the assumption that no vehicles vanish without scheduling, which requires high reliability of the first two components.

Table 2: Accuracy out of ten rounds of test

Initial	300	250	200	150	100	50
Target	150	125	100	75	50	25
Accuracy	60%	100%	100%	100%	80%	90%

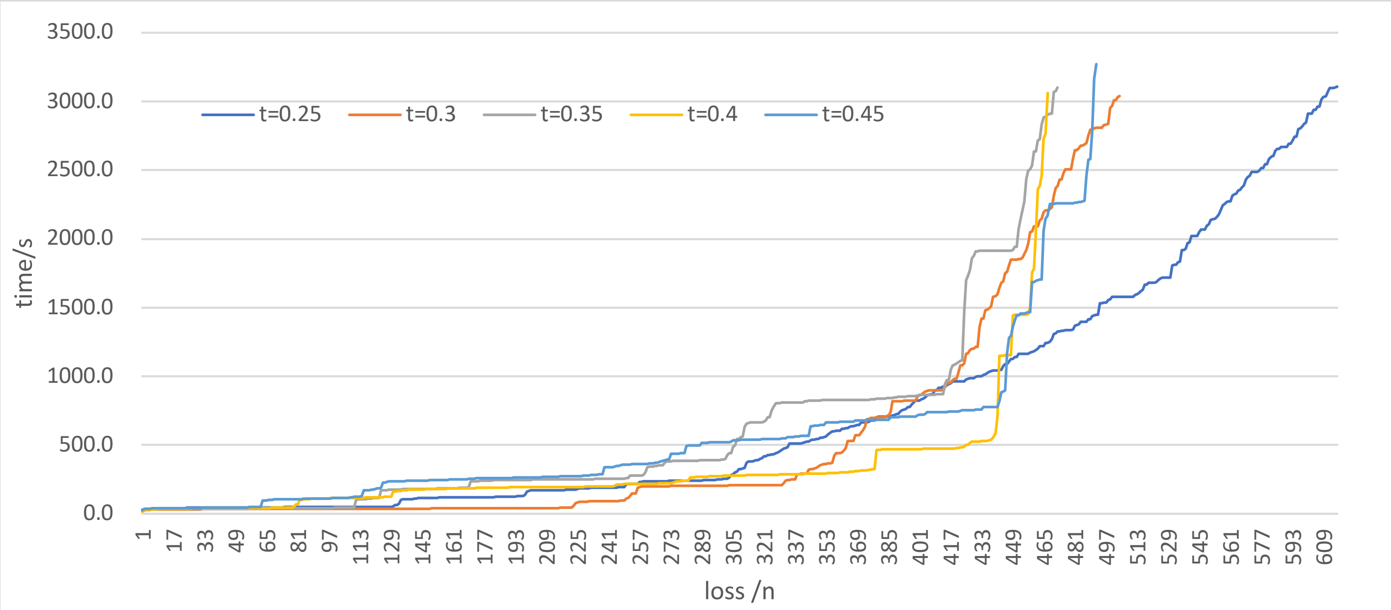
Basing on the experiment, the system cannot work well in some situations. If the initial size is too big (above 250), the system cannot guarantee the assumption during information exchanging, some vehicles may have been added to the list but disappear because of running of out charge. If it is too small (below 100) information sometimes cannot be fully exchanged because time is not enough. However, when

[1] Number of vehicles alive after run for 5 minutes

[2] Same as 1

the terminate starting time was set as 60 (framerate above 25Hz), the last two test cases can achieve 100% accuracy. So, such a conclusion can be drawn that, within the system capacity, the only factor that can influence the accuracy is the time for information exchange. Once vehicles vanish for more than expected and the swarm size is below 250, it can be fixed by increasing the time span allowed to exchanging information.

Figure 5: Comparison of loss rate in different threshold(t)



Another reliability indicator is the loss rate in random globes mode.

This diagram (Figure 5) shows the loss number against time using different charge thresholds. The start number is 800 and vehicles vanish dramatically to about 420 under all situations because 800 outweighs the capacity. When the threshold is 0.25, the loss rate is significantly higher than others. The reason is that globes move fast in random globes mode and it becomes difficult for vehicles to reach globes by steering to the position for once. So, detection or further communication is needed. During this period, some vehicles may have vanished because they do not have enough energy to finish this process.

Some horizontal sections after (almost) vertical lines can be observed, which mean a dramatic reduction after number of vehicles keeps stable for a long period. This is evidence of what mentioned in Section 3.2 Capacity.

4. Conclusion

To maintain as many vehicles alive as possible for as long as possible and shrink the swarm to a specific size, a robust method was proposed. Although this strategy cannot work well under some extreme situations, such as big swarm size and long-time running in random globes environment, reliable performance can be expected in moderate situation.

References

[1] Zimmer, U. (2019). GLIDING IN SPACE. [eBook] Canberra , p.1. Available at: https://cs.anu.edu.au/courses/comp2310/Assignments/Assignment%201.pdf

[2] Zimmer, U. (2019). GLIDING IN SPACE. [eBook] Canberra , p.1. Available at: https://cs.anu.edu.au/courses/comp2310/Assignments/Assignment%201.pdf

[3] Zimmer, U. (2019). GLIDING IN SPACE. [eBook] Canberra , p.3. Available at: https://cs.anu.edu.au/courses/comp2310/Assignments/Assignment%201.pdf

[4] freeCodeCamp.org. (2019). Understanding the Raft consensus algorithm: an academic article summary. [online] Available at: https://www.freecodecamp.org/news/in-search-of-an-understandable-consensus-algorithm-a-summary-4bc294c97e0d/

Discusses

u6921112 Method of stage d
u6921098 Raft algorithm
u6920122 Method of improving maximum
u6921163 Method of improving maximum
u6920831 Simplify the system
u6920336 Message passing

Appendix: Configuration for Experiment

To achieve the data displayed in experimental analysis, parameters in package *Vehicle_Control* need to be adjusted.

Safty_Distance	is the	*Radius*;
Charge_Lower_Bound	is the	*Threshold*;
Cruise_Throttle	is the	*Cruise Speed*;
(Terminate_Stage_Start, Terminate_Stage_End)	is the	*bound of Termination Check*.

1. Upper and lower bound

Configurations are shown in Table 1.

Table 1: Configuration for Bound Test

Mode	Target	Initial No.	Threshold	Speed	Radius
Single	350	400	0.2	0.5	0.3
	5~10	5~10	0.8	1	0.3
	<5	<5	1	1	0.1
Random	440	800	0.35	0.2	0.4
	<20	<20	0.9	1	0.1

2. Random Charing Threshold

Mode = Single; Initial No. = 350; Speed = 0.6; Radius = 0.3;
Random first-time charge threshold: (0.5, 1.0);
Fixed first-time charge threshold: 0.5
After first charge: Threshold = 0.5

3. Comparison

a. Speed: [0.1, 1.0]

Mode = Single; Initial No. = 400; Threshold = 0.5; Radius = 0.3

b. Thresholds: [0.1, 1.0]

Mode = Single; Initial No. = 400; Speed = 0.5; Radius = 0.3

4. Accuracy of Shrink Strategy

Configurations are shown in Table 2.
Mode = Random

Table 2: Configuration for Accuracy Test

Initial No.	Target	Speed	Radius	Threshold	Terminate Stage
300	150	0.6	0.3	0.5	150~155
250	125	0.6	0.3	0.5	120~125
200	100	0.6	0.3	0.6	100~105
150	75	0.6	0.3	0.6	80~85
100	50	0.6	0.3	0.6	45~50
50	25	0.8	0.3	0.6	25~30

5. Loss Rate in different Thresholds

Mode = Random; Initial No. = 800; Speed = 0.2; Radius = 0.3