Shing-Tung Yau Awards Research Project

Understanding Flocking Dynamics in Nature

by

Li Ang[1], Lim Sung Hyun[1] and Wang Qi[1]

[1]NUS High School of Mathematics and Science

Teacher Advisor

Chai Ming Huang[1]

[1]NUS High School of Mathematics and Science

External Mentors

Prof Bao Weizhu[2] and Cai Yongyong[2]

[2]Department of Mathematics, Faculty of Science, National University of Singapore

Summary

Flocking is a behavior exhibited by birds, and other living beings. The research results on flocking have been applied to numerous real-life situations like fire evacuations and animation modeling. Studies on this behavior have been done since 1986 when Craig Reynolds first came out with a computer simulation. In our project, we constructed our own rigorous mathematical models on flocking; and we also wrote programs which simulate real-life flocking situations on the computer. We have successfully found a suitable mathematical model to describe certain flocking behaviors to some extent.

Abstract

Many of the previous researches have made great discoveries about flocking. The Reynolds rules [17], the most basic knowledge in flocking, are cohesion, separation and alignment. And like many others, this project aims to construct mathematical models and computer programs to simulate the flocking behavior.

Constructing mathematical models is crucial to the studies of flocking. Normally, a mathematical model describes the motion of each agent in a flock in order to make agents follow the Reynolds rules.

To apply the theory into real-life, computer simulation of the behavior is necessary. In this project, computer simulations are made by MATLAB. In order to use the program efficiently, we did the programs in modules, so some common parts can be used for all kinds of models. With the modules which simulate real-life situations such as constructing obstructions, and those that allow operators to change the parameters inside a model, much revision on the mathematical model has been done. Hence the model is made to better resemble real life situations.

Background

Since the mathematical model in this paper is constructed based on the Reynolds Rules, it is especially important to make an introduction on the three factors, namely cohesion, separation and alignment.

Cohesion is easy to understand, and it is one of the necessary conditions for flocking phenomenon to occur because only when there is a crowd do we observe flocking.

Separation is easy to explain if we consider real-life situations. When birds fly or fish swim or people walk, sufficient space is needed between individuals. And last but not least, alignment is definitely the most important factor in flocking.

Alignment is the key factor to determine whether this is a flocking phenomenon and what kind of flocking the crowd is exhibiting.

Flocking phenomena can be formulated using mathematical models. And there are many ways to do so. Since the motion of each agent is the key point of the models, certain amount of physics knowledge is also needed when we construct our models. The models may be either simple or complicated based on the methods. In this paper, the models range from the simple ones which only consider the velocity of each agent, to the models which consider even the agents' eyesight and are formulated in the complex plane.

Classification of Flocking Behaviors

A lot of classification work has been done before models are constructed. And, as a result, a decision is made to classify all flocking phenomena into four categories, namely: attractive and repulsive interaction, toruses, flocks, and flocks with leaders.

Attractive and repulsive is easy to explain. It is natural for agents to repel each other when they get too close, and sometimes agents tend to attract each other to form crowds. This happens in almost all flocking phenomena.

Toruse is a term adapted from *Collective Behavior Coordination with Predictive Mechanisms* [9]. It occurs when agents move around an empty center. It is commonly observed in bird flocks.

Flock is also adapted from *Collective Behavior Coordination with Predictive Mechanisms* [9]. It refers to the phenomenon when agents move collectively in one direction.

Last but not least, flocks with leaders occur when there is a leader which changes the direction of which the flocking is heading in.

However, real-life situations are far more complicated. This is because in real-life, when flocking phenomenon occurs, it may fall into multiple categories. For example, when there is a fire in a building, people have to follow an instructor. On the other hand, those who cannot see the instructor tend to catch up with the crowd; and at the same time, each person needs sufficient space; when the crowd gets overly crowded, people tend to scatter. Hence, this situation falls into three categories which are attractive and repulsive interaction, flocks, and flocks with leaders.

Objective

The objective of the project is to come up with an authentic mathematical model to describe the flocking behavior with the aid of computer programming.

Results

Section I: A Model Which Considers the Eyesight of Agents

The main concern of this model is the 'eyesight' of an agent. This means that an agent has a certain eyesight range and hence would concentrate mostly on the objects that come into its eyes, while not paying much attention to other agents off its eyesight. *Figure 1* illustrates how an agent considers other agents with different degree of importance.

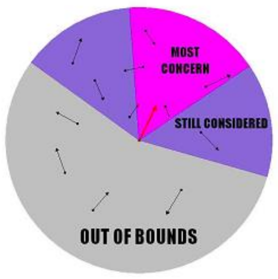


Figure 1

1. **Eyesight function and General Direction**

We hence shall construct a function that determines how much degree of importance is put on an agent's (to be referred as Observers) neighbors, namely, “eyesight

function" s_r***:*** $\boldsymbol{C \to R}$ The observer in this eyesight function is assumed to be at origin, pointing at $+x$ direction, but by rotating and translation, the function can be applied to an arbitrary agent on the same plane.
To summarize, an agent i with position x_i and velocity v_i (Be reminded that these values are complex numbers) would consider a position z with degree of importance $S_r\left(\frac{z-x_i}{\widehat{v_i}}\right)$.

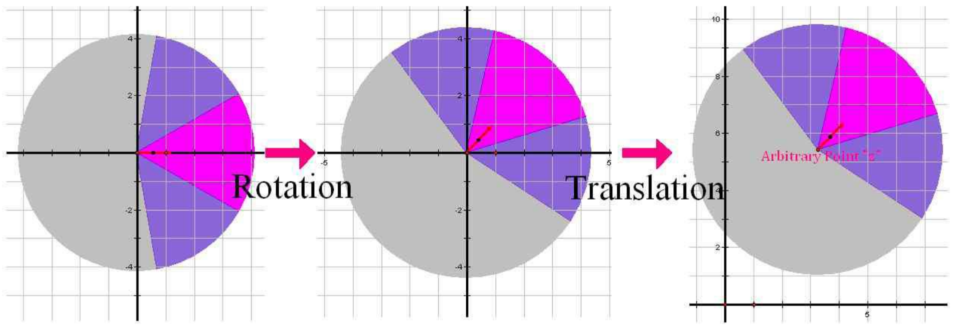


Figure 2

At every moment, each agent determines "general direction" of its neighbors, determined with respect to its own eyesight. General Direction g_i hence can be considered as the weighted mean of velocities of other agents with weight (degree of importance) $S_r\left(\frac{z-x_i}{\widehat{v_i}}\right)$, i.e.

$$g_i = \textit{weighted mean} = \frac{\displaystyle\sum_{\substack{j=1\sim n \\ j\neq i}} s_r\left(\frac{x_j-x_i}{\hat{v}_i}\right)v_j}{\displaystyle\sum_{\substack{j=1\sim n \\ j\neq i}} s_r\left(\frac{x_j-x_i}{\hat{v}_i}\right)}$$

2. Constructing Eyesight Function

Now we construct the eyesight function. Since the function is defined for complex domain, it is also easy to consider these two factors separately, each using argument and magnitude. *i.e.*

$$s_r(z) = A_r(\arg(z)) \cdot D(|z|)$$

a. Angle Function

The archetype of function we will use for angle function would be $\frac{1}{2}(cos(x)+1)$, and since sight range would differ from different types of agent, we introduce a parameter r that indicates sight range: if the sight range ranges from – to $+\theta$, denote r= θ. According to the sight range, the graph of $\frac{1}{2}(cos(x)+1)$ will be stretched along *x-axis* (angle axis), and so we have a prototype angle function $\frac{1}{2}\left(cos\left(\frac{\pi x}{r}\right)+1\right)$

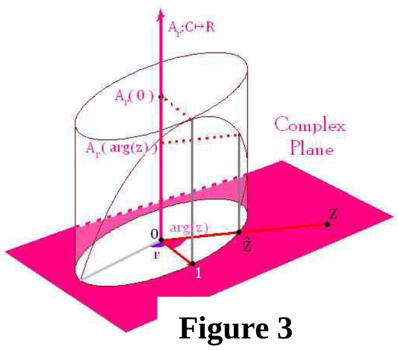


Figure 3

But we would need to eliminate the negative part by multiplying a function *h*:

$$h(\theta) = \begin{cases} 1 & \theta \in (-r, +r) \\ 0 & otherwise \end{cases}$$

So that we have the angle function:

$$A_r(u) = \frac{1}{2}\left(cos\left(\frac{\pi u}{r}\right) + 1\right) \bullet h(u) \qquad where\ u = arg(v_j) - arg(v_i)$$

b. Distance Function

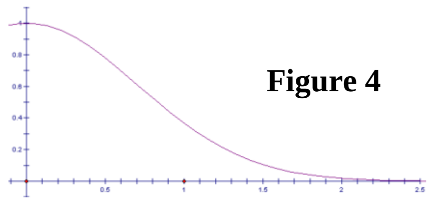


Figure 4

We use $\frac{1}{1+x^2}$, as shown in *Figure 4*, the graph fits quite well what a distance function should look like intuitively, including the fact that the function has fast convergence rage (hence indicating the fact that objects at far distance are of negligible effect to observer.) Hence, in conclusion, the eyesight function *s(x)* can be expressed as:

$$s_r(z) = \frac{h(\arg(z))}{2(1+|z|^2)}\left(1+\cos\left(\frac{\pi \arg(z)}{r}\right)\right) \text{ where } h(\theta) = \begin{cases} 1 & \theta \in (-r,+r) \\ 0 & \theta \notin (-r,+r) \end{cases}$$

3. Cohesion and Avoidance Factor

Among the three Reynolds Rules for flocking, the alignment factor is considered above, however, we have yet to apply the cohesion and avoidance rule. This can be done simply by superposing it alignment factor. The rules to be followed are:

(1): An agent would not want being too close to its neighbors.

(2): Rule (1) however does not mean that the avoidance factor would shoot to infinity near distance zero.

(3): An agent would move towards any agent that comes into its eyesight. Hence the cohesion/avoidance function should be asymptotic to x-axis.

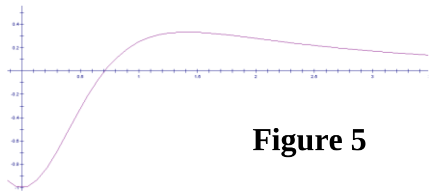


Figure 5

As a result, the preferred shape of the cohesion/repulsion force graph (plotted against distance) is shown in *Figure 5,* which is well-demonstrated using $\frac{2x^2-1}{(1+x^2)^2}$, cohesion and avoidance factor is determined by weighted mean of the cohesion/avoidance function with weight $S\left(\frac{x_j-x_i}{\widehat{v_i}}\right)$ in order to calculate the cohesion/avoidance factor. i.e.

$$c_i = \frac{\sum\limits_{\substack{j=1\sim n\\ j\neq i}} s\left(\frac{x_j-x_i}{\hat{v}_i}\right)\gamma(x_j-x_i)}{\sum\limits_{\substack{j=1\sim n\\ j\neq i}} s\left(\frac{x_j-x_i}{\hat{v}_i}\right)} \quad \text{where} \quad \gamma(z)=\left(\frac{2|z|^2-1}{(|z|^2+1)^2}\right)\hat{z}$$

4. Turnaround Factor

An unfavorable situation illustrated in Figure 6 however might occur by only considering three Reynolds Rules in our model, as an agent looks away from the main cluster of other agents while straying away because there are no other agents to affect its pathway.

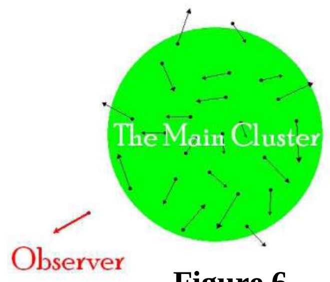


Figure 6

Therefore to prevent such a situation, we shall introduce another factor which is responsible for turning around when there is not a big number of agents in the observer's eyesight. The preferred graph for the "turnaround factor" is shown in Figure 7.

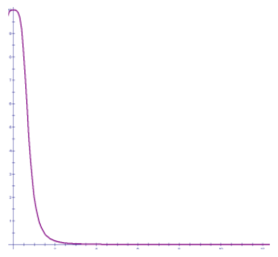


Figure 7

Again it is to be considered that the value does not shoot to infinity at zero, but should reach reasonably a big value. A function fitting well to this graph is:

$$\frac{k_1}{1+(4\,x/L)^4}$$

where k_1 is the function value at origin, the "turnaround strength" which is to be of moderately big value about 10, and L is the "loneliness factor" which determines the minimal number of x (*i.e.* number of agents) such that the observer does not feel "lonely" so not turning around, hence having negligible function value at $x=L$. .
Number of the agent to be input on above expression is of course to be determined using weight function as usual:

$$\sum\limits_{\substack{j=1\sim n\\ j\neq i}} s\left(\frac{x_j-x_i}{\hat{v}_i}\right)$$

Turnaround factor is to be added into acceleration, and in order to not alter the agent's speed, turnaround factor should act in perpendicular direction to the velocity, *i.e.* $i\hat{v}_i$. Hence, the turnaround factor t_i is determined as:

$$t_i = \frac{k_1}{1 + \left(\frac{4}{L} \sum_{\substack{j=1\sim n \\ j \neq i}} s\left(\frac{x_j - x_i}{\hat{v}_i} \right) \right)^4} \cdot i\hat{v}_i$$

5. The Model

By combining derivations above, we obtain:

$$a_i = g_i + c_i + t_i$$

as the differential equation of this flocking model.

Computer Simulation and Conclusion

Programming is a very important part in this project as it will justify the accuracy of each model that has been constructed and also help in determining the parameters in a model. In this project, the program is done in MATLAB.

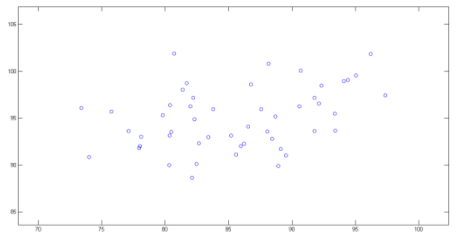
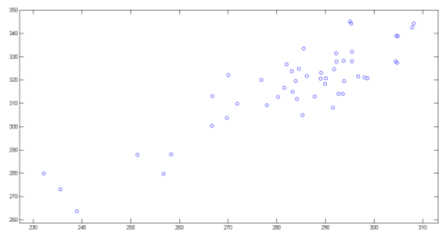
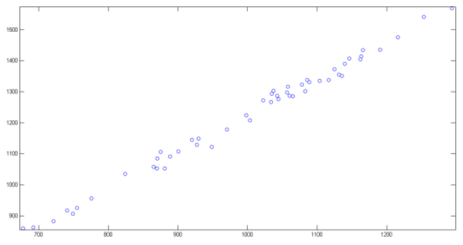
The process of programming is broken into following steps:

- Figure out what is necessary for a model;
- Construct coding for each necessary part to form coding modules;
- Connect the modules together to construct programs for each model.

Before programming the final model, a few simulations are done on existing models. So, the program is written in modules in order to reuse the common parts of programs. An example of the module on constructing obstructions for the agents is attached in the appendix, and it can be reused for any model.

The simulation helps in making adjustments to parameters in the model. Therefore, a few parameters are requested from the operator before the program is launched. And by changing some parameters, different types of flocking phenomena can be observed. The module for the model in this paper is attached in the appendix, where the loneliness factor and the range must be manually entered before the simulation starts.

After much adjustment, our finalized model resembles real-life situations successfully. An example would be the bird flocks. Table1 shows the similarity between our computer simulation and bird flocks in real life.

[18]	At t=5s, like birds that are about to fly off, the agents in the simulation are scattered .	
[19]	At t=10s, like birds which just fly off the land, the agents in the simulation are still scattered, but they are moving toward a common direction and tend to form a line or a crowd.	
[20]	At t>30s, most of the agents in the simulation would line up and move towards a common direction, resembling the birds which are high up in the sky, moving in flocks.	

And the simulation has shown that this specific model follows the Reynolds rules as they exhibit cohesion, separation and alignment. Hence, the model is successful to a certain extent.

However, the following problems have occurred in the simulation even after many attempts to amend the model:

- Agents get faster and faster as time passes by;
- The turnaround factor does not work well enough for agents who are at the head or the tail of the queue, while they are expected to turn around when they are moving towards a place with few agents.

Possible Extension

Our model is not yet perfect and we can make further improvements in these areas:

- Introduce a cap for the velocities of agents. This is because in real-life, animals do have a speed limit, instead of accelerating freely;
- Introduce a term in order to achieve better cohesion. This is more likely to be an improvement on the turnaround factor;
- A 3D simulation has been done on existing models, and it resembles the real-life situation better. However, the model in this paper does not allow so as it has been done in the complex plane, so constructing a model in real number plane is a way to improve the project;
- In this current model, the motion of each agent is considered. However, it is more often to consider a whole crowd of agents in real-life. Further investigation can be done in this field too.

Appendix

- **Program codes:**

Creation of obstructions:

```
%Creation of objects
nx=input('Number of objects in x direction: ');
for mx=1:nx
    xs(:,mx)=input('starting x value: ');
    xe(:,mx)=input('ending x value: ');
    ay(:,mx)=input('y value: ');
end
ny=input('Number of objects in y direction: ');
for my=1:ny
    ys(:,my)=input('starting y value: ');
    ye(:,my)=input('ending y value: ');
    ax(:,my)=input('x value: ');
end

%Bouncing back at the walls

choice=input('Enter 1 if agents bouces back when they are in
coneact with walls, or 2 if agents move along theobjects');
for at=1:a
    if choice==1    %boucing back
    for px=1:nx
    if abs(yt-ay(:,px))<0.1
        if xe(:,px)>xt(at,:)>xs(:,px)
            vyt1=-vyt1;
        end
    end
    end
    for py=1:ny
    if abs(xt-ax(:,py))<0.1
        if ye(:,py)>yt(at,:)>xs(:,py)
            vxt1=-vxt1;
        end
    end
    end
elseif choice==2;    %going along the walls
    for qx=1:nx
        if abs(yt-ay(:,qx))<0.1
             if xe(:,qx)>xt(at,:)>xs(:,qx)
                 vyt1=0;
             end
        end
    end
    for qy=1:ny
        if abs(xt-ax(:,qy))<0.1
             if ye(:,qy)>yt(at,:)>ys(:,qy)
                 vxt1=0;
             end
        end
    end
else
    disp('The input is faulty. The objects are not functioning');
    end
end
```

Sample codes for the model in this paper:

```
a=input('number of birds= ');
r=input('range(0<r<2*pi)= ');
L=input('loneliness factor= ');
k1=input('testing k= ');
s=rand(a,1);Ar=rand(a,1);at=rand(a,1);g=rand(a,1);c=rand(a,1);at=
rand(a,1);p=rand(a,1);d=rand(a,1);u=rand(a,1);vcap=rand(a,1);ti=r
and(a,1);
x=10*rand(a,1)+(10*rand(a,1))*1i;            %initial position
v=rand(a,1)+rand(a,1)*1i;
axis([-200 200 -200 200]);        %axis dimention
vep=1.0e-9;

while a>1
    for k=1:a

vcap(k,1)=(v(k,1))/sqrt(((real(v(k,1)))^2)+(imag(v(k,1)))^2);
        nom=0;
        sums=0;
        nomc=0;
        sumc=0;
        for m=1:a
            if m ~= k
               u(m,1)=angle(v(m,1))-angle(v(k,1));
               d(m,1)=(x(m,1))-(x(k,1));
               p(m,1)=d(m,1)/vcap(k,1);
               if -r<u(m,1) && r>u(m,1)
                  h(m,1)=1;
               else
                  h(m,1)=0;
               end
               Ar(m,1)=(cos((pi*u(m,1)/r))+1)*h(m,1);
               s(m,1)=Ar(m,1)*exp(-(abs(p(m,1))^2));
               nom=nom+s(m,1)*v(m,1);
               sums=sums+s(m,1);
               cm(m,1)=(1-2*abs(p(m,1))^2)/((1+abs(p(m,1))^2)^2);
               nomc=nomc+cm(m,:)*v(m,:);
               sumc=sumc+cm(m,:);
            end
        end
        ti(k,:)=(k1*1i*vcap(k,1))/(1+((4/L)*(sums+vep)));
        g(k,:)=nom/(sums+vep);
        c(k,:)=nomc/(sumc+vep);
        at(k,:)=g(k,:)+c(k,:)+ti(k,:);
        v(k,:)=0.05*at(k,:)+v(k,:);
        x(k,:)=0.05*v(k,:)+x(k,:);
    end
        clf;
        xmin=min(real(x(:,:)));xmax=max(real(x(:,:)));
ymin=min(imag(x(:,:)));ymax=max(imag(x(:,:)));
        plot(real(x(:,:)),imag(x(:,:)),'o') ;
        axis([xmin-5 xmax+5 ymin-5 ymax+5]);
        pause(0.2);
end
```

For more complete codes, please email wqhello@gmail.com

- **Reference**

1. Duan, H.B et al, *Optimal formation reconfiguration control of multiple UCAVs using improved particle swarm optimization*. Journal of Bionic Engineering , 5,340-347 (2008),
2. Guo,C.X et al, *A pooled –neighbour swarm intelligence approach to optimal reactive power dispatch*. Journal of Zhejiang University SCIENCE A , 7(4),615-622 (2006),
3. Li, Q et al, *A new clustering algorithm Based on flocking with virtual leaders. Journal of Electronics and Information Technology*, 31(8), 1847-1851(2009)
4. Li, Z.G et al, *Flocking for swarm systems with fixed topology in a changing environment*. Control Theory Appl, 6(3),333-339 (2008),
5. Liu, X.P et al, *Swarm intelligence for classification of remote sensing data*. Science in China Series D: Earth Sciences (2008)
6. Lu, S.Y.et al, *Consensus algorithm for second-order networks with time-delays.* International Journal of Systems and Control,3,199-209 (2008),
7. Wang, D.M et al, *Leader-following coordination of multi-agent systems with information feedback*. Journal of Systems Engineering and Electronics, 20(4), 823-828 (2009)
8. Yang, S.X et al, *A review of flocking*. Control Theory and Applications, 26(09), 10-14 (2007)
9. Hai-Tao Zhang, Michael ZhiQiang Chen, Guy-Bart Stan, Tao Zhou, and Jan M. Maciejowski *Collective Behavior Coordination with Predictive Mechanisms*, IEEE CIRCUITS AND SYSTEMS MAGAZINE
10. Sylverin Kemmoe´ Tchomte´, Michel Gourgandc, *Particle swarm optimization: A study of particle displacement for solving continuous and combinatorial optimization problems*
11. Felipe Cucker and Steve Smale, *Emergent Behavior on Flocks*

12. O. Burchan Bayazit Jyh-Ming Lien Nancy M. Amato, *Roadmap-Based Flocking for Complex Environments*
13. Reza Olfati-Saber, *Flocking for Multi-Agent Dynamic Systems: Algorithms and Theory*, IEEE TRANSACTIONS ON AUTOMATIC CONTROL, VOL. 51, NO. 3, MARCH 2006
14. Felipe Cucker, Ernesto Mordecki, *Flocking in noisy environments*, Science Direct
15. John Toner, Yuhai Tu, *Flocks, herds, and schools: A quantitative theory of flocking*, PHYSICAL REVIEW E OCTOBER 1998 VOLUME 58, NUMBER 4
16. J. R. Raymond, M. R. Evans, *Flocking regimes in a simple lattice model*, PHYSICAL REVIEW E 73, 036112 2006
17. Craig Reynolds, Boids, http://www.red3d.com/cwr/boids/
18. Picture adapted from http://www.treehugger.com/seagull-flock-jj.jpg
19. Picture adapted from http://kirkcarterphotography.com/look_up_1/4109_bird_flock.jpg
20. Picture adapted from http://fridays.ws/uploaded_images/birds-flying-764101.jpg

- **Statement of Contribution**

All student authors contributed equally to this work. Lim Sung Hyun and Li Ang worked on the mathematical model, and Wang Qi came out with the computer programs. Mr. Cai Yongyong helped us in debugging programs. Prof. Bao Weizhu and Mr. Chai Ming Huang Royce vetted our research paper and gave valuable comments on our project.

- **Acknowledgement**

We would like to express our deepest appreciation to Prof Bao Weizhu, Mr. Cai Yongyong and Mr. Chai Ming Huang for their guidance and help throughout the project.