Dynamic Field of View as Collision Detection for Autonomous Multi-agent

Nazreen Abdullasim, Norhaida Mohd Suaib, Abdullah Bade, Zhigeng Pan, Qingshu Yuan

Abstract—In this paper we present a hybrid collision detection technique in crowd simulation by constructing a field of vision for each agent. The field of vision is represented as a bounding volume that is dependent to the agent's locomotion variation that results in a variation of its length and angle. The variation will hence create more dynamic execution and selective toward agent's collision testing between object and its field of vision. In many crowd simulation behaviors such as flocking, agents are considered as a whole summation of force value that determine the agent's behavior and decision. This is similar to what happens in the real world; human and animals behave and react upon what they perceive. This research presents agents' unique perception based on their own speed variation thus producing more dynamic and selective collision response execution. This technique also gives more possibility to design conceptual bounding volume that represent agent's field of vision based on linear line intersection.

Index Terms—Animation, Collision Detection, Computer Graphics, Simulation, Virtual Reality

1 Introduction

According to Musse and Thalman, crowd simulation is about modeling a collection of interacting agents [1]. Many applications have been using the technology for movie making, crowd management training systems and computer gaming. In computer gaming, it enhances the immersion of the virtual environment to become livelier instead of "ghost-town" like environment with no other non-playable character to interact with[2]. However, simulating a large number of pedestrian can be a challenging task. Pedestrian is a good example to depict crowd in real world environment. However there are many approaches on how crowd are modeled in real-time computer graphics simulation.

There are two main approaches on how crowd simulation is developed. It can be categorized as macroscopic and microscopic approach. Macroscopic approach is focusing on crowd entities in larger worldview. Macroscopic is about concerning more on the fluidity and the flows of the crowd itself rather than each of the crowd agent's behavior. Regression model[4] and Gas-Kinetic model[5] are few examples of macroscopic crowd modeling where the crowd is considered as one unit behavior as such that can be seen on liquid behavior.

On the other hand, Microscopic approach is modeled based on agent's individuality. Each agent is represented in their own respective attributes such as their own velocity, position and etc. The emergent behavior resulted from microscopic approach is more wavering and more lifelike than macroscopic approach. However, it does consume more computation resources as the number of agents for the crowd simulation increase. There are some work that adapt microscopic approach to represent groups as one entity such as [6]–[8]. Moreover this approach can still preserves the individuality of each agent within each group[9].

Microscopic approach can be designated into three sub categories which are social force, cellular automata and rule-based. Each category has its own distinctive approach such as social force is using second Newton's as a basis to define each agent by having its own local force that elaborates agents direction and magnitude[10]. Cellular automata approach on the other hand is about defragmenting virtual space or planar into segments where it is usually known as cells[11]. The behavior of the agent can be determined by the cell occupancy state. Therefore, this approach can guarantee free collision.

Rule based offer many diversity on the emergent of crowd behavior. The foundation for the crowd emergence is derived from steering behavior. Steering behavior is a set of methods that the agent executes when it reacts with its virtual environment." Boids" is one of the best notable work that incorporates steering behavior in crowd simulation[12]. It introduced alignment, cohesion and separation as its basic steering behavior that emerge into flocking behavior.

Steering behavior had becoming more diverse and many behaviors were introduced[13]. It represents a specific desired steering force that agent executing upon reacting with its virtual environment. It has to be expandable and adaptable with other steering behavior in order to generate emergence and realistic behavior[14]. A basic steering behavior is designated to perform only one behavior at a time and thus complex steering behavior

- *Nazreen Abdullasim is with the Kulliyyah of Information and Communication Technology, International Islamic University Malaysia. E-mail: ajendgreat@gmail.com*
- *Norhaida Mohd Suaib is with the UTMViCubeLab, Faculty of Computing, Universiti Teknologi Malaysia. E-mail: haida@utm.my*
- *Abdullah Bade is with the School of Science & Technology, Universiti Malaysia Sabah, Kota Kinabalu, Sabah. E-mail: abade08@yahoo.com*
- *Zhigeng Pan is with the Digital Media Interaction, Hangzhou Normal University, China. E-mail: zgpan@cad.zju.edu.cn*
- *Qingshu Yuan is with the Digital Media Interaction, Hangzhou Normal University. Email: yuanqs@cs.zju.edu.cn.*

can be generated by the execution of its combination [15]. However, there is one particular steering behavior that is going on constant improvement which is collision avoidance behavior. The behavior usually comprises of three steps which are construction of the agent's perception, collision handling, and collision response.

We organized our paper into the following sections: section 2 discusses briefly about the related works, section 3 deals on our proposed model, section 4 is about our implementation and result and we conclude at section 5.

2 Related Works

One of the earliest avoiding steering behaviors is separation[12]. It is the simplest form of action to move away from obstacles. Although it does not guarantee free collision between agent and obstacles, however it successfully creates a plausible collision avoidance maneuver for flocking behavior. Another similar work is on Social Force Model (SFM) where all the agents and other objects are assume as one summation of obstacle force value[10]. In many crowd simulations, the agent's locomotion information is predicted as by providing the information of the location few time steps ahead of its current position. Reciprocal Velocity Obstacles (RVO) approach is about a method of choosing the right velocity of obstacles to be avoided[16][17]. This approach can simulate collision avoidance in large amount of agents compare to its predecessor. Synthetic Vision Based incorporating field of vision or perception as a visual cognitive mechanism that detects obstacle that to be avoided[18]. This method able to produce smoother avoidance behavior and also decreases the pedestrian movement into congestion. Spatial Temporal Patterns (STP) enhance the field of vision design by creating segmentation within it[19]. Similar to the previous work, Speed Variant Information Space (SVIS) elaborate more on the design of the field of vision by proposing a more dynamic shape where the angle of the field is variant according to the speed of the agent[20]. Unlike the previous method, this approach offer more selective collision detection thus trigger less collision avoidance behavior.

3 Methodology

As describe in artificial intelligent model[15], the behavior of character AI comprises of:

1. Perception or sensory, that describe on how it perceives its environment
2. Motor or Action, a specific or combination of behavior as part of the character reaction stimuli

As a result, the character or agent will generate dynamic animation of various behaviors according to what it perceives from their environment. Taking the similar process, collision avoidance behavior can be induced into three simple stages, which are 1) construction of perception, 2) collision handling and 3) collision response. Figure 1 shows the typical process for collision avoidance behavior.

4 Velocity Perception

Fig. 1. Collision Avoidance Behavior Process.

Construction of agent's perception is about creating a bounding volume or ray-cast that represents the agent's field of view and its movement. From this construction we are able to distinguish the two objects for collision testing which are agent's perception and agent's line prediction. Agent's perception is where the agent only perceives and reacts upon its limited vision toward environment and obstacles. It is represented in a bounding volume form such as arc or triangle, or other common primitive bounding volume such as sphere, align axis bounding box, oriented bounding box and many other. Unlike arc shape such in[20], in our method we choose triangle as a bounding volume to represent the agent's field of view. Triangle naturally has three rays that represent it frontal, left and right perception distinctively and thus create a similar feature such in [19] by having spatial partitioning from the field of view itself. Moreover the partitioning can invoke specific movement according to the side of where the intersection happens. The agent's prediction line is basically a linear line that results from the projection of the current agent's position to the future agent position in certain few time steps ahead. Therefore, the collision testing can be conducted using between agent's perception which is in triangular form and agent's movement prediction where it is represented as linear line. Figure 2 shows the example of testing between the objects.

5 Construction of Velocity Perception

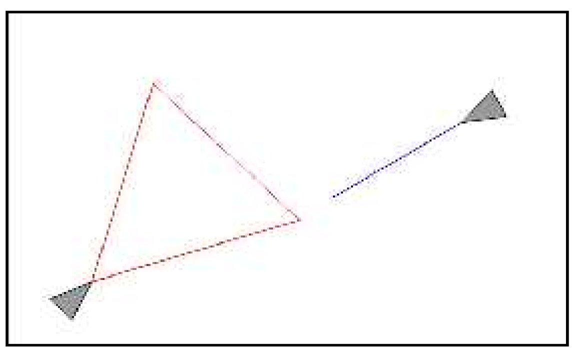

Fig. 2. The shaded triangle is presented as agent, the bigger triangle is the agent's perception and the single linear line is the agent's future movement.

In SVIS[20], the agent's speed variant becomes the factor that changes the value of the arc's angle. Similar to this process, the agent's speed variant becomes the factor not only for the size of the triangle but also the length of the triangle. This length depicting the length of agent's focal in relationship with its speed. Thus our method proposed more dynamic and variant field of view shape that offers more variant decision making toward detecting collision and obstacles. In order to construct the perception, we outline few steps based on the agent's speed information of the current particular time steps. As depicted in figure 3, the steps of constructing the perception are as follows;

1. Determine the coefficient value of predict time steps
2. Find the predict location
3. Find the right location
4. Find the left location

Since the dynamic triangle bounding volume size depends on the agent's speed, it is the first information that needs to be acquired constantly during each time steps. In order to determine the predicted position as in formula (1) of the agent for the next particular time step, a certain value has to be defined as a constant coefficient value. This value is called FPSCoefficient. Thus, we able to generate agent's future position by implementing linear interpolation to its current position. The dynamic of the agent's future position are changing based on the value of agent's velocity.

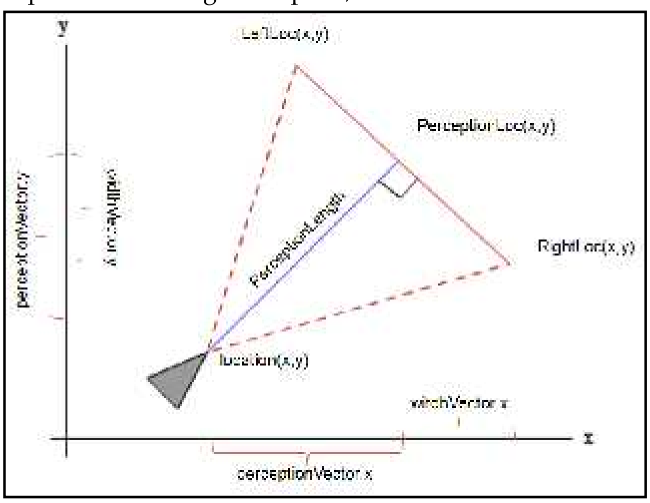

Fig. 3. The construction of the agent's perception.

PredictVector = Velocity + FPSCoefficient
PerceptionLoc = Location + PredictVector (1)

The next step is to determine the left and right location of the triangle. Since the perception length is perpendicular with frontal ray, the left and right location can be determined by finding the width vector of the frontal ray. The perpendicular relationship between width vector and predict vector can be shown as in (2). In order to depict the relationship of speed and angle variant as in SVIS, we can assume that "the higher speed will produce narrower angle of the arc and vice versa". Therefore, although not same but similar shape can be generated by assuming "higher the speed produces shorter frontal length and vice versa". The assumptions led to the idea of the modified width vector formula (3) that in conjunction with the said relationship between the agent's frontal length and its speed. To create more natural dynamic size changes in quadratic manner, the divisionCoefficient with the power of two is applied. Hence, the rightLocation and leftLocation can be determined as in (4) and (5).

widthVector.x = predictVector.y (2)
witdhVector.y = predictVector.x

WidthVector .x = predictVector.x / (divisionCoefficient^2) (3)
WidthVector .y = predictVector.y/ (divisionCoefficient^2)

RightLocation.x = predictLoc.x + widthVector.x (4)
RightLocation.y = predictLoc.y – widthVector.y

LeftLocation.x = predictLoc.x – widthVector.x (5)
LeftLocation.y = predictLoc.y + widthVector.y

The construction of the field of perception will result four linear lines that will be used for collision testing process. These four lines are, the line that represent in correspond with the agent's future position is from the *location to perceptionLoc*. On the other hand, the triangle lines are represented into frontal rays which are from *leftLoc to RightLoc*, Right Rays is from *location to rightLoc* and Left Rays is from *location to leftLoc*.

6 Collision Detection

The next process after constructing field of perception is collision detection. In this process we are testing the triangle perception with the single ray cast that represent as an obstacle as shown in figure 2. In order two simulate the ambiguity of agent's perception as in what in real world situation, there will be no testing between two triangle perceptions and similar between two predicted rays. The collision test can be simplified as in table 1.

TABLE 1. COLLISION TESTING BETWEEN PERCEPTION AND PREDICT RAY

Main Object	Collision Testing Object	Test Validity
Perception	Predict Ray	True
Predict Ray	Perception	False
Perception	Perception	False
Predict Ray	Predict Ray	True

Similar to STP, we can also segregate the triangle becoming similar to spatial partitioning within the perception by testing the collision as a three different rays which are frontal, left and right rays instead of single unit triangle primitive. Each of the triangle rays can be tested with the predict rays. The simplest algorithm to test triangle rays and predict rays is as follows;

```
result1 = line2lineTesting(leftLoc, rightLoc, locationRay, predictLoc);
result2 = line2lineTesting(locationTri, leftLoc, locationRay, predictLoc);
result3 = line2lineTesting(locationTri, rightLoc, locationRay, predictLoc);

if(result1 == intersected || result2 == intersected || result3 == intersected){
    agent status = "INTERSECTED!!";
}
```

By implementing velocity perception, each agents will generate more varieties of field of vision shapes thus this will give more different reaction toward agent's collision response. Agents with more speed will have more focal length and thus will be the one who react first instead both which traditionally have similar length. Similar to the real world situation, pedestrian reaction based on their perception, thus the reaction may be varies where some may response to avoid and some may just continue their current locomotion traits. Figure 4 and 5 demonstrate the scenarios of velocity perception.

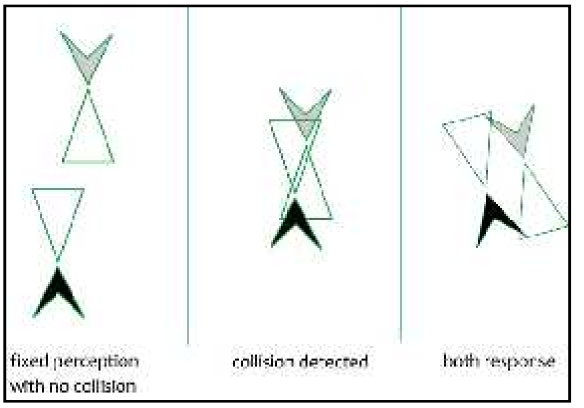

Fig. 4. Collision Avoidance using Fixed Perception.

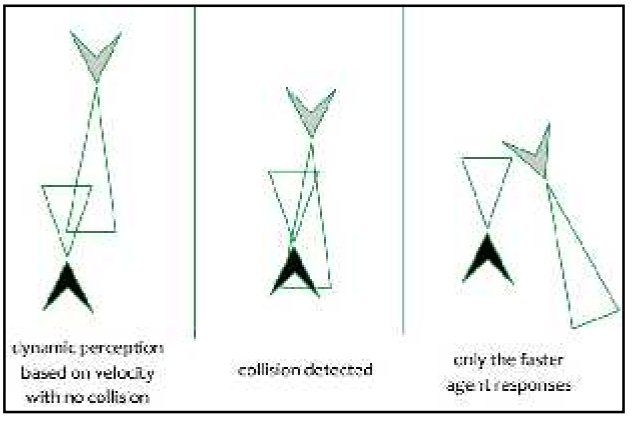

Fig. 5. Collision Avoidance using Velocity Perception.

7 IMPLEMENTATION AND RESULTS

The proposed perception model is validated through a series of experiments. The velocity perception is tested in crowd simulation environment such in "Flocking" simulation. The simulation was developed in the continuation project of Daniel Shiffman in "Nature of Code" using Processing[21]. In order to make triangle-ray cast collision testing more efficient, we applied simple broad phase and narrow phase collision testing concept. We applied sphere bounding volume to each agent's as broad phase collision testing. This way, broad phase collision detection has filtered out non-colliding pairs so that triangle-ray cast collision test needs only to be performed. The simulation runs on Windows 7, with Intel Processor i5 Core with CPU of 2.67Ghz and 4GB of memory.

Figure 6 shows ray casting as an agent's future position. The length of the ray is based on the linear interpolation of agent's current velocity. The agent's future movement can be represented in more dynamic and more variety length instead of having a fixed length. This enhances the possibilities of collision detection and collision response to be more dynamic. Figure 7 shows Dynamic Field of vision that is represented in triangle bounding volume form. Similar to the basic principle of constructing ray-casting for the agent, the construction of field of vision is based on the agent's current velocity as well. Thus, it also creates more dynamic size that showing the agent's focus area that to be tested with other agent's ray casting.

The combination of dynamic ray casting and dynamic field of view will increase the variation of collision detection and collision response for collision avoidance behavior. Figure 9 shows more individualistic collision avoidance behavior due to the implementation of dynamic field of view. In comparison of figure 8, the collision avoidance behavior traditionally depends on separation behavior which is more unison and lack of dynamic in term of executing the collision avoidance behavior.

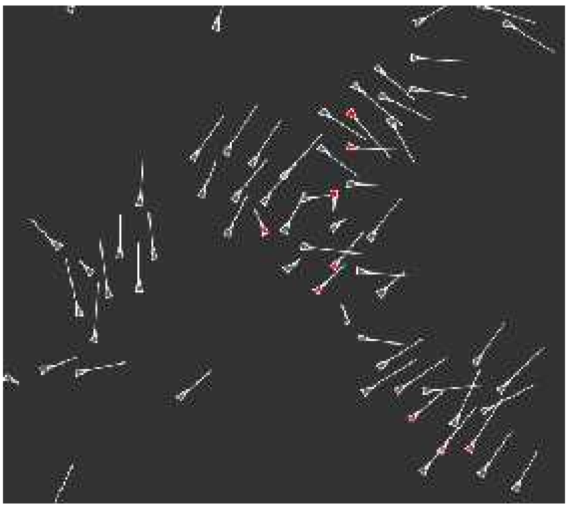

Fig. 6. Ray Casting as a linear interpolation to represent the agent's predicted position for particular time-steps.

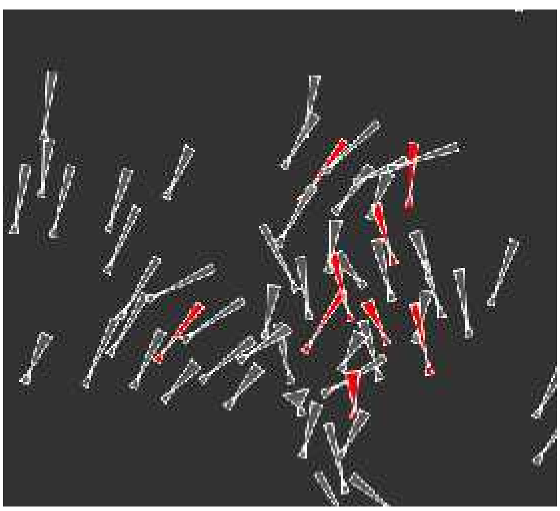

Fig. 7. Field of Perception are being tested with Ray casting. Red indicates intersection and thus trigger collision response.

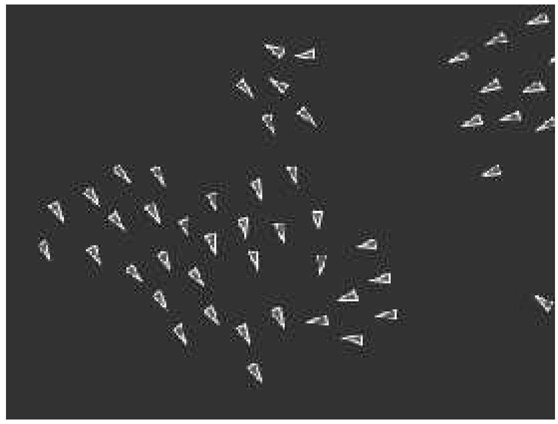

Fig. 8. Normal Flocking Behaviour.

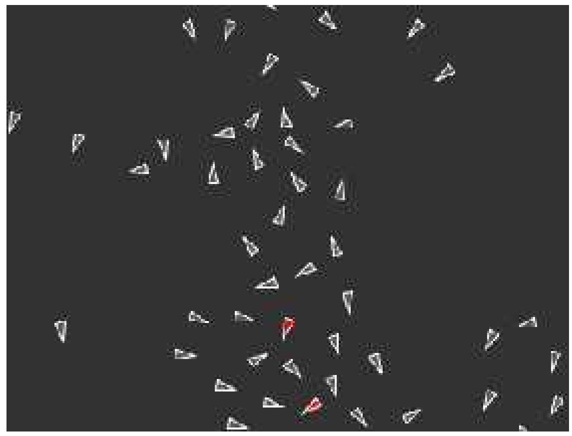

Fig. 9. Flocking Behaviour with Dynamic Field of Perception.

8 Conclusion and Future Works

In the paper we proposed a collision handling method that resembles the pedestrian field of view that changes depending on their locomotion variable. The locomotion variable will affect the agent's width and its focal length. Therefore, the agents will have more perception options so that the collision avoidance behavior may be triggered more differently and in more depth. The construction of velocity perception that was based on three linear line that form as triangle bounding volume will give more possibilities of more different ways of collision avoidance reaction for each linear line intersected.

As far as the implementation on "Flocking" is concern, Velocity Perception show the ability to preserve free collision between each agent. Similar to real situation, pedestrian may have some sense of prediction of other pedestrian movement within its perception. Therefore, velocity perception also acts as collision detection bounding volume and it is tested between single-ray that represents as agent's projected movement for certain time-frame ahead.

Although our current method is focusing on velocity perception as collision detection, it can be expand into one complete collision avoidance behavior by implementing our own collision response in the future. The efficiency of velocity perception can also be improved by implementing KD-tree as broad phase collision detection. Moreover, the simulation can also be expanded into 3D environment to simulate more realistic view and depth

ACKNOWLEDGMENT

We would like to extend our gratitude to our Universities and all parties involved who contribute to the success of this research.

REFERENCES

[1] D. Thalmann and S. R. Musse, “Crowd Simulation,” 2013.
[2] N. A. B. . Abdullah, A. Bin Bade, and S. Kari, “A Review of Collision Avoidance Technique for Crowd Simulation,” in International Conference on Information and Multimedia Technology, 2009. ICIMT ’09, 2009, pp. 388–392.
[3] N. Pelechano, J. M. Allbeck, and N. I. Badler, “Virtual Crowds: Methods, Simulation, and Control,” Synth. Lect. Comput. Graph. Animat., vol. 3, no. 1, pp. 1–176, 2008.
[4] J. S. Milazzo, N. M. Rouphail, J. E. Hummer, and D. P. Allen, “Effect of pedestrians on capacity of signalized intersections,” Transp. Res. Rec. J. Transp. Res. Board, vol. 1646, no. 1, pp. 37–46, 1998.
[5] S. Hoogendoorn and P. H. Bovy, “Gas-kinetic modeling and simulation of pedestrian flows,” Transp. Res. Rec. J. Transp. Res. Board, vol. 1710, no. 1, pp. 28–36, 2000.
[6] D. Thalmann, S. R. Musse, and M. Kallmann, Virtual Humans’ Behaviour: Individuals, Groups, and Crowds. .
[7] S. R. Musse, B. Ulicny, A. Aubel, and D. Thalmann, “Groups and crowd simulation,” 2005, p. 2.
[8] Chunlin He, He Xiao, Wen Dong, and Liping Deng, “Dynamic group behavior for real-time multi-agent crowd simulation,” in 2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE), 2010, vol. 1, pp. 544–546.
[9] I. Karamouzas and M. Overmars, “Simulating the local behaviour of small pedestrian groups,” 2010, p. 183.
[10] D. Helbing and P. Molnar, “Social force model for pedestrian dynamics,” Phys. Rev. E, vol. 51, no. 5, p. 4282, 1995.
[11] J. Dijkstra, J. Jesurun, and H. Timmermans, “A multi-agent cellular automata model of pedestrian movement,” in Pedestrian and Evacuation Dynamics, 2002, pp. 173–180.
[12] C. W. Reynolds, “Flocks, herds and schools: A distributed behavioral model,” 1987, pp. 25–34.
[13] C. Reynolds, “Steering Behaviors for Autonomous Characters,” in Game Developers Conference 1999, San Jose, California, 1999, pp. 763–782.
[14] W. L. Koh, L. Lin, and S. Zhou, “Modelling and Simulation of Pedestrian Behaviours,” 2008, pp. 43–50.
[15] I. Millington and J. Funge, Artificial Intelligence for Games, Second Edition. Morgan Kaufmann Publishers Inc., 2009.
[16] S. J. Guy, J. Chhugani, C. Kim, N. Satish, M. Lin, D. Manocha, and P. Dubey, “ClearPath,” 2009, p. 177.
[17] J. van den Berg, M. C. Lin, and D. Manocha, “Reciprocal Velocity Obstacles for Real-Time Multi-Agent Navigation,” in IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2008, pp. 1928–1935.
[18] J. Ondřej, J. Pettré, A.-H. Olivier, and S. Donikian, “A synthetic-vision based steering approach for crowd simulation,” ACM Trans Graph, vol. 29, no. 4, pp. 1–9, 2010.
[19] N. Hu, M. Lees, and S. Zhou, “Pattern based motion for crowd simulation,” in Transactions on edutainment VI, Springer, 2011, pp. 99–110.
[20] J. H. Park, F. A. Rojas, and H. S. Yang, “A collision avoidance behavior model for crowd simulation based on psychological findings,” Comput. Animat. Virtual Worlds, vol. 24, no. 3–4, pp. 173–183, 2013.
[21] D. Shiffman, “The Nature of Code,” Chapter 6. Autonomous Agents, 2012.