# Virtual Ethology: Simulation of Aquatic Animal Heterogeneous Behaviours as Particle-Based Autonomous Agents

Chen Kim Lim, Kian Lam Tan
*Faculty of Art, Computing & Creative Industry (FSKIK),*
*Sultan Idris Education University (UPSI), 35900 Tanjung Malim, Perak.*
*kim@fskik.upsi.edu.my*

***Abstract*—In the virtual world, the simulation of flocking behaviour has been actively investigated since the 1980 through the boid models. However, ethology is a niche study of animal behaviour from the biological perspective that is rarely instil in the interest of the younger learners nowadays. The keystone of the research is to be able to disseminate the study of animal behaviours through the boid model with the aid of technology. Through the simulation, complex movement of animal behaviours are reproduced based on the extension of basic behaviours of boid algorithm. The techniques here are to (i) Analyse a high-level behavioural framework of motion in the animal behaviours and (ii) Evolves particles to other animal representations to portray more real-time examples of steering behaviours. Although the generality of the results is limited by the number of case study, it also supports the hypothesis that interactive simulation system of virtual ethology can aid the improvement of animal studies.**

***Index Terms*—Virtual Ethology; Fish Simulation; Boid Algorithm; Heterogeneous Behaviours.**

## I. Introduction

Although people have long been fascinated by the behaviour of animals, the formal discipline of animal behaviour: ethology is actually relatively new [1]. An understanding of animal behaviour can be useful when educating young learners about ecosystem and how to live harmoniously with animals. With a goal of maximizing the learning theories of species-specific behaviours, a simulation system is vital for illustrating some of the animal behaviours in a more interactive and eye-catching manners for young learners. Besides, this research is intended to help young learners to gain some basic understanding and be more familiar with the way animal behaves using the aid of technology which guided by the particle-based algorithms that are well-suited to define the characteristic of animal behaviours. Nonetheless, through a computer model of coordinated and animated animal motions, young learners can observed how the animals manoeuvres, learn about individual and group motion and complex system based on models with regards to the study of animal behaviours and ecological issues. Thus, this simulation could be a source of fascination not only for children, but also for the naturalists and artists.

Therefore, this research aims to analyse a high-level behavioural framework of modelling motion and simulating physics of underwater world animals by plausibly inspecting their common interactions and coordination. This simulation also evolves particles sitting on screens that flop around when affected by forces in the environment to other animal representations to add interest in the study of animal behaviours. At the end of the simulation, several tested hypotheses will be carried out such as (i) animal that lives individually is observed to have very high separation and containment forces and giving low-priority to other forces such as cohesion and alignment in which animal that lives in a group giving high-priority on and (ii) there is a positive correlation between the improvement of learning theories of animal behaviours and the interactivity of the simulated system.

## II. State Of The Art

Flocking is a computer model for the coordinated motion of groups of entities called birds. The first flocking algorithm dated back in 1986 was developed by [2], and it has been widely used until today to develop other application such as computer animation, social network simulation and modelling opinion flow. Tim Burton's Batman Returns (1992) [3] is widely quoted as having been nominated for an Oscar for its bat swarms which were procedurally generated using boid algorithm. [2] Has described the conduct of separation, alignment and cohesion steering forces while [4] proposed the homing, velocity regulation and interaction models for simulating flocking behaviours. [5] Later focussed on combined behaviours and groups such as crowd path following, leader following, unaligned collision avoidance and queuing behaviours. In an online interactive simulation called Massive Battle, complex movements of platoons of soldiers engaging in a battle is accomplished through the extension of the boids algorithm by [6].

There were many related research done such as Prey-Predator model [7], automatic animation system for jellyfish [8] and fish-like boid simulator [9]. On one hand, [10] has further elaborated the boid equation with population-based stochastic optimization algorithm where a new variants which are the divergent and convergent models that are employed to ameliorate the performance. On the other hand, [11] has also proposed to improve the optimisation problems in simulating a fish of school by using Particle Swarm Algorithm (PSO) and Genetic Algorithm (GA). Adding moving vector coefficients to the population based on heuristic search techniques resulted in realistic interactions. As the derivation of the new proposed method is based on rule-based boid algorithm [2], [5] finding the balanced weights and coefficients values of the behavioural rules has always been a challenge. [12] Proposed to use the average nearest-neighbour distance to determine the density and size of schools of fishes. Besides, for simulation of jelly fish, [13] proposed to couple the dynamic swimming movement with the behaviours of fluid flow as the governing rules in the simulation. Therefore, there is a need for more complex steering behaviours that provide a high degree of control fidelity in order to enter a new age of interactive virtual world applications [14].

## III. Research Methodology

There are many models of design instruction that can be used as a guide in carrying out the research. Figure 1 shows the research methodology of the simulation of animal behaviours, which intends to facilitate further the processes to execute this research in more easily, and systematically manner. In the first phase, multiple data sources have to be collected including observation data of animal behaviours such as finding food and feeding, swimming style, aggression and defence, migration and possibly communication and courtship. Then, the common behaviours are filtered through a thorough analysis of the recorded animals' motion clips. Besides, particle-based boid algorithm study and analysis are carried out. In term of the algorithm studied, steering forces such as containment, obstacle avoidance, cohesion, alignment and separations are concern to be the common patterns of behaviours by the animals that live dominantly or in a group.

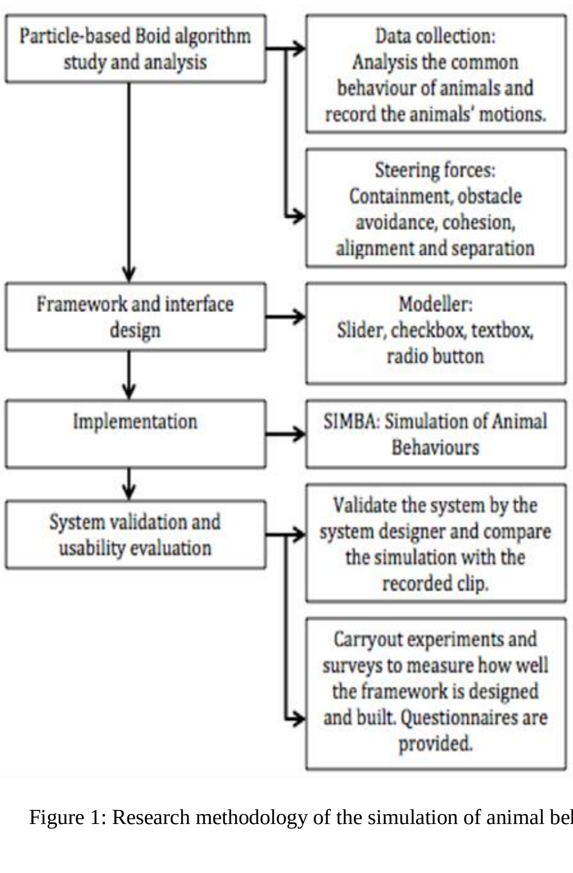


Figure 1: Research methodology of the simulation of animal behaviours

In the second phase, a framework for an interactive interface is designed. The problem to be solved is to identify the need of the targeted users who are mainly the animal learners and that must be fulfilled in producing the simulation system. By identifying the problems and objective, target user or end user and the contents or resources, the framework and interface are carefully designed with stativity and dynamicity of the environment. In this simulation system, various user interfaces are used such as slider and radio button for changes of movement and type of animals. For instance, the slider acts as the modeller of altering the parameters for alignment, separation and containment forces to influence the movement of animals. This phase also concerns about the combination colour, layout, multimedia elements, text, graphic, audio and animation that are applied in the simulation to ensure the user friendliness and the flexibility of the system. According to [11], a good layout concept will help to deliver balanced information on the interface and make good interaction between user and computer to stay better focus and attention on the framework designed.

In the third phase, a simulation system of animal behaviour (SIMBA) is developed. At this stage, additional enhancement and refinement of the models, algorithms and techniques of this research will be integrated into the system. In the final stage, the evaluation consists of two parts. The first part is about the system validation and the second part is about the usability evaluation. System validation concerns about accuracy in simulation of the system and the associated models, algorithms and techniques of this research. The usability study is to make sure the effectiveness of the simulation that involves specific design evaluation, feedbacks on contents, strategy and element multimedia using surveys and questionnaires methods. Case-by-case experiment and survey will be executed at this phase. The experiment and survey will measure how well the framework is and questionnaires are provided during testing. All constructive feedbacks are compiled for further improvement of the research.

## IV. Implementation

In the implementation stage, a framework, SIMBA is developed. At the beginning of this work, animal is taken as particles represented by colourful circles and the obstacles are rocks. It was modelled in 2D environment using OpenGL on the terminal platform of X Code. Many possible solutions were found in making the movements realistic and able to avoid static obstacles (water plants) and dynamic obstacles (other animals). However, this collision avoidance was only tested with hybrid of containment, alignment, cohesion and separation forces. These types of forces become a core part of the research where different kinds of forces are only applied each time for a particular scenario for optimization purpose. The focus of the modelling at this stage was on the types of obstacles as there are different kinds of obstacle. So the class polygon, wall and rocks were implemented with respects to the general obstacles in the shapes of any closed objects, lines and rounded objects too. Figure 2 shows the functional overview of the particle-based simulation using OpenGL through the UML diagram.

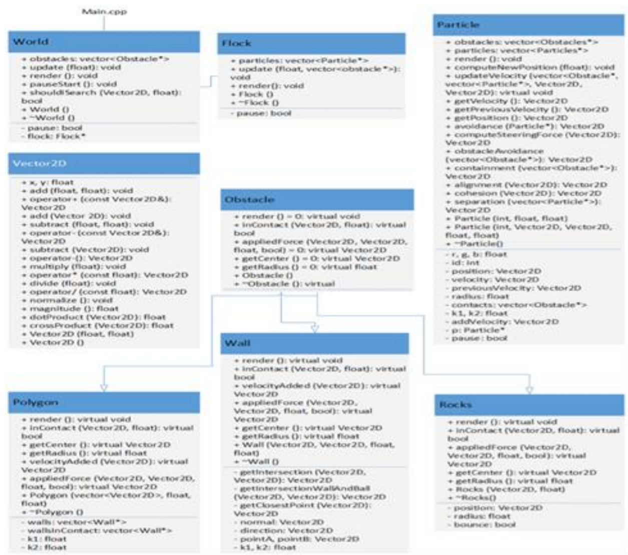


Figure: 2 UML diagram of the particle-based simulation using OpenGL

Main.cpp is the main classes which constantly updates with a predefined delta time and then render the output at each frame. The module of World.cpp is a main class that receives calling from Main.cpp. In order to generate rounded obstacles of various sizes, the initial position and radius of the obstacles are predefined in a given range. Then, each particle has a safe distance where the particle will not collide with other particle. The function should Search will determine if there is a possibility for collision before determining the next action. In order to maintain the particles in a group, the average velocity and average position are computed. As the parameters of the average velocity and average position are constantly changed, Flock.cpp is a sub-class to World.cpp. Particle.cpp is a class where the particles are acting individually when the particles are not considered as a group of flockers. The steering forces applied to each particle adopted the containment, alignment, cohesion and separation behaviours from [2], Obstacle.cpp is a main class to Polygon.cpp, Wall.cpp and Wall.cpp. Therefore, there is no function defined in this class. Besides, Obstacle.cpp stores five virtual functions that are then defined repetitively in Polygon.cpp, Wall.cpp and Wall.cpp. Although within pond as the boundary area, there is no clear representation of polygon as an obstacle, in this work, the centre position of at least three particles that are closed enough within the predefined neighbourhood radius with another particle, the latter will regard these three points as a closed polygon and thus avoiding them like a group of obstacles with steering forces. Figure 3 highlights the importance of the algorithm used for wall avoidance.

Given a scenario for example, when the particle is already hitting on the wall obstacle, it is required that the wall gives a force to the particle so that the particle can steer away in a correct direction. Based on illustration in Figure 4, the algorithm applied is as shown in Figure 5.

```
PROVIDE the direction and normal of a plane in the constructor
RENDER the lines
COMPUTE inContact: Use the middlePoint to check if the particle is 'over' the wall
horizontally and vertically
COMPUTE the closestPoint by using getIntersection function
COMPUTE the velocity to add along the normal of the plane using the dot product through
velocityAdded
COMPUTE the velocity to add along the direction of the plane using dot product
RETURN the total velocity using the coefficients
COMPUTE appliedForce: Get the closestPoint and middlePoint
SEARCH the direction of the force using the closestPoint and position of the particle
SEARCH the distance between the wall and the particle
NORMALISE the direction
IF the particle has to follow the wall
        THEN bounce will be FALSE
ELSE bounce will be TRUE
IF the particle is moving away or IF the intersection point is not on the wall
        IF the particle is in the predefined range
                IF the particle would not be hitting the wall
                        THEN do not APPLY any force
ELSE
        APPLY a force
COMPUTE getIntersection: Compute the intersection with the wall from the position and
the direction using cross product
```

Figure 3: Algorithm at Wall.cpp

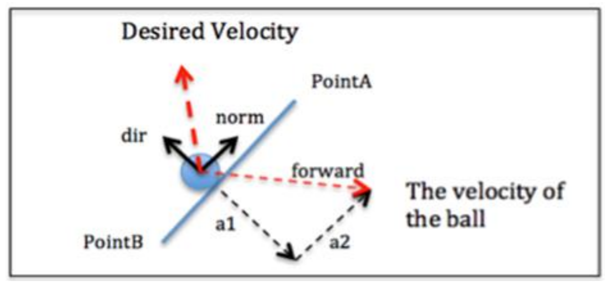


Figure 4: An Illustration of the particle hitting the wall

```
COMPUTE norm = (PointA − PointB).normalized
COMPUTE dir = (norm.y, − norm.x)
COMPUTE dist = forward.magnitude
COMPUTE a1 = dist.dotProduct(dir)
COMPUTE a2 = dist.dotProduct(norm)
COMPUTE dv = −a1∗dir∗k1 + a2∗normal∗k2
COMPUTE Steering force = dv − forward
```

Figure 5: Algorithm to avoid wall obstacle

When the obstacle is a rounded entity, this implementation avoids the obstacles like avoiding another particle as illustrated in Figure 6. By subtracting the position between two particles and normalising it, the direction and the distance or magnitude of the two objects (One flocker and one rock) are obtained. If the distance is more than RockRadius + flocker- Radius + 2*Range, the repulsive force will be applied. The forward vector of the particle will give the norm (x1, y1). By negating (y1, -x1), the right vector is obtained. The ultimate goal is to steer the flocker with right vector * scalar. The scalar which is the force which is inversely proportional to the range has to be obtained. The reason of using the right vector is that if the particle is moving into the direction of the rock for example, with a force, the particle will jerk and retreat backward and this is certainly an unrealistic move for an agent later on. The particle can sway a little with the force to look more realistic. The force is applied based on the algorithm as shown in Figure 7.

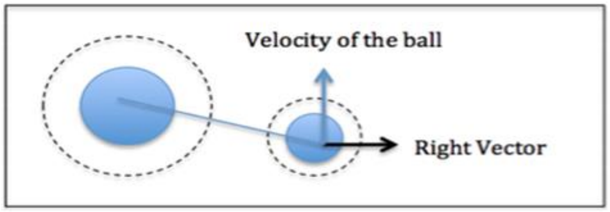


Figure 6: Collision detection and avoidance between two rounded objects

```
CHECK_IF two rocks are in contact or not
        RENDER the rocks
SEARCH the length through dot product between the direction and the velocity
SEARCH another length through dot product between the normal and the velocity
STEER the particles go to the right or left through the new force depending on the normal
direction IF it is positive or negative
```

Figure 7: Algorithm to Rocks.cpp

Dynamic rounded obstacles are mainly the particles themselves. The particles have to be ensured that they are not in contact all the time so that the avoidance can be executed in advance before the collision takes place. The result of applying the algorithm as shown in Figure 7 is that one particle will continue with its current goal while the other particle that is almost colliding with the former will steer a little to either the left or the right without any jerking.

## V. System Validations and Discussions

### *A. A High-Level Behavioural Framework*

There are three distinct ways in computing the global force based on various steering forces in order to bring the life to the animal that portrays adequate heterogeneous behaviours. The methods mentioned are (i) competitive method (the method applies all the forces at the same time but each force is given a certain weight so that certain forces will be applied more often. This is often called as direct control where only one type of action will be displayed per frame.), (ii) direct alternate method (a random number between 0 and 1 is generated and each of the forces is also given a number between 0 and 1. If the number generated exceeds the weight of the force, the force will not be applied and vice versa.) and (iii) Monte Carlo method (the method picks a random number between 0 and the sum of the total weight of the forces and applies only one force that corresponds to the generated random number and thus, less computational time is required).

For the high-level behavioral framework, the competitive and direct alternate method have been tested and finally the traditional Monte Carlo method [15] is applied with an adaptation of arbitrary maximum computational force of 3.2 where containment is 0-0.6, alignment is 0.6-1.3, avoidance is 1.3-1.5, cohesion is 1.5-2.8 and separation is 2.8-3.2. The greatest challenge here is in adapting the computational forces into this simulation and tuning the right forces for each type of forces based on their priorities.

### *B. Particle Flexible Evolvements*

In this preliminary simulation, we have only managed to involve the animals as fishes from particles to portray more real-time examples of steering behaviours. Figure 8(a) shows the fishes with 100% of containment forces and ignoring the rest of the forces while Figure 8(b) shows a great adjustment to other forces. Both scenarios are validated and the changes in percentage of forces applied changes the behaviours of the animals accordingly.

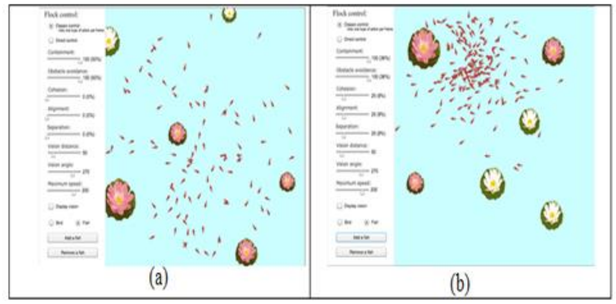


Figure 8: Example of particles evolvement from particles to fishes (a) With only the containment forces (b) With the involvement of other forces

## VI. Conclusion

As a conclusion, this paper has shown the heterogeneous behaviours of the animals in the context of virtual ethology with simulation of up to 500 fishes that could display a variety of individualised behaviours. The heterogeneous behaviours include the classical alignment, cohesion, separation and containment movements in a pond. These animals are simulated in real-time as particle-based agents. On top of these behaviours, a combination of its steering forces is able to portray the aquatic animals to be living in a group or individually. The next direction would be carrying out thorough user evaluation of how this simulation affects the young learners in the learning theories of virtual ethology.

## Acknowledgment

This work presented is funded by University Research Grant (No. 2015-0121-109-01). Thanks to Quentin Galvane for the gradual help in deploying the final framework.

## References

[1] Mench, J. “Why is it important to understand animal behavior?” *IJAR Journal*. Vol. 39(1), pp. 20–26, 1998.

[2] Reynolds, C. W. “Flocks, herds, and schools: a distributed behavioral model”. ACM SIGGRAPH ’87 *Conference Proceedings on Computer Graphics.* Vol. 21(4), pp. 25 – 34, 1987.

[3] Returns, B., Burton, T., Kane, B., Waters, D., Keaton, M., Devito, D. and Pfeiffer, M. Batman Returns. 1992; IMDb

[4] Heppner, F. and Grenander, U. “A stochastic nonlinear model for coordinated bird flocks”. *The Ubiquity of Chaos*, 1990, pp. 233–238.

[5] Reynolds, C. W. “Steering behaviour for autonomous characters”. *Proceedings of Game Developers Conference*; 1999, pp. 763 – 782.

[6] Boccardo, A., Chiara, R. D. and Scarano, V. Massive. “Battle: coordinated movement of autonomous agents”. *CASA Workshop on 3D Advanced Media in Gaming and Simulation*. 2009, pp.40 – 47.

[7] Plotnik, A. and Rock, S. “Hybrid estimation using perceptional information: Robotic Tracking of Deep Ocean Animals. *IEEE Journal of Oceanic Engineering.* Vol. 36(2), pp. 298-315, 2011.

[8] Rudolf, D. and Mould, D, *An interactive fluid model of jellyfish for animation 2010*. VISIGRAPP 2009, CCIS 68, *Springer-Verlag Berlin Heidelberg. pp.* 69-72.

[9] Sugino, R., Kobayashi, M., Fukda, K. and Morizumi, N. “Development of fish-like boid simulator based on biological code of behavior”. pp. 2899-2902, 2011.

[10] Cui, Z. and Shi, Z. "Boid particle swarm optimisation". *International Journal of Innovative Computing and Applications*. Vol. 2(1), pp. 77–85, 2009.

[11] Alaliyat, S., Yndestad, H. and Sanfilippo, F. "Optimisation of boids swarm model based on genetic algorithm and particle swarm optimisation algorithm (comparative study)". *Proceedings 28th European Conference on Modelling and Simulation (ECMS*); 2014

[12] Hemelrijk, C. K. and Hildenbrandt, H. "Schools of fish and flocks of birds: their shape and internal structure by self-organization". *Behavioural Ecology and Self-Organisation*. vol. 730, pp. 726-737, 2012.

[13] Inomoto, T., Matsuno, K., Yamakawa, M., Asao, S. and Ishihara, S. "Numerical simulation of flows around jellyfish in a current. *The International Conference on Cognitive Modelling (ICCM)*. pp. 1-12, 2015.

[14] Kapadia, M., Shoulson, A., Boatright, C. D., Huang, P., Durupinar, F. and Badler, N. I. 2012. "What's next? The new era of autonomous virtual humans". *Motion in Games*., pp. 170–181, 2012.

[15] Van den Berg, J., Patil, S., Sewall, J., Manocha, D. and Lin, M. "Interactive navigation of multiple agents in crowded environments". *Proceedings of the Symposium on Interactive 3D Graphics and Games*. 2008, pp. 139 – 147.