Schooling Behavior in Fish

An effort to simulate the schooling behaviour of fish in presence of external agents such as food and obstacles

Ruchika
Department of Electrical and Computer Engineering
University of Waterloo
Waterloo, Ontario, Canada
r2ruchik@uwaterloo.ca

***Abstract*—The animal kingdom is a diverse and vibrant place and the way animals move, look, and evolve is often taken and adapted by experts for their products and developments, for example, Boston Dynamic's big dog, helium-filled robots built by Festo that can swim in the air, and bats are even inspiring the creation of new drones. Another big example is that Scientists at Georgia Tech recently 3D-printed a cat's tongue to learn about the sharp "spines" on its surface that help the animals brush their fur. It was discovered that cat tongues are self-cleaning, as it is easy to remove the hair beneath the spines by brushing the tongue from tip to end. The findings could be used in soft robotics as researchers are struggling to find ways for soft materials to grip surfaces and will also be used to develop a new hairbrush for human grooming [4]. Similarly, one very impressive thing is how some animals seem to be able to organize themselves into larger groups so effortlessly: birds, sheep, fish, swarms- among others. If we can understand the mechanism of the emergent behavior of these different organized species, we might be able to apply the same mechanism to our problems and achieve what we have not achieved so far. But, one of the first stepping stones to get to an understanding of flocking behavior in animals is to be able to simulate it [11]. This paper explores one such algorithm that exhibits similar collective behavior found in fish. Since the flocking algorithms are simulated by the designers having expertise in the computer domain, it becomes difficult for nonprogrammers to change the parameters and observe the required behavior. This algorithm is simulated keeping in mind the user-friendly interface. The complex interaction of simple behaviors of independent fish results in aggregate synchronized motion of fish. Each fish navigates following some simple rules that drive the motion. The rules are followed based on the perception of the immediate local neighbors within the school. There is no centralized control. The rules in the algorithm have been coded in C sharp and the simulation has been implemented in the virtual environment on the Unity platform.**

***Keywords*—aggregate motion, simulation, school, Boids, algorithm, design, collective behavior.**

I. Introduction

A. Social Behaviour in animals

The range of social behavior is best understood by considering how sociality benefits the individuals involved. Because interacting with other individuals is inherently dangerous and potentially costly, both the costs and benefits of social behavior and the costs and benefits of aggregating with others play a role in the evolution of aggregation. Aggregation provides individuals with increased access to food through information sharing and cooperative defense against non-group members as it leads to more efficient detection of predators. Aggregation is advantageous due to the energy saved by huddling during cold weather, increased survival through group defense, or increased ability to acquire, hold, and make efficient use of resources. Animals may aggregate to each other, by mutual attraction to limited resources, or as a side effect of having hatched from eggs laid together in a clutch. In some cases, more than one mechanism of attraction is involved. For example, bark beetles (family Scolytidae) form large aggregations by mutual attraction to the bark of a fallen log and also to the odors of other members of their species [14].

B. Introduction to Flocking

"Flocking" is the collective motion by a group of self-propelled entities and is a collective animal behavior exhibited by many living beings such as birds, fish, bacteria, and insects. It is considered an emergent behavior arising from simple rules that are followed by individuals and does not involve any central coordination [23]. We are all surrounded by collective behavior, birds in the sky migrating together in search of food, a massive shimmer of fish swim in unison in the aquarium, a cloud of bees move following their queen, herds of land animals during migration In each of these cases, individuals move through their environment and respond to threats and opportunities almost simultaneously, forming an an undulating enclave that seems to operate as a single entity [7]. Such coordinated behavior demands rapid and efficient communication among individuals, but interpreting exactly how this information is transferred through the group has been no easy task for the scientists.

C. Flocking or Schooling

Flocking is the term for this spectacular phenomenon, though schooling is the term that often comes into play while describing the same phenomenon occurring in fish. The term flock would apply to any aggregation of homogeneous species in this paper, size or density factors not being taken into consideration. The term 'flock' is used interchangeably with 'schooling behavior', however, it's highly possible that every flocking species has developed specific strategies that are accustomed to its context. The behaviors may seem simple but it evolves with the complex interaction between the species [25].

D. Introduction to Schooling in Fish

Fish schooling and aggregation behaviors are some of the most prominent social and group activities exhibited by fishes. Fish may school or form aggregations for many reasons, including foraging, reproduction, and defense from predators. One of the most enduring hypotheses regarding fish schooling is that fish in schools can obtain a hydrodynamic advantage, thus reducing the cost of locomotion, by taking advantage of the wakes shed by neighbors within the school [9].

Fig. 1. Example of Schooling behavior exhibited by fish [24]

E. Why Simulation?

Figuring out the coordinated synchronized movement that relies on the unbelievable communication between the entities is a challenging task. Since the cues that drive the schooling behavior in fish occur at lightning speed and these cues come from different directions and that too underwater, studying this behavior in fish has been challenging to an incredible extent. But now scientists are getting some insight into collective behavior by studying the schooling of fish [3]. Therefore, one of the first stepping stones to get to an understanding of flocking behavior is to be able to simulate it. Computer animators have tried to invent wholly new types of abstract motion and to duplicate (or make variations on) the motions found in the real world [11]. One such successful simulation was done by Craig Reynolds in 1986, where he developed a computer program with the name- "Boids" that exhibited flock-like behavior. The word "Boid" is the shortened form of "Bird - oid -object", meaning bird-like object. The approach assumed that collective behavior is simply the outcome of the complex interaction between individual birds. It is still considered as one of the most popular models for simulating flocking behavior. The algorithm illustrated in this paper is an elaboration of the algorithm by Craig Reynolds. Each fish in the school is an independent actor or entity and the resulting aggregate motion is the result of a complex interaction between the entities. The decision of each entity is based on its perception of the group through their immediate neighbors [16].

This paper is organized as follows. In section II, related work of different eras and advent of high-speed GPUs leading to high-quality animation has been discussed. Section III shines some light on Reynold's work on Boid models. Section IV discusses the approach used for simulation of collective behavior including the pseudo-code for the rules of the algorithm. Section V outlines the analysis of results and the behavior obtained on changing different parameters. Section VI mentions some of the applications of the flocking algorithm. Section VII describes the scope of future work. Finally, section VIII presents the conclusion drawn from the project.

II. Related Work

The schooling behavior of fish always attracted great attention from scientists. Special studies on basic characteristics of schools and schooling mode of life of fish had been made beginning from the first half of the 20th century.

A. Initial Developments

The earlier decades had seen books by Ashby (1947), Wiener (1948) and von Bertalanffy (1968) all of which aimed at providing a framework for the study of collective behavior. Von Bertalanffy argued for the existence of general growth laws of social entities as diverse as manufacturing companies, urbanization, and Napoleon's empire. His approach was 'empirico-intuitive', in that he used observations on the growth of organisms and applied the principles to other systems [22].

Later, Nicolis and Prigogine (1977) in their 'Self-organization in Nonequilibrium Systems'. have tried to go beyond simple analogies and pin down a rigorous theory of the 'self-organized' systems. Such a theory should explain how complex structures arise from repeated interactions between the individual units. They have tried to show why certain structures are created and persist and explain the similarities between systems at very different scales and levels of biological organization. Nicolis and Prigogine were partially successful in their enterprise. They showed that many of the mathematical equations used for describing chemical reactions could equally well be considered as models of, for example, predator-prey interactions or the building behavior of termites. By applying the same mathematical models to very different systems they argued for formal correspondence, above that of mere analogy, between the modeled systems [22].

B. Obstacles

However, neither von Bertalanffy, Nicolis and Prigogine nor any of the other early pioneers of self-organization experimentally validated the models they proposed for collective phenomena in humans or animals. There was good reason for the lack of experimental testing of the theories of self-organization. For most collective phenomena involving humans, it is simply impossible to perform experiments. Even for the collective behavior of animals, such as flocking birds, the collection of field data requires a large number of cameras and sophisticated computer tracking software [22].

C. Breakthrough

However, around the time of Nicolis and Prigogine's' another member of their Brussels group, Jean–Louis

Deneubourg, began to develop and test a theory of self-organization in social insect societies. Based on the assumption that insect societies follow simple behavioral rules, Deneubourg's approach was to write down mathematical models of how colonies of ants forage and termites build mounds (Deneubourg 1977; Deneubourg and Goss 1989). By conducting laboratory experiments where the insects were constrained to relatively simple environments, Deneubourg and his colleagues were able to test and, in some cases, validate these models (Camazine et al. 2001) [22]. Using computer simulations, Deneubourg et al. (1989) showed that the complex trail networks created by army ants during a raid could be reproduced by the simple rules for pheromone laying and following found in the double bridge experiments. By manipulating the food distribution, Franks et al. (1991) showed that the model accurately predicted network structure. It is in this sense we call ant foraging self-organized: ants follow only local rules regarding the laying and following of pheromone, but the resulting trail structure is built on a scale well beyond that of a single ant [22].

D. Contribution by Couzin et al. (2002)

Fish schools also come in many different shapes and sizes: stationary swarms; predator avoiding vacuoles and flash expansions; hourglasses and vertices; highly aligned cruising parabolas, herds, and balls. These shapes are often seen on a scale that far exceeds the size or even the range of interaction of individual fish. Herring schools can consist of more than 5000 individuals spread over 700·m2. Despite the variety of shapes and motions of animal groups, many of the different collective patterns may be generated by small variations in the rules followed by individual group members. Several authors have developed computer simulation models, known as self-propelled particle (SPP) models, that attempt to capture the collective behavior of animal groups in terms of the interactions between group members [22]. It was Couzin et al. (2002) in which a model was proposed in which individual animals follow three simple rules of thumb:
(i) move away from very nearby neighbors
(ii)adopt the same direction as those that are close by
(iii)avoid becoming isolated
Each individual thus has three zones—repulsion, alignment, and attraction—which increase in size, so that individuals are attracted to neighbors over a larger range than they align, but decrease in priority, so that an individual would always move away from neighbors in the repulsion zone, keeping the repulsion and attraction radii constant [22]. Couzin et al. (2002) found that as the alignment radius increased, individuals would go from a loosely packed stationary swarm to a torus where individuals circle round their center of mass and, finally, to a parallel-group moving in a common direction. Far from requiring a distinct set of behaviors, these three very different collective patterns self-organize in response to a small adjustment to one factor: the radius over which individuals align with each other (Couzin and Krause 2003). Hoare et al. (2004) later used a similar model to explain the group size distribution they observed in laboratory experiments [22].

E. Modeling Techniques

There are different ways of modeling fish behavior such as Discrete-time models and Continuous-time models. Examples of the Discrete-time models include models created by "Couzin and coworkers", "Hemelrijk and coworkers" and "Barbaro and coworkers". Continuous-time models by researchers such as "Bertozzi, D'Orsogna and coworkers" and "Barbaro, Birnir and coworkers" have been able to provide important information which has made the modern-day simulations possible [6].

F. Modern Enhancements

In the first simulations, fish were just points in two-dimensional space which moved with constant speed. Later, the speed has been made adjustable while an average cruising speed was maintained, possibly dependent on the environmental and/or physiological state of the fish. In recent models, the two-dimensional settings have been replaced by more realistic simulations in three dimensions, although mostly with periodic boundary conditions in all directions (if the boundary conditions are specified at all). This means that fish that swim out on one side will enter again on the other side, in all directions [6]. Incorporation of boundaries as water surface and seafloor seems to be rare. We will incorporate these in our approach to explore any possible effects of these boundary conditions There are models where also the size and shape of the fish are incorporated. We will also incorporate body size in our approach.

III. Craig Reynolds' "Boids"

Since this work is an elaboration of Craig Reynold's work on 'Boids', this section discusses boids in detail. To begin with the definition, "Boids" is an artificial life program, developed by Craig Reynolds in 1986, which simulates the flocking behavior of animals especially birds. The name "boid" corresponds to a shortened version of "bird-oid object", which refers to a bird-like object. Craig Reynolds explored an approach based on simulation as an alternative to scripting the paths of each bird individually. The simulated flock was considered an elaboration of a particle system, with the simulated birds being the particles. The aggregate motion of the simulated flock was created by a distributed behavioral model much like that at work in a natural flock; the birds choose their course. Each simulated bird has been implemented as an independent actor that navigates according to its local perception of the dynamic environment, the laws of simulated physics that rule its motion, and a set of behaviors programmed into it by the "animator." The aggregate motion of the simulated flock is the result of the dense interaction of the relatively simple behaviors of the individual simulated birds [18].

To build a simulated flock, the boid model needs to support geometric flight, behaviors that correspond to the opposing

forces of collision avoidance and the urge to join the flock. Therefore, the behaviors that lead to simulated flocking are:
1. Collision Avoidance: avoid collisions with nearby flockmates [18].
2. Velocity Matching: attempt to match velocity with nearby flockmates [18].
3. Flock Centering: attempt to stay close to nearby flockmates [18].

Static collision avoidance and dynamic velocity matching are complementary. Together they ensure that the members of a simulated flock are free to fly within the crowded skies of the flock's interior without running into one another. Collision avoidance is the urge to steer away from an imminent impact. Static collision avoidance is based on the relative position of the flockmates and ignores their velocity. Conversely, velocity matching is based only on velocity and ignores position. It is a predictive version of collision avoidance: if the boid does a good job of matching velocity with its neighbors, it is unlikely that it will collide with any of them any time soon. With velocity matching, separations between boids remain approximately invariant with respect to ongoing geometric flight. Static collision avoidance serves to establish the minimum required separation distance; velocity matching tends to maintain it. [18]

Flock centering makes a boid want to be near the center of the flock because each boid has a localized perception of the world. "center of the flock" actually means the center of the nearby flockmates [11]. Flock centering causes the boid to fly in a direction that moves it closer to the centroid of the nearby boids. if a boid is deep inside a flock, the population density in its neighborhood is roughly homogeneous; the boid density is approximately the same in all directions. In this case, the centroid of the neighborhood boids is approximately at the center of the neighborhood, so the flock centering urge is small. But if a boid is on the boundary of the flock, its neighboring boids are on one side. The centroid of the neighborhood boids is displaced from the center of the neighborhood toward the body of the flock. Here the flock centering urge is stronger and the flight path will be deflected somewhat toward the local flock center. Real flocks sometimes split apart to go around an obstacle. To be realistic, the simulated flock model must also have this ability. Flock centering correctly allows simulated flocks to bifurcate. As long as an individual boid can stay close to its nearby neighbors, it does not care if the rest of the flock turns away. More simplistic models proposed for flock organization (such as a central force model or a follow the designated leader model) do not allow splits. [18]

Based on the above behaviors one can define the set of simple rules which can be applied to individual agents as depicted in Figure 2:
1. Separation: steer to avoid crowding local flockmates
2. Alignment: steer towards the average heading of local flockmates
3. Cohesion: steer to move towards the average position (center of mass) of local flockmates.

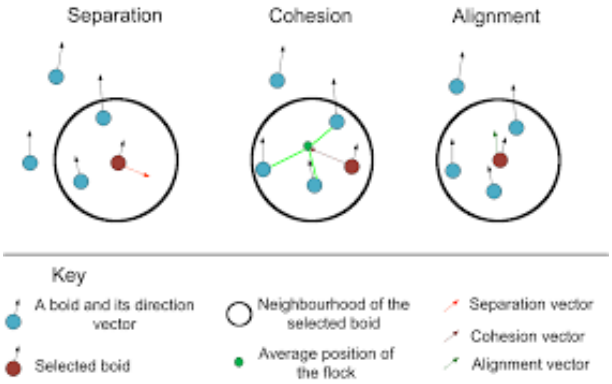


Fig. 2. Basic Rules of Boid Implementation [8]

The basic model has been extended in several different ways since Reynolds proposed it. For instance, Delgado-Mata et al [14]. extended the basic model to incorporate the effects of fear. Olfaction was used to transmit emotion between animals, through pheromones modeled as particles in a free expansion gas. Hartman and Benes [7] introduced a complementary force to the alignment that they call the change of leadership. This steer defines the chance of the boid to become a leader and try to escape. Similarly, we will be extending the boid implementation to a fish school and adding more complex, such as obstacle avoidance and goal-seeking.

The above flowchart [19] in Figure 3 gives an overall view of the Reynold's Boid model and the idea of the process that we will follow in this paper. The selection of initial velocity and initial location of each boid is done from a uniform distribution. To repeat the experiments and compare the results, a known seed is used for random number generator. All other aspects of the boids model are deterministic [13]. There is no need for Randomization. The boids could start from locations chosen on regular patterns also. Released boids at different locations begin to flock together and force their way at positions. The boids stay near one another (flock centering) but always maintain prudent separation from their neighbors' (collision avoidance), and the flock quickly becomes "polarized"-its members heading in approximately the same direction at approximately the same speed (velocity marching); when they change direction they do it in synchronization [18]. Solitary boids and smaller flocks join to become larger flocks, and in the presence of external obstacles, larger flocks can split into smaller flocks [18].

One of the most interesting behaviors of the simulated flock arises with the interaction of flock with other objects in the environment that may be dynamic. Reynold's work implements two different shapes of environmental collision avoidance- Force field concept and steer-to-avoid. Steer-to avoid is more robust. When the simulation is executed, reacting to the initial conditions is the first action of the flock. There is an initial expansion in the flock if the boids started with the positions too close to one another. There is a natural tendency to avoid a collision, which drives the boids to

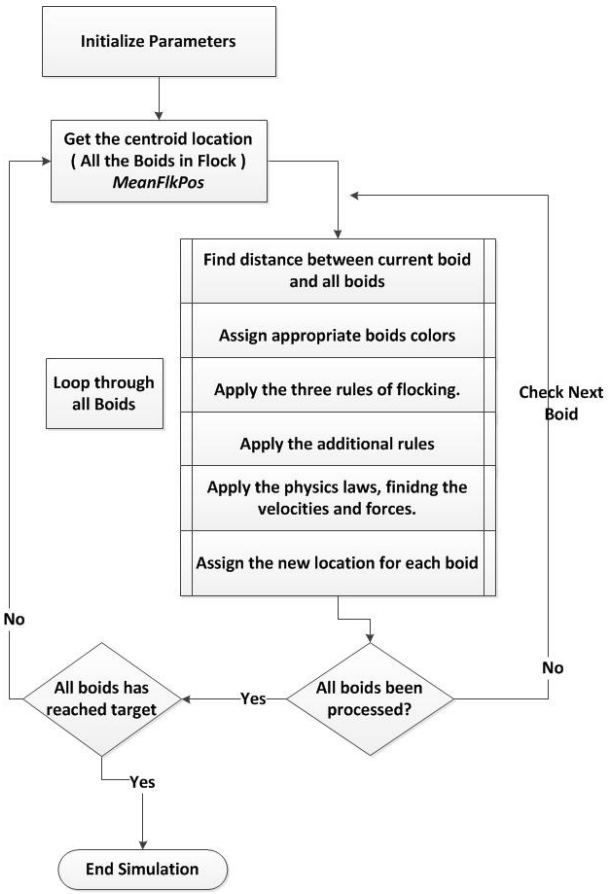


Fig. 3. Flowchart of Boid Implementation [19]

move radially outward away from the overpressured zone. If released in a spherical shell with a radius smaller than the "neighborhood" radius, the boids contract toward the sphere's center; otherwise they begin to coalesce into small flockettes that might themselves begin to join together [18]. Smaller flocks coalesce to form a single flock if the boids are confined within a region and left to wander for long enough time.

IV. Proposed Framework

The algorithm described in this section is an elaboration of Craig Reynold's work on Boids as described in the previous section. Simulation of flocking or schooling behavior of fish has been implemented in a virtual environment created using the Unity platform [10]. The Interface developed is user friendly which will enable any non-programmer biologists to alter different parameters at their will and observe the required behavior. The fish models that have been used in the project have been downloaded from the Unity Asset store [21]. These models have the basic animation of a fish body and two fins. However, it is also possible to create other fish models by customizing properties and adding the code for animation for simulation. In addition to the set of behaviors implemented by boids model- collision avoidance, velocity matching and flock centering, another set of behaviors have also been added in this proposed framework. The simulated environment created here also features some environmental obstacles to add the behavior of obstacle avoidance (Appendix B-B). These obstacles have been as 3D object situated in the normal pathway of the fish. In addition to obstacle avoidance, the fish school implements another rule of goal seeking. The goal has been shown as a 3-D sphere in the simulated virtual environment. The position of the goal changes randomly at irregular intervals between the dimensions specified. Fish moves towards the target while following other basic rules. While implementing the above 2 additional behaviors; chasing the goal avoiding obstacles the school may run out of the screen. The camera is static and cannot adapt to the motion of the school, because this implementation would require obstacles to be placed at different locations at infinite distance. To limit the fish behavior in a specified screen format, sea dimensions have been specified. The position of the goal also changes within these dimensions specified, so fish seek the goal within the dimensions specified and never runs out of the screen.

The implementation consists of two C# scripts. The IndividualFish.cs is used for specifying the behavior of the individual fish. It assigns random velocity to the individual fish of the school in the beginning and contains parameters like group center, group velocity, rotation speed and the minimum distance between two fish to avoid a collision. The set of behavior or rules has also been coded in this script. The second script, called School.cs is used to define the behavior of the complete school. It is responsible for features such as assigning random positions to the individual fish at the beginning of the simulation, the parameter of several fish in the school and the code to change the position of the goal or target randomly.

ApplyRules() is the function in which the rules have been coded, it can be applied randomly depending upon the size of the school. Although there is no need for randomness for a small-sized school but a large school if the rules are applied frequently, it will lead to arbitrary.

The pseudo-code of the ApplyRules() function has the following structure:

```
FOR EACH FISH f
    IF Distance with Neighbour Fish ≤
    Minimum Distance to become part of School
    THEN
        Make that fish member of the group
        Add up the centers of fish again

        IF Distance with Neighbour Fish ≤
        Minimum Distance for Collision
        THEN
            Change the orientation of the vector to face
            in another direction to avoid a collision
        END IF
        Add the speed of the neighboring fish to
        the group Calculate the new group speed
        by adding the speed of the neighboring
        fish
```

END IF

END

These are the few lines from code that calls the function randomly.

if(Random.Range(0,4)≤1) //Creating randomness

ApplyRules();

start() and update() are the two functions that are common to both scripts. The start is called before the first frame. In IndividualFish.cs script, it contains the random velocities of fish whereas in School.cs it defines the random positions of fish The update is used for instantiation, for example, instantiating the Prefab at the beginning (A prefab is a template for creating new instances of objects in the scene). It is called once per frame and contains the rest of the code for both the scripts.

V. Analysis

Although we have been able to simulate the schooling behavior of fishes, our main purpose would be solved when we can describe the effect of different parameters on the behavior of the school. The behavior depends on features such as Size of School, Turning Speed, Distance between Neighbors, Distance to Avoid Collision and Group Speed. In this section, we would observe the effect of these parameters. For comparing the results, these parameters would be modified in the scripts and the results will be observed and noted. For example, consider the ”DistanceOfNeighbour” variable, which defines the distance between the fish and the nearest fish in the school. Altering this parameter should change the behavior of the school drastically.

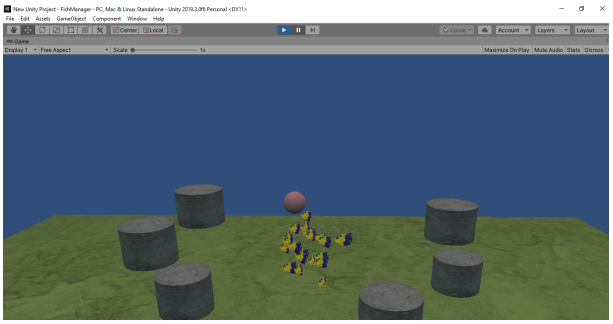


Fig. 4. Simulation of fish school with school size = 25

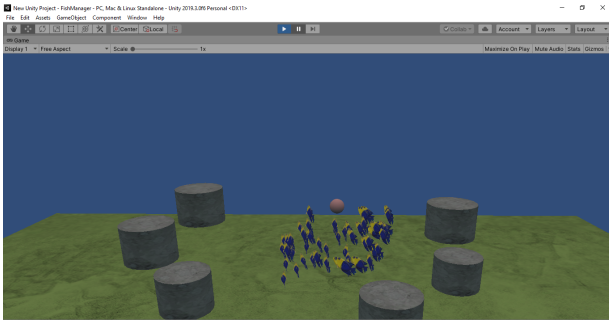


Fig. 5. Simulation of fish school with school size = 50

A. Effect of Size of School

School Size changes the schooling behavior. The school size should be increased or reduced by taking into account a few items such as the screen should be able to fit all the fish, the camera is static, keeping the fish animation of high quality. . Refer to Figure 4 and Figure 5 for simulation of school behavior with different school sizes. Now, if the size of the school has been increased, the ApplyRules() function must also be changed. It has to be called randomly otherwise it will lead to arbitrary behavior in a large group of fish within the specified size of the screen. The frequency of calling the function can be increased or decreased in the random function.

B. Effect of Turning Speed

Turning speed is the speed of rotation of the fish. In other words, the speed of rotation of the fish is the speed at which the fish turn the other way or move in a different direction to avoid collisions with other fish in the school or environmental barriers. Turn speed also comes into play when the position of the fish has to be limited to the dimensions of the camera’s view. The fish must turn back to be in the view of the camera. In our experiment, we have observed that, by increasing the speed of rotation of the individual fish, school behavior becomes absurd. The fish bounces back with a great deal of speed to bypass the fish school and override the behavioral patterns. Turn speed should be changed to keep the behavior life-like to implement the algorithm effectively.

C. Effect of Distance between Neighbors

Distance between Neighbors is the distance between a fish and the fish that is closest to it and a member of the school. This parameter sets the minimum distance between fish to be considered in school. In our experiment, we observed that no schooling behavior is observed when the distance is set too large, the fish are far away from each other and are unable to merge with the school. Only when we set the distance to some optimum value will the behavior become real-life-like. Refer Figure 6 (distance set to 3.0f) and Figure 7 (distance set to 7.0f) for the change in behavior on altering the minimum distance to merge with the school.(Appendix B-A)

D. Effect of Distance to Avoid Collision

Distance to avoid a collision in a way defines the minimum distance to be maintained between the fish to avoid a collision. Distance between neighbors is designed to close the fish to make it part of the school, while Distance to Avoid Collision sets a limit to that proximity so that collisions can be avoided. Once this distance has been reached, the fish must orient its direction to avoid a collision. Here, we have noticed that when this distance is set too high, a school is never formed, as they always stay at a greater distance from each other. Therefore,

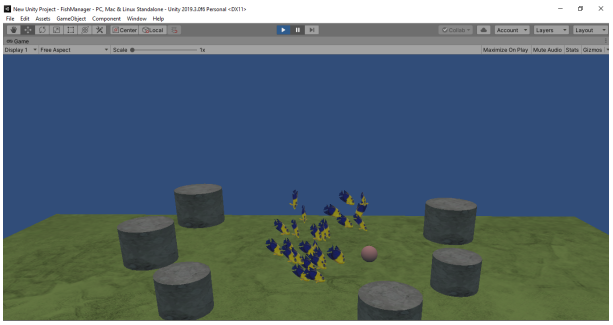


Fig. 6. Minimum distance between neighbors is 3.0f

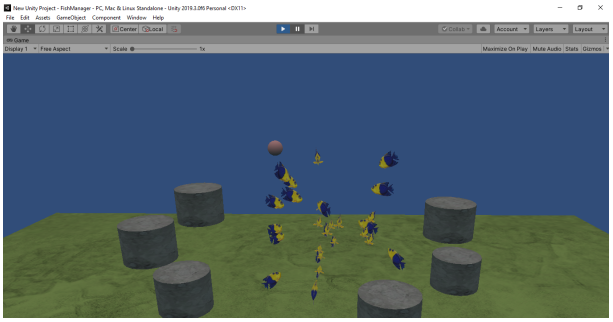


Fig. 7. Minimum distance between neighbors is 7.0f

this distance should be chosen in such a way that fish do not seem to collide with one another or with environmental barriers and that they also form a school in simulation.

E. Effect of Group Speed

The group speed of the school also has an impact on the behavior of the school. Group speed is initially set at the start of the simulation. If the speed of the group is high, schooling behavior is still observed, but it doesn't seem realistic. Fish move with so much speed that it becomes difficult to observe where the school breaks or forms or when the solitary fish joins the school. The speed of the group shall be calculated by obtaining the average speed of the fish at school so that the behaviour remains normal and observations can be made carefully.

Although there is no formal definition that defines the intended behavior of the flock but based on our experiments and observations and for the interest of a comparison, schooling of fish in aggregation can be defined as:

- All the fish are close to each other [17].
- All the fish move at approximately the same speed [17].
- All the fish move in approximately the same direction [17].

And, "good" schooling behavior can be defined as:

- The school members should not collide with each other [17].
- The school should not spontaneously lose members [17].
- The school should not spontaneously split off into new flocks [17].

VI. Applications

Flocking algorithms have applications in a wide variety of areas such as films and computer games. Often in video games, non-player characters must move in cohesive groups rather than independently. Let's consider some examples. Consider a flock of birds that prey on the game's human inhabitants. Here, birds that hunt in flocks rather than independently would seem more realistic and pose the challenge to the player of dealing with somewhat cooperating groups of predators. It's not a huge leap of faith to see that you could apply such flocking behavior to giant ants, bees, rats, or sea creatures as well. Also, this flocking behavior is not limited to fauna and can extend it to other non-player characters. For example, in a real-time strategy simulation, you can use group movement behavior for non-player unit movement. These units can be computer-controlled humans, trolls, orcs, or mechanized vehicles of all sorts. In a combat flight simulation, you can apply such group movement to computer-controlled squadrons of aircraft [2]. Reynold's model was a big stride in comparison to other traditional techniques being used in motion pictures for animation. The first animation created with the model was Stanley and Stella in Breaking the Ice (1987), followed by a feature film debut in Tim Burton's film Batman Returns (1992) with computer-generated bat swarms and armies of penguins marching through the streets of Gotham City [1]. The flocking algorithm was used for the 'Lion King' movie to animate automatically all of the herd scenes in the movie.

Additionally, flocking algorithms can be used in animated screensavers, the boids model to automatically program Internet multichannel radio stations [11], for visualizing information [15] and for optimization tasks [5]. Flocking algorithms are also used to control the behavior of Unmanned Air Vehicles (UAVs). The concept of drone swarms is becoming popular. The collective behavior of drone swarms is being fully controlled and improved. It is being developed at a rapid pace in the coming years. Fabian Schilling, Julien Lecoeur, Fabrizio Schiano, and Dario Floreano used machine learning and flocking algorithms in their work on flight drones. Their paper presented a machine learning approach to the problem of the collision-free and coherent motion of a dense swarm of quadcopters [20].

At the heart of such a group, behavior lies flocking algorithms such as the one presented by Craig Reynolds in his 1987 SIGGRAPH paper, "Flocks, Herds, and Schools: A Distributed Behavioral Model." You can apply the algorithm Reynolds presented in its original form to simulate flocks of birds, fish, or other creatures, or in modified versions to simulate group movement of units, squads, or air squadrons.

VII. Future Possibilities

Future studies should attempt to identify the interaction rules between individuals in a comparative way across species since there are fish species that differ in age, gender, response

to danger and have some other unique features that impact schooling behavior. Under different selection pressures, different interaction rules are likely to be selectively advantageous [12]. Our aim should be to compare the interaction rules adopted by different species, classify the group-level properties generated by these rules, and discuss these in the light of evolution and phylogenetic considerations. If the rules adopted by individuals are similar across distantly related species, then this will show their effectiveness in driving coordinated group movement under different evolutionary pressures [12]. On the other hand, universality in the patterns generated by groups may mean that each species has a very distinct set of rules, each leading to the same set of global patterns.

Also, More cameras can be used to observe the behavior in a wide range of areas. High-quality cameras and high-speed GPU can be used to achieve high frame rates and simulate the environment in high quality. Since the flocking algorithms are simulated by the designers having expertise in the computer domain, it becomes difficult for non-programmers to change the parameters and observe the required behavior. More User-friendly interface than the one implemented in this framework can be developed that will allow biologists to change parameters at their will and observe the required behavior

VIII. Conclusion

Many fish spend their whole life in groups. The internal organization of such groups and patterns of the relationships between their members may differ widely, depending on fish species, mode of life, age, motivation, as well as on many biotic and abiotic factors of the environment. Contacts and interactions between individuals within a group are based on various social behavioral responses. Schooling behavior is one of the most widespread forms of social behavior in fish and is observed in numerous fish species. The aggregation of fish into a highly integrated supergroup leads to an emergence of a qualitatively different behavioral pattern, very different from the behavior of the constituent individuals. The biological significance of schooling behavior is thus extremely high and concerns a wide variety of adaptive functions and attracts many applications(section VI). However, with the assistance from knowledge domain experts, the implementation of flocking algorithms can achieve new heights.

References

[1] I. L. Bajec and F. H. Heppner. Organized flight in birds. vol.78, 2009.

[2] D. M Bourg and G. Seemann. Flocking. https://www.oreilly.com/library/view/ai-for-game/0596005555/ch04.html, 2020. [Online; accessed 12-March-2020].

[3] M. Breyer. How do animals flock, school, herd and swarm as synchronized units?

[4] M. BURGESS. Amazing animals: how wildlife is inspiring science and technology. https://www.wired.co.uk/article/animal-facts-nature-science, 2017. [Online; accessed 12-March-2020].

[5] C.Zhihua and S. Zhongzhi. Boid particle swarm optimisation. *International Journal of Innovative Computing and Applications*, vol.2, 2009.

[6] L. de Boer. What makes fish school? 2010.

[7] Spine films. Lens of Time: Secrets of Schooling. https://www.biographic.com/lens-of-time-secrets-of-schooling/, 2017. [Online; accessed 12-March-2020].

[8] A. Thom N. Beume B. Naujoks H. Danielsiek, R. Stuer and M. Preuss. Intelligent moving of groups in real-time strategy games. *IEEE Symposium on Computational Intelligence and Games (CIG'08)*, 2008.

[9] Harvard. Schooling behavior in fishes.

[10] Holistic3d. Flocking fish in unity 5: Creating schooling behaviour with simple ai.

[11] Jes´us Ib´ a˜nez, Blat G´omez-Skarmeta, F. Antonio, and Josep. Djboids: emergent collective behavior as multichannel radio station programming. *Proceedings of the 8th international conference on Intelligent User Interfaces*, 2003.

[12] R. P. Mann T. M. Schaerf D. J. T. Sumpter J. E. Herbert-Read, A. Perna and A. J. W. Ward. Inferring the rules of interaction of shoaling fish. *PNAS*, vol. 105(46), 2011.

[13] K. Merrick J. Harvey and H. A. Abbass. Application of chaos measures to a simplified boids flocking model. *Swarm Intelligence*, vol.9(1), 2015.

[14] W. Koenig and J. Dickinson. Animal social behaviour. https://www.britannica.com/topic/animal-social-behaviour/The-range-of-social-behaviour-in-animalsref#1044766, 2018. [Online; accessed 12-March-2020].

[15] A. V. Moere. Time-varying data visualization using information flocking boids. *Proceedings of the IEEE Symposium on Information Visualization*, 2004.

[16] U. Erra R. De. RChiara and V. Scarano. An architecture for distributed behavioral models with gpus. *The Eurographics Association*, 2006.

[17] W. Reeves. Particle systems-a technique for modeling a class of fuzzy objects. *Computer Graphics*, vol. 17, 1983.

[18] C. W. Reynolds. Flocks, herds, and schools:a distributed behavioral model. *Computer Graphics*, 1987.

[19] S. Ganesan S. Oweis and K. C. Cheok. Server based control flocking for aerial-systems. 2014.

[20] J. Lecoeur F. Schiano Schilling, Fabian and D. Floreano. Learning vision-based cohesive flight in drone swarms. *ArXiv abs*, 2018.

[21] Unity Asset Store. Colorful Sea-Fish Pack. https://assetstore.unity.com/packages/3d/characters/animals/colorful-sea-fish-pack-12389. [Online; accessed 11-March-2020].

[22] D.J.T Sumpter. The principles of collective animal behaviour. *Philos Trans R Soc Lond B Biol Sci*, 2005.

[23] Wikipedia. Flocking (behavior). https://en.wikipedia.org/wiki/Flocking(behavior), 2020. [Online; accessed 11-March-2020].

[24] Wikipedia. Shoaling and schooling. https://en.wikipedia.org/wiki/Shoaling_and_schooling, 2020. [Online; accessed 11-March-2020].

[25] Wikipedia. Swarm behaviour. https://en.wikipedia.org/wiki/Swarm_behaviour, 2020. [Online; accessed 14-March-2020].

APPENDIX A
CODE

A. School.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class School : MonoBehaviour
{
    public GameObject Prefab; //Creating a reference to the prefab
                              // A prefab is a template for creating new instances of object in the scene


    public GameObject Prefab_for_Goal;
    public static int tankSize = 5;
    static int FishCount = 25; //Total number of fish in school
    public static GameObject[] AllFish = new GameObject[FishCount]; //To access each fish later on

    //public static float Dimensions = 5.3f;

   public static Vector3 Target = Vector3.zero; //for defining the position of the goal for fish.
                                                //Generating random behaviour in fish
                                                //Initially it is set in the middle of the tank


    // Start is called before the first frame update
    //It instantiates the Prefab at the beginning
    void Start()//creating the flock of the fishes
    {
        for (int i = 0; i < FishCount; i++)//looping through each fish
        {
            Vector3 Position = new Vector3(Random.Range(-tankSize, tankSize),
                Random.Range(-tankSize, tankSize),
                Random.Range(-tankSize, tankSize));//creating a position for the fish.
                                                   //Basically the position would be in cube of dimensions 10*10

            AllFish[i] = (GameObject)Instantiate(Prefab, Position, Quaternion.identity); //Instanting each fish
        }
    }
    void Update()
    {
        if (Random.Range(0, 10000) < 50)// For changing the position of the target randomly.
                                        //Fish move towards the target. This number can be increased or decreased.
        {
            Target = new Vector3(Random.Range(-tankSize, tankSize),
                Random.Range(-tankSize, tankSize),
                Random.Range(-tankSize, tankSize));

            Prefab_for_Goal.transform.position = Target;
        }
    }
}
```

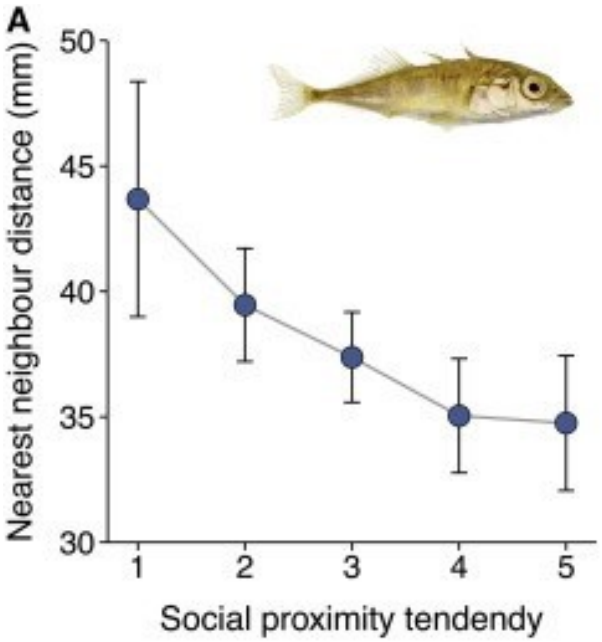
B. IndividualFish.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class IndividualFish : MonoBehaviour
{
    public float Speed = 0.7001f; //speed of the fish
    float TurnSpeed = 0.40f; // how fast fish will turn
    Vector3 AverageHeading; //Flock should head towards the average heading of the group.
    Vector3 AveragePosition; //Average position of group
    float DistanceOfNeighbour = 7.0f; //Distance between two fish to become member of the flock

    bool Return = false;//This will set to true when the fish will reach the boundary of the space defined in Dimensions variable

    // Start is called before the first frame update
    void Start()
    {
        Speed = Random.Range(0.4f, 1.0f);
    }

    // Update is called once per frame
    void Update()
    {
        if (Vector3.Distance(transform.position, Vector3.zero) >= School.tankSize)//If this is greater than the dimensions of the space, then the school should turn back
            Return = true;
        else
            Return = false;

        if (Return)
        {

            Vector3 Direction = Vector3.zero - transform.position;//Turn the other way

            transform.rotation = Quaternion.Slerp(transform.rotation,
                                                  Quaternion.LookRotation(Direction),
                                                  TurnSpeed * Time.deltaTime);
            Speed = Random.Range(0.5f, 0.9f);
        }

        else
        {
            // if(Random.Range(0,4) <1)//Not applying the rules frequentlty
            ApplyRules();
        }

        transform.Translate(0, 0, Time.deltaTime * Speed);//This moves the fish forward
    }



    void ApplyRules()
    {
        GameObject[] GameObjects;
```

```
        GameObjects = School.AllFish;

        Vector3 GroupCentre = Vector3.zero;  // centre of group and alligning it ti the centre
        Vector3 Avoid = Vector3.zero;  //avoidance vector, fish must avoid collison with someone that is near
        Vector3 Target = School.Target;//Referring to the Target of the school

        float GroupSpeed = 0.25f; //group speed

        float Distance;

        int GroupMembers = 0;//Intially size of the group is zero. We keep on adding members later to the group depending upon the distance between the fishes

        foreach (GameObject ob in GameObjects)
        {
            if (ob != this.gameObject)
            {
                Distance = Vector3.Distance(ob.transform.position, this.transform.position);
                if (Distance <= DistanceOfNeighbour)
                {
                    GroupMembers += 1; //Make that fish member of the group
                    GroupCentre += ob.transform.position; //Adding up the centres when that fish becomes part of the group

                    if (Distance < 1.5f) // Setting minimum distance for collison
                        Avoid += (this.transform.position - ob.transform.position);//This vector will face in other direction to avoid collision


                    //For calculating average group speed, we need to add the speed of the neighbouring fish to the group
                    IndividualFish Fish = ob.GetComponent<IndividualFish>();//Calculating the group speed by adding the speed of the neighbouring fish
                    GroupSpeed += Fish.Speed;

                }
            }

        }


        if (GroupMembers > 0)
        {
            //We need to calculate the group centre
            GroupCentre = GroupCentre / GroupMembers + (Target - this.transform.position);
            Speed = GroupSpeed / GroupMembers;

            Vector3 Direction = (GroupCentre + Avoid) - transform.position;//For getting the position in which fish must turn

            if (Direction != Vector3.zero)
                transform.rotation = Quaternion.Slerp(transform.rotation,
                                                      Quaternion.LookRotation(Direction),
                                                      TurnSpeed * Time.deltaTime);
        }
    }
    void CheckForObstacle()
    {
        Bounds bounds = new Bounds(_manager.transform.position, _manager.swimLimit * 2);
        RaycastHit hit = new RaycastHit();

        if (!bounds.Contains(transform.position))
        {
            _turning = true;
            _direction = _manager.transform.position - transform.position;
        }

        else if (Physics.Raycast(transform.position, transform.forward * _manager._obstacleReaction * Time.deltaTime, out hit))
        {
            _turning = true;
            _direction = Vector3.Reflect(transform.forward, hit.normal);
        }
        else
            _turning = false;


        if (_turning)
            transform.rotation = Quaternion.Slerp(transform.rotation, Quaternion.LookRotation(_direction), _manager._rotationSpeed * Time.deltaTime);
        else
        {
            if (Random.Range(0, 100) < 10)
                _speed = Random.Range(_manager.minSpeed, _manager.maxSpeed);

            if (Random.Range(0, 100) < 20)
                ApplyRules();
        }
    }


}
```

APPENDIX B
FIGURES

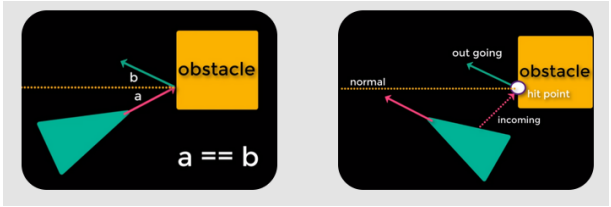
A. *Effect of distance between nearest neighbour on schooling behavior*



Source:https://ars.els-cdn.com/content/image/1-s2.0-S0960982217310138-gr2.jpg

The above graph helps us in understanding the relationship between the schooling behavior(labelled as Social proximity tendency) and the distance of the nearest neighbor to a fish. We can notice that a lower distance corresponds to higher proximity tendency.

B. *Obstacle avoidance rules*



Source: http://oliviafollin.com/projects/fish-tank.html

The above diagram illustrates basic rules that have been used to implement obstacle avoidance.Since angle of incidence is equal to the reflections angle, this helps us to calculate a new direction for our fishes.