AGENT-BASED MODELING TO SIMULATE THE MOVEMENT OF A FLOCK OF BIRDS

A Paper
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Naga Chaitanya Byrisetty

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Department:
Computer Science

October 2012

Fargo, North Dakota

North Dakota State University

Graduate School

Title

AGENT BASED MODELING TO SIMULATE THE MOVEMENT OF A FLOCK OF BIRDS

By

NAGA CHAITANYA BYRISETTY

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

SUPERVISORY COMMITTEE:

Dr. Kendall Nygard
Chair

Dr. Simone Ludwig

Dr. Saeed Salem

Dr. Sumathy Krishnan

Approved by Department Chair:

04/03/2013
Date

Dr. Brian Slator
Signature

ABSTRACT

The most beautiful, mysterious movements of bird flocks have always amused the human and led his thinking towards what makes these complex movements possible. This paper discusses the simulation of such bird behavior based on Craig W. Reynolds's work on bird flocking, satisfying three main objectives and trying to mimic the natural behavioral characteristics of the flock in various circumstances using Netlogo5.1.

- **Objective One:** To develop working software that simulates movement for a flock of birds using agent-based modeling in a two-dimensional world.
- **Objective Two:** To direct the flock towards forage grounds while sustaining the wind and obstacles affecting the flock's propagation.
- **Objective Three:** To introduce a predator into the world, study, and simulate the flock's escape behavior.

ACKNOWLEDGEMENTS

I am very grateful to my adviser, Dr. Kendall Nygard, for having confidence in me, believing in what I was working on and helping me in every possible way. Sincere thanks go to him for being patient with me and sparing a significant amount of time in spite of his busy schedule. I would like to acknowledge his invaluable support and guidance towards driving me into researching on swarm intelligence and his motivation in materializing the subject.

I would like to extend my sincere gratitude to Dr. Simone Ludwig and Dr. Saeed Salem for obliging my request. I am honored to have them on my graduate supervisory committee.

I take immense pleasure in thanking Dr. Sumathy Krishnan for her support as the adviser for one of the student committees of which I was a member. It is through her that I learned about being humble and disciplined. I am indebted to her for treating me as her family.

I need to thank the faculty members of the Computer Science Department, NDSU. It is their teachings and encouragement that helped me gain and grow. I also need to thank my teachers from undergrad college, high school, middle school, and primary school who taught me the goodness of life and made me a better person.

I am grateful for having friends like Vijay Kumar Boddu, Naresh Pillarikuppam, Keerthi Sathiraju, Manu Bhogadi, Suresh Paturu, and Muthu Kumar for their unconditional help and support throughout my career. Special thanks go to Haribabu Bavanari for helping me with the research. Finally, I am bound to my respected parents, Uma Maheswar Rao Byrisetty and Nooka Ratna Byrisetty, for teaching me good over bad in life and for the motivation and support they showed me all these years, helping me achieve all my goals. Thanks would be a small word for the kind love and affection they have shown me throughout my life.

I would like to acknowledge Uri Wilensky, Center for Connected Learning and Computer-Based Modeling at Northwestern University, Evanston, IL, for his Netlogo model on flocking which helped me understand the algorithm and apply a similar kind of rule in my paper.

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

The flocking of birds and the schooling of fish both have remarkable, coordinated movements and are among the most mysterious maneuvers. Group navigation as a single unit and instant direction changes have led researchers to hypothesize that there is an electromagnetic communication between the birds or some sort of "thought transference" involved with this kind of behavior. Evidently, this emergent behavior is not dependent on the movement of a single bird but on the property of the group itself. Several simple interactions among the neighboring birds and fish lead to these complex behaviors. Unlike large birds, such as pelicans or geese, which fly in V-formations with a leader steering the direction of the wave, no leader is involved in the flocking formation, and there is no overall control; instead, it is the time-to-time response and the individual birds' decisions which drive the flock, following simple rules in response to the neighbor interactions [1].

There are many reasons why small birds or fish have a movement in flocks or schools. One of the main reasons is to defend against predators. It would be easy to spot a predator because a large number of eyes are working together and because there is a probable chance of spotting the predator. Also, the predator might have trouble focusing on a single target, and flocks could be dangerous to the predator because it could incur injuries or fall prey to the flocks. A flock offers a coordinated defense and fends off a predator [1]. The second reason would be locating food resources, controlled navigation while returning home from the feeding area or while migrating from one place to another, and also for reproductive reasons [1].

Large birds, such as pelicans and geese, have a leader for a temporary period of time and also have an energetic benefit because the birds following the leader take advantage of air

vortexes produced by the ones ahead of them. Meanwhile, the leader, after some time, falls back to take advantage of the vortexes because it does not gain any advantage from its position [2].

A flock of birds, on the other hand, does not enjoy this energetic benefit but has the power of togetherness in defending fellow members. The birds do not have a regular rotation, but the larger and stronger members of the team are ahead most of the time [2].

Members of the flock (such as sandpipers or starlings) form a compact mass when confronting a flying predator. If the predator is a falcon and if it is to bomb into such a bunch, it would try not to, avoiding a fatal injury during the collision as well as trying to pick off the laggards or the birds flying away from the flock. The flock has sufficient ammunition to steer and turn in aberrant styles, making it difficult for the predator to predict the flock's movements [3].

"Maneuver waves" involve bird movement in all directions, including back to front and up and down. However, the flock's response depends on the birds banked in the flock instead of the ones away from it because, if the ones away from the center turn away from the flock, they have a risk of being separated and falling prey to the blood-thirsty predators; the other members would not follow those birds. Besides, flocking helps prevent flock indecisiveness and allows a rapid response to the attack [3].

When predators are not around, the flock leaves the roost site in search of a feeding area, and the flock wanders aimlessly in every direction possible because a single movement from the individual may easily steer a direction change. Eventually, a consensus would develop based on the motivation of the majority of flock members, and the flock would reach the destination in a direct manner [2].

One of the benefits of flocking is reducing the predation risk. Constant, collective surveillance by the birds; confusing the predator with rapid, cohesive behavioral movement; or thinning the risk of being a prey to the predator by forming a large-sized flock are some techniques used by the birds for anti-predation [4]. Imagine a single bird feeding and watching for a predator; the risk of being a prey is 50%. Now, imagine two birds, with one of them feeding and the other one watching for a predator, exchanging the work randomly; that scenario would contribute to a 25% chance of being at risk. Now, imagine a large flock at a feeding area; a significant number of birds will be doing surveillance while others concentrate on feeding, allowing a significant decrease in the risk to about 0. 1% [4]. Although this behavior might be a benefit, sometimes, it may lure the predator to attack the flock because the predator sees nothing but a huge chunk of food in the form of a flock, hence the predator may neglect a lone bird. However, for an above-intermediate sized flock, the birds may detect the predator easily and have a high probability of escaping, and if the birds made a swift flight following an early detection, then the predator would get surprised because it did not anticipate the sudden flight of the flock. Also, it would not be easy for the predator to concentrate on a single member of a flock which makes constant, cohesive moves.

The escape flights for the preys may vary because the more vigilant ones would escape faster than the non-vigilant ones. However, the escape flights of non-vigilant birds depend on the escape speeds of the neighboring birds because the non-vigilant birds are not likely to notice the predator approaching the flock. The nearer the neighbors are in a flock, the faster the overall speed response of the flock. The response time of a bird decreases if its distance to the nearest neighbor increases. The response time of the flock from the first detection to the final flight also depends on the flock size. The response time increases with an increased flock size. Also, in a

large flock, the birds' response time depends on the escape flights of the neighboring birds rather than detecting the predator [4].

In this paper, the simulation of such motion in a flock of birds is explained using Craig Reynolds' research and his creation of "boids" (flocking creatures). This simulation follows an object-oriented concept of determining the direction and separation, and avoiding other birds, unlike the traditional concept of programming each individual particle and its path. Each bird tries to move towards the centroid of its neighborhood which contributes to determining the direction of entire flock. Each bird follows some simple rules to form this complex movement.

Craig Reynolds' extensive research on this type of behavior led to the creation of "boids" (generic, flocking computer organisms) in 1986. Reynolds formulated three fundamental laws of flocking:

- **Separation** - steer to avoid crowding local flock mates;
- **Alignment** - steer towards the average heading of local flock mates; and
- **Cohesion** - steer to move toward the average position of local flock mates. [5]

Each bird has a certain direction and maintains a distance from the other birds. Each bird has direct access to all of the world's geometry (region in which a bird is placed), but this system requires the bird to react to the flock mates in its neighborhood. This neighborhood is determined by "distance" (like a perimeter around the bird) and an "angle" measured from the bird's direction of flight. Each bird gives itself a region to follow; other birds outside this region are ignored, and this neighborhood (region) is a model of limited perception. This neighborhood region is responsible for steering the flock [6]. Figure 1.1 shows the neighborhood of the bird.

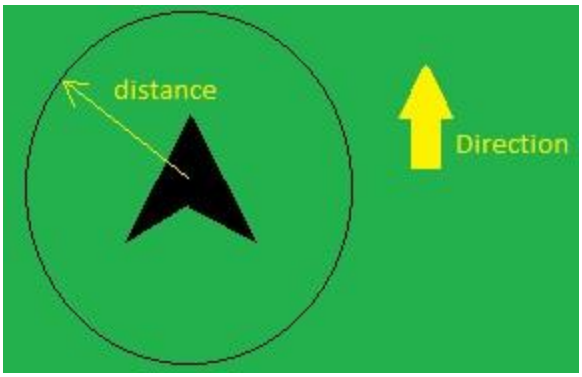


Figure 1.1. A bird's neighborhood [2].

Uri Wilensky, inspired by the previously stated rules about flocking from Craig Reynolds, worked on simulating the flock using Netlogo, and he was successful in implementing the overall model. His model of flocking in Netlogo was the motivation behind this paper's idea. By modifying the algorithm behind Uri Wilensky's model and taking Craig Reynolds' work on boids into consideration, this paper tries to solve objective 1 of simulating the flock [7].

This paper also focuses on predictive obstacle avoidance and goal seeking; the birds would avoid these static obstacles and make their way through the simulated world towards the goal [8]. A wind factor is applied to the simulation model to display more natural movement for the flock. Because the model is a 2D animation, the wind factor along the x-axis and y-axis has been considered and rendered to show how the flock is driven by the upwind and downwind effects.

Finally, a predator is introduced into the world and is set to target the flock. The escape behavior of the flock in response to the approaching predator is simulated. The simulation is performed on a varying population of birds with a single predator traveling at a constant velocity in the model, and the response is observed in each case. Responses with varying predator velocities are also observed.

2. RELATED WORK

"Birds of a feather flock together," a famous proverb, dates back to the mid-16th century words of William Turner from "The Rescuing of Romish Fox" as "Byrdes of on kynde and color flok and flye allwayes together" [9]. This simple proverb shows evidence of early research on birds' flocking behavior. Beginning in the early 20th century, many researchers started speculating on the movement of birds as gregariousness [10], which John Emlen, Jr. (1952) described as the tendency of the same kind of birds responding to each other. In 1916, Trotter described this behavior as a tendency of togetherness in the flock and to restrain from anything which opposes it, and in 1918, Craig stated it as an "appetite" which was a commotion that continues until the birds crave company [10]. A sense of togetherness, or attraction, between birds in a flock has the opposite forces acting as well to prevent collisions in the flock. A combination of centripetal and centrifugal forces acts on the birds; the centripetal force acts as the driving force in bringing the members of a flock together, and the centrifugal force helps regulate the birds' movement. The breadth and compactness of a flock are explained by these opposing forces [10].

In 1958, Beer described flocking as a group of birds comprised of two or more members associating with one another because of an intuitive, gregarious drift [11]. After observing the swirling movement of European Starlings, mostly during sunset over a roost for about half hour to an hour, researchers suggested their own theory about the reason for this behavior. In 1962, Wynne Edwards stated flocking as an "epideictic" show of the birds, allowing the birds to estimate the flock population, and that information can be used to manage the breeding process which is considered a smaller portion of the present-day "naïve group selection" [11]. Another suggestion from Major and Dill in 1978 stated that this wheeling and turning motion is a kind of

"protean" to confuse predators. Recent studies in the early 2000s revealed that these movements help guide pigeons to their nests.

The concept of a leader for a flock of birds has been studied, and many early researchers in the 1970s hypothesized that the swirl movements are caused by a leader turning another direction, triggering neighboring birds to turn accordingly which, in turn, causes the flock to turn as a whole. In 1984, Potts presented a "Chorus Line" hypothesis which says that a single turn by a random individual in the flock could steer the neighboring individuals, leading the birds far from the individual turn their direction (can be related to "Mexican Wave") [3]. A "self-generated synchronous activity" was suggested by David in 1980s; this activity causes the turning and wheeling movements for birds in a large flock. It was also the time when computers and programming languages were developed, and many scientists worked on developing a computer model to display bird flocking based on the idea that every bird in a flock follows some simple rules in order to result in a complex behavior. Akira Okubo (1986) [12], Craig Reynolds (1987) [5], and Frank H. Heppner (2009) [11] developed computer-simulated models for bird flocking. Craig Reynolds designed the computer animation for a flock of birds using three simple rules: separation, alignment, and cohesion. This model was then used in several motion pictures. Later, Heppner and Grenander attempted a similar feat where the birds rested in a roosting area, were able to fly at similar speeds with varying velocity regulators, and were able to move away from each other when they got too close and to move closer when they got too far. Natural impacts, such as wind and moving obstacles, were also included in the computer model using the Poisson Stochastic Process. In 1993, Meiko produced a computer animation with 100 slow-moving birds and 6 frames per second. In the 2000s with fast processors and graphic-

dedicated hardware, Craig Reynolds achieved 60 frames per second and ran 10,000 birds with high quality [11].

In 1995, Kennedy and Eberhart introduced the concept of artificial intelligence called swarm intelligence to bird flocking. They proposed an algorithm called "Particle Swarm Optimization" to regulate the swarm's behavior. Each individual in a swarm learns from the environment and the behavior of its neighbors, designing its behavior based on the knowledge attained [11].

In the 1990s, physicists started to research the physics and mathematics behind the organized flock, relating birds to particles or molecules in a fluid that would follow the same rules as Craig Reynolds's computing model and the perceptive model from Heppner and Grenander [11]. In 2000, Tam´as Vicsek and Andr'as Czir'ok introduced the concept of self-propelled particles where each particle has a similar speed and velocity, but moves in different directions based on the particle's average direction plus the direction of its close neighbors [13]. In 2007, Wood and Ackland simulated a predator, subjected the predatorto a flock of birds, and studied the flock formation. Couzin and his team (2002) formulated a new approach that was unlike Reynolds' model or Heppner and Grenander's model. He modified the perception model of the birds and named it "Zones." The perception level was divided into three zones: 1) Zone of Repulsion, 2) Zone of Orientation, and 3) Zone of Attraction. The separation rule applies when the neighbor is in the bird's Zone of Repulsion. If no neighbors are in the region, then the bird has to increase its speed to get back with the nearest neighbors, and the alignment and cohesion are averaged. The cohesion rule applies when the neighbors are in the Zone of Attraction [14]. In later studies, Couzin and his team introduced the leader concept to the model and allowed a certain number of individuals to have the knowledge of predefined direction. He observed that,

in a large flock, a small number of individuals was enough to steer the system. In 2009, Lebar Bajec and Frank Heppner brought fuzzy logic into flocking by following similar simulation techniques from Reynolds and Couzin, and by negating the blind spot in the sphere model of perception. Bajec and Heppner used the terms "far" and "close" instead of using the distance and angle in order to achieve a more realistic behavior. Mosken (2007) extended the fuzzy-logic model by introducing a new factor, hunger, to the system [11].

2.1. Collective Memory and Spatial Sorting in Animal Groups

Following Craig Reynolds' rules of flocking, Couzin and his team (2002) designed a computer model to display birds' collective memory by capturing variations in the spatial position of a single member in a large group that is following some simple local rules and interacting with other members without having knowledge of its present spot. The conversion from one kind of behavior to another for a large flock has been modeled depending upon the previous records of the group's structure [14].

Every other collective behavior was explained by modifying some parameters in the system, such as the Zone of Repulsion, Zone of Attraction, etc., which is explained in the following discussion. Also, the physical and motivational factors that are effective on the birds' positions in the group were examined. The results for how the structuring phenomenon of a natural flock is formed were studied [14].

The space around an individual bird is divided into three zones: Zone of Repulsion (Z_{or}), Zone of Orientation (Z_{oo}), and Zone of Attraction (Z_{oa}). Each bird tries to uphold a close distance with other birds. If a bird falls under another bird's Zone of Repulsion, the birds tend to move away from each other. If no neighbors fall under the Zone of Repulsion, then the bird tries to

communicate and move towards birds in the Zone of Orientation or Zone of Attraction. The space behind the bird is considered a "blind spot" because it is not visible [14].

Parameter changes, such as increasing or decreasing the width of these three zones, result in shifting the collective behavior of the flock from one form to another, aggregating to four physical forms [14]:

- *Swarm:* A behavior which is formed by lowering the alignment factor and increasing the cohesive angle. This type of behavior involves low or negligible parallel orientation.
- *Torus:* This physical form is achieved by lowering the alignment and increasing the cohesion and when the radius of Zone of Orientation (D_{ro}) is small and the radius of Zone of Attraction (D_{ra}) is large.
- *Dynamic Parallel Group:* The group with high alignment and less cohesion achieves this physical form and the radii of Zone of Orientation and Attraction are of intermediate or high values.
- *Highly Parallel Group:* As the name suggests, the group with very high radius of Zone of Orientation D_{ro}, with very high parallel alignment and very less cohesion achieves this type of physical formation. [14]

2.2. Collective Behavior of Interacting, Self-Propelled Particles

In the Self-Propelled Particle model proposed by Vicsek and Czir'ok, the particles propagate through the plane with constant speed, but with varying directions that are influenced by the neighboring birds, which, in turn, decide the mean course of movement for every time step with some level of disturbance added [13]. Through simple interactions and with disturbance added, this model could achieve some realistic phase transitions and, thus, can be

called a more dynamic form of the ferromagnetic model type with a basic function, depending on the course of a particle at a given time minus the previous time step.

The 2D simulation of this non-equilibrium model is performed with L representing the size of the window in which the particles are allowed to propagate freely with a certain pre-defined velocity. The distance between the particles is denoted by r, and the time interval between each time step is denoted by Δt. N is the number of particles; v is the velocity of an individual particle; and v_i is the velocity of all particles at a given time. θ is the direction change angle of the particles. Initially, all particles are distributed at time $t = 0$. $[\theta(t)]_r$ is the mean direction for all particles in the neighborhood of a particle in the give radius of r [15].

The position of a particle is changed using the formula $x_i(t + 1) = x_i(t) + v_i(t)\,\Delta t$ [15].

The direction change for the particles is formulated using $\theta(t + 1) = [\theta(t)]_r + \Delta\theta$ [15]. $\Delta\theta$ is considered the disturbance introduced into the system and is some random value taken from the range $[-\eta/2, \eta/2]$. ρ is the density of the particles calculated by N/L^2.

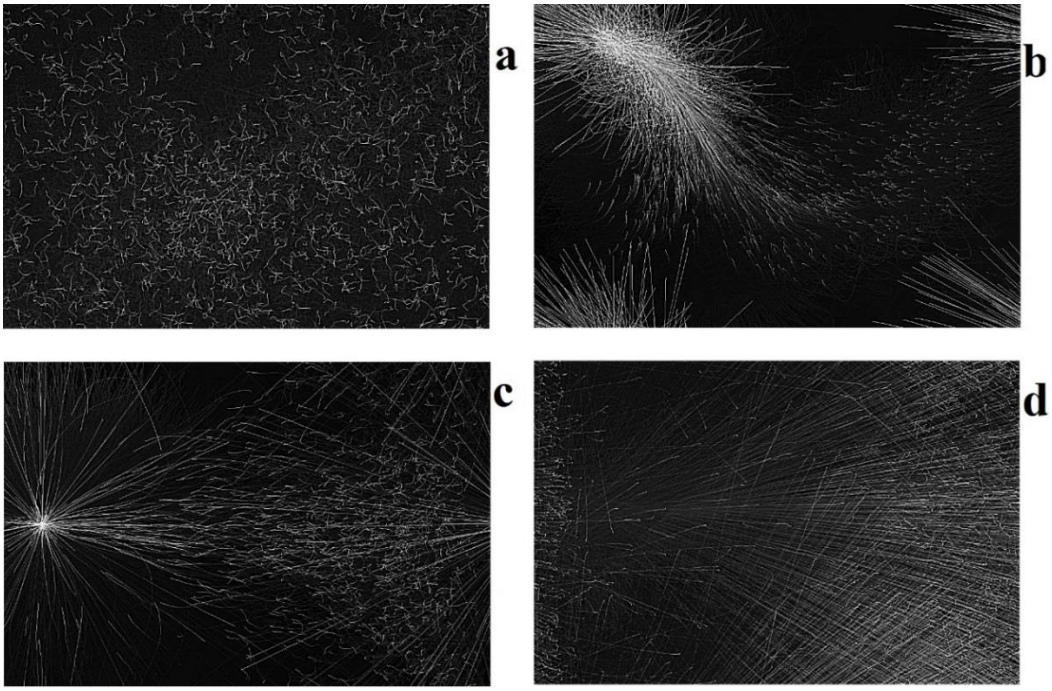


Figure 2.1. Movement of particles with a varying density and disturbance.

In Figure 2.1, (a) represents the movement of the particles at time $t = 0$, $L = 7$ and $\eta = 2.0$. These values influence the density and disturbance in the system; (b) represents the particles tending to aggregate in several clusters when the density is low and the disturbance in the medium is low ($L = 25$, $\eta = 0.1$); (c) represents the particles in varied motion with high density and high disturbance after some time, $t + \Delta t$ ($L = 7$, $\eta = 2.0$); (d) represents the rectilinear propagation of the particles more aligned with less disturbance and increased density ($L = 7$, $\eta = 0.1$) [15].

These transitions are observed by having the velocity of each particle at a constant value. The average velocity of the particles may be altered with a varying disturbance in the system, and the average velocity can be calculated by

$$v_a = [1 / (N*v)] * | \Sigma v_i | \quad (1)$$

The value of v_a is approximately zero when the particle flow is dispersed and is approximately 1 when the flow course is aligned. Hence, the average velocity, v_a, is considered as the order parameter of kinetic transition. Further studies revealed that the order parameter in this non-equilibrium model is analogous to that of an order parameter in an equilibrium model. All the SPP (Self-Propelled Particles) models are comprised of particles communicating with neighboring particles and moving with a constant speed [15].

2.3. A Minimalist Flocking Algorithm for Swarm Robots

A simple algorithm, without involving the interaction between the particles or robots, knowledge, and information of the system, has been developed by three researchers who incorporated the preexisting flocking algorithms for effective obstacle avoidance [16]. A small number of robots with infra-red sensors have been allowed to move freely in a closed area with several obstacles. The swarm robots travel from the start position to the end by avoiding blocks

or obstacles. The separation rule of flocking is satisfied by using the IR sensors placed on each side of the swarm robot. The reflection rate of the IR light is used for obstacle avoidance and the robot-separation rule. If the reflection from the other surface is over the limit for the threshold value, then the robots change direction. Sensors in front behave as the robots' eyes to avoid obstacles, and sensors on the east, west, and south side of the robot geometry are used to satisfy the separation rule [16].

Simple changes in the threshold levels for each sensor achieved the alignments and cohesion rules. The threshold level for the sensor to the back of each robot has been adjusted so that each robot follows the remaining robots [16].

Analyzing this technique revealed that the maneuverability of the swarm robots was influenced by the swarm's population. Visible tests suggested that, with more robots, the swarm had less maneuverability. A flock with 5 robots was able to move 2/3 the distance that a single robot could move in a specific time. When the size was about 25, the swarm had a mobility of $1/4^{th}$ that of a single robot [16].

This model is not a computer simulation but is an example of how the flocking algorithm is incorporated in real-world mechanics to influence objects in the real world.

2.4. A Simulation Study about the Form of Fish Schooling to Escape from a Predator

A simulation model of fish schooling in the presence of a predator, analyzing a school of fish's behavior when evading a predator, was performed. An evolutionary model of fish behavior was taken as a parameter, and the fish's pattern when encountering a predator was studied. The above hypothesis was extended by projecting computer simulations of the prey-predator system on an artificial ecology where the prey and the predator coexist. The evading behavior of the prey and the predator also formed a vital portion of the analysis [17].

In 2003, Kato and his team developed a model based on their study of fish interaction with neighboring fish while predator evasion and a hypothesis were designed.

1. First, the model was assumed to be developed in a 2D world.
2. Second, the time variable was quantized, and each individual movement was derived for each interval of Δt; i.e., every movement was based on the previous step.
3. Third, speed and direction were assumed to be two components that determine the fish movement and were considered to be mutually exclusive.
4. Fourth, the velocity of each individual was explained separately without depending on the other fish [17].

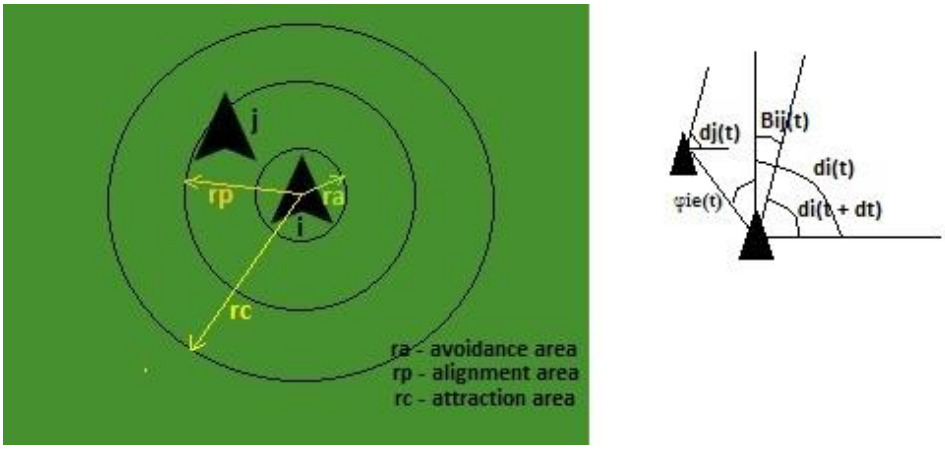


Figure 2.2. Range of the basic behavior patterns [17]**.**

Figure 2.2 shows the various regions of avoidance, attraction, alignment, and search for an individual, with the individual marked in black. In this example study, there are two individuals, i and j. i and j are in each other's neighborhood, and i behaves according to j's behavior, which makes j the reference individual for i. Hence, the movement of i is comprised of direction and speed, and di (t) is the direction of i at time t. di (t + Δt) is defined as the sum of βij(t) (the turning angle of i for j) and β0 which means to wiggle about every decision of the direction [17].

2.4.1. Decision about the direction of movement

When a predator is in the individual's vision, a sense of urgency mode is triggered in the school of fish. The fish tend to aggregate and turn in the direction of the predator. As the predator approaches, the fish try to separate into groups, and as the predator moves away, the fish tend to aggregate and perform schooling. If i is an individual, j is the reference individual for i, and e is the predator (in this case, a shark), the urgent mode for the direction decision is calculated as,

$$di(t + \Delta t) = di\,(t) + \beta ije(t) + \beta 0 \quad (2)$$

$$\beta ije(t) = (Aijt + Bijt + Ciet + Diet) \quad (3)$$

where Aij(t), Bij(t), Cie(t), and Die(t) are turning angles for parallel to j, attracted to j, averting from e, and away from e, respectively [17].

$$Aijt = dj(t) - di(t) \quad (4)$$

$$Bijt = \varphi ij(t) \quad (5)$$

$$Ciet = \min(\varphi ie(t) \pm 90°)f_f \quad (6)$$

$$Diet = \varphi ie(t) - 180° \quad (7)$$

2.5. Agent-Based Modeling

Agent-based modeling (ABM), a powerful simulation technique, has been used in many real-time situations to resolve many real-world problems. Its major application areas, such as flow simulation, organizational simulation, market simulation, and diffusion simulation, are discussed using real-world applications [18].

With agent-based modeling, the cooperative action of interacting or decision-making individuals called agents is required to model a simulation. Every agent is set to follow certain rules in the process of decision making based on the situation and may behave appropriately for

the system. A simple agent-based model has agents and various relationships among them. Every agent can simulate complex behavior by following simple rules and can evolve, resulting in unpredictable behaviors [18].

2.5.1. Benefits

- ABM displays emergent phenomena;
- ABM provides an easy description of a system; and
- ABM is flexible [18].

2.5.1.1. Emergent phenomena

The interaction between the agents is responsible for the emergent behavior of the agents. The interaction between the agents makes the entire system on the whole and not just the sum of the agents.

2.5.1.2. Natural description of the system

ABM provides a natural and easy way of explaining the simulating behavior whether it might be to describe a traffic jam, stock market, how the birds flock, or diffusion of a crowd from a hall. It is easier to display an animation of how the shoppers at a shopping center move or check out instead of designing mathematical equations to regulate the shoppers' density. Also, ABM helps in studying the aggregate behavior of systems.

2.5.1.3. Flexibility

ABM is flexible because we can regulate the population of agents; ABM allows the users to tune the various complexities of agents, such as behavior, interaction, rationality, etc. There are several areas of application:

1. Flow Simulation: evacuation, traffic, and customer flow management.
2. Market Simulation: stock market, shop bots and software agents, and strategic simulation.

3. Organization Simulation: operational risk and organizational design.
4. Diffusion Simulation: diffusion of innovation and adoption dynamics. [18]

There are many areas of application for ABM, especially in the social context where people are influenced by other people. In business areas, the simulation of social behaviors has not been a hit because the attention was to use ABM as a predictive tool instead of a learning tool. For example, a newly appointed manager can understand the situation of the market where he/she is going to work by running the simulation of the organization but cannot quantify the actual benefits because the tool is just a simulation and not the real world.

2.5.2. Useful approaches

1. When the behavior of individuals is altered by other agents due to interactions between them.
2. When the position of the agents is random in the system and the space of simulation is critical. Example: fire escape, theme park, supermarket or traffic.
3. When each individual is different from each other and the population of the system varies.
4. When the individual interactions are heterogeneous and complex.
5. When the system requires complex behavior of learning and adaptation. Example: flocking, schooling, etc. [18]

2.5.3. Issues

Every agent-based model serves a particular purpose, but a general purpose model may not be helpful for explaining all scenarios. A social model cannot explain a political behavior and vice versa.

In the social sciences, human agents are often considered with similar behavior, but in reality, humans exhibit irrational behaviors and psychology. Quantification of these features is difficult and cannot be justified. [18]

2.6. Netlogo

Netlogo is a powerful programming tool that is used for agent-based modeling to simulate the social and natural phenomena observed in everyday life. In 1999, Dr. Uri Wilensky, the founder and director of the Center for Connected Learning and Computer-Based Modeling, designed Netlogo, and it has been in use ever since by researchers around the world. Netlogo is used in a wide sense for restructuring predefined equations with agent-based modeling. [7]

Netlogo is well prepared for simulating complex behaviors. Users can give commands to thousands of agents, and every agent functions without influencing other agents and helps move simple, individual behaviors into complex, group behaviors; i.e., simple interactions between the agents result in complex behavioral outcomes. [7]

A successful Netlogo simulation lets users experiment with the system by varying controls and helps them understand the purpose of the model that imitates the natural or social behavior. Students and teachers use the simple design technique of Netlogo to implement curriculum projects, and at the same time, very advanced researchers use it as a high-end tool in various fields. [7]

2.6.1. What Netlogo offers

Netlogo offers three different spaces: one for user interaction, the second for documentation (used to write up a brief tutorial for the model), and the third for coding the system. A “models library” is integrated, consisting of a large database of example models that

are designed and developed by Netlogo users. Figure 2.3 shows the sample Netlogo tool and the three tabs offered: Interface, Info, and Code. [7]

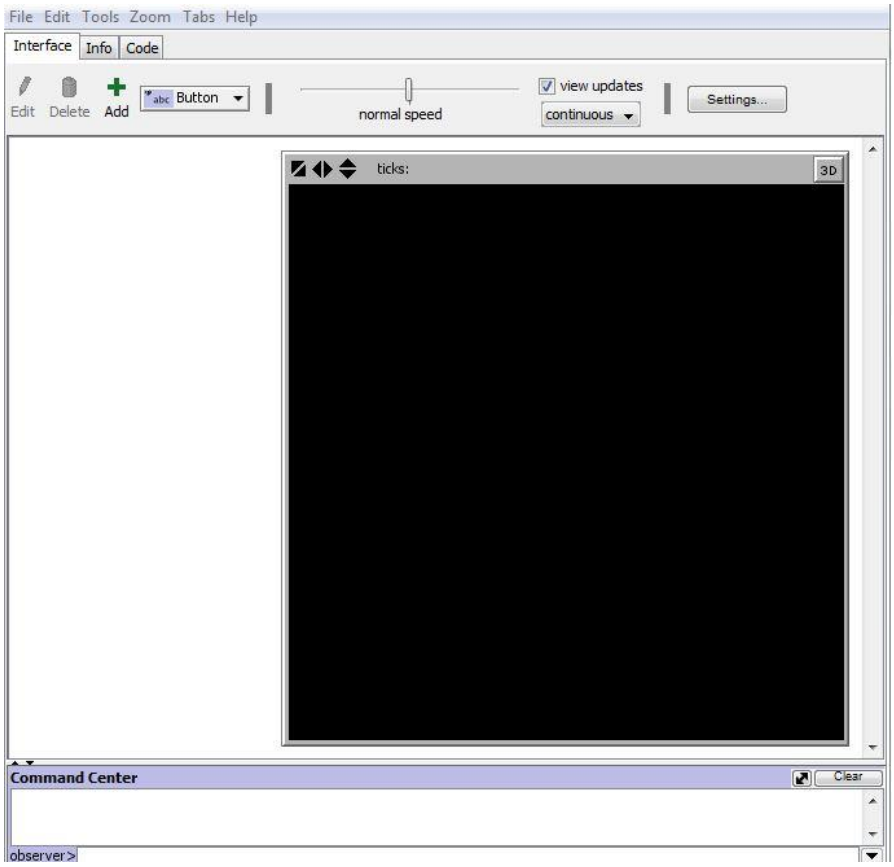


Figure 2.3. Netlogo interactive screen.

The interface space of Netlogo lets the users set up various interactive tools, such as buttons, sliders, switches and graphs, etc., in the system to make the model easy to use, and the simulation space can be adjusted based on the simulation's need. Users are also given the ability of controlling the simulation speed from a slower speed to a faster speed. Usually, simulations are run at normal speeds. Users can export or import any Netlogo model into the system and run it. [7]

The second space for documentation (Info) allows users to write something about the simulation if the user wants to explain what the model does and how it is operated. Figure 2.4

shows the Info space for Netlogo. The third space (Code) for the system contains the actual procedures functioning behind the system. Figure 2.5 shows the coding space for NetLogo.

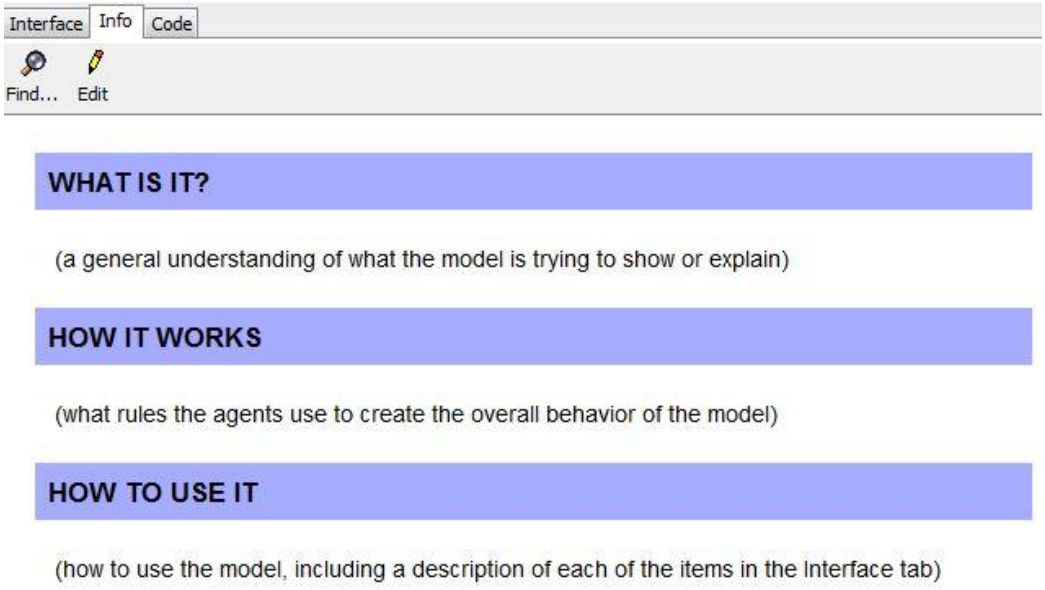


Figure 2.4. Info tab for Netlogo.

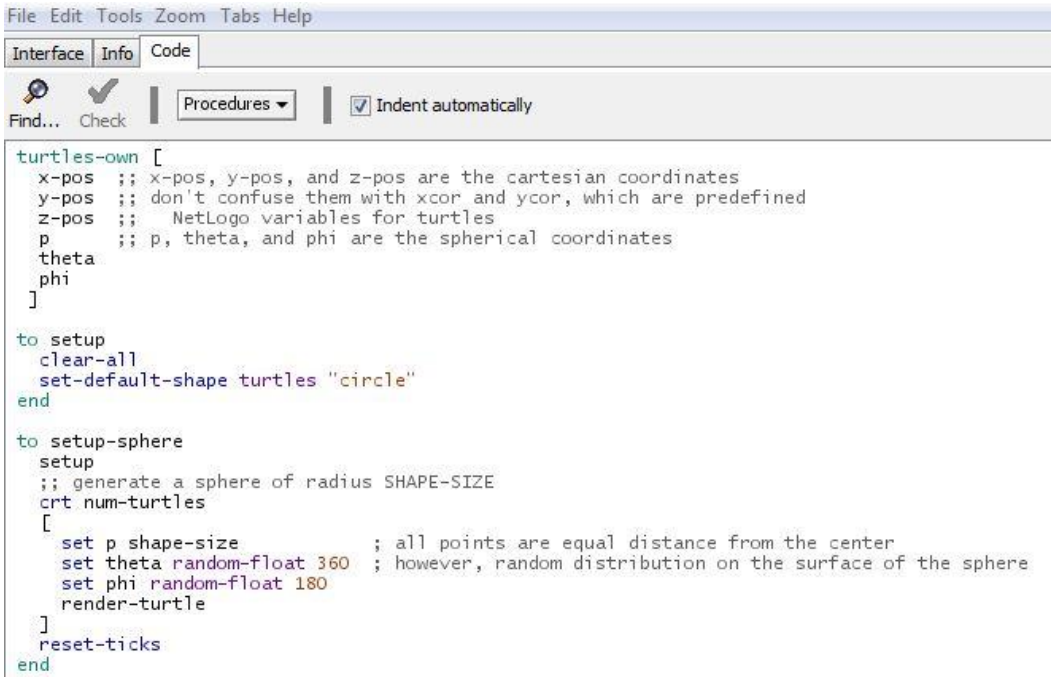


Figure 2.5. Code tab for Netlogo.

The simulation space is comprised of agents such as patches, turtles, links, and the observer. The 2D world is comprised of several grids called patches. A patch represents a rectangular grid area on which the turtles move. A turtle is an agent which flies around the simulation space. A link is a connecting agent which is used to connect the turtles. Figure 2.6 shows the turtles and patches in the 2D space. Figure 2.7 shows the links between the turtles. [7]

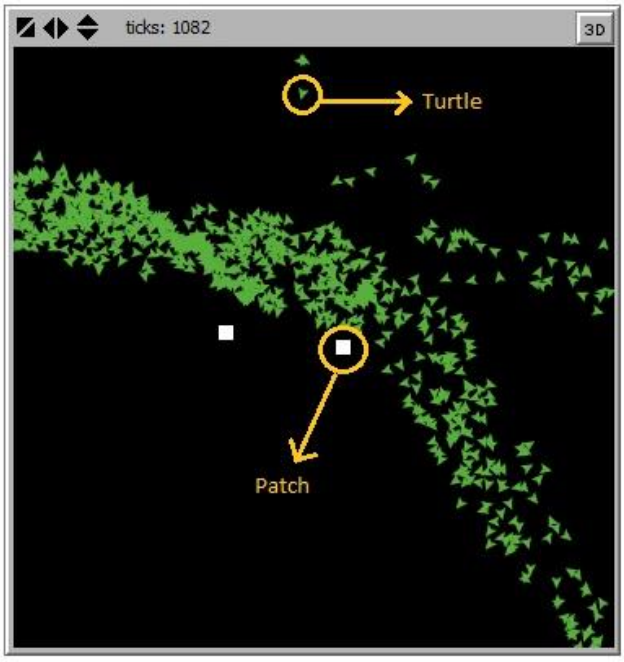


Figure 2.6. Turtles moving above the patches.

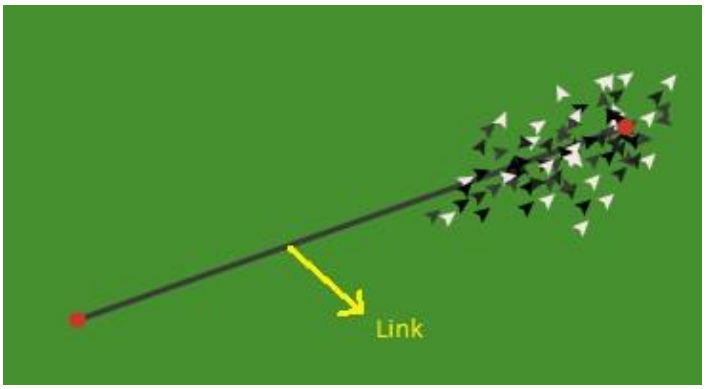


Figure 2.7. Link connecting the two turtles.

The user has the ability to command agents from the provided command-line console just below the simulation space. Figure 2.8 shows the command-line console.

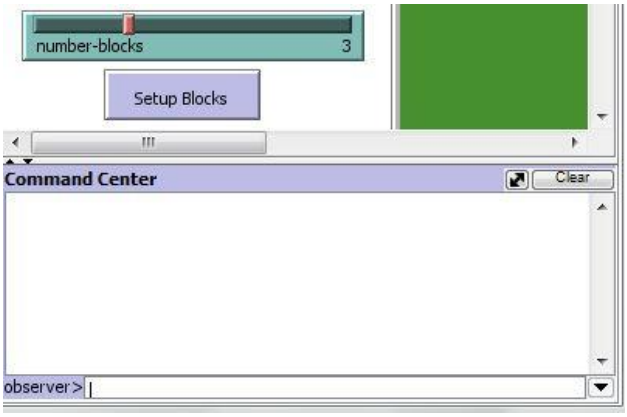


Figure 2.8. Observer's command-line console.

Netlogo is platform-independent software which works on all types of operating systems because it is based on the Java Virtual Machine and even the command-line operation is supported. Netlogo is the derivative of yesteryear's StarLogo which was developed in 1997 by Dr. Uri Wilensky, and since 2000, Netlogo has been undergoing constant upgrades. The current version of Netlogo is 5.0.3; it started with version 1.0 in 2002, 2.0 in 2003, 3.0 in 2005, etc. [7]

Netlogo was basically inspirational software developed on the basis of the Logo programming language. Logo programming was developed as an education tool with the features of interaction, modulation, and flexibility. Logo programming had a real-world robot called a turtle which moved on the floor based on the computer commands. Later, the turtle was shifted to graphics, extending the turtle's usage in designing patterns and several shapes, and even computer games. At the present time, Logo programming is remembered for its turtle graphics. [7]

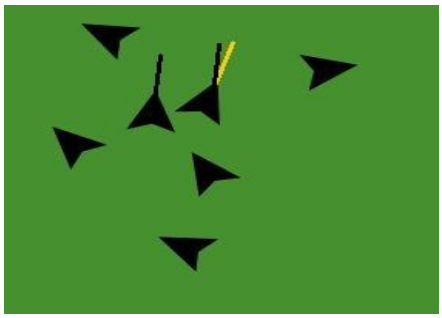
3. PROBLEM DESCRIPTION

All the papers discussed in the previous chapter introduce the concept of swarm intelligence and how the approach can be incorporated into real-world examples. This chapter introduces the problem statement and the research approach formulated to solve the stated problem. The problem statement is divided into three objectives, and the approach for each objective is explained.

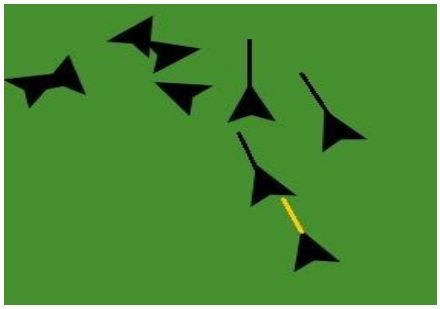
3.1. Objective 1

The objective 1 is to develop working software that simulates movement for a flock of birds using agent-based modeling in a two-dimensional world. Birds' flocking behavior is one of the most fascinating and complicated behaviors to understand, and replicating that behavior through graphics used to be difficult. In solving this problem, a 2D closed world was assumed, and Netlogo was used to program the simulation. Craig Reynolds' work creating a computer program to simulate the flocking behavior [5] was adopted as the basis of designing such complex behavior. The three basic rules from Craig's research, separation, cohesion, and alignment, were incorporated into the system with little modification. The Self-Propelled Particle model from Vis'cek [13] was considered in defining constant velocity for the birds. The turtle agents representing the birds were named "burtles" for convenience.

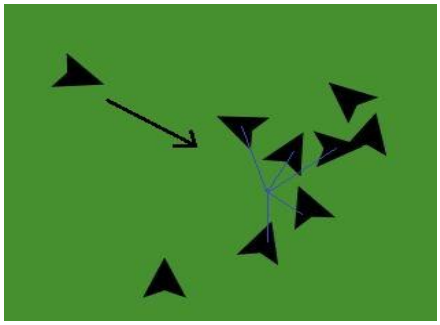
Three separate procedures for separation, cohesion, and alignment were programmed based on Craig's algorithm and applied on the burtles. Figure 3.1 explains Craig's rules for flocking.



Separation: Steer to avoid crowding local flock mates [5]



Alignment: Steer towards the average heading of local flock mates [5]



Cohesion: Steer to move toward the average position of local flock mates [5]

Figure 3.1. Craig Reynolds's rules for flocking.

Also, the decision of separation between the birds and setting up the number of birds in the system were inspired by Uri Wilensky's work that simulated flocking in Netlogo [7]. Figure 3.2 shows a sample flocking simulation prepared for the present model with a population of 20 burtles following the 3 flocking rules. A simple overview of the algorithm is as follows:

1. Users can set up a random number of burtles in the world
2. All burtles are set to propagate at fixed velocity and find the other burtles in the vision
3. If there is a burtle in the vision, then apply the flock rules: separate, align, and Cohesion
4. If the burtles cannot move or if they reach a closed boundary, then turn the direction.

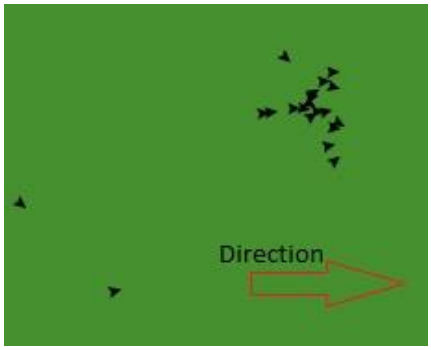


Figure 3.2. Simulation of 20 burtles performing flocking in the 2D space.

3.2. Objective 2

The objective 2 is to direct the flock towards the high-priority foraging area and to study the behavior with natural factors, such as wind and obstacles, hindering the flock's movement. This objective of foraging, obstacle avoidance, and wind-factor influence was divided into three sub categories. Each category was addressed individually to avoid complexity and to be integrated safely into the system through debugging.

3.2.1. Foraging

Netlogo supports user interaction with the simulation model. Hence, the best approach to create forage is considered by allowing the user to interact with the simulated area. Users can create forage using a mouse-click event. The most-recently created forage denotes the high-priority forage, and the burtles are driven to the high-priority forage. Figure 3.3 shows the flock at the forage 41.

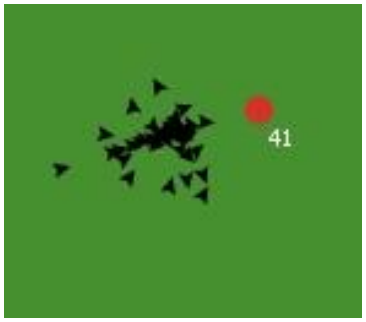


Figure 3.3. Flocking behavior at forage (displayed in red).

3.2.2. Obstacle avoidance

The user is given the ability to decide the number of obstacles to be placed in the world. With a maximum of 8 obstacles, the user can choose from 1 to 8. While propagating through the space, if a block is in the burtle's vision and is within the distance of 5 patches, it is programmed to change its direction left or right towards the neighboring burtles. Figure 3.4 shows the flock trying to reach forage 47 by avoiding the obstacles.

Figure 3.4. Burtles performing flocking and avoiding obstacles.

3.2.3. Wind factor

Wind makes the burtles sway around the world from top to bottom, left to right, and vice versa. The speed of the burtles varies depending on the flight direction. The user has the ability to change the wind effect on the system and can select from a set of negative and positive values for wind effect towards the x-axis and y-axis. A negative wind effect towards the x-axis speeds up the burtles propagating from left to right, slows down the ones propagating from right to left, and vice versa. A negative wind effect towards the y-axis speeds up the burtles propagating from bottom to top, slows down the ones propagating from top to bottom, and vice versa.

3.3. Objective 3

The objective 3 is to introduce a predator into the world, study, and simulate the flock's escape behavior. A predator is introduced into the world and is driven towards the center of the flock's gravity. In response to the movement of the predator, the flock tries to move away from

the predator. As the predator dives into the flock, the burtles tend to move towards the centroid of the local neighbors; as the predator approaches, the flock tries to separate from other burtles and the predator; and as the burtles are outside the region of attack, they tend to aggregate and perform flocking. This escape behavior is accomplished by subtracting the burtle's heading with the average heading of the other burtles.

4. IMPLEMENTATION

This chapter explains how the burtles are set up in the 2D world and how the burtles propagate around, avoiding the obstacles and swaying with the wind having an effect on their movement. Also, the implementation of how the burtles try to escape when the predator dives into the flock is explained.

4.1. Assumptions Considered in the Model

The burtles are assumed to possess similar properties such as size, shape, and color. The birds' age has not been considered. The initial assumption for placing the burtles is a circle layout. Each burtle is set to propagate at a similar speed initially and possesses a similar mode of vision. Individuals possess a blind spot while flocking and also during obstacle avoidance and predator evasion; a vision angle of 270 degrees is considered by following the spatial sorting from Couzin's model [14]. No leader burtle is considered, and each burtle is set to follow three simple rules of separation, cohesion, and alignment. Coming to the predator, a single hawk and three predator prone zones are considered, and the hawk is set to propagate with a speed faster than the burtles and along the center of the flock's gravity.

4.2. Algorithm Description

In this section, the algorithm and the code working behind the simulation are explained. We have the algorithms listed for objective 1, 2, 3. Each objective is explained, and how the algorithm tries to satisfy each objective is mentioned.

4.2.1. Objective 1

This section explains how the burtles are created and placed in the world as well as their propagation following the rules of separation, alignment, and cohesion. A 2D world of 40x40 has been considered, and all burtles with similar characteristics, such as shape, size, and color, are

placed at random positions in a circular layout. Figure 4.1 shows the shape of the burtles and the layout in which the burtles are initially placed while setting up the environment.

Figure 4.1. Shape and initial placement of the burtles.

The burtles are set to move at constant speed of 0.3 patches/sec, and while moving forward, if a burtle is in the vision range of the other burtles, then the cohesion rule is applied on the burtle which tends to move the burtle closer to other burtles in the range. If the burtles are too close to each other, then the rule of separation comes into action, and the other two rules are nullified. The burtles move away from each other until they are outside the range of repulsion. Once the burtles are outside the range of repulsion, the rules of alignment and cohesion come into play, and the burtles tend to move along the direction of propagation for the other majority close by to the burtle. A fixed angle for maximum separation, cohesion, and alignment is considered. The angle by which the burtle is set to turn is dependent on this fixed max angle. If a burtle is within the attraction range of its neighbor, the cohesion and alignment forces act upon the burtle; while aligning, the burtle is driven towards the heading direction for the remaining burtles; and while attracted to other burtles, the burtle is driven towards the flock's center of gravity. Once the burtle is in another burtle's range of rupulsion, the heading of that burtle is dependent on the direction of the one closest to it.

4.2.1.1. Algorithm

The algorithm designed to satisfy the objective 1 is stated below. This algorithm explains the initial setup and movement of the burtles.

Step 1: Specify the shape, size, and color of the burtles, and set up the burtles' placement

Step 2: Specify the burtles' speed

Step 3: Move the burtles forward with the specified speed

Step 4: For each burtle, find the nearest neighboring mates

Step 5: Avoid any obstacles within the vision

Step 6: If the separation between the burtles is less than the threshold, i.e., if any burtles are in the range of attraction, then apply cohesion and alignment on that burtle

Step 7: If the separation between the burtles is above the threshold, then apply the separation rule

Step 8: If the burtle touches the world's boundary, then change the burtle's direction

Step 9: Finish

In Netlogo, the commands are incorporated into buttons and sliders, and the procedures are called from these tools. The initial stage of the simulation is to create the environment and to set up the population of burtles in the environment. As shown in Figure 4.2, a "start" button is considered to create the environment. Users can select from 1 to 80 burtles utilizing the slider for the "number-of-birds" and up to 8 blocks with the "number-blocks" slider which act as obstacles. Obstacle avoidance is explained further in the next sections. The "setup" button is used to place burtles in the environment, and the "go" button starts the simulation.

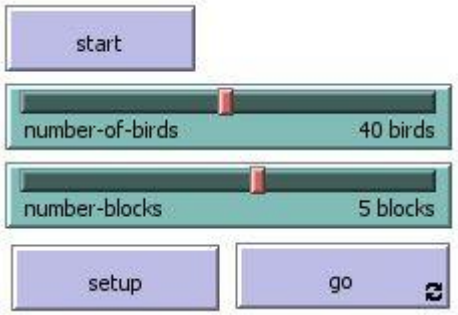


Figure 4.2. Simulation tools of operation.

4.2.2. Objective 2

This subsection explains how the forage is created and how the burtles are driven towards the forage. The burtles try to avoid the obstacles in the path and overcome the wind force acting on the flock in the process.

4.2.2.1. Creating forage

To create forage area, users can click on any part of the designated environment. Users can create any number of forages as desired, and the high-priority forage is defined by how the latest the creation is, meaning that the newly formed forage is of high interest for the burtles. When the user clicks on a particular area, the X and Y coordinates of the mouse position (x_f, y_f) are saved, and forage is created; the burtles tend to move towards (x_f, y_f). New forage is created when the user clicks another position (x_{fi}, y_{fi}), and the burtles are attracted towards (x_{fi}, y_{fi}), the newly formed forage.

4.2.2.2. Algorithm

The algorithm designed to satisfy the first part of objective 2 of goal seeking is stated below. This algorithm explains the initial setup of forages and movement of the burtles towards the forages.

Step 1: If no forage is created, then select the mouse's X and Y coordinates, and save the coordinates in (x_f, y_f)

Step 2: Create a forage area at (x_f, y_f), and set the heading of the burtles towards (x_f, y_f)

Step 3: When the user clicks another area, the new mouse coordinates are saved in (x_{fi}, y_{fi}),

and new forage is created

Step 4: Set (x_f, y_f) = (x_{fi}, y_{fi})

Step 5: Set the burtles' heading to (x_f, y_f)

Step 6: Finish

4.2.2.3. Code

The program code behind the first part of objective 2 of goal seeking is described below.

This code explains how the flock is directed towards the forage.

```
BEGIN PROCEDURE
is first forage?
     set xcor mouse-xcor
     set ycor mouse-ycor
     set x-cord xcor
     set y-cord ycor
     create forage i
if not the first forage
    set xcor mouse-xcor
    set ycor mouse-ycor
    set x-cord xcor
    set y-cord ycor
    create forage i+1
retx:
if xcor = x-cord
    report xcor
else
    report x-cord
rety:
if ycor = y-cord
    report ycor
else
    report y-cord
Set heading of burtle towards (retx, rety)
END
```

4.2.2.4. Obstacle avoidance

This subsection explains how the obstacles are placed in the simulation and how the burtles try to avoid these obstacles while propagating around the space or while moving towards foraging area. For obstacle avoidance, first we consider the creation of obstacles. Users can select from 1 to 8 blocks for optimum usage of the simulation space. A slider bar is provided to choose the number of obstacles to place, and the "Create Blocks" button is used to place the obstacles in the environment randomly around the space. While propagating, if a block is in the burtle's vision, then it tends to change direction and follow the other burtles.

4.2.2.5. Algorithm

The algorithm designed to satisfy the second part of objective 2 of obstacle avoidance is stated below. This algorithm explains the movement of the burtles by avoiding the obstacles.

Step 1: Select the number of blocks, and set them up

Step 2: If a block is in the burtle's vision and the distance from the block is 5 patches, then

Step 2.1 Change the turn angle

Step 2.2 Turn towards the average direction of other birds

Step 3: Finish

4.2.2.6. Code

The program code behind the second part of objective 2 of obstacle avoidance is described below. This code explains how the flock propagates avoiding the obstacles.

```
BEGIN PROCEDURE
if count blocks > 0 [
      if ((any blocks in-vision with [ distance one-of blocks < 5 ] ))
          if (random 2 = 0) /* returns 0 or 1 */
               direction = 1
          else
               direction = -1
          right-turn = direction * turn-angle
```

]
set heading towards average heading of other birds
END

4.2.2.7. Wind effect

This subsection explains the wind's effect on flock propagation as well as how the wind factor along the X and Y axes have an effect on the horizontal and vertical propagation of the flock. An on/off switch for the wind factor and two sliders each for wind effect along the X-axis and y-axis are provided. Each slider has values from -40 to +40. A negative value on the X-axis indicates that the wind is blowing from left to right and vice versa. Similarly, on the Y-axis, the negative value indicates that the wind is blowing from bottom to top and vice versa. Hence, the burtles traveling upstream will have less speed compared to the downstream burtles.

4.2.2.8. Algorithm

The algorithm designed to satisfy the third part of objective 2 of wind effect is stated below. This algorithm explains the movement of the burtles in the presence of wind.

If Wind-Factor = true, then

Step 1: Get the direction of the burtle

Step 2: If Wind-X is greater than 0 and Wind-Y is greater than 0, then

If the direction is left to right, decrease the velocity, else increase the velocity

If the direction is bottom to top, decrease the velocity, else increase the velocity

Step 3: If Wind-X is greater than 0 and Wind-Y is less than 0, then

If the direction is left to right, decrease the velocity, else increase the velocity

If the direction is bottom to top, increase the velocity, else decrease the velocity

Step 4: If Wind-X is less than 0 and Wind-Y is greater than 0, then

If the direction is left to right, increase the velocity, else decrease the velocity

If the direction is bottom to top, decrease the velocity, else increase the velocity

Step 5: If Wind-X is less than 0 and Wind-Y is less than 0, then

If the direction is left to right, increase the velocity, else decrease the velocity

If the direction is bottom to top, increase the velocity, else decrease the velocity

Step 6: Finish

4.2.2.9. Code

The program code behind the third part of objective 2 of wind effect is described below.

This code explains how the flock propagates in the presence of wind.

```
BEGIN PROCEDURE
If Wind = true (
      if Wind-effect-x > 0 and Wind-effect-y > 0
           if heading towards left
             set x-velocity velocity * 2
           else
             set x-velocity velocity/2
           if heading towards top
             set y-velocity velocity * 2
          else
             set x-velocity velocity/2
     if Wind-effect-x > 0 and Wind-effect-y <  0
           if heading towards left
             set x-velocity velocity * 2
           else
             set x-velocity velocity*2
           if heading towards top
             set y-velocity velocity/2
          else
             set x-velocity velocity * 2
   if Wind-effect-x < 0 and Wind-effect-y >  0
           if heading towards left
             set x-velocity velocity/2
           else
             set x-velocity velocity*2
           if heading towards top
             set y-velocity velocity*2
          else
             set x-velocity velocity * 2
    if Wind-effect-x > 0 and Wind-effect-y <  0
           if heading towards left
             set x-velocity velocity/2
```

```
          else
            set x-velocity velocity * 2
          if heading towards top
            set y-velocity velocity/2
         else
            set x-velocity velocity * 2
set velocity sqrt ( (( x-velocity ) ^ 2 ) + (( y-velocity ) ^ 2) )
fd velocity
END
```

4.2.3. Objective 3

This section explains how the predator is set up and is made to target the flock's center of mass. The flock's predator-avoidance behavior and how the simulation tries to satisfy this behavior have also been illustrated. Initially, three dwelling places for the hawk have been considered, and the hawk is in one of the dens; the burtles are moved to a new forage area formed along the X-axis. Only a single hawk has been considered as the predator for the current model to reduce the complications and for a smooth simulation run. Users can set up the predator by utilizing the "Setup-Predator" button and can set the predator to hunt burtles using the "Go-Predator" button. When a hawk is set for predation, it tends to move towards the flock's center of mass. If the flock notices that the hawk is approaching, then the flock tends to scatter and avert the hawk. After the predator fails to dive in, the flock tries to aggregate itself. By performing this behavior, the polarization of the flock, the angle deviation of the hawk, and the placement of the individual burtles with respect to the hawk are considered. Three scenarios are prepared to explain the evading behavior of the flock; these scenarios explain the evading nature based on the burtles' position with respect to the predator, (a) right opposite, (b) opposite, and (c) slant opposite [17], following the Kato's model of predator evasion for a school.

(a) Right Opposite: The predator faces the center of the flock.

(b) Opposite: The predator faces the side of the flock.

(c) Slant Opposite: The heading of the predator is much more deviated from the center of the flock.

The flock tries to avoid predation by following three rules.

i) Dispersion

ii) Deflection

iii) Aggregation

Dispersion is a situation when the predator is nearing the flock. Deflection is the heading action of the individual birds based on the heading of the predator. Aggregation is the congregation of the scattered individuals after a successful evasion. The flock only performs deflection when the predator targets the side of the flock.

4.2.3.1. Algorithm

The algorithm designed to satisfy the objective 3 of predator is stated below. This algorithm explains the movement of the burtles with respect to the movement of the predator.

Step 1: Set up the predator and the predator resting areas (den)

Step 2: Calculate the flock's center of gravity (C_x, C_y) by

Step 2.1: $\mathbf{C_x = 1/M * \sum_{1\ to\ n} (m_i\ x_i)}$, where M is the total mass of the burtles, n is the population of the burtles, x_i is the x-coordinate value of the individual burtle, and m_i is the mass of the burtle; in this case, the mass is 1.

Step 2.2: $\mathbf{C_y = 1/M * \sum_{1\ to\ n} (m_i\ y_i)}$, where M is the total mass of the burtles, n is the population of the burtles, y_i is the y-coordinate value of the individual burtle, and m_i is the mass of the burtle; in this case, it is 1.

Step 3: Set the predator to target (C_x, C_y)

Step 4: If the distance from the predator to the flock is less than 10 blocks, then

Step 4.1: Get the direction of movement “h” of the predator;

Step 4.2: Get the direction of movement “hb_i” of the each burtle

Step 4.3: Set the direction of movement of the burtles to “h”

Step 5: If the distance from the predator to the flock is less than 5 blocks, then

Step 5.1: If X_i > P-X and Y_i > P-Y

If hb_i >= h and then

Set the heading of the bird $bh_i = bh_i + 60$

else

Set the heading, $bh_i = bh_i + 60$

Step 5.2: If X_i < P-X and Y_i > P-Y

If hb_i >= h and then

Set the heading of the bird $bh_i = bh_i - 60$

else

Set the heading, $bh_i = bh_i + 60$

Step 5.3: If X_i > P-X and Y_i < P-Y

If hb_i >= h and then

Set the heading of the bird $bh_i = bh_i + 60$

else

Set the heading, $bh_i = bh_i – 60$

Step 5.4: If X_i < P-X and Y_i < P-Y

If hb_i >= h and then

Set the heading of the bird $bh_i = bh_i - 60$

else

Set the heading, $bh_i = bh_i - 60$

Step 5.5: If X_i = P-X

If $hb_i >= h$ and then

Set the heading of the bird $bh_i = bh_i + 60$

else

Set the heading, $bh_i = bh_i - 60$

Step 5.6: If Y_i = P-Y

If $hb_i >= h$ and then

Set the heading of the bird $bh_i = bh_i - 60$

else

Set the heading, $bh_i = bh_i + 60$

Step 6: Finish

4.2.3.2. Code

The program code behind the objective 3 of predator evasion by burtles is described below. This code explains how the burtles navigate to avoid the predation by following simple rules of avoidance.

```
BEGIN PROCEDURE
Let p-position random 3
if p-position = 0 [
     setxy hawk-x - 5 hawk-y
     facexy bird-x bird-y]
 if p-position = 1 [
      setxy hawk1-x - 5 hawk1-y
      facexy bird-x bird-y ]
if p-position = 2 [
          setxy hawk2-x - 5 hawk2-y
          facexy bird-x bird-y]
foreach sort-on [ number-of-birds ] birds [
```

```
   ask ? [
   set bird-x bird-x + xcor
   set bird-y bird-y + ycor ]
   let burt-x ( bird-x / number-of-birds)
   let burt-y ( bird-y / number-of-birds)
   ask hawk [ facexy burt-x burt-y ]
ask birds with [distancexy hawk-xcor hawk-ycor <= 10  [ set heading hawk-heading ]
ask birds with [distance cur-hawk < 5 ] [
     if xcor > hawk-xcor and ycor > hawk-ycor
          if h >= hawk-heading
                turn-away-predator (subtract-headings (heading + 60) (hawk-heading)) 10
          else
                turn-away-predator (subtract-headings (heading + 60) (hawk-heading)) 10

if xcor < hawk-xcor and ycor > hawk-ycor
          if h >= hawk-heading
                turn-away-predator (subtract-headings (heading - 60) (hawk-heading)) 10
          else
                turn-away-predator (subtract-headings (heading + 60) (hawk-heading)) 10
     if xcor > hawk-xcor and ycor < hawk-ycor
          if h >= hawk-heading
                turn-away-predator (subtract-headings (heading + 60) (hawk-heading)) 10
           else
                turn-away-predator (subtract-headings (heading - 60) (hawk-heading)) 10
     if xcor < hawk-xcor and ycor < hawk-ycor
          if h >= hawk-heading
                turn-away-predator (subtract-headings (heading - 60) (hawk-heading)) 10
          else
                turn-away-predator (subtract-headings (heading -60) (hawk-heading)) 10
      if xcor = hawk-xcor
          if h >= hawk-heading [ set heading heading + 60 ]
          else [ set heading heading – 60 ]
      if ycor = hawk-ycor
          if h >= hawk-heading  [ set heading heading – 60 ]
          else [ set heading heading + 60 ]
END
```

5. EXPERIMENTAL RESULTS

This chapter explains the experimental results for the designed simulation with valid test-case scenarios in place. Here, we demonstrate the simulation, how the model is able to achieve the objectives, and the limitations and performance-related constraints of the model. We start with Objectives 1, 2 and 3 illustrated in Chapter 3, and test each scenario and correlate with the real-world flock nature of the birds.

Netlogo is an interactive tool where the user can create buttons, sliders, and on/off switches for user interaction with the created model. The function of these tools is explained in Chapter 4. Figure 5.1 shows the tools that are used to control the simulation.

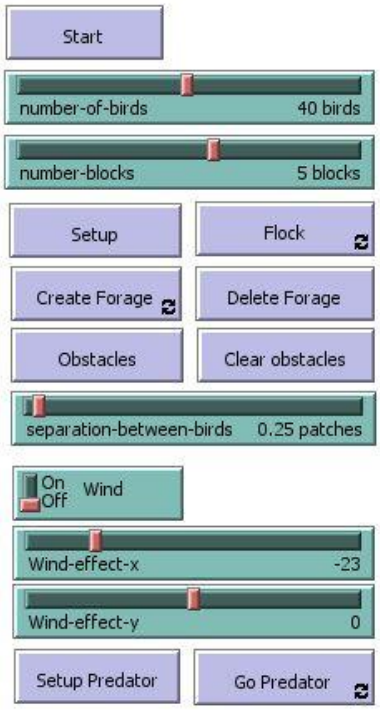


Figure 5.1. Tools of operation.

As shown in Figure 5.1, we start by creating the environment for the burtles (name proposed for the agent set of birds) which is a green background and is set up using the “Start” button. The “Setup” button is used to place the burtles in the environment. Using the “number-

of-birds" slider, the user can select from 1 to 80 burtles. The "Flock" button starts the flocking simulation. Netlogo comes with an option of running a button forever once it is clicked until the button is pressed again. The flock button has a procedure which has to run again and again, hence we check "forever" in the button options. The "Create Forage" and "Delete Forage" buttons are used to create foraging areas for the burtles and to clear them as well. The creation of forage involves an interactive user action, hence the "forever" option is selected for this button. The "Obstacles" and "Clear obstacles" buttons are used to set up obstacles in the environment and to clear them respectively; also, the "number-of-blocks" slider is used to select the number of blocks. The on/off switch is used for turning the wind effect on and off in the environment. Slider bars are provided for controlling the wind effects along the X and Y axes. Finally, the "separation-between-birds" slider helps select the distance between burtles.

5.1. Test Cases

The test cases in this section are designed based on the objectives initially formulated to perform simulations of bird's flocking behavior, obstacle avoidance and wind effect and predator evasion. Initially, the test cases deal with objective 1, then move on to objective 2 and finally, objective 3.

5.1.1. Test case 1

In this test case, the implementation of Objective 1, as stated in Section 3.1, is investigated and verified to see if the agents are performing the flock behavior following the three rules of cohesion, alignment, and separation. A sample of 30 burtles is set up along a circle layout in the 2D world with dimensions of 45x35, and the flock procedure is called. Section 4.2.1.1 explains the algorithm behind the flock procedure, and Figure 5.2 shows a sample flocking performed with a population of 30 birds.

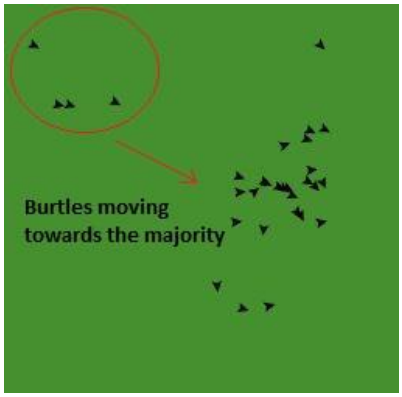
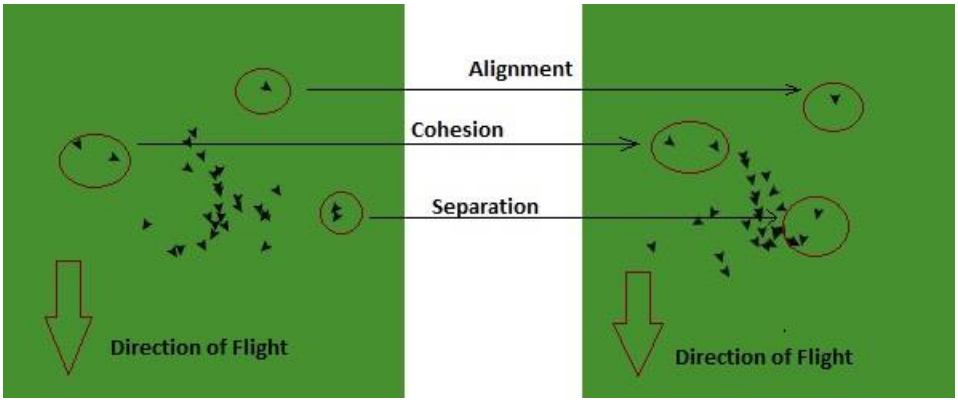


Figure 5.2. Population of 30 burtles performing flock behavior.

Figure 5.2 shows 30 burtles swaying around the space. While performing flock behavior, burtles that are far away from the flock are attracted towards the majority if the majority is in the burtles' vision and tend to move the same direction.



(a) (b)

Figure 5.3. Burtles following the three flocking rules.

Figure 5.3 shows how the burtles try to flock together following the rules of flocking, i.e., alignment, cohesion, and separation. The circled burtles help us understand how these rules are satisfied. The circled burtle labeled "Alignment" has a direction of flight that deviates from the flock in Figures 5.3(a) and (b); it is clearly visible that the direction of the flight is turned towards the flock's heading. The circled burtles labeled "Cohesion" tend to move towards the other burtles which is visible from Figures 5.3(a) and (b). The burtles labeled "Separation" tend

to move away from each other because they are well inside the threshold level of repulsion. In Figure 5.3(a), these burtles, as observed, come close to each other, and in Figure 5.3(b), the same burtles are being separated because they tend to avoid colliding with each other once the separation between them is above the threshold. This behavior of burtles explains how the burtles satisfy cohesion and alignment when the separation between the burtles is below the threshold level and satisfy separation when the separation is above the threshold. The threshold level can be varied using the "separation-between-birds" slider.

5.1.1.1. Performance constraints

Next, a sample of 60 burtles is taken, and the simulation is run at a normal speed of operation; the simulation showed lag in the frame rate while propagating. It has been observed that, as the population size increased, the lag in the frame rate increased, and the response rate of the simulation decreased. For the smooth run of a simulation, a maximum of 80 burtles is considered.

5.1.2. Test case 2

In this test case, the first part of Objective 2, foraging, as explained in Section 3.2.1, has been investigated for how the burtles are driven towards the foraging areas depending on the creation of these areas. The burtles are driven towards high-priority forage areas, and the priority is dependent on how new the forage area is. Initially, after setting up the burtles and calling the "Flock" procedure, the "Create Forage" procedure is called. This function allows the user to create new forage areas by mouse clicks on the surface of the system.

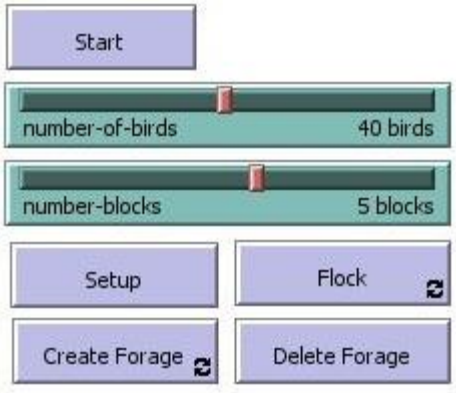


Figure 5.4. Forage creation.

A sample of 40 birds is selected, and the "Flock" procedure is called. As shown in Figure 5.4, the user can utilize the "Create Forage" button to create a forage area by clicking on the simulation space and can clear the same space using the "Delete Forage" button.

Figure 5.5 shows $Forage_{70}$ that was created and the burtles' movement towards this foraging area. Figure 5.6 shows the flock at $Forage_{70}$. Until new forage is created or the forages are cleared, the flock continues to stay at $Forage_{70}$.

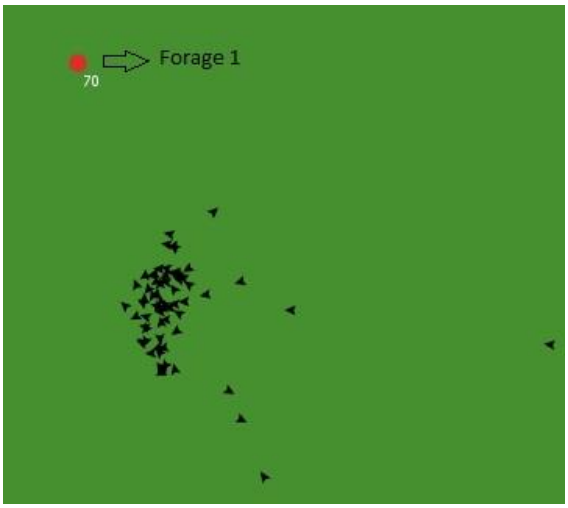


Figure 5.5. Flock driven towards forage.

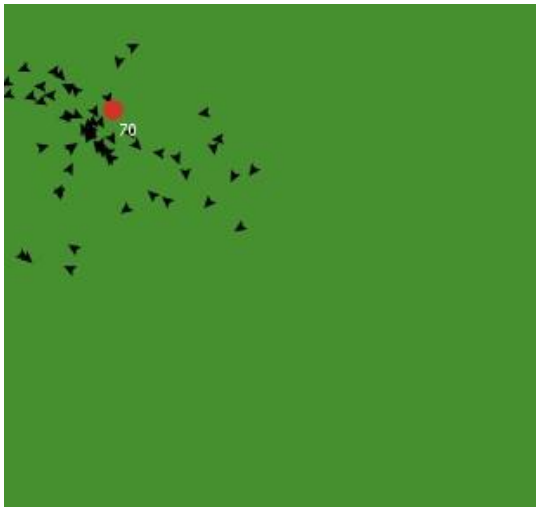


Figure 5.6. Flock at Forage$_{70}$.

If new forage is created, then the flock is driven towards it. Figure 5.7 explains how the flock moves towards the newly created forage area, Forage$_{71}$, from the Forage$_{70}$. The flock continues to stay at Forage$_{71}$ until new forage is created.

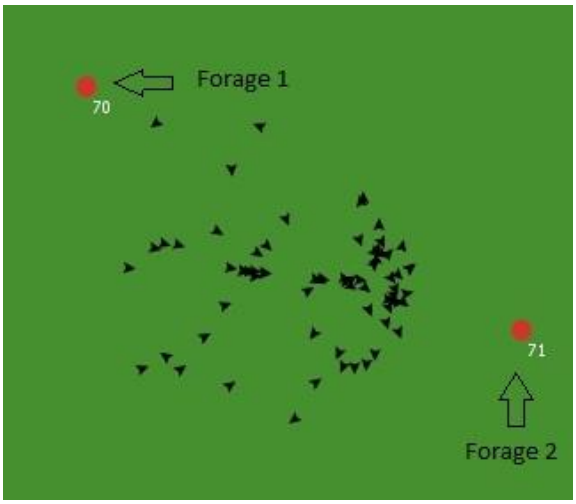


Figure 5.7. Flock propagating towards Forage$_{70}$ from Forage$_{71}$.

5.1.3. Test case 3

In this test case, the second part of Objective 2, obstacle avoidance, as explained in Section 3.2.2 has been investigated. It shows how the burtles satisfy the objective of avoiding obstacles.

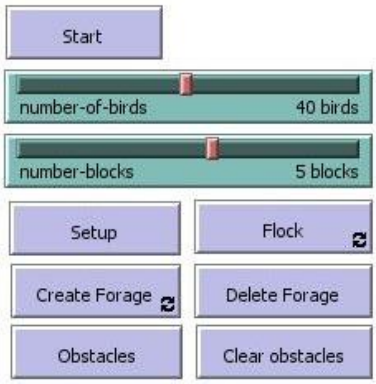


Figure 5.8. Select and set up blocks.

Figure 5.8 shows the "number-blocks" slider which allows the user to select from 1 to 8 blocks. The "Obstacles" blocks button is used to place the blocks randomly in the simulation space, and "Clear obstacles" is used to clear all the obstacles in the path of the burtles. A sample of 40 birds and 5 blocks is selected to place in the environment. "Obstacles" is used to set up these blocks. To explain the obstacle avoidance, forages are created within the environment, and the birds are driven towards these forages, avoiding the obstacles. The algorithm behind the obstacle avoidance is explained in the Section 4.2.2.5. Figures 5.9 (a), (b), and (c) show the propagation of the flock from $Forage_{77}$ to $Forage_{83}$, overcoming the obstructions in the flock's path. Figure 5.9 (a) shows the flock at $Forage_{77;}$ Figure 5.9 (b) shows where new $Forage_{83}$ is created and the flock is driven towards the $Forage_{83}$, avoiding the obstacles. Figure 5.9 (c) shows the flock at $Forage_{83}$.

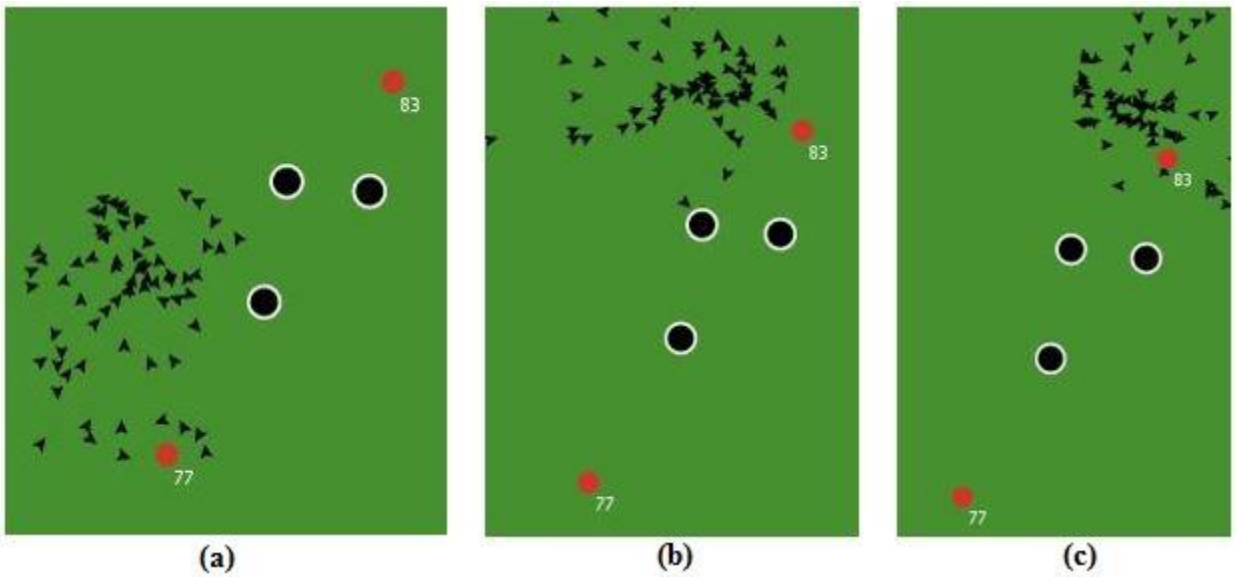


Figure 5.9. Flock moving from Forage$_{77}$ to Forage$_{83}$, avoiding the obstacles.

5.1.4. Test case 4

In this test case, the third part of Objective 2, wind effect, as explained in Section 3.2.3, has been investigated. It shows how the burtles propagate around the world with the wind effect acting on the system.

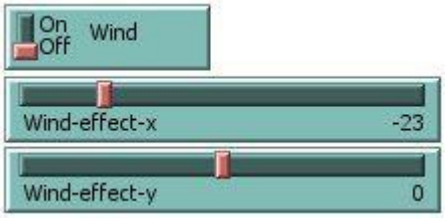


Figure 5.10. Tools used to control the wind effect.

Figure 5.10 shows the wind switch to turn the wind effect on/off. “Wind-effect-x” is used to control the wind effect along the X-axis. “Wind-effect-y” is used to control the wind effect along the Y-axis. The wind effect is attained in the system by manipulating the speed of the burtles based on the values of the wind effects. The lower the value of the wind effect along the axes, the faster the burtles propagate around the world. To explain the effect of wind in the system, we take a single burtle and advance the ticks (quantity of time in Netlogo) when we set

the burtle to flock. Ticks give us the time taken for the burtle to travel from the center of the system to the top boundary of the system. Here, we consider 0 wind effect along the X-axis and a negative value for the wind effect along the Y-axis; for example, we take -26. Because the wind effect is negative, i.e., it is downstream for the burtles traveling upwards, and the speed of the burtles increases as it is supported by the wind, the burtle in this scenario should take less time to reach the top boundary of the system. Figure 5.11 shows that the wind effect is turned on; it is set at -26 for the wind effect along the Y-axis and 0 for the wind effect along the X-axis. First, the initial setup of the burtle at time t_0 = 0 ticks is shown. Figure 5.12 shows the initial setup of the burtle at time 0 ticks.

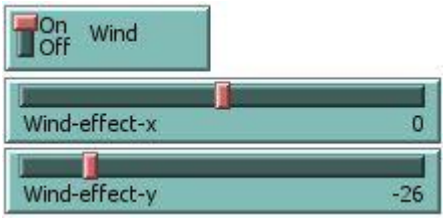


Figure 5.11. Wind effect along the Y-axis set at -26.

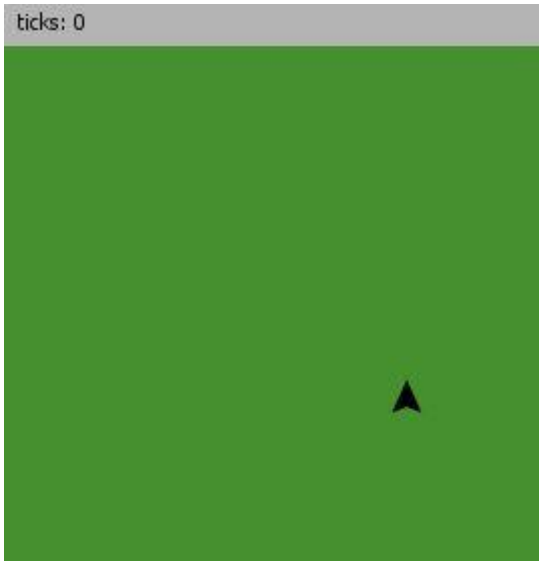


Figure 5.12. Initial setup of the burtle at time t = 0.

Figure 5.13 shows the burtle reaching the top boundary of the system in a time of 23 ticks. The wind factor along the Y-axis is set at -26 but the winf effect is not turned on.

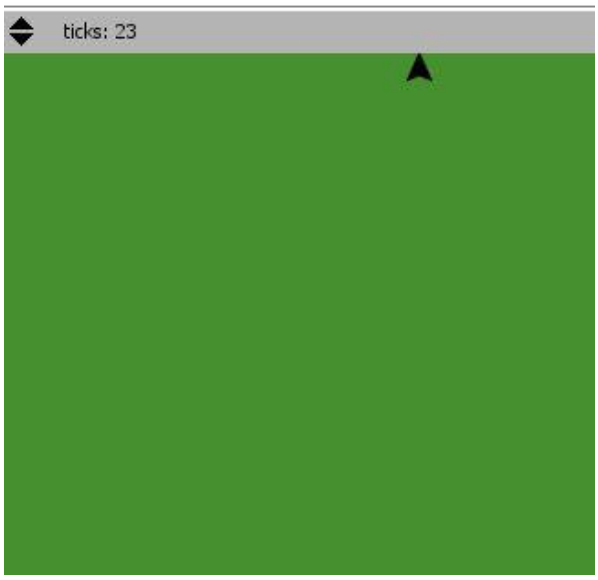


Figure 5.13. Burtle reaching the top at time t = 23 ticks.

Next, the wind effect is turned off in the system, and the behavior of the burtle is studied. Figure 5.14 shows the burtle reaching the top boundary of the system at 29 ticks.

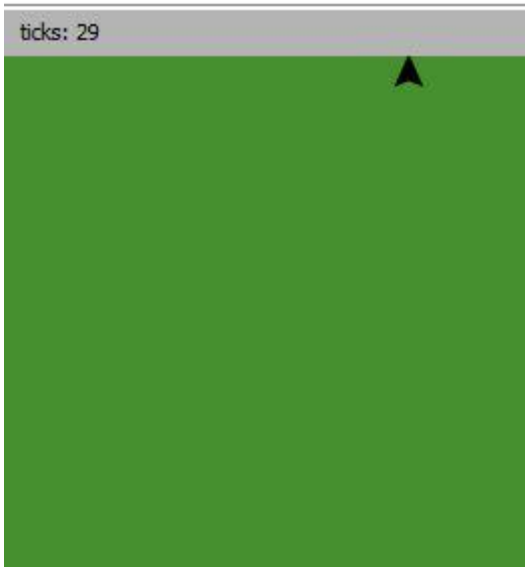


Figure 5.14. Burtle reaching the top at time t = 29 ticks.

From Figure 5.14 it is evident that the wind effect is acting on the system as expected. With the decreased wind effect along the Y-axis, the speed of the burtle propagating from the bottom to the top increases, and the speed of the burtle propagating from the top to the bottom decreases. With the increased wind effect along the Y-axis, the speed of the burtle propagating from the bottom to the top decreases, and the speed of the burtle propagating from the top to bottom increases. Similarly along the X-axis, the speed of the burtle propagating from left to right decreases when the wind effect along the X-axis increases, and the speed of the burtle propagating from right to left increases. As the wind effect along the X-axis decreases, the speed of the burtle propagating from left to right increases, and the speed of the burtle propagating from right to left decreases.

5.1.5. Test case 5

In this test case, the implementation of the Objective 3, predator evasion, is investigated and tested to see how the burtles perform the predator evasion based on the predator's behavior. While performing the flock, if a predator comes into the burtles' vision, the burtles try to get close to each other; as the predator tries to dive into the flock, the flock tries to separate, dodging the predator; and as the predator moves away, the flock tries to get close again. Initially, three dens for the predator are set up, and the burtles try not to move around these dens. Next, when the predator is set for predation, the burtles try to come closer to each other and perform evasion.

Figure 5.15. Setup of predator and the dens.

Figure 5.15 shows the setup of the hawks and their dens. Orange areas show the setup for three dens; the predator is facing the flock, and the burtles are flocking at a distance. Next, the "Go Predator" procedure is called using the "Go Predator" button. Figure 5.16 shows the predator diving into the flock.

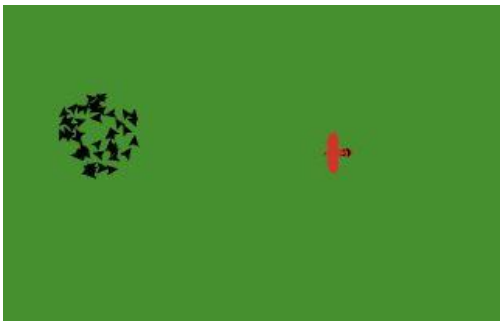


Figure 5.16. Predator approaching the flock.

Figure 5.17 shows how the burtles try to avoid the predator's path of motion. The birds try to separate into several groups as the predator approaches them and try to avoid the contact with the predator.

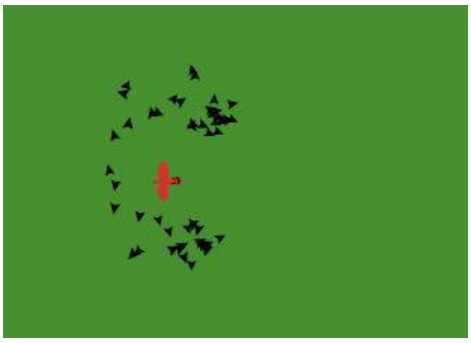


Figure 5.17. Flock separating into groups to avoid the predator.

Figure 5.18 shows how the flock tries to coagulate as the predator moves away from the flock. As the burtles feel that they are at a safer distance from the predator, they try to congregate and perform flocking.

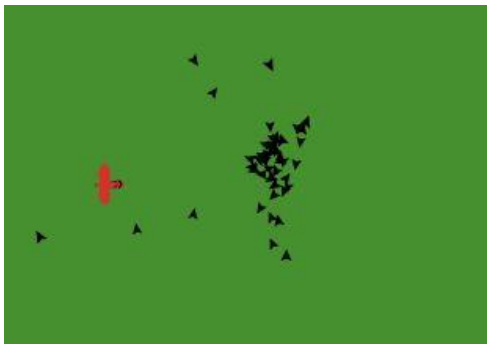


Figure 5.18. Flock aggregations after successful evasion.

5.1.6. Test case 6

In this test case, the implementation of Objective 3, Predator evasion with the wind effect in the environment, is investigated and tested to see how the burtles perform predator evasion based on the predator's behavior in the influence of wind. The flock performed a similar behavior as discussed in Test case 5, but because the wind is acting in the system, the flock

found it difficult to evade the predator's path as the wind effect forced the burtles to fly in a direction to which they did not intend, but the flock managed to make a successful evasion at times.

Figure 5.19. Setup of predator and the dens with the wind turned on.

Figure 5.19 shows the setup of burtles and the predator with the wind effect set at -29 along the X-axis and 21 along the Y-axis. Next, the "Go Predator" procedure is called, and the behavior of the burtles and the predator is studied.

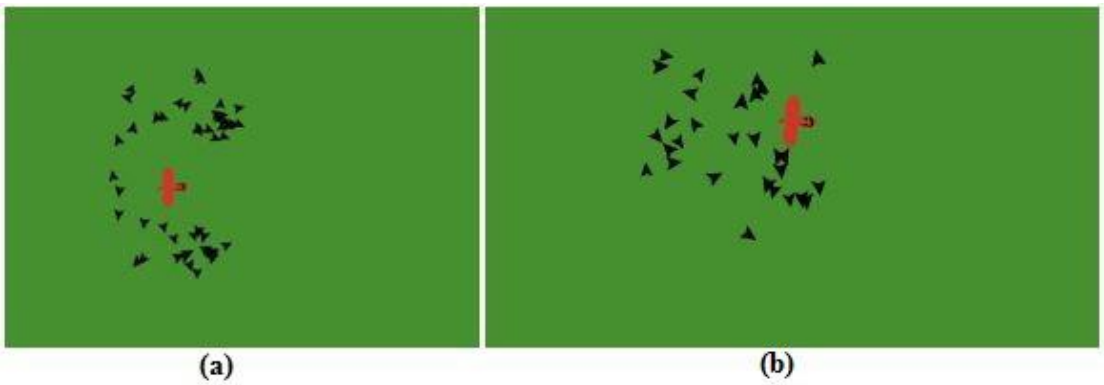


Figure 5.20. Predator evasion without wind effect and with wind effect.

Figures 5.20(a) and (b) show the evasion behavior of the burtles without the wind effect in the system and with the wind effect in the system. Figure 5.20(a) shows the flight of burtles with no wind acting on the system, and the burtles' clear distance management with the predator is visible when compared to Figure 5.20(b) where the distance between the predator and the burtles is small with the wind turned on in the system. Figure 5.21 shows how the flock tries to aggregate after evading the predator.

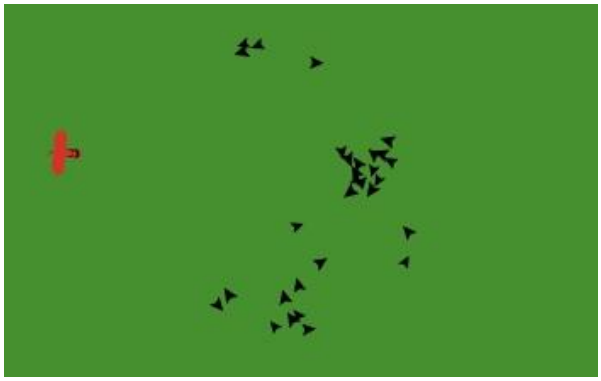


Figure 5.21. Flock aggregations after evasion.

5.1.7. Test case 7

In this test case, we discuss the burtles' implementation of predator evasion in the presence of obstacles. The burtles try not to hit the obstacles, instead swaying around the obstacles, so the predator tries to avoid the obstacles and heads straight to the flock if the flock is in the predator's vision. The following figures, Figure 5.22 and Figure 5.23 show how the predator and the burtles try to avoid obstacles as well as how the burtles escape the predator.

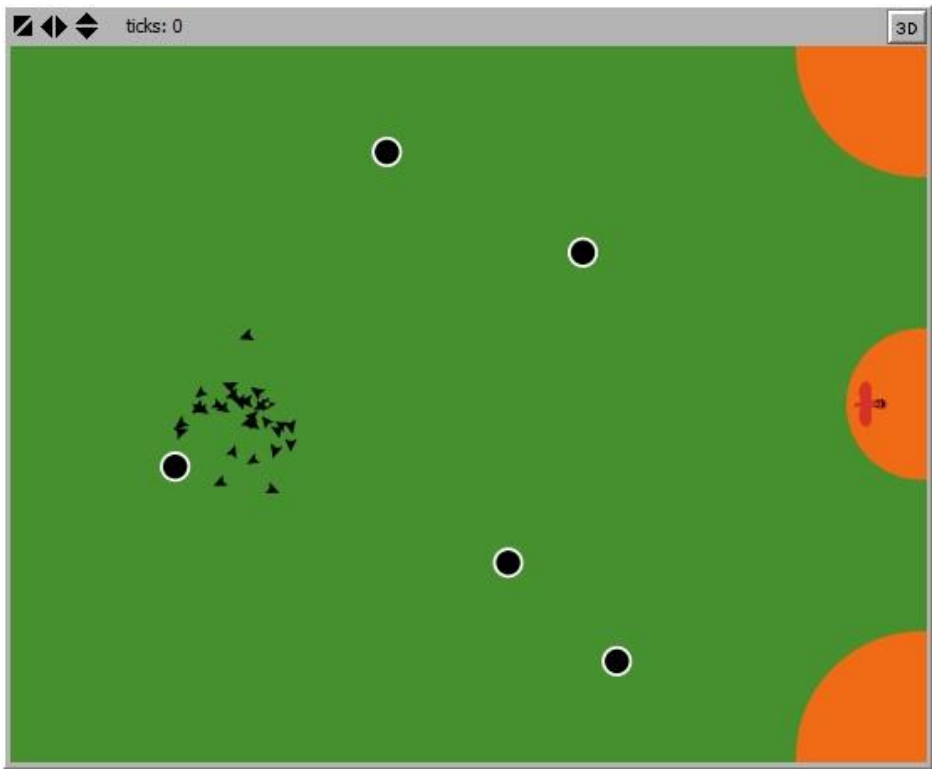


Figure 5.22. Setup of the predator and the obstacles.

Figure 5.22 shows the setup of the obstacles around the environment and the predator facing the flock. Next, the “Go Predator” procedure is called to set the predator for predation. Not only are the burtles trying to avoid the obstacles, even the predator does; therefore, if a block is in the predator’s path, then the predator tries to steer away. The following figures show how the burtles try to escape the predator by avoiding the obstacles.

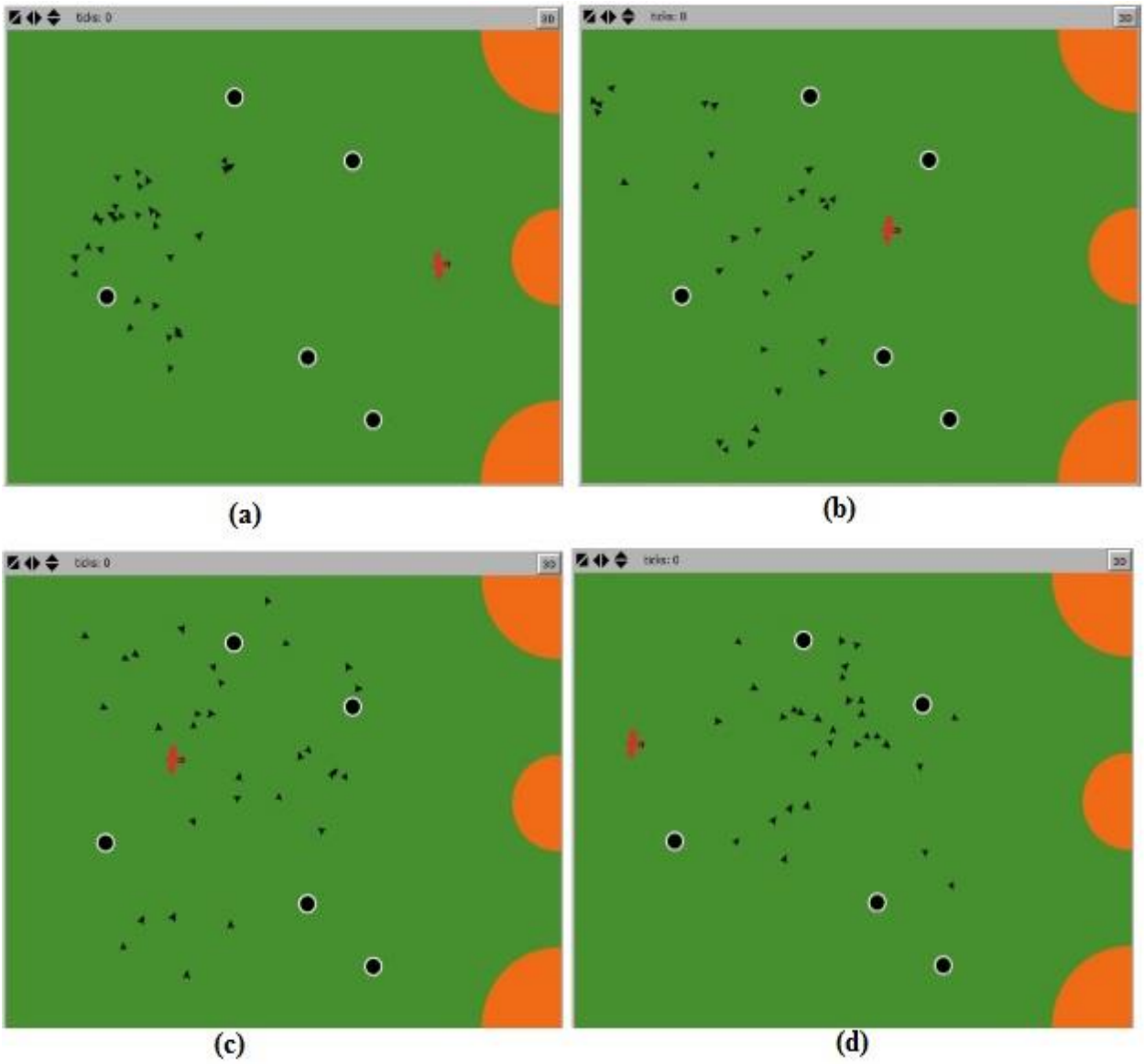


Figure 5.23. Predator approaching the flock and flying away.

Figures 5.23 (a) (b) (c) (d) show the predator approaching the flock as well as the flock trying to avoid the obstacles and escape the predator. In Figures 5.23(a) and (b), the predator approaches the flock; in Figure 5.23(c), the burtles try to escape the predator; and in Figure 5.23(d), the predator moves away from the flock, and the flock tries to aggregate.

5.1.8. Test case 8

In this test case, the implementation of objective 2, avoidance of the obstacle, is tested in the presence of wind. The initial setup is to let the burtles flock around the space; we set up the blocks as obstacles and turn on the wind. The wind is set up at 25 along the X-axis and -14 along the Y-axis. The burtles had no difficulty interpreting the obstacles in front and avoiding the blocks.

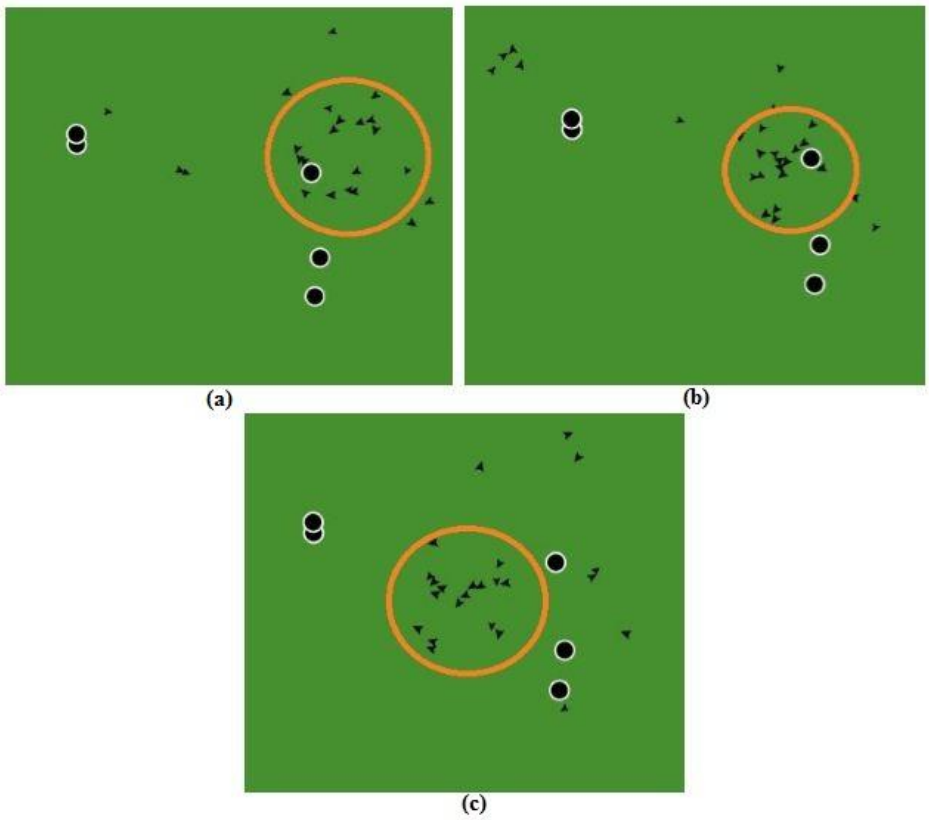


Figure 5.24. Burtles trying to avoid the blocks.

Figure 5.24 shows the burtles circled in orange trying to avoid the block in front. In Figure 5.24(a), the burtles are approaching the obstacle. In Figure 5.24(b), the burtles try to propagate along the sides of the obstacles. In Figure 5.24(c), the burtles have successfully avoided the obstacles and propagated along their path of motion.

5.1.9. Test case 9

In this test case, the implementation of objective 2, driving towards the forage grounds, is tested and studied for how the burtles behave in the presence of wind. The wind is setup at 25 along the X-axis and -14 along the Y-axis. Initially, after setting up the burtles and calling the "Flock" procedure, the "Create Forage" procedure is called. This functionality allows the user to create new forage areas by mouse clicks on the surface of the system.

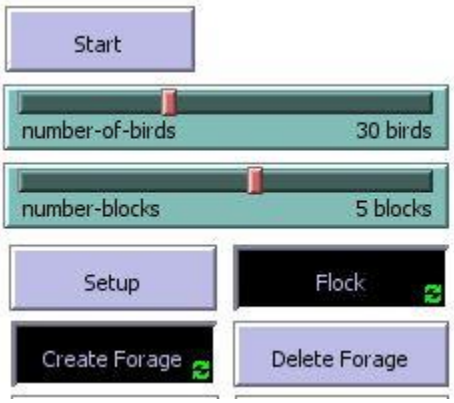


Figure 5.25. Forage creation.

A sample of 30 birds is selected. The "Flock" procedure is called, and as shown in Figure 5.25, the user can click the "Create Forage" button to create a forage area by clicking on the simulation space. The same area can be cleared using the "Delete Forage" button.

Figure 5.26(a) shows $Forage_{36}$ created and the movement of the burtles towards this foraging area. Figure 5.26(b) shows the flock at $Forage_{36}$. $Forage_{38}$ is created in Figure 5.26(c), and the flock navigates towards $Forage_{38}$. In Figure 5.26(d), $Forage_{39}$ is created, and the flock is propagated towards it. The effect of wind was evident in slowing down the flock's motion while propagating to $Forage_{36}$ and from $Forage_{38}$ to $Forage_{39}$. Also, while propagating from $Forage_{36}$ to $Forage_{38}$, the flock showed a reasonable increase in the overall speed.

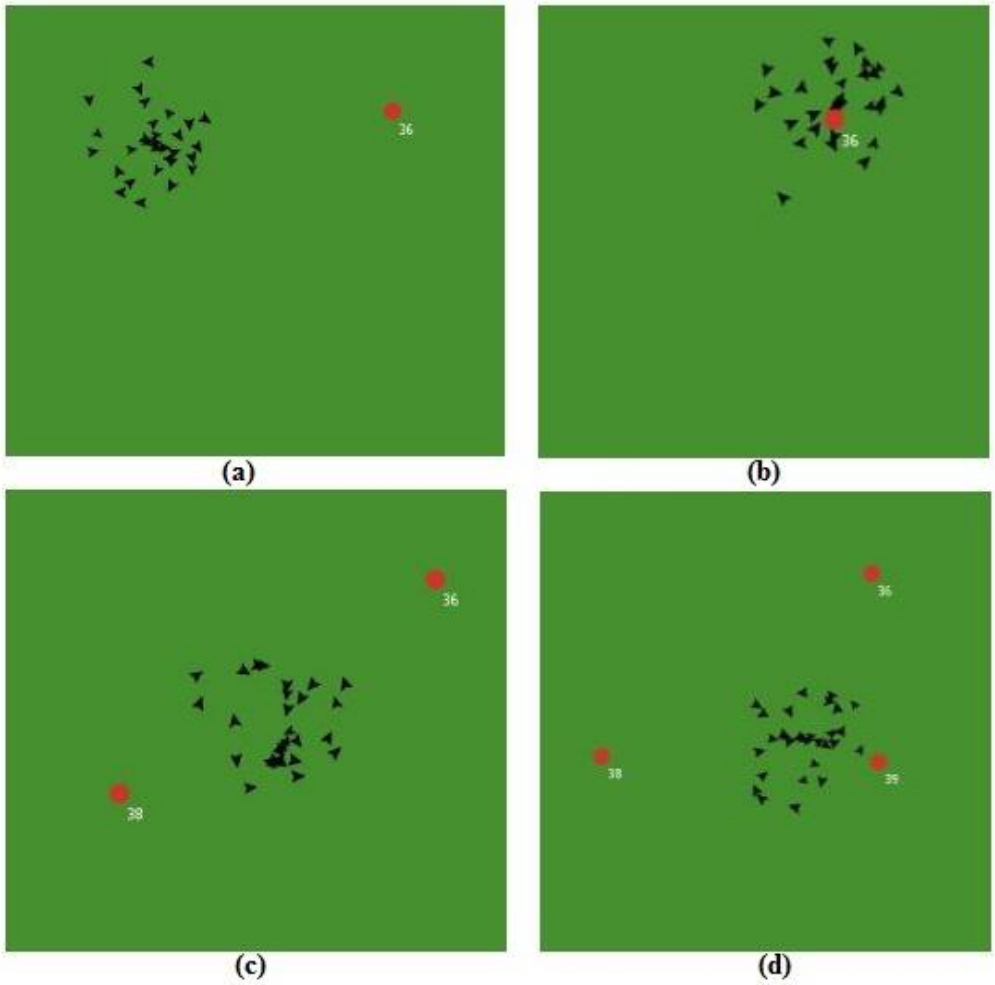


Figure 5.26. Flock navigated to various forages.

5.1.10. Test case 10

In this test case, the implementation of objective 2 on a whole is taken into account; i.e., navigation to forage grounds in the presence of obstacles and wind is tested and studied to see how the flock moves from one forage ground to another in the influence of the wind and avoiding the obstacles. Figure 5.27 shows how the burtles are placed in the environment and the wind effect setup. The wind effect is set up at 25 along the X-axis and -14 along the Y-axis. A population of 30 burtles is set up with 5 blocks, and the wind effect is turned on. A new forage is created by selecting the “Create Forage” button and clicking the simulation surface using the mouse. The burtles are set to flock using the “Flock” procedure.

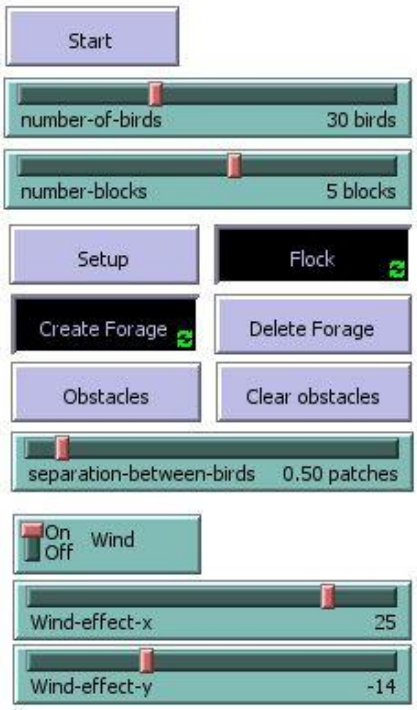


Figure 5.27. Forage, obstacle, and wind setup.

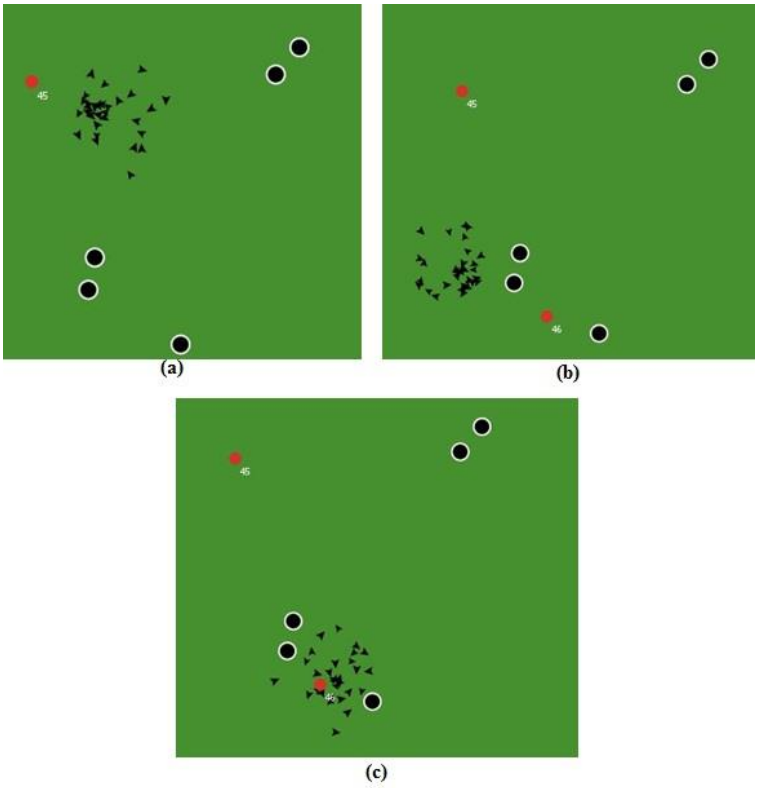


Figure 5.28. Flock propagating from Forage$_{45}$ to Forage$_{46}$.

Figure 5.28 shows flock propagating from $Forage_{45}$ to $Forage_{46}$ under the influence of wind and obstacles. Figure 5.28(a) shows the flock moving towards $Forage_{45}$. Figure 5.28(b) shows how the flock tries to avoid the blocks and move towards $Forage_{46}$. Figure 5.28(c) shows the flock reaching $Forage_{46}$ and staying there until a new forage ground is created or the forages are cleared.

5.1.11. Test case 11

In this test implementation of objectives 2 and 3, predator evasion in the presence of wind and obstacles is tested, and the behavior of the burtles is studied by varying wind effects in the system and changing the obstacles' placement. The wind effect is set up at 14 along the X-axis and 23 along the Y-axis.

Figure 5.29. Tools controlling the simulation.

Figure 5.29 shows the tools of operation for the predator setup. A population of 30 burtles is taken and set to flock with wind and obstacles turned on in the system. “Setup Dens” is used to set up three predator prone zones in the system, and the flock tries to not fly around these dens. “Setup Hawk” is used to set up the predator in one of the dens, facing the flock. As the predator is set up, the burtles come close to each other. Next, the “Go Predator” procedure is called, and the predator is set to dive into the flock avoiding the obstacles. The figures below show how the burtles try to propagate along the world with the wind and obstacles affecting the motion for the burtles and the predator. Figure 5.30 shows the initial set up of the predator, burtles, and obstacles in the system.

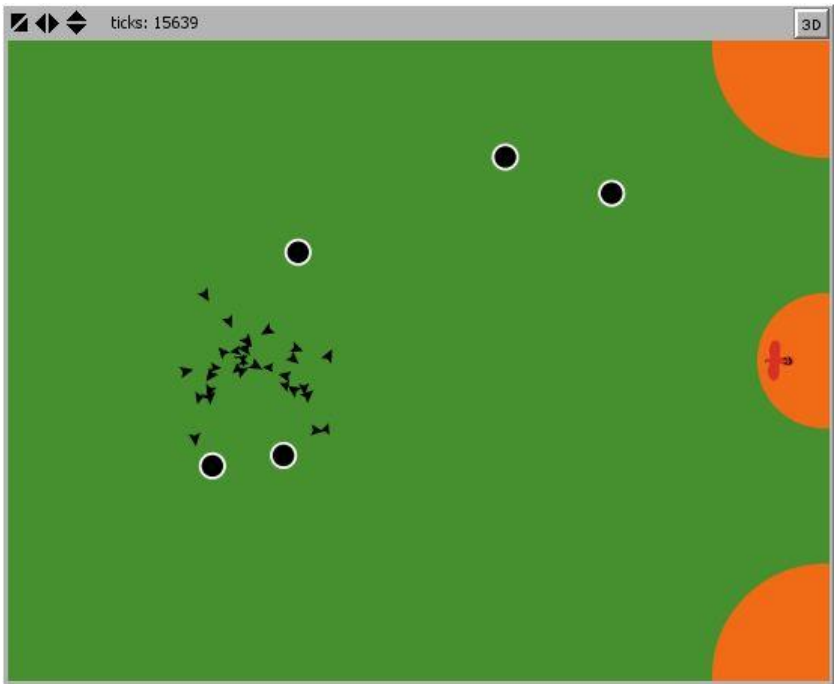


Figure 5.30. Initial setup of burtles, predator and the obstacles.

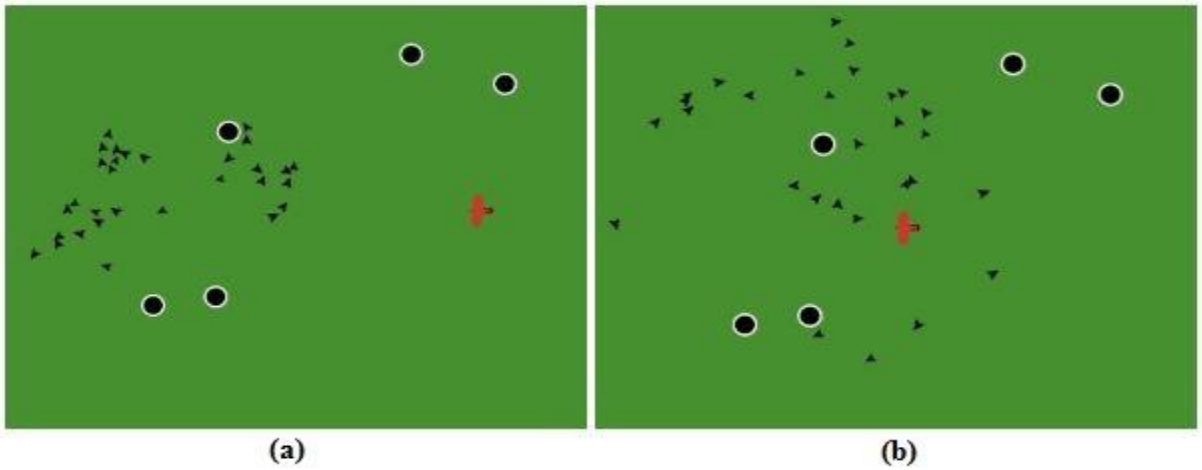


Figure 5.31. Predator approaching the flock.

Figures 5.31(a) and (b) show the predator approaching the flock and the flock reacting to the changing conditions in the world. Figure 5.31(a) shows the predator approaching the flock, and Figure 5.31(b) shows how the flock tries to escape the predator and avoid the blocks with wind acting on the burtles' motion.

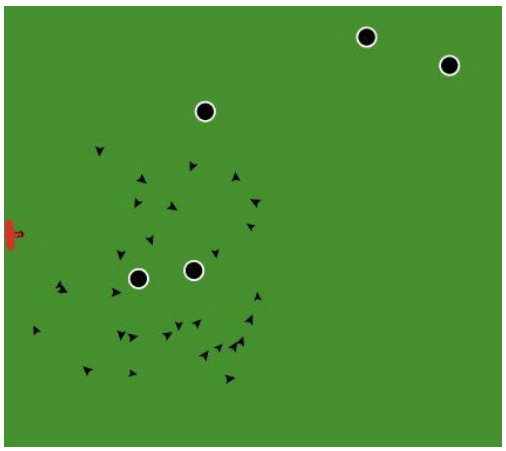


Figure 5.32. Flock aggregation after predator evasion.

Figure 5.32 shows the predator moving away from the burtles and the burtles aggregating after the predator moves away from the burtles. The simulation was successful in performing the expected behavior in the presence of wind and obstacles.

6. CONCLUSION

The research, implementation, and the experimental results from the previous chapters suggest that the performed simulation could achieve the objectives that were set.

1. The model was clearly able to achieve objective 1 which is to develop an artificial flock of birds which flies around the 2D space following the three rules of flocking, separation, cohesion, and alignment, and is evident from Test case 1.
2. Objective 2 was achieved by dividing the objective into three different goals and working on each goal separately. Test cases 2, 3, 4, 8, 9, and 10 showed how the flock was directed towards each forage ground by avoiding the obstacles, and moving with and against the wind based on the flock's direction of propagation. From the test results, it is perceptible that the agents were able to behave accordingly with the changing environmental conditions.
3. Test cases 5, 6, and 7 showed how the flock tried to escape from predation when a predator approached the flock, satisfying objective 3 and how the escape behavior was achieved even under the influence of obstacles and wind factors, respectively.
4. Finally, in Test case 11, an accumulation of objectives 1, 2, and 3 was tested, and the flock was able to demonstrate a reasonable predator evasion in the presence of obstacles and wind at the same time.

Thus, the application was able to achieve the objectives with a limited population of burtles (bird agents).

6.1. Limitations and Future Work

The performance of the application declined with an increased burtle population in the flock, so a limit of 80 is set for selection. It helps in executing the model efficiently. Currently, a

2D simulation is performed with a limited scope of extensibility; e.g., the wind effect is limited to the X and Y axes while predator evasion for the flock separates, and turns left or right along the direction of motion but not up or down.

In future work, the model could include performance tuning with the capacity of running more agents in the system and the capability of 3D rendering, which would help develop more scenarios for a bird flock, such as the wind effect could include the Z-axis along with the X and Y axes, expanding the scope for more natural-looking simulations.

Future work may also include the design of physical robots, which are capable of flying and displaying the characteristics of the simulated agents. This type of design would help the research organizations and other agencies in developing spy agents to help protect the realms from external threats.

REFERENCES

[1] J. Hagey, "Flocking Birds and Schooling Fish," *Biods,* vol. 2, p. 21, 2000.

[2] R. Parent, "Computer Animation: Algorithms and Techniques," San Francisco, Morgan Kauffman, 2002, pp. 246 - 258.

[3] W. K. Potts, "The chorus-line hypothesis of coordination in avian flocks," *Nature,* vol. 309, pp. 344 - 345, 1984.

[4] G. M. Hilton, W. Cresswell and G. D. Ruxton, "Intraflock variation in the speed of escape-flight response on attack by an avian predator," *Behavioral Ecology,* vol. 10, no. 4, pp. 391-395, 1998.

[5] C. W. Reynolds, "Flocks, Herds, and Schools: A Distributed Behavioral Model," *Computer Graphics,* vol. 4, no. 21, pp. 25-34, 1987.

[6] P. Maes, "Artificial Life meets Entertainment: Lifelike Autonomous Agents," *Communications of the ACM,* vol. 38, no. 11, pp. 108-113, 1995.

[7] U. Wilensky, "NetLogo Flocking model," Center for Connected Learning and Computer-Based Modeling, Northwestern University, 1988. [Online]. Available: http://ccl.northwestern.edu/netlogo/models/Flocking.

[8] S. Carlson, "Artificial Life: Boids of a Feather Flock Together," *Scientific American Magazine,* vol. 283, no. 5, pp. 112, 114, 2000.

[9] G. Y. Titelman, "Random House Dictionary of Popular Proverbs and Sayings," vol. 8, Random House Reference, 1996, p. 31.

[10] J. T. Emlen Jr., "Flocking Behavior in Birds," vol. 69, pp. 160-170, 1952.

[11] I. L. Bajec and F. H. Heppner, "Organized flight in birds," *Animal Behaviour,* vol. 78, no. 4, pp. 777-789, 2009.

[12] A. Okubo, "Dynamical aspects of animal grouping: Swarms, schools, flocks, and herds," *Advances in Biophysics,* vol. 22, pp. 1-94, 1986.

[13] T. Vicsek and A. Czir'ok, "Collective behavior of interacting self-propelled particles," *Physica A: Statistical Mechanics and its Applications,* vol. 281, no. 1-4, pp. 17-29, 2000.

[14] I. D. Couzin, J. Krause, R. James, G. D. Ruxton and N. R. Franks, "Collective Memory and Spatial Sorting in Animal Groups," *J. theor. Biol,* vol. 218, pp. 1-11, 2002.

[15] T. Vicsek, A. Czir´ok, E. Ben-Jacob, I. Cohen and O. Shochet, "Novel type of phase transition in a system of self-driven particles," *Physical Review Letters,* vol. 75, no. 6, p. 1226–1229, 1995.

[16] C. Moeslinger, T. Schmickl and K. Crailsheim, "A Minimalist Flocking Algorithm for Swarm Robots," in *Advances in Artificial Life. Darwin Meets von Neumann*, vol. 5778, Springer Berlin Heidelberg, 2009, pp. 375-382.

[17] T. Oboshi, S. Kato, A. Mutoh and H. Itoh, "A Simulation Study on the Form of Fish Schooling for Escape from Predator," *Forma,* vol. 18, no. 2, pp. 119 - 131, 2003.

[18] E. Bonabeau, "Agent-based modeling: Methods and techniques for simulating human systems," *PNAL,* vol. 99, no. 3, pp. 7280-7287, 2002.