Massive Multi-Agent Flocking Simulation in 3D

Masaru Aoyagi and Akira Namatame

National Defense Academy
Japan

Abstract

The challenge is how to give the creatures the ability to follow spatial restriction while keeping the complexity low enough to still allow for real-time simulation of the herd. This method involves making modifications to Reynolds' flocking algorithm. We would like the herd to move in natural-looking paths. Also, we would like the creatures to travel smoothly in 3D space, with speed regulation in curve.

Keywords: Flocking Behavior, Spatial Restriction, Individual Based Model

1. Introduction

To creatures a large number of objects by hand would be a tedious job. To make the job easier, we would like to try to automate as much of the process as possible. For the case of flocks or herds of creatures, Craig Reynolds [1,2] introduced a simple agent-based approach to animate a flock of creatures through space. In this method, each creature makes its own decisions on how to move, according to a small number of simple rules that consider the neighboring members of the flock. Reynolds suggested that further modifications could lead to a herd model by giving the creatures the ability to follow spatial restriction.

The challenge is how to give the creatures the ability to follow spatial restriction while keeping the complexity low enough to still allow for real-time simulation of the herd. We would like the herd to move in natural-looking paths. Also, we would like the creatures to travel smoothly in 3D space, with speed regulation in curve.

Increasing use of autonomous unmanned air vehicles in a variety of civil and military applications is putting increasing pressure on traditional airspace management capabilities. One solution to this problem is decentralize the management function by delegating control such that individual aircraft manage their own airspace by negotiating with neighboring aircraft, i.e. free-flight. For example, two aircraft on a potential collision course will negotiate changes to their respective flight plans to remove the risk of collision. This negotiation takes place independently of a ground air traffic controller. However, for very high airspace densities, real time negotiation of alternative flight paths becomes impractical and it becomes necessary to impose a set of flight rules that minimizes the probability of collisions a priori.

In nature, aggregations of large numbers of mobile organisms are also faced with the problem of organizing themselves efficiently. This selective pressure has led to the evolution of behavior such as flocking of birds, swarming of insects, herding of land animals and schooling of fish. The reasons why organisms form flocks are varied and include protection from predation, improved food search and improved social cohesion. However, the actual dynamics of the flocking behavior are essentially constrained by the dynamics of the individual organisms and the flock is relatively limited in the types of behavior it can exhibit. This gives a flock of a given organism, be it fish or bird, its characteristic look and feel.

2. Modeling of Flocking Behavior

A flock may be loosely defined as a clustered group of individuals with a common velocity vector. Note that flocking of aircraft is different from formation flying. In the latter, aircraft are arranged according to predefined relationships that generally remain fixed during the flight. With flocking flight, there are no predefined relationships and the flock members may constantly change their position within the group. The fixed relationships within aircraft formations make them relatively difficult to maneuver, whereas the fluid nature of a flock allows relatively rapid changes in flock direction.

The aim of the present work is to demonstrate flocking behavior of a group of simulated unmanned air
developing realistic motions of groups of 'actors' in from simple rules was first demonstrated by Reynolds few of these efforts follows. A brief review of a requirements, their method of behavioral integration, and robots may be distinguished by their sensing and robots. Approaches to formation generation in community have drawn from these biological studies to robotics researchers and those in the artificial life agents could similarly benefit from formation tactics, members of the group [4]. Since groups of artificial schooling show that these behaviors emerge as a maximum the chance of detecting predators or to more efficiently forage for food. Studies of flocking and schooling show that these behaviors emerge as a combination of a desire to stay in the group and yet simultaneously keep a separation distance from other members of the group [4]. Since groups of artificial agents could similarly benefit from formation tactics, robotics researchers and those in the artificial life community have drawn from these biological studies to develop formation behaviors for both simulated agents and robots. Approaches to formation generation in robots may be distinguished by their sensing requirements, their method of behavioral integration, and their commitment to preplanning. A brief review of a few of these efforts follows.

The viability of obtaining coherent flocking behavior from simple rules was first demonstrated by Reynolds [1]. The primary application for this work was in developing realistic motions of groups of 'actors' in computer animation. Conventionally, each 'actor' (known generically as a 'boid' in flocking work) has a scripted path predetermined by the animator. For large numbers of boids, for example a flock of birds, the process was cumbersome and did not produce realistic results. This led to the use of relatively simple flocking rules that would automatically govern the dynamic behavior.

Improvements to this approach have recently been made by Tu and Terzopoulos and separately by Brogan and Hodgins. Tu and Terzopoulos [5] developed more realistic simulated fish schooling by accurately modeling the animals' muscle and behavioral systems. Brogan and Hodgins [6] developed a system for realistically animating herds of one-legged agents using dynamical models of robot motion. Both results are more visually realistic than Reynolds' because they simulate the mechanics of motion; Reynolds' approach utilized particle models only.

The individual components of Reynolds' flocking and Brogan's herding behaviors are similar in philosophy to the motor schema paradigm used here, but their approaches are concerned with the generation of visually realistic flocks and herds for large numbers of simulated animals, a different problem domain than the one this article addresses. In contrast, our research studies behaviors for a small group (up to four) of mobile robots, striving to maintain a specific geometric formation.

The dynamics and stability of multi-robot formations have drawn recent attention [7], [8]. Wang [7] developed a strategy for robot formations where individual robots are given specific positions to maintain relative to a leader or neighbor. Sensory requirements for these robots are reduced since they only need to know about a few other robots. Wang's analysis centered on feedback control for formation maintenance and stability of the resulting system. It did not include integrative strategies for obstacle avoidance and navigation. In work by Chen and Luh [8] formation generation by distributed control is demonstrated. Large groups of robots are shown to cooperatively move in various geometric formations. Chen's research also centered on the analysis of group dynamics and stability, and does not provide for obstacle avoidance. In the approach forwarded in this article, geometric formations are specified in a similar manner, but formation behaviors are fully integrated with obstacle avoidance and other navigation behaviors.

Mataric has also investigated emergent group behavior [9], [10]. Her work shows that simple behaviors like avoidance, aggregation and dispersion can be combined to create an emergent flocking behavior in groups of wheeled robots. Her research is in the vein of Reynolds' work in that a specific agent's geometric position is not designated. The behaviors described in this article differ in that positions for each individual robot relative to the group are specified and maintained.

Related papers on formation control for robot teams include [11], [12], [13], [14]. Parker's thesis [11] concerns the coordination of multiple heterogeneous robots. Of particular interest is her work in implementing "bounding over watch," a military movement technique for teams of agents; one group moves (bounds) a short distance, while the other group over watches for danger. Yoshida [12], and separately, Yamaguchi [13], investigate how robots can use only local communication to generate a global grouping behavior. Similarly, Gage [14] examines how robots can use local sensing to achieve group objectives like coverage and formation.
Parker simulates robots in a line-abreast formation navigating past waypoints to a final destination [15]. The agents are programmed using the layered subsumption architecture [5]. Parker evaluates the benefits of varying degrees of global knowledge in terms of cumulative position error and time to complete the task. Tucker and Ronald [17] presented reactive behavior for formation keeping and demonstrated successfully outdoors on cars.

3. Individual Based Simulation for Flocking Behavior

A method is presented for flocking behavior of creatures, birds, fishes and so on, that can form herds by evading obstacles in airspace, terrain or ocean floor topography in 3D space while being efficient enough to run in real-time. This method involves making modifications to Reynolds' flocking algorithm [2] as following.

- Cohesion: steer to move toward the average position of local flockmates
- Separation: steer to avoid crowding local flockmates
- Alignment: steer towards the average heading of local flockmates

Cohesion behavior gives a boid (outlined triangle located in the centre of the diagram) the ability to cohere with (approach and form a group with) other nearby boids. See Fig.1. Steering for cohesion can be computed by finding all characters in the local neighborhood (as described above for separation), computing the average position (or center of gravity) of the nearby characters. The steering force can be applied in the direction of that average position (subtracting our character position from the average position, as in the original boids model). The flocking rules used in the present work are illustrated schematically. The cohesion rule, acts such that the active flock member (outlined triangle located in the centre of the diagram) tries to orient its velocity vector in the direction of the centroid (average spatial position) of the local flock. The degree of locality of the rule is determined by the sensor range of the active flock member, represented diagrammatically by the light colored circle. Note that for effective cohesion it is necessary to vary the speed of the active flock member as well as its heading such that the speed of boids far from the flock centroid is increased. This is referred to as speed cohesion and allows wayward boids to catch up with the rest of the flock.

Separation behavior gives a boid the ability to maintain a certain separation distance from others nearby. This can be used to prevent boids from crowding together. To compute steering for separation, first a search is made to find other characters within the specified neighborhood. This might be an exhaustive search of all characters in the simulated world, or might use some sort of spatial partitioning or caching scheme.
to limit the search to local characters. For each nearby character, a repulsive force is computed by subtracting the positions of our character and the nearby character, normalizing, and then applying a $1/r$ weighting. (That is, the position-offset vector is scaled by $1/r^2$.) Note that $1/r$ is just a setting that has worked well, not a fundamental value. These repulsive forces for each nearby character are summed together to produce the overall steering force. See Fig.2.

Alignment behavior gives a character the ability to align itself with (that is, head in the same direction and/or speed as) other nearby characters, as shown in Fig.3. Steering for alignment can be computed by finding all characters in the local neighborhood (as described above for separation), averaging together the velocity (or alternately, the unit forward vector) of the nearby characters. This average is the desired velocity, and so the steering vector is the difference between the average and our boid's current velocity (or alternately, its unit forward vector). This behavior will tend to turn our character so it is aligned with its neighbors.

Simulations show that reasonable flocking behavior can be obtained using just cohesion and alignment rules.

Left unchecked, the cohesion rules will tend to lead to flock overcrowding. To balance this, a separation rule is used, where the active flock member tries to translate away from the local flock centroid. Note that for effective flocking behavior, the sensor range of the cohesion rules will generally be much larger than the separation rule, i.e. cohesion acts at a global level whereas separation works locally.

Each boid has direct access to the whole scene's geometric description, but flocking requires that it react only to flockmates within a certain small neighborhood around itself. The neighborhood is characterized by a distance (measured from the center of the boid) and an angle, measured from the boid's direction of flight. Flockmates outside this local neighborhood are ignored. The neighborhood could be considered a model of limited perception (as by fish in murky water) but it is probably more correct to think of it as defining the region in which flockmates influence a boids steering.

The modifications use only local properties of the spatial restriction, and thus have low complexity and wide application. The flocking algorithm with these modifications produces naturally behaving herds that follow the spatial restriction.

The evading obstacles rule added to the flocking algorithm has a constant parameter that can be adjusted to produce different behaviors.

The flocking algorithm works as follows: For a given boid, centroids are calculated using the sensor characteristics associated with each flocking rule. Next, the velocity vector the given boid should follow to enact the rule is calculated for each of the rules. These velocity vectors are then weighted according to the rule strength and summed to give an overall velocity vector demand. Finally, this velocity vector demand is resolved in to a heading angle, pitch attitude and speed demand, which is passed to the control system. The control system then outputs an actuator vector that alters the motion of the aircraft in the appropriate manner.

4. Simulation

3D simulation has to be for flying behavior. We develop 3D simulation using Java 3D. Java 3D is an addition to Java for displaying three-dimensional graphics.

Sometimes, unmanned vehicles fly through spatial restriction. There will be the unmanned vehicle's missions of flying low in the mountain area. We study that Unmanned Vehicles fly through spatial restriction as they continue flocking. A flock often breaks out by avoiding an obstacle (Fig.5). We improve composition of flocking rules, cohesion, separation and alignment. Fig.6 is the simulation result of this case. We also study about flocking behavior as boid has parallel intentions. Fig.7 is this case. In this case, boid has two intentions. They are flocking behavior and heading for destination. We combine the two intentions with a probabilistic method. Fig.8 is the simulation result of this case.

Simulation results will be presented to show the effect of changing rule weightings. The baseline test case is a flock initially has orientations. Heading, attitude and velocity guidance demand vectors for each boid based on flocking rules are calculated. Note simulation time increases linearly with increasing guidance update rate and exponentially with number of boids.

The two most important attributes of a flock were found to be its density (how close its members are to each other) and the degree of alignment of member velocity vectors. For the present work flock density is represented by the mean radius between flock members, and flock alignment by the standard deviation of the flock member heading angles.

![Fig.5 A Flock breaks up by avoiding an obstacle.](image-url)
on the nature of the flight control system available on the target flight vehicle. We will evolve flying through spatial restriction. We want to manage flying through maze in future. (Fig. 9)

5. Conclusions

Flocking offers a potentially simple and efficient way of managing the flight paths of a large number of small autonomous UAVs such that the risk of collision and/or the need for evasive maneuvers is reduced. The way in which flocking rules are implemented depends strongly

References


