
Flocking and Steering Behaviors

Outline

- Real Flocks
- Particle Systems to Modified Models
- Flocking Behaviors
  - Separation
  - Cohesion
  - Alignment
- Additional Steering Behaviors
  - Obstacle Avoidance
  - Goal Seeking
- Forces
- Orientation
-
-

Real Flocks and Schools

- No upper bound on size
  - 17 mile schools of herring with millions of fish
  - Localized reasoning
- Collision avoidance
- Centering
  - Protection from predators
  - Social advantages
  - Better search

Our Foreflocks

- Algorithmically-simulated flocking using a “force field” implementation (SIGGRAPH Electronic Theater 1985)
- “Flocks, Herds, and Schools: A Distributed Behavioral Model” by Craig Reynolds (SIGGRAPH 1987)
  - Defined the popular “Boids” model for flocking.
- “Steering Behaviors for Autonomous Characters” Reynolds (GDC 1999)
  - Summarized navigational and steering behaviors (including flocking).

Basic Boids

- The term "boid" is used to describe a flock member.
- Can boids be particles?
  - To some extent, yes:
    - A boid has an internal state (position, velocity, mass).
    - Can be represented as a Newtonian particle in the particle system implementation previously described in lecture.

$$\frac{d}{dt}\begin{bmatrix} x \\ \dot{x} \end{bmatrix} = \begin{bmatrix} \dot{x} \\ f/m \end{bmatrix}$$

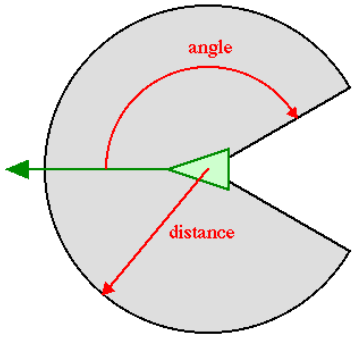
- Is this sufficient?

A Better Model

- Several differences between particles and boids:
  - A boid is not a uniform point. Specifically, it has a complex geometric state and orientation.
  - More complex behaviors
  - Boid behavior is dependant on internal *and* external state.
    - Internal state: particle parameters
    - External state: knowledge about other flock members.
- Key idea: Local rules lead to compelling flock behavior.
  - Boids only have a local (limited) knowledge of the flock. All rules take advantage of this local knowledge.

External State

- A boid also has some notion of "external" state.
- A neighborhood, or field of view, is generally used to describe the range of a boid's perception. Most behavioral rules apply based on conditions in the neighborhood.



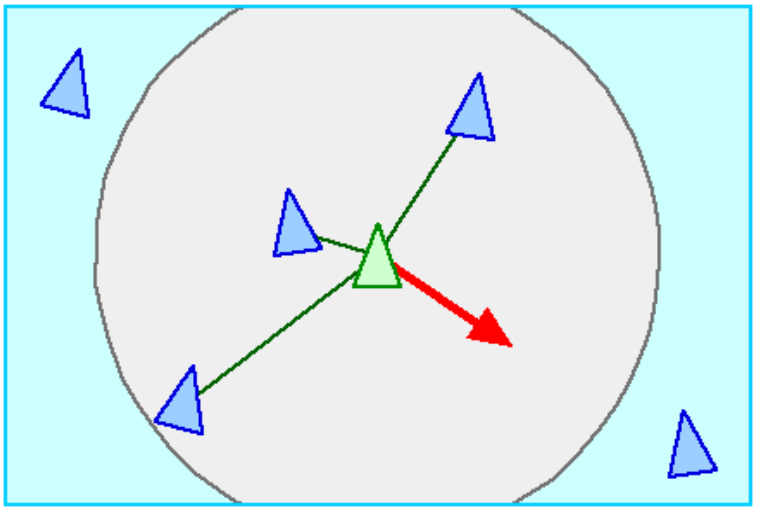
- Note: for Project 5, you can approximate the neighborhood as a sphere to avoid complex geometry intersections.

Steering Rules

- Steering behaviors formulated as rules:
- Concerned primarily with five (for Project 5 at least)
  - 3 original flocking rules
    - Separation
    - Cohesion
    - Alignment
  - 2 additional steering rules
    - Obstacle Avoidance
    - Goal-Seeking (“seek”)
- Represent these as dynamic forces in a modified particle system.

Separation

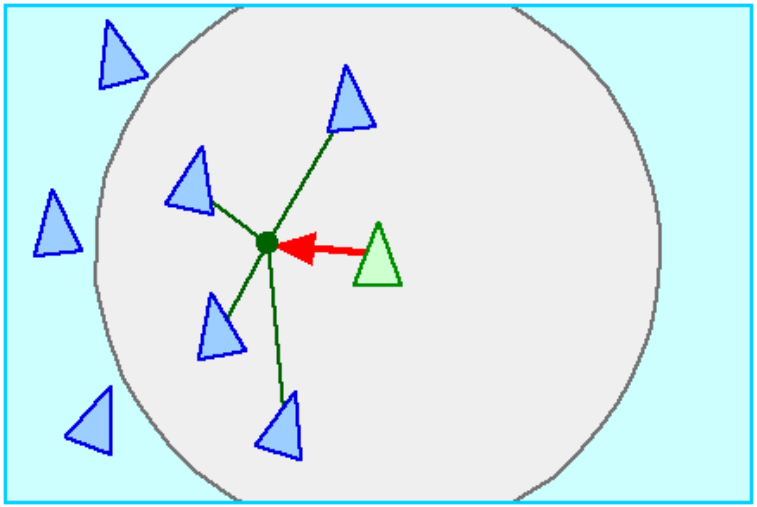
- Pushes boids apart to keep them from crashing into each other by maintaining distance from nearby flock mates.
- Each boid considers its distance to other flock mates in its neighborhood and applies a repulsive force in the opposite direction, scaled by the inverse of the distance.



(blackboard math)

Cohesion

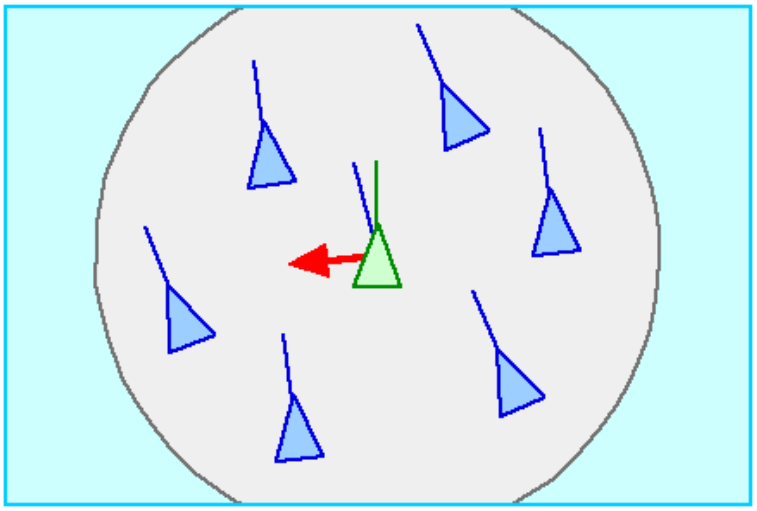
- Keeps boids together as a group.
- Each boid moves in the direction of the average position of its neighbors.
- Compute the direction to the average position of local flock mates and steer in that direction.



(blackboard math)

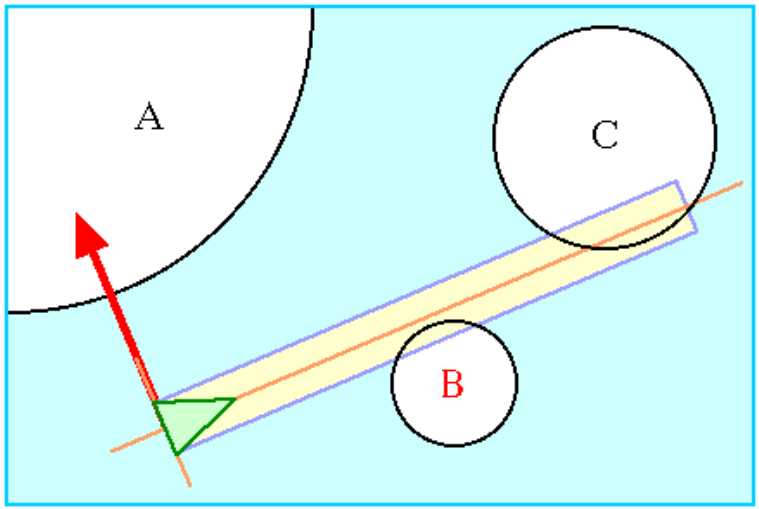
Alignment

- Drives boids to head in the same direction with similar velocities (velocity matching).
- Calculate average velocity of flock mates in neighborhood and steer towards that velocity.



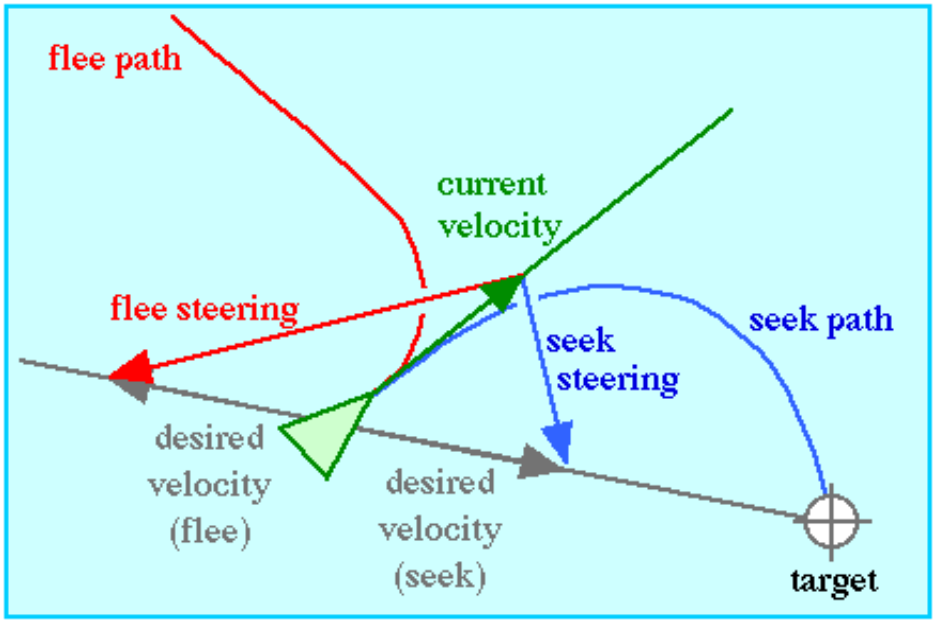
Obstacle Avoidance

- Allows the flock to avoid obstacles by steering away from approaching objects.
- Reynolds uses the method shown below:
  - Assume a cylindrical line of sight
  - Compute cylinder-sphere intersection and veer away from any objects in path.



Goal Seeking

- Drives the flock in the direction of a target/goal.
- Each boid determines the direction to the goal and then steers in that direction
- (Note: this is basically the same as cohesion).

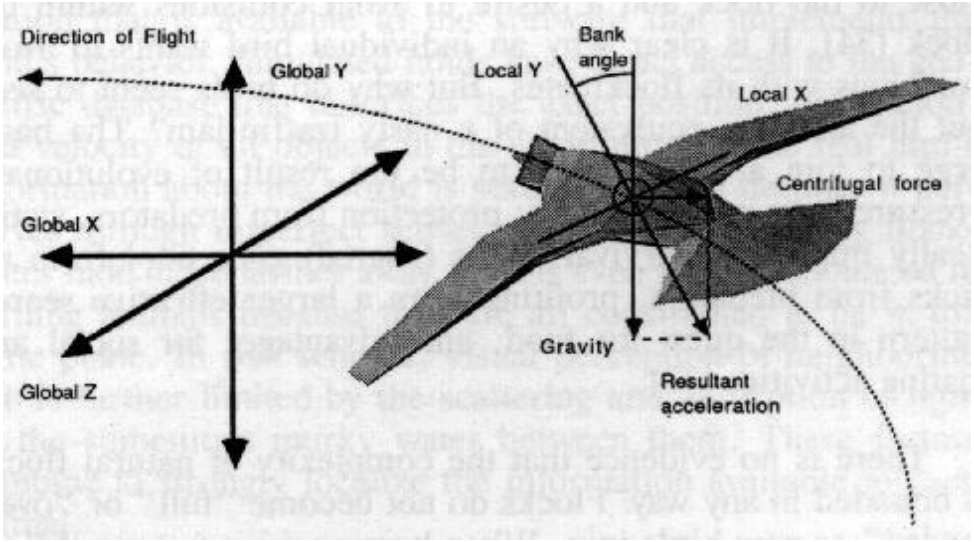


Force Ordering Scheme

- Behaviors can be assigned priorities (in order of increasing priority):
  - Alignment
  - Cohesion
  - Goal-seeking
  - Separation
  - Obstacle Avoidance
- Forces can be given priority (higher priority forces can cancel out lower priority ones).
- Note: for Project 5 combine these into a force accumulator and integrate!
  - Simple (potentially cleverer ways of combining forces)

Orientation

- One last thing to consider is orientation.
- Since a boid has a (generally) non-uniform geometry, we want it to change orientation and smoothly display behaviors, such as banking.
  - For banking, we want to adjust the object's roll (modify local x, y axes).
  - To solve for the new up-vector (y-axis), we take a weighted sum of the resultant acceleration (due to centrifugal force and gravity) and the previous up-vector.



(blackboard math)

- Note: for project 5, you will be required to handle banking

.

Exercise: BOID FLOCKING 알고리즘을 Unity3d 를 이용해 구현하기

```
initialise_positions()
    LOOP
        draw_boids()
        move_all_boids_to_new_positions()
     END LOOP
```

```
PROCEDURE move_all_boids_to_new_positions()
    Vector v1, v2, v3
    Boid b

    FOR EACH BOID b
      v1 = rule1(b)
      v2 = rule2(b)
      v3 = rule3(b)
      b.velocity = b.velocity + v1 + v2 + v3
      b.position = b.position + b.velocity
    END END PROCEDURE
```

```
PROCEDURE rule1(boid b_J)
   Vector pc_J
  FOR EACH BOID b
     IF b != b_J THEN pc_J = pc_J + b.position
     END IF
  END
  pc_J = pc_J / N-1
  RETURN (pc_J - b_J.position) / 100

 END PROCEDURE
```

```
PROCEDURE rule2(boid b_J)
  Vector c = 0;
  FOR EACH BOID b
     IF b != b_J THEN
          IF |b.position - b_J.position| < 100
            THEN c = c - (b.position - b_J.position)
          END IF
     END IF
  END
  RETURN c

END PROCEDURE
```

```
PROCEDURE rule3(boid bJ)
  Vector pvJ
   FOR EACH BOID b
      IF b != bJ THEN pvJ = pvJ + b.velocity
      END IF
    END
    pvJ = pvJ / N-1
    RETURN (pvJ - bJ.velocity) / 8

END PROCEDURE
```