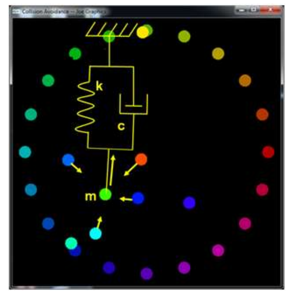
Collision Avoidance

(this is actually an excuse to discuss Functional Animation)

Oregon State University

Mike Bailey

mjb@cs.oregonstate.edu



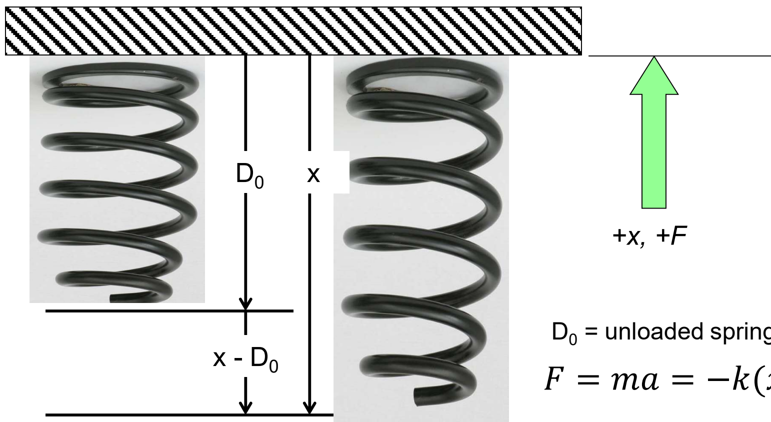
Collision Avoidance Isn't the Same as *Collision Detection*

Both recognize that objects cannot occupy the same space at the same time, but …

Collision Detection lets the objects collide and bounce off each other

Collision Avoidance tries to get the objects to *change their paths* so that they don't collide in the first place

How do Springs Work?



D_0 = unloaded spring length

$$F = ma = -k(x - D_0)$$

k = ***spring stiffness*** in Newtons/meter or pounds/inch

Or, as a differential equation,

$$m\ddot{x} + k(x - D_0) = 0$$

Oregon State University
Computer Graphics

mjb – July 31, 2021

3

Functional Animation

In **Functional Animation**, we setup a fake force system (with fake springs and other mechanical components) to make an object “want” to go to a certain place without us having to actually animate it to go there.

If this was *all* we were going to do, then keyframe animation would get the object there just as well.

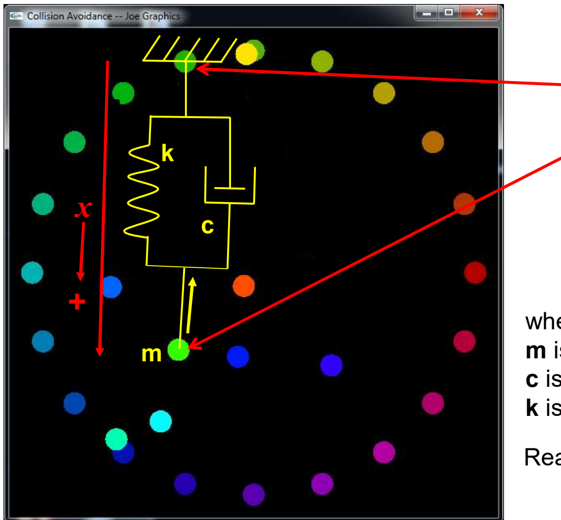
But the *big* advantage of Functional Animation is that we can add *other* fake forces to make the objects behave in more complex ways, such as avoiding each other.

Oregon State University
Computer Graphics

mjb – July 31, 2021

4

First Goal – Make the Free Body Move Towards its Final Position



Differential equation:

$$m\ddot{x} + c\dot{x} + kx = 0$$

where:
m is the mass
c is the damping (≈ a shock absorber)
k is the spring constant

Rearrange to get just the acceleration:

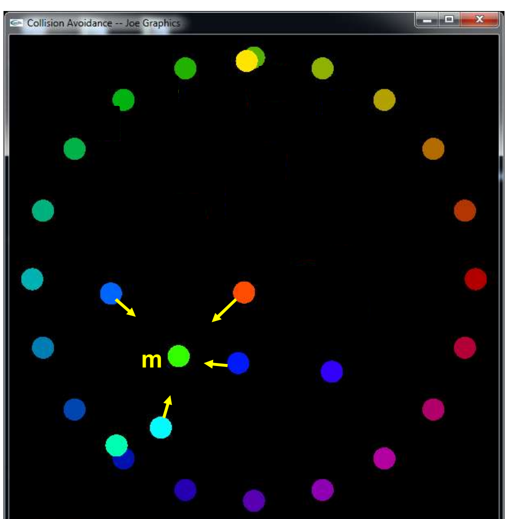
$$\ddot{x} = \frac{-c\dot{x} - kx}{m}$$

Oregon State University
Computer Graphics

mjb – July 31, 2021

5

Second Goal – Make the Free Body Want to Move Away From All the Other Bodies



Differential equation:

$$m\ddot{x} = \sum F$$

Rearrange to get just the acceleration:

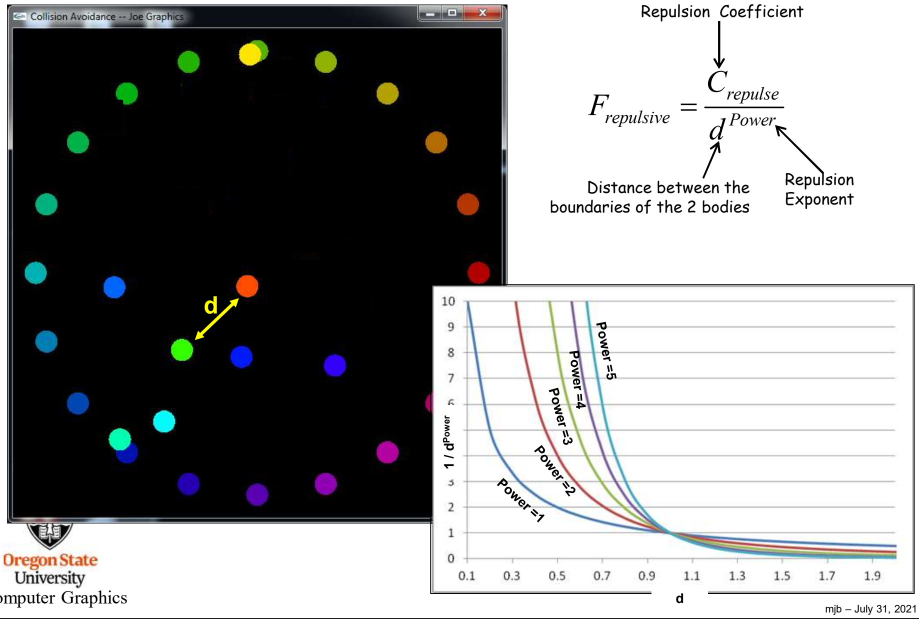
$$\ddot{x} = \frac{\sum F}{m}$$

Oregon State University
Computer Graphics

mjb – July 31, 2021

6

Repulsive Force



Repulsion Coefficient

$$F_{repulsive} = \frac{C_{repulse}}{d^{Power}}$$

Distance between the boundaries of the 2 bodies

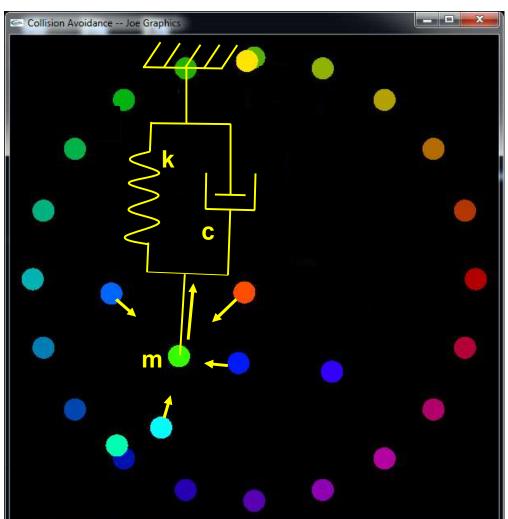
Repulsion Exponent

Oregon State University Computer Graphics

mjb – July 31, 2021

7

Total Goal – Make the Free Body Move Towards its Final Position While Being Repelled by the Other Bodies



$$m\ddot{x} + c\dot{x} + kx = \sum F$$

$$\ddot{x} = \frac{\sum F - c\dot{x} - kx}{m}$$

Oregon State University Computer Graphics

mjb – July 31, 2021

8

Accelerating an Object Towards a Target, Under the Influence of Outside Forces

$$m\ddot{x} + c\dot{x} + kx = \sum F$$

Fundamental equation of a second order system, if anchored at the origin

$$m\ddot{x} + c\dot{x} + k(x - x_T) = \sum F$$

Fundamental equation of a second order system, if anchored at a target position. (We're assuming that the target final velocity wants to be 0.)

$$\ddot{x} + c\dot{x} + k(x - x_T) = \sum F$$

We're not doing a real physics simulation, just going for an effect. Therefore, we can unitize the mass, and scale *c*, *k*, and the external forces appropriately

$$\ddot{x} = -c\dot{x} - k(x - x_T) - \sum F$$

Solve for the acceleration that moving towards the target needs and the outside forces influence

$$\iint$$

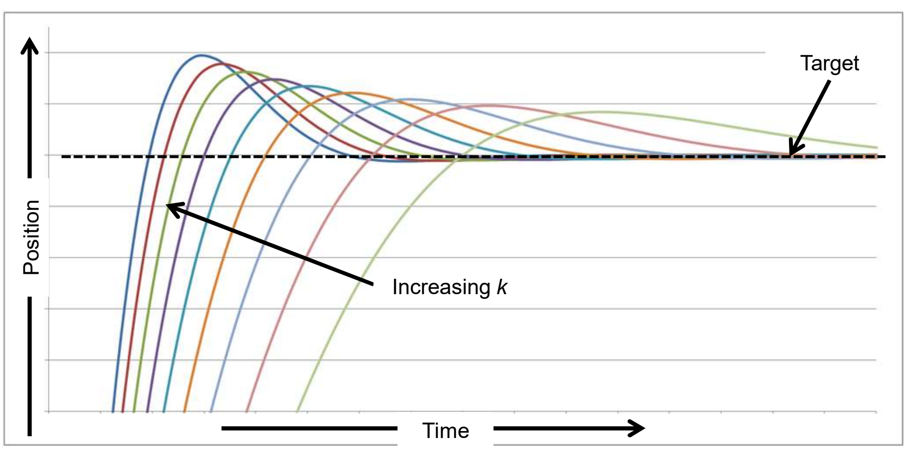
Use that acceleration to compute the next position and velocity (1st order, 2nd order, 4th order, etc.)

Oregon State University Computer Graphics

mjb – July 31, 2021

9

An Increased *k* Gets the Object to the Target Faster (But Increases Target Overshoot)

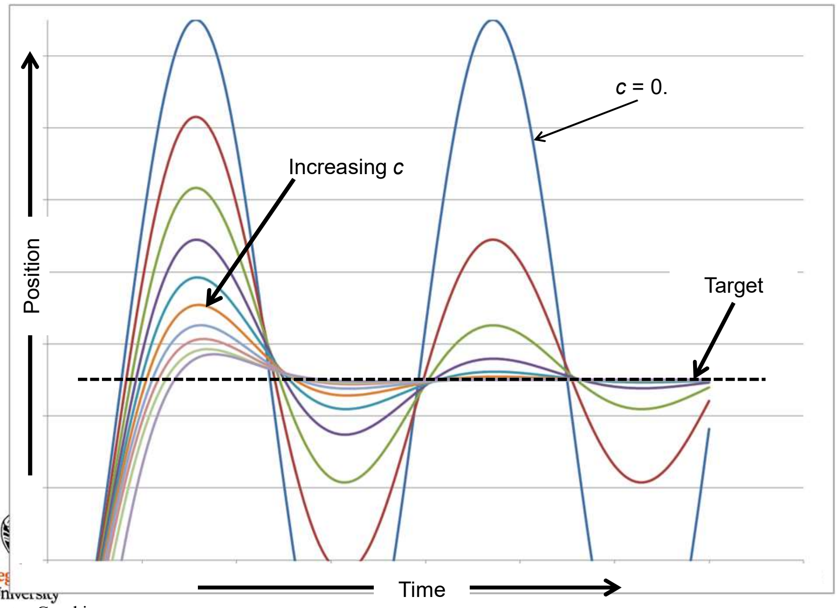


Oregon State University Computer Graphics

mjb – July 31, 2021

10

An Increased *c* Reduces Target Overshoot (But the Object Moves More Sluggishly)

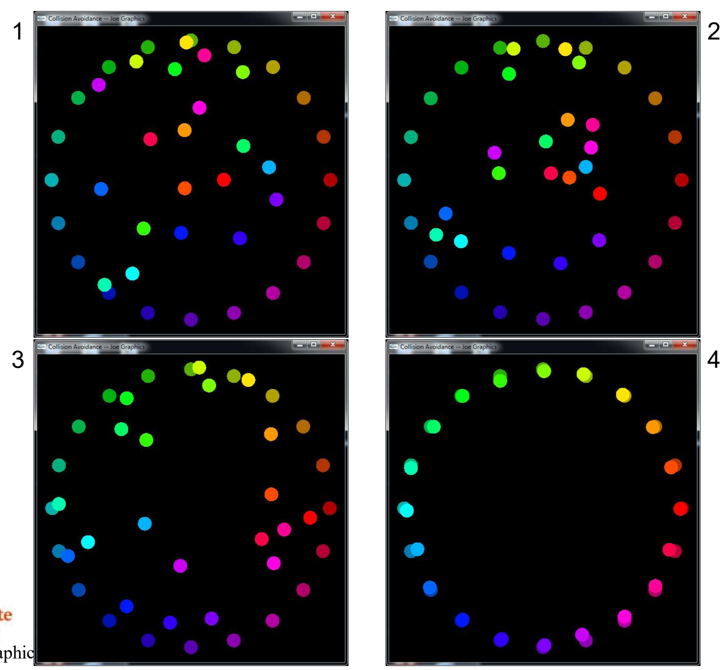


Oregon State University Computer Graphics

mjb – July 31, 2021

11

Collision Avoidance in Action – Time Animation

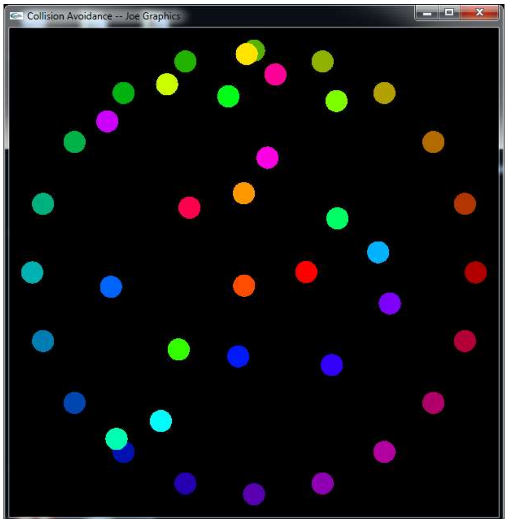


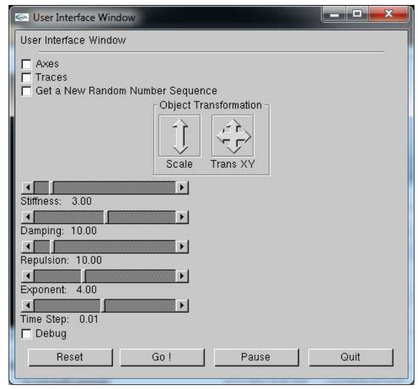
Oregon State University Computer Graphics

mjb – July 31, 2021

12

Varying the Parameters -- Start with This



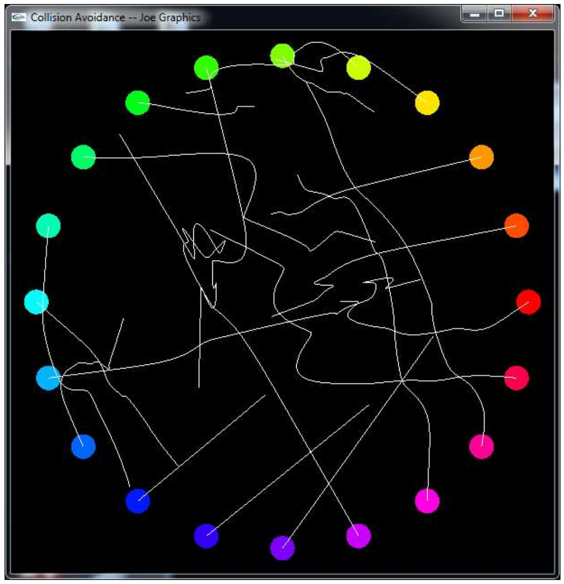


13

Increasing Stiffness

14

Increasing Stiffness

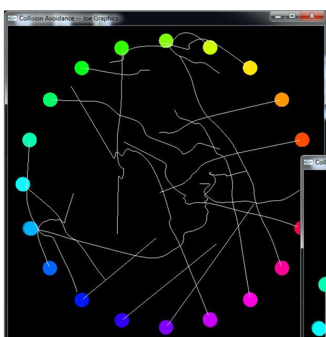


Stiffness = 3, 6, 9

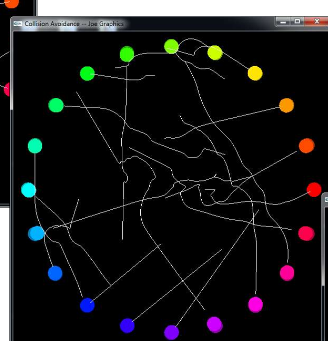
Oregon State University Computer Graphics

15

Increasing Damping

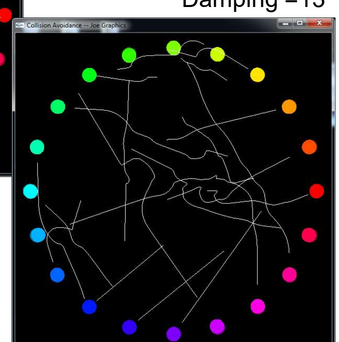


Damping = 7



Damping = 10

Damping =13

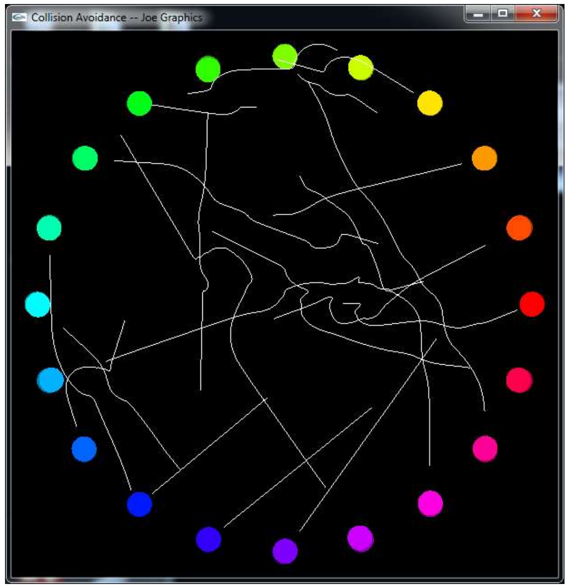


Oregon State University Computer Graphics

mjb – July 31, 2021

16

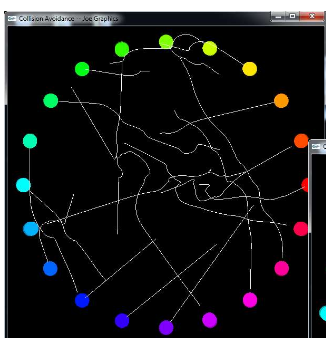
Increasing Damping



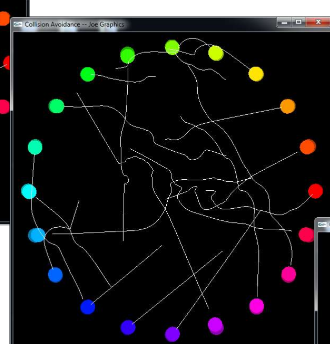
Damping = 7, 10, 13

Oregon State University Computer Graphics

Increasing the Repulsion Coefficient

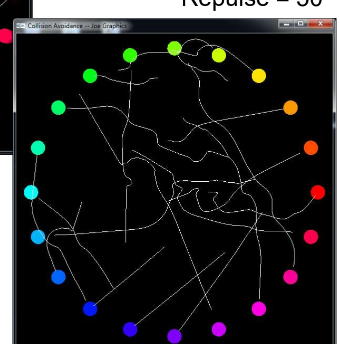


Repulse = 10



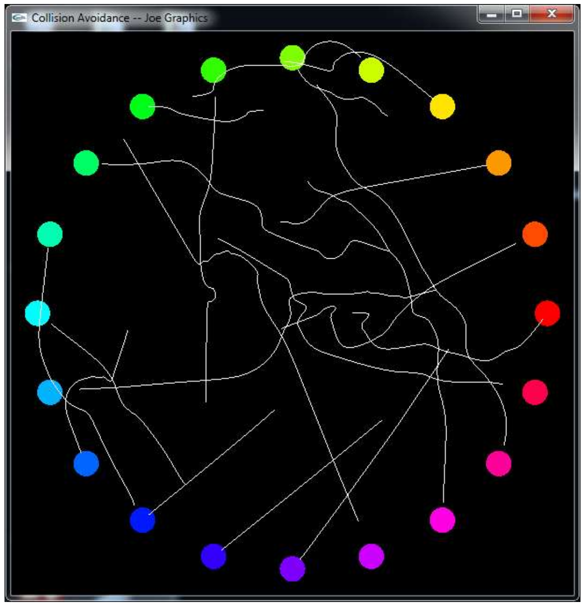
Repulse = 30

Repulse = 50



Oregon State University Computer Graphics

Increasing the Repulsion Coefficient



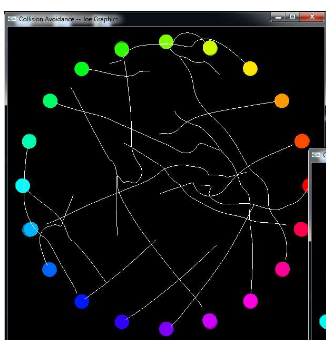
Repulse = 10, 30, 50

Oregon State University
Computer Graphics

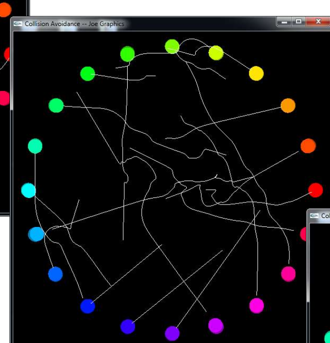
mjb – July 31, 2021

19

Increasing the Repulsion Exponent

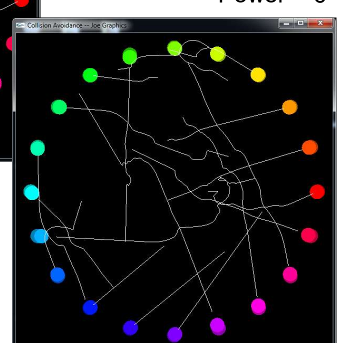


Power = 2



Power = 4

Power = 6

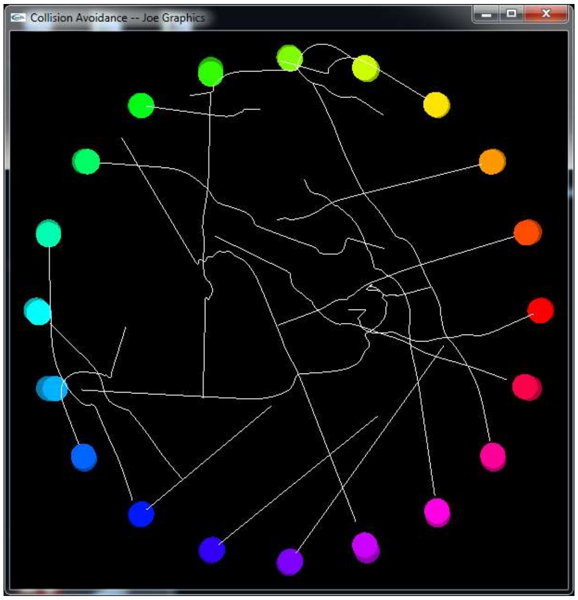


Oregon State University
Computer Graphics

mjb – July 31, 2021

20

Increasing the Repulsion Exponent



Power = 2, 4, 6

Oregon State University Computer Graphics

mjb – July 31, 2021

21

Parameter Rules of Thumb

A larger *k* value:
- Gets the object to its goal faster
- Possibly overshoots

A larger *c* value:
- Gets the object to its goal slower
- Decreases spurious wiggles

A larger *Repulsion Coefficient*:
- Objects give each other a wider berth
- It can get unnecessarily wide

A larger *Repulsion Exponent*:
- Influence waits to start until the objects are closer
- Influence increases quickly as the objects get closer
- Sometimes the influence increases too quickly and the objects do less avoiding and more bouncing

Oregon State University Computer Graphics

mjb – July 31, 2021

22