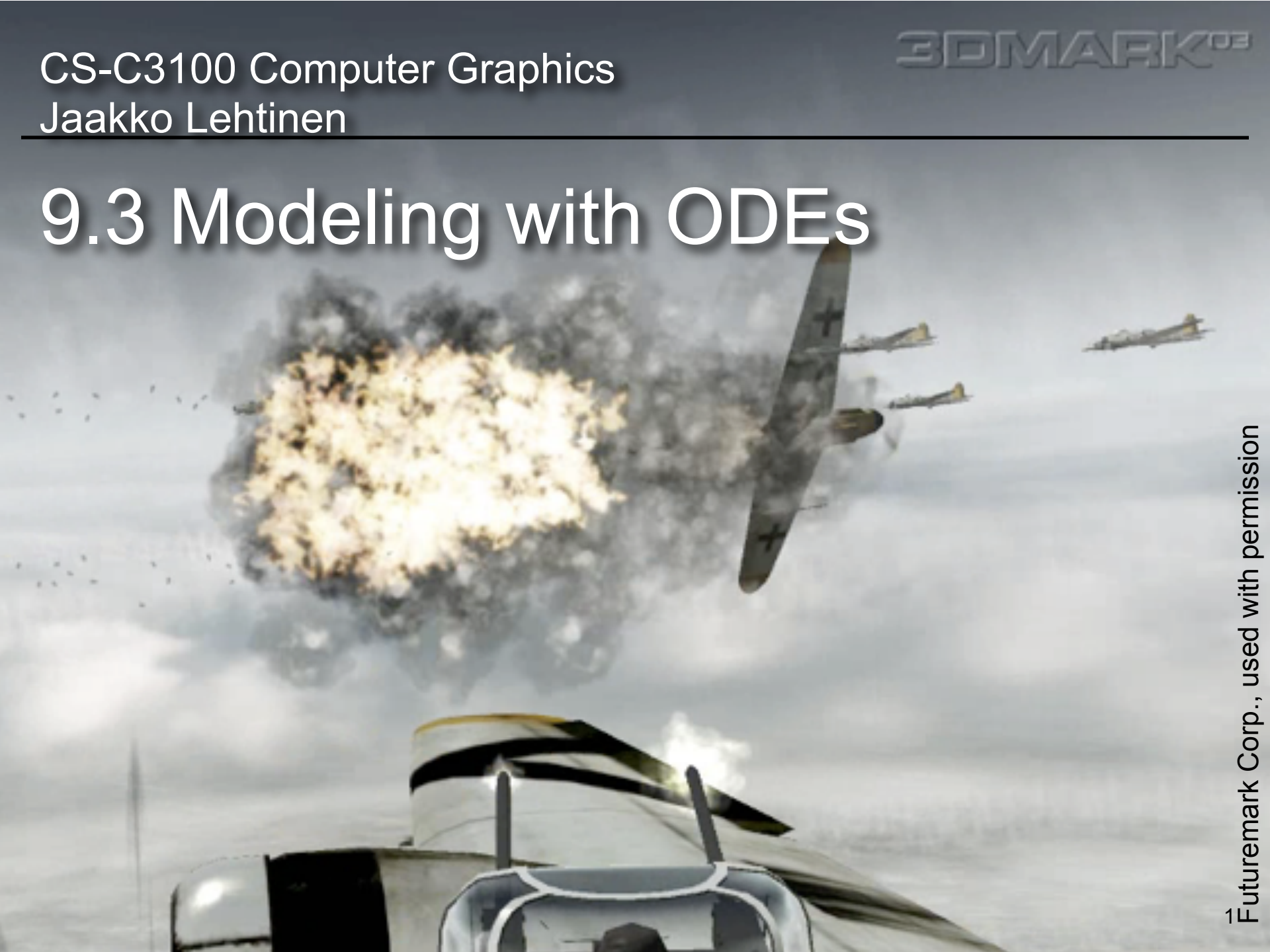
9.3 Modeling with ODEs



In This Video

- Modeling dynamics by ordinary differential equations (ODEs)
  - System state consists of positions, velocities, etc.
  - Forces change the state over infinitesimal time intervals

Differential Equations

- Motion of physical systems is modelled by these
  - "If I am in state $\boldsymbol{X}$ at time t, what is my state at time $t+\mathrm{d}t$?

Differential Equations

- Motion of physical systems is modelled by these
  - "If I am in state $\boldsymbol{X}$ at time t, what is my state at time $t+\mathrm{d}t$?
- What is the "state $\boldsymbol{X}$"? A vector that describes all the free parameters of the physical system.
  - For a single point-like mass: position + velocity
  - For a rigid object: position + orientation + linear momentum + angular momentum
  - etc.
- Basically all of physical reality is described by these

Ordinary Differential Equations

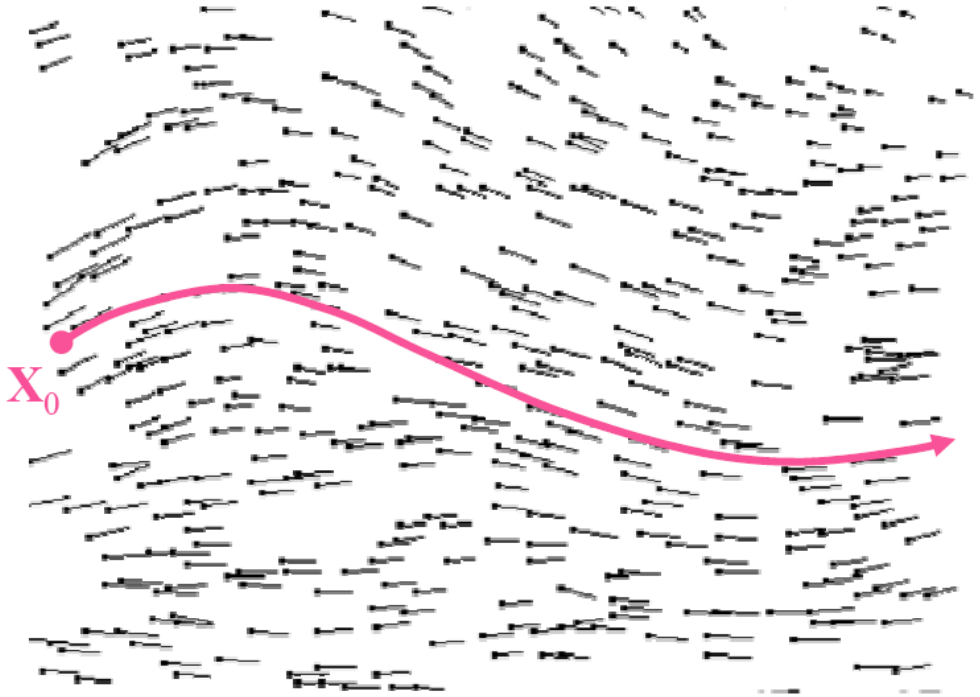
$$\frac{d\,\mathbf{X}(t)}{dt} = f(\mathbf{X}(t), t)$$

- Given a function $f(\mathbf{X},t)$ compute $\mathbf{X}(t)$
- Typically, *initial value problems*:
  - Given values $\mathbf{X}(t_0)=\mathbf{X}_0$
  - Find values $\mathbf{X}(t)$ for $t > t_0$

- We can use lots of standard tools

Path through a Vector Field

- ***X(t)****:* path in multidimensional phase space

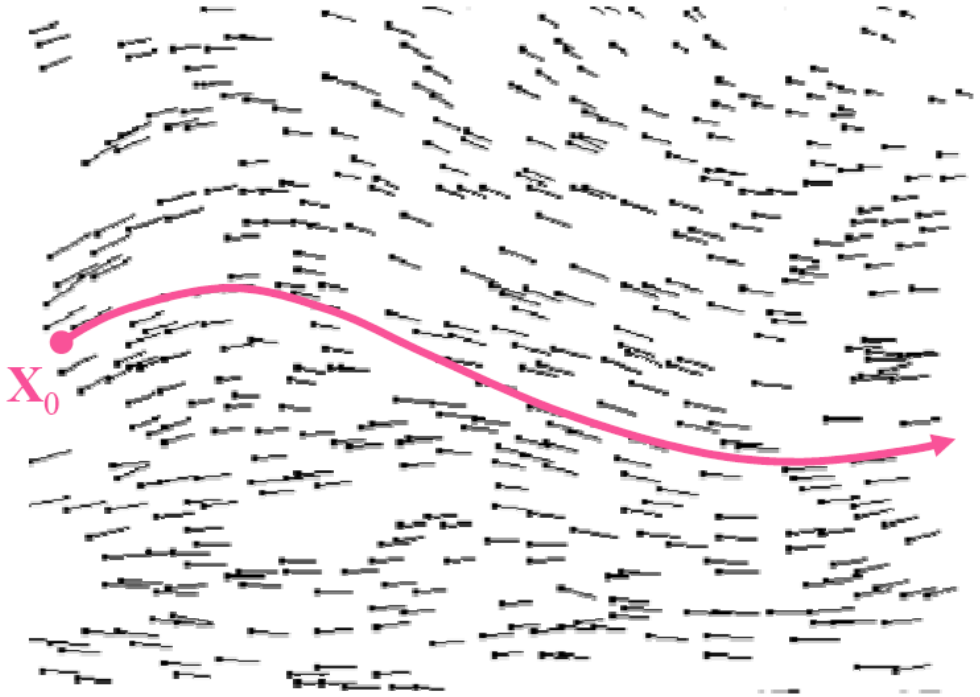


$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{X} = f(\boldsymbol{X}, t)$$

"When we are at state **X** at time *t*, where will **X** be after an infinitely small time interval d*t* ?"

Path through a Vector Field

- ***X(t)****:* path in multidimensional phase space

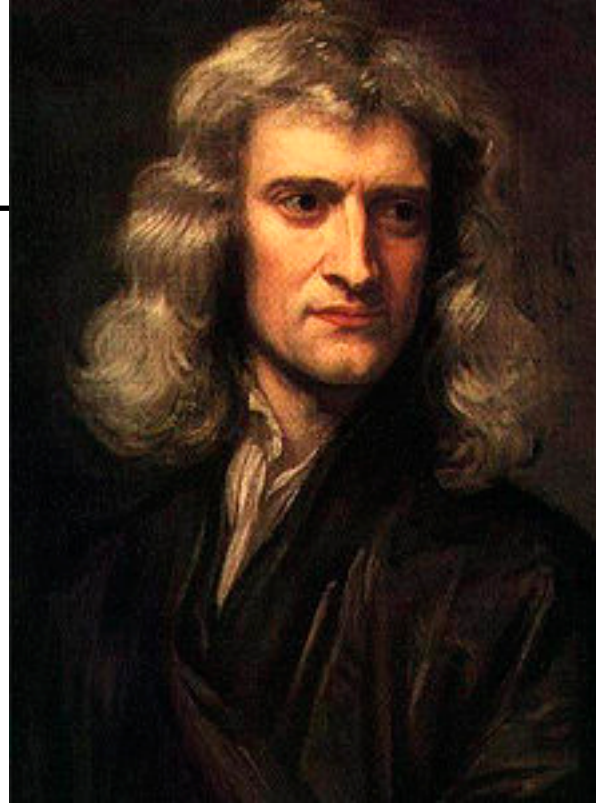


$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{X} = f(\boldsymbol{X}, t)$$

"When we are at state **X** at time *t*, where will **X** be after an infinitely small time interval d*t* ?"

- $f = \mathrm{d}/\mathrm{d}t\ \mathbf{X}$ is a vector that sits at each point in phase space, pointing the direction.

Newtonian Mechanics



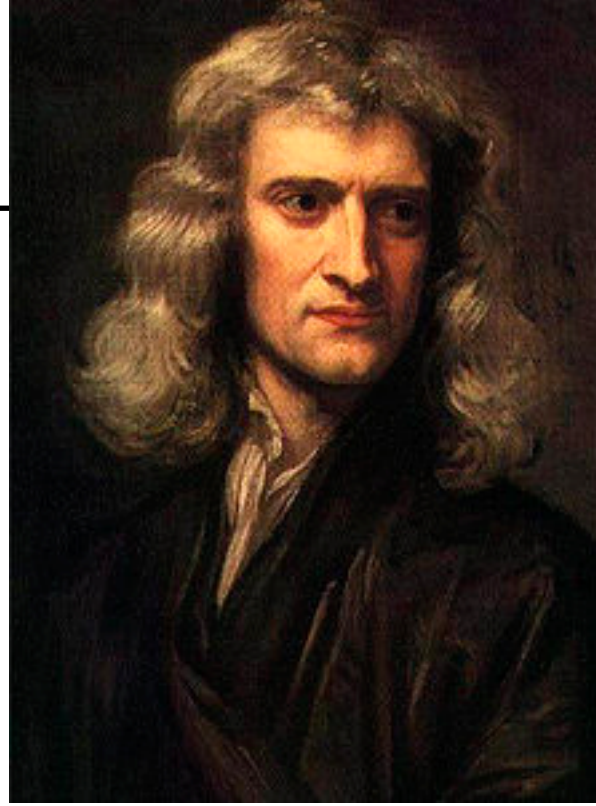
- Point mass: 2nd order ODE

$$\vec{F} = m\vec{a} \quad \text{or} \quad \vec{\boldsymbol{F}} = m\frac{d^2\vec{\boldsymbol{x}}}{dt^2}$$

- Position $\boldsymbol{x}$ and force $\boldsymbol{F}$ are vector quantities
  - We know $\boldsymbol{F}$ and m, want to solve for $\boldsymbol{x}$

- You've all seen this a million times before

Reduction to 1st Order



- Point mass: 2nd order ODE

$$\vec{F} = m\vec{a} \quad \text{or} \quad \vec{\boldsymbol{F}} = m\frac{d^2\vec{\boldsymbol{x}}}{dt^2}$$

- Corresponds to system of first order ODEs

$$\begin{cases} \frac{d}{dt}\vec{\boldsymbol{x}} = \vec{\boldsymbol{v}} \\ \frac{d}{dt}\vec{\boldsymbol{v}} = \vec{\boldsymbol{F}}/m \end{cases}$$

2 unknowns (**x**, **v**) instead of just **x**

Reduction to 1st Order

$$\begin{cases} \frac{d}{dt}\vec{\boldsymbol{x}} = \vec{\boldsymbol{v}} \\ \frac{d}{dt}\vec{\boldsymbol{v}} = \vec{\boldsymbol{F}}/m \end{cases}$$

2 variables (**x**, **v**)
instead of just one

- Why reduce?

Reduction to 1st Order

$$\begin{cases} \frac{d}{dt}\vec{\boldsymbol{x}} = \vec{\boldsymbol{v}} \\ \frac{d}{dt}\vec{\boldsymbol{v}} = \vec{\boldsymbol{F}}/m \end{cases}$$

2 variables (**x**, **v**)
instead of just one

- Why reduce?
  - Numerical solvers grow more complicated with increasing order, can just write one 1st order solver and use it
  - Note that this doesn't mean it would always be easy :-)

Notation

- Let's stack the pair (**x**, **v**) into a bigger state vector **X**

$$\boldsymbol{X} = \begin{pmatrix} \vec{\boldsymbol{x}} \\ \vec{\boldsymbol{v}} \end{pmatrix}$$

For a particle in 3D, state vector **X** has 6 numbers

$$\frac{d}{dt}\boldsymbol{X} = f(\boldsymbol{X}, t) = \begin{pmatrix} \vec{\boldsymbol{v}} \\ \vec{\boldsymbol{F}}(x, v)/m \end{pmatrix}$$

Now, Many Particles

- We have N point masses
  - Let's just stack all **x**s and **v**s in a big vector of length 6N

$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{x}_1 \\ \boldsymbol{v}_1 \\ \vdots \\ \boldsymbol{x}_N \\ \boldsymbol{v}_N \end{pmatrix} \qquad f(\boldsymbol{X}, t) = \begin{pmatrix} \boldsymbol{v}_1 \\ \boldsymbol{F}^1(\boldsymbol{X}, t) \\ \vdots \\ \boldsymbol{v}_N \\ \boldsymbol{F}^N(\boldsymbol{X}, t) \end{pmatrix}$$

Now, Many Particles

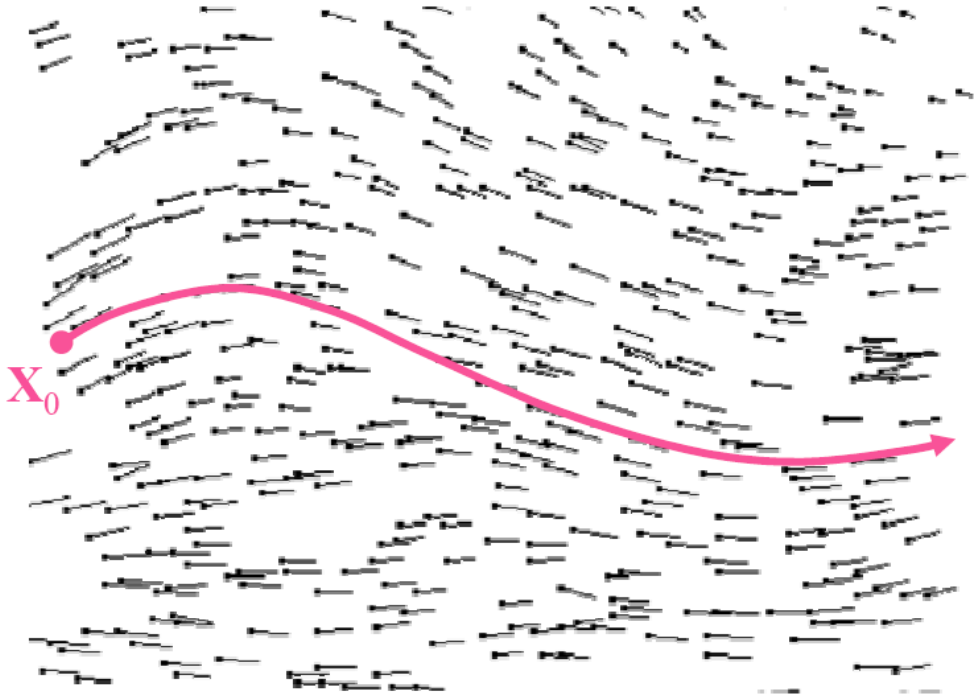
- We have N point masses
  - Let's just stack all **x**s and **v**s in a big vector of length 6N
  - $\mathbf{F}^i$ denotes the force on particle *i*
    - When particles don't interact, $\mathbf{F}^i$ only depends on $\mathbf{x}_i$ and $\mathbf{v}_i$.

$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{x}_1 \\ \boldsymbol{v}_1 \\ \vdots \\ \boldsymbol{x}_N \\ \boldsymbol{v}_N \end{pmatrix} \qquad f(\boldsymbol{X}, t) = \begin{pmatrix} \boldsymbol{v}_1 \\ \boldsymbol{F}^1(\boldsymbol{X}, t) \\ \vdots \\ \boldsymbol{v}_N \\ \boldsymbol{F}^N(\boldsymbol{X}, t) \end{pmatrix}$$

f gives d/dt X, remember!

Recap: Path through a Vector Field

- ***X(t)****:* path in multidimensional phase space



$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{X} = f(\boldsymbol{X}, t)$$

"When we are at state **X** at time *t*, where will **X** be after an infinitely small time interval d*t* ?"

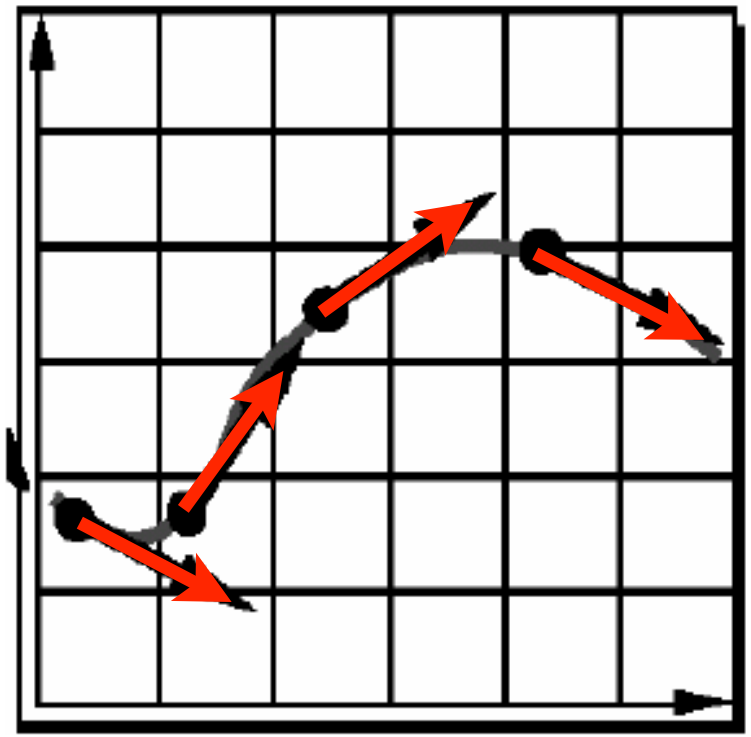
- $f = \mathrm{d}/\mathrm{d}t\ \mathbf{X}$ is a vector that sits at each point in phase space, pointing the direction.

Numerics of ODEs

- Numerical solution is called “integration of the ODE”
- Many techniques
  - Today, the simplest one
  - Next time we’ll look at some more advanced techniques

Intuitive Solution: Take Steps

- Current state **X**
- Examine f(**X**,t) at (or near) current state
- Take a step to new value of **X**



$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{X} = f(\boldsymbol{X}, t)$$

$$\Rightarrow \text{“}\mathrm{d}\boldsymbol{X} = \mathrm{d}t\, f(\boldsymbol{X}, t)\text{”}$$

f = d/d*t* ***X*** is a vector that sits at each point in phase space, pointing the direction.

Euler's Method

- Simplest and most intuitive
- Pick a **step size** h
- Given $\mathbf{X}_0=\mathbf{X}(t_0)$, take step:

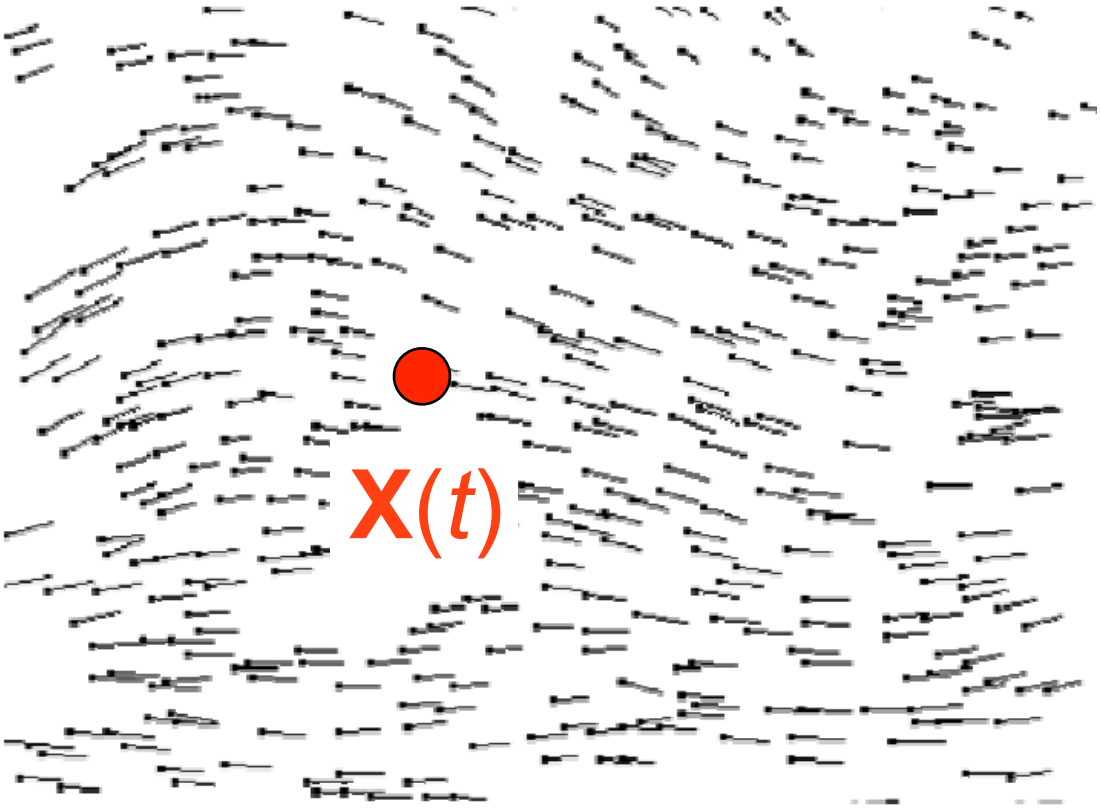
$$t_1 = t_0 + h$$

$$\mathbf{X}_1 = \mathbf{X}_0 + h\, f(\mathbf{X}_0, t_0)$$

- Piecewise-linear approximation to the path
- **Basically, just replace dt by a small but finite number h**

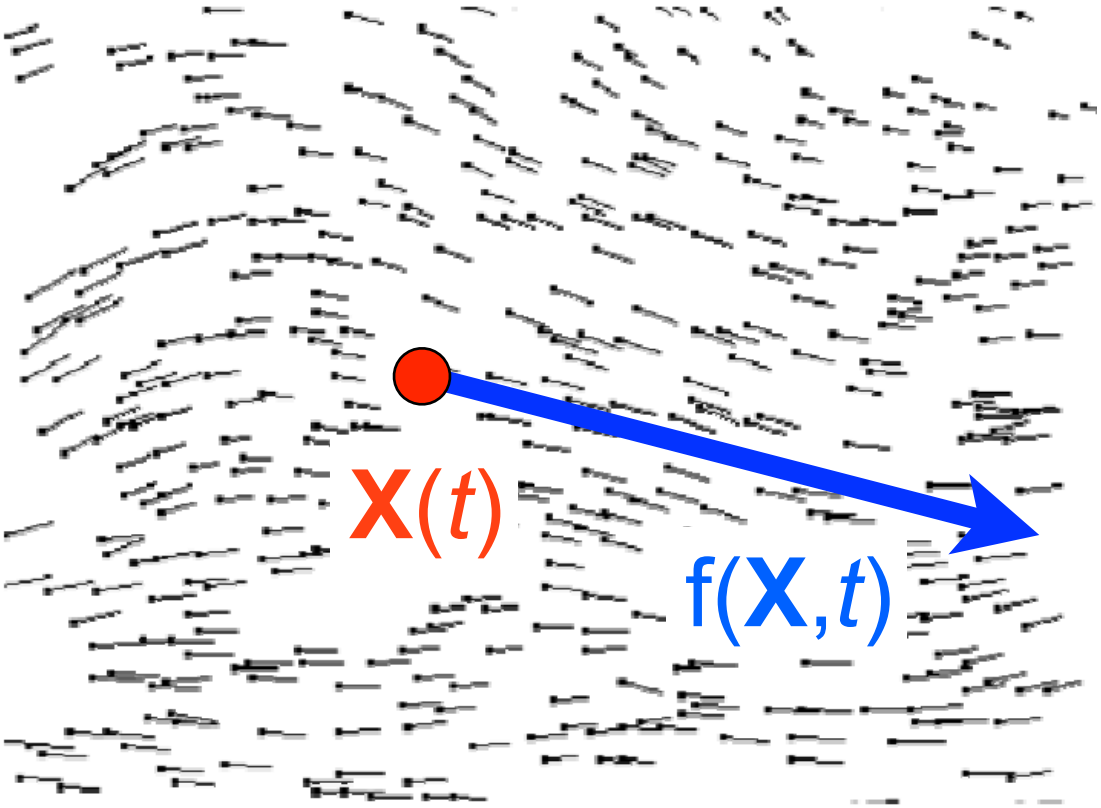
Euler, Visually

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{X} = f(\boldsymbol{X}, t)$$



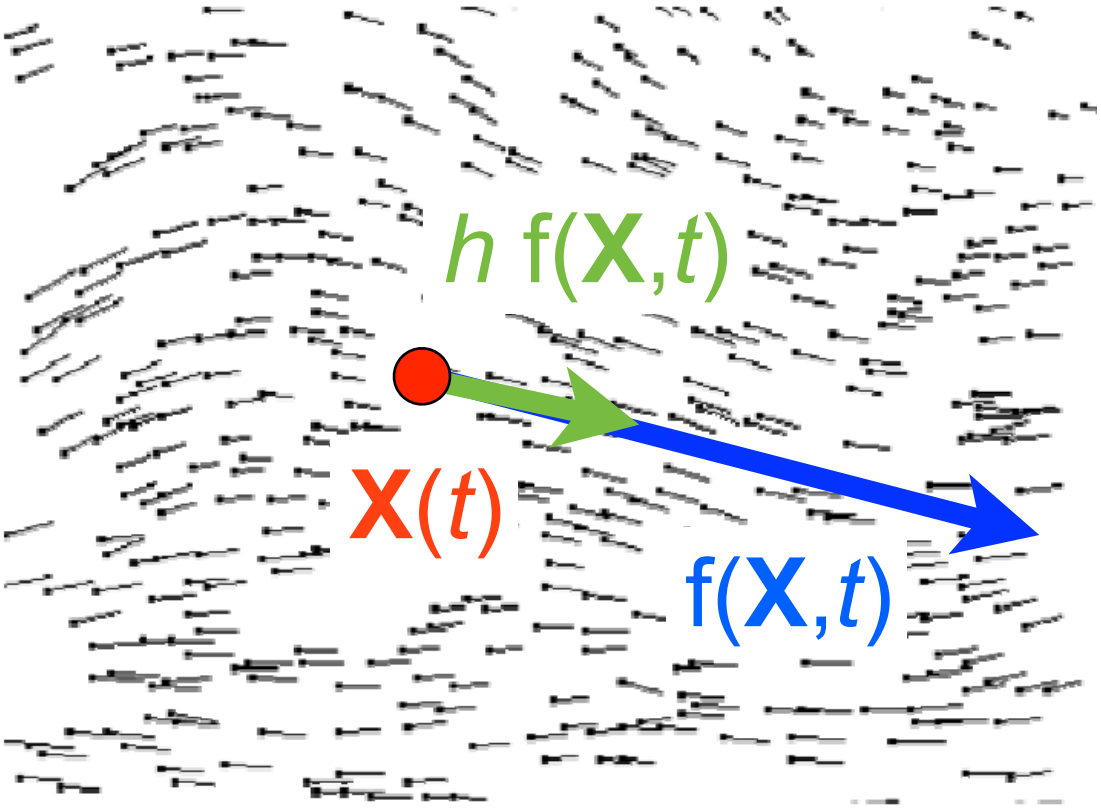
Euler, Visually

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{X} = f(\boldsymbol{X}, t)$$



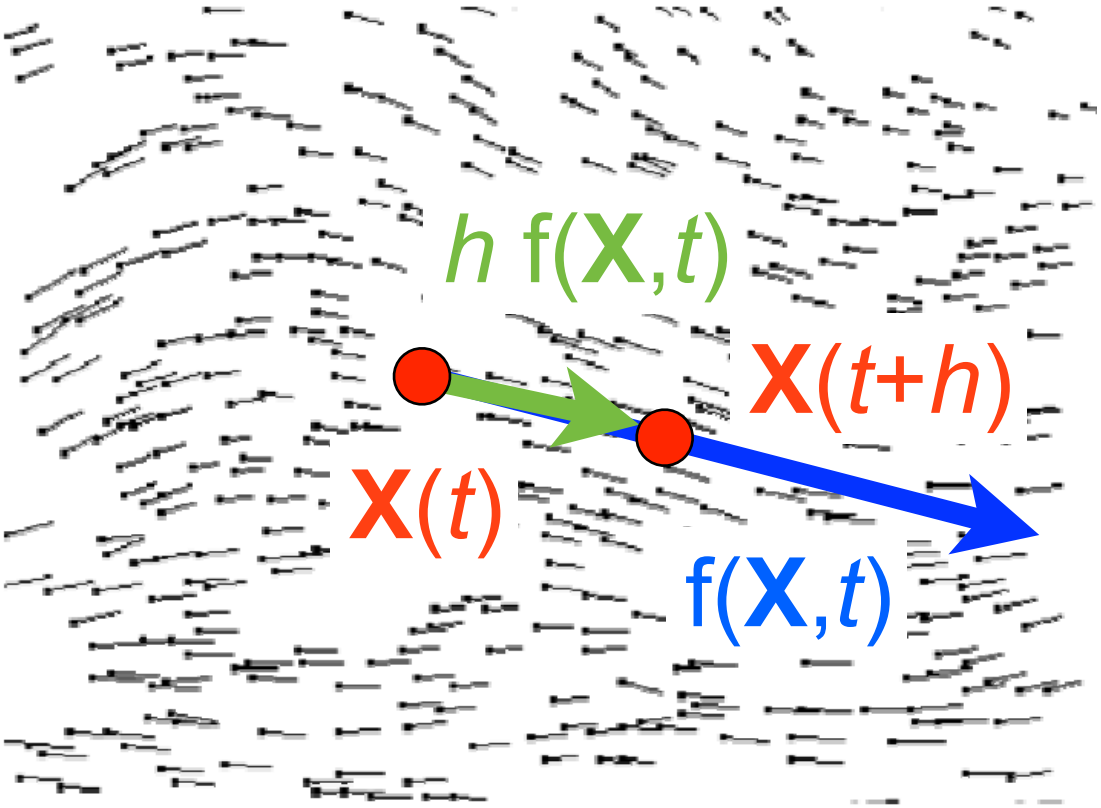
Euler, Visually

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{X} = f(\boldsymbol{X}, t)$$



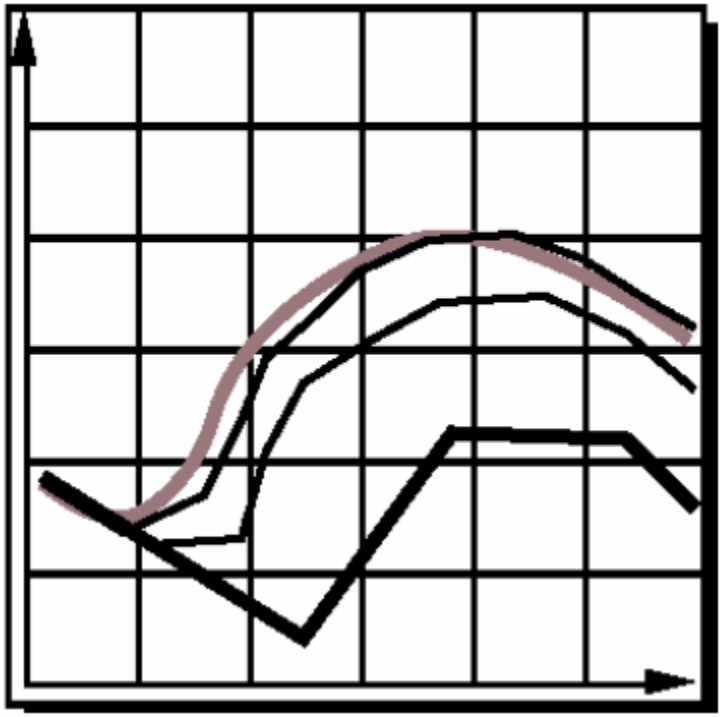
Euler, Visually

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{X} = f(\boldsymbol{X}, t)$$



Effect of Step Size

- Step size controls accuracy
- Smaller steps more closely follow curve
  - May need to take many small steps per frame
  - Properties of $f(\mathbf{X}, t)$ determine this (more later)



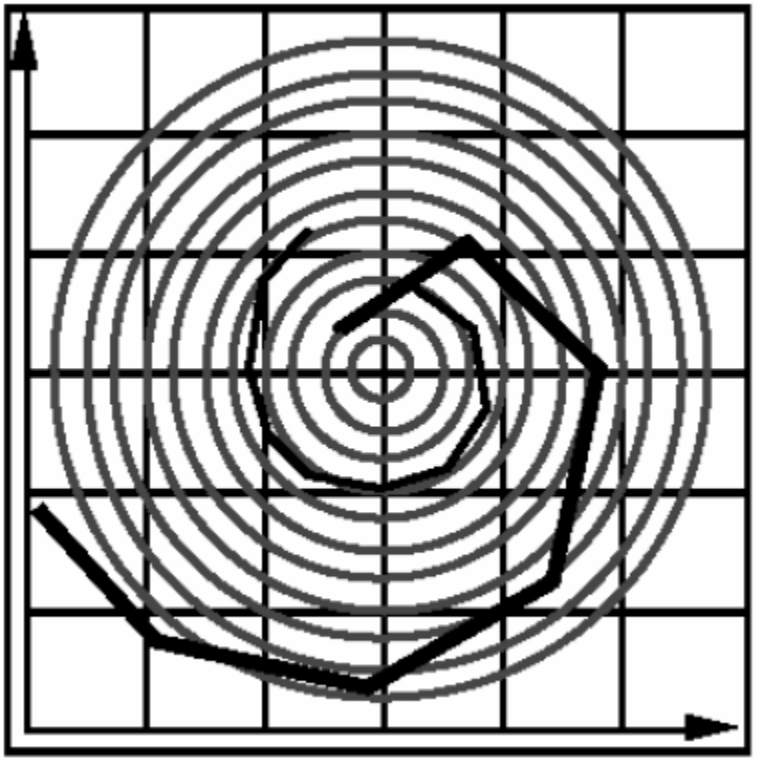
Euler's method: Inaccurate

- Moves along tangent; can leave solution curve, e.g.:

$$f(\mathbf{X},t) = \begin{pmatrix} -y \\ x \end{pmatrix}$$

- Exact solution is circle:

$$\mathbf{X}(t) = \begin{pmatrix} r\cos(t+k) \\ r\sin(t+k) \end{pmatrix}$$



Euler's method: Inaccurate

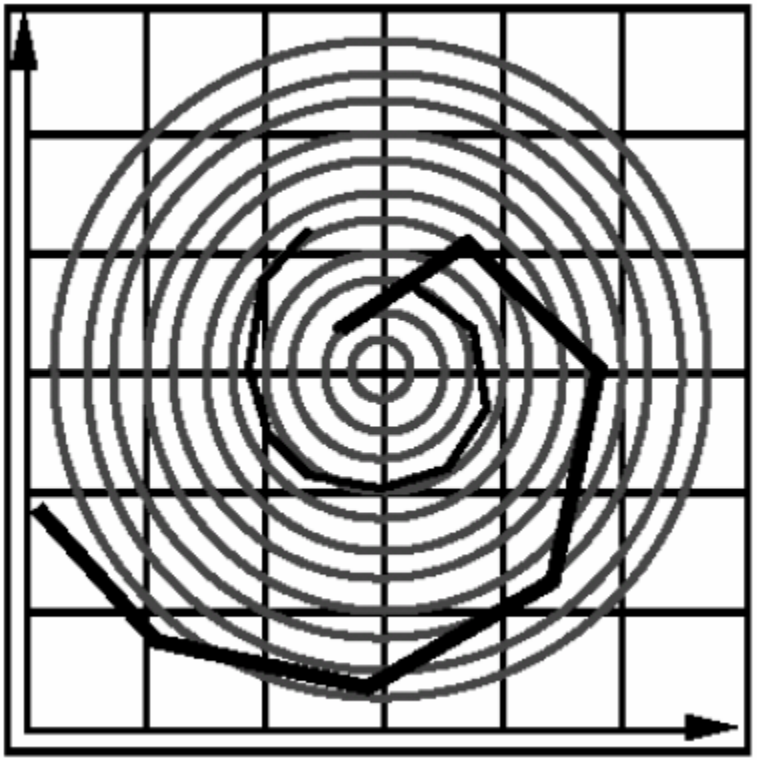
- Moves along tangent; can leave solution curve, e.g.:

$$f(\mathbf{X},t) = \begin{pmatrix} -y \\ x \end{pmatrix}$$

- Exact solution is circle:

$$\mathbf{X}(t) = \begin{pmatrix} r\cos(t+k) \\ r\sin(t+k) \end{pmatrix}$$

- Euler spirals outward no matter how small h is
  - will just diverge more slowly



More Accurate Alternatives

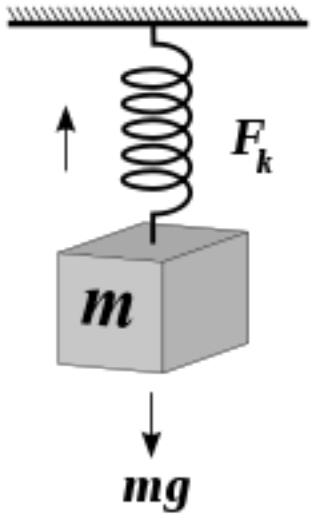
- Midpoint, Trapezoid, Runge-Kutta
  - Also, "implicit methods" (next week)

More on this next time

- Extremely valuable resource: SIGGRAPH 2001 course notes on physically based modeling

What is a Force?

- A force changes the motion of the system
  - Newton says: When there are no forces, motion continues uniformly in a straight line
- Forces can depend on location, time, velocity
  - Gravity, spring, viscosity, wind, etc.
- For point masses, forces are vectors
  - Ie., to get total force, take vector sum of everything



Wikipedia

Let's Tie This to Math

- A force changes the motion of the system
  - Newton says: When there are no forces, motion continues uniformly in a straight line (good enough for us)

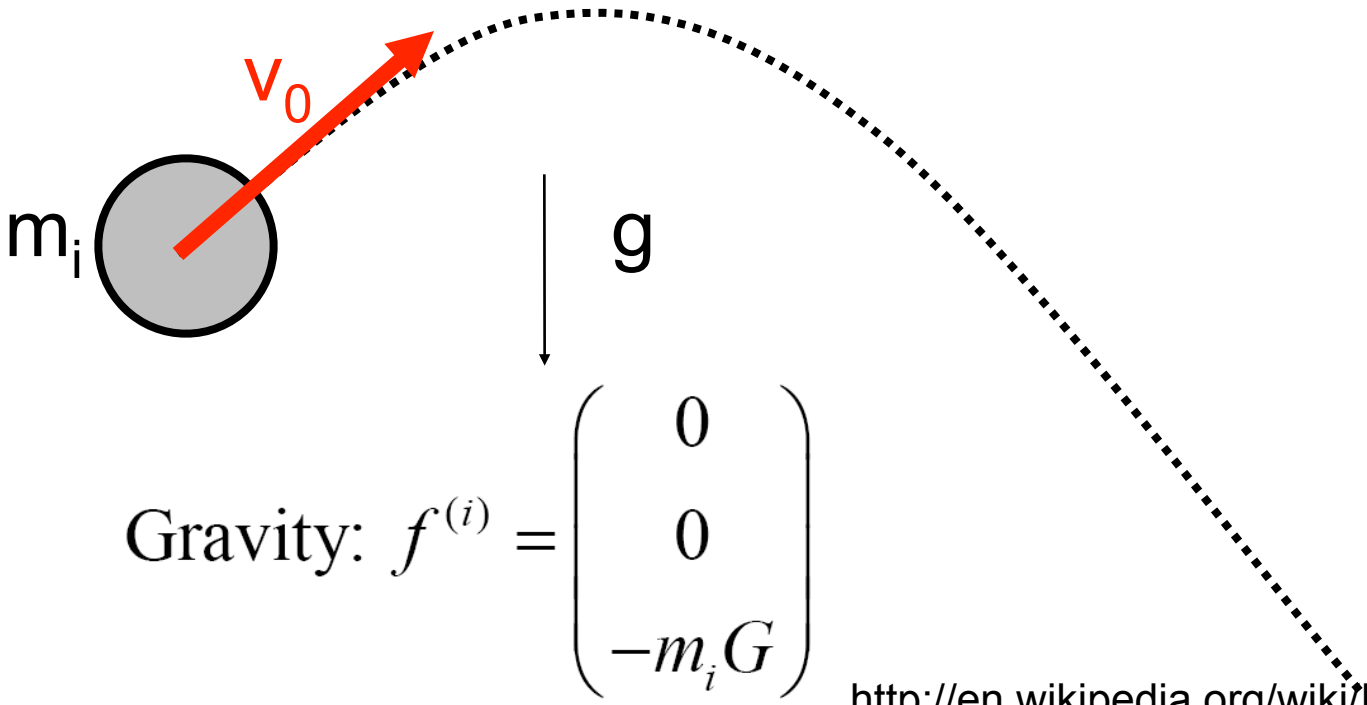
$$\begin{cases} \frac{d}{dt}\vec{\boldsymbol{x}} = \vec{\boldsymbol{v}} \\ \frac{d}{dt}\vec{\boldsymbol{v}} = \vec{\boldsymbol{F}}/m \end{cases}$$

2 variables (**x**, **v**) instead of just one

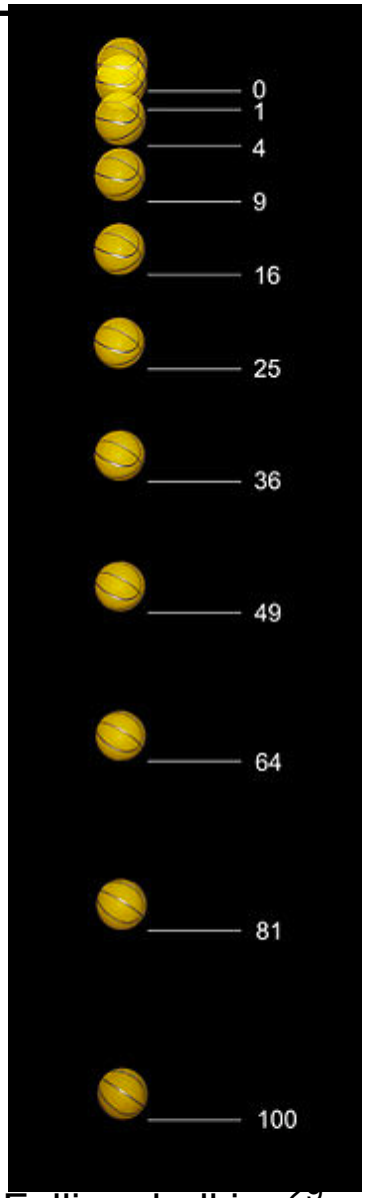
  - This shows in the above equation as follows:
    F is zero => d/dt **v** is zero =>
    v is constant => **x** moves in a straight line

Forces: Gravity on Earth

- Depends only on particle mass
- $f(\mathbf{X},t)$ = constant



$$\text{Gravity: } f^{(i)} = \begin{pmatrix} 0 \\ 0 \\ -m_i G \end{pmatrix}$$



http://en.wikipedia.org/wiki/Image:Falling_ball.jpg

Let's Tie This to Math: Gravity

- $f(\mathbf{X},t)$ = constant

$$\begin{cases} \frac{d}{dt}\vec{x} = \vec{v} \\ \frac{d}{dt}\vec{v} = \vec{F}/m \end{cases}$$

$$\text{Gravity: } f^{(i)} = \begin{pmatrix} 0 \\ 0 \\ -m_i G \end{pmatrix}$$

  - **F** is constant => d/dt **v** is constant =>
    v changes linearly => **x** moves in a parabola

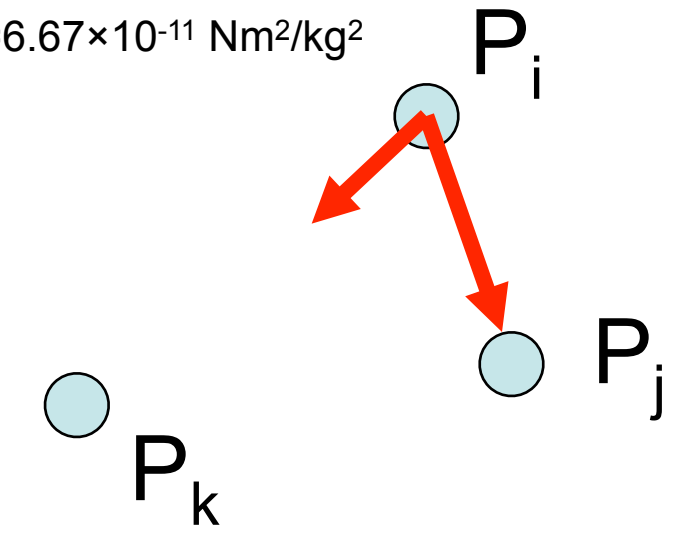
  - Yes, this is high school physics.. :)

Forces: Gravity (N-body problem)

- For massive bodies, gravity depends on all other particles, opposite for pairs of particles
- Force in the direction of $\mathbf{p}_i$-$\mathbf{p}_j$ with magnitude inversely proportional to square distance

$$\|\boldsymbol{F}_{ij}\| = \frac{G\, m_i\, m_j}{r^2}$$

where $G=6.67\times10^{-11}$ Nm^2/kg^2



- Testing all pairs is $O(n^2)$!

Particles are not independent!

Real-Time Gravity Demo

- Link to video

An Aside on Gravity

- That was Brute Force
  - Meaning all O(n^2) pairs of particles were considered when computing forces
  - Yes, computers are fast these days, but this gets prohibitively expensive soon. (The square in n^2 wins.)

- *Hierarchical techniques* approximate forces caused by many distant attractors by one force, yields O(n)!
  - "Fast Multipole Method", Greengard and Rokhlin, J Comput Phys 73, p. 325 (1987) This inspired very cool hierarchical illumination rendering algorithms in graphics (hierarchical radiosity, etc.)

Forces: Viscous Damping

$$f^{(i)} = -dv^{(i)}$$

- Damping force on particle i determined its velocity
  - Opposes motion
  - E.g. wind resistance
- Removes energy, so system can settle
- Small amount of damping can stabilize solver
- Too much damping makes motion like in glue

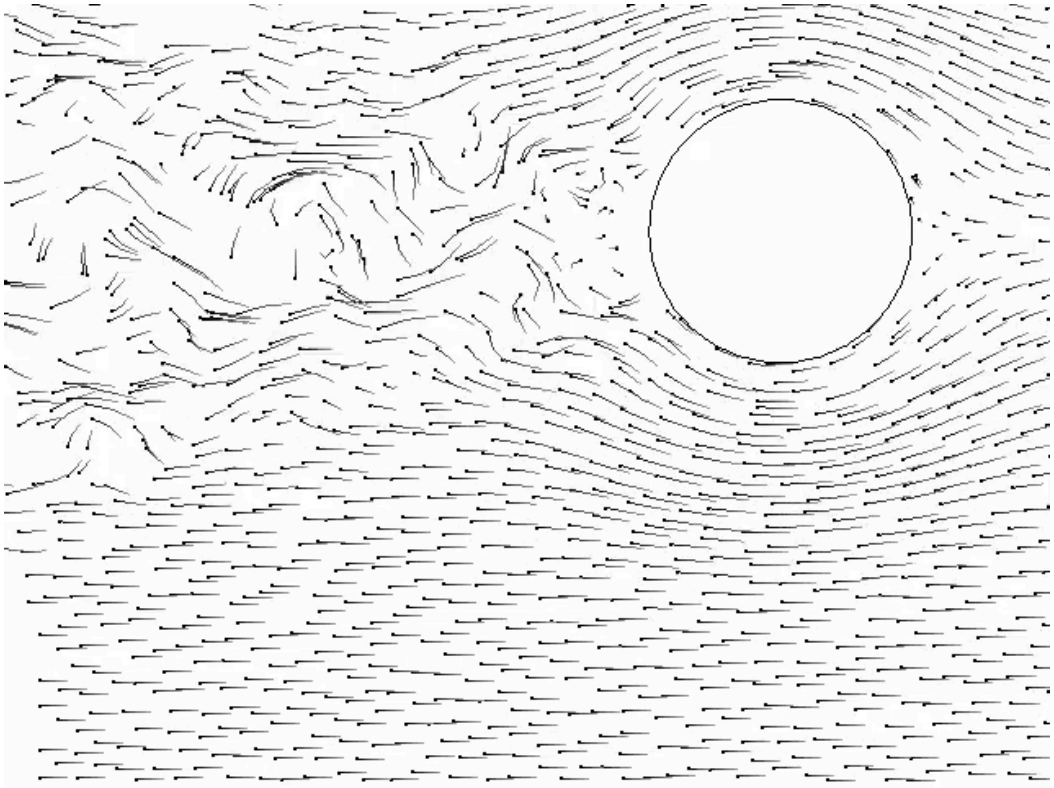
Forces: Spatial Fields

- Externally specified force (or velocity) fields in space
- Force on particle i depends only on its position
- Arbitrary functions
  - wind
  - attractors, repulsors
  - vortices
- Can depend on time
- Note: these add energy, may need damping

Bridson et al.

Example: Procedural Spatial Field

- Curl noise for procedural fluid flow, R. Bridson, J. Hourihan, and M. Nordenstam, Proc. ACM SIGGRAPH 2007.



Plausible, conrollable force fields – just advecting particles along the flow gives cool results

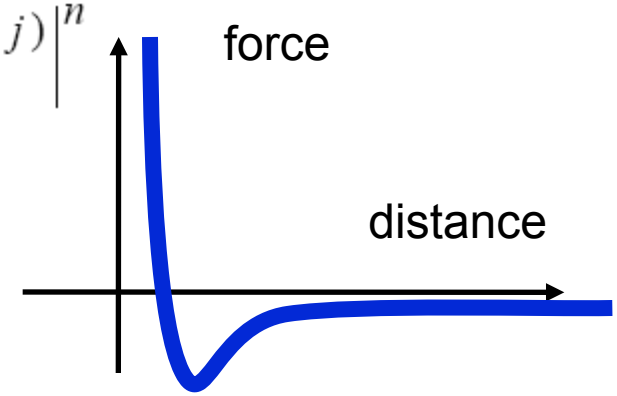
And it's simple, too!

Forces: Other Spatial Interaction

$$\text{Spatial interaction: } f^{(i)} = \sum_j f(x^{(i)}, x^{(j)})$$

- E.g., approximate fluid using Lennard-Jones force:

$$f(x^{(i)}, x^{(j)}) = \frac{k_1}{\left|x^{(i)} - x^{(j)}\right|^m} - \frac{k_2}{\left|x^{(i)} - x^{(j)}\right|^n}$$



- Repulsive + attractive force
- Again, O(N²) to test all pairs
  - usually only local
  - Use buckets/hierarchies to optimize

Particles are not independent!

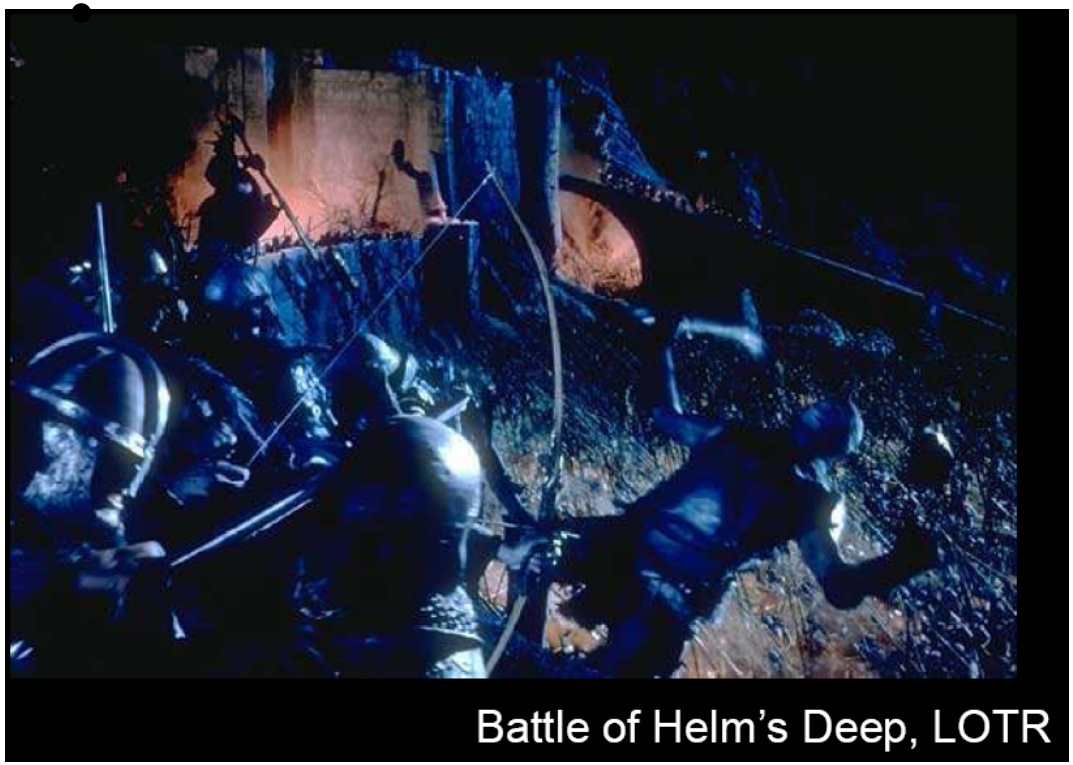
More Eyecandy from NVIDIA

- Fluid flow solved using a regular grid solver
  - This gives a velocity field
- 0.5M smoke particles advected using the field
  - That means particle velocity is given by field
- Particles are for rendering, motion solved using other methods
- Link to video

NVIDIA

More Advanced “Forces”

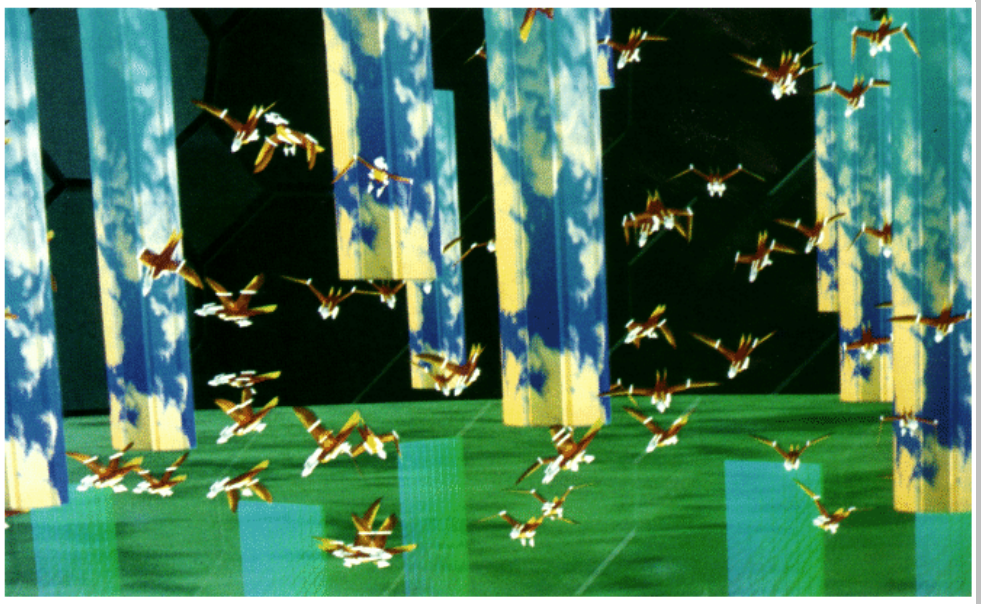
- Flocking birds, fish shoals
  - http://www.red3d.com/cwr/boids/
- Crowds (www.massivesoftware.com)

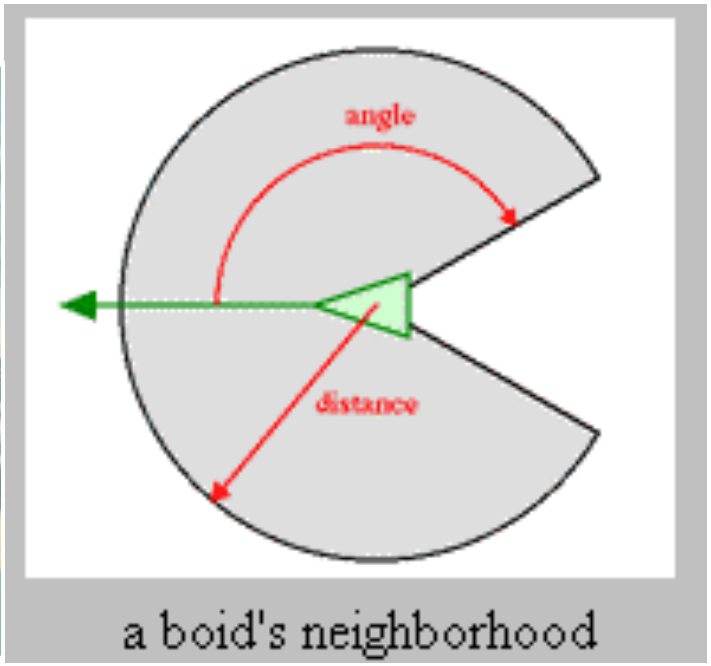


Battle of Helm’s Deep, LOTR

Flocks (“Boids”)

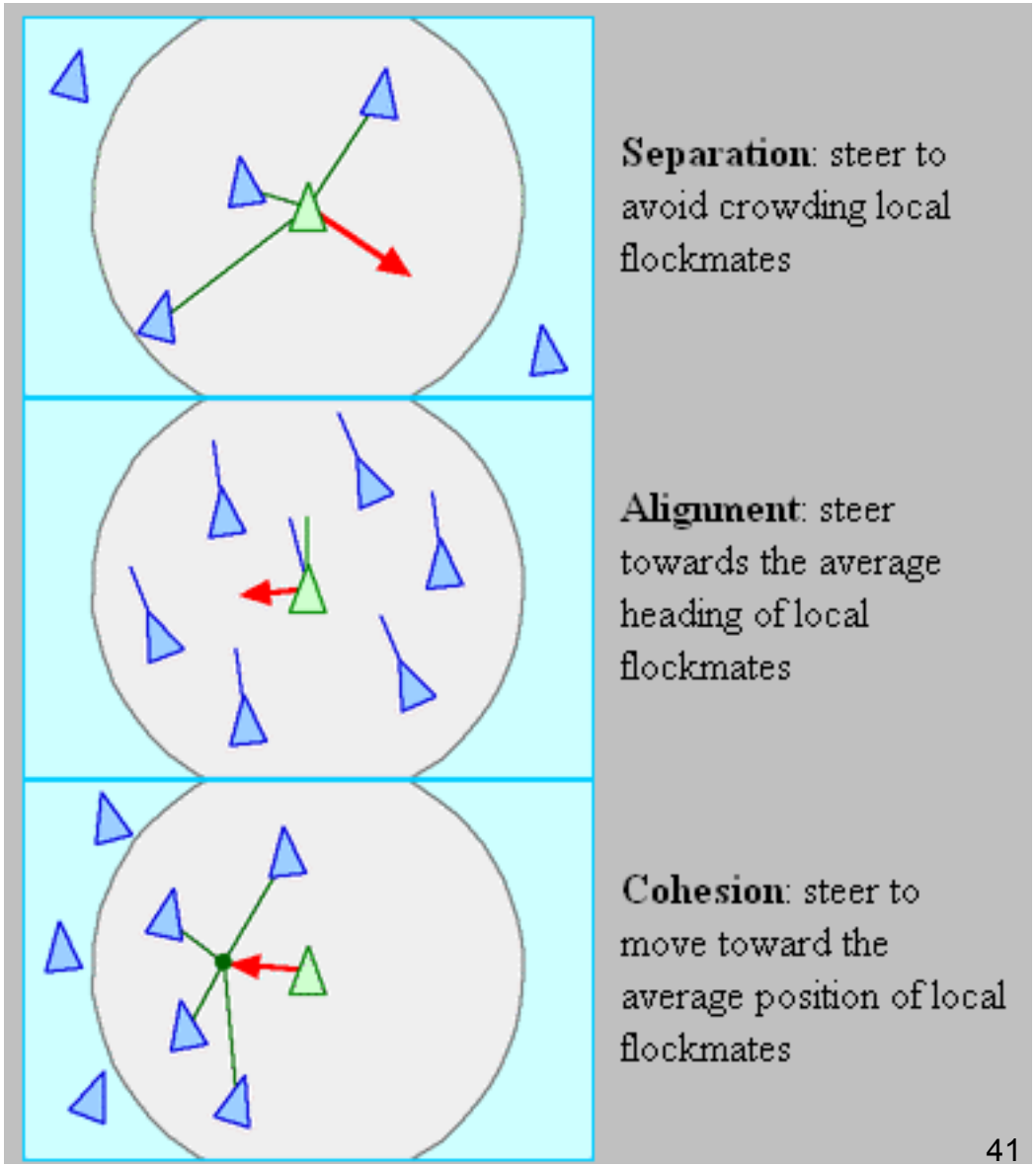
- From Craig Reynolds
- Each bird modeled as a complex particle (“boid”)
- A set of forces control its behavior
- Based on location of other birds and control forces



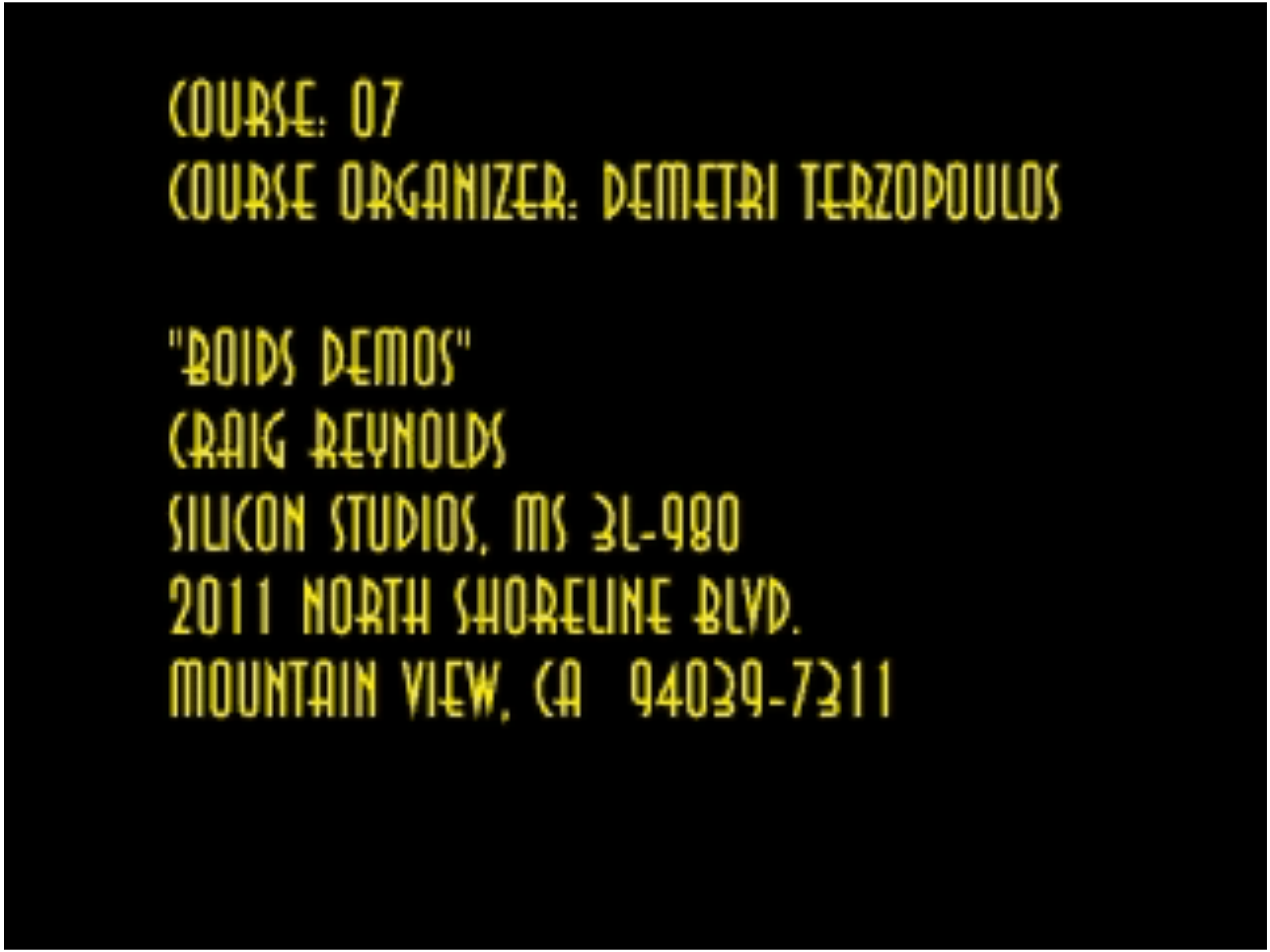


Flocks (“Boids”)

- (“Boid" was an abbreviation of "birdoid". His rules applied equally to simulated flocking birds, and shoaling fish.)



Flocks (“Boids”)



Craig Reynolds

That's All!

- Further reading
  - Witkin, Baraff, Kass: Physically-based Modeling Course Notes, SIGGRAPH 2001
    - **Extremely good, easy-to-read resource. Highly recommended!**
  - William Reeves: Particle systems—a technique for modeling a class of fuzzy objects, Proc. SIGGRAPH 1983
    - The original paper on particle systems