CS 4732: Computer Animation

Physically Based Modeling

Robert W. Lindeman
Associate Professor
Interactive Media & Game Development
Department of Computer Science
Worcester Polytechnic Institute
gogo@wpi.edu

Physics-Based Animation

- Golden Rule of animation (or CG for that matter)
  - If it ***looks*** good enough, it ***is*** good enough.
  - *Physically plausible* or *physically realistic*
- *Forces may or may not reflect actual physics*
  - Directing a ball toward a bat
- This section is about applying forces to control the motion of objects to achieve a desired effect

Physically Based Modeling

- So, why do we call this *modeling* if this course is about *animation*?
  - We are modeling a system of forces
- We need to decide on the level of the process to model
  - Could model the folds in a piece of cloth, or model the forces on each piece of thread, giving rise to wrinkles
  - Tradeoff between physical correctness and computational/ expressive complexity
  - E.g., let the animator manipulate the overall shape of the cloth, and the computer will add wrinkles.
  - Classic control vs. automation balance

Newton is your Friend!

- To animate:
  - Describe all the forces in the system
  - Consider each object in turn
  - Calculate its linear acceleration (ignore rotational dynamics for now) using:

$$f = ma \qquad a = \frac{f}{m}$$

  - From the object's current velocity, *v*, and position, *p*, compute the new velocity, *v′*, and position, *p′*:

$$v' = v + a\Delta t \qquad p' = p + \frac{1}{2}(v + v')\Delta t$$

Use the Force(s)!

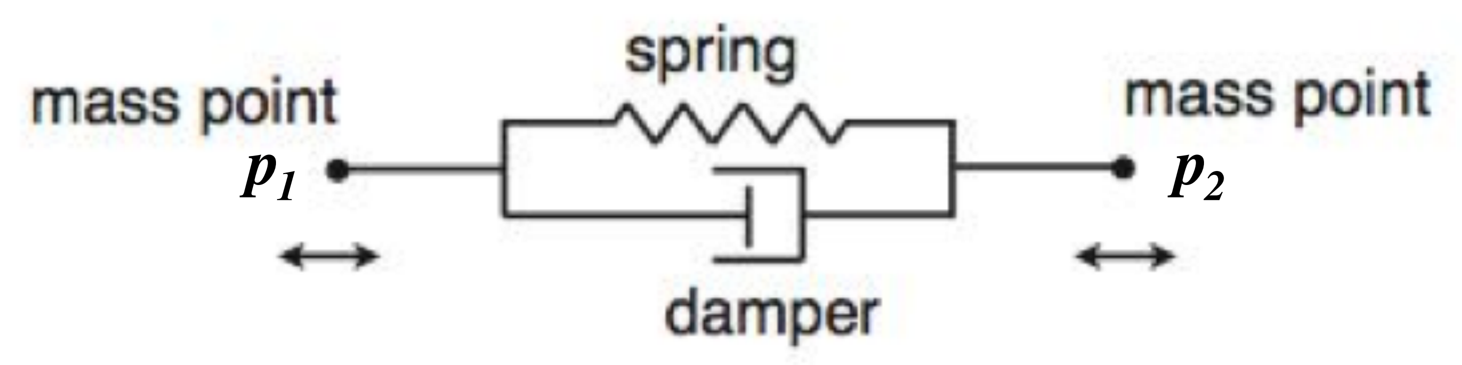
- A variety of forces can be used
  - Position and velocity are very useful
  - Can do ***gravity*** too, if need be, given the masses involved
  - ***Springs*** are also common for flexible objects
    - Cloth, jell-o
  - ***Dampers*** can come in handy too
  - ***Viscosity*** is good for resistance through a medium
  - Conserving ***momentum*** can also be useful for realism

Rotation

- ***Angular velocity*** and ***angular acceleration*** can be used to control object spin
- The ***moment of inertia***, similar to ***mass***, is the measure of resistance to change in rotation

Spring-Damper Pair

- A useful “tool” in animation for realism



$$f = \left(k_s \left(L_c - L_r \right) - k_d \left(\dot{p}_2 - \dot{p}_1 \right) \bullet \left(\frac{p_2 - p_1}{\left\| p_2 - p_1 \right\|} \right) \right) \left(\frac{p_2 - p_1}{\left\| p_2 - p_1 \right\|} \right)$$

- $\boldsymbol{L_r}$ & $\boldsymbol{L_c}$ are the rest and current spring lengths, respectively
- $\boldsymbol{k_s}$ & $\boldsymbol{k_d}$ are the spring and damper constants, respectively

Spring-Damper Animation

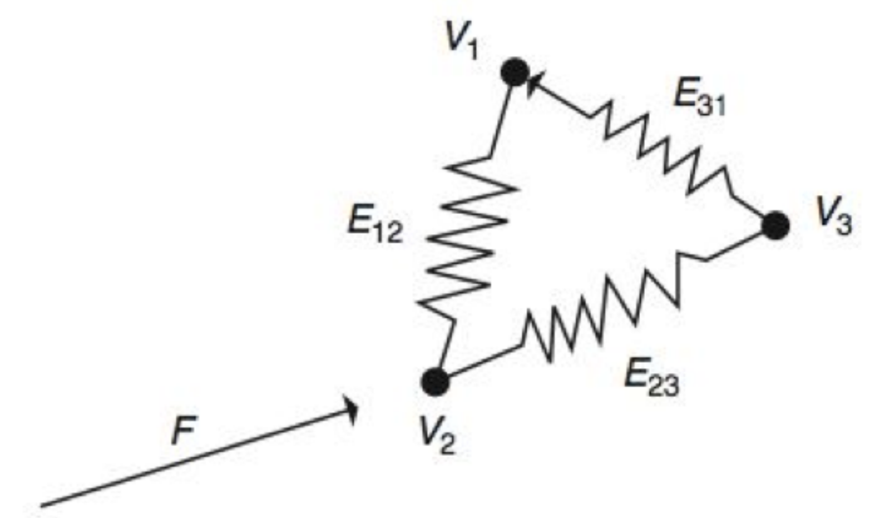
- Flexible shapes are most often modeled using a mesh of vertices (points) and edges (spring-dampers)
- Rest length (L_r) set to the original edge length
- Mass is evenly distributed amongst the vertices
- Spring/damper constants (K_s & K_d) assigned uniformly

Spring-Damper Animation (cont.)

- As forces are applied (e.g., wind, collisions, or whatever), vertices are displaced
  - Leads to spring/damper forces on adjacent vertices
  - Leads to more displacements, etc.
- Result is a wiggling, jiggling object
  - Cool!
- One problem is the time-delay for propagation of the displacements
  - Need to care for number of vertices/edges

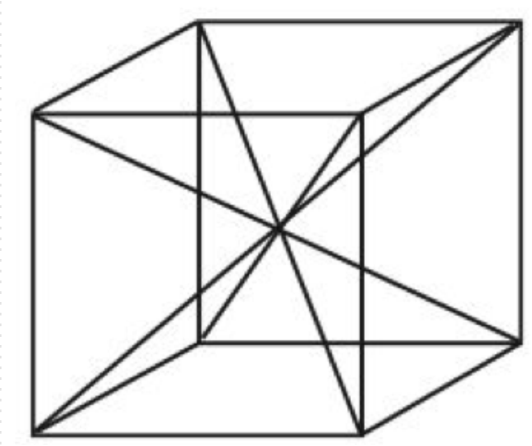
Spring-Damper Animation Example

- Consider an equilateral triangle of springs
- Momentary force is applied to v_2
- Acceleration causes displacement

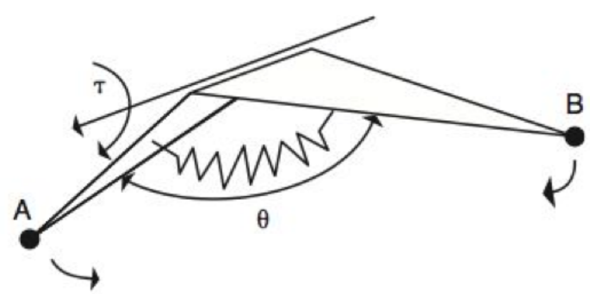


Spring-Damper Animation Example

- Other springs can also use used to improve stability

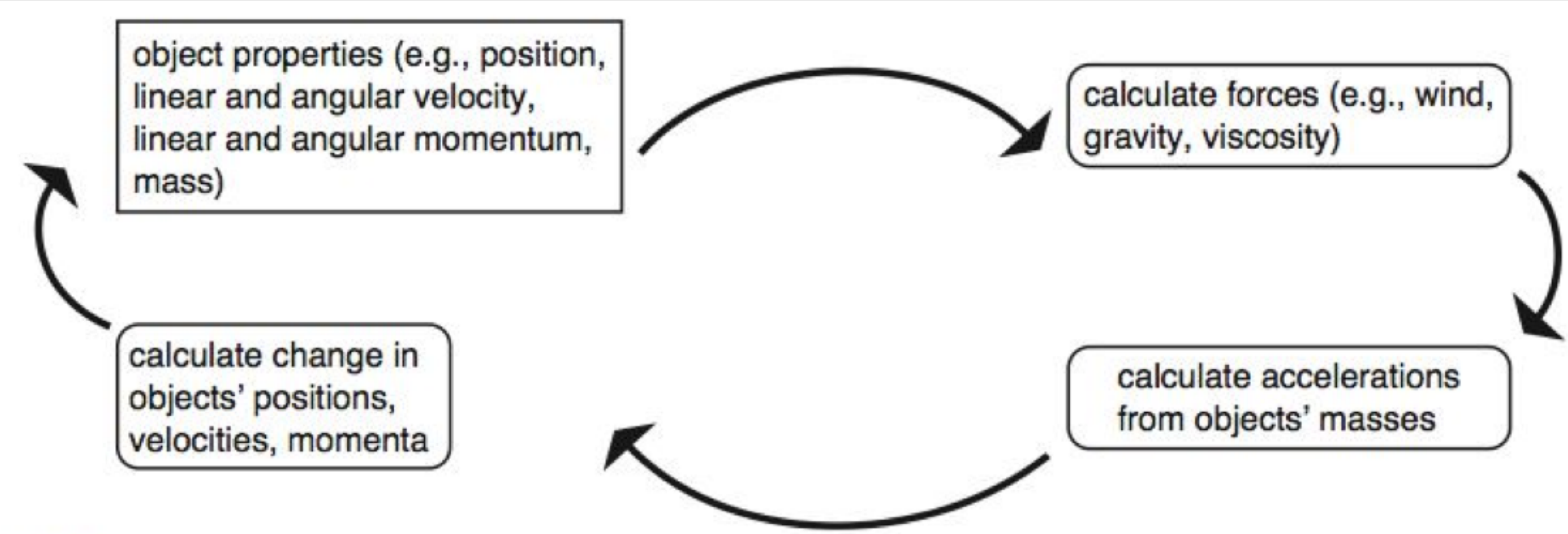


- Angular springs can maintain angles between faces



Rigid Body Simulation

- Given the tradeoff between control and automation, physics can help!



Rigid Body Simulation (cont.)

- Movement in free space is relatively easy
- Collisions, rolling, sliding are another matter!
- Modeling continuous processes in discrete steps can always have problems too
  - E.g., missing collisions
- Traditional physics deals with what to do when events happen
- Computer animation must deal with continuous changes over time

I'm Free, Free Falling…

- Interval between uniform time steps: Δt
- Need to update position (*p*), velocity (*v*), and acceleration (*a*) over time (*t*)
  - Model as functions of time: *x*(*t*), *v*(*t*), *a*(*t*), respectively

$$x(t+\Delta t) = x(t) + v(t)\Delta t$$

$$v(t+\Delta t) = v(t) + a(t)\Delta t$$

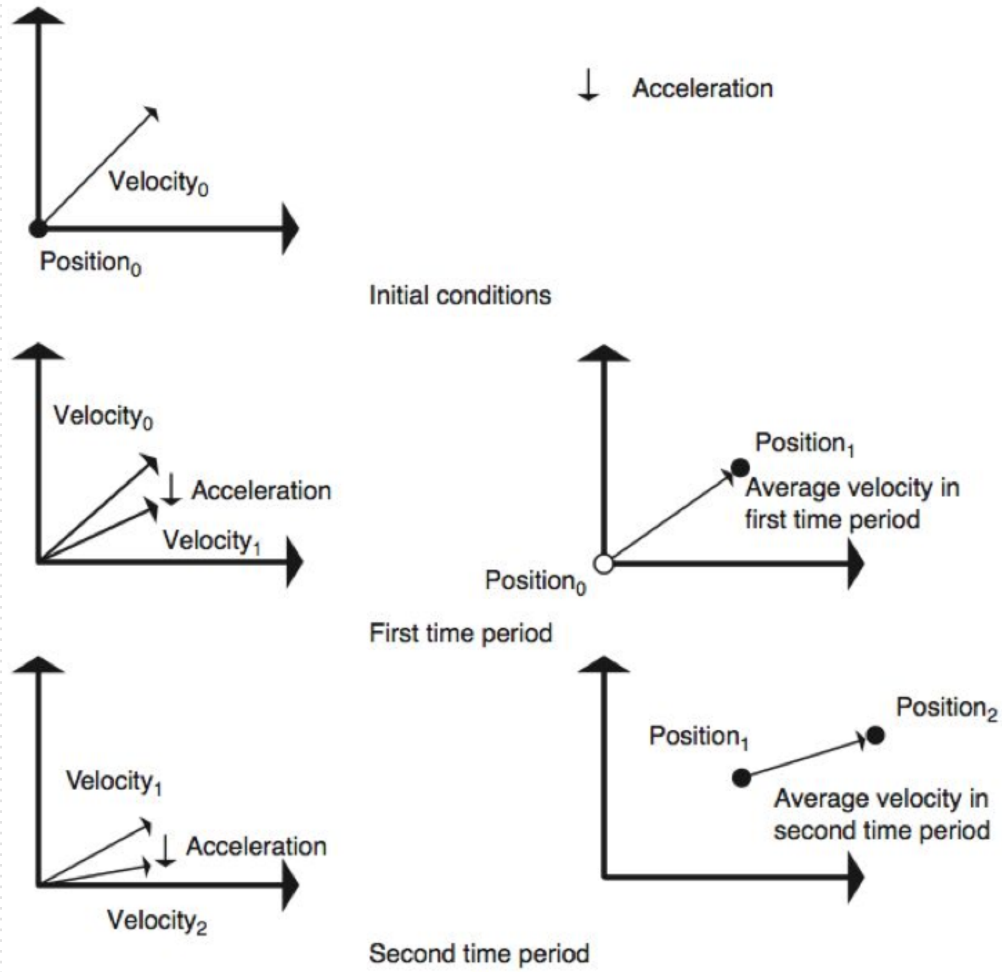
$$x(t+\Delta t) = x(t) + \frac{v(t) + v(t+\Delta t)}{2}\Delta t$$

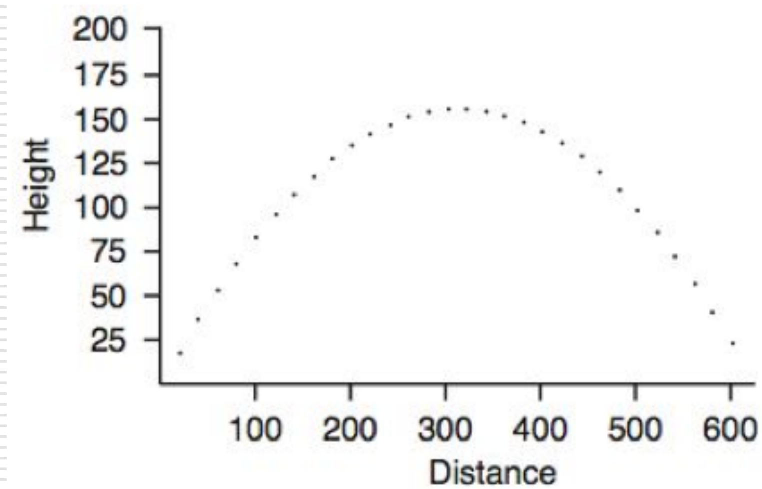
$$x(t+\Delta t) = x(t) + \frac{v(t) + v(t) + a(t) * \Delta t}{2}\Delta t$$

$$x(t+\Delta t) = x(t) + v(t) * \Delta t + \frac{1}{2}a(t) * \Delta t^2$$

*Assume constant acceleration for the duration of the time step

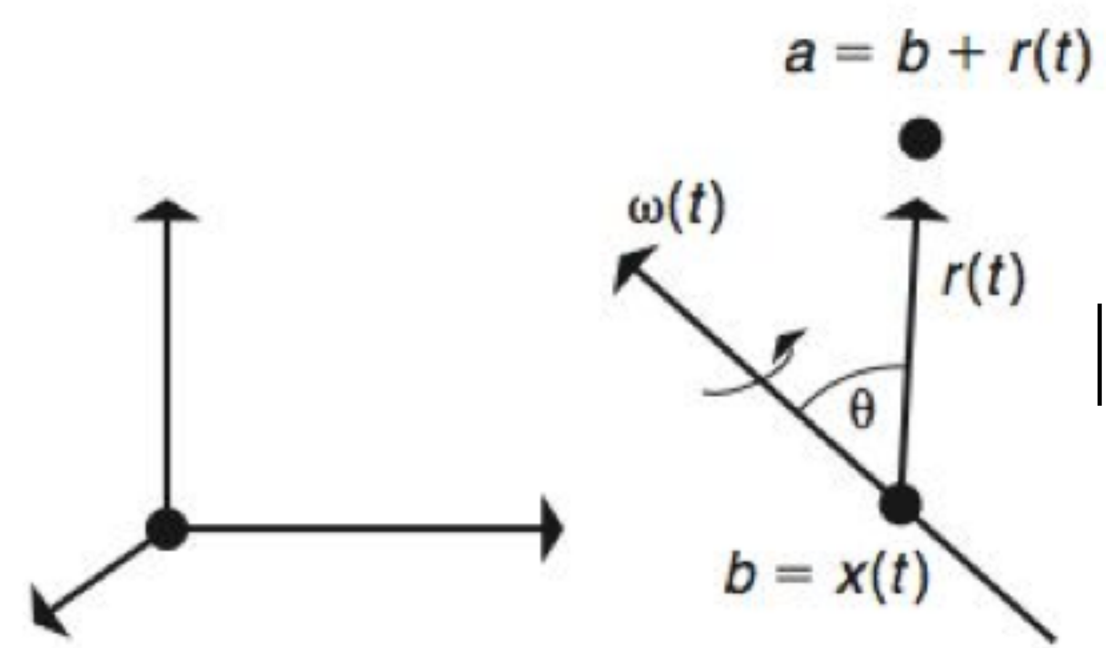
Like a Shot!


Acceleration
Velocity$_0$
Position$_0$
Initial conditions
Velocity$_0$
Acceleration
Velocity$_1$
Position$_1$
Average velocity in first time period
Position$_0$
First time period
Position$_2$
Position$_1$
Velocity$_1$
Average velocity in second time period
Acceleration
Velocity$_2$
Second time period


200
175
150
125
100
75
50
25
Height
100
200
300
400
500
600
Distance

You Spin Me Right Round...

- Consider a point *a* relative to a point *b*
- *a* is rotating about an axis passing through *b*



$$\dot{r}(t) = \omega(t) \times r(t)$$

$$|\dot{r}(t)| = |\omega(t)||r(t)| \sin\theta$$