Physically Based Animation:

Mass-Spring Systems

CS114, Spring 2016

Shuang Zhao

Computer Science Department

University of California, Irvine

Previous Lectures

- WebGL

- Physics of light
- Radiometry
- Rendering equation
- Path tracing

Today's Lecture

- Physically-based animation

- Particle systems
  - Equations of motion
  - Numerical integration (Euler's method)

- Mass-spring systems

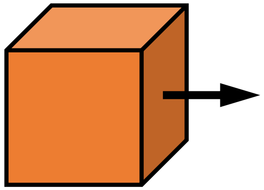
Particle Systems

Physically Based Animation:
Mass-Spring Systems

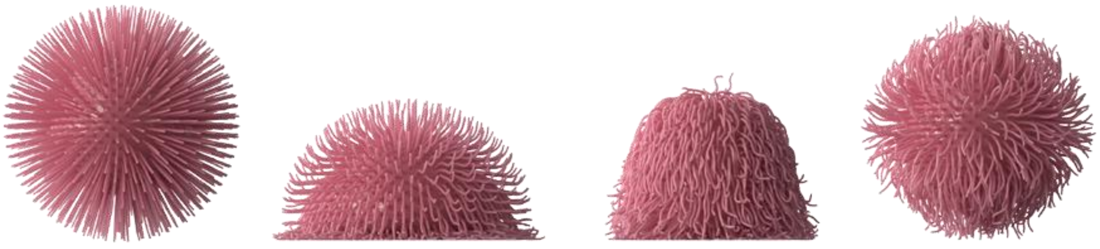
Type of Dynamics

- Point (particle)

- Rigid body



- Deformable body
  (e.g., elastic materials, fluids, smoke)



[Zheng & James 2012]

Particle Systems

- Collections of small particles maintaining certain **states**:
  - E.g., position, velocity, etc.
- Particle motion affected by (internal/external) **forces**
- Simulating particle motion = solving ODEs
- To model:
  - Sand, flame, fluid, **cloth**, etc.

Particle Motion

- Mass m, position $\boldsymbol{x}$, velocity $\boldsymbol{v}$
- Equations of motion: at time t,

Newton's notation

Velocity: $$\frac{d}{dt}\boldsymbol{x}(t) = \boxed{\dot{\boldsymbol{x}}(t)} = \boldsymbol{v}(t)$$

Acceleration: $$\frac{d}{dt}\boldsymbol{v}(t) = \boxed{\ddot{\boldsymbol{x}}(t)} = \frac{1}{m}\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{v}, t)$$

Newton's Second Law

- Simulating a particle's motion:
  - Given $\boldsymbol{x}(0)$, $\boldsymbol{v}(0)$;
  - Need to find $\boldsymbol{x}(t)$, $\boldsymbol{v}(t)$ for $t > 0$.

Initial value problem

Solving Initial Value Problem

- Analytical solutions do NOT exist in general

- We find approximated solutions numerically
  - Euler's method
  - Mid-point method
  - etc.

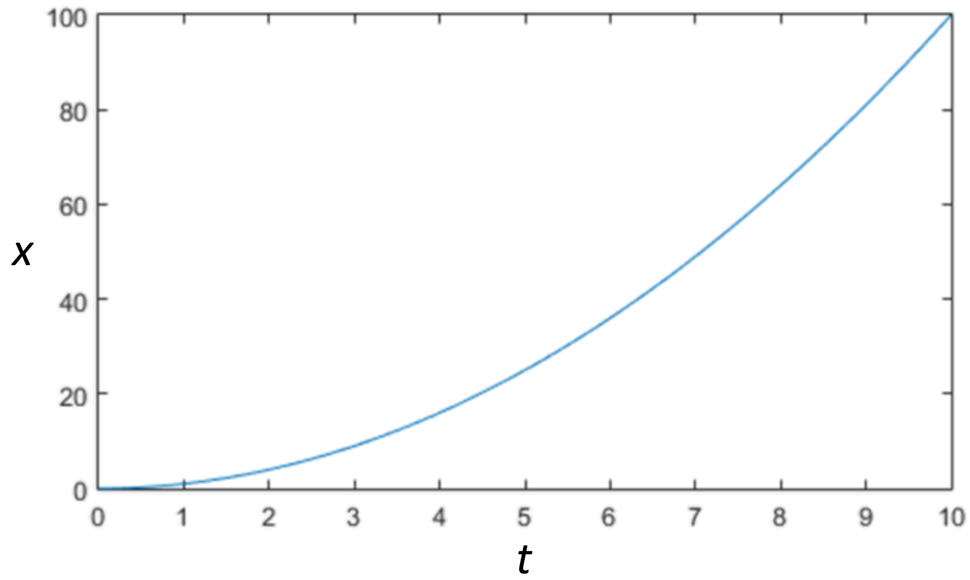
Initial Value Problem: Example

Problem:

$$\begin{cases} \dot{x}(t) = 2t \\ x(0) = 0 \end{cases}$$

Analytical solution:

$$x(t) = t^2$$



Euler's Method

$$\dot{x}(t) = \lim_{\Delta t \to 0} \frac{x(t + \Delta t) - x(t)}{\Delta t} \Rightarrow x(t + \Delta t) \approx x(t) + \Delta t \cdot \dot{x}(t)$$

To solve *x*(*t*), one can pick small Δ*t* > 0 and set

$$\begin{aligned} x(\Delta t) &= x(0) + \Delta t \cdot \dot{x}(0) \\ x(2\Delta t) &= x(\Delta t) + \Delta t \cdot \dot{x}(\Delta t) \\ x(3\Delta t) &= x(2\Delta t) + \Delta t \cdot \dot{x}(2\Delta t) \\ &\cdots \end{aligned}$$

Initial Value Problem: Example

Problem:
$$\begin{cases} \dot{x}(t) = 2t \\ x(0) = 0 \end{cases}$$

Analytical solution:
$$x(t) = t^2$$

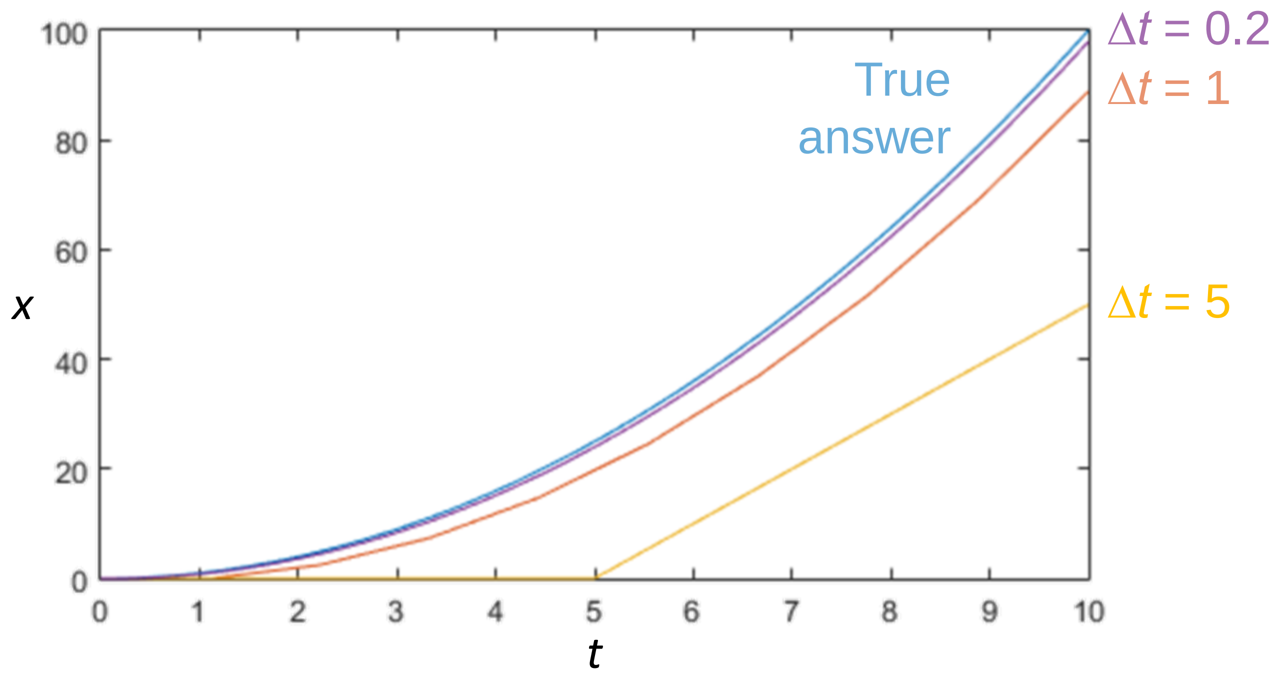
Euler's method:

$$x(\Delta t) = x(0) + \Delta t \cdot \dot{x}(0) = 0$$
$$x(2\Delta t) = x(\Delta t) + \Delta t \cdot \dot{x}(\Delta t) = 2(\Delta t)^2$$
$$x(3\Delta t) = x(2\Delta t) + \Delta t \cdot \dot{x}(2\Delta t) = 6(\Delta t)^2$$
$$\cdots$$

Initial Value Problem: Example

Problem: $$\begin{cases} \dot{x}(t) = 2t \\ x(0) = 0 \end{cases}$$

Smaller Δt = Better accuracy, lower performance



Euler's Method for 2nd Order ODEs

Known

$$\dot{\boldsymbol{x}}(t + \Delta t) = \dot{\boldsymbol{x}}(t) + \Delta t \cdot \boxed{\ddot{\boldsymbol{x}}(t)}$$

$$\boldsymbol{x}(t + \Delta t) = \boldsymbol{x}(t) + \Delta t \cdot \dot{\boldsymbol{x}}(t)$$

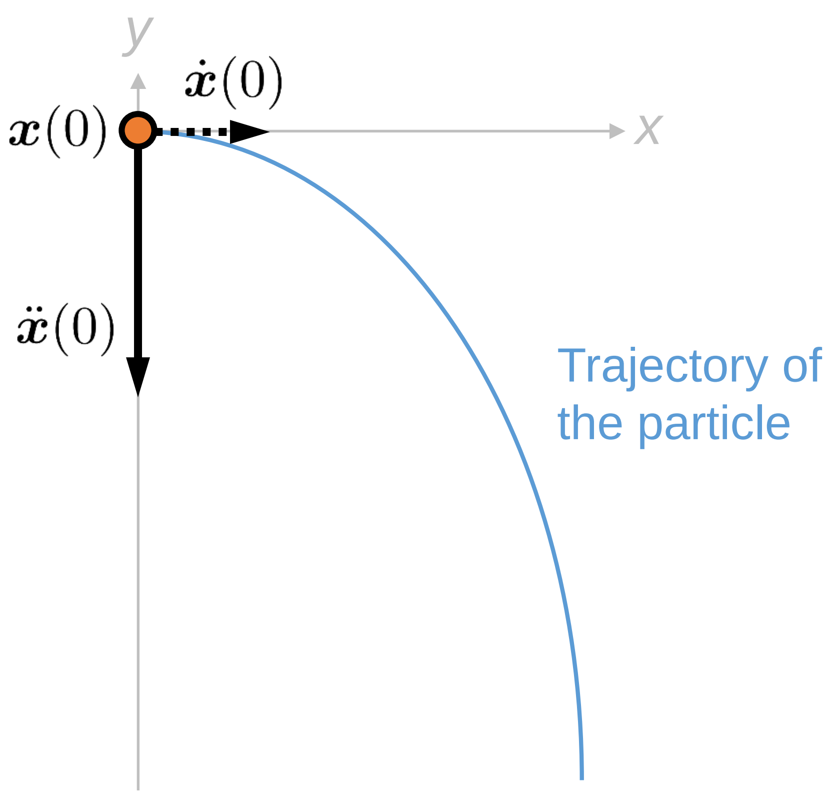
Particle motion problem:

$$\ddot{\boldsymbol{x}}(t) = \frac{1}{m} \boldsymbol{F}(\boldsymbol{x}, \boldsymbol{v}, t)$$

Particle Motion: Example

Problem:

$$\begin{cases} \ddot{\boldsymbol{x}}(t) = \begin{pmatrix} 0 \\ -g \end{pmatrix} & \text{Gravity!} \\ \boldsymbol{x}(0) = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\ \dot{\boldsymbol{x}}(0) = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \end{cases}$$



Analytical solution: $\boldsymbol{x}(t) = \begin{pmatrix} t \\ -\frac{1}{2}gt^2 \end{pmatrix}$

Particle Motion: Example

Problem:

$$\begin{cases} \ddot{\boldsymbol{x}}(t) = \begin{pmatrix} 0 \\ -g \end{pmatrix} \\ \boldsymbol{x}(0) = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\ \dot{\boldsymbol{x}}(0) = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \end{cases}$$

Euler's method:

$$\dot{\boldsymbol{x}}(\Delta t) = \dot{\boldsymbol{x}}(0) + \Delta t \cdot \ddot{\boldsymbol{x}}(0) = \begin{pmatrix} 1 \\ \Delta t \cdot g \end{pmatrix}$$

$$\boldsymbol{x}(\Delta t) = \boldsymbol{x}(0) + \Delta t \cdot \dot{\boldsymbol{x}}(0) = \begin{pmatrix} \Delta t \\ 0 \end{pmatrix}$$

Step 1

$$\dot{\boldsymbol{x}}(2\Delta t) = \dot{\boldsymbol{x}}(\Delta t) + \Delta t \cdot \ddot{\boldsymbol{x}}(\Delta t) = \begin{pmatrix} 1 \\ \Delta 2t \cdot g \end{pmatrix}$$

$$\boldsymbol{x}(2\Delta t) = \boldsymbol{x}(\Delta t) + \Delta t \cdot \dot{\boldsymbol{x}}(\Delta t) = \begin{pmatrix} 2\Delta t \\ (\Delta t)^2 g \end{pmatrix}$$

Step 2

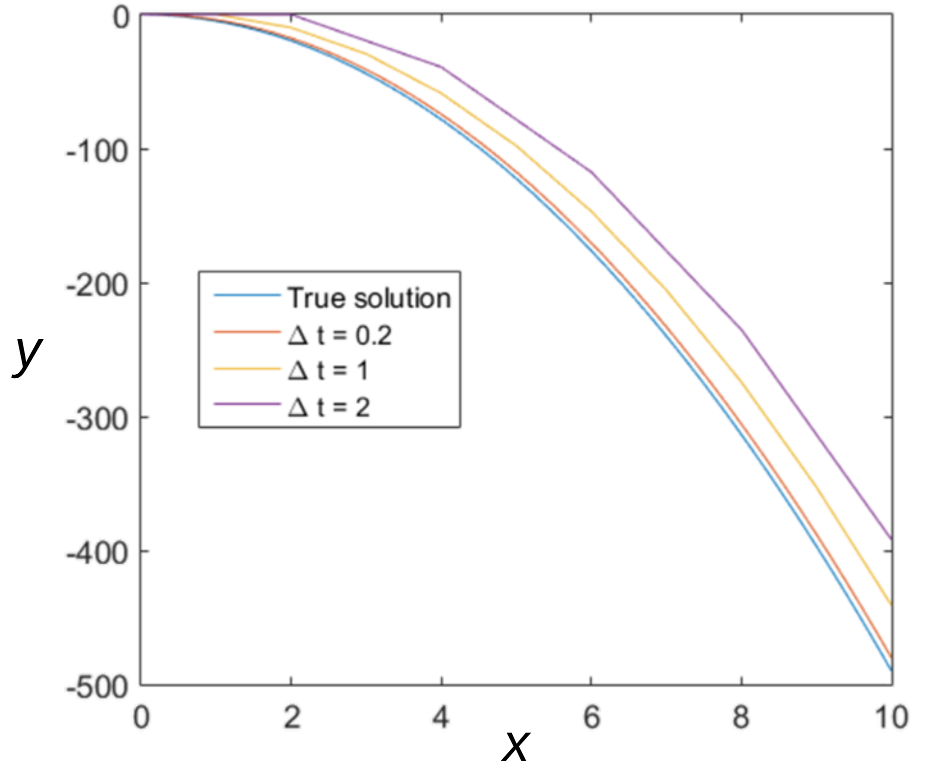
… …

Particle Motion: Example

Problem:

$$\begin{cases} \ddot{\boldsymbol{x}}(t) = \begin{pmatrix} 0 \\ -g \end{pmatrix} \\ \boldsymbol{x}(0) = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\ \dot{\boldsymbol{x}}(0) = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \end{cases}$$

Euler's method:

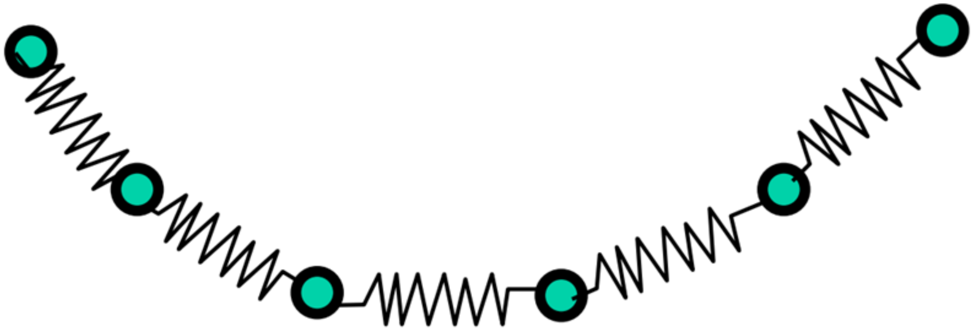


Mass-Spring Systems

Physically Based Animation:
Mass-Spring Systems

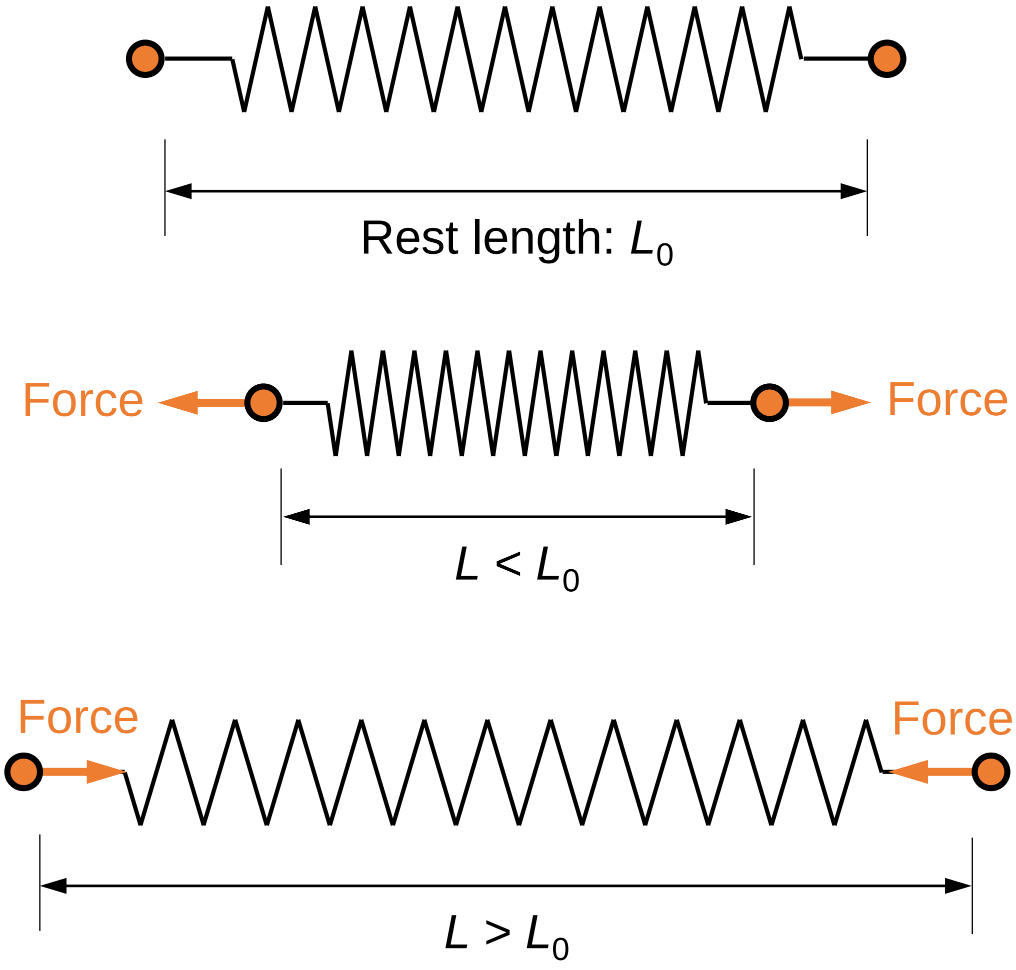
Mass-Spring System

- A collection of particles connected with springs

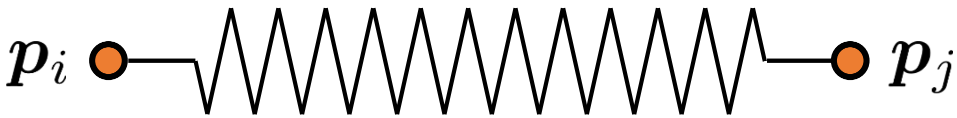


- Forces to consider
  - Spring force
  - Gravity
  - Spatial fields
  - Damping force

Forces: Spring Force



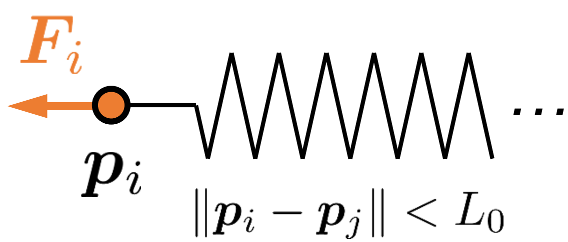
Forces: Spring Force

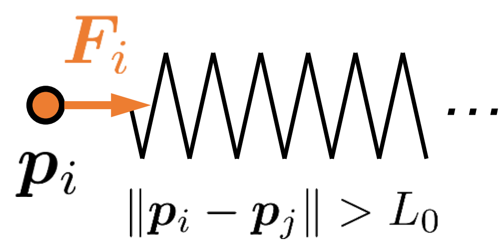


Rest length: L_0

$$\boldsymbol{F}_i = K(L_0 - \|\boldsymbol{p}_i - \boldsymbol{p}_j\|)\frac{\boldsymbol{p}_i - \boldsymbol{p}_j}{\|\boldsymbol{p}_i - \boldsymbol{p}_j\|}$$

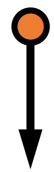
K: stiffness





Forces: Gravity

- Simple gravity:
  - Depends only on particle mass *m*



$$\boldsymbol{F}_{\text{gravity}} = \begin{pmatrix} 0 \\ 0 \\ -mg \end{pmatrix}$$

(g = 9.8 on earth)

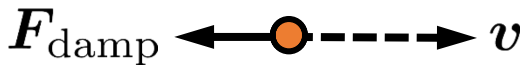
Forces: Spatial Fields

- Force on a particle only depends on its location
  - E.g., wind

- Can change over time

Forces: Damping

$$\boldsymbol{F}_{\mathrm{damp}} = -c_d\, \boldsymbol{v}$$



- Force on a particle
  - Only depends on its velocity
  - Opposes motion

- Removes energy, so system can settle
- Small amount of damping can stabilize solver

Animating Mass-Spring System

- Input:
  - For each particle *i:*
    - Mass m_i, initial position $\boldsymbol{x}_\mathrm{i}(0)$, initial velocity $\boldsymbol{v}_\mathrm{i}(0)$
  - For each spring connecting particles *i* and *j*:
    - Stiffness $K_{i,j}$
- Output:
  - Trajectory of each particle $\boldsymbol{x}_\mathrm{i}(t)$, $t > 0$
  - Normally represented with positions at discrete time-steps: $\boldsymbol{x}_\mathrm{i}(\Delta t)$, $\boldsymbol{x}_\mathrm{i}(2\Delta t)$, $\boldsymbol{x}_\mathrm{i}(3\Delta t)$, …

Animating Mass-Spring System

- Pseudo-code

```
t = 0
while t < t_max:
    for each particle i:
        compute accumulated force F_i at this particle

        # Euler's method
        a_i = F_i/m_i                          # acceleration
        v_i(t + Δt) = v_i(t) + (Δt)*a_i        # velocity
        x_i(t + Δt) = x_i(t) + (Δt)*v_i(t)     # position

    t = t + Δt
```

Computing Accumulated Force

```
# Given particle i at time t
```

$\boldsymbol{F}_i$ = [0, 0, $-m_i * g$] # gravity

$\boldsymbol{F}_i$ = $\boldsymbol{F}_i$ – $c_d * \boldsymbol{v}_i(t)$ # damping

for each particle j connected to i by a Spring:

  $\boldsymbol{e} = \boldsymbol{P}_i - \boldsymbol{P}_j$

  # $K_{i,j}$: Stiffness, $L_{i,j}$: rest length

  $\boldsymbol{F}_{spring} = K_{i,j} * [L_{i,j} - \text{length}(\boldsymbol{e})] * \boldsymbol{e} / \text{length}(\boldsymbol{e})$

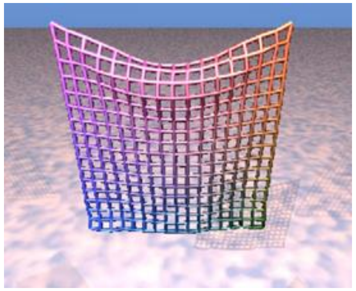
  $\boldsymbol{F}_i = \boldsymbol{F}_i + \boldsymbol{F}_{spring}$

Mass-Spring System for Cloth Simulation

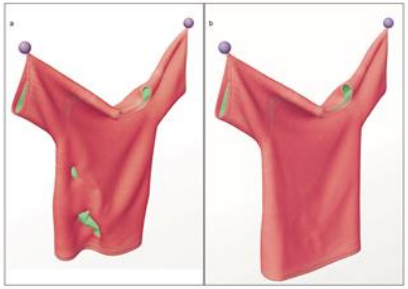
Physically Based Animation:
Mass-Spring Systems

Cloth Simulation

- Studied for decades, still an active area
- Elastic sheet model



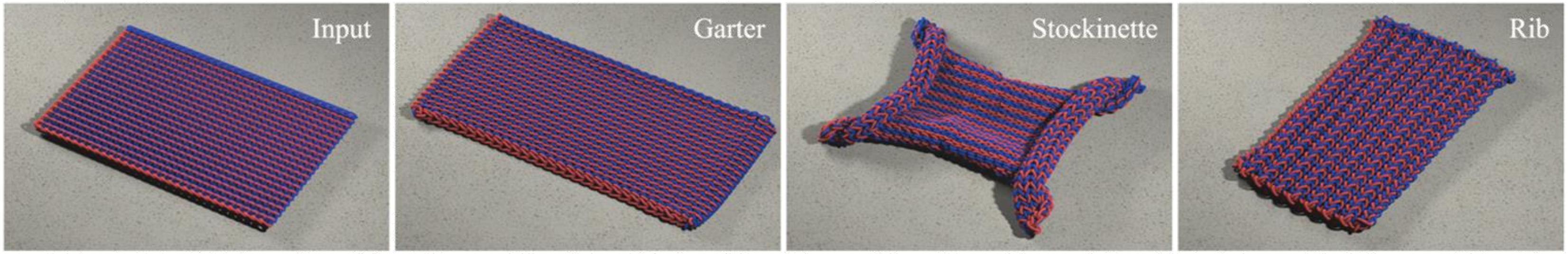
[Provot 1995]



[Baraff et al. 2003]

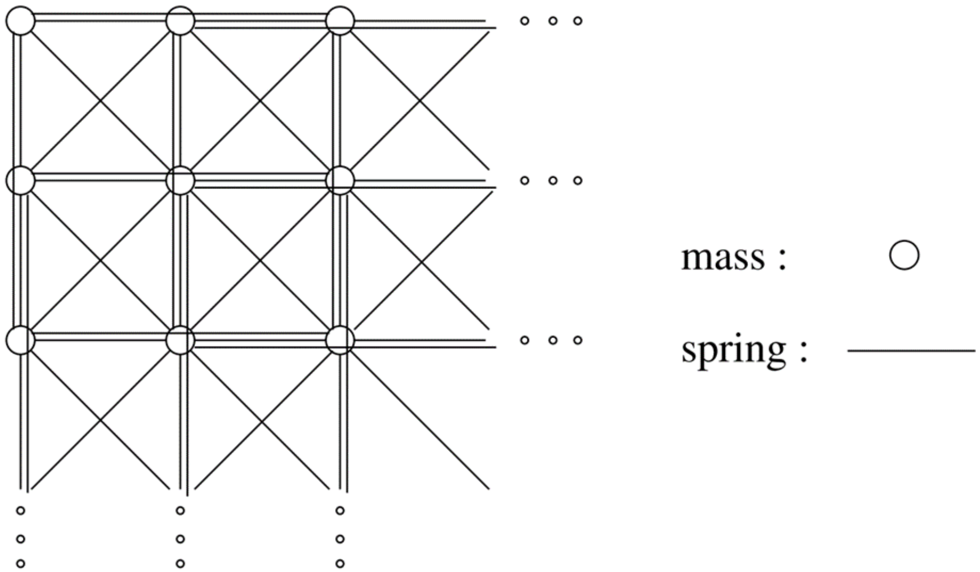
… …

- Yarn-based model



[Kaldor et al. 2008]

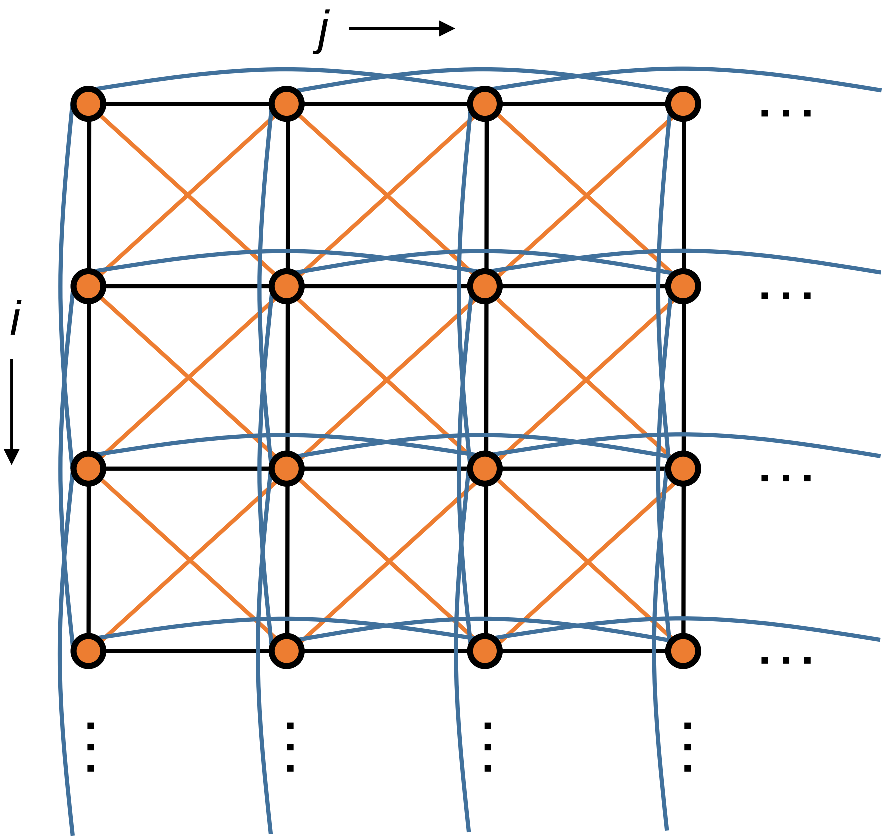
Mass-Spring System for Cloth



- Cloth = a “web” of particles and springs

Mass-Spring System for Cloth

- Consider a rectangular cloth with $m \times n$ particles



Types of springs

Structural

$[i, j]$—$[i, j + 1]$; $[i, j]$—$[i + 1, j]$

Shear

$[i, j]$—$[i + 1, j + 1]$; $[i + 1, j]$—$[i, j + 1]$

Flexion (bend)

$[i, j]$—$[i, j + 2]$; $[i, j]$—$[i + 2, j]$

Mass-Spring System for Cloth

- Type of Forces
  - Spring
  - Gravity
  - Damping
  - Contact
  - …

Discretization

- What happens if we discretize our cloth more or less finely?
- Do we get the same behavior?

- Usually NOT! It takes a lot of effort to design discretization-independent schemes.