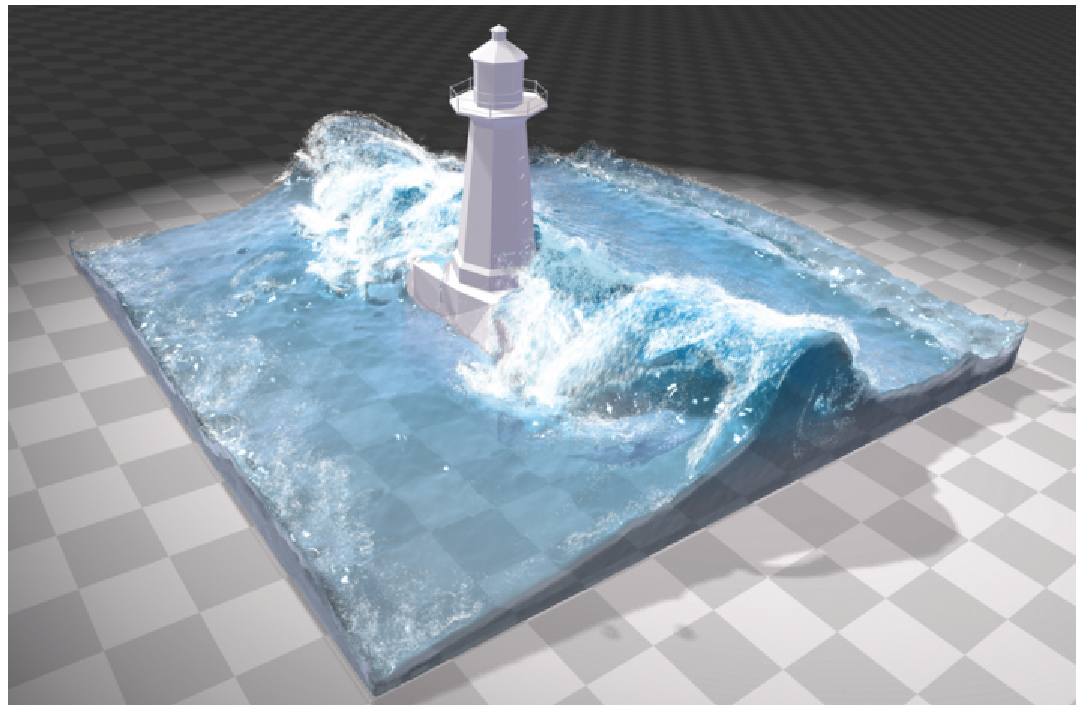
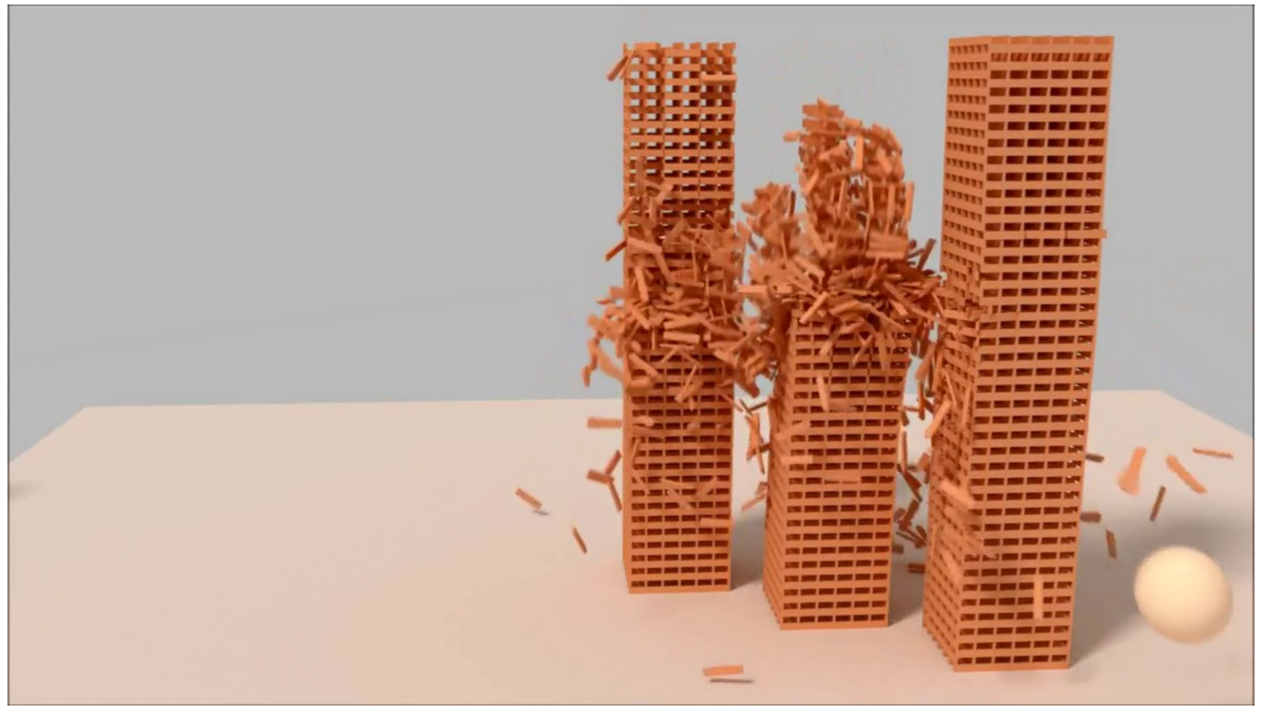
Physically-based simulation

Deformable models

When physically based simulation is needed

- Accurate dynamics
- Teadious to model by hand or procedurally
  - Multiple interacting elements: ex. Multiple collisions: rigid bodies, hairs, etc.
  - Complex animated geometry: Cloths, fluids





Material model

Elasticity: Shape goes back toward its original rest position when external forces are removed.

-Purely elastic models don't loose energy when deformed (potential ↔ kinetic)

Plasticity: Oppositve of elasticity. Plastic material don't come back to their original shape (/change their rest position during deformation).

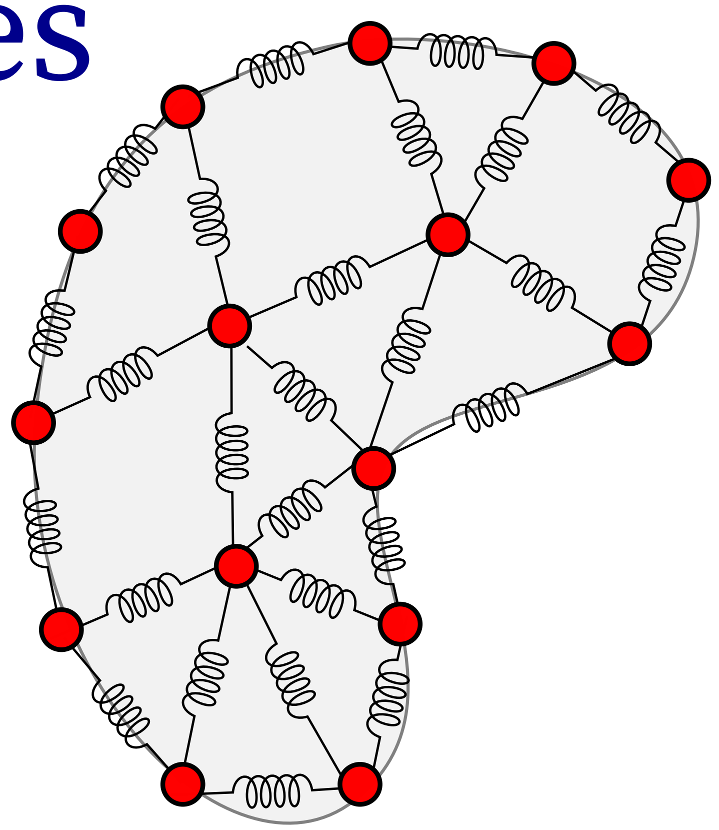
- Ductile material - can allow large amount of plastic deformation without breaking (plastic)
- Brittle - Opposite (glass, ceramics)

Viscosity: Resistance to flow (usually for fluid, ex. honey)

In reality

- *Elasto-plastic materials*: Allow elastic behavior for small deformation, and plastic at larger one.
- *Visco-elastic materials*: Elastic properties with delay.

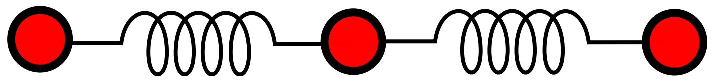
Modeling elastic shapes with particles



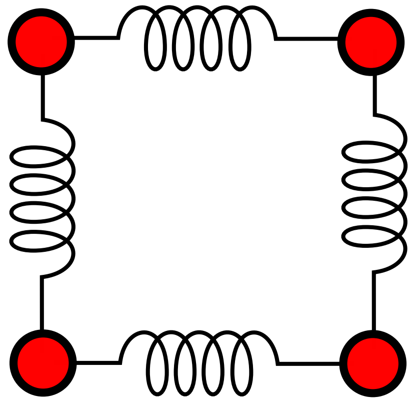
Spring mass systems

- Particles (position, speed, mass): samples on shape
- Springs : link closed-by particles in the reference shape

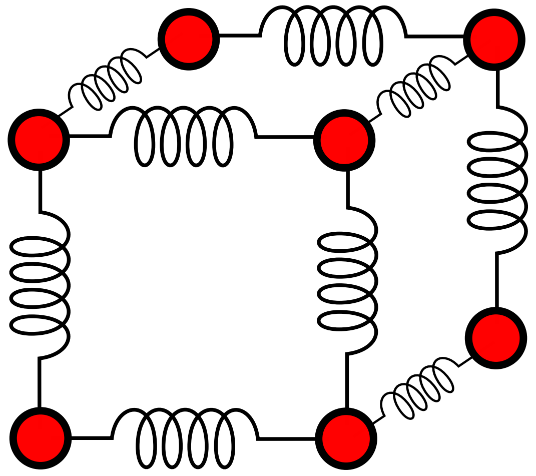
1D curve structure



2D surface structure



3D volume structure

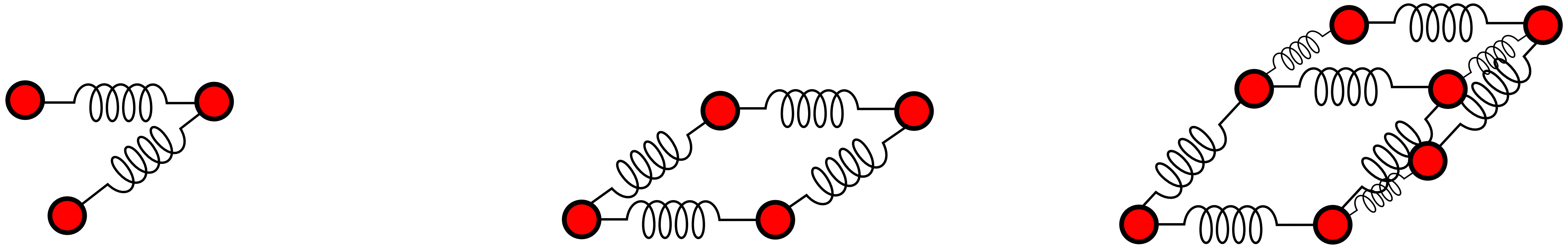


Spring structure

How to model spring connectivity ?

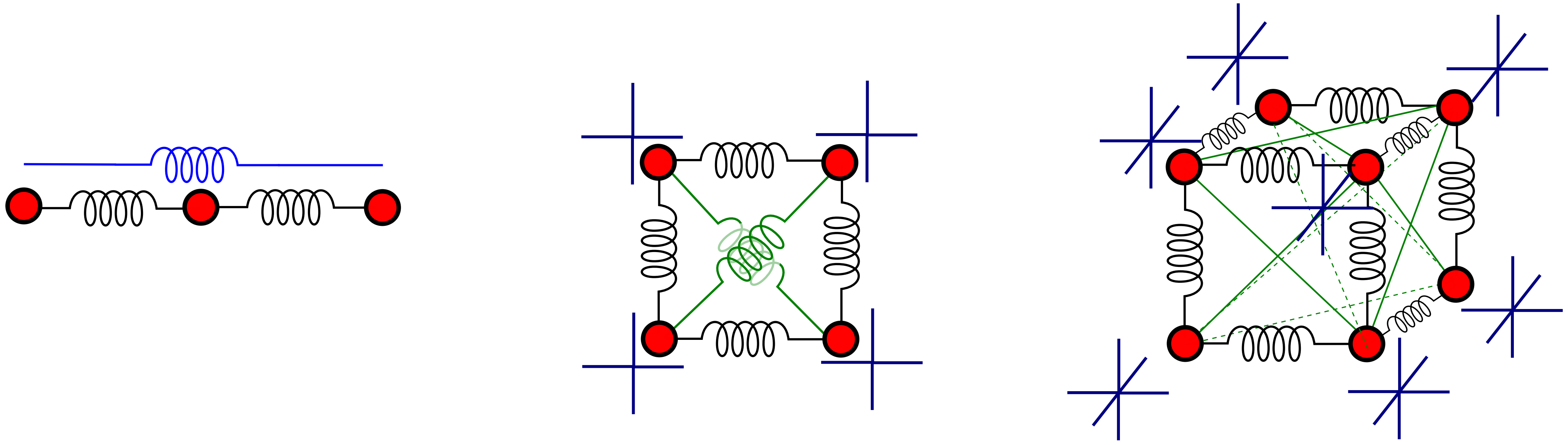
- **Structural springs**: 1-ring neighbors springs ($\simeq$ mesh edges)

  (+) Limit elongation/contraction, (-) Allows shearing, and bending



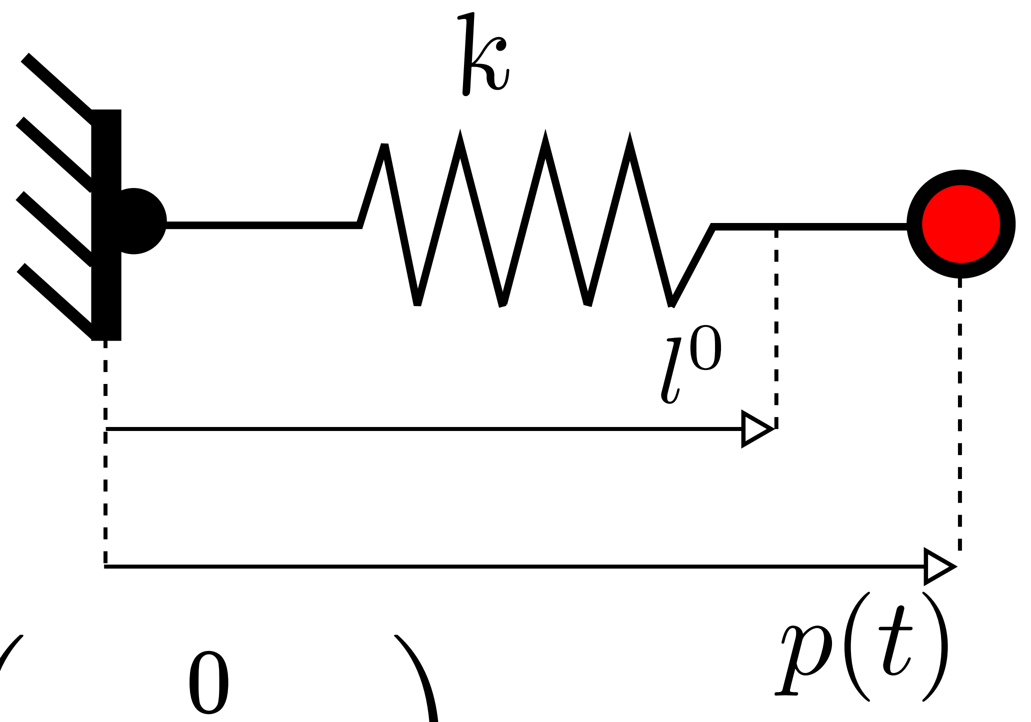
$\Rightarrow$ Add extra springs connectivity

- **Shearing springs**: Diagonal links
- **Bending springs**: 2-ring neighborhood



Example: 1D spring (/Harmonic oscillator)

- Force $F(t) = -k\,(p(t) - l^0)$, k spring stiffness, l^0 rest length
- Second order differential equation: $m\,p''(t) + k\,p(t) = k\,l^0$



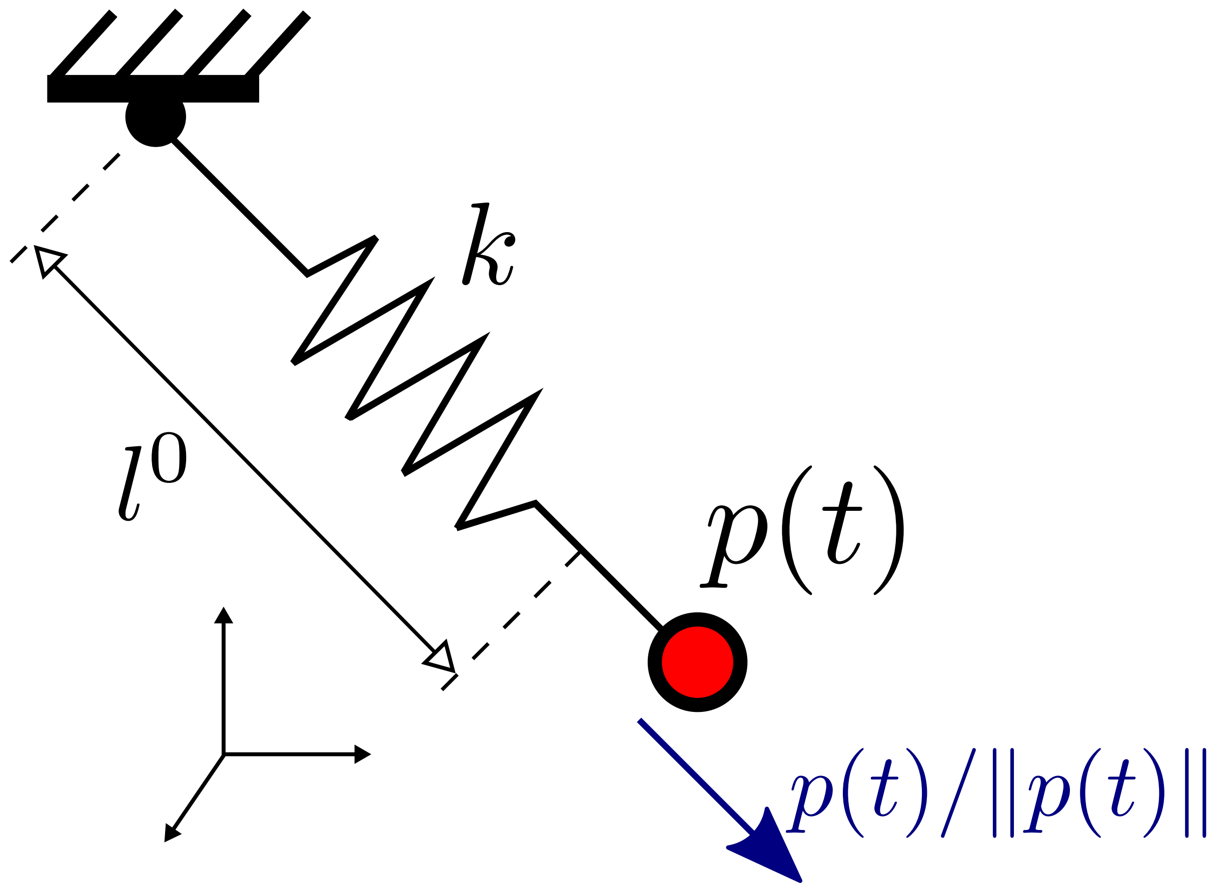
- First order system $\underbrace{\begin{pmatrix} p \\ v \end{pmatrix}'(t)}_{u'(t)} = \underbrace{\begin{pmatrix} v(t) \\ -k/m\,\left(p(t) - l^0\right) \end{pmatrix}}_{\mathcal{F}(u,t)}$
- Linear system $\underbrace{\begin{pmatrix} p \\ v \end{pmatrix}'(t)}_{u'(t)} = \underbrace{\begin{pmatrix} 0 & 1 \\ -k/m & 0 \end{pmatrix}}_{\mathrm{A}} \underbrace{\begin{pmatrix} p \\ v \end{pmatrix}(t)}_{u(t)} + \underbrace{\begin{pmatrix} 0 \\ k/m\,l^0 \end{pmatrix}}_{b}$

- Exact solution known: $p(t) = A\,\sin(\omega\,t + \varphi) + l^0$

$$\omega = \sqrt{k/m},\, A^2 = (p^0 - l^0)^2 + \left(\frac{v_0}{\omega}\right)^2,\, \tan(\varphi) = \frac{p^0 - l^0}{v^0/\omega}$$

Example: 3D mass spring

- Force $F(t) = m\,g - k\,(\|p(t)\| - l^0)\,\frac{p(t)}{\|p(t)\|}$, k spring stiffness, l^0 rest length
- Second order differential equation: $m\,p''(t) = mg - k\,(\|p(t)\| - l^0)\,\frac{p(t)}{\|p(t)\|}$
- First order system $\underbrace{\begin{pmatrix} p \\ v \end{pmatrix}'(t)}_{u'(t)} = \underbrace{\begin{pmatrix} v(t) \\ g - k/m\,(\|p(t)\| - l^0)\,\frac{p(t)}{\|p(t)\|} \end{pmatrix}}_{\mathcal{F}(u,t)}$
- Not linear
- No simple explicit solution



Numerical integration of ODE

General formulation: $u'(t) = \mathcal{F}(u,t), u(t) = (p(t), v(t))$.

Explicit Euler

$u^{k+1} = u^k + \Delta t\, \mathcal{F}(u^k, t^k)$

(+) Easy to implement

(-) Worst scheme in all cases (divergence, low accuracy)

Explicit Runge-Kutta

$u^{k+1} = u^k + \Delta t \sum_j \alpha_j k_j$

(+) Good **accuracy**

(+) Efficient to apply

(+/-) Stability OK for non-stiff problem, diverge on stiff problem

(-) Artificial damping for constant energy system

Implicit methods

$u^{k+1} = u^k + \Delta t\, \mathcal{F}(u^{k+1}, t^{k+1})$

(+) Good to deal with **stiff problem** - very stable

(-) Add numerical damping (converge even if solution oscillates)

(-) Hard/computationally costly to apply on non linear problem

Symplectic integrator

$v^{k+1} = v^k + \Delta t\, F^k/m$

$p^{k+1} = p^k + \Delta t\, v^{k+1}$

(+) Handle well constant energy system, preserves energy (Hamiltonian systems)

(+) Simple and efficient to implement

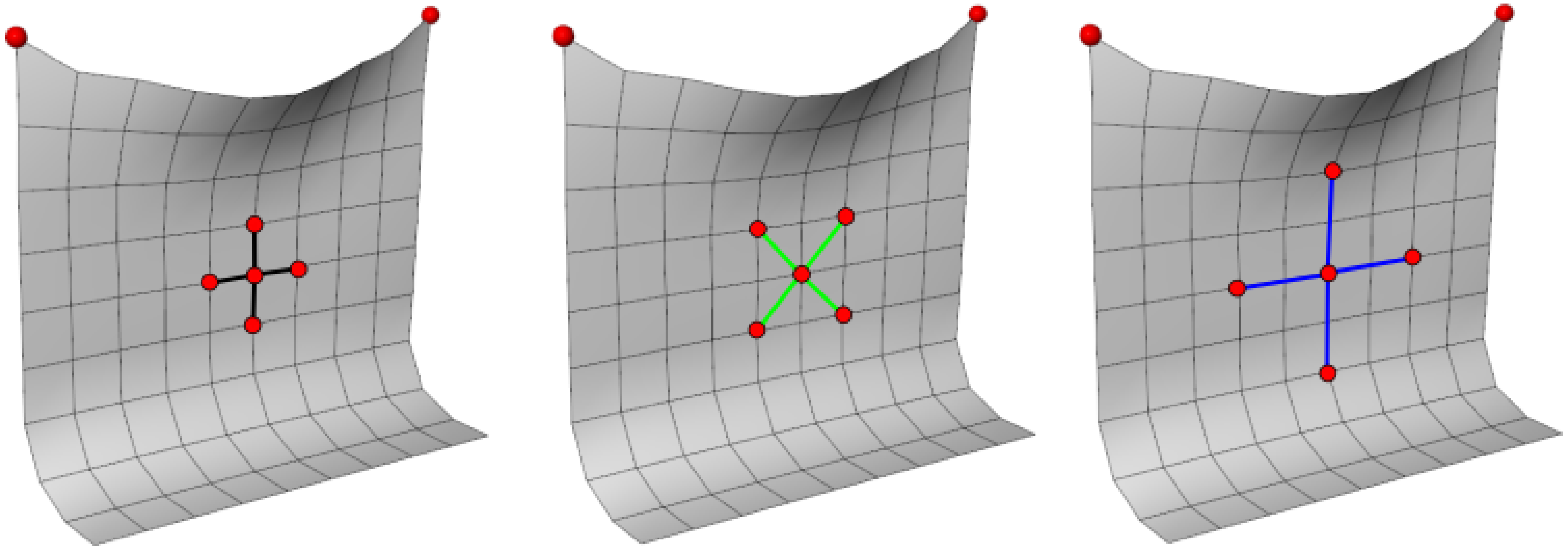
(-) Less accurate than RK

(-) Diverge on stiff problem

Cloth Simulation

Mass-spring cloth simulation

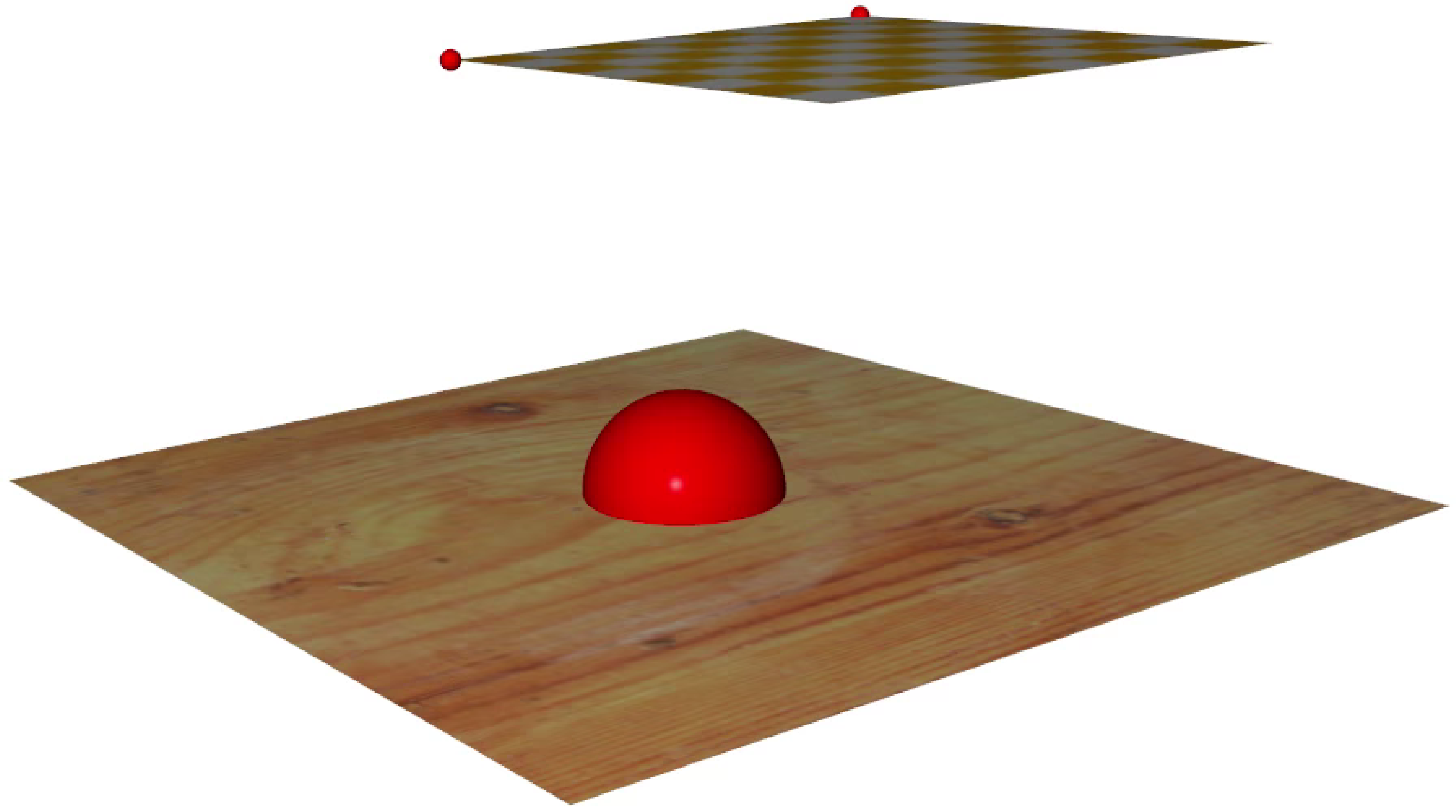
- Particles are sampled on a $N \times N$ grid.
  - Each particle has a mass m ($m_{cloth} = N^2\, m$)
- Set structural, shearing and bending springs.



Forces

- On each particle: gravity + drag + spring forces
- $F_i(p, v, t) = m_i\, g - \mu\, v_i(t) + \sum_{j \in \mathcal{V}_i} K_{ij}\, \left(\|p_j(t) - p_i(t)\| - L_{ij}^0 \right) \frac{p_j(t) - p_i(t)}{\|p_j(t) - p_i(t)\|}$
  - $\mathcal{V}_i$: neighborhood of particle i
  - L_{ij}^0: rest length of spring ij

Associated ODE $\forall i, \begin{cases} p_i'(t) = v_i(t) \\ v_i'(t) = F_i(p, v, t) \end{cases}$

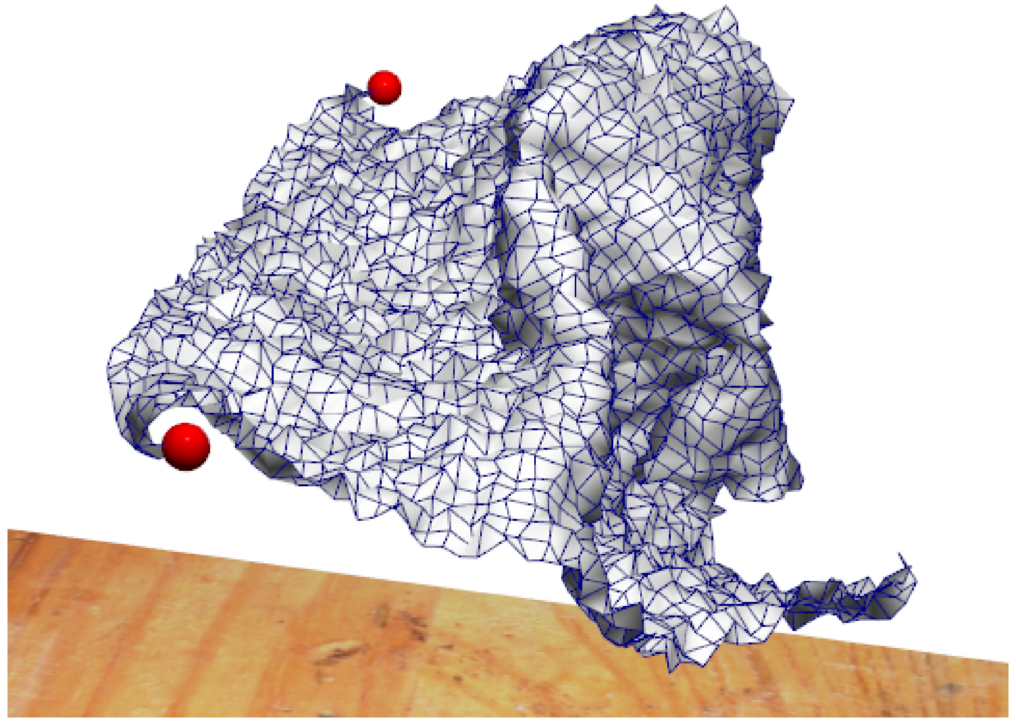


Note on Mass-Spring numerical solution

- Non-linear ODE

- Large K_{ij} : good length preservation, but stiff ODE
  $\Rightarrow$ divergence of explicit schemes.
- Avoid explicit Euler (divergence)

- Semi-implicit Euler/Verlet works fine for low K_{ij}
  Semi-implicit Euler + PBD allows simple integration + stable stiff springs
  [Muller et al. PBD , Inextensible clothing in Computer Games]



- RK4 more accurate (but higher complexity than Verlet)

- Implicit Euler : requires linearization, but very stable

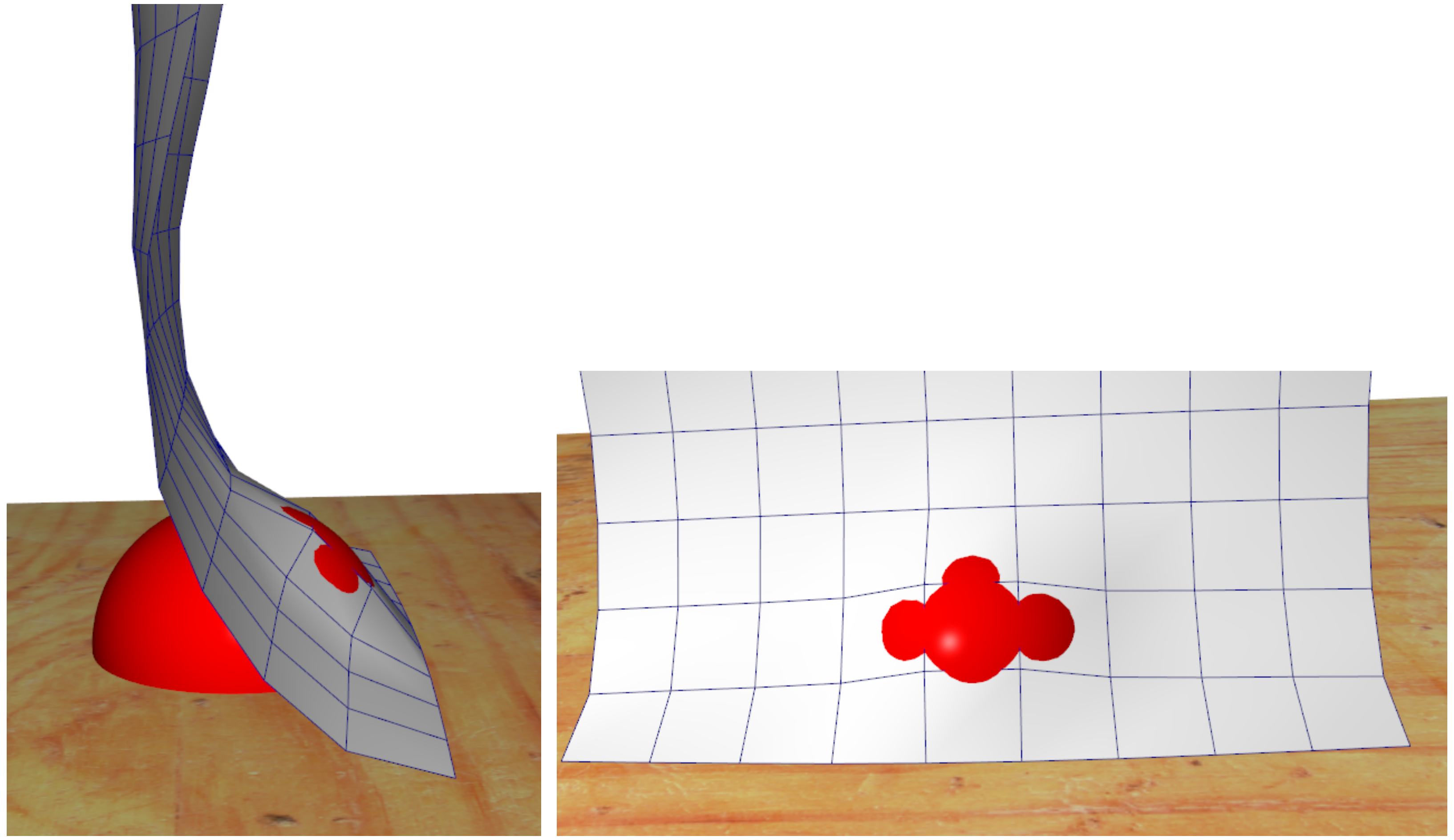
Collisions

- Simple approach : Handled as collision between particles and shapes

  (+) Simple and efficient

  (-) Collision may still appears within a triangle

  $\Rightarrow$ Exaustive approach: point-face + edge-edge



Limitation of mass spring model and continuous model

- Does mass-spring system converge toward a unique solution when sampling increase ?

$\Rightarrow$ No :(

Depends on the connectivity $\rightarrow$ bad for physical accuracy

Corollary

- Mass-springs work well for grid-mesh structure (draping)
- Less for arbitrary triangular meshes

1st improvment: Change toward energy formulation for bending springs (limits locking effect)

$$F = \frac{\partial E}{\partial p}$$

$$E = \frac{1}{2} K\, L\, \kappa^2$$, κ: curvature

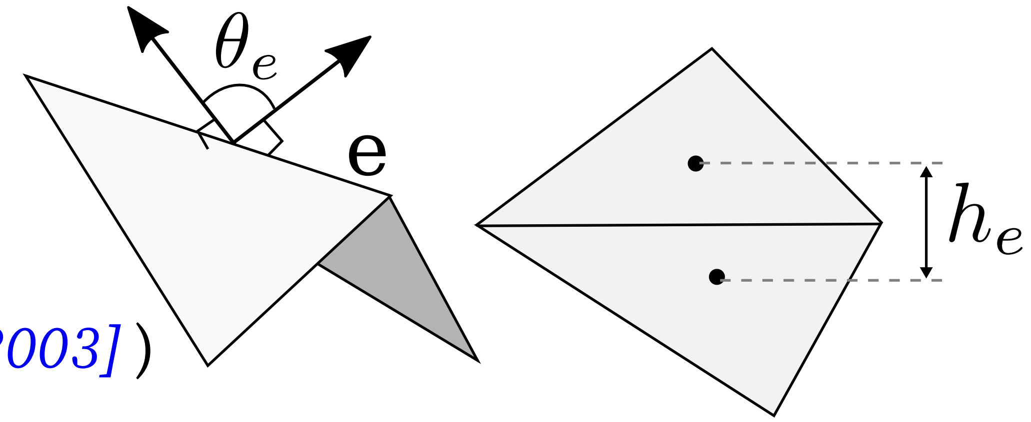
[Cho et al, Stable but Responsive Cloth, ACM SIGGRAPH 2002]

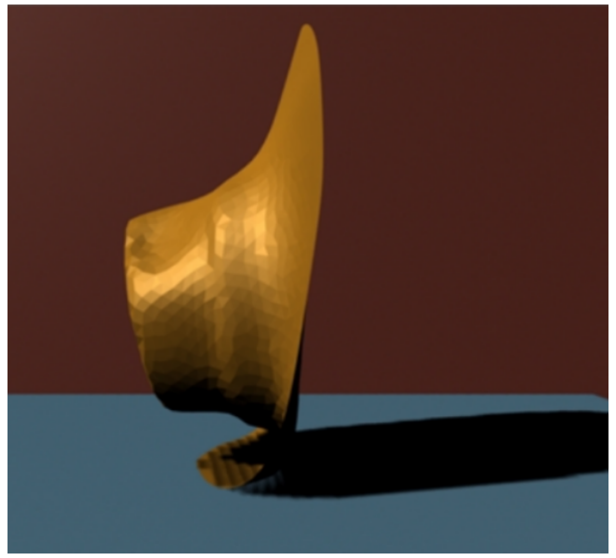
Triangle as continuous elements

- Defining Bending Energy between triangles
  - $W_B(x) = \sum_{edges\ e} (\theta_e - \theta_e^0) \frac{\|e^0\|}{h_e^0}$

  [E. Grinspun et al., Discrete Shells, SCA 2003]

  (or expressed using forces in [R. Bridson et al., SCA 2003])





- Going toward full FEM numerical resolution
  - B. Thomaszewski et al. [SCA 2006], [VRIPHYS 2008], [EG 2009].