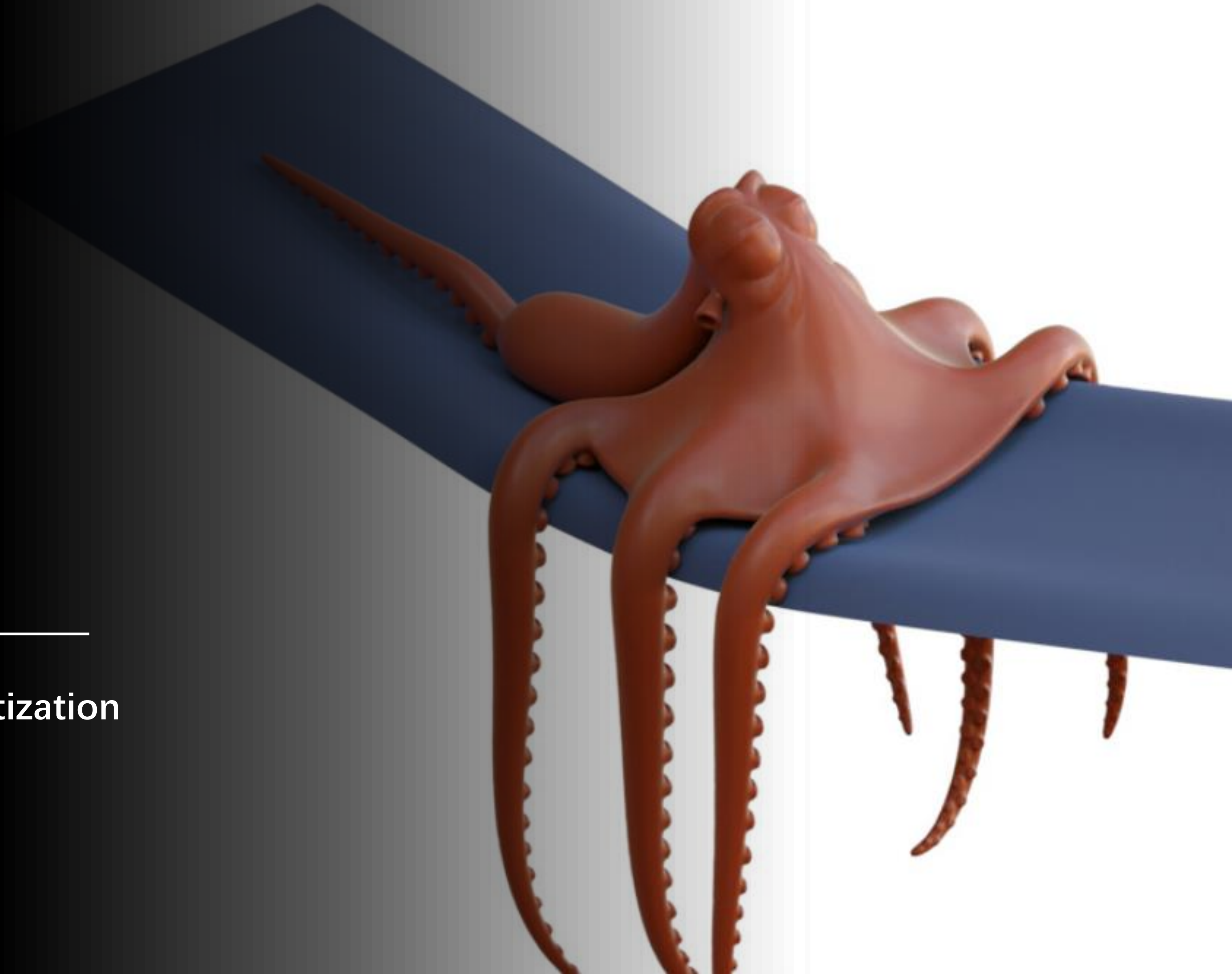
Deformable Simulation

Spatial and Temporal Discretization



Outline today

- The temporal integration
  - Explicit
  - Implicit
  - Symplectic
- The spatial integration
  - Mass-Spring system
  - The finite element method

Equations of motion (general cases)

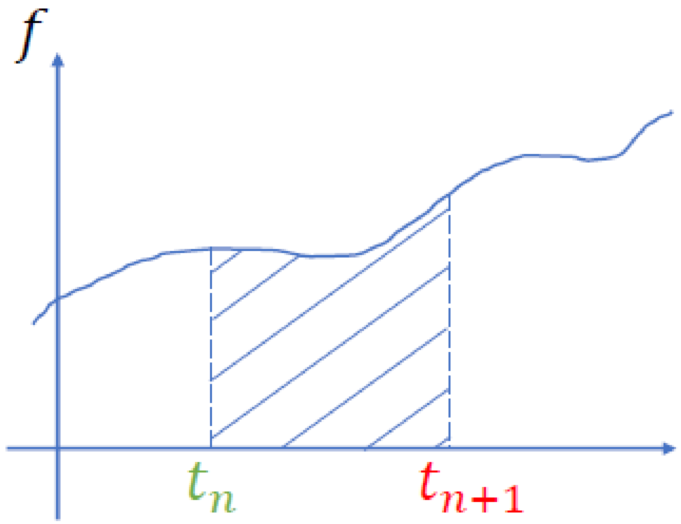
$$M\ddot{\boldsymbol{x}} = f(\boldsymbol{x})$$

$$\dot{\boldsymbol{x}} = \boldsymbol{v} \qquad \dot{\boldsymbol{v}} = \boldsymbol{a} = M^{-1}\boldsymbol{f}$$

$$\boldsymbol{x}(t_n + h) = \boldsymbol{x}(t_n) + \int_0^h \boldsymbol{v}(t_n + t)dt$$

$$\boldsymbol{v}(t_n + h) = \boldsymbol{v}(t_n) + \int_0^h M^{-1}f(t_n + t)dt$$

$h = t_{n+1} - t_n$ is the time-step size



We don't know how to integrate this quantity

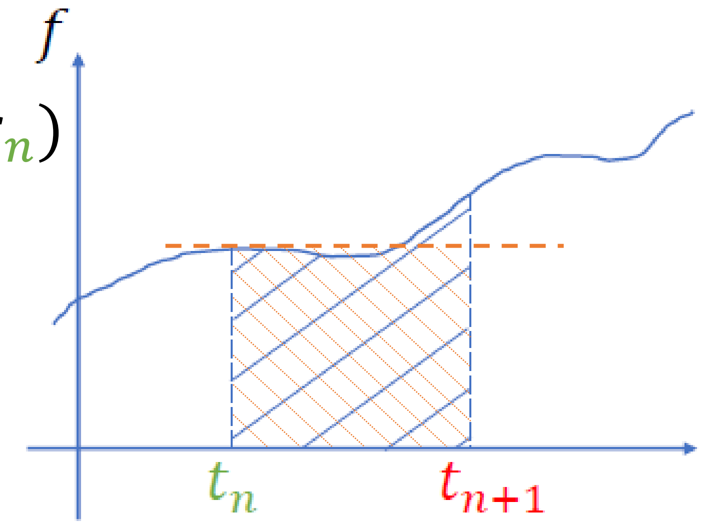
We don't know anything after t_n

Temporal integration (Explicit)

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}(t_n) + \int_0^h \boldsymbol{v}(t_n + t)dt \approx \boldsymbol{x}_n + h\boldsymbol{v}_n$$

$$\boldsymbol{v}_{n+1} = \boldsymbol{v}(t_n) + \int_0^h M^{-1} f(t_n + t)dt = \boldsymbol{v}_n + hM^{-1} f(\boldsymbol{x}_n)$$

Accuracy: 1st-order accurate



$$\int_0^h \boldsymbol{v}(t_n + t)dt = h\boldsymbol{v}(t_n + 0) + \frac{h^2}{2}\boldsymbol{v}'(t_n + 0) + \cdots$$

$$= h\boldsymbol{v}_n + O(h^2)$$

Temporal integration (Explicit)

Stability:

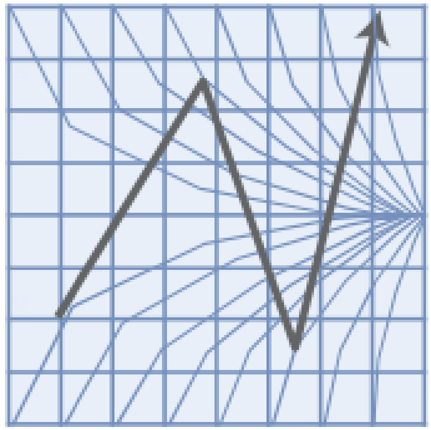
$$\frac{d}{dt}x = \lambda x$$

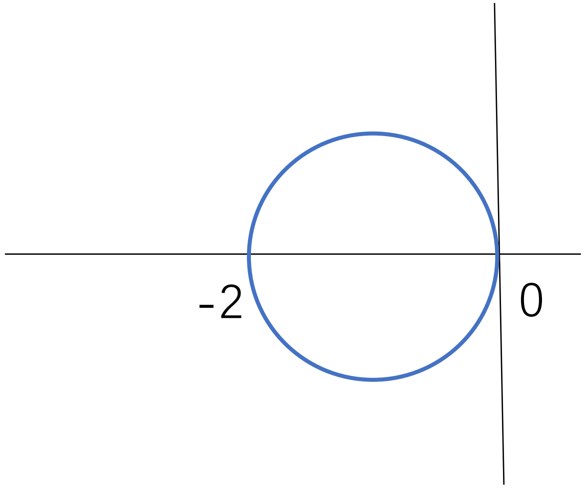
$$x_{n+1} = x_n + h\lambda x_n$$

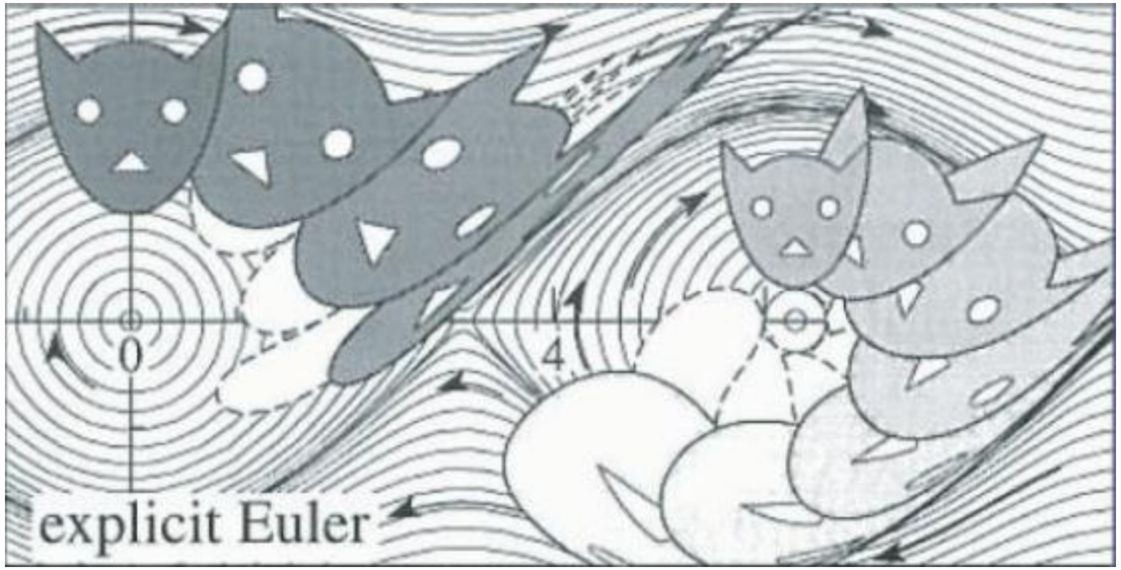
$$x_n = (1 + h\lambda)^n x_0$$

Condition of stability:

$$|1 + h\lambda| \leq 1, \;\; h \leq \frac{2}{|\lambda|}$$







Explicit Euler is **extremely fast**, but it will also increase the system energy gradually. **Not recommended in almost all situations.**

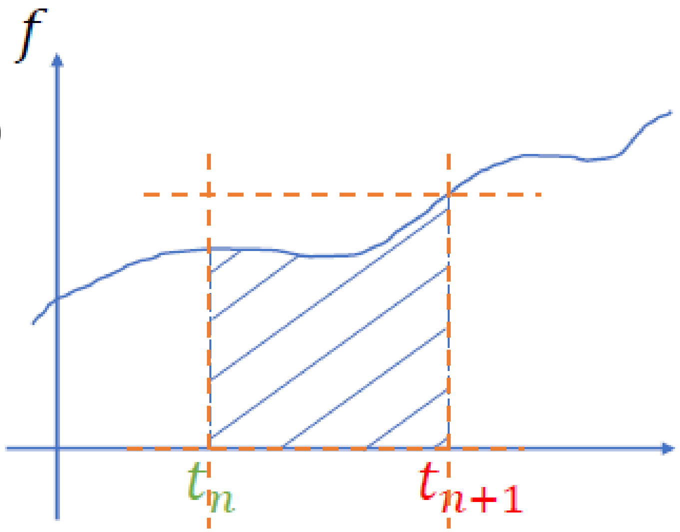
Temporal integration (Implicit)

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}(t_n) + \int_0^h \boldsymbol{v}(t_n + t)dt \approx \boldsymbol{x}_n + h\boldsymbol{v}_{n+1}$$

$$\boldsymbol{v}_{n+1} = \boldsymbol{v}(t_n) + \int_0^h M^{-1} f(t_n + t)dt \approx \boldsymbol{v}_n + hM^{-1} f(\boldsymbol{x}_{n+1})$$

$$g(\boldsymbol{x}) = \frac{1}{2h^2}(\boldsymbol{x}_{n+1} - \boldsymbol{y}_n)^T M(\boldsymbol{x}_{n+1} - \boldsymbol{y}_n) + E(\boldsymbol{x}_{n+1})$$

Accuracy: 1st-order accurate



$$\int_0^h \boldsymbol{v}(t_n + t)dt = h\boldsymbol{v}(t_n + h) - \frac{h^2}{2}\boldsymbol{v}'(t_n + h) - \cdots$$

$$= h\boldsymbol{v}_{n+1} + O(h^2)$$

Expand where $\Delta t = h$

Temporal integration (Implicit)

Stability:

$$\frac{d}{dt}x = \lambda x$$

$$x_{n+1} = x_n + h\lambda x_{n+1}$$

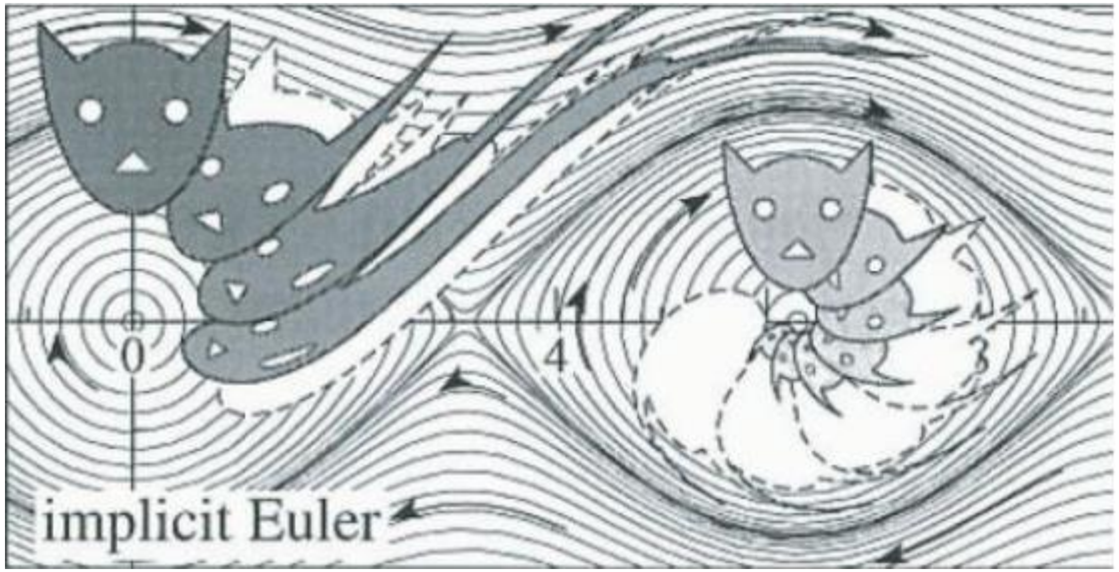
$$x_n = (1 - h\lambda)^{-n} x_0$$

Condition of stability:

$$|1 - h\lambda| \geq 1$$

Implicit Euler integrals are unconditionally stable!

Implicit Euler is often **expensive** due to the nonlinear optimization, it is often stable for **large time steps** and is widely used in performance centric applications. (game / design / animation)



Temporal integration (Symplectic)

$$\boldsymbol{v}_{n+1} = \boldsymbol{v}(t_n) + \int_0^h M^{-1} f(t_n + t)dt \approx \boldsymbol{v}_n + hM^{-1} f(\boldsymbol{x}_n)$$

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}(t_n) + \int_0^h \boldsymbol{v}(t_n + t)dt \approx \boldsymbol{x}_n + h\boldsymbol{v}_{n+1}$$

Accuracy: 2nd-order accurate

$$\boldsymbol{v}_{n+0.5} \approx \boldsymbol{v}_{n-0.5} + hM^{-1} f(\boldsymbol{x}_n)$$

$$\boldsymbol{x}_{n+1} \approx \boldsymbol{x}_n + h\boldsymbol{v}_{n+0.5}$$

Midpoint

$$\int_0^h \boldsymbol{v}(t_n + t)dt = \int_0^{h/2} \boldsymbol{v}(t_n + t)dt + \int_{h/2}^h \boldsymbol{v}(t_n + t)dt$$

$$= \frac{h}{2}\boldsymbol{v}(h/2) - \frac{h^2}{2}\boldsymbol{v}'\left(\frac{h}{2}\right) + O(h^3) +$$

$$\frac{h}{2}\boldsymbol{v}(h/2) + \frac{h^2}{2}\boldsymbol{v}'\left(\frac{h}{2}\right) + O(h^3) = h\boldsymbol{v}_{n+0.5} + \boxed{O(h^3)}$$

Temporal integration (Symplectic)

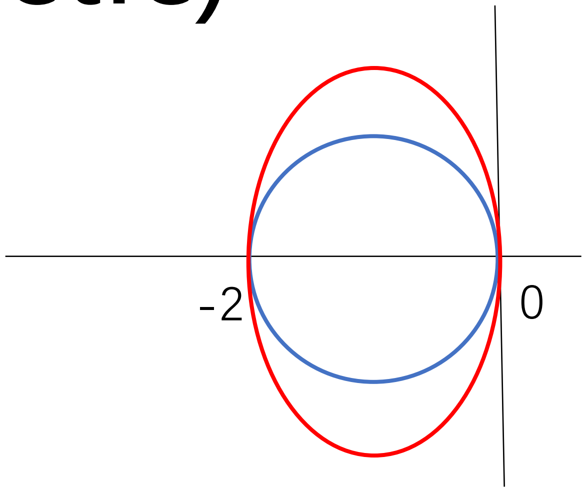
Stability:

$$\frac{d}{dt}x = \lambda x$$

$$x_{n+1} = x_n + h\lambda x_{n+0.5} = x_n + h\lambda(x_n + 0.5h\lambda x_n)$$

$$= x_n(1 + h\lambda + 0.5(h\lambda)^2)$$

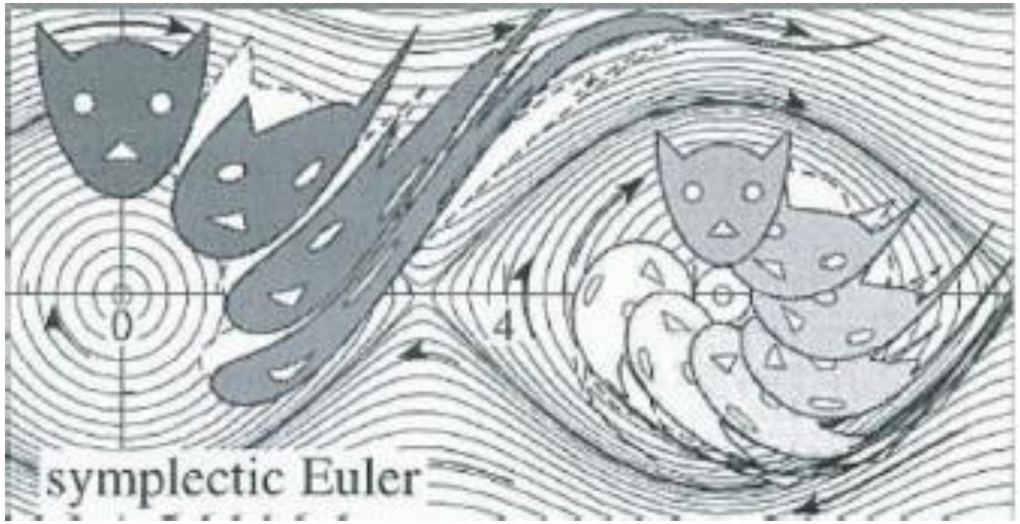
$$x_n = (1 + h\lambda + 0.5(h\lambda)^2)^n x_0$$



Condition of stability:

$$h\lambda = a + ib, \;\; |1 + h\lambda + 0.5(h\lambda)^2| \leq 1$$

$$\left\| \begin{bmatrix} 1 + a + \frac{a^2 - b^2}{2} \\ b + ab \end{bmatrix} \right\| \leq 1$$

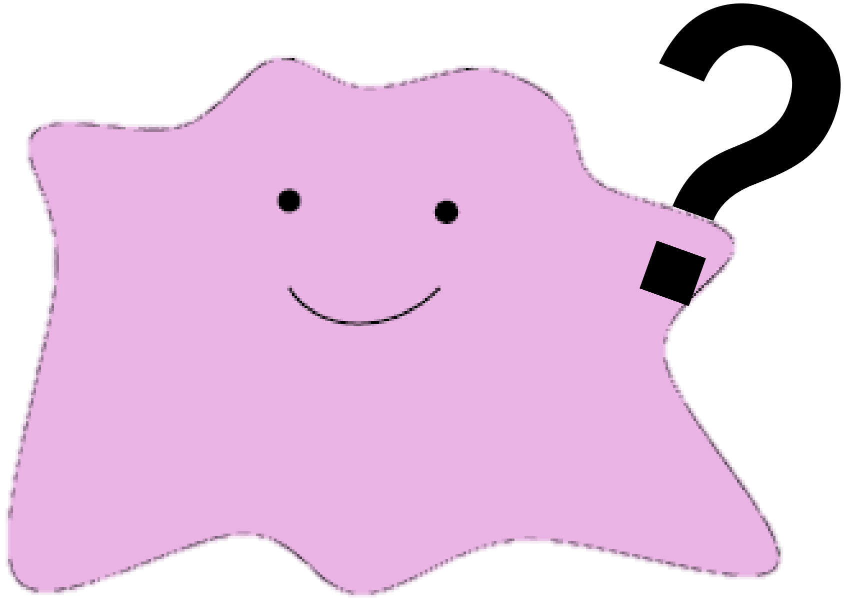


Symplectic Euler is as **fast** as forward Euler. It has been widely used in **accuracy centric applications** (astronomy simulation / molecular dynamics etc).

Spatial integration

- For conservative forces (as most of the elastic forces are), the energy definition is all we need for their simulations.
- A deformable object is a continuum body

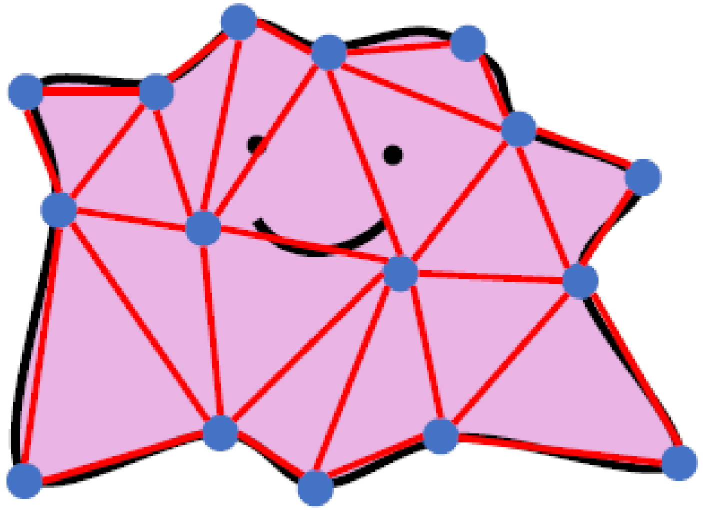
$$E = \int$$



How to describe the deformation?
How to describe the elastic energy?

Mass-spring system

- Aggregate the volume mass to the vertices
- Link the mass vertices with springs

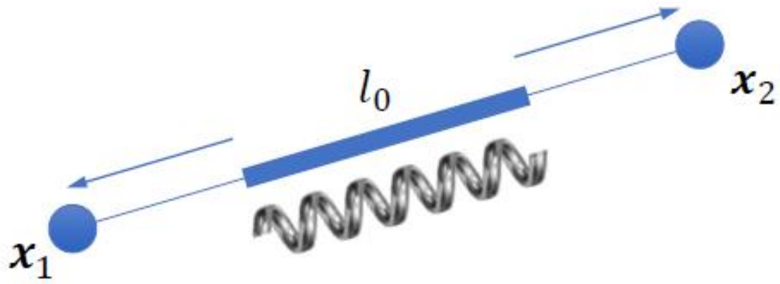


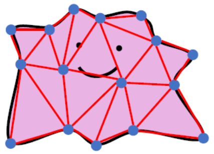
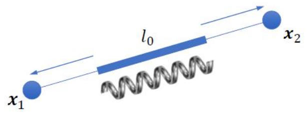
- How to describe the deformation?

  “Deformation” $= l - l_0$

- How to describe the elastic energy?

  Hooke’s Law: $E(\boldsymbol{x}_1, \boldsymbol{x}_2) = \frac{1}{2} k (l - l_0)^2$



$E = \int$  $= \sum$ 

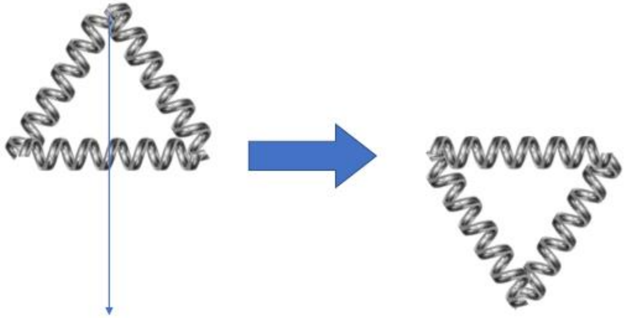
Mass-spring system

- Mass-spring systems are particularly useful in:

Cloth Sim
Dinev et al. 2018

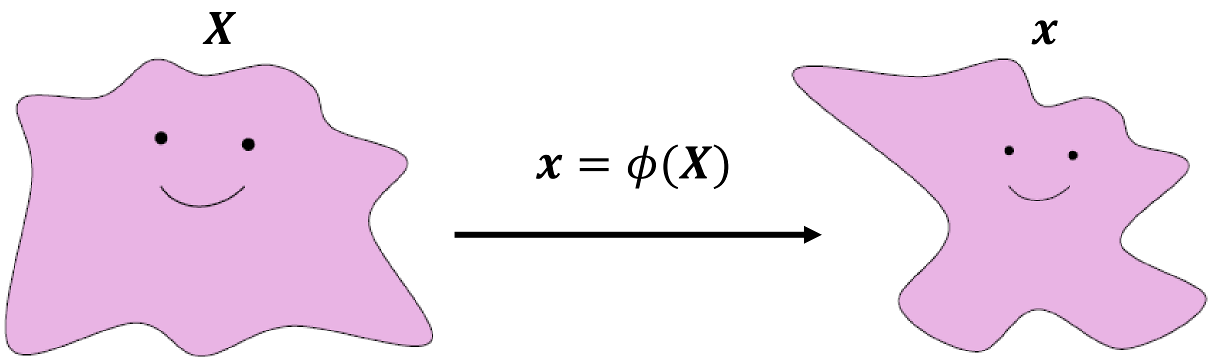
Hair Sim
Selle et al. 2018

- Mass-spring systems are NOT the best choices when simulating continuum area/volume



Linear finite element method (FEM)

- How to describe the deformation?



$$\phi(\boldsymbol{X}) \approx \underbrace{\frac{\partial \phi}{\partial \boldsymbol{X}}}_{F} \boldsymbol{X} + \underbrace{\left(\phi(\boldsymbol{X}^*) - \frac{\partial \phi}{\partial \boldsymbol{X}} \boldsymbol{X}^*\right)}_{t}$$

Deformation gradient

$$\boldsymbol{x} \approx F\boldsymbol{X} + t$$

- Translation: $\phi(\boldsymbol{X}) = \boldsymbol{X} + \boldsymbol{t}$

$$F = \frac{\partial \phi}{\partial \boldsymbol{X}} = I$$

- Rotation: $\phi(\boldsymbol{X}) = R\boldsymbol{X}$

$$F = \frac{\partial \phi}{\partial \boldsymbol{X}} = R$$

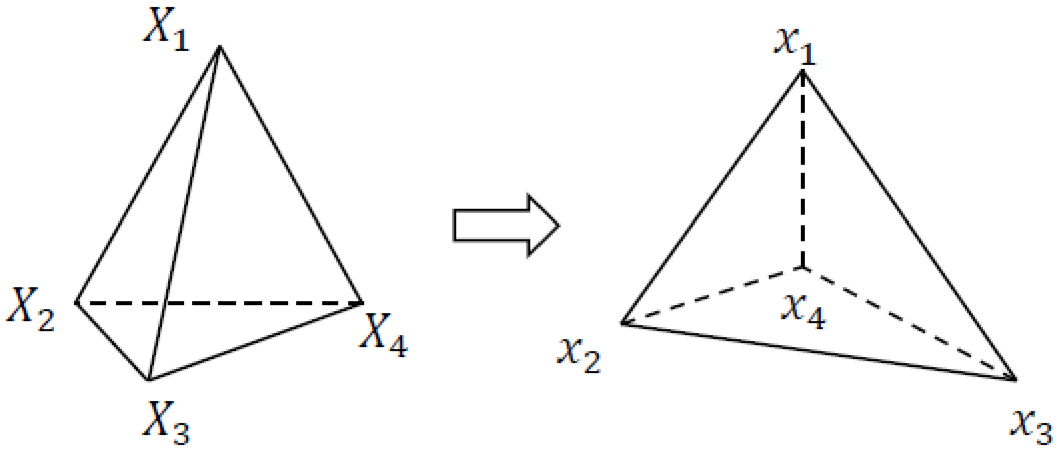
- Scaling: $\phi(\boldsymbol{X}) = S\boldsymbol{X}$

$$F = \frac{\partial \phi}{\partial \boldsymbol{X}} = S$$

$$R = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}, S = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$$

Linear finite element method (FEM)

- How to describe the elastic energy?
- Define: $\Psi(\boldsymbol{x}) = \Psi(F)$ is an energy density function at $\boldsymbol{x} = \phi(\boldsymbol{X})$, so $E = \sum w_{\mathrm{e}} \Psi(F_e)$
- Is it appropriate to do so?



$$\underbrace{[x_1 - x_4 \quad x_2 - x_4 \quad x_3 - x_4]}_{D_s} = F\underbrace{[X_1 - X_4 \quad X_2 - X_4 \quad X_3 - X_4]}_{D_m}$$

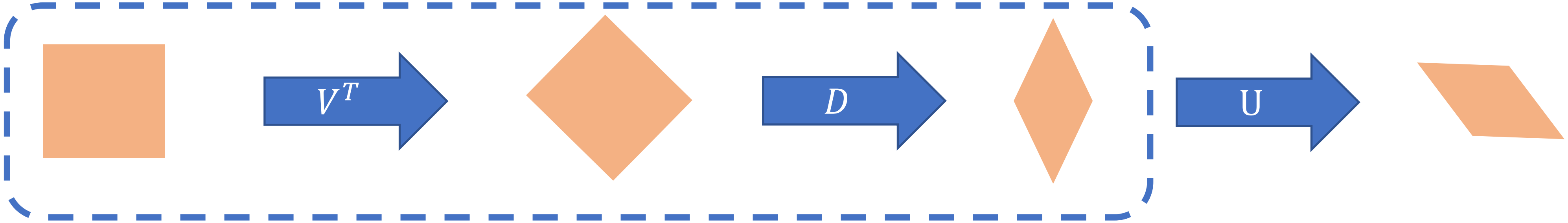
$$x_1 = FX_1 + t$$
$$x_2 = FX_2 + t$$
$$x_3 = FX_3 + t$$
$$x_4 = FX_4 + t$$

$$F = D_s D_m^{-1}$$

F is related to deformation,
But it contains **rotation**.

Linear finite element method (FEM)

Ideally, we need a tensor to describe shape deformation only. Recall that SVD gives $F = UDV^T$, where only V^T and D are relevant to deformation.



So we get rid of U as:

$$\epsilon_G(F) = \frac{1}{2}(F^T F - I) = \frac{1}{2}(V D^2 V^T - I)$$ Green strain

Strain: Descriptor of severity of deformation

- $\epsilon(I) = 0$
- $\epsilon(F) = \epsilon(RF)$

Linear finite element method (FEM)

- St. Venant-Kirchhoff model (StVK):
  - Strain: $\epsilon_{stvk}(F) = \frac{1}{2}(F^T F - I)$
  - Energy density: $\Psi(\epsilon_{stvk}) = \mu\|\epsilon_{stvk}(F)\|_F^2 + \frac{\lambda}{2} tr\left(\epsilon_{stvk}(F)\right)^2$
  - $P = \frac{\partial \Psi}{\partial F} = F[2\mu\epsilon_{stvk} + \lambda tr(\epsilon_{stvk})I]$ **1st-Piola-Kirchhoff stress tensor**

- Co-rotated linear model:
  - Strain: $\epsilon_c(F) = S - I$, where $F = RS$
  - Energy density: $\Psi(\epsilon_c) = \mu\|\epsilon_c\|_F^2 + \frac{\lambda}{2} tr(R^T F - I)^2$
  - $P = \frac{\partial \Psi}{\partial F} = R[2\mu\epsilon_c + \lambda tr(\epsilon_c)I]$

Eventually we will need the gradient of Ψ to run simulations.

$$\frac{\partial \Psi}{\partial \boldsymbol{x}} = \frac{\partial F}{\partial \boldsymbol{x}} : \boxed{\frac{\partial \Psi}{\partial F}}$$

A $(12 \times 1) \times (3 \times 3)$ tensor

A (3×3) tensor (matrix)

A (12×1) tensor (vector)

Note (matrix contraction): $B: A = A: B = \sum_{i,j} A_{ij} B_{ij} = tr(A^T B)$

Linear finite element method (FEM)

- Chain rule in detail: $\frac{\partial \Psi}{\partial x_j^i} = \frac{\partial F}{\partial x_j^i} : P$
  - Let's compute $\frac{\partial F}{\partial x_j^i}$ first:

  Since $F = D_s D_m^{-1}$, so $\frac{\partial F}{\partial x_j^i} = \frac{\partial D_s}{\partial x_j^i} D_m^{-1}$

  $\frac{\partial D_s}{\partial x_j^i} = \delta_i \delta_j^T$, for $j = 1,2,3$

  Thus: $\frac{\partial F}{\partial x_j^i} = \delta_i \delta_j^T D_m^{-1}$

  - $\frac{\partial F}{\partial x_j^i} : P = \delta_i \delta_j^T D_m^{-1} : P = tr\left(D_m^{-T} \delta_j \delta_i^T P\right)$

$$= tr\left(\delta_i^T P D_m^{-T} \delta_j\right) = \delta_i^T P D_m^{-T} \delta_j = [P D_m^{-T}]_{ij}$$

$$D_s = [x_1 - x_4 \quad x_2 - x_4 \quad x_3 - x_4] = \begin{bmatrix} x_1^1 - x_4^1 & x_2^1 - x_4^1 & x_3^1 - x_4^1 \\ x_1^2 - x_4^2 & x_2^2 - x_4^2 & x_3^2 - x_4^2 \\ x_1^3 - x_4^3 & x_2^3 - x_4^3 & x_3^3 - x_4^3 \end{bmatrix}$$

$$\frac{\partial D_s}{\partial x_1^2} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}^T = \delta_2 \delta_1^T$$

- The energy gradient: $\frac{\partial E_e}{\partial x_{e,j}} = w_e \frac{\partial \Psi_e}{\partial x_{e,j}}$ = the j-th col of $P D_m^{-T}$ for j=1, 2, 3 in the e-th tetrahedron
  - For j = 4: $\frac{\partial E_e}{\partial x_{e,4}} = w_e \frac{\partial \Psi_e}{\partial x_{e,4}} = -w_e \sum_{j=1}^{3} \frac{\partial \Psi_e}{\partial x_{e,j}}$

The sum of all internal forces in a tetrahedron should be **zero**

Note (matrix contraction): $B : A = A : B = \sum_{i,j} A_{ij} B_{ij} = tr(A^T B)$

Linear finite element method (FEM)

- Now we get the internal (elastic) force in a tetrahedron, while a particle often exists in multiple tetrahedra.
- At each time step, execute:
  - $\boldsymbol{f} \leftarrow \boldsymbol{0}$
  - For e-th tetrahedron:
    - $D_s = [x_1 - x_4 \quad x_2 - x_4 \quad x_3 - x_4]$
    - $F = D_s D_m$
    - $P \leftarrow P(F)$
    - $H \leftarrow -w_e P D_m^{-T}$， $H = [\boldsymbol{h}_1 \quad \boldsymbol{h}_2 \quad \boldsymbol{h}_3]$
    - $\boldsymbol{f}_1 \mathrel{+}= \boldsymbol{h}_1$， $\boldsymbol{f}_2 \mathrel{+}= \boldsymbol{h}_2$， $\boldsymbol{f}_3 \mathrel{+}= \boldsymbol{h}_3$
    - $\boldsymbol{f}_4 \mathrel{+}= -(\boldsymbol{h}_1 + \boldsymbol{h}_2 + \boldsymbol{h}_3)$
  - $\boldsymbol{v}_{new} = \boldsymbol{v} + hM^{-1}\boldsymbol{f}$
  - $\boldsymbol{x}_{new} = \boldsymbol{x} + h\boldsymbol{v}$

Reference and Futher Readings

- Finite Element Method, Part I [SIGGRAPH 2012 Course] [Link]
- Dynamic Deformables: Implementation and Production Practicalities [SIGGRAH 2020 Course] [Link]
- TaichiCourse 01: Lecture 08 [Link]
- GAMES Course 103 – Lecture 7: Other Constrained Methods and Linear Finite Element Method I [Link]

All relevant files (ppt, pdf, image, code, video):
https://github.com/LILKOTYO/Lab-Presentation/tree/master/20220725_Spatial_and_Temporal_Discretization