

La simulation physique pour l'animation d'objets 3D:

Déformation et animation réaliste d'objets 3D

Principe

1. Défini un système S_0 à $t=0$
positions, vitesses, forces
2. Modélise son évolution temporelle
par EDP (/PDE)

$$\mathcal{F} \left(S(t), \frac{\partial^n S}{\partial t^n}, \frac{\partial^m S}{\partial x^m} \right) = 0$$

Généralement, on aura:
$$\frac{\partial S}{\partial t} = A S + \mathcal{H} \left(S(t), \frac{\partial^m S}{\partial x^m} \right)$$

3. On résout numériquement en avançant temporellement suivant Δt

Modèle

Différents niveaux d'approximations

1. **Méthodes particulières** (particle system)

On réduit tout à un point (pas de rotation+EDO)

+ simple

- fidèle

2. **Mécanique du solide** (rigid bodies)

Paramètres identiques pour tout le solide

+ inertie

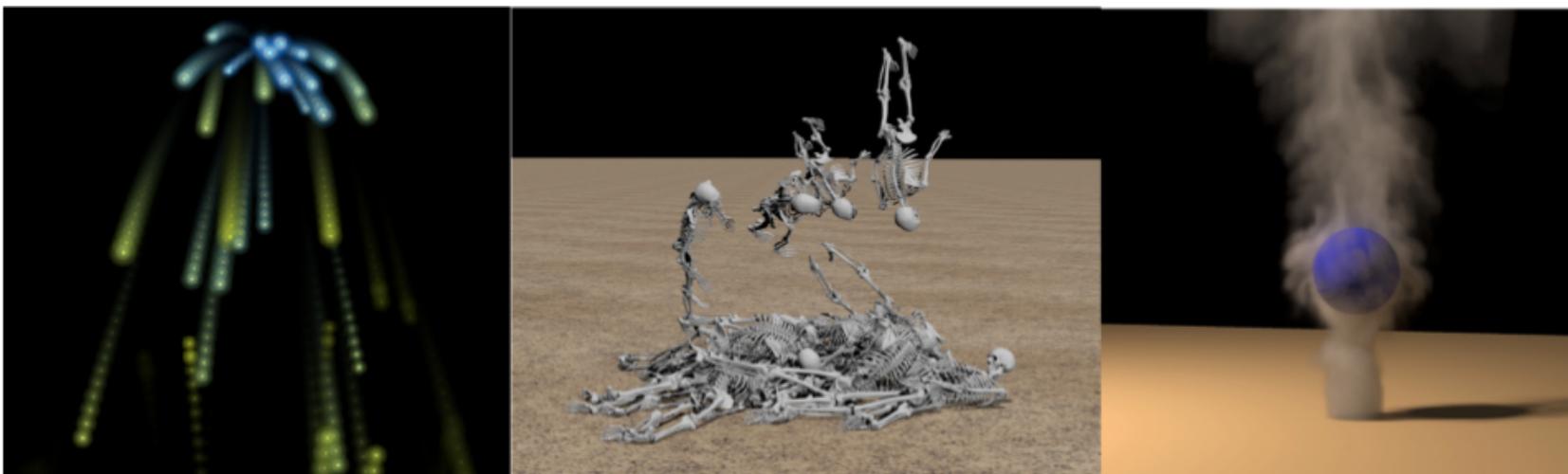
- collisions

3. **Mécanique des milieux continus** (continuum mechanics)

Paramètres variables dans le forme: EDP (FEM, FV, ...)

+ fidèle

- lourd



Modèle

Différents niveaux d'approximations

1. Méthodes particulières

$$ma = \sum_i f_i \Rightarrow x_i'' = F(x, x', t)$$

2. Mécanique du solide

$x(t)$ positions

$R(t)$ orientation

$\omega(t)$ vitesse angulaire

$L(t)$ moment angulaire

$\tau(t)$ torque

$$\frac{d}{dt} \begin{pmatrix} x(t) \\ R(t) \\ P(t) \\ L(t) \end{pmatrix} = \begin{pmatrix} v(t) \\ \omega(t)^* R(t) \\ F(t) \\ \tau(t) \end{pmatrix}$$

3. Mécanique des milieux continus

$$\nabla \cdot \overset{\text{stress}}{\sigma} + \mathbf{F}_{\text{ext}} = \eta \frac{d^2 \overset{\text{deformation}}{\mathbf{u}}}{dt^2}$$

$$\sigma = E \epsilon \quad \text{strain}$$

$$\epsilon = \frac{1}{2} (\nabla \mathbf{u} + [\nabla \mathbf{u}]^T + [\nabla \mathbf{u}]^T \nabla \mathbf{u})$$

Influence du type d'intégration numérique

Chute libre

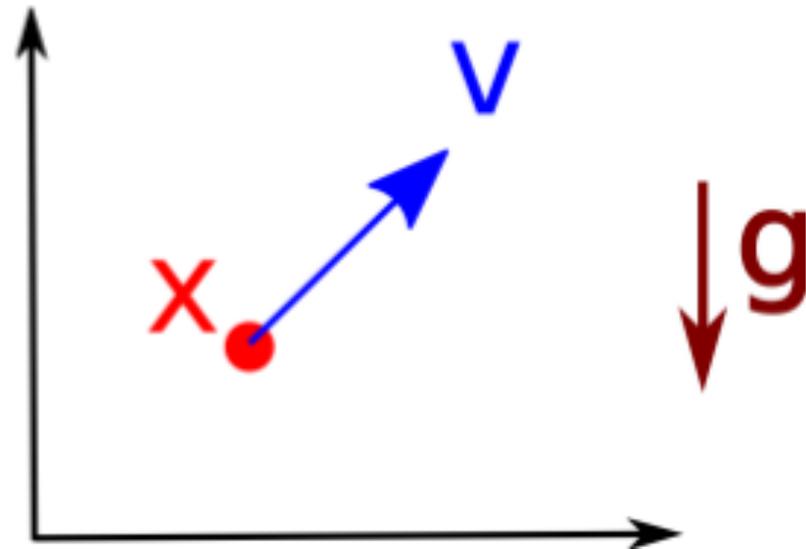
1-Description du système :

- Particule de masse m tombant en chute libre. Systeme décrit par $u = (x, v)$.
 - Position initiale $x(t = 0) = x_0$.
 - Vitesse initiale $v(t = 0) = v_0$.

2-Equation différentielle :

- Équation de la dynamique :
 $ma = \sum F = mg$

$$\Rightarrow \begin{cases} x'(t) = v(t) \\ v'(t) = g \end{cases}$$



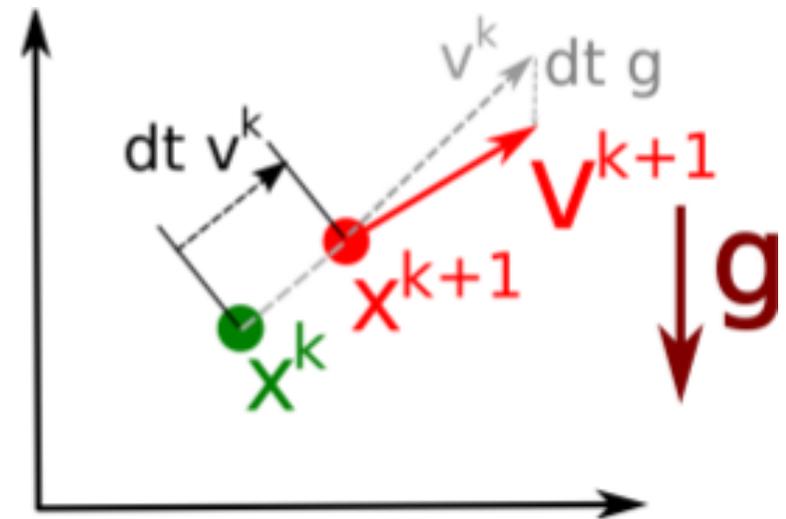
Chute libre

Résolution numérique directe :

$$\begin{cases} v^{k+1} = v^k + (\Delta t)g \\ x^{k+1} = x^k + (\Delta t)v^k \end{cases}$$

Implémentation triviale

```
x=x0;  
v=v0  
for (k=0; k<N; ++k)  
{  
    x=x+dt*v;  
    v=v+dt*g;  
}
```



Quelle erreur commet-on?

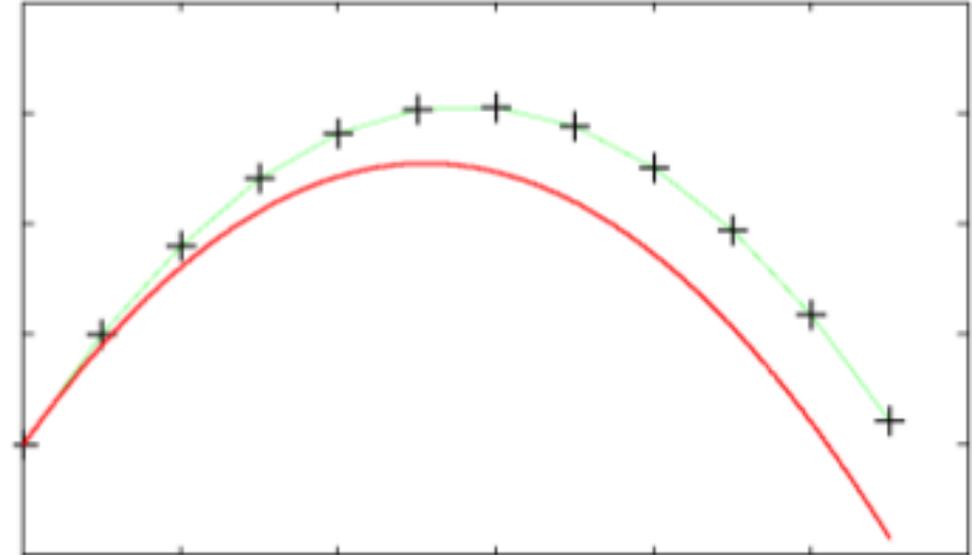
Chute libre

Equation numérique :

$$\begin{cases} v^{k+1} = v^k + (\Delta t)g \\ x^{k+1} = x^k + (\Delta t)v^k \end{cases}$$

Réccursion sur x^k :

$$\Rightarrow \begin{cases} x^{k+2} = 2x^{k+1} - x^k + (\Delta t)^2 g \\ x^0 = x_0 \text{ et } x^1 = x_0 + \Delta t v_0 \end{cases}$$



Par récurrence :

$$\Rightarrow x(t = k\Delta t) = x_0 + (k\Delta t) v_0 + \frac{k(k-1)}{2} (\Delta t)^2 g$$

Or solution exacte de $\frac{\partial x}{\partial t}(t) = g$: $\tilde{x}(t) = 1/2gt^2 + v_0 t + x_0$.

$$\tilde{x}(t = k\Delta t) = x_0 + (k\Delta t)v_0 + \frac{1}{2}(k\Delta t)^2 g$$

Précision

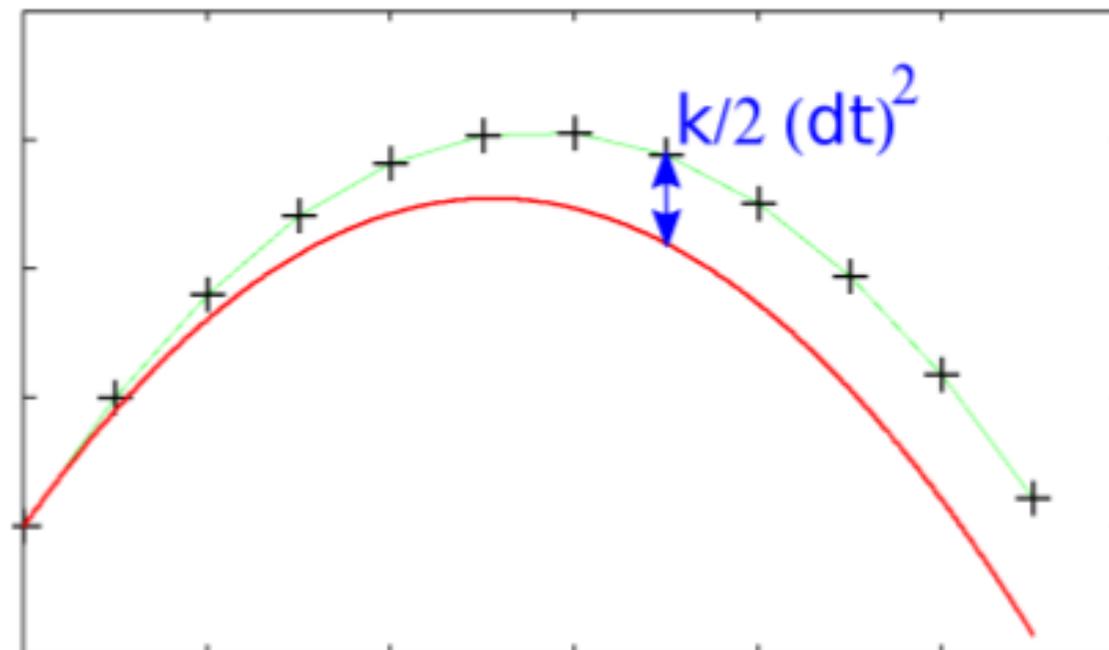
- Erreur :

$$\|x(k\Delta t) - \tilde{x}(k\Delta t)\| = \frac{k}{2}(\Delta t)^2$$

- Précision (accuracy) d'une méthode d'intégration d'ordre h :

$$\|x(k\Delta t) - \tilde{x}(k\Delta t)\| = \mathcal{O}((\Delta t)^{h+1})$$

- Notre schéma d'intégration est d'ordre 1. On ne résout pas l'équation continue !



Approche matricielle

Approche matricielle :

- On ne traite qu'avec une équation d'ordre 1 en posant $u = (x, v)$:

$$\frac{\partial u}{\partial t}(t) = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} u(t) + \begin{pmatrix} 0 \\ g \end{pmatrix}$$

$$\Leftrightarrow u_{,t} = Au + b$$

- 2 Possibilités de discrétisation temporelle :
 - $(u^{k+1} - u^k)/\Delta t = Au^k + b$: Euler explicite
 - $(u^{k+1} - u^k)/\Delta t = Au^{k+1} + b$: Euler implicite

Resolution explicite

- Euler explicite :

$$u^{k+1} = (I + \Delta t A)u^k + \Delta t b$$

$$\begin{pmatrix} x \\ v \end{pmatrix}^{k+1} = \begin{pmatrix} 1 & \Delta t \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ v \end{pmatrix}^k + \Delta t \begin{pmatrix} 0 \\ g \end{pmatrix}$$

Même résultats que précédemment

$$\Rightarrow \begin{cases} x^{k+2} = 2x^{k+1} - x^k + (\Delta t)^2 g \\ x^0 = x_0 \text{ et } x^1 = x_0 + \Delta t v_0 \end{cases}$$

$$\Rightarrow x(k\Delta t) = x_0 + (k\Delta t) v_0 + \frac{k(k-1)}{2} (\Delta t)^2 g$$

Resolution implicite

■ Euler implicite :

$$u^{k+1} = (I - \Delta t A)^{-1} (u^k + \Delta t b)$$

$$\begin{pmatrix} x \\ v \end{pmatrix}^{k+1} = \begin{pmatrix} 1 & -\Delta t \\ 0 & 1 \end{pmatrix}^{-1} \left[\begin{pmatrix} x \\ v \end{pmatrix}^k + \Delta t \begin{pmatrix} 0 \\ g \end{pmatrix} \right]$$

$$\Rightarrow \begin{cases} x^{k+2} = 2x^{k+1} - x^k + (\Delta t)^2 g \\ x^0 = x_0 \text{ et } x^1 = x_0 + \Delta t v_0 + (\Delta t)^2 g \end{cases}$$

$$\Rightarrow x(k\Delta t) = x_0 + (k\Delta t) v_0 + \frac{k(k+1)}{2} (\Delta t)^2 g$$

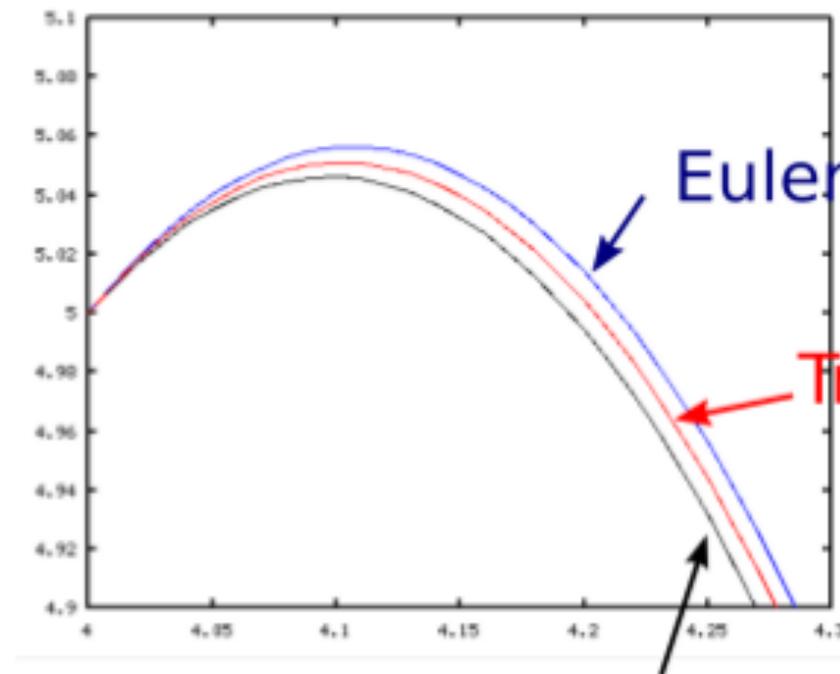
On ne résout toujours pas la bonne équation, et précision d'ordre 1 :
 $\|\tilde{x}(k\Delta t) - x(k\Delta t)\| = \frac{k}{2} (\Delta t)^2.$

Comparaison

- Euler explicite : $u^{k+1} = u^k + \Delta t F^k$
- Euler implicite : $u^{k+1} = u^k + \Delta t F^{k+1}$

Remarque :

- Euler explicite sur-estime \tilde{x} .
- Euler implicite sous-estime \tilde{x} .



Euler implicite

Monter en degré?

Derivée seconde incalculable en pratique

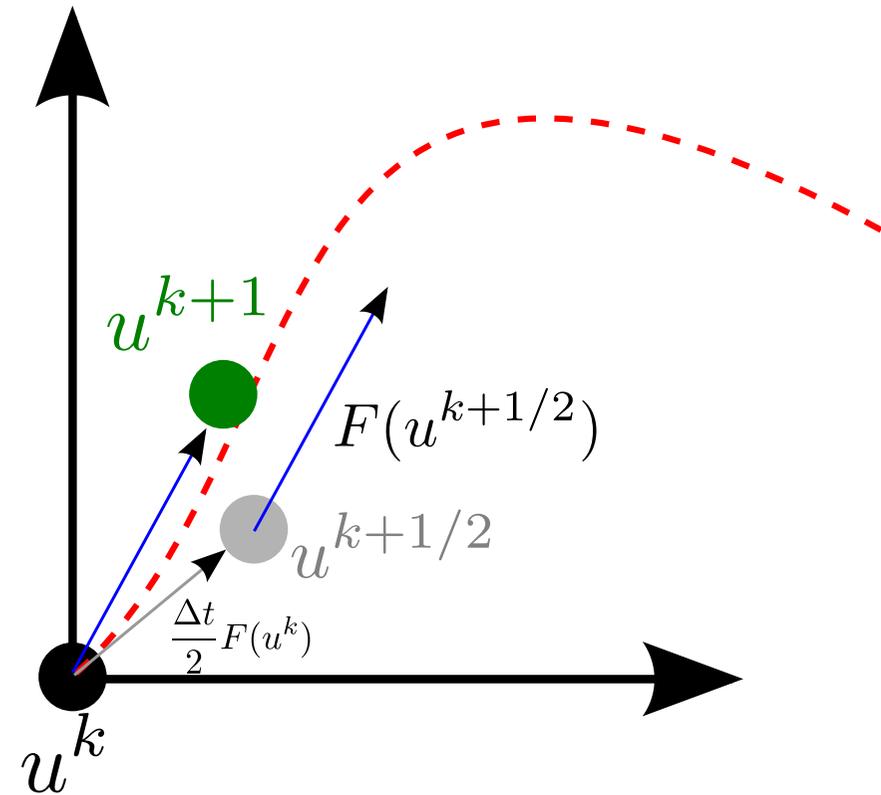
$$\frac{u(t + \Delta t) - u(t)}{\Delta t} = \sum_{k=1}^p \frac{1}{k!} (\Delta t)^{k-1} u^{(k)}(t) + \frac{1}{p!} (\Delta t)^p \int_0^1 (1 - \theta)^p u^{(p+1)}(t + \theta \Delta t) d\theta$$

$$u^{(k)}(t) = f^{[k-1]}(t, u(t))$$

Runge Kutta

$$u^{k+1/2} = u^k + \frac{\Delta t}{2} F(u^k)$$

$$u^{k+1} = u^k + \Delta t F(u^{k+1/2})$$



Runge Kutta

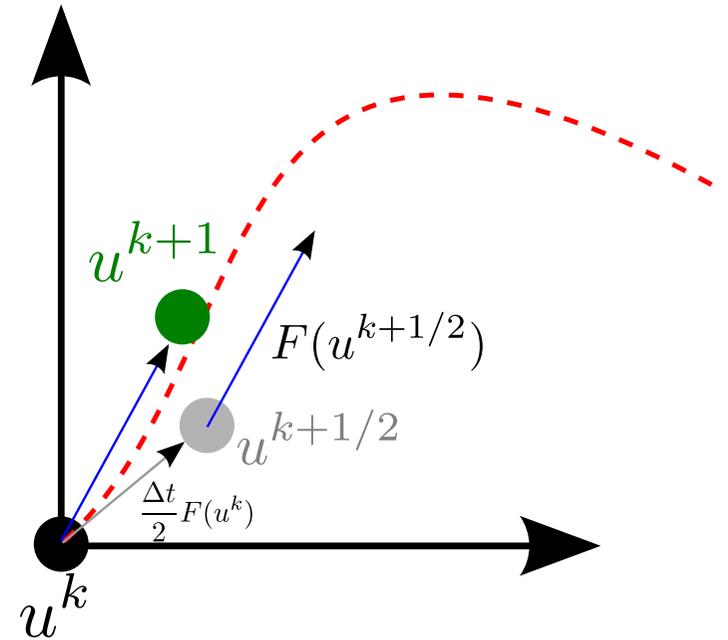
$$u^{k+1/2} = u^k + \frac{\Delta t}{2} F(u^k)$$
$$u^{k+1} = u^k + \Delta t F(u^{k+1/2})$$

Application à la chute libre:

$$F : u^k = (x^k, v^k) \mapsto \begin{pmatrix} v^k \\ g \end{pmatrix}$$

$$u^{k+1/2} = \begin{pmatrix} x^k + \Delta t/2 v^k \\ v^k + \Delta t/2 g \end{pmatrix}$$

$$F(u^{k+1/2}) = \begin{pmatrix} v^k + \frac{\Delta t}{2} g \\ g \end{pmatrix}$$



Runge Kutta

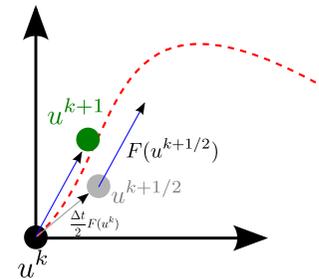
$$u^{k+1/2} = u^k + \frac{\Delta t}{2} F(u^k)$$

$$u^{k+1} = u^k + \Delta t F(u^{k+1/2})$$

$$F : u^k = (x^k, v^k) \mapsto \begin{pmatrix} v^k \\ g \end{pmatrix}$$

$$u^{k+1/2} = \begin{pmatrix} x^k + \Delta t/2 v^k \\ v^k + \Delta t/2 g \end{pmatrix}$$

$$F(u^{k+1/2}) = \begin{pmatrix} v^k + \frac{\Delta t}{2} g \\ g \end{pmatrix}$$



Application à la chute libre:

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

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

$$\begin{cases} x^{k+2} = 2x^{k+1} - x^k + (\Delta t)^2 g \\ x^0 = x_0 \\ x^1 = x_0 + \Delta t v_0 + \frac{(\Delta t)^2}{2} g \end{cases}$$

\Rightarrow

$$x^k = x_0 + (k \Delta t) v_0 + \frac{(k \Delta t)^2}{2} g$$

Runge Kutta

Ordre 4: Dormand-Prince (ode45)

Estimation erreur ordre 5

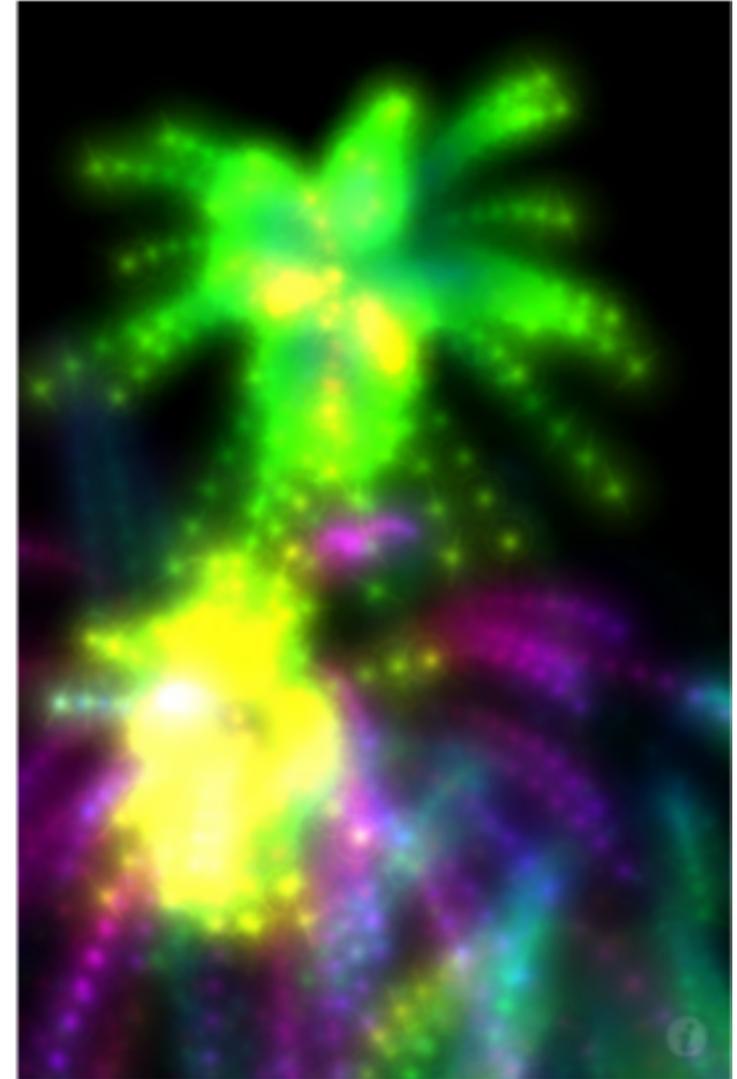
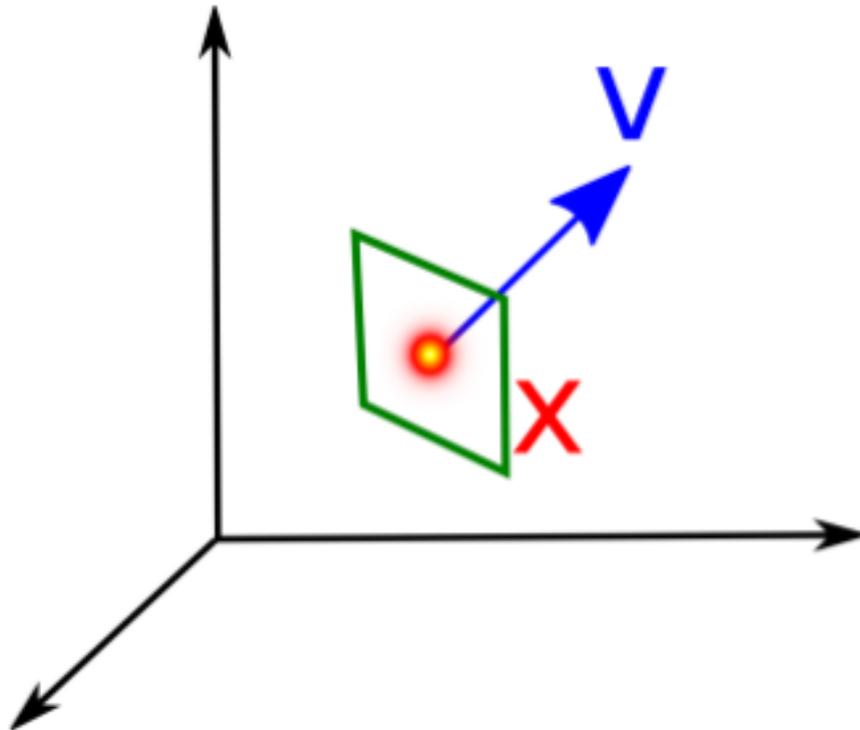
$$\begin{aligned}
 k_1 &= F(t_n, u_n) \\
 k_2 &= F\left(t_n + \frac{1}{5}\Delta t, u_n + \frac{1}{5}\Delta t k_1\right) \\
 k_3 &= F\left(t_n + \frac{3}{10}\Delta t, u_n + \left(\frac{3}{40}k_1 + \frac{9}{40}k_2\right)\Delta t\right) \\
 k_4 &= F\left(t_n + \frac{3}{5}\Delta t, u_n + \left(\frac{3}{10}k_1 - \frac{9}{10}k_2 + \frac{6}{5}k_3\right)\Delta t\right) \\
 k_5 &= F\left(t_n + \Delta t, u_n + \left(-\frac{11}{54}k_1 + \frac{5}{2}k_2 - \frac{70}{27}k_3 + \frac{35}{27}k_4\right)\Delta t\right) \\
 k_6 &= F\left(t_n + \frac{7}{8}\Delta t, \right. \\
 &\quad \left. u_n + \left(\frac{1631}{55296}k_1 + \frac{175}{512}k_2 - \frac{575}{13824}k_3 + \frac{44275}{110592}k_4 + \frac{253}{4096}k_5\right)\Delta t\right)
 \end{aligned}$$

$$\begin{aligned}
 u_{n+1}^4 &= u_n + \Delta t \left(\frac{2825}{27648}k_1 + \frac{18575}{48384}k_3 + \frac{13525}{55296}k_4 + \frac{277}{14336}k_5 + \frac{1}{4}k_6 \right) \\
 u_{n+1}^5 &= u_n + \Delta t \left(\frac{37}{378}k_1 + \frac{250}{621}k_3 + \frac{125}{594}k_4 + \frac{512}{1771}k_6 \right)
 \end{aligned}$$

Applications

Sprites=

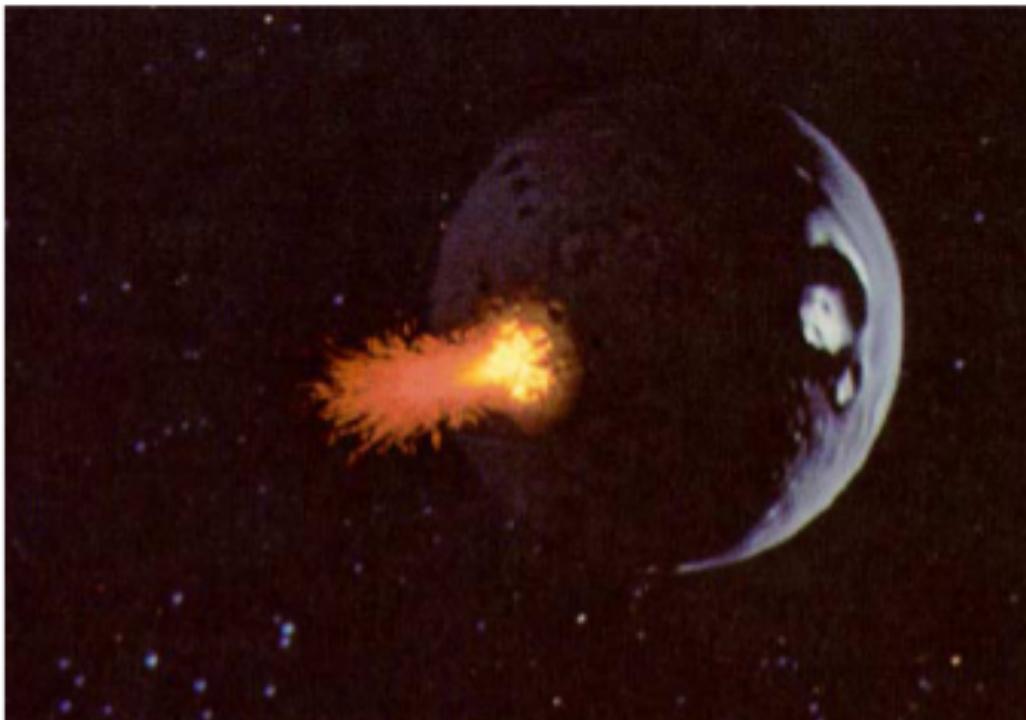
- Particule tombant sous gravité
- Durée de vie limitée
- Plaque une texture transparente animée



Applications

- On peut afficher la trajectoire

[William T. Reeves. **Particle Systems. A Technique for Modeling a Class of Fuzzy Objects.** *ACM Transaction on Graphics*, 17(3). 1983]



Masses ressorts

- Ressort 1D : $F(t) = K(L_0 - x(t))$
- Equation : $x''(t) = K/m(L_0 - x(t))$

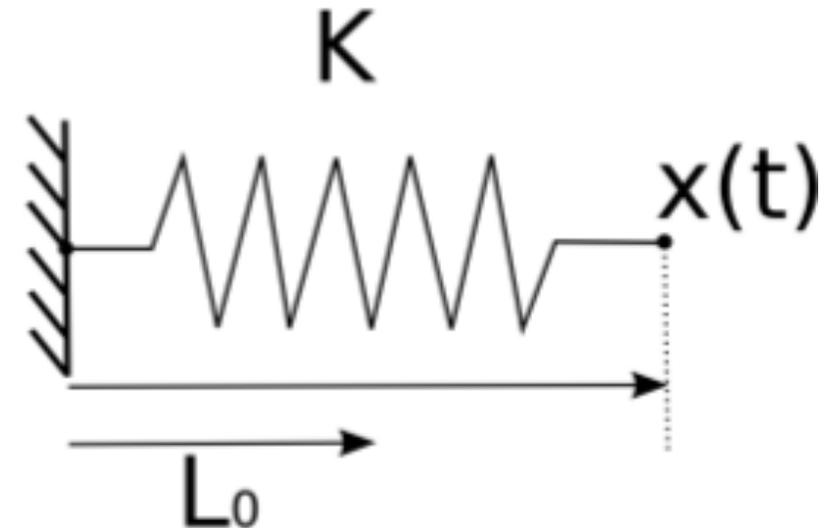
$$u' = \begin{pmatrix} 0 & 1 \\ -K/m & 0 \end{pmatrix} u + \begin{pmatrix} 0 \\ kL_0/m \end{pmatrix}$$

- Vraie solution : oscillations
- Euler explicite :

$$x^{k+2} = 2x^{k+1} - \left(1 + (\Delta t)^2 \frac{K}{m}\right) x^k + (\Delta t)^2 \frac{K}{m} L_0$$

Stable si $(1 + \sqrt{(\Delta t)^2 \frac{K}{m}}) < 1 \Rightarrow$ Euler explicite diverge forcément !

- Euler implicite converge vers 0.



Masses ressorts

Ajout d'un terme de frottement fluide : $F(t) = -\mu v(t)$

$$u' = \begin{pmatrix} 0 & 1 \\ -K/m & -\mu/m \end{pmatrix} u + \begin{pmatrix} 0 \\ kL_0/m \end{pmatrix}$$

Nouvelle récurrence pour Euler explicite :

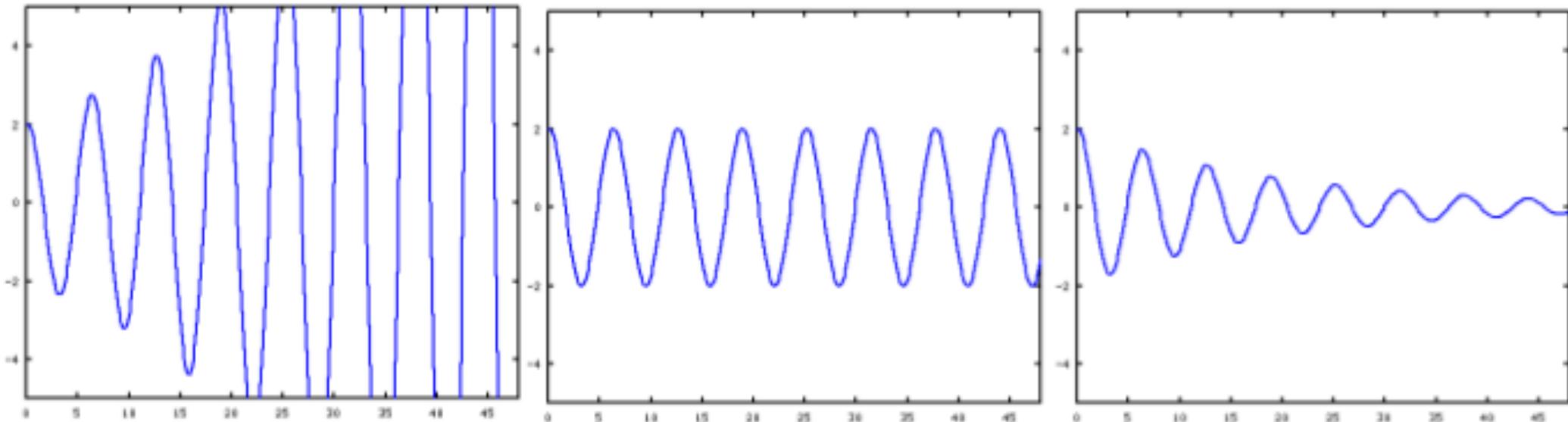
$$x_{k+2} - \left(2 - \frac{\mu}{m} \Delta t\right) x^{k+1} + \left(1 + (\Delta t)^2 \frac{K}{m} - \frac{\mu}{m} \Delta t\right) x^k = (\Delta t)^2 \frac{K}{m} L_0$$

Conditionnellement stable

Ressort raides: K grand \Rightarrow stiff

Precision/Stabilité

! Précision \neq Stabilité



- Schémas explicite \Rightarrow petits pas de temps / rapide.
- Schémas implicites \Rightarrow grands pas de temps / inversion système.

Euler implicite

$$\begin{pmatrix} 1 & -\Delta t \\ -\Delta t \frac{K}{m} & 1 \end{pmatrix} u^{k+1} = u^k + \Delta t \begin{pmatrix} 0 \\ K \frac{L_0}{m} \end{pmatrix}$$

Valeur propres M^{-1}

$$1 - \Delta t \sqrt{\frac{K}{m}} < 1$$

Inconditionnellement stable

Euler implicite

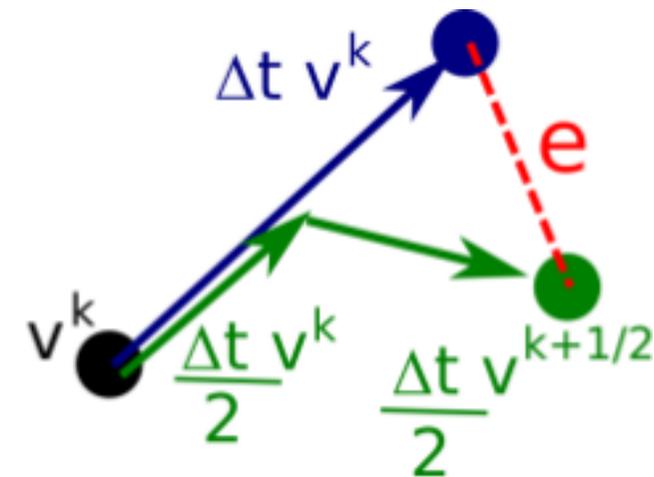
■ Calcul automatique du pas de temps Δt :

- 1 Calcul $u_1^{n+1} = F(\Delta t, u^n)$
- 2 Calcul $u_2^{n+1/2} = F(\frac{\Delta t}{2}, u^n)$;
 $u_2^{n+1} = F(\frac{\Delta t}{2}, u_2^{n+1/2})$
- 3 $e = \|u_1^{n+1} - u_2^{n+1}\|$

- $\Rightarrow e > \text{threshold}_1 : (\Delta t)_{\text{new}} = \frac{\Delta t}{2}$
 $\Rightarrow e < \text{threshold}_2 : (\Delta t)_{\text{new}} = 2\Delta t$
- 3 évaluations

■ Avec des ordres différents (Dormant-Prince !)

- $e = \|u_{\text{order}h} - u_{\text{order}h+1}\|$
- 2 évaluations.



Faire

```
u1=F (dt, u)
u2=F (dt/2, u) ;
u2=F (dt/2, u2) ;
e=norm(u1-u2)
si e<E_min
    dt*=2;
si e>E_max
    dt/=2;
```

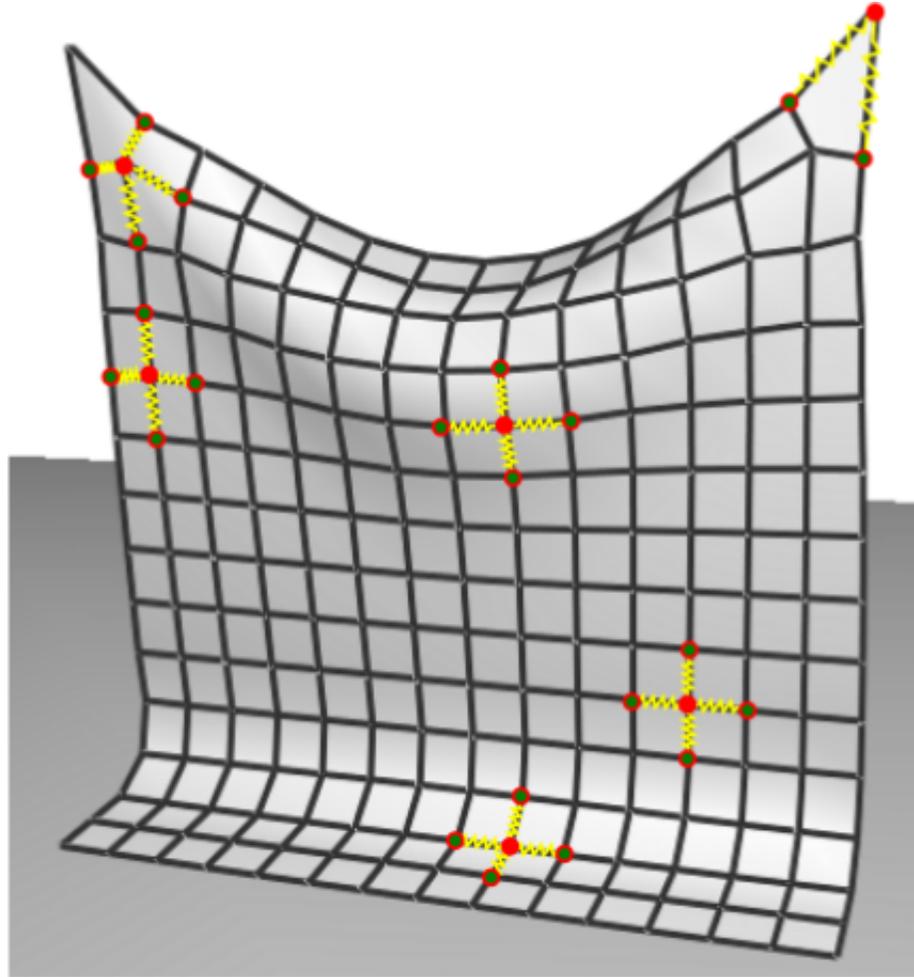
tant que

```
!(E_min<e<E_max) 025
```

Simulation de tissus

- 1/ Principes
- 2/ Intégration
- 3/ Tissus

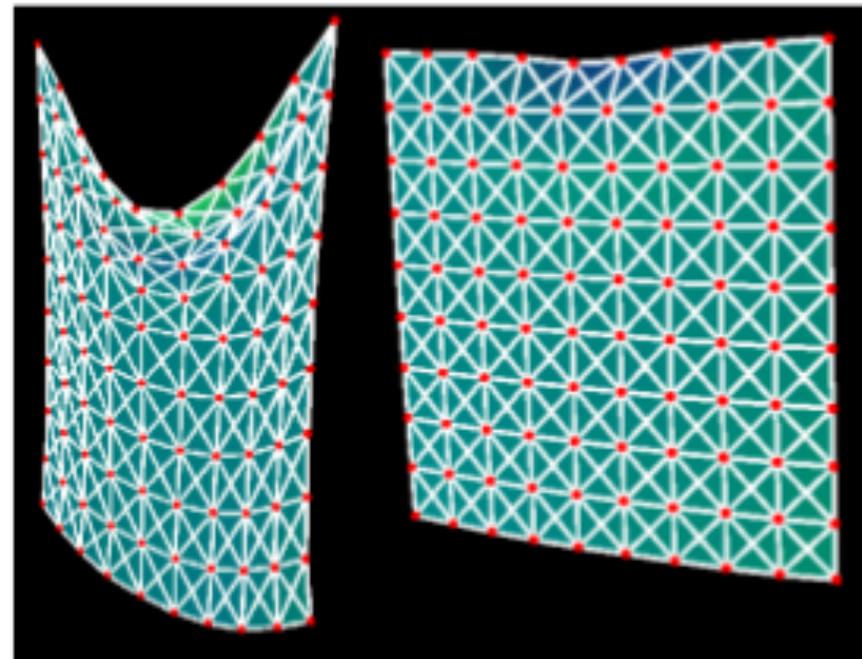
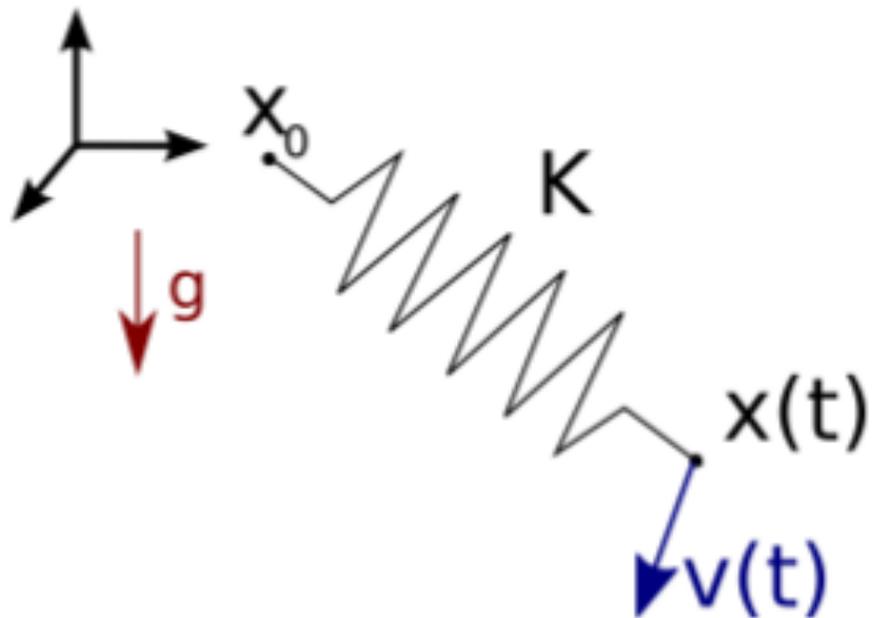
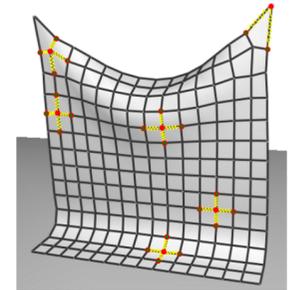
Simulation de tissus



Simulation de tissus

■ En 3D :

$$F(t) = K (L_0 - \|x - x_0\|) \frac{x - x_0}{\|x - x_0\|}$$



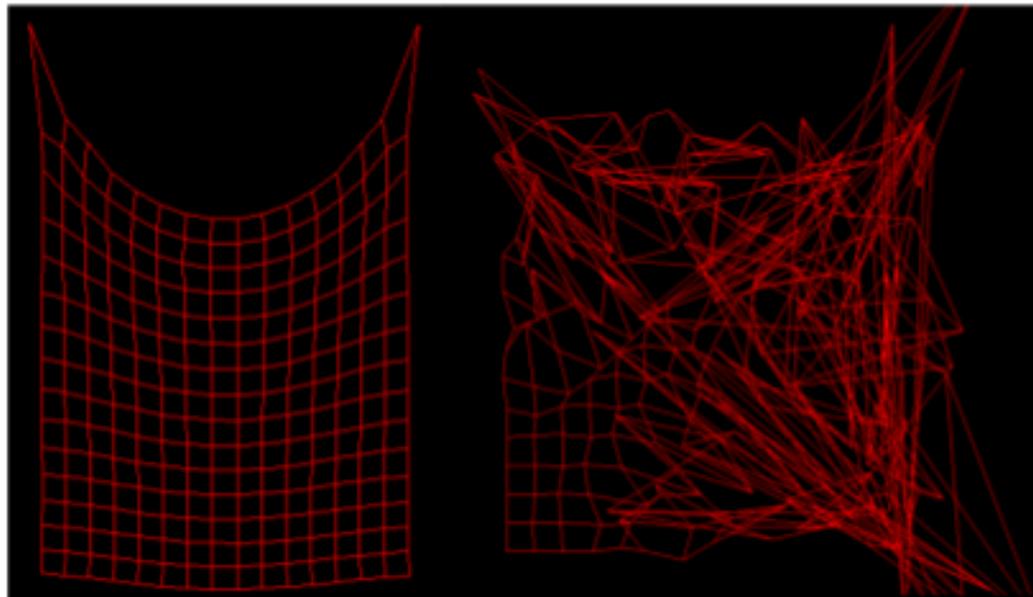
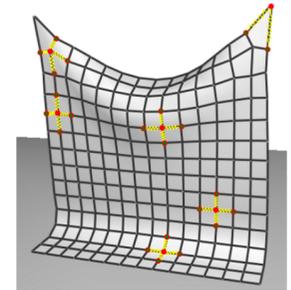
Simulation de tissus

- Modélisation d'un tissu par des ressorts couplés.
- Force pour un sommet i de voisins \mathcal{V}_i

$$F(x_i, t) = \sum_{j \in \mathcal{V}_i} K^{ij} \left(L_0^{ij} - \|x_i - x_j\| \right) \frac{x_i - x_j}{\|x_i - x_j\|} + g$$

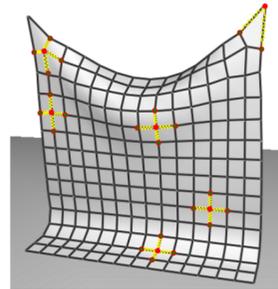
$$\forall i, \begin{cases} x_i'(t) = v_i(t) \\ v_i'(t) = \frac{1}{m_i} \sum_j K^{ij} \left(L_0^{ij} - \|x_i - x_j\| \right) \frac{x_i - x_j}{\|x_i - x_j\|} + g \end{cases}$$

⇒ Schéma d'intégration favori

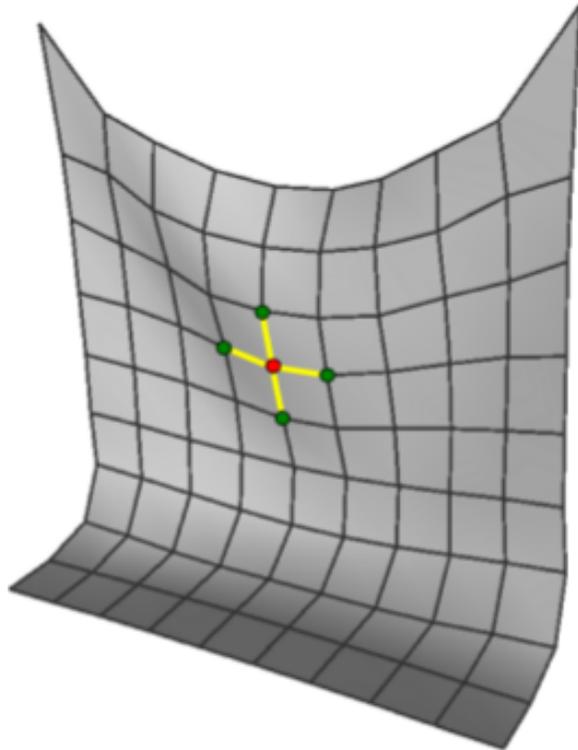


- 1/ Principes
- 2/ Intégration
- 3/ Tissus

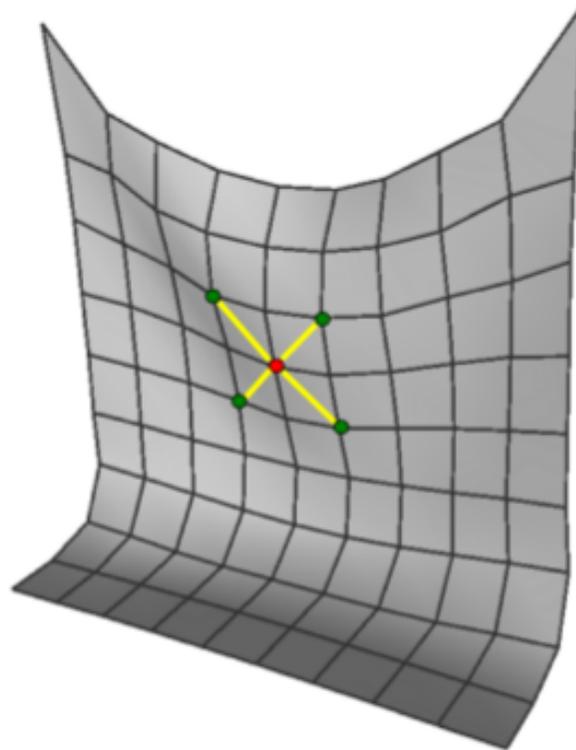
Simulation de tissus



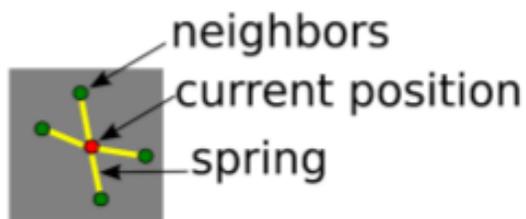
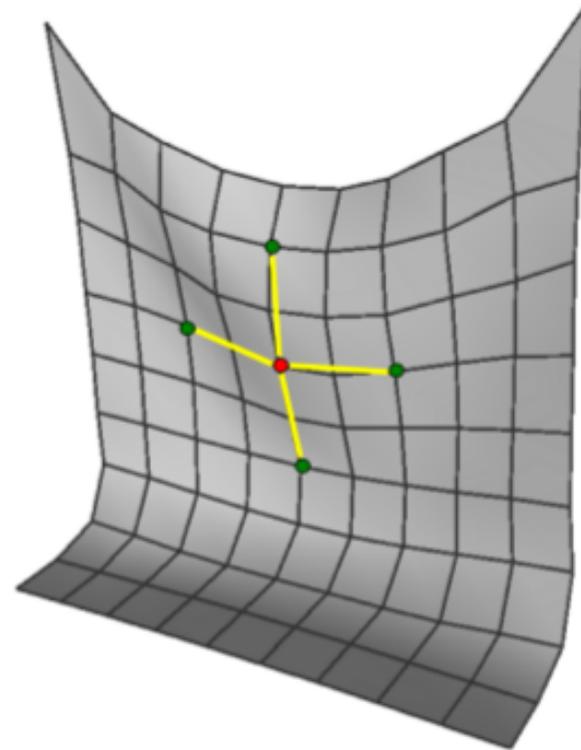
Structural springs



Shearing springs



Bending springs

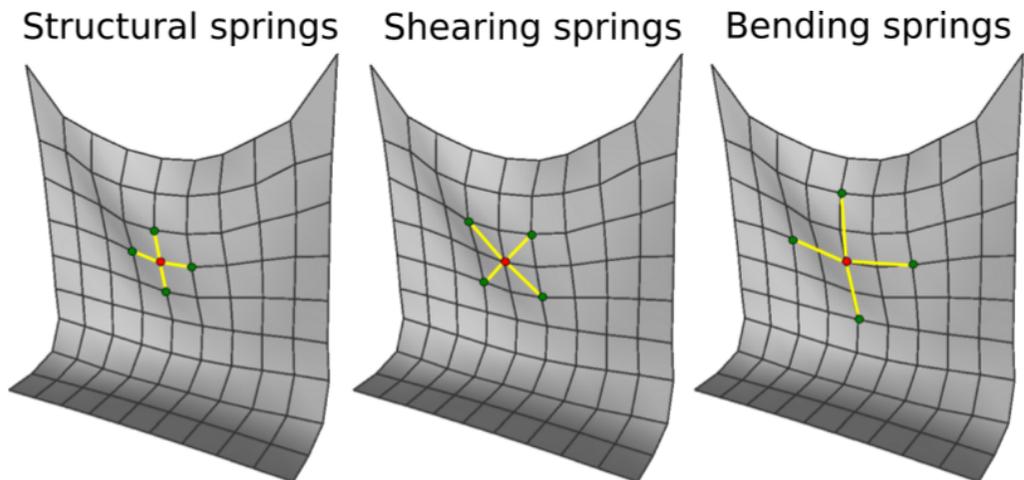
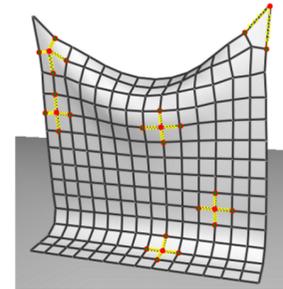


Simulation de tissus

Algorithme Euler explicite

```
//calcul des forces
Pour tout i
  Pour j: 4 voisins directes (structure) K=K1
          4 voisins diagonaux (shear) K=K2
          8 voisins (bend): K=K3
    u=x[i]-x[j]
    F[i] += K (L0-norm(u)) *u/norm(u)
  fin Pour
fin Pour

//MAJ
Pour tout i
  v[i] += dt *F[i]
  x[i] += dt*v[i]
fin Pour
```



Simulation de tissus

- Vêtement = Tissu + collisions complexes



[Grinspun, SIGGRAPH 09]



Limitations

Dépendance au maillage

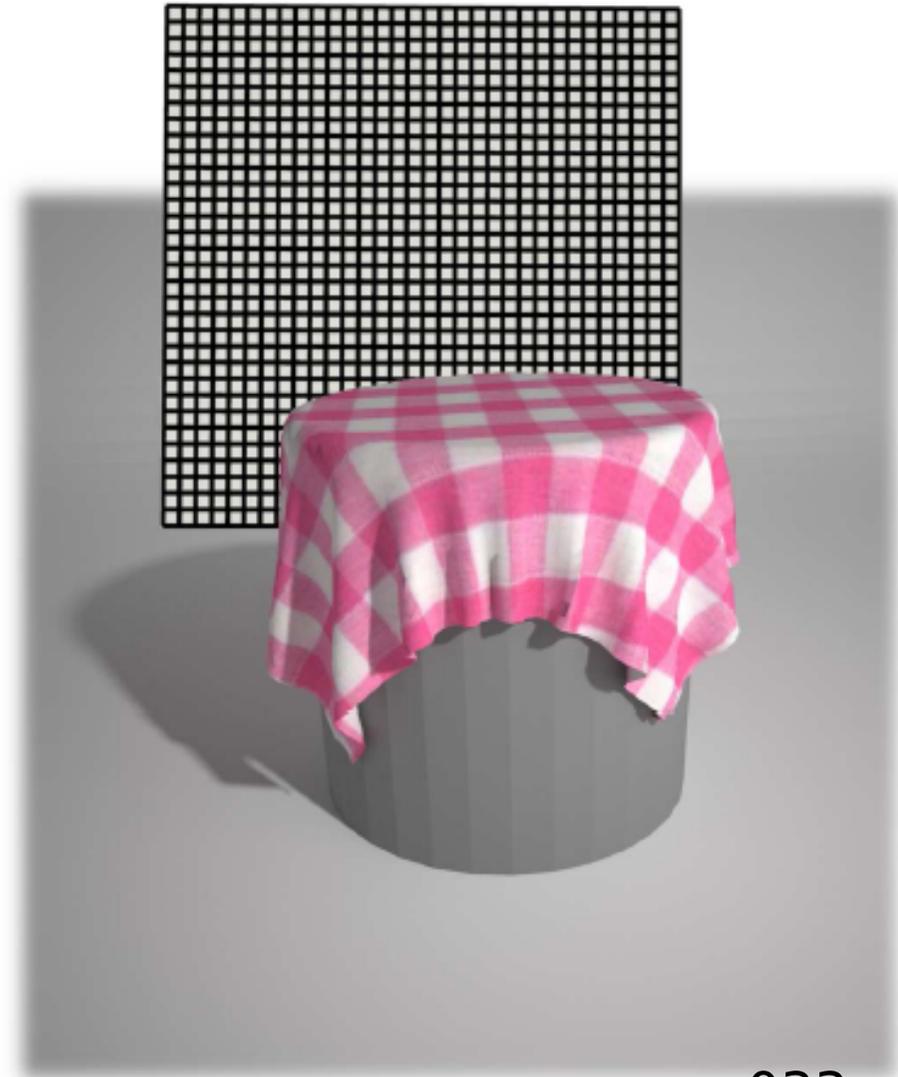
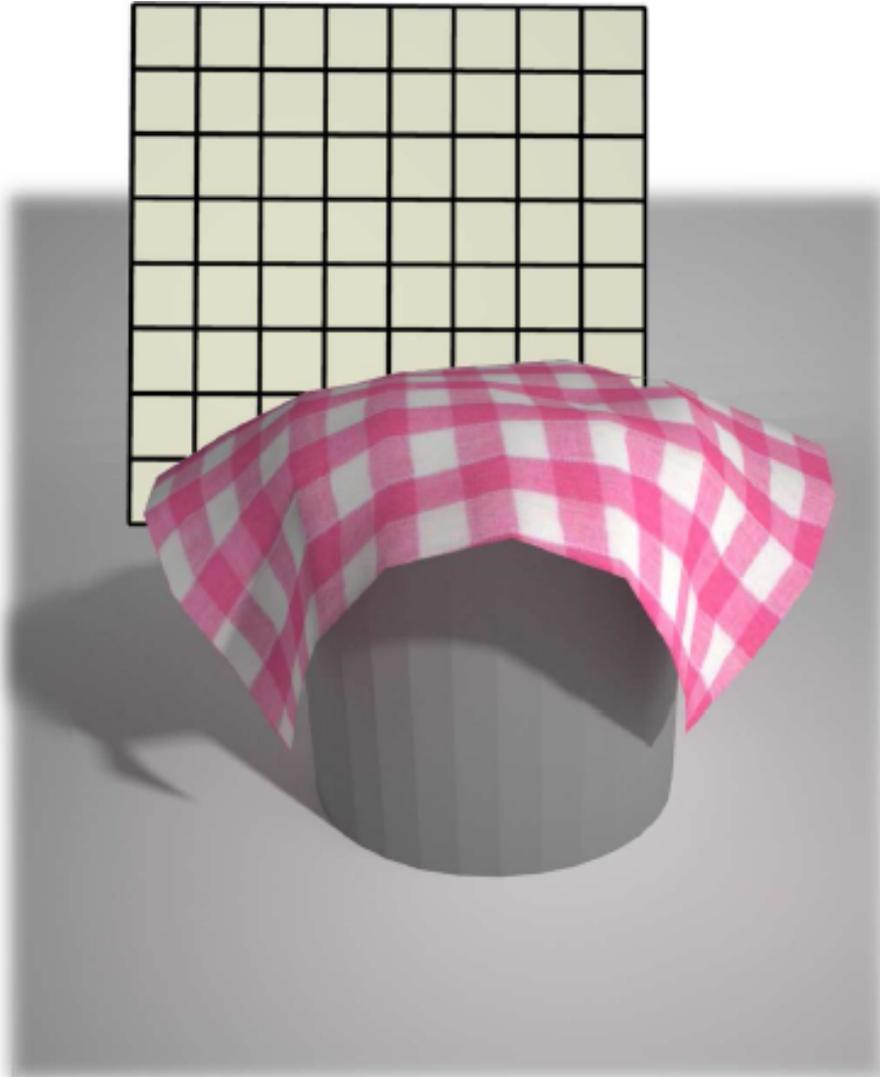


Schéma implicite

- Forme vectorielle :

$$u(t) = (x_0(t), x_1(t), \dots, x_{N-1}(t), v_0(t), \dots, v_{N-1}(t))$$

$$u'(t) = \mathcal{F}(u(t))$$

- ⇒ Système non linéaire à N inconnues : On ne sait pas inverser.
- ⇒ Linéarisation en $x(t) - x_0 = \Delta L v(t) + \mathcal{O}((\Delta L)^2)$
 - On se ramène à un système linéaire.
 - Perte de la stabilité inconditionnelle : En pratique très stable.

Linéarisation:

$$\mathbf{f}(\mathbf{x}_n + \Delta\mathbf{x}, \mathbf{v}_n + \Delta\mathbf{v}) = \mathbf{f}_n + \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \Delta\mathbf{x} + \frac{\partial \mathbf{f}}{\partial \mathbf{v}} \Delta\mathbf{v}$$

Systeme linéaire à chaque pas de temps:

$$A\Delta\mathbf{v} = \mathbf{b},$$

$$A \equiv \left(I - \Delta t M^{-1} \frac{\partial \mathbf{f}}{\partial \mathbf{v}} - \Delta t^2 M^{-1} \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right)$$

$$\mathbf{b} \equiv \Delta t M^{-1} \left(\mathbf{f}_n + \Delta t \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \mathbf{v}_n \right).$$