ETI5 Majeure Image: Animation physique

CPE Lyon
damien.rohmer@cpe.fr

26 Novembre 2010

Animation physique

- Quand on ne sait pas le déformer *à la main*.
- Quand il faut modéliser des phénomènes physiques.

[Fedkiw SIGGRAPH 06],[Grinspun SIGGRAPH 07]

Principe générale

1. Défini un système S_0 (positions, vitesses, forces, ...) à l'instant $t = 0$.
2. On modélise son évolution temporelle par une équation aux dérivées partielles (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 pourra écrire

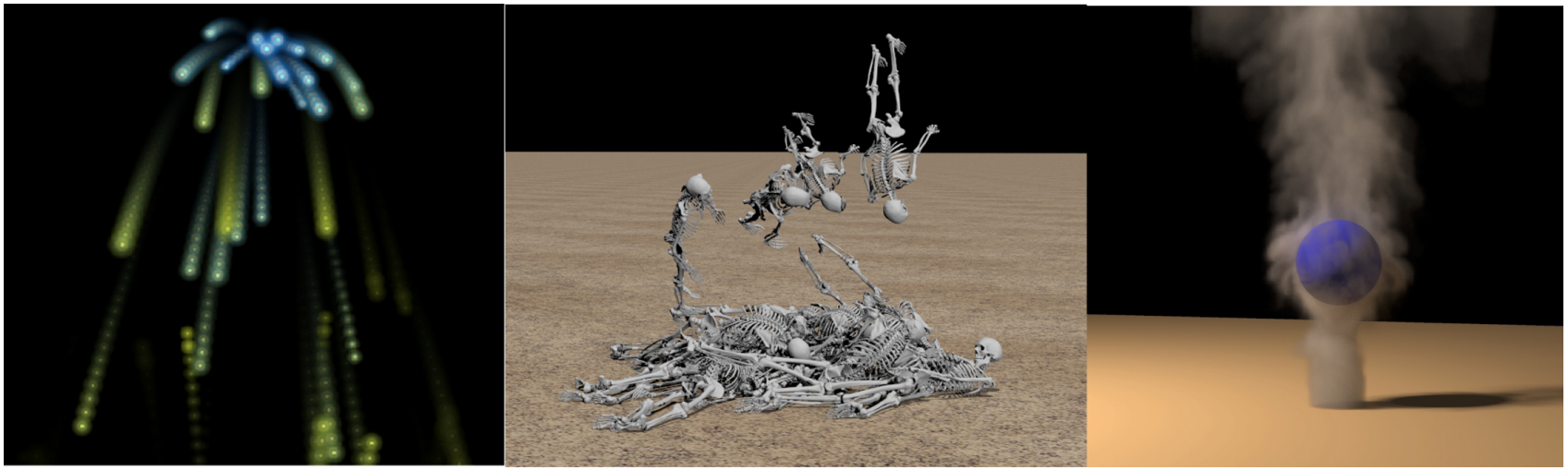
$$\frac{\partial S}{\partial t} = \mathrm{A}S + \mathcal{H}\left(S(t), \frac{\partial^m S}{\partial x^m}\right)$$

3. On résoud numériquement le système pour avancer temporellement suivant un pas Δt.

Approximation

Différents niveaux d'approximations

1. **Méthodes particulaires** (particle system) = on réduit tout à un point (pas de rotation + EDO) : le + simple, le moins fidèle.
2. **Mécanique du solide** (rigid bodies) = paramètres identiques pour tout le solide.
3. **Mécanique des milieux continus** (continuum mechanics) = paramètres variables dans la forme : EDP (FEM, FV, . . .).



[Fedkiw IEEE TVCG 06],[Fedkiw SIGGRAPH 01]

Système de particules

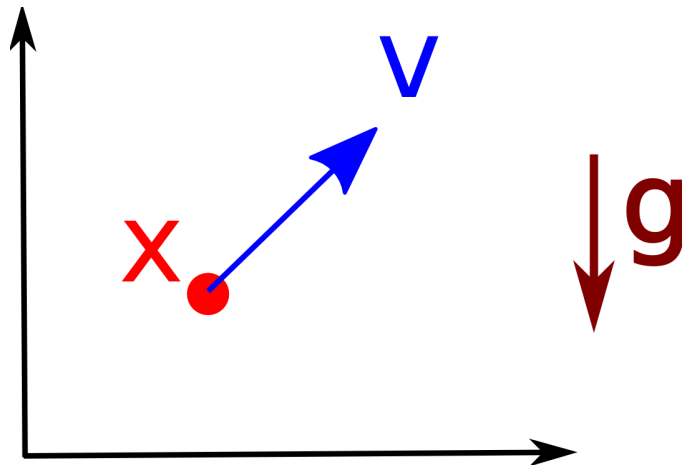
1-Description du système :

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

2-Equation différentielle :

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

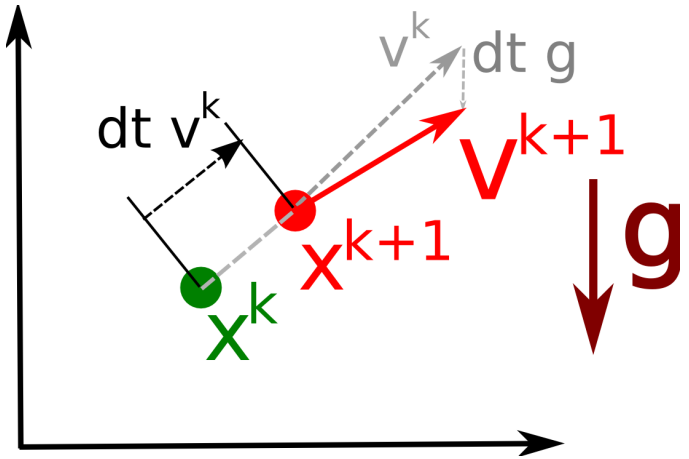
$$\Rightarrow \begin{cases} x'(t) = v(t) \\ v'(t) = g \end{cases}$$



Résolution numérique

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

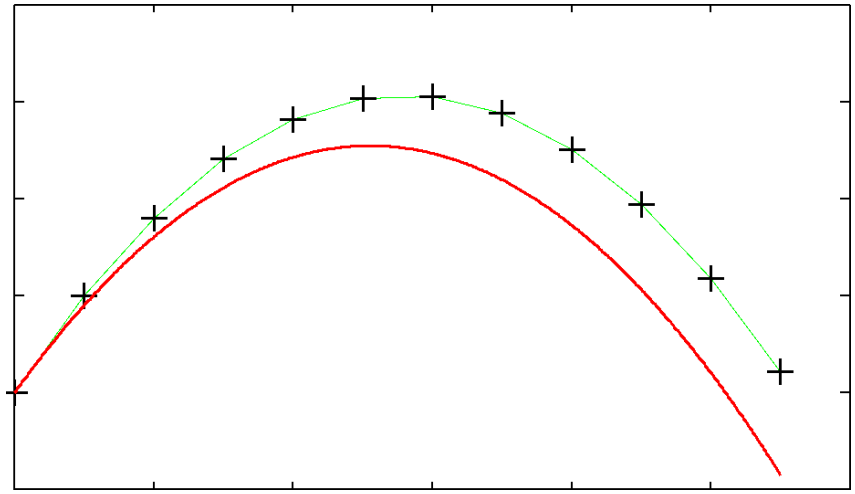
```
x=x0;
v=v0
for(k=0;k<N;++k)
{
   x=x+dt*v;
   v=v+dt*g;
}
```

Quel erreur commet-on en résolvant numériquement ?

Equation résolue

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 récurence :

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

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

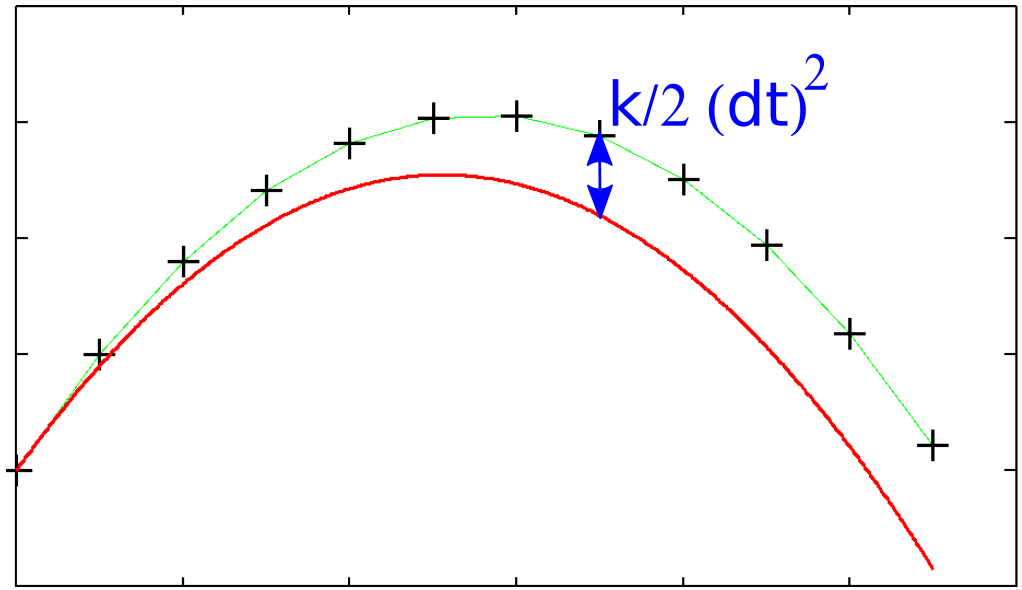
- Erreur :

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

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

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

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



Résolution matricielle

Approche matricielle :

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

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

$$\Leftrightarrow u_{,t} = \mathrm{A}u + b$$

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

Euler explicite

- Euler explicite :

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

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

Même résultats que précédemment

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

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

Euler implicite

- Euler implicite :

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

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

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

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

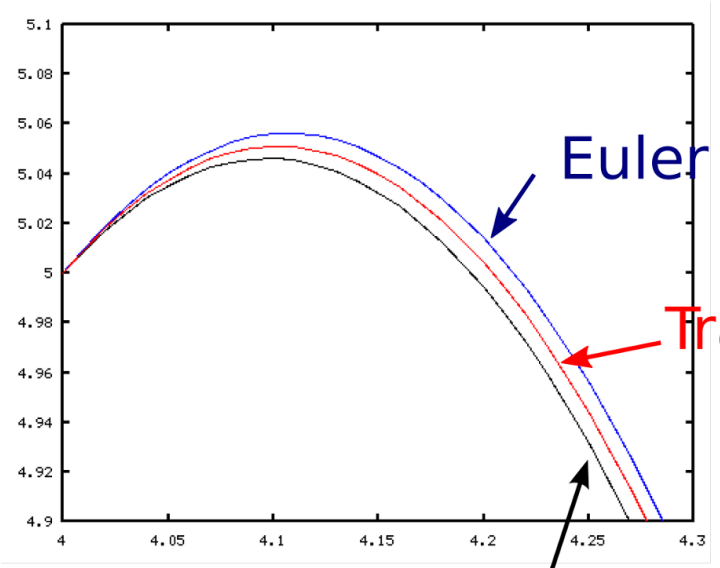
On ne résout toujours pas la bonne équation, et précision d'ordre 1 : $\|\tilde{x}(k\Delta t) - x(k\Delta t)\| = \frac{k}{2}(\Delta t)^2$.

Euler explicite/implicite

- 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 $\tilde{x}$.



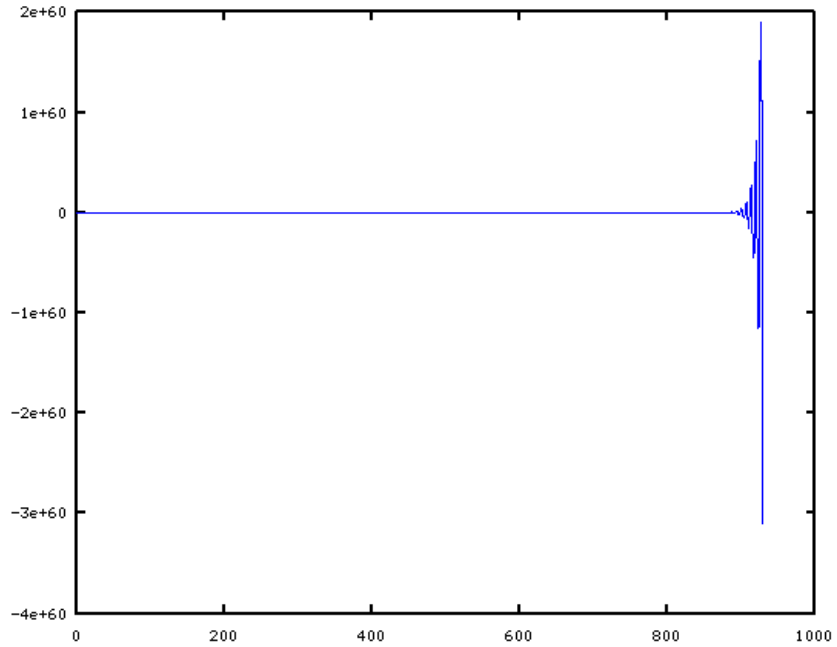
Implicite vs Explicite

Euler explicite

+ Trivial
- Potentiellement divergeant

Euler implicite

+ Inconditionnelement stable
- Inversion d'un système matriciel



Méthode d'ordre supérieur

- Méthodes explicites précise : Runge-Kutta pour résoudre $u'(t) = F(u,t)$.
- Ordre 4, estimation d'erreur d'ordre 5 : Dormand-Prince (ode45)

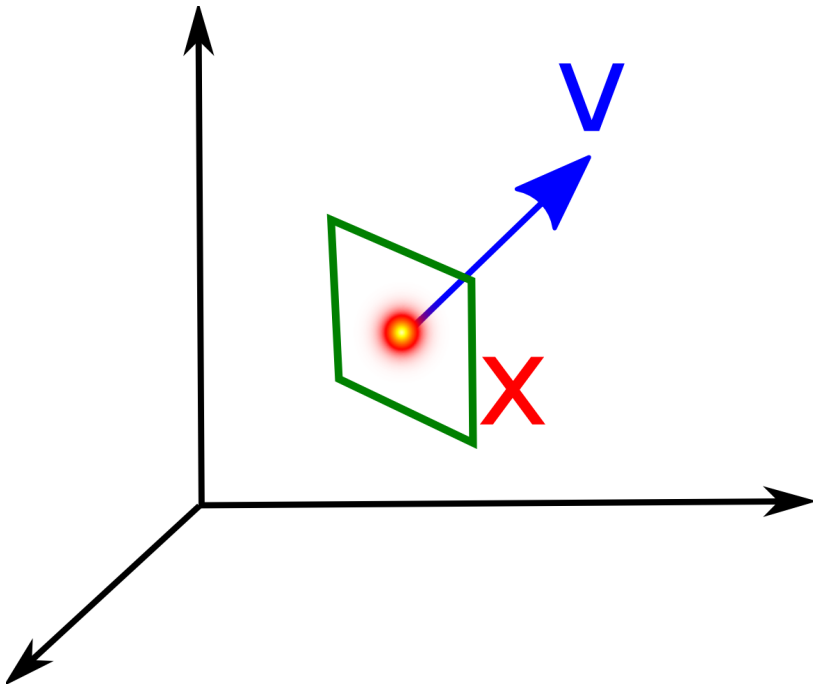
$$
\begin{aligned}
k_1 &= F(t_n, u_n) \\
k_2 &= F\left(t_n + \tfrac{1}{5}\Delta t, u_n + \tfrac{1}{5}\Delta t\ k_1\right) \\
k_3 &= F\left(t_n + \tfrac{3}{10}\Delta t, u_n + \left(\tfrac{3}{40}k_1 + \tfrac{9}{40}k_2\right)\Delta t\right) \\
k_4 &= F\left(t_n + \tfrac{3}{5}\Delta t, u_n + \left(\tfrac{3}{10}k_1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

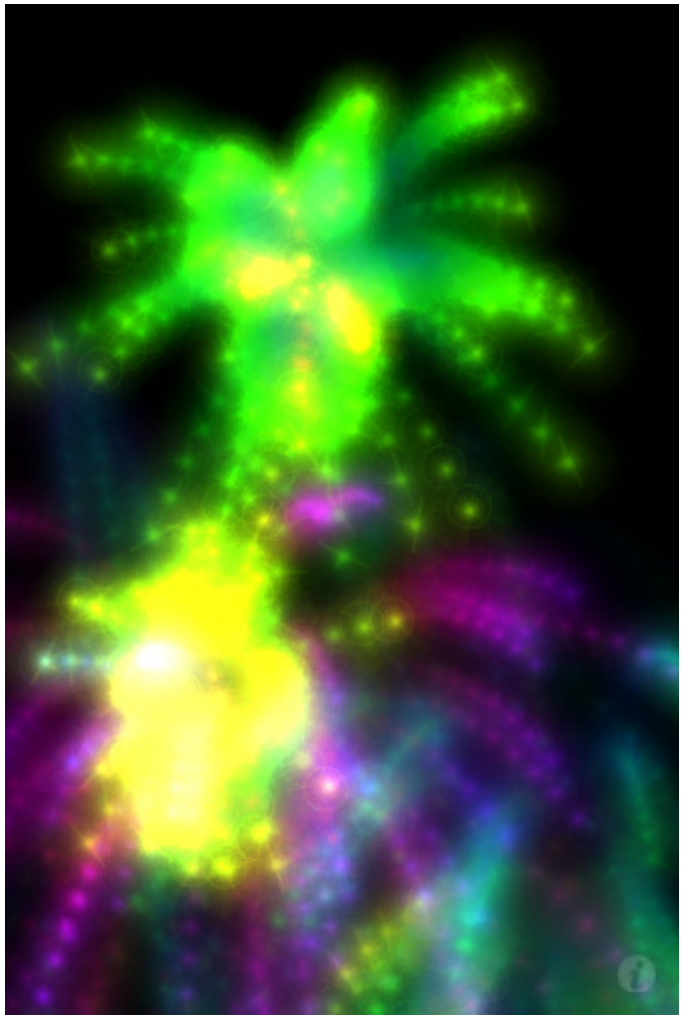
$$
\begin{aligned}
u^4_{n+1} &= u_n + \Delta t\left(\tfrac{2825}{27648}k_1 + \tfrac{18575}{48384}k_3 + \tfrac{13525}{55296}k_4 + \tfrac{277}{14336}k_5 + \tfrac{1}{4}k_6\right) \\
u^5_{n+1} &= u_n + \Delta t\left(\tfrac{37}{378}k_1 + \tfrac{250}{621}k_3 + \tfrac{125}{594}k_4 + \tfrac{512}{1771}k_6\right)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

Application : Sprites

Sprites=

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



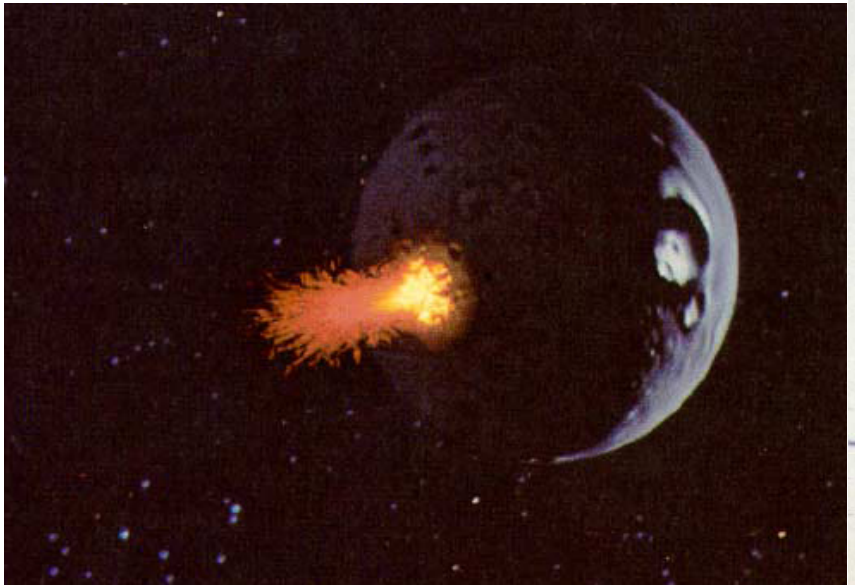


©Apple

Particules + trajectoires

- 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]



©Lucasfilm, Star Trek II

[Reeves, TOG 83]

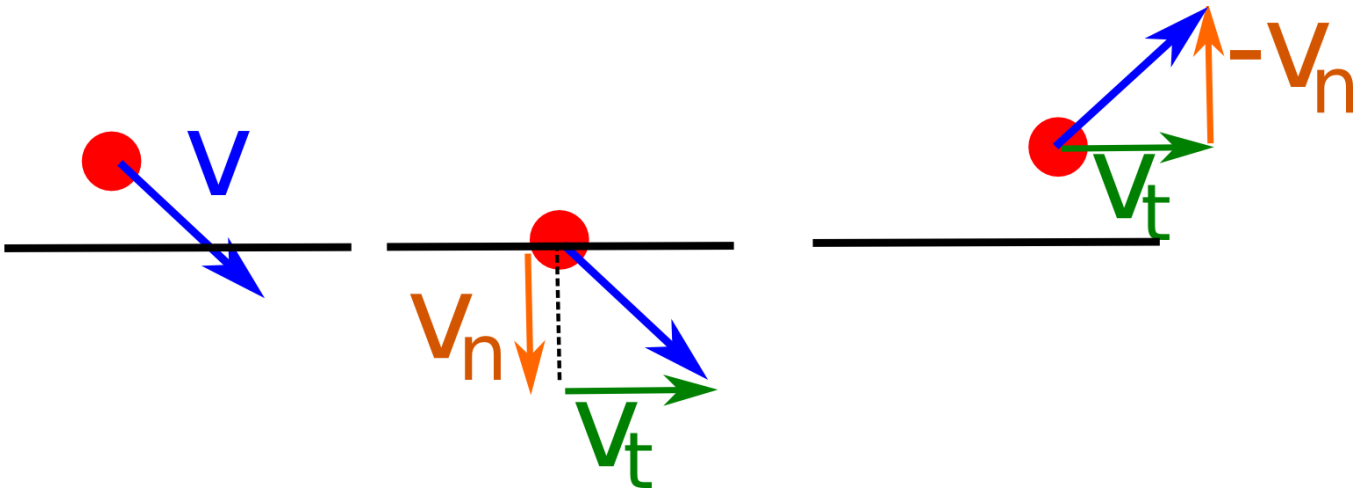
Système de particules + collision

- Les particules peuvent entrer en collision avec un plan : $\mathcal{P} :< x - p_0, n >= 0$
- Détection : $< x - p_0, n >< 0$
- Séparation : vitesse tangentielle v_t, vitesse normale v_n.

$$\begin{cases} v_n =< v, n > n \\ v_t = v - < v, n > n \end{cases}$$

- Après collision, perte d'energie : amortissement par μ=coeff de restitution

$$v^{k+1} = -\mu < v^k, n > n + (v^k - < v^k, n > n)$$

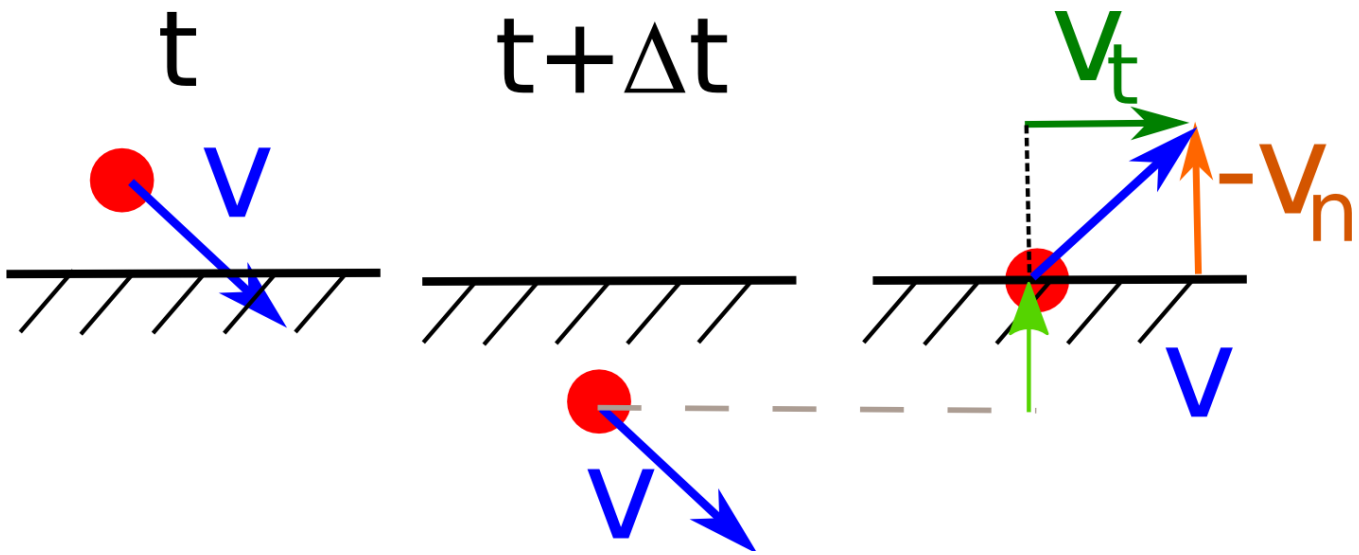


Système de particules + collision

- Attention à la discrétisation temporelle !
  1. Projection : Simple, faux (instabilitées).

$$x^{k+1} = x^k - < x^k - p_0, n > n$$

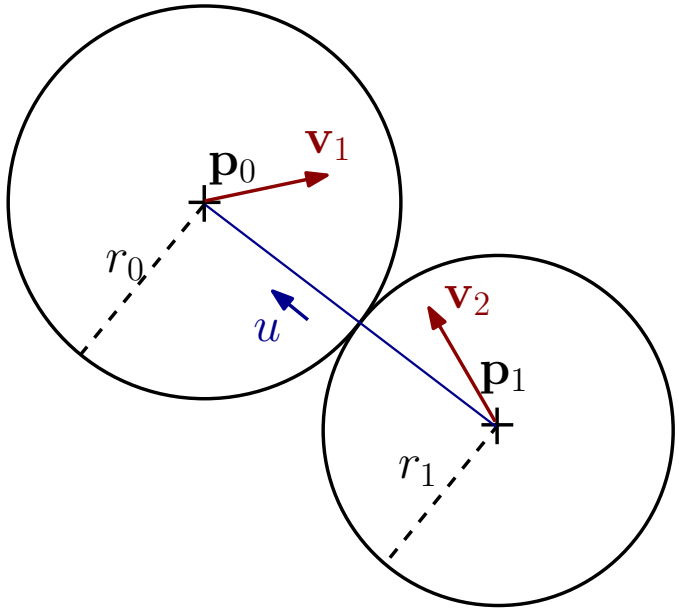
  2. Recherche arrière : Moins simple, moins faux.



Sphères dures

- Particules = Sphères masse m, centre x, rayon r.
- Particules en collision si

$$\|x_1 - x_2\| < r_1 + r_2$$



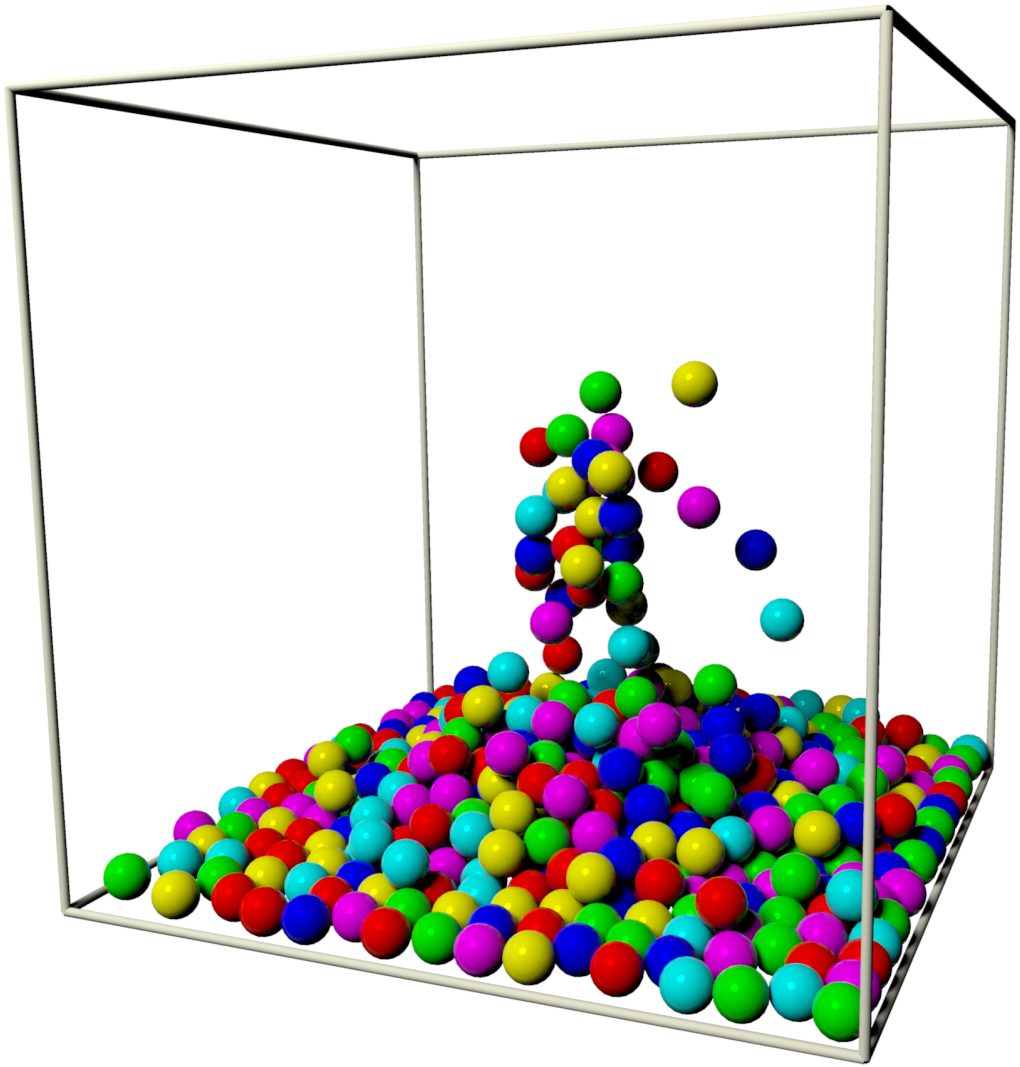
- Nouvelle vitesse en choc elastique

$$\begin{cases} v_1^{k+1} = v_1^k + \frac{1}{m_1+m_2}\left[m2 < v2^k, u > -\frac{1}{2}(m_1 + 3m_2) < v1^k, u >\right] u \\ v_2^{k+1} = v_2^k + \frac{1}{m_1+m_2}\left[m1 < v1^k, u > -\frac{1}{2}(m_2 + 3m_1) < v2^k, u >\right] u \\ u = x1 - x0 \end{cases}$$

- Penser à reprojeter sur la surface de contact

Sphères dures

- On génère beaucoup de sphères

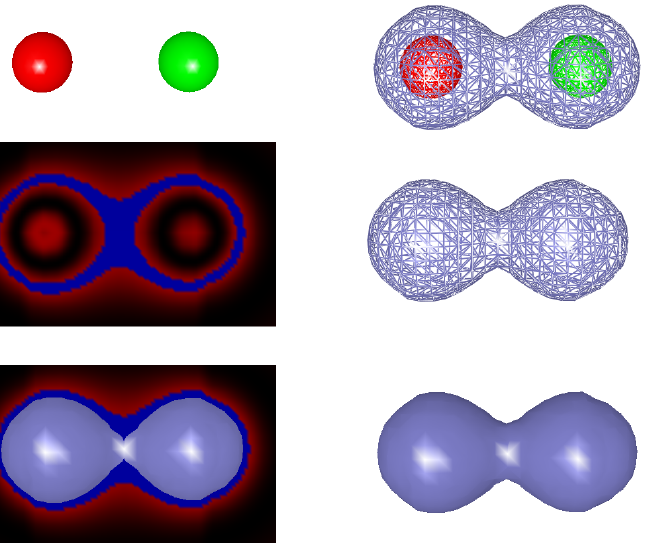


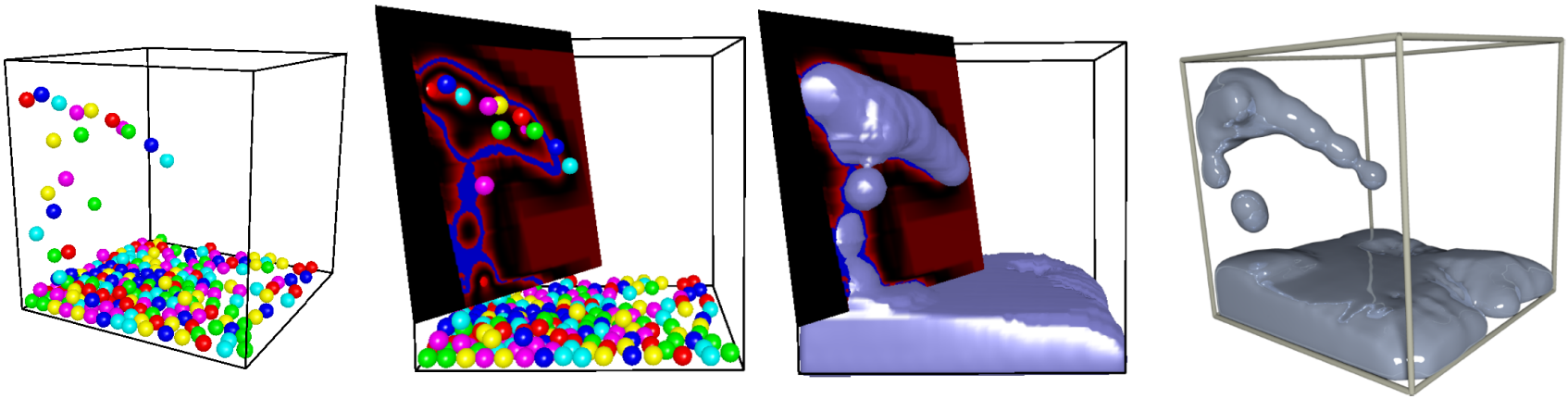
Sphère : Potentiel implicite

- Affecte potentiel de l'espace à chaque particule (ex. blobs)

$$f(x) = \sum_i \exp(-a\|x - x_i\|^n)$$

$\Rightarrow$ Simulation de fluides.



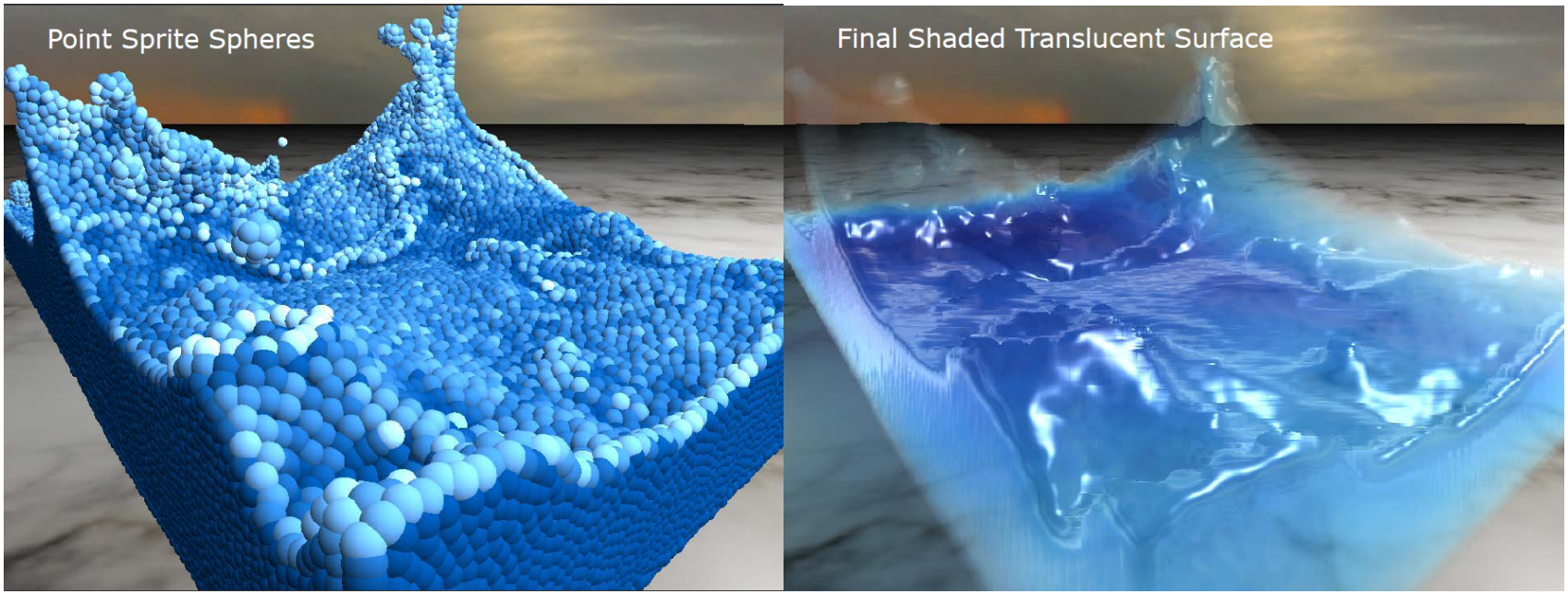


Detection efficace de collision

- Algorithme force brute :

```
for(i=0;i<N;++i)
 for(j=i+1;j<N;++j)
  if( norm(xi-xj)<r1+r2 )
   CollisionResponse(i,j)
```

- Complexité $\mathcal{O}(N^2)$
- Impossible pour 10^{3-6} particules en temps réel.



©NVIDIA, [Green, SIGGRAPH 10]

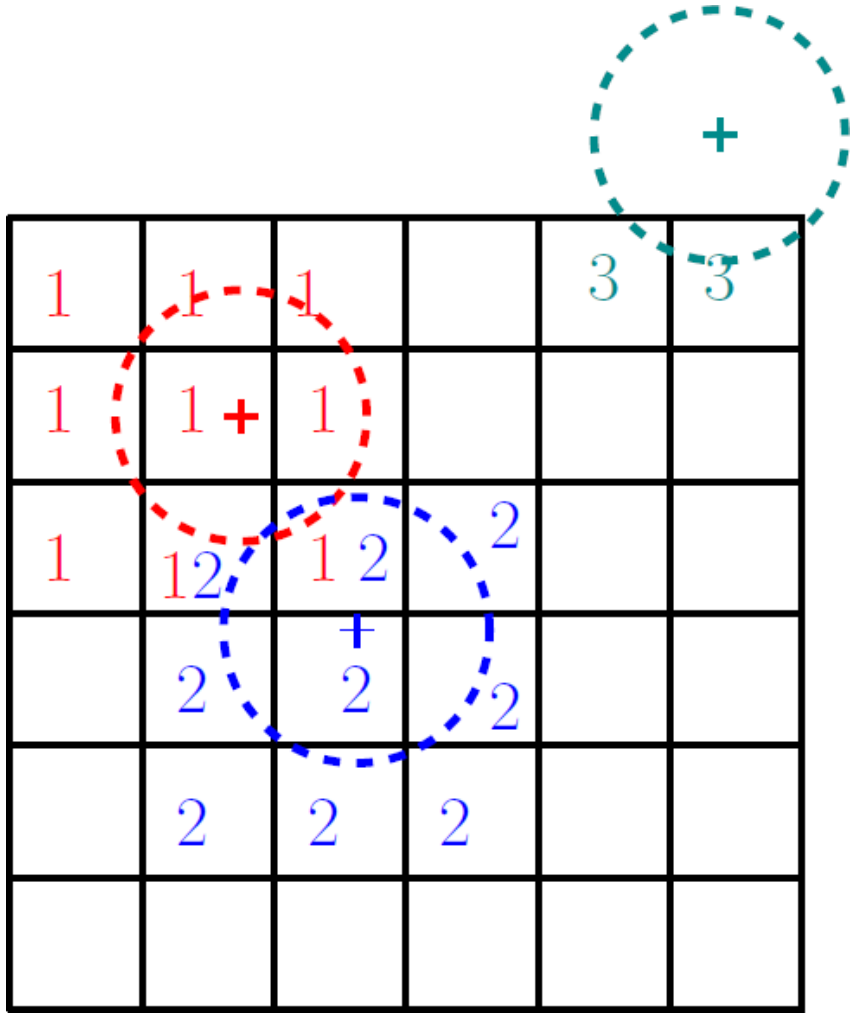
Grilles acceleratrices

- Grille régulière
- Recherche en $\mathcal{O}(1)$

\+ Simple, recherche spatial efficace

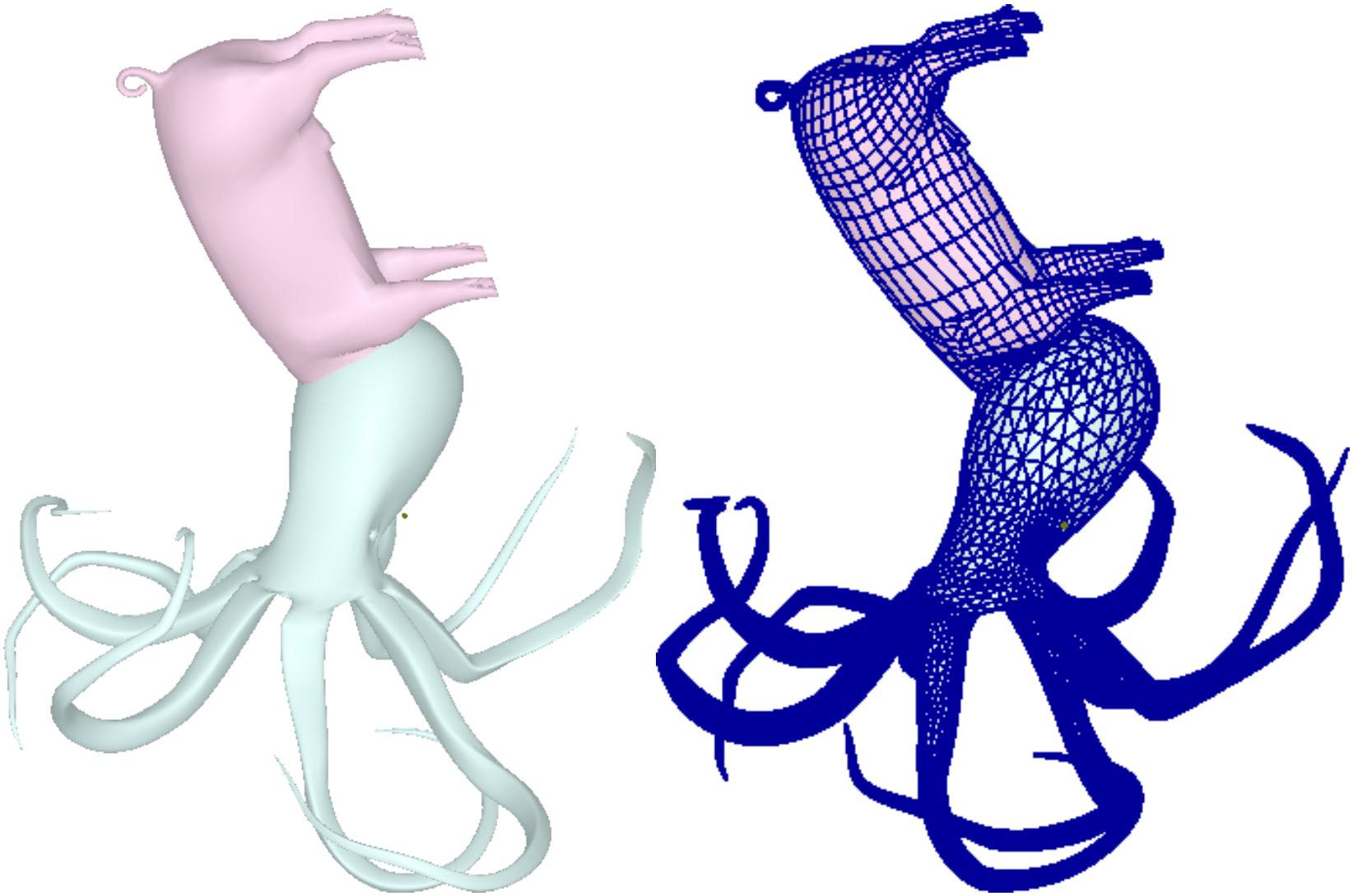
\+ Ok pour échantillonage uniforme

\- Utilisation mémoire



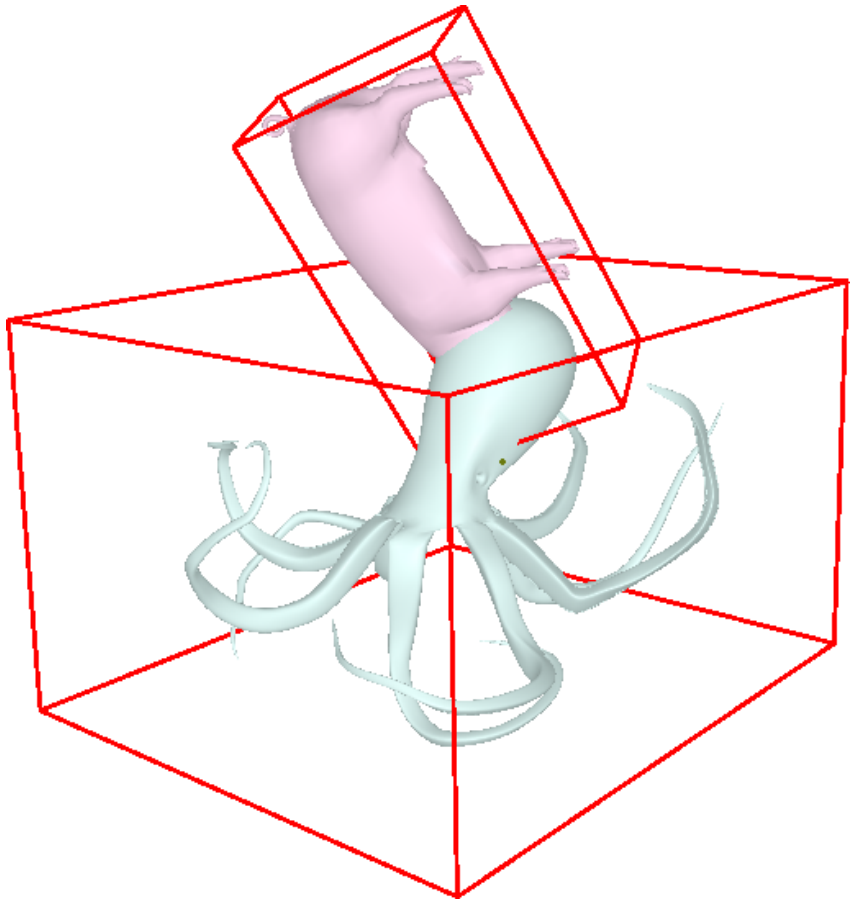
Détection de collision objets

- Recherche triangles/triangles en $\mathcal{O}(N_{T1}N_{T2})$.
- Zones vides.



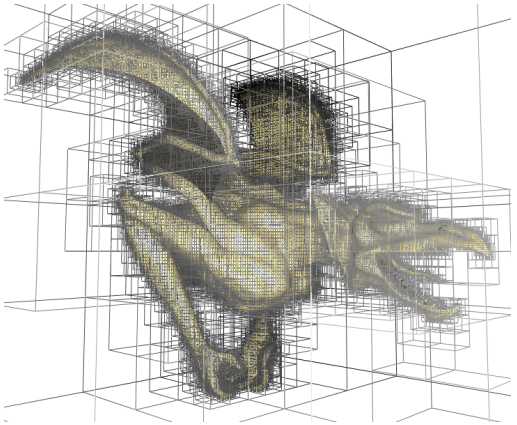
Bounding Box

- Boite englobante = Bounding Box (BB)
- Le plus simple : AABB (Axis Aligned Bounding Box)
- Sphère englobantes
- \+ Détection de Non-Collision en $\mathcal{O}\left(N_{\text{obj}}^2\right)$

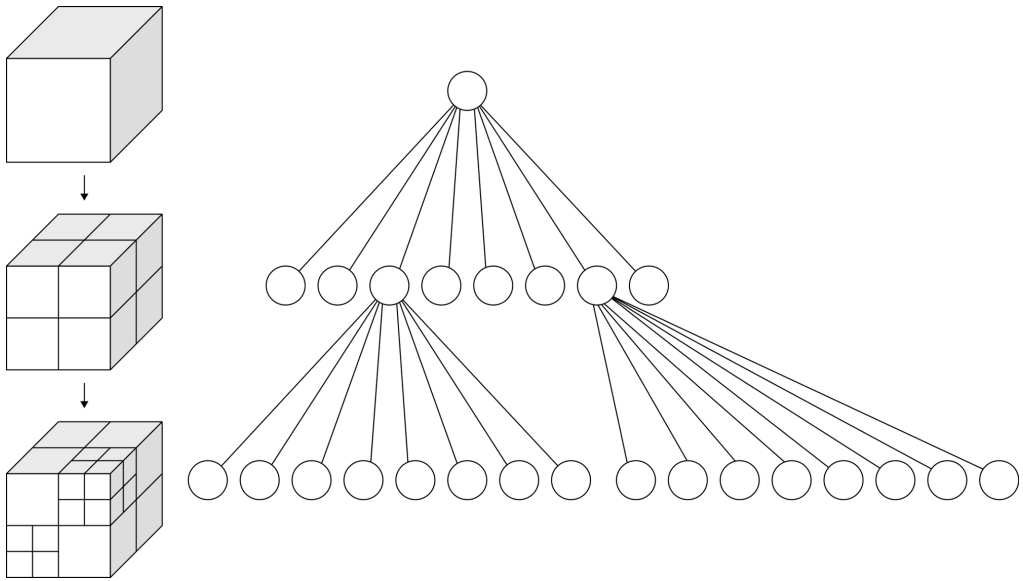


Octree

- Grille auto-adaptable : **Octree**
- \+ Géométrie complexe
- \+ Recherche en $\mathcal{O}(log(M))$, M taille arbre.



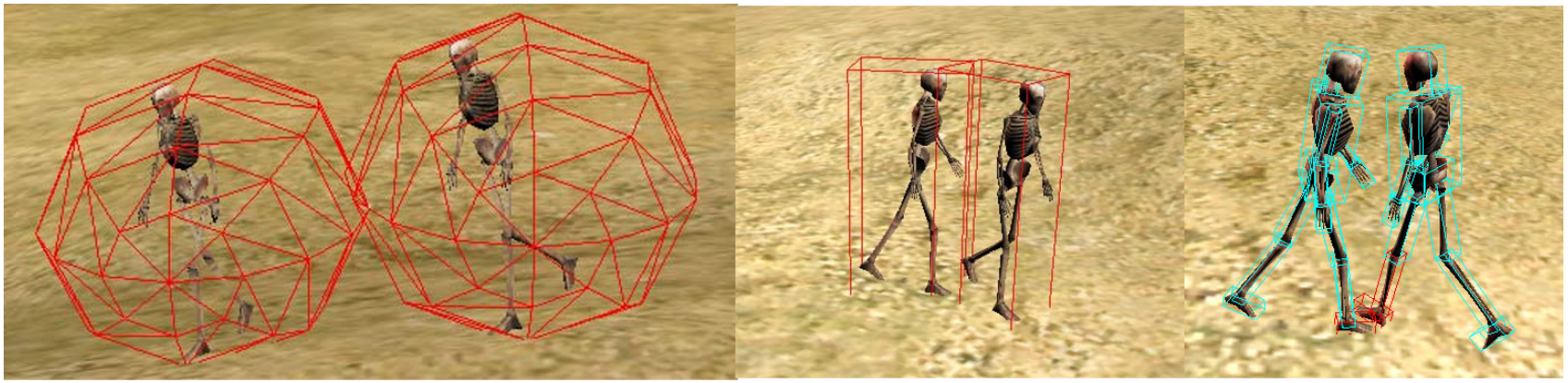
[Lefebvre GPU Gems 2, 2004]



Wikipedia

Collisions

- En pratique : Niveaux de détails de boites englobantes
  - ex. Spheres dans BB dans Octree
  - ex. OBB dans AABB dans Spheres



Ditchburn

1. Test grossier
2. Test fin
3. ...
4. Optionnel : Calcul de la vraie collision

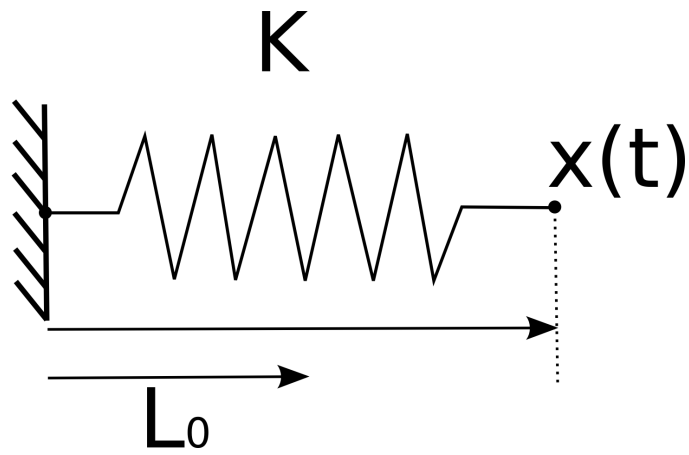
$\Rightarrow$ Tests de Non-collision rapide

$\Rightarrow$ Détection de collision lent

Masse-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 $\left(1 + (\Delta t)^2 \frac{K}{m}\right) < 1 \Rightarrow$ Euler explicite diverge forcément !

  - Euler implicite converge vers 0.

Masse-ressorts

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

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

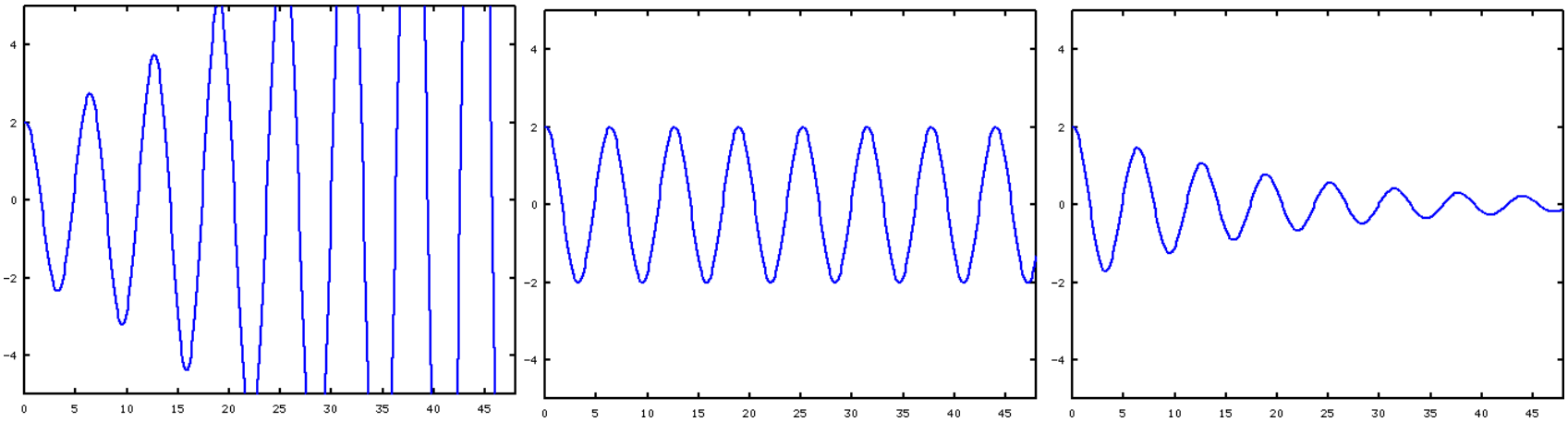
- Nouvelle récurence pour Euler explicite :

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

- Stable si $\Delta t < \sqrt{\frac{\mu}{K}}$
- Ressorts raides : K grand $\Rightarrow$ Système raide (stiff).

Masse-ressorts

! Précision $\neq$ Stabilité

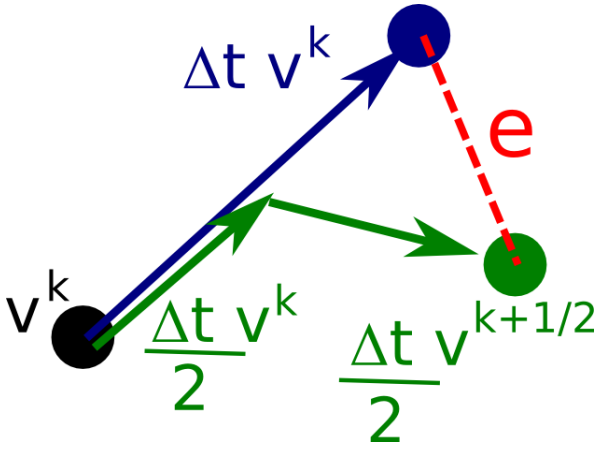


- Schémas explicite $\Rightarrow$ petits pas de temps / rapide.
- Schémas implicites $\Rightarrow$ grands pas de temps / inversion système.

Pas de temps

- Calcul automatique du pas de temps Δt :
  1. Calcul $u_1^{n+1} = F(\Delta t, u^n)$
  2. Calcul $u_2^{n+1/2} = F(\frac{\Delta t}{2}, u^n)$; $u_2^{n+1} = F(\frac{\Delta t}{2}, u^{n+1/2})$
  3. $e = \|u_1^{n+1} - u_2^{n+1}\|$

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

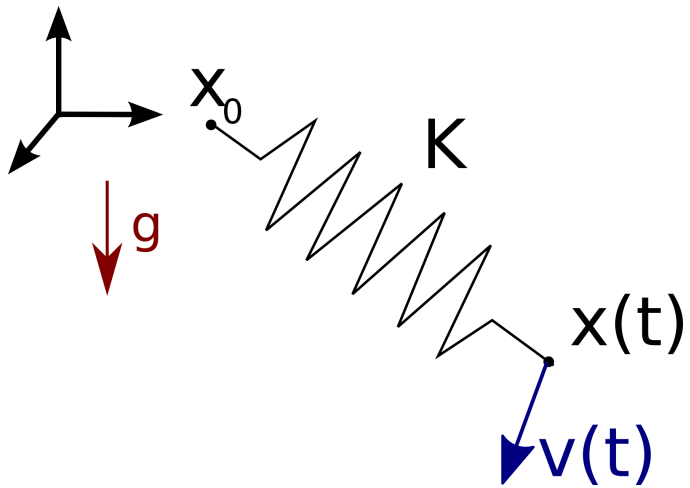


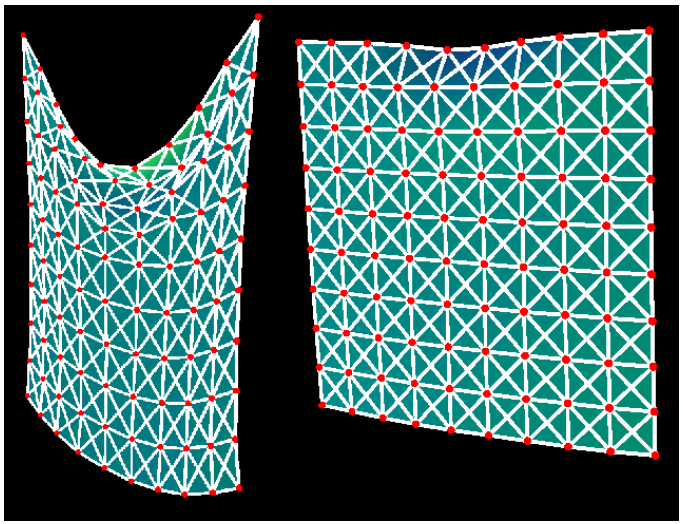

```
Faire
   u1=F(dt,u)
   u2=F(dt/2,u);
   u2=F(dt/2,u2);
   e=norm(u1-u2)
   si e<E_min
     dt*=2;
   si e>E_max
     dt/=2;
tant que
   !(E_min<e<E_max)
```

Ressorts 3D

- En 3D :

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





M. Fisher

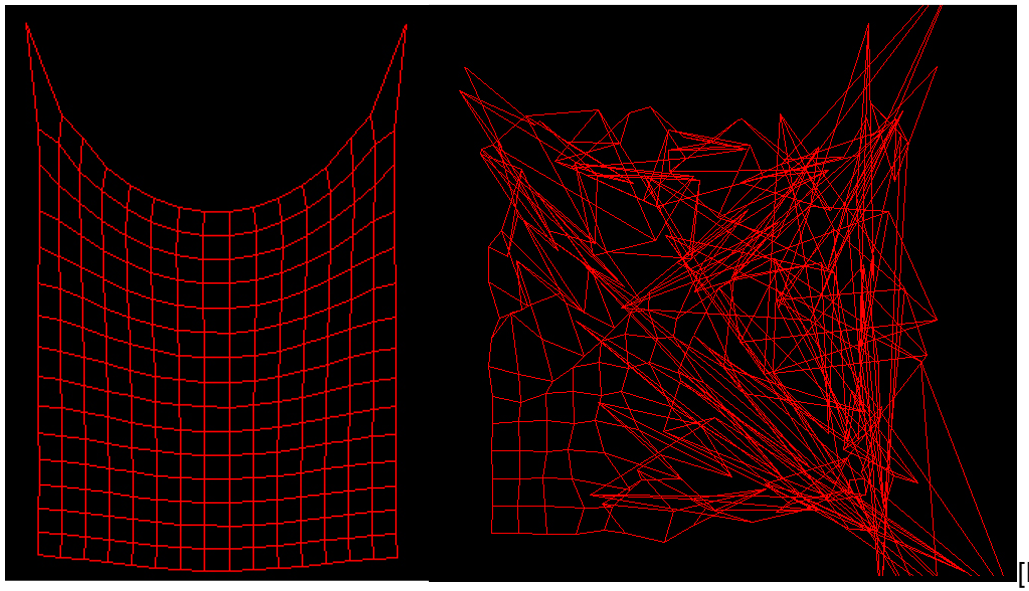
Masses-ressorts : Animation de tissus

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

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

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

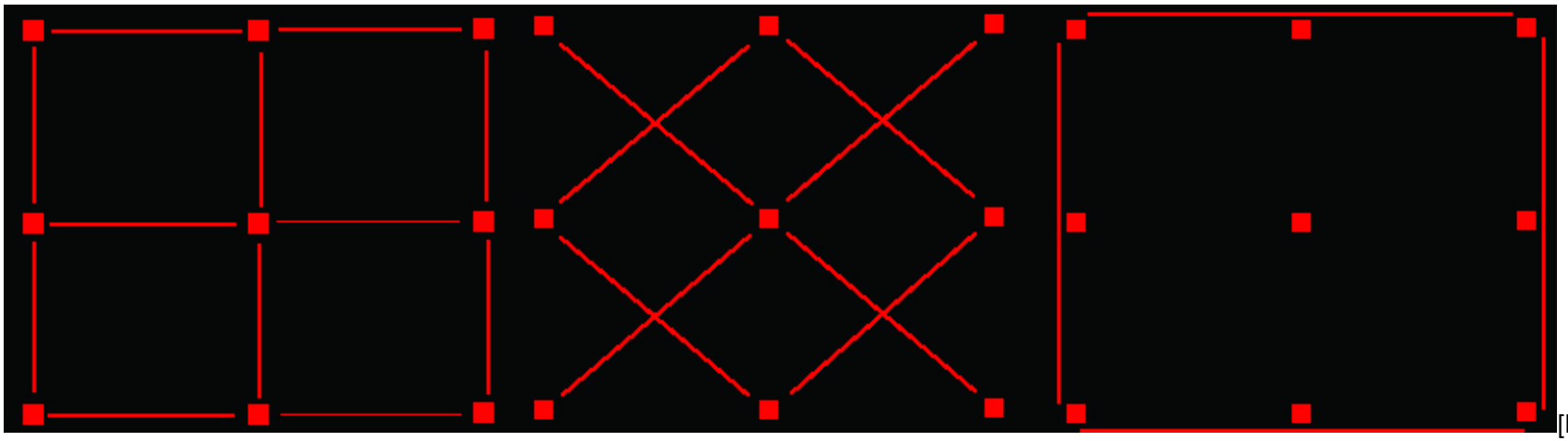
$\Rightarrow$ Schéma d'intégration favori



[P. Jacobs]

Masses-ressorts, animation de tissu

- Ressorts d'elongation (stiffness) : K_1
- Ressorts de cisaillement (shear) K_2
- Ressorts de courbure (bending) K_3



[P. Jacobs]

Algorithme

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
```

Tissu implicite ?

- Forme vectorielle :
  $u(t) = (x_0(t), x_1(t), \ldots, x_{N-1}(t), v_0(t), \ldots, v_{N-1}(t))$

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

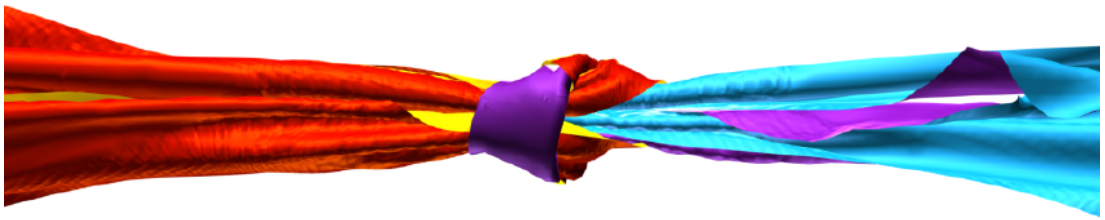
$\Rightarrow$ Système non linéaire à N inconnues : On ne sait pas inverser.

$\Rightarrow$ 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 inconditionelle : En pratique très stable.

Vêtement

- Vêtement = Tissu + collisions complexes



[Grinspun, SIGGRAPH 09]

©DAZ3D, Dynamic Clothing