Chapitre 3 : Mouvement et Collision des particules

Image et Interaction 2D

Fabrice Aubert
fabrice.aubert@univ-lille.fr

Université de Lille
FACULTÉ DES SCIENCES ET TECHNOLOGIES
Département Informatique

Licence Informatique 2018-2019

1 Mouvement

Problème

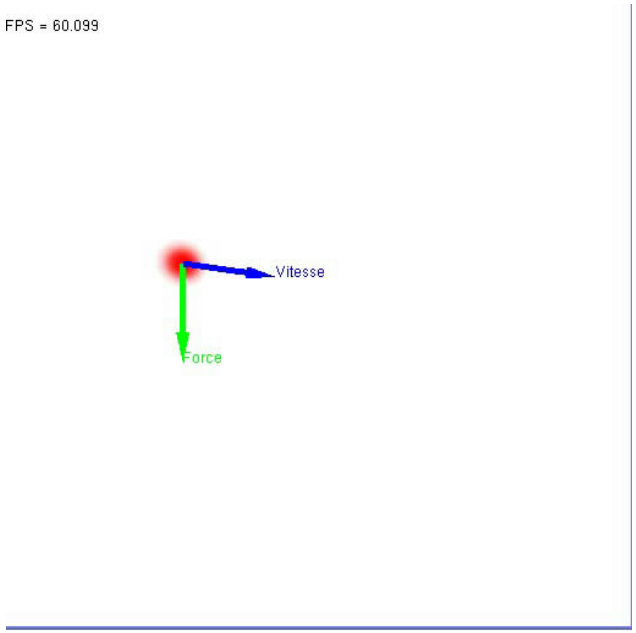
- A chaque itération de l'animation il faut définir la nouvelle position des particules en mouvement.

```
var loop = function() {
    time+=deltaTime;
    updateData(); // traiter les données dont calculer les nouvelles positions
    draw(); // tracer
    window.requestAnimationFrame(loop); // recommencer
}
```

- On note $x(t)$ la position de la particule à l'instant t
- x correspond au champ `position` de la particule dans le code, t correspond à la variable `time`.
- On note également δt le pas de temps (qui correspond à la variable `deltaTime` dans le code).

Comment calculer $x(t)$?

- On peut déterminer "manuellement" la position à chaque instant (formule connue, suivie d'une trajectoire faite à la main, mouvement artistique, ...)
- Ici on choisit de faire une simulation "basée physique" qui repose sur le fait que le mouvement est généré par des forces.



- Une force appliquée en un point (en une particule par exemple) est représentée par un vecteur (coordonnées x et y en 2D).
- Exemple de force : le poids issue de la gravité calculé par $F = \text{masse} \times g$ avec $g = (0, 9.81)$ sur la Terre (i.e. une force "vers le bas" ; on peut bien sûr envisager n'importe quelle valeur...).

- L'intérêt est de se contenter de définir des forces, puis de laisser dérouler l'animation. Si les forces sont "réelles", on obtiendra un mouvement "naturel".
- Le problème devient : comment calculer $x(t)$ à partir des forces ?

Relation fondamentale de la dynamique

- A partir de l'ensemble des $F(t)$ appliquée à une particule, on peut calculer son accélération $a(t)$

$$a(t) = \frac{F(t)}{m} \text{ avec m la masse de la particule}$$

- A partir de l'accélération $a(t)$, nous allons déterminer la vitesse $v(t)$. Puis à partir de la vitesse $v(t)$, nous déterminerons la position $x(t)$.
- Une vitesse est une variation de position dans le temps. Une accélération est une variation de vitesse dans le temps. Plus formellement :

$v(t) = \frac{\partial x}{\partial t}(t) = \dot{x}(t)$ (dérivée de la position par rapport au temps)
$a(t) = \frac{\partial v}{\partial t}(t) = \dot{v}(t) = \ddot{x}(t)$ (dérivée seconde de la position par rapport au temps)

- Le calcul de la vitesse $v(t)$ puis de la position $x(t)$ peut s'avérer délicat selon la compléxité des forces appliquées (il faut calculer des primitives de $a(t)$).

Approximation (principe d'Euler explicite)

- On considère que la vitesse $v(t)$ est la vitesse **moyenne** de la particule pour aller de la position $x(t-\delta t)$ (i.e. la position précédente de la particule qui est connue) à la position actuelle $x(t)$ (i.e. la nouvelle position qui est à déterminer).

$$v(t) = \frac{x(t) - x(t-\delta t)}{\delta t}$$

- Ce qui nous donne $x(t) = x(t-\delta t) + v(t) \times \delta t$ (ou plus "informatiquement" : `x_new=x_old+v_new*deltaTime`)
- On fait la même chose sur l'accélération pour déterminer $v(t)$: $a(t)$ est l'accélération moyenne entre les vitesses précédentes et nouvelles :

$$a(t) = \frac{v(t) - v(t-\delta t)}{\delta t}$$

- Ce qui nous donne $v(t) = v(t-\delta t) + a(t) \times \delta t$ (ou plus "informatiquement" : `v_new=v_old+a_new*deltaTime`)

Pour résumer

- A chaque itération de l'animation (i.e. au temps *t*), on définit/calcule des forces pour chaque particule (gravité, vent, frottement, ressorts entre les particules, ...).
- Pour chaque particule, on fait la somme de toutes les forces pour donner le vecteur `F`.
- On calcule l'accélération de la particule par `a_new = F/m`.
- On calcule alors la vitesse et la position de la particule par :

```
v_new=v_old+a_new*deltaTime // faire et appliquer les méthodes de Vector pour traduire les opérateurs * et +
x_new=x_old+v_new*deltaTime // faire et appliquer les méthodes de Vector pour traduire les opérateurs * et +
```

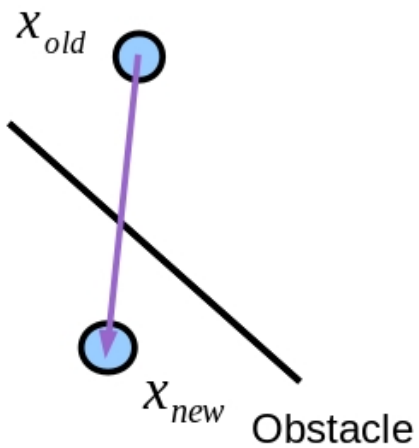
2 Collision

Problème

- Les particules en mouvement peuvent rentrer en collision avec des obstacles.
- Obstacles : cercles et segments dans la scène
- Les étapes du traitement de la collision sont :
  1. La détection de collision :
     - déterminer s'il y a intersection entre la trajectoire de la particule et l'obstacle.
     - déterminer les informations nécessaires à l'étape suivante de réponse (par exemple : le point où se produit la collision ?).
  2. La réponse à la collision : modifier la trajectoire de la particule ("rebond").
- Ces 2 étapes sont intimement liées : il faut déterminer quelles sont les informations nécessaires à la réponse pour offrir une détection de collision pertinente :
  - un algorithme de détection qui indiquerait uniquement s'il y a collision ou non ne suffit sans doute pas.
  - à l'inverse un algorithme couteux qui donnerait des informations non nécessaires peut s'avérait inutile.

Résolution statique des collisions

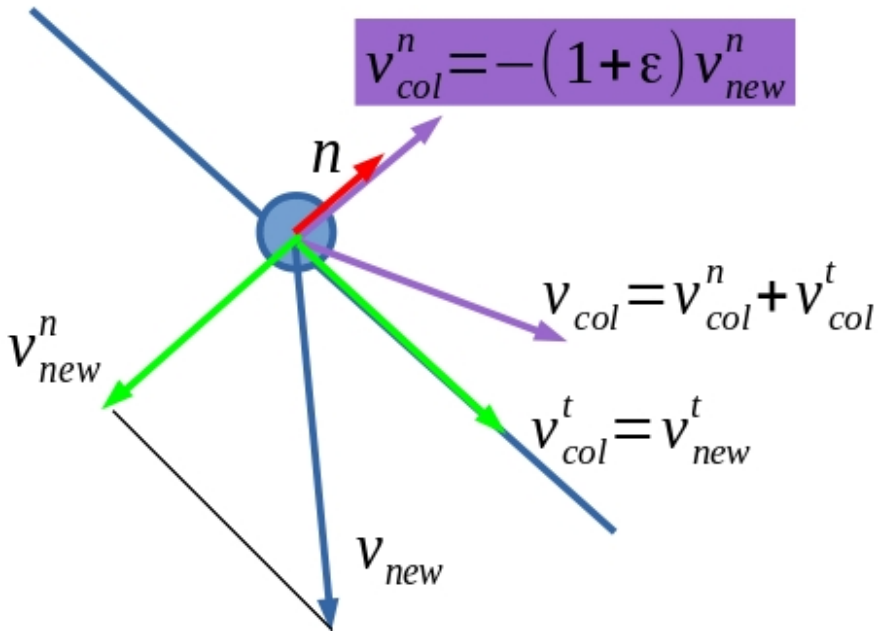
- La résolution statique consiste à uniquement considérer l'instant $t - \delta t$ (l'image d'avant) et l'instant t (l'image courante).
- Rappel : entre ces 2 positions, on considère que la particule se déplace en ligne droite et à vitesse constante qui est `v_new` (cf calcul du mouvement).
- On connait les positions `x_old` (la particule à l'instant $t - \delta t$) et `x_new` (la particule à l'instant t).
- Pour la détection : intersection entre le segment [`x_old`,`x_new`] et l'obstacle.



- La détection est dite statique car on ne se préoccupe pas de ce qui pourrait se passer entre les deux instants $t - \delta t$ et t (à opposer à une détection dite dynamique où on cherche explicitement l'instant de collision).

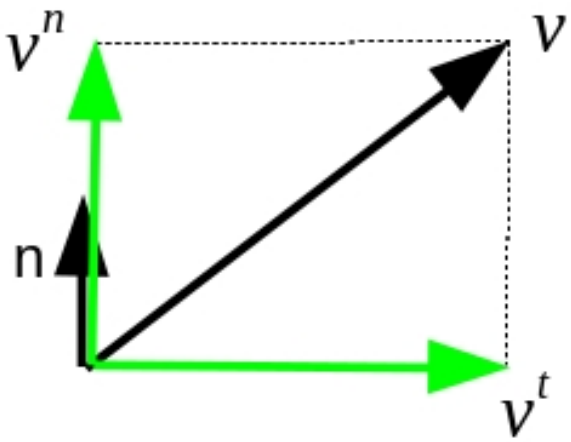
Réponse par impulsion

- Pour la réponse : on corrige `x_new` et `v_new` pour que cela corresponde à un "rebond". On notera ces valeurs corrigées `x_col` et `v_col`.
- Au moment de la collision, la vitesse `v_new` change instantanément :
  - La composante normale de la vitesse est opposée en lui appliquant un coefficient de restitution ε (à 1 la particule rebondit parfaitement ; à 0 la particule s'écrase).
  - La composante tangentielle de la vitesse reste inchangée.
- La composante normale de la vitesse ? (cf schéma)
- Il faudra donc déterminer la direction normale au point de collision : pour le moment on la suppose connue.

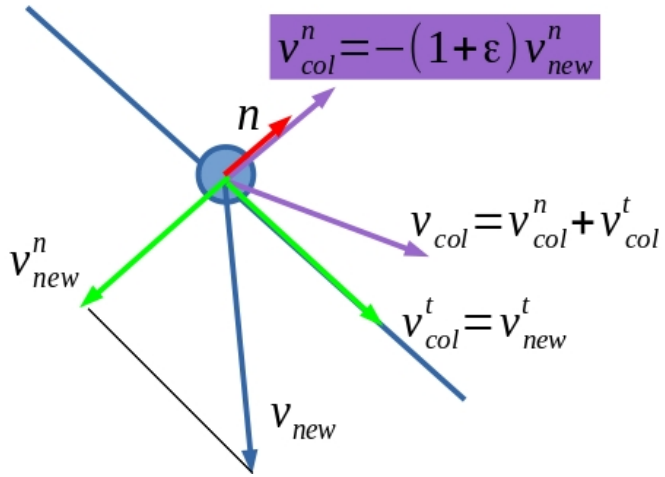


Composante normale/tangentielle ?

- On peut toujours décomposer une direction en une somme arbitraire $v = v^n + v^t$.
- On peut choisir v^n telle qu'elle corresponde à la projection de v sur la direction n.
- v^t est obtenue par $v^t = v - v^n$: cela correspond à la projection de v sur une direction t orthogonale à n.
- Comment calculer v^n ? Par le produit scalaire.



Composante normale/tangentielle de la vitesse

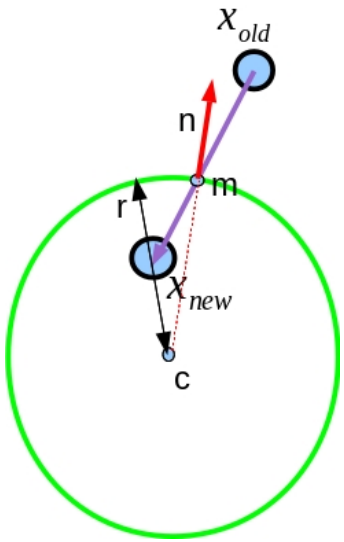


$v^n_{\text{new}} = (v_{\text{new}} \cdot n)n$ **Attention :** avec n unitaire

- Rendre n unitaire : $n' = \frac{n}{\|n\|}$
- Calcul de v_{Col} :
  - $v^n_{\text{col}} = -\varepsilon v^n_{\text{new}}$ et $v^t_{\text{col}} = v^t_{\text{new}} = v_{\text{new}} - v^n_{\text{new}}$
  - donc $v_{\text{col}} = v^n_{\text{col}} + v^t_{\text{col}} = v_{\text{new}} - (1+\varepsilon)v^n_{\text{new}}$
  - (i.e. on corrige v_{new} en lui retirant $(1+\varepsilon)v^n_{\text{new}}$).

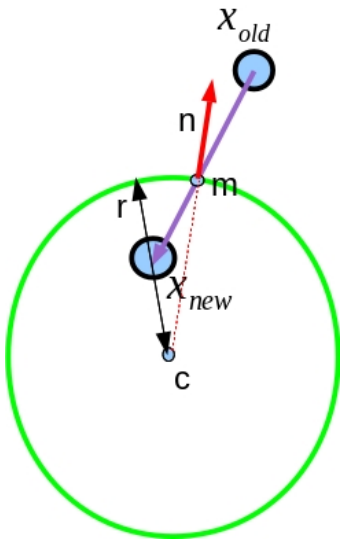
3 Détection de collision

Cercle



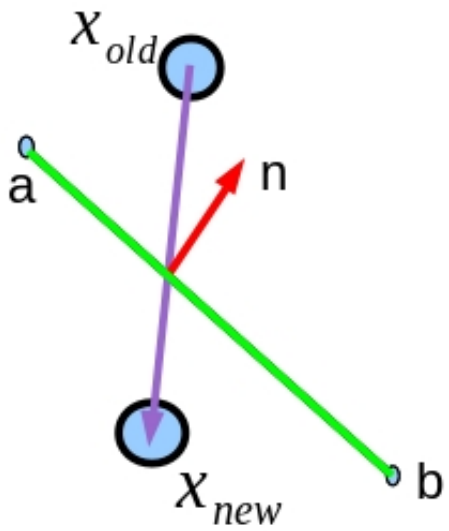
- Un cercle sera défini par son centre (une position $c = (x, y)$) et un rayon r (en pixels).
- Pour savoir s'il y a collision, on va simplement regarder si `x_old` est à l'extérieur (resp. intérieur) du cercle et `x_new` est à l'intérieur (resp. extérieur).
- **Attention :** ne pas oublier qu'il faut fournir la normale n au point de collision pour la réponse.
- A priori, pour déterminer cette normale, il faut donc déterminer le point d'intersection (point d'intersection entre 1 segment et 1 cercle : résolution d'une équation du 2nd degré).
- Mais on fait **une approximation** : on considère que l'intersection se fait sur `x_old` (relativement cohérent si les déplacements sont petits).

Cercle : calcul



- Un point P à l'intérieur ? Distance de P au centre plus petit que le rayon r (à l'extérieur sinon).
- Remarque : on évite de considérer les cas limites en informatique graphique car c'est rarement utile (par exemple ici : "sur" le cercle pourra être considéré "intérieur").
- Remarque : ce type de test est appelé **test de localisation** (*In/Out Test* ; qui se traduit par un booléen).
- Position de la collision ? Prendre `x_old`.
- Direction de la normale ? C'est la direction entre le centre du cercle et `m` (on peut la normaliser tout de suite).

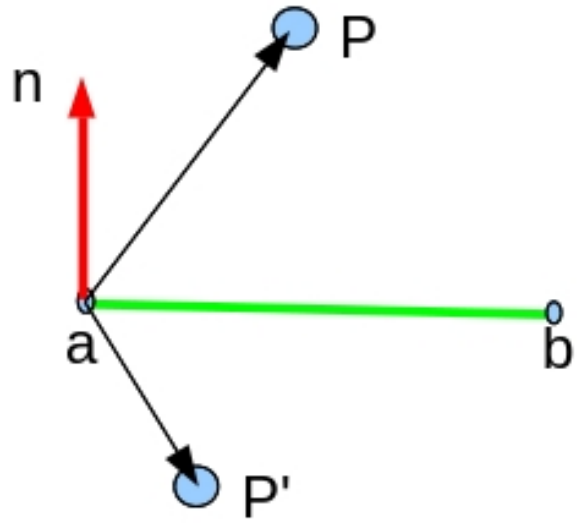
Segment



- Un segment est défini par ses 2 extrémités a et b.
- Pour savoir, s'il y a collision, on va "simplement" regarder si le segment $[a, b]$ coupe le segment [`x_old`,`x_new`].
- **Attention :** ne pas oublier qu'il faut fournir la normale au point de collision pour la réponse.
- Quelque soit le point d'intersection, c'est la normale au segment.
- La normale est la même pour tous les points du segment.

Segment : calcul (1/2)

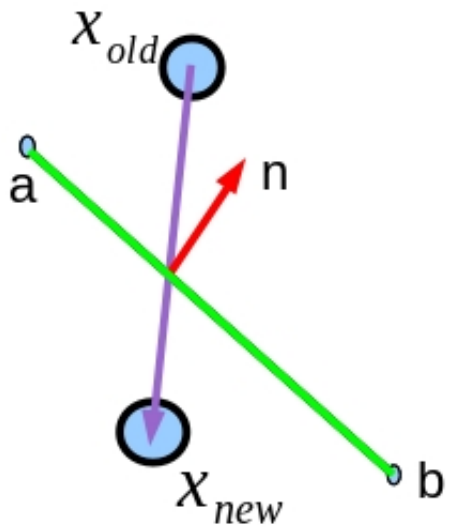
- Segment $[a, b]$ coupe [x_old,x_new] ? On s'appuie sur la localisation d'un point par rapport à une droite.



  - Si n est une normale d'une droite et a un point de cette droite : le point P est dit positif par rapport à la droite s'il est du même coté que la normale.
  - Par le calcul : c'est le signe de $(P - A) \cdot n$

Segment : calcul (2/2)

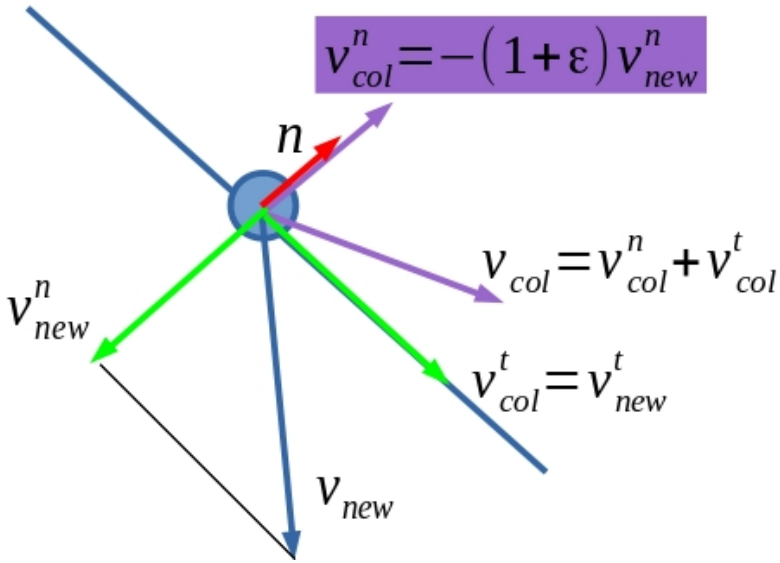
- Pour le segment, il y a intersection si
  - les signes de `x_old` et `x_new` par rapport à $[a, b]$ sont distincts (P et P' de part et d'autre de (ab)).
  - **Et** les signes de a et b par rapport à [`x_old`,`x_new`] sont distincts (a et b de part et d'autre de (PP'))



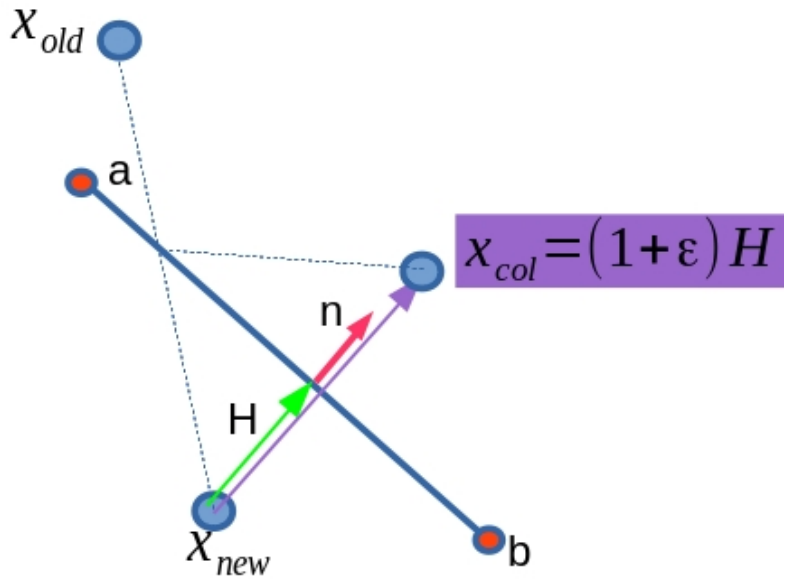
- Position de la collision ? en fait, n'importe quel point de $[a, b]$ suffira (**sera justifié plus loin**). Donc l'extrémité a convient.
- Direction de la normale ? une normale à $u = b - a$ est $n = (-u_y, u_x)$ (à normer).
- **Remarque :** inutile de calculer explicitement une intersection droite/droite (résolution d'équation).

Réponse à la collision : calcul

- Il reste à calculer explicitement la position corrigée `x_col` et la vitesse corrigée `v_col`.



Correction en vitesse



Correction en position

- Le vecteur H se calcule par projection ($H = ((a - x_{\text{new}}) \cdot n)n$ pour le segment : inutile de déterminer explicitement le point de collision).

4 Limitations et pour aller plus loin...

Restrictions

- Les obstacles sont statiques pour la simulation :
  - Ils ne sont pas mis en mouvement par des lois mécaniques (il faudrait, pour les segments, intégrer la mécanique des solides : intégration des moments)
  - Ils ne subissent pas d'impulsions : ils ne bougent pas du fait du choc par les particules (il faudrait également complexifier un peu le calcul d'impulsion).
- Les multi-collisions ne sont pas gérées :
  - La correction de la position et de la vitesse sont faites par un seul obstacle (à chaque image).
  - La correction peut remettre en situation de collision sur la même image (qui ne sera pas détectée).