MUX104
Synthèse d'image et réalité virtuelle

# Déformations

Pierre Cubaud
cubaud @ cnam.fr

mars 2020

le cnam

**Plan de la séance :**

1. **Collisions**
2. **Structures articulées**
3. **Cinématique inverse (IK)**
4. **Systèmes masse-ressort**

# 1. Collisions

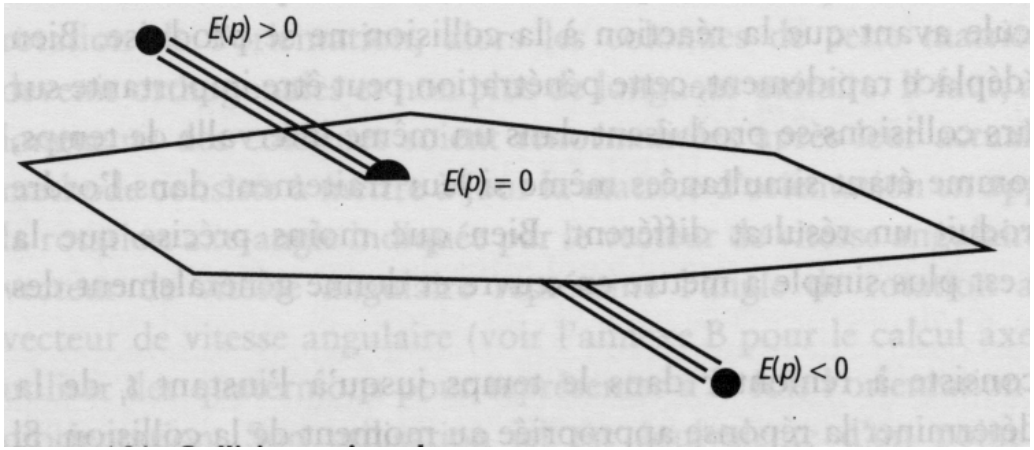
## 1.1 Détection de collisions

Exemple de la collision point-plan :

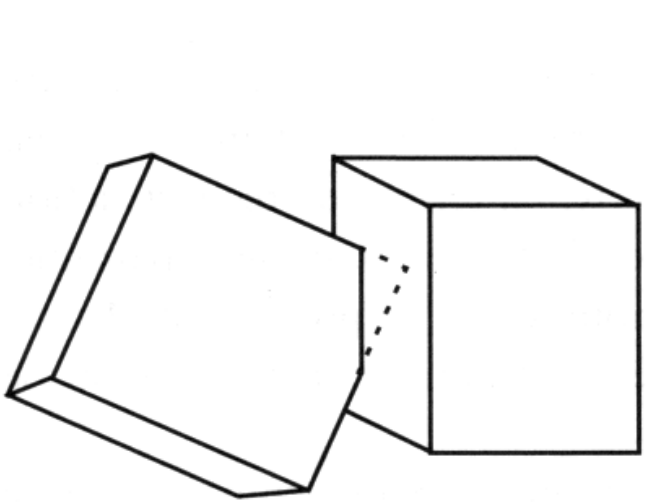
Equation du plan en 3D : Ax + By + Cz = D
Connaissant 3 points non alignés du plan, on a :

$$\begin{pmatrix} A \\ B \\ C \end{pmatrix} = \overrightarrow{P_0P_1} \otimes \overrightarrow{P_0P_2} \quad \text{et} \quad D = A.P_{0x} + B.P_{0y} + C.P_{0z}$$

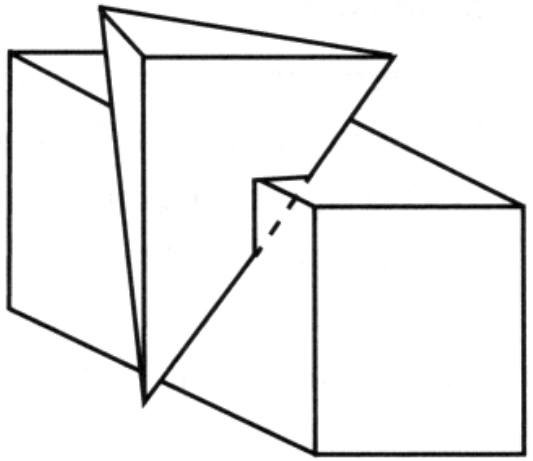
on étudie le signe de APx + BPy + CPz - D :



## Test sur des polyèdres



Sommet à l'intérieur d'un polyèdre



Pénétration d'un objet sans qu'un sommet d'un objet soit contenu dans l'autre objet

Figure 4.36 **Détection d'intersections de polyèdres** [PAR]

intersections droite / polygone : coûteux

=> voir cours « élimination parties cachées »

## Optimisation par sphères englobantes

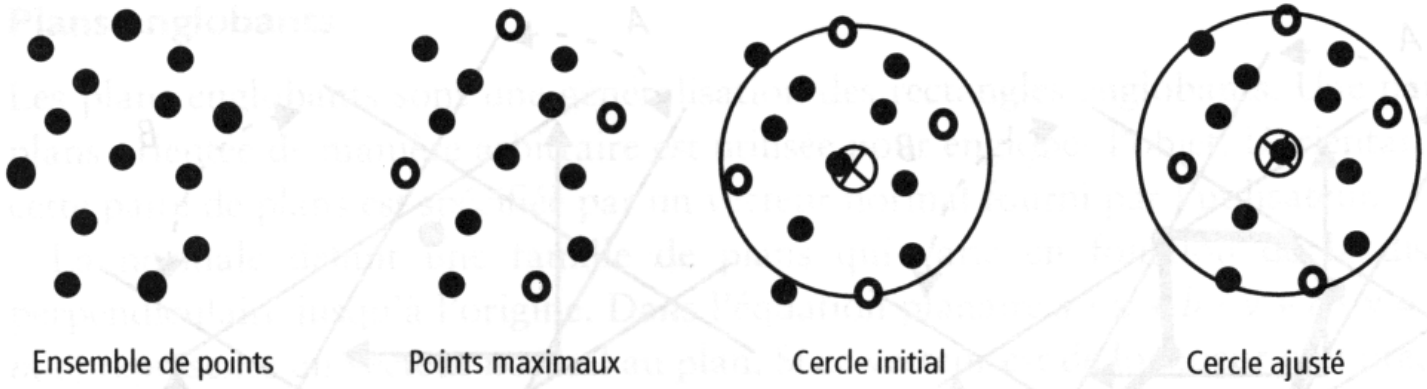


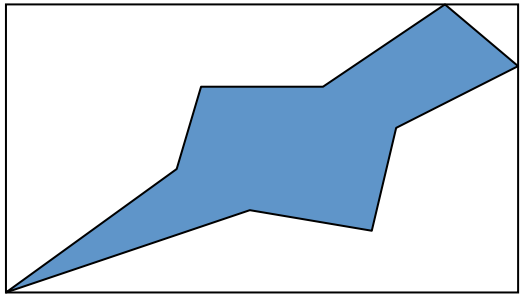

Figure B.20 **Calcul d'un cercle englobant pour un ensemble de points** [PAR]

-recherche des 6 valeurs extremes Xmin, Ymin, Zmin, Xmax, etc...
et des 6 points concernés iXmin, iYmin, iZmin, etc…
-trouver la paire de points la plus éloignée des 3 paires : P1, P2
- le centre de la sphère est C = (P1+P2)/2
son diamètre est la plus grande distance Dmax
- passer en revue tous les points P et ajuster la sphère si PC > Dmax

## Autres techniques de volumes englobants

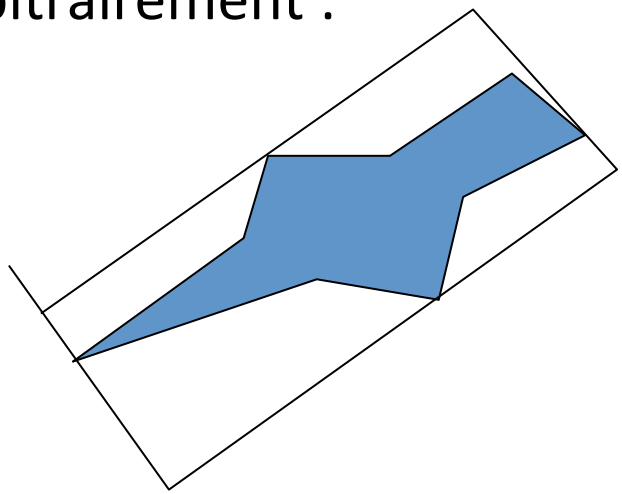
- boîte englobante alignée dans l'axe :

ABB : axis-aligned bounded box

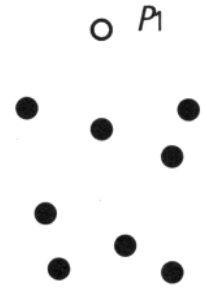


- boîte englobante orientée arbitrairement :

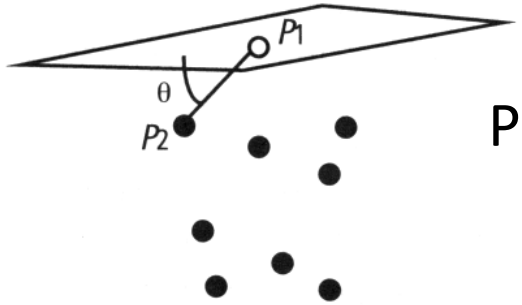
OBB : oriented bounded box



# Construction de l'enveloppe convexe (ex. de solution)



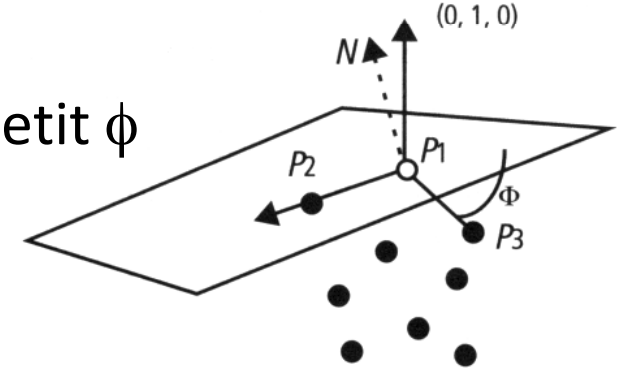

Étape 1 : Trouver le point le plus élevé



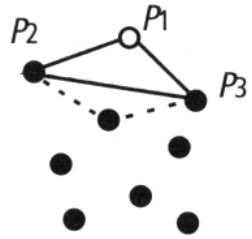

Étape 2 : Trouver une arête de l'enveloppe convexe

P2: plus petit θ

P3: plus petit ϕ




Étape 3 : Trouver un triangle initial




Étape 4 : Construire chaque triangle de l'enveloppe convexe en utilisant une ou deux arêtes existantes de triangles d'enveloppe convexe tracés antérieurement

algo. complet dans [PAR] p. 445

## 1.2 Réaction à la collision

On peut réagir aux collisions trop tard :

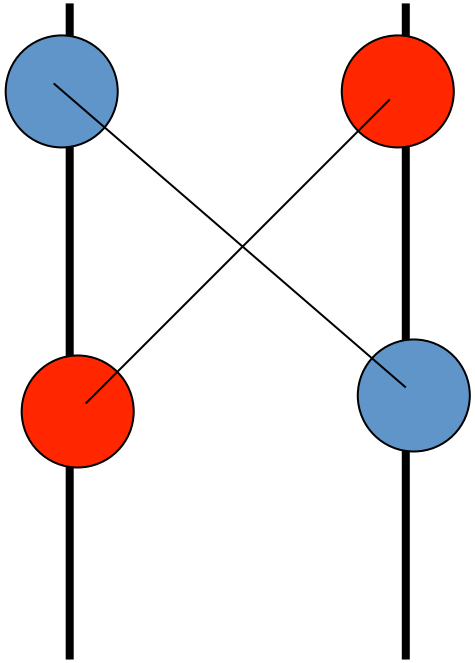


image #n
date T

#n+1
date T+dt

FAUX

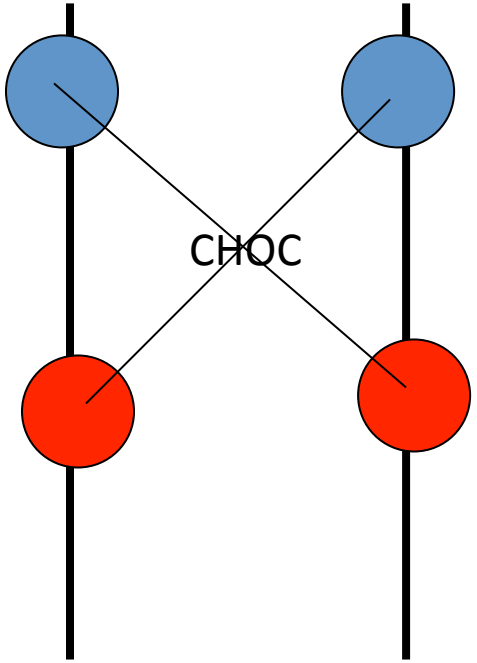



image #n
date T

#n+1
date T+dt

VRAI

## Méthode par calcul de la force d'impulsion

1- remonter le temps au point d'impact

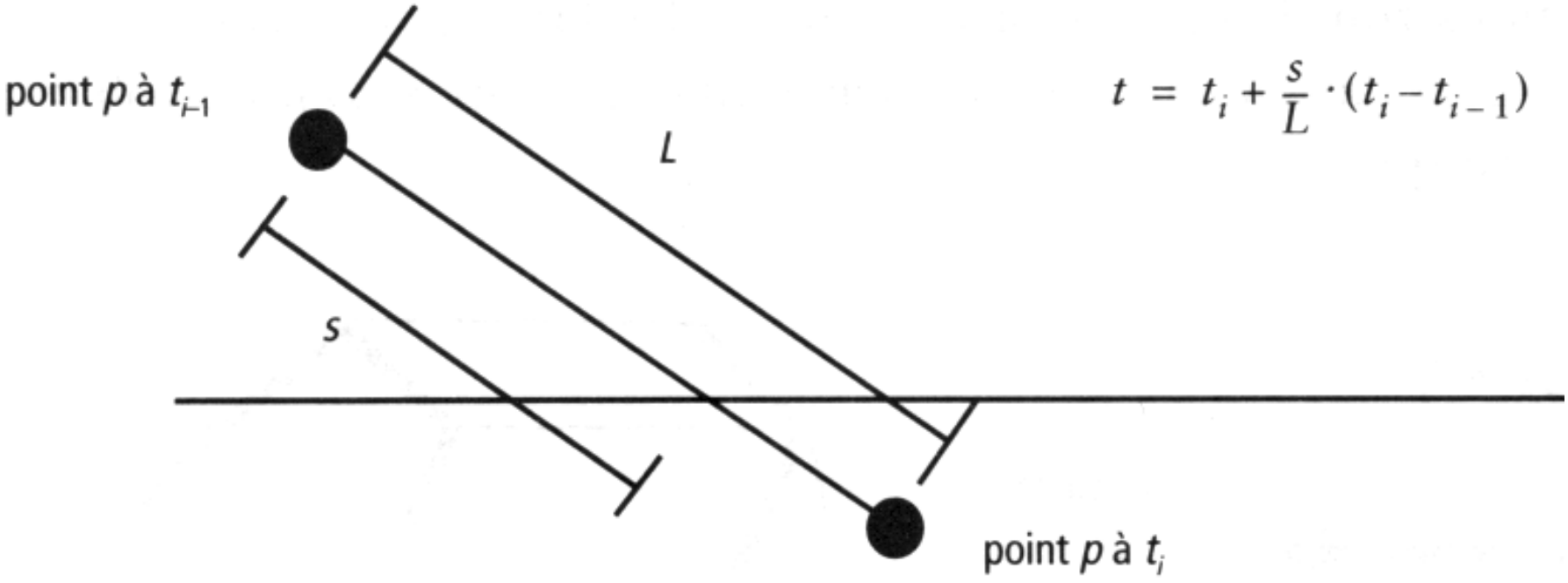
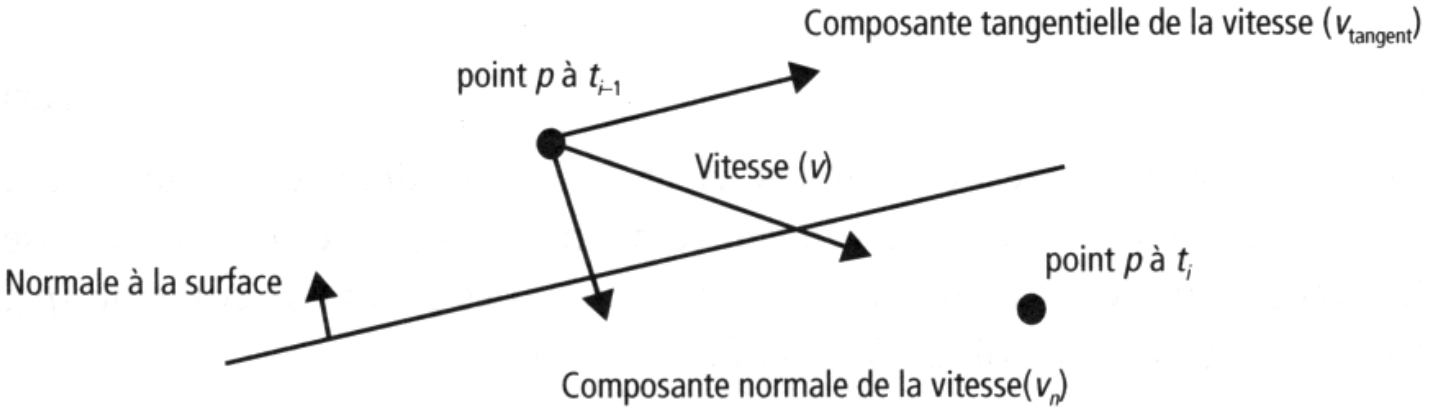


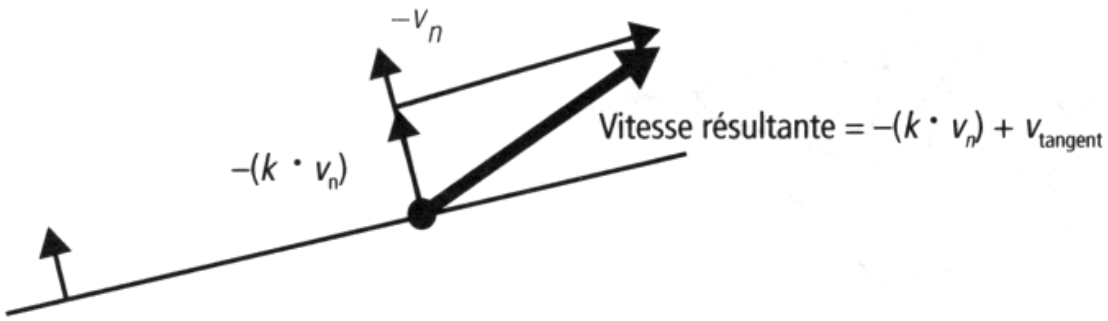

Figure 4.38 **Estimation linéaire de l'instant d'impact, t**

[PAR]

# 2- calculer les conséquences de l'impact




Composantes de la vitesse d'une particule entrant en collision avec un plan




Calcul de la vitesse résultant d'une collision ($k$ est le coefficient de restitution)

Figure 4.39 **Réponse à l'impact entre un point et un plan** [PAR]

# Un état de l'art sur les logiciels de détection de collision

Tangi.Meyer@irisa.fr
Guillermo.Andrade@irisa.fr

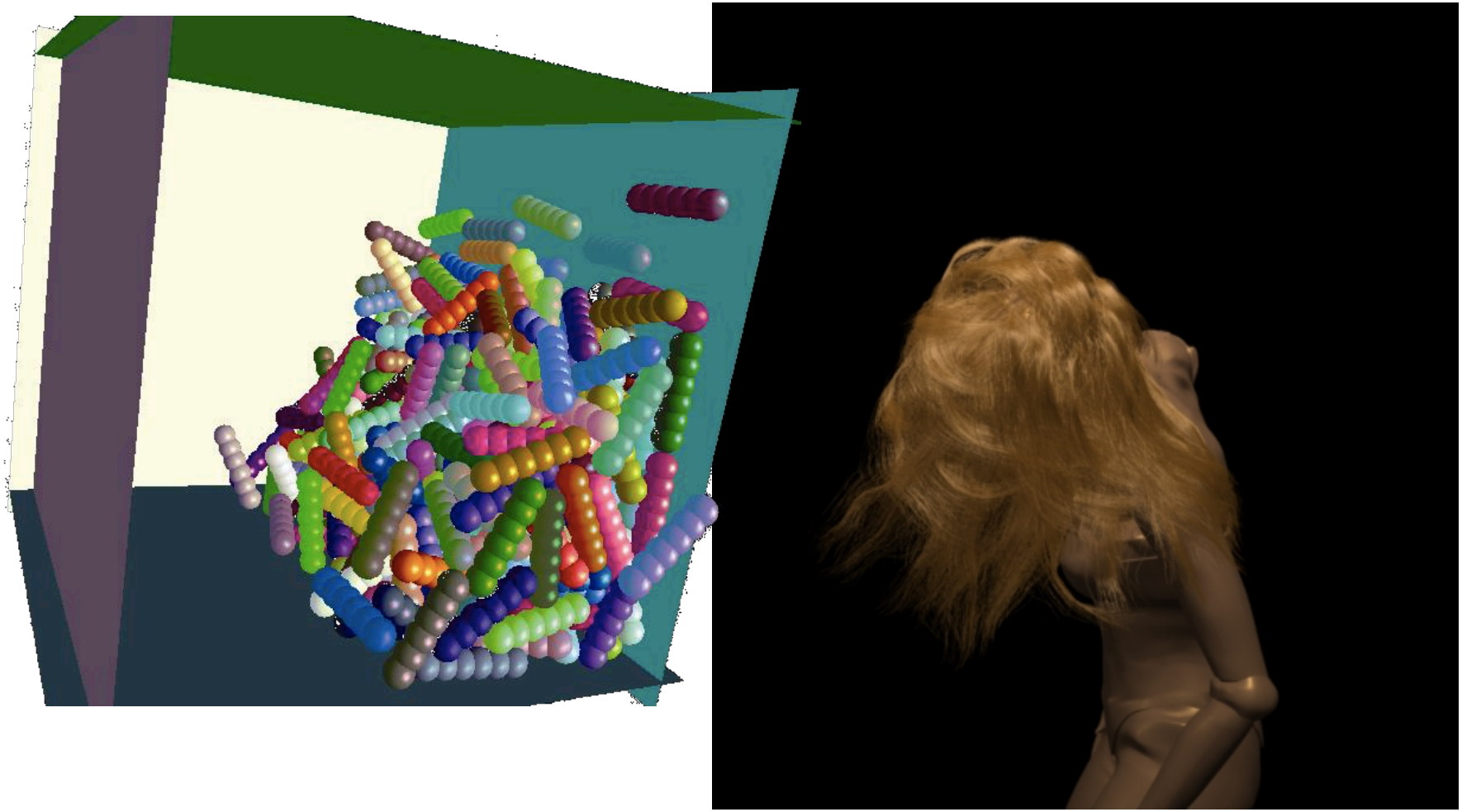
Présenté par Jean-Marie Souffez

IRISA SIAMES

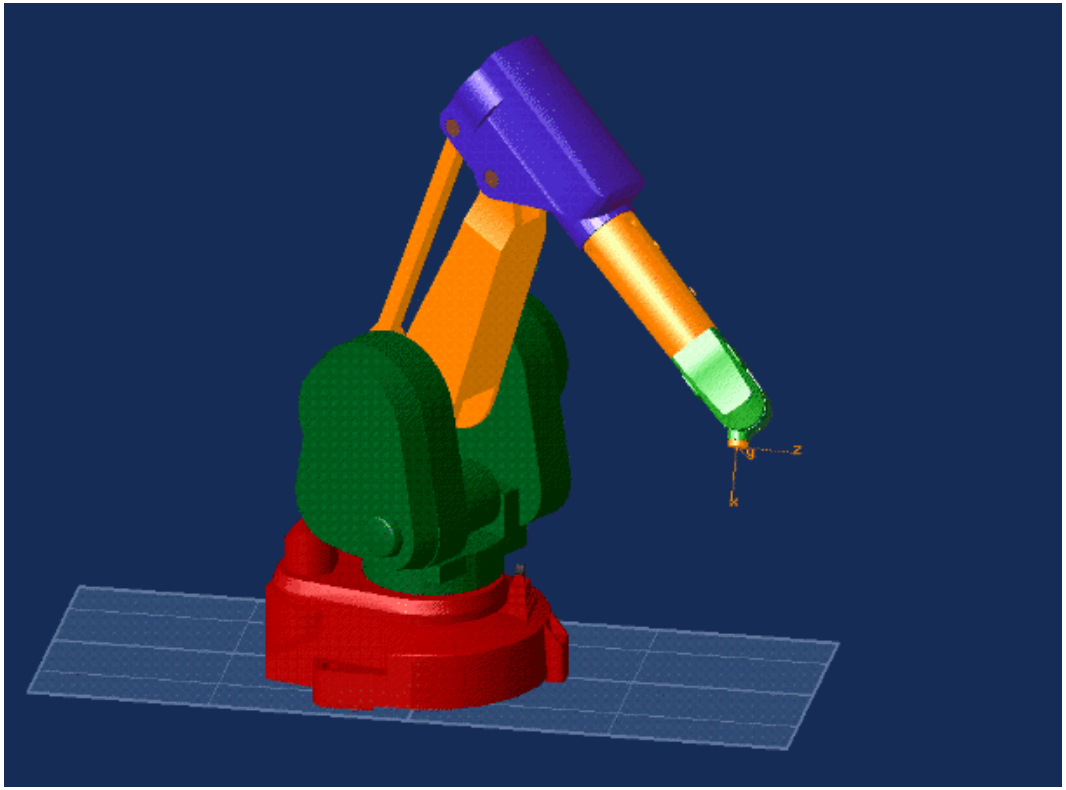
| *Outils* | *Méthodes* | *Entrées* | *Sorties* | *Principes* |
|---|---|---|---|---|
| RAPID | OBB-Trees | Soupe de Polygones | Détection / paires intersectantes | 2Body |
| VCollide | Surcouche de Rapid | Soupe de Polygones | Détection / paires intersectantes | Nbody - Cohérence temporelle |
| ICollide | Diagramme de Voronoi | Polyhèdres convexes | Distance / paires | Nbody - Cohérence temporelle |
| Solid | ABBTrees | Soupe de Polygones | Point de contact | Nbody - Cohérence temporelle |
| Quick_CD | K-Dops trees | Soupe de Polygones | Collision entre triangles | 2Body |
| Swift | Voronoi+Multilevel model | Polyhèdres convexes | Distance avant collision | Nbody - Cohérence temporelle |
| V-Clip | Voronoi | Polyhédres convexes | Distance avant collision | NBody |
| VPS | Voxel+ Intersections géo. | Nuage de points | Pénétrations points-voxels | 2Body |
| Hcollide | Spatial décomp.+OBBTrees | Polyhèdres | Pénétrations points-surfaces | Cohérence Temporelle |
| Contact | OBB-Trees | Soupe de Polygones | Point de contact | 2Body - détection continue |
| ColDet | OBB-Trees | Soupe de Polygones | Collision exacte | 2Body |
| | | | | |

A consulter : présentation de O. GALIZZI (IMAG/iMAGIS)

« Animation de solides en contact par modèle physique »



# 2. Structures articulées



A LIRE : cours de robotique de J. Gangloff (DEA Strasbourg)
http://eavr.u-strasbg.fr/library/teaching/robotics/

# 2.1. Description

voir « techniques de l'ingénieur »

## Mécanique générale

### Cinématique générale

*par* **Jean-Pierre BROSSARD**
*Professeur de Mécanique à l'Institut National des Sciences Appliquées (INSA) de Lyon*

## 3.6 Degré de liberté

C'est une question qui n'a pas de réponse unique car les points de vue peuvent être différents et, de toute façon, ce n'est pas au fond une question d'une grande utilité pratique en cinématique et en mécanique générale comme nous le montrerons.

Le problème se formule ainsi : soit un système à $n$ paramètres $q_1 ..., q_i ..., q_n$ et supposons qu'il y ait $m$ relations de liaison indépendantes.

On appelle degré de liberté le nombre $k$ :

$$k = n - m$$

Que peut-on dire de ce nombre $k$ ? Pas grand-chose avec sûreté sans une étude approfondie.

## Les différents types de liaisons

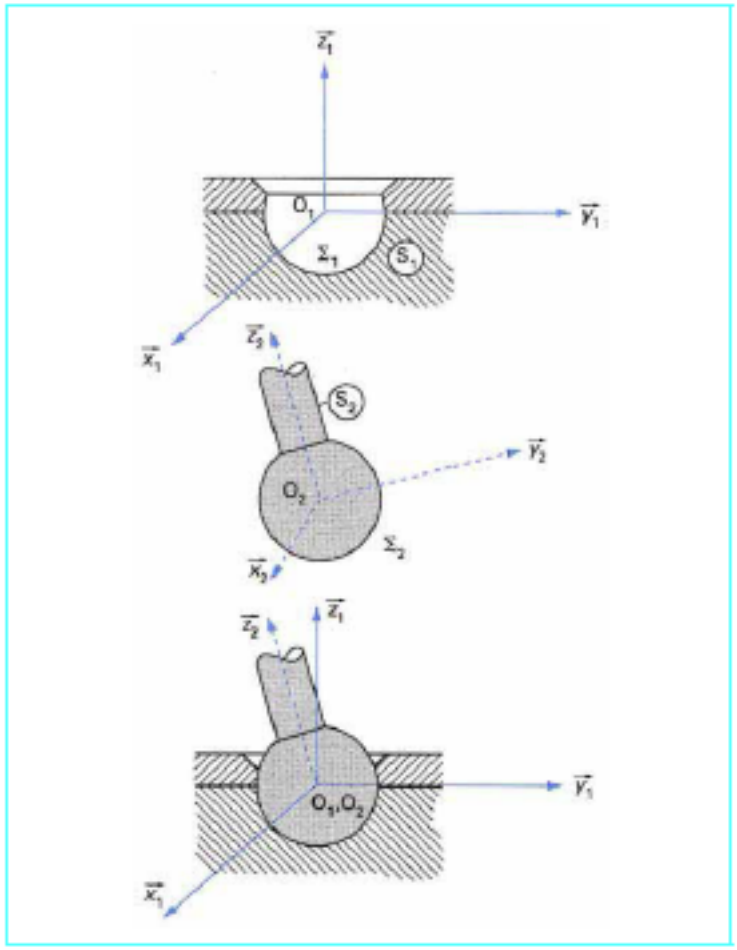


Figure 65 – Liaison sphérique

k=3 (rotations)

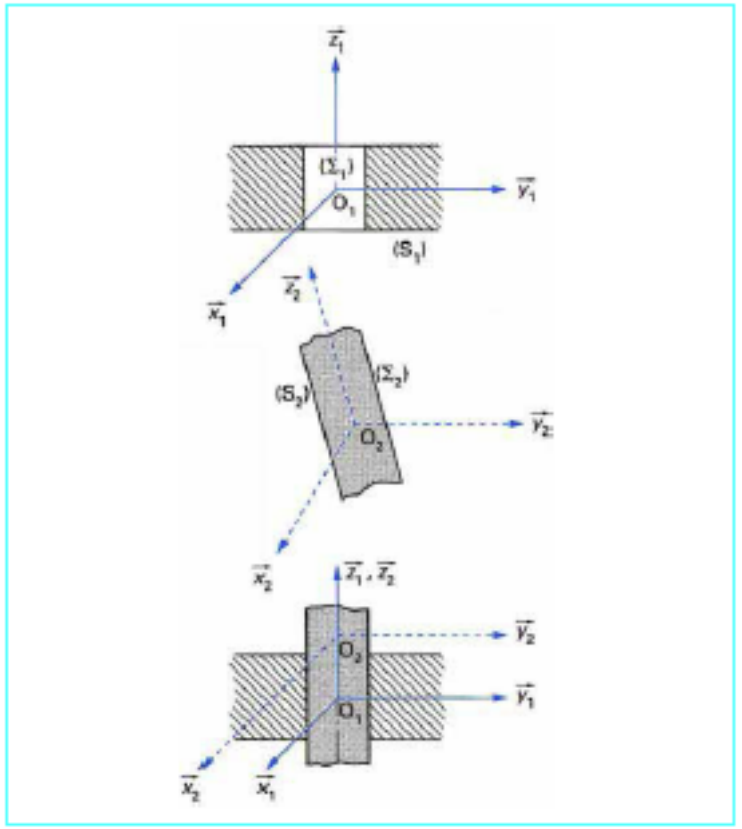


Figure 69 – Liaison cylindrique

k=2 (translation ou rotation)

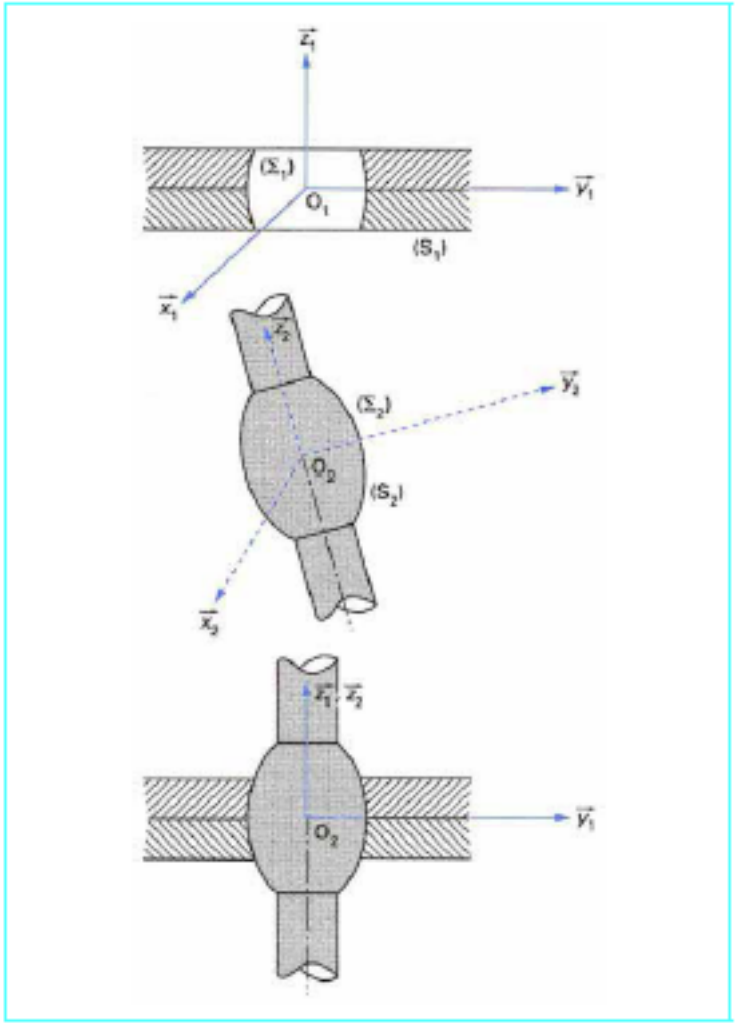



Figure 72 – Liaison rotoïde

k=1 (rotation)

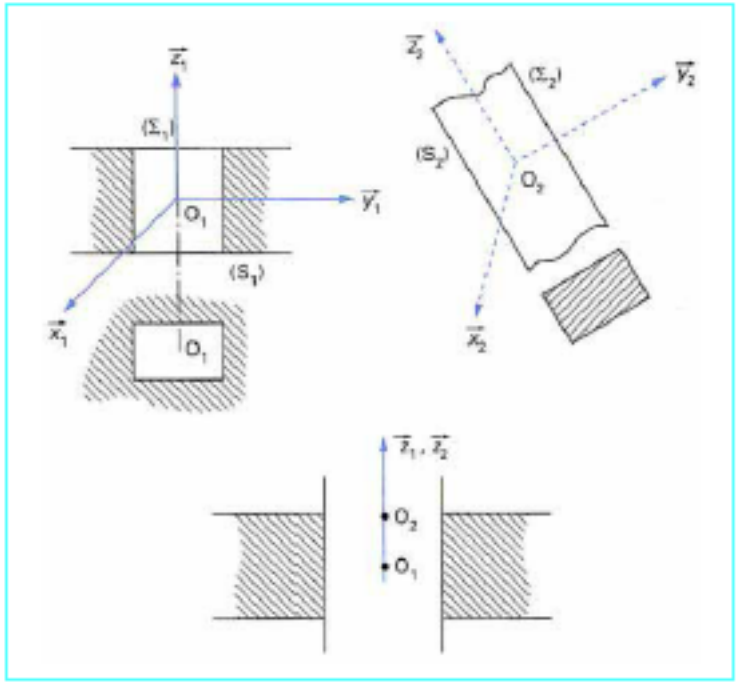



Figure 75 – Liaison prismatique

k=1 (translation)

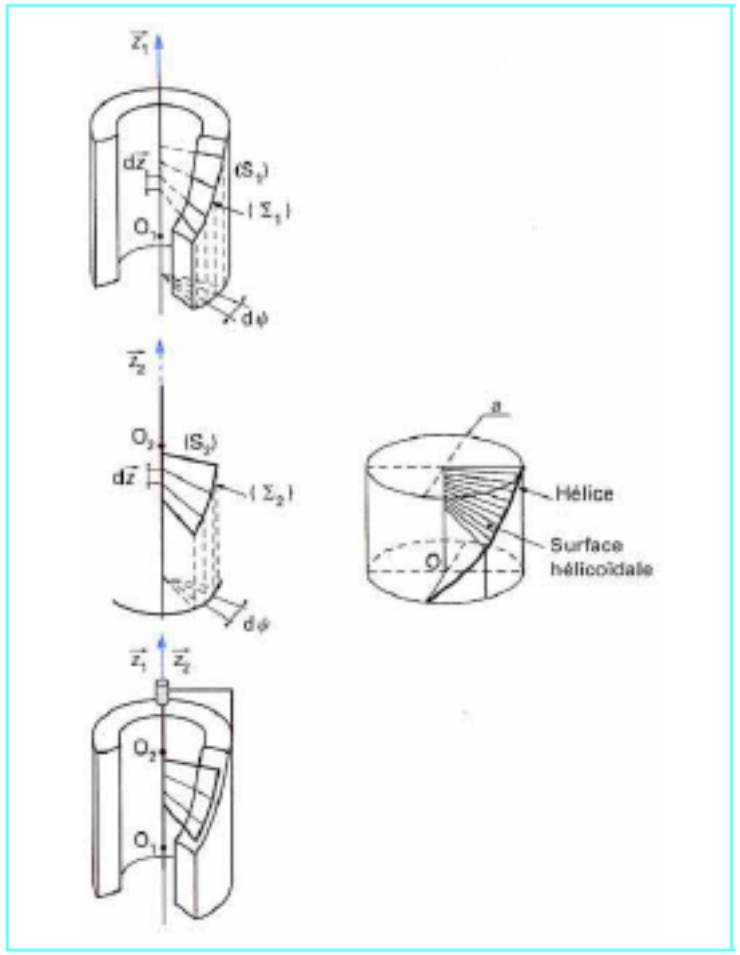



**Figure 79 – Liaison hélicoïdale**

k=1 (rotation ET translation)

### 3.7.7 Examen du degré de liberté d'un système

De nombreux auteurs ont essayé de déterminer le degré de liberté d'un mécanisme par simple décompte des liaisons et du nombre d'équations de liaison que chacune d'elles implique. Malheureusement, ces méthodes n'ont aucune sûreté. Elles n'arrivent en général qu'à prévoir des résultats... connus. Seules les méthodes basées sur les équations de liaison et la détermination de leur indépendance ont une valeur sûre.

#### 3.7.7.1 La méthode ne permet pas de prévoir les degrés de liberté

**Exemple 1** (figure 82*a*) : comme indiqué sur la figure, c'est un **système plan à 4 barres**. Nous avons vu que, pour chaque solide en mouvement plan, il fallait trois paramètres. Il n'y a que 4 liaisons rotoïdes ; chaque liaison rotoïde impose deux relations (un point fixe dans le plan). Le degré de liberté est donc $k = 3 \times 3 - (4 \times 2) = 1$, ce qui est bien conforme à ce que l'on trouverait par un examen détaillé des équations de liaison du système.

**Exemple 2** (figure 82*b*) :

$$k = 4 \times 3 - 6 \times 2 = 0$$

Le système est bloqué, ce qui est bien conforme à ce que donnerait une étude détaillée.

**Exemple 3** (figure 83) : il y a le même nombre de liaisons que dans l'exemple précédent. Il est aisé de voir que cette fois le degré de liberté est 1.

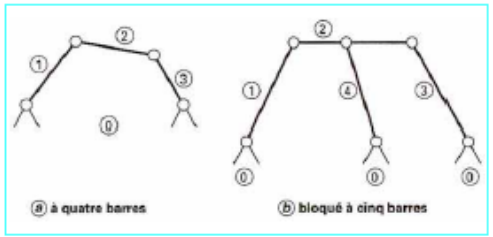



**Figure 82 – Systèmes plans**

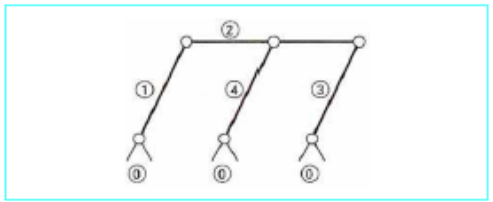


**Figure 83 – Système plan à cinq barres, totalement mobile**

## 2.2 Cinématique directe

$$\vec{X} = f(\vec{\theta}) = f(\theta_1, \theta_2, \cdots, \theta_N)$$

$$\theta_i = (x, y, z, \mu, \phi, \Phi)$$

Exemple d'un bras manipulateur plan :

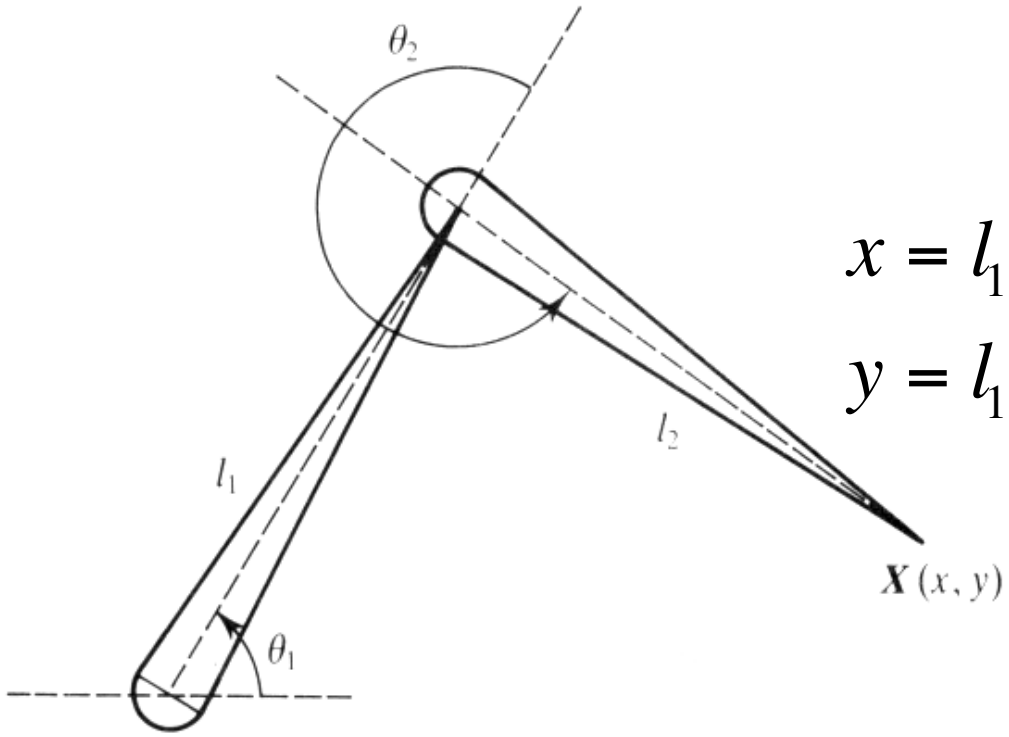



**Figure 16.2** A simple two-link structure. [WATT]

$$x = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2)$$

$$y = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2)$$

# **Notation DH** (Denavit-Hartenberg, 1955)

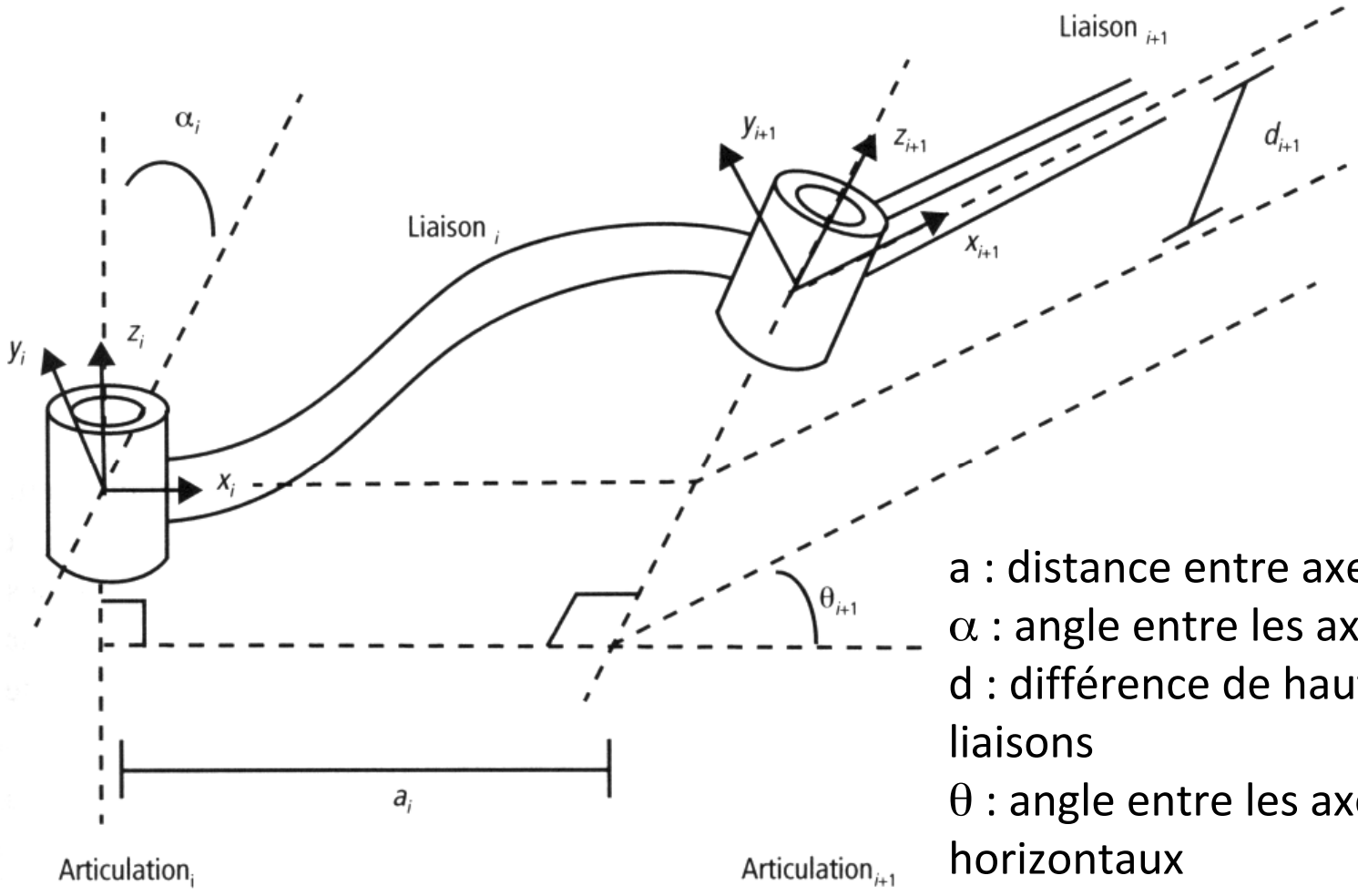



Figure 4.12 **Paramètres de Denavit-Hartenberg**

a : distance entre axes des liaisons
$\alpha$ : angle entre les axes verticaux
d : différence de hauteur des liaisons
$\theta$ : angle entre les axes horizontaux

[PAR]

Calcul des coordonnées de la liaison L(i) dans le repère de L(i-1) :

$$M = Tx(a) \bullet Rx(\alpha) \bullet Tz(d) \bullet Rz(\theta)$$

$$M = \begin{pmatrix} \cos\theta & \sin\theta\cos\alpha & \sin\theta\sin\alpha & 0 \\ -\sin\theta & \cos\theta\cos\alpha & \cos\theta\sin\alpha & 0 \\ 0 & -\sin\alpha & \cos\alpha & 0 \\ a & -d\sin\alpha & d\cos\alpha & 1 \end{pmatrix}$$

Calcul des coordonnées de la liaison L(i) dans le repère de la base :

$$M = M(0,1) \bullet M(1,2) \bullet \ldots \bullet M(i-1,i)$$

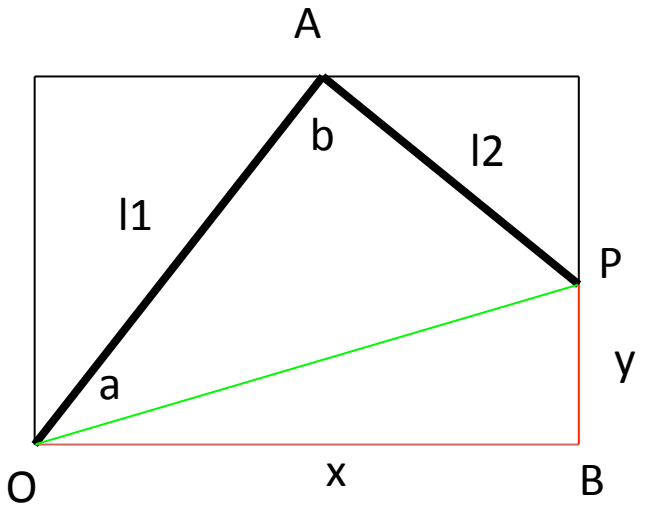
# 3. Cinématique inverse

**Problème inverse !**

$$\vec{\theta} = f^{-1}(\vec{X})$$

Reprise de l'exemple :

-recherche angle <BOP>= θT
-recherche angle <POA>= θ1-θT=a
-recherche angle <OAP>=π-θ2=b



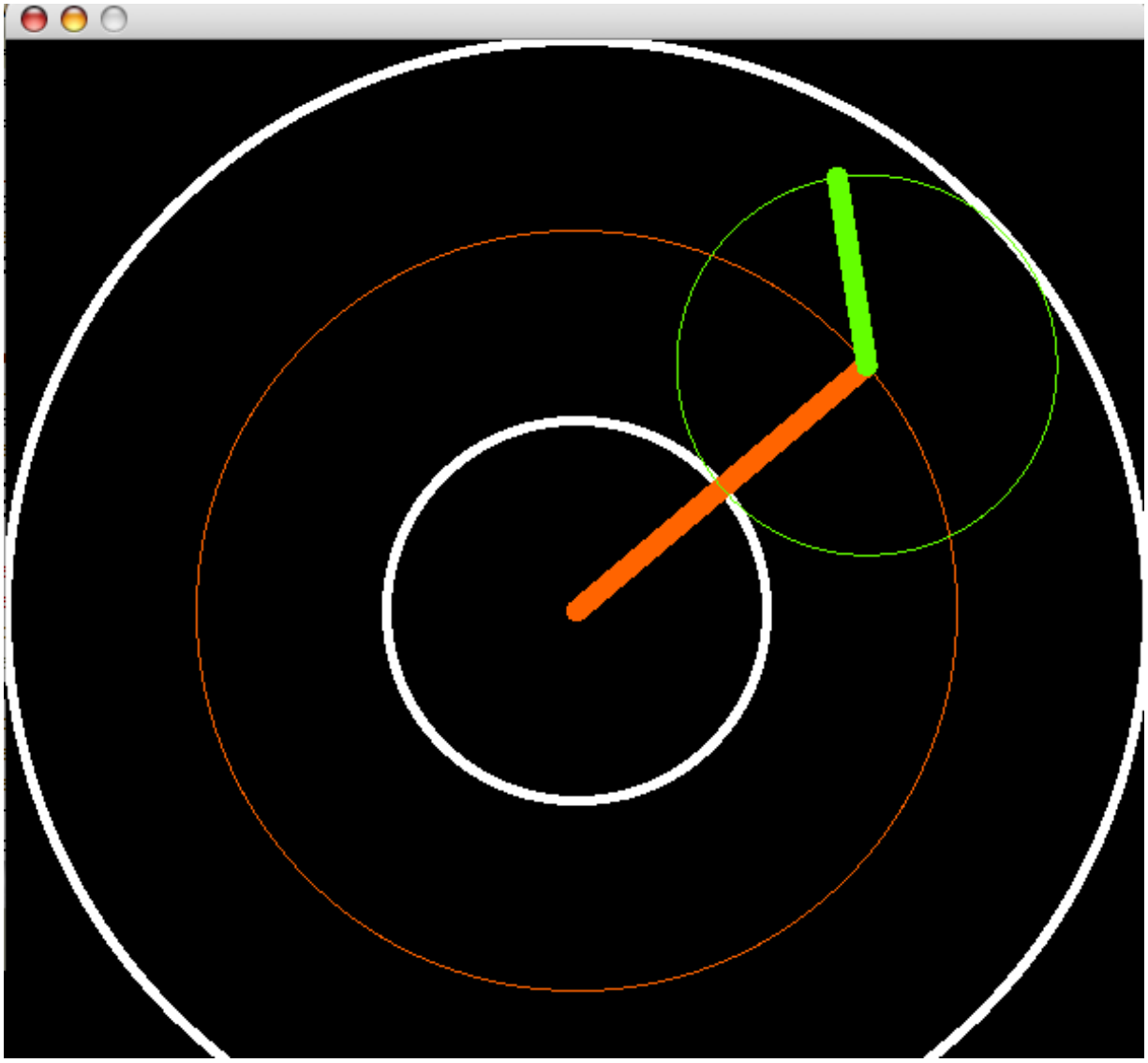

$$tg\theta_T = BP/OB = y/x \Rightarrow \theta_T$$

$$\cos a = \frac{OP^2 + OA^2 - AP^2}{2|OP||OA|} = \frac{d^2 + l_1^2 - l_2^2}{2dl_1} \Rightarrow \theta_1$$

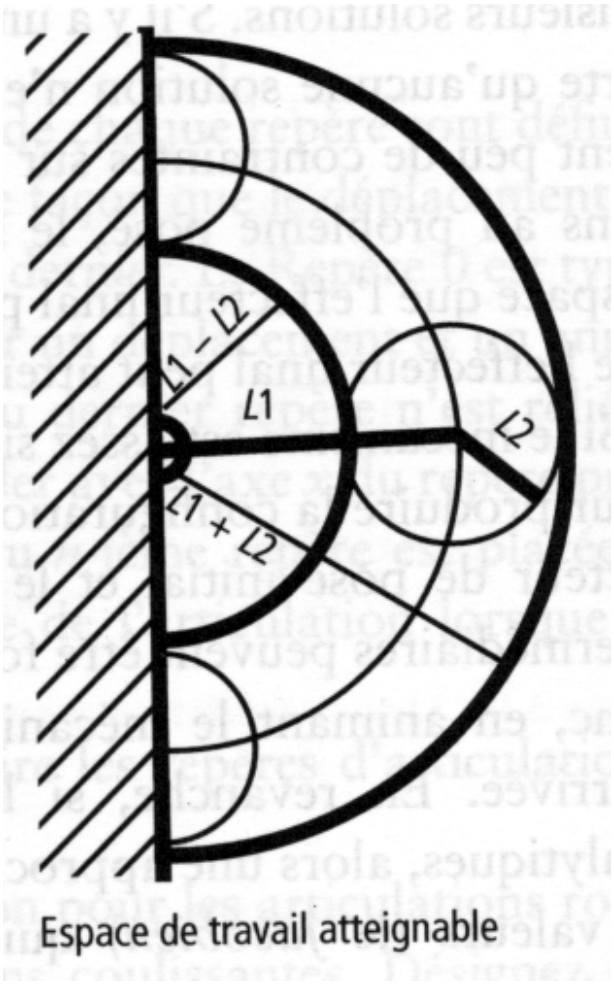
$$\cos b = \frac{AP^2 + AO^2 - OP^2}{2|AP||AO|} = \frac{l_2^2 + l_1^2 - d^2}{2l_1 l_2} \Rightarrow \theta_2$$

avec $d^2 = x^2 + y^2$

Réalisation avec Processing : cineminverse



**Problèmes :**




Espace de travail atteignable

[PAR]

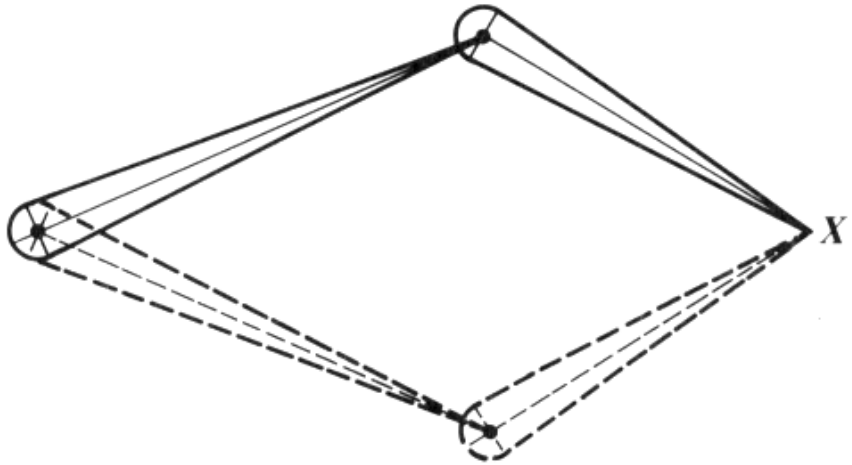
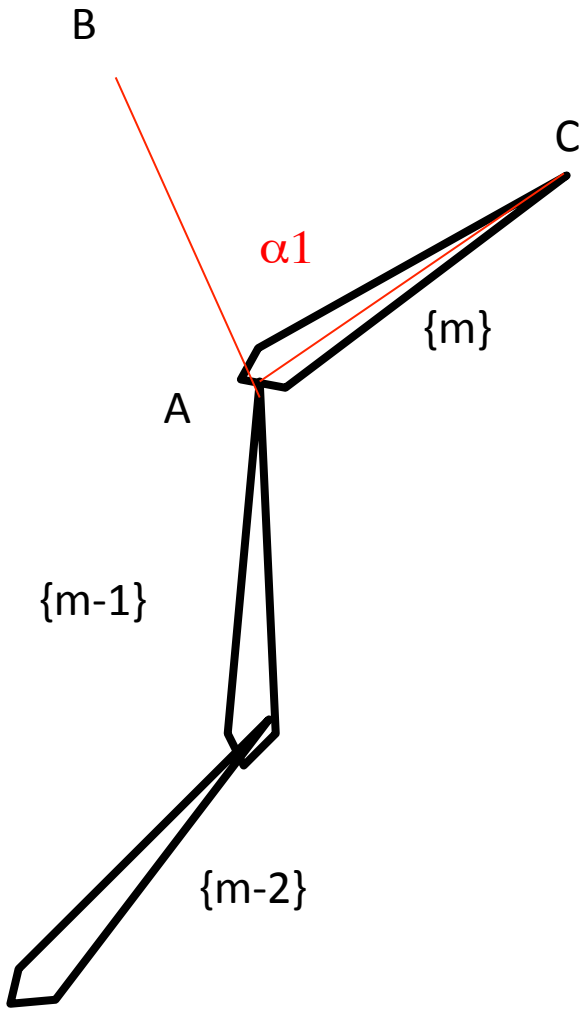



**Figure 16.3** Two solutions for a two-link mechanism to position the end effector at ***X***.

[WATT]

# Une heuristique : méthode CCD (Cyclic Coordinate Descent)

Chris Welman. *Inverse Kinematics and Geometric Constraints for Articulated Figure Manipulation*.
http://fas.sfu.ca/pub/cs/theses/1993/ChrisWelmanMSc.ps.gz B.Sc, SFU, 1989

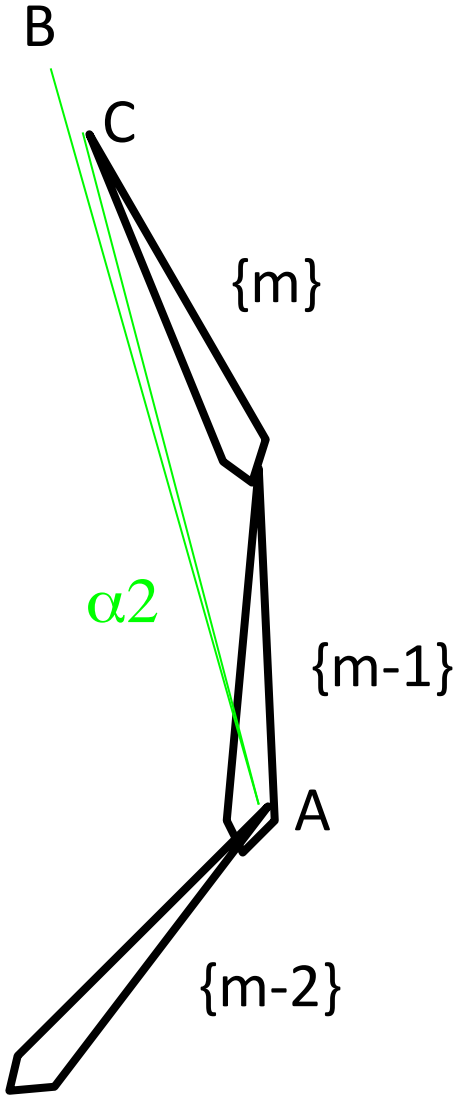


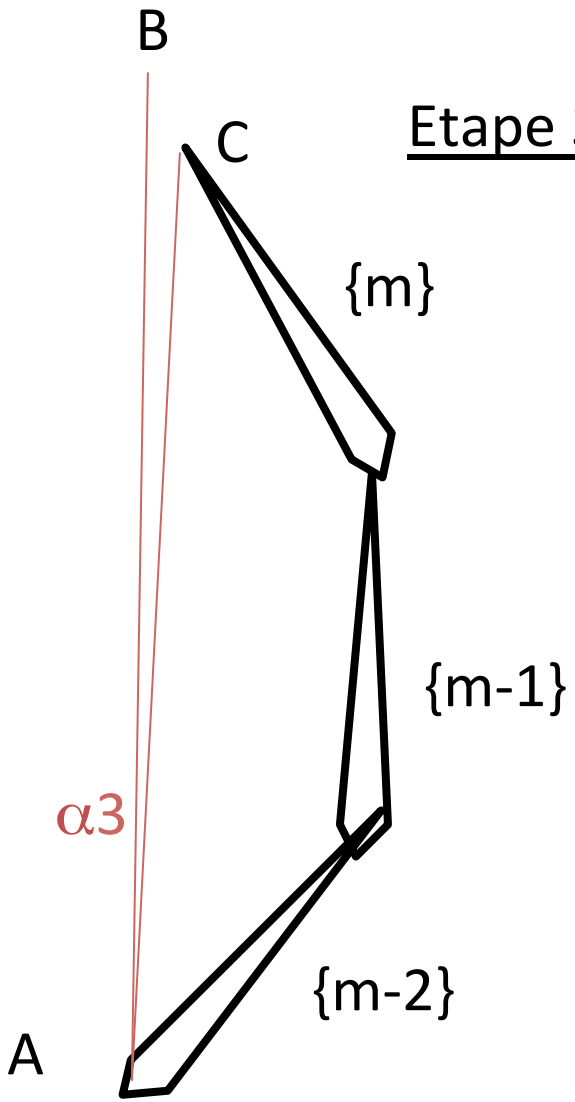

Etape 1 : approcher l'effecteur (en C) du but (en B) en calculant l'angle
$\alpha 1 = \langle CAB \rangle$

-calcul de l'angle par le produit scalaire de AC par AB

-calcul du sens de rotation par le produit vectoriel de AC et AB (signe de la composante en z)

Etape 2 : approcher le bras {m-1}
de B en calculant l'angle $\alpha 2$ = <CAB>

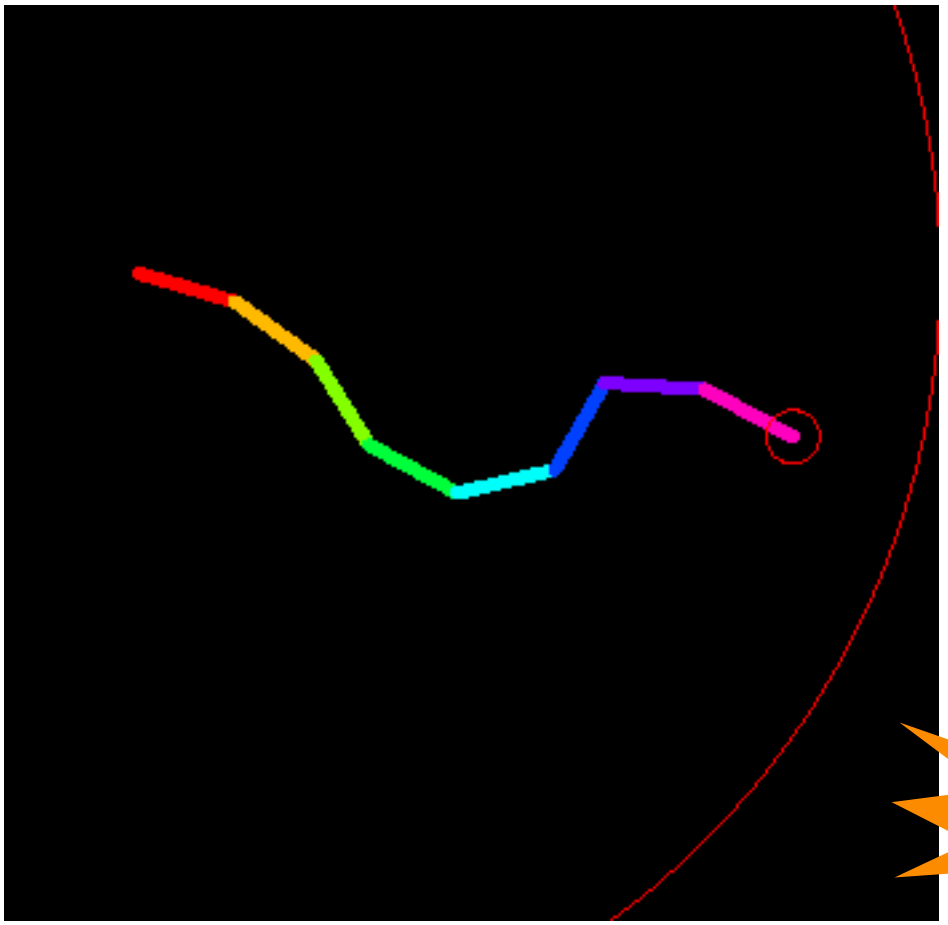



Etape 3 :idem avec {m-2}

etc...
on s'arrête si
|BC| < seuil

# Réalisation avec Processing : ccdV2



demo

Exemple d’utilisation: le « politicien » de Ken Perlin


http://mrl.nyu.edu/~perlin/experiments/fiend/

Allez voir le site de K. Perlin (NYU) !

**Ken Perlin**
Professor of Computer Science
NYU Future Reality Lab

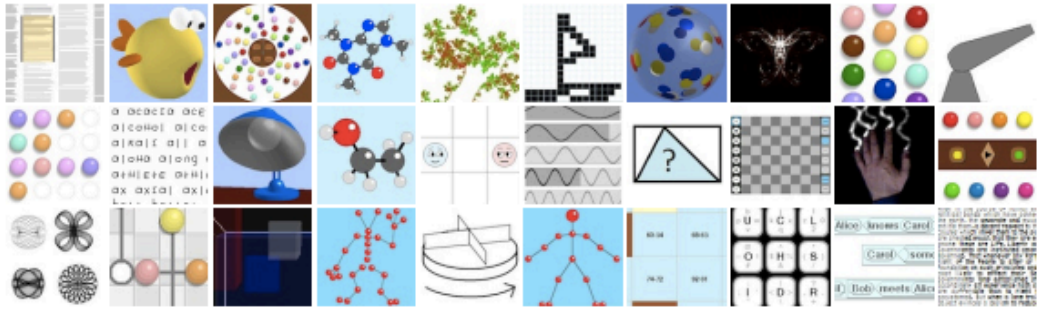
60 Fifth Ave, 3rd floor,
NY, NY 10003

Member of MAGNET
***email:*** last name **at** cs **dot** nyu **dot** edu
***start-ups:*** tactonic holojam parallux
***blog:*** blog.kenperlin.com
vita,bio,courses,alligators

©A.M.P.A.S.®

*``Que não seja Imortal, posto que é Chama. Mas que seja Infinito enquanto dure...''*

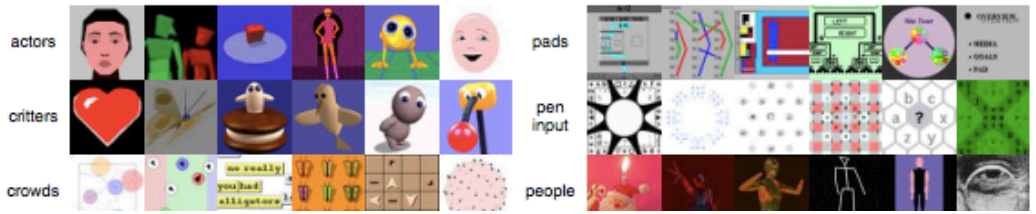
**SOME TOYS FROM THE BLOG:**



**OTHER EXPERIMENTS:**

***If the Java applets below don't run, try this:***

1. Download the latest Java runtime by clicking here;
2. Add this site as a "trusted site" to your computer by following these instructions.




https://mrl.nyu.edu/~perlin/

# 4. Systèmes masse-ressort

(wikipedia)

## Loi de Hooke pour les ressorts [ modifier | modifier le code ]

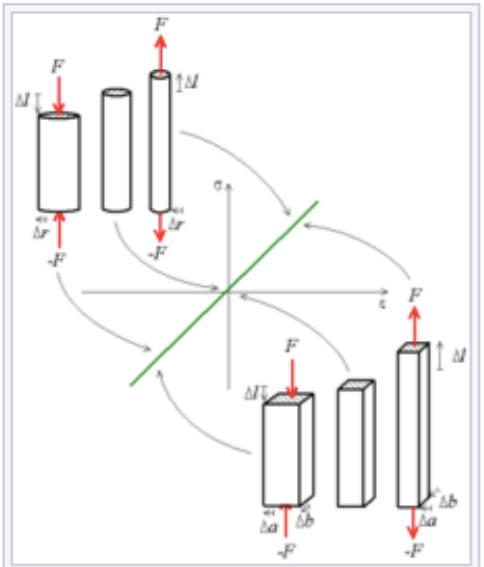
Le mode de déformation le plus simple est la traction (étirement) ou la compression selon un axe. Pour de petites déformations, la variation de longueur $\Delta\ell$ est proportionnelle à la force de traction/compression $\mathrm{F}$ :

$$\Delta\ell \propto \mathrm{F}$$

que l'on écrit plus volontiers :

$$\mathrm{F} = -k\,\Delta\ell$$

où $k$ est la raideur de la pièce, aussi appelée constante de rappel. C'est en fait la loi des ressorts. Ici, le signe négatif signifie que la force s'oppose donc à toute déformation, et est donc de sens opposé à la déformation du ressort.



Cylindre soumis à de la traction/compression

## Équation du mouvement d'une particule soumise à la force de Hooke en une dimension [ modifier | modifier le code ]

L'équation différentielle qui traduit l'action de la force de Hooke sur la particule peut s'écrire :

$$m\ddot{x} = -kx$$

où $m$ est la masse de la particule, $x$ est la position de la particule par rapport à son point d'équilibre (i.e. le point où il n'y a pas de force qui s'exerce sur la particule) et $\ddot{x}$ est l'accélération de la particule. Une solution de cette équation peut s'écrire :

$$x = A\cos(\omega t + \delta)$$

où $A$ et $\delta$ sont des paramètres déterminés par les conditions initiales du système et représentent l'amplitude du mouvement et une phase sur l'oscillation, respectivement. Ici, $\omega$ est la fréquence angulaire et vaut $\sqrt{\frac{k}{m}}$ et n'est donc dictée que par les caractéristiques du système.

Dans tout système réel, une partie de l'énergie totale est dissipée, le plus souvent en chaleur, ce qui crée une force d'amortissement.

En mécanique, celle-ci dépend de la vitesse du corps. Dans de nombreux cas, on peut supposer que le système est linéaire, l'amortissement étant alors proportionnel à la vitesse (voir Système oscillant à un degré de liberté).

En électricité, l'amortissement désigne l'effet résistif d'un circuit RLC.

On définit le coefficient d'amortissement *c* par :

$$\mathbf{F} = -c\mathbf{v}.$$

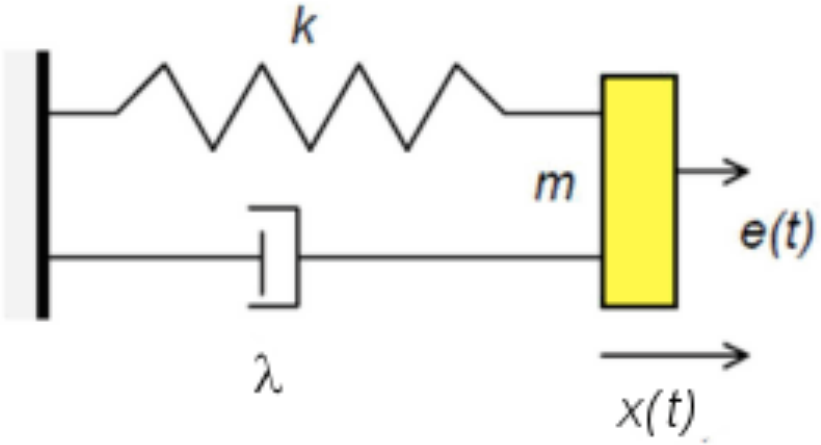
## Exemple : Masse-Ressort-Amortisseur [ modifier | modifier le code ]

Étudions un système idéal Masse-Ressort-Amortisseur, avec une masse *m* fixée (dans le sens où le corps garde la même masse tout au long de l'étude), une constante de raideur *k*, et un coefficient d'amortissement *c* :

$$\mathbf{F_r} = -k\mathbf{x}$$
$$\mathbf{F_a} = -c\frac{d\mathbf{x}}{dt}.$$

La masse est un corps libre. On suppose le repère inertiel, donc le premier vecteur est parallèle au ressort et à l'amortisseur. D'après la conservation de la quantité de mouvement :

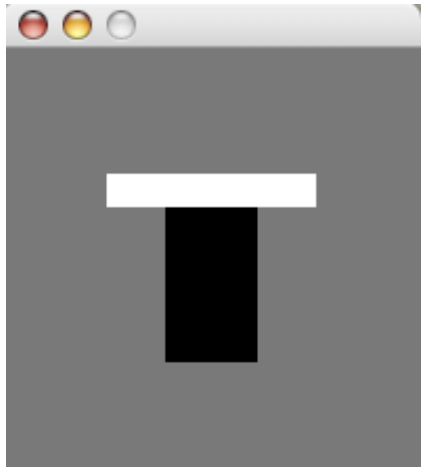
$$\mathbf{F_r} + \mathbf{F_a} = m\frac{d^2\mathbf{x}}{dt^2}$$
$$-kx - c\frac{dx}{dt} = m\frac{d^2x}{dt^2}.$$

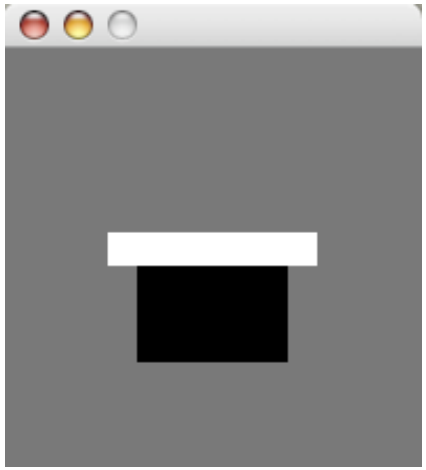


# « Spring » : exemple simple dans les démos Processing :

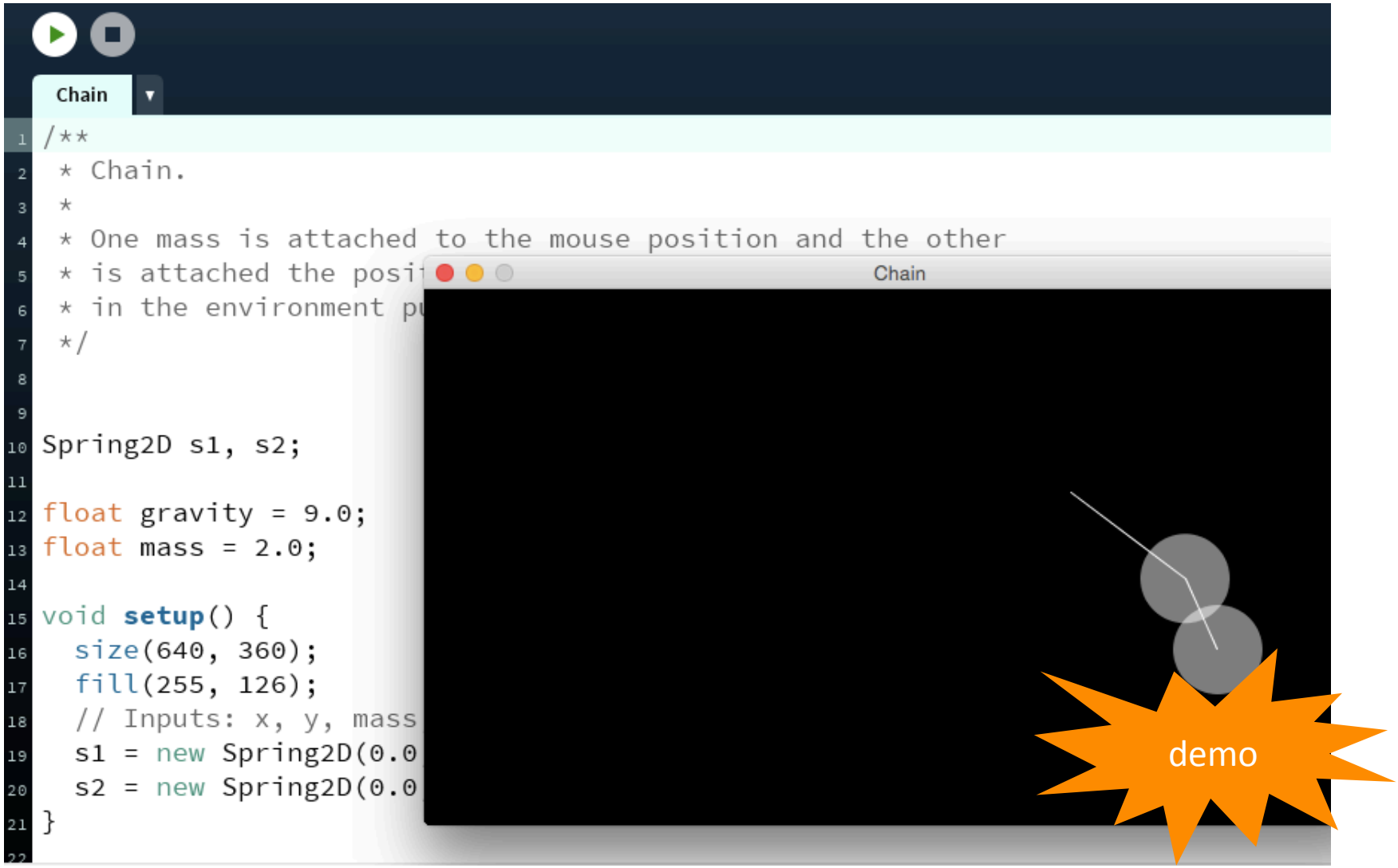
```
// Spring simulation constants
float M = 0.8;   // Mass
float K = 0.2;   // Spring constant
float D = 0.92;  // Damping
float R = 60;    // Rest position

// Spring simulation variables
float ps = 60.0; // Position
float vs = 0.0;  // Velocity
float as = 0;    // Acceleration
float f = 0;     // Force
```

```
void updateSpring()
{
  // Update the spring position
  if(!move) {
    f = -K * (ps - R);    // f=-ky
    as = f / M;           // Set the acceleration, f=ma == a=f/m
    vs = D * (vs + as);   // Set the velocity
    ps = ps + vs;         // Updated position
  }
  if(abs(vs) < 0.1) {
    vs = 0.0;
  }
```

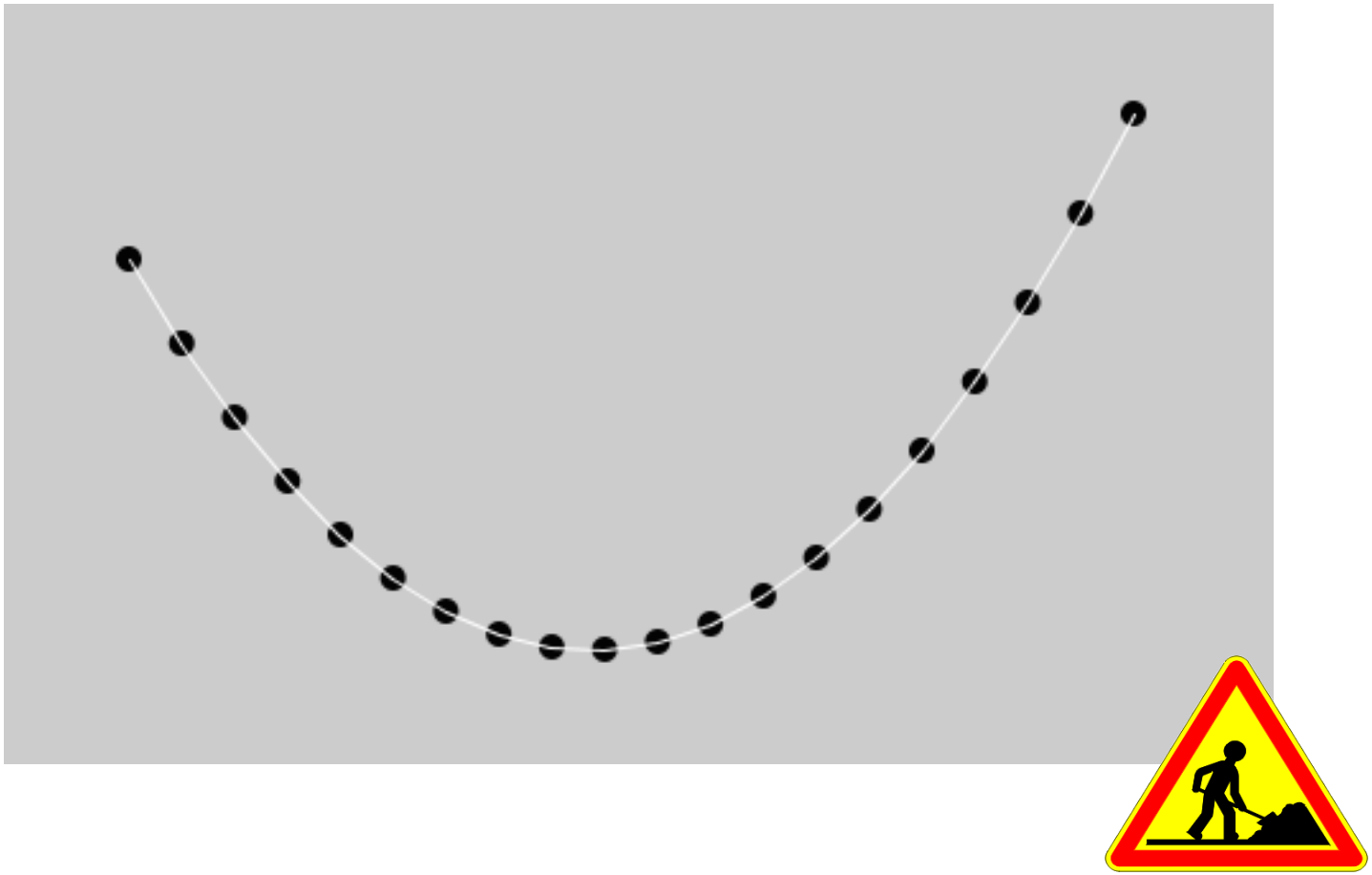




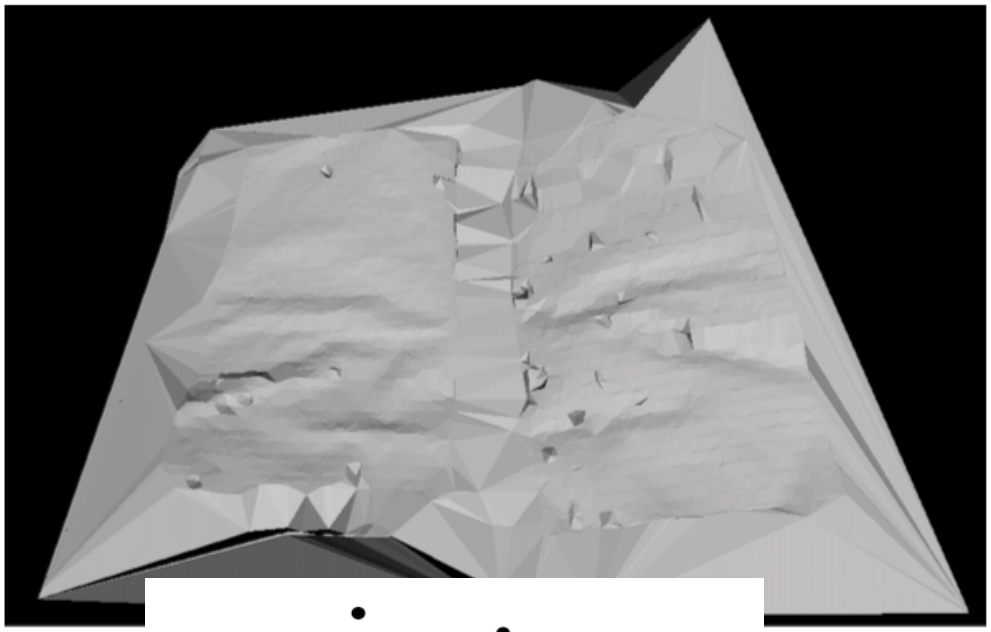
Autre exemple : « chain »



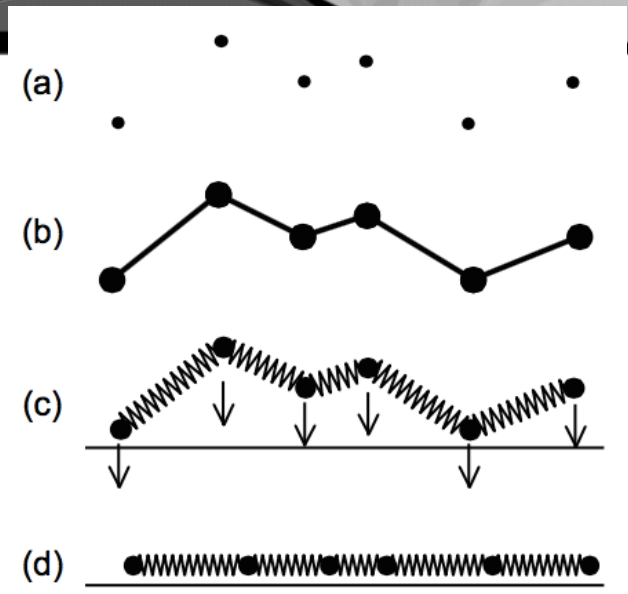
# Application à la simulation d’une chainette

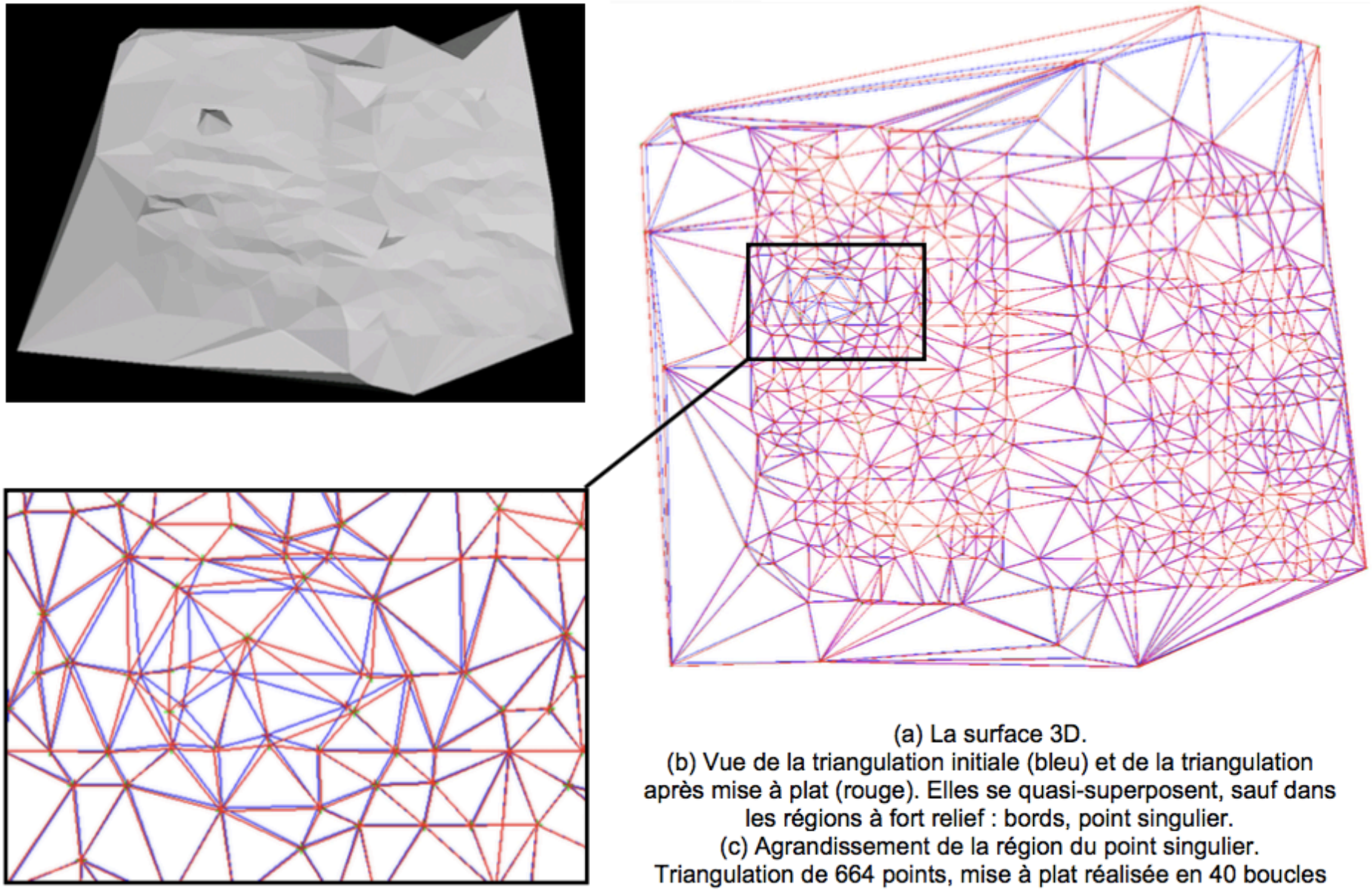


# Application aux étoffes, papier …

J.F. Haas, P. Cubaud CIFED'04

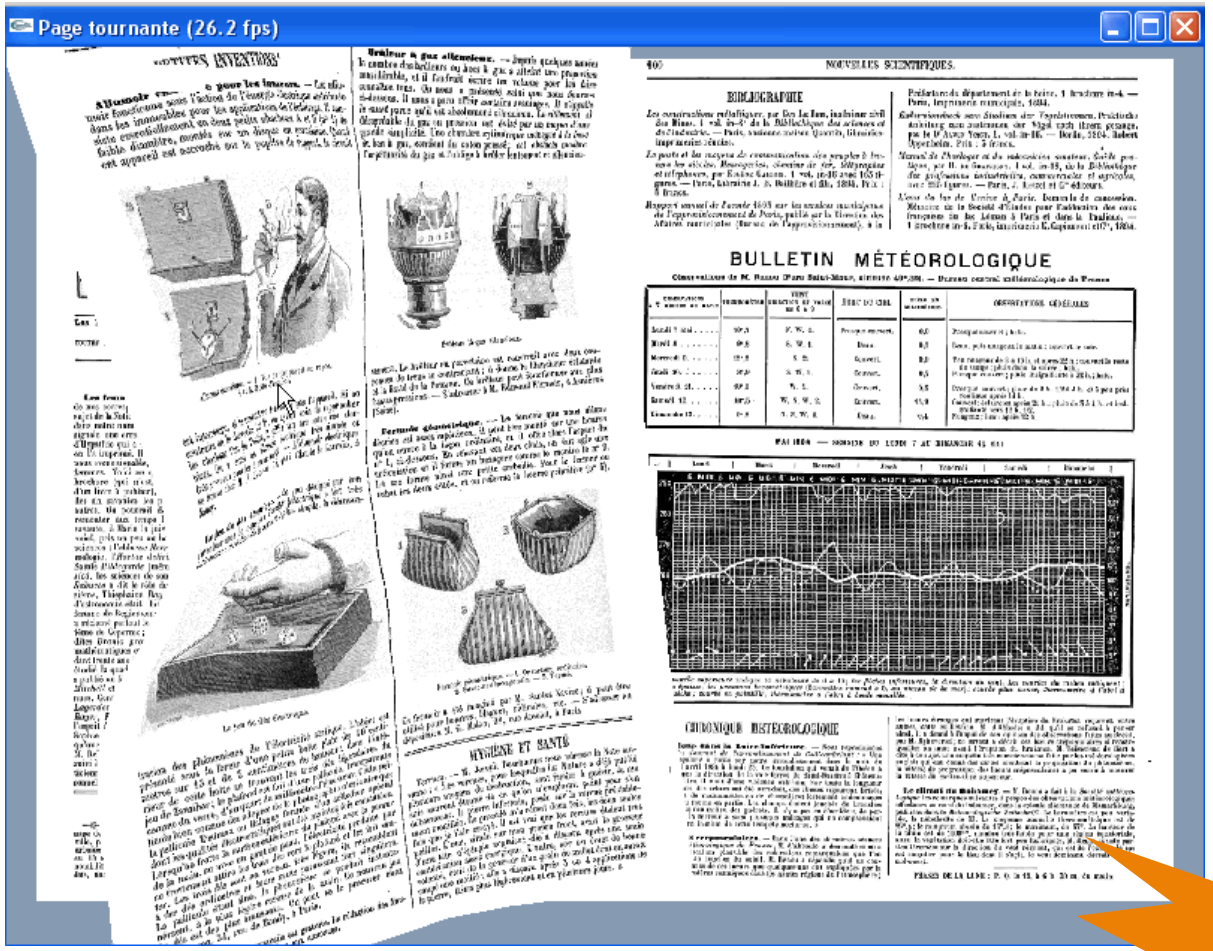




(a) La surface 3D.
(b) Vue de la triangulation initiale (bleu) et de la triangulation après mise à plat (rouge). Elles se quasi-superposent, sauf dans les régions à fort relief : bords, point singulier.
(c) Agrandissement de la région du point singulier.
Triangulation de 664 points, mise à plat réalisée en 40 boucles

J.F. Haas, P. Cubaud CIFED'04

Utilisation d'un moteur physique pour simuler le dechirage de pages




video

A. Topol et al. , CIDE'09

# Nombreuses autres méthodes : voir Wikipedia

## Soft-body dynamics

From Wikipedia, the free encyclopedia
(Redirected from Soft body dynamics)

**Soft-body dynamics** is a field of computer graphics that focuses on visually realistic physical simulations of the motion and properties of deformable objects (or *soft bodies*).[1] The applications are mostly in video games and films. Unlike in simulation of rigid bodies, the shape of soft bodies can change, meaning that the relative distance of two points on the object is not fixed. While the relative distances of points are not fixed, the body is expected to retain its shape to some degree (unlike a fluid). The scope of soft body dynamics is quite broad, including simulation of soft organic materials such as muscle, fat, hair and vegetation, as well as other deformable materials such as clothing and fabric. Generally, these methods only provide visually plausible emulations rather than accurate scientific/engineering simulations, though there is some crossover with scientific methods, particularly in the case of finite element simulations. Several physics engines currently provide software for soft-body simulation.[2][3][4][5][6][7]

**Contents** [hide]




Softbody objects react to forces and are able to collide with other 3D objects. This example has been created with Blender.