

# Algorithmes d'exploration et de mouvements

GMIN20A

## Flocking et éviteme nt

V2.1 – Mars 2015

**Jacques Ferber**

# Comportements collectifs

## ◆ Types de comportements

- Comportements coopératifs
- Comportements collectifs de défense et d'attaque



# Coordination coopérative

- ◆ On suppose que tous les autres agents sont des amis
- ◆ On cherche à créer des configurations spécifiques

# Types de formation coopératives

## ◆ Coordination de mouvements

- Flocking (escadrille)
- Entourer
- Se mettre en ligne
- Positionnement géométrique

## ◆ Formation dépendant de la nature des agents

- Ex: agents d'exploration, chasseurs, croiseurs
- Se mettre dans une formation quelconque
  - ☞ Carré, rond, triangle, en fonction des rôles, etc..

## ◆ Formations dynamiques

- Recrutement d'agents,
- Ex: récupération de ressources

# Qu'est ce qu'une escadrille, troupeau (flock)

**definition: (flock) un groupe d'oiseaux, de poissons de mammifères, d'humains (d'agents) qui avancent ensemble en formation.**



# Boids!

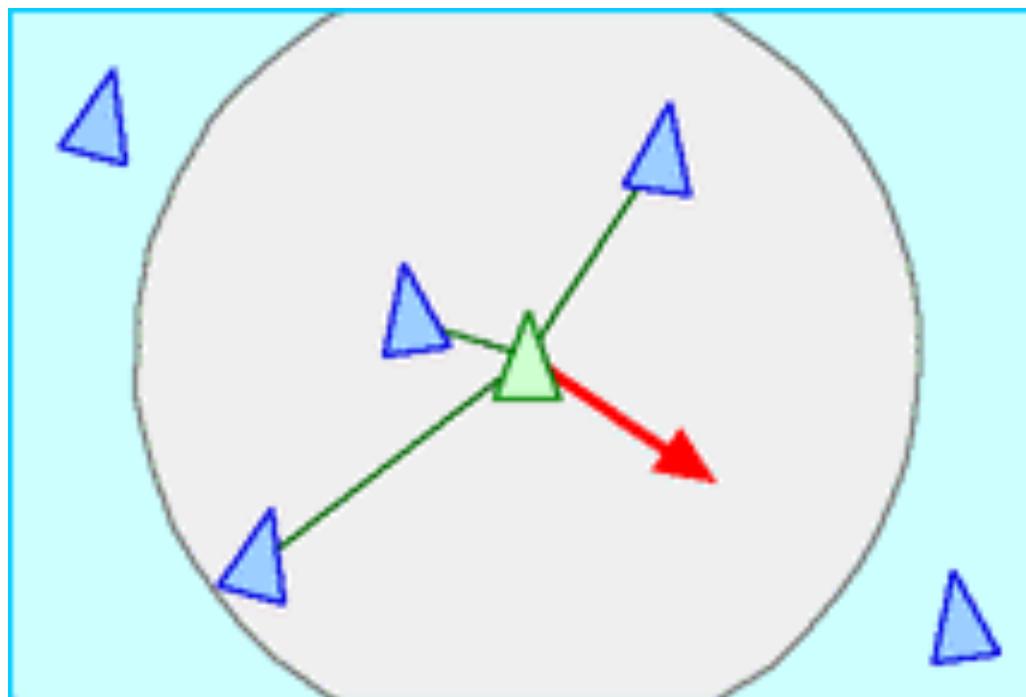
- ◆ “boids” vient de “bird-oids” - Article fondateur de Craig Reynolds en 1986.
- ◆ Mécanisme semblable à des systèmes de particules, mais avec une orientation
- ◆ Mouvement dirigé par le comportement (Behavior-based motion)
  - Algorithme distribué, pas de calcul centralisé.
- ◆ Vitesse constante
- ◆ Contrainte uniquement en terme de rotation

# Mouvement de troupeau (flocking)

- ◆ Les Boids doivent se coordonner avec leur voisins
- ◆ Deux tendances principales
  - Rester près des autres
  - Eviter les collisions avec les autres
- ◆ Dans la nature, les mouvements ont évolués
  - Prédatation
  - Trouver de la nourriture
  - Rencontre amoureuse/sexuelle (mating)

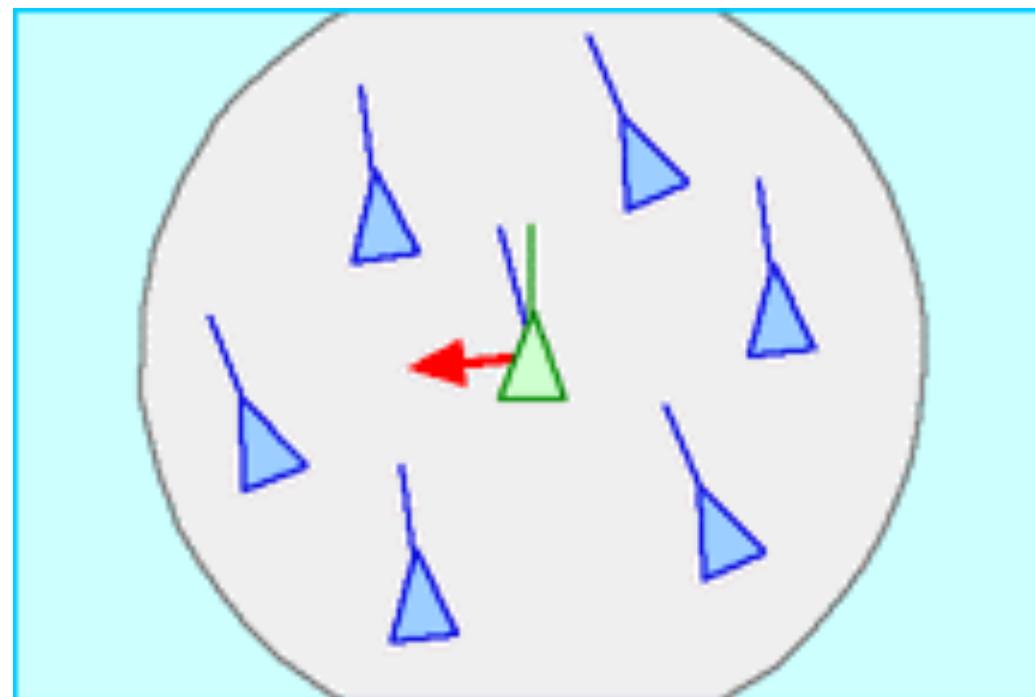
# 1- Séparation: éviter les collisions

- ◆ **Evitement de collision:** éviter les collisions avec les voisins



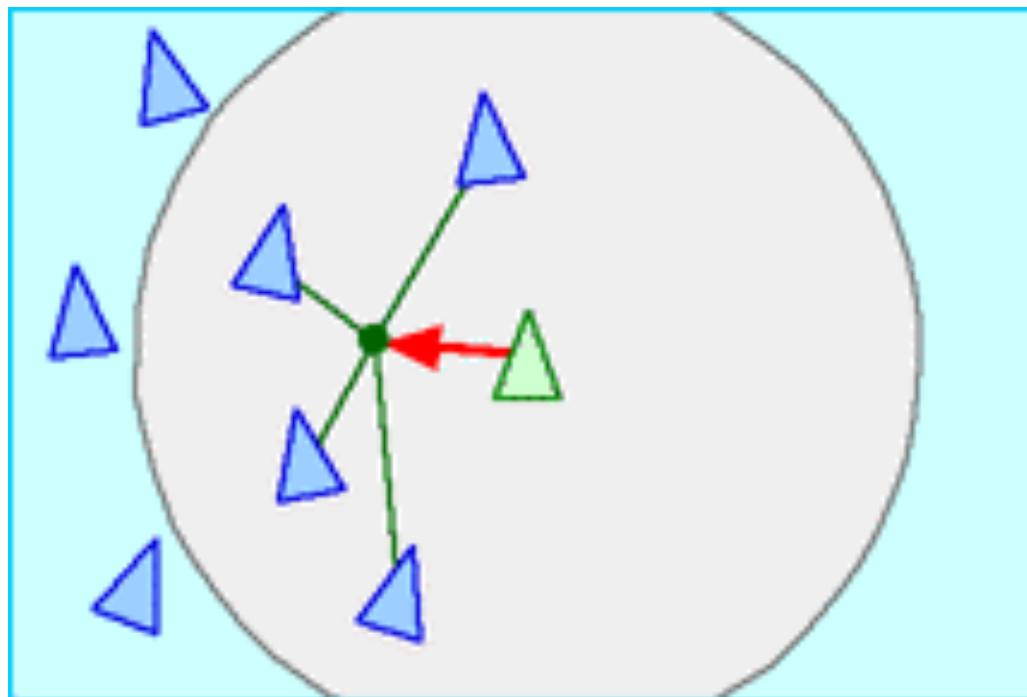
## 2 – Alignement : s'aligner avec les autres

- ◆ Essayer de s'adapter à la vitesse et direction des voisins



## 3 - Centrage

- ◆ Centrage: essayer d'être le plus près des voisins, d'être plus au centre du troupeau..



# Composer et arbitrer ces comportements

- ◆ Technique simple (implémentée dans NetLogo):

Soit E l'ensemble des voisins

Soit x le plus proche de self parmi E

Si  $\text{dist}(\text{self},x) < \text{dist-min}$

s'éloigner de x

sinon

s'aligner avec E  
aller vers E

S'aligner (E )

dir = Somme(direction(x)) pour tout x de E

tourner-vers(dir) // au plus de k degré

Se mettre en cohérence avec E

dir = moyenne(vecteurs(x,self)) pour tout x de E

tourner-vers(dir) // au plus de k degré

# Formation en V: le vol des oies sauvages

Nathan, A. & Barbosa, V. C (2008)

- ◆ Règle 1: si agent est trop loin des autres agents, il accélère pour se rapprocher du plus proche.
- ◆ Règle 2: si un agent est suffisamment près d'un autre, il va venir sur l'un de ses côtés, pour que sa vue ne soit pas obstruée
- ◆ Règle 3: Si un agent est trop proche d'un autre, il ralentit
- ◆ Règle 4: quand les trois autres conditions sont remplies, l'agent adapte sa vitesse et direction à ses voisins visibles

*NetLogo models library: Flocking Vee Formations*

```

to adjust ;; ajuster la direction et position par rapport aux autres
  set closest-neighbor min-one-of visible-neighbors [distance myself]
  let closest-distance distance closest-neighbor
  ;; if I am too far away from the nearest bird I can see, then try to get near them
  if closest-distance > updraft-distance [
    turn-towards (towards closest-neighbor)
    set speed base-speed * (1 + speed-change-factor)
    set happy? false
    stop
  ]

  ;; if my view is obstructed, move sideways randomly
  if any? visible-neighbors in-cone vision-distance obstruction-cone [
    turn-at-most (random-float (max-turn * 2) - max-turn)
    set speed base-speed * (1 + speed-change-factor)
    set happy? false
    stop
  ]

  ;; if i am too close to the nearest bird slow down
  if closest-distance < too-close [
    set happy? false
    set speed base-speed * (1 - speed-change-factor)
    stop
  ]

  ;; if all three conditions are filled, adjust
  ;; to the speed and heading of my neighbor and take it easy
  set speed [speed] of closest-neighbor
  turn-towards [heading] of closest-neighbor
  set happy? true
end

```

# Approche vectorielle

## ◆ Idée :

- Composer l'ensemble des motivations et décisions d'actions sous la forme de vecteurs
- Utiliser de simples compositions vectorielles linéaires (la plupart du temps) ou non-linéaires pour déterminer le mouvement

## ◆ Comportement $B$ est déterminé par la somme des comportements élémentaires $B_i$

$$\vec{B} = \alpha_1 \vec{B}_1 + \dots + \alpha_n \vec{B}_n$$

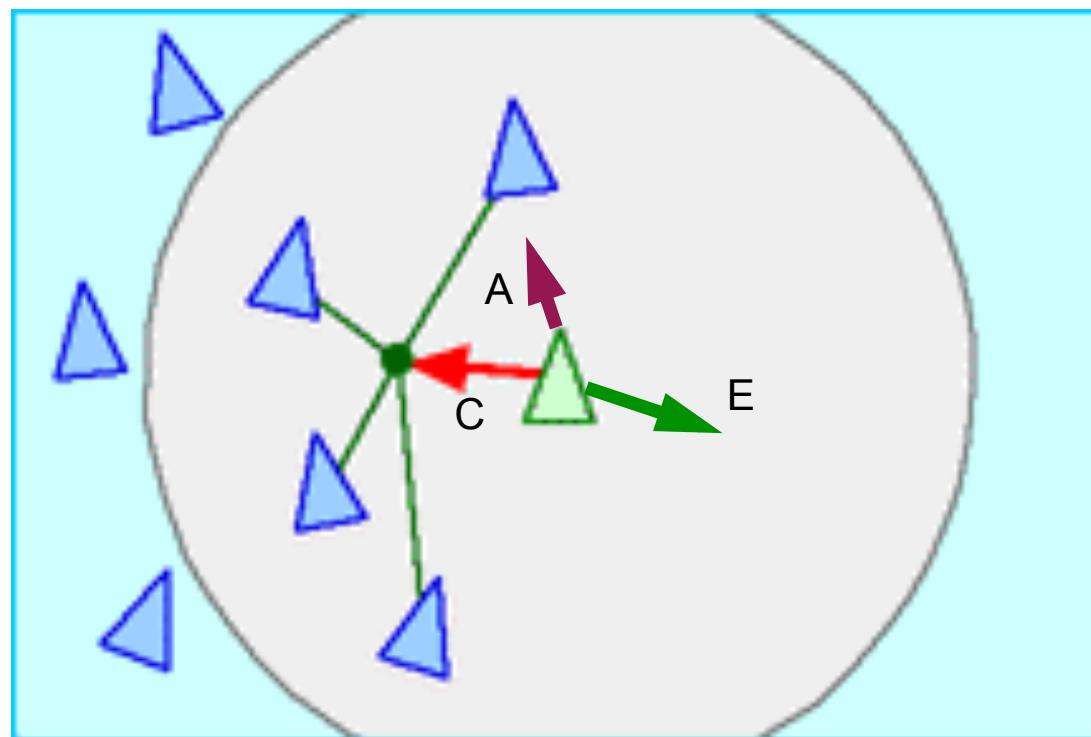
$$\vec{B} = \sum_{i=1}^n \alpha_i \vec{B}_i$$

# Application au flocking

- ◆ Chaque comportement (évitement, alignement, centrage) définit un vecteur
    - On somme ces vecteurs en fonction de leur importance:
- $$\vec{D} = a \vec{A} + e \vec{E} + c \vec{C}$$
- A = alignement, E = Evitement, C = centrage

## 3 - Centrage

- ◆ Centrage: essayer d'être le plus près des voisins, d'être plus au centre du troupeau..



A = alignement,  
E = Evitement,  
C = centrage  
D = direction  
résultante

$$\vec{D} = a \vec{A} + e \vec{E} + c \vec{C}$$

# Intégrer les accélération (cinématique)

- ◆ Un agent est défini par
  - sa position (x, y)
  - sa direction d (heading)
- ◆ Vecteur vitesse instantanée:
  - $v$  = vitesse linéaire
  - $\omega$  = vitesse angulaire
- ◆ L'action est défini par une accélération instantanée
  - $a$  = accélération (ou action) à l'instant t
  - $r$  = accélération angulaire

## Formules

$$v_{t+1} = v_t + a_t \cdot \Delta t$$

$$\omega_{t+1} = \omega_t + r_t \cdot \Delta t$$

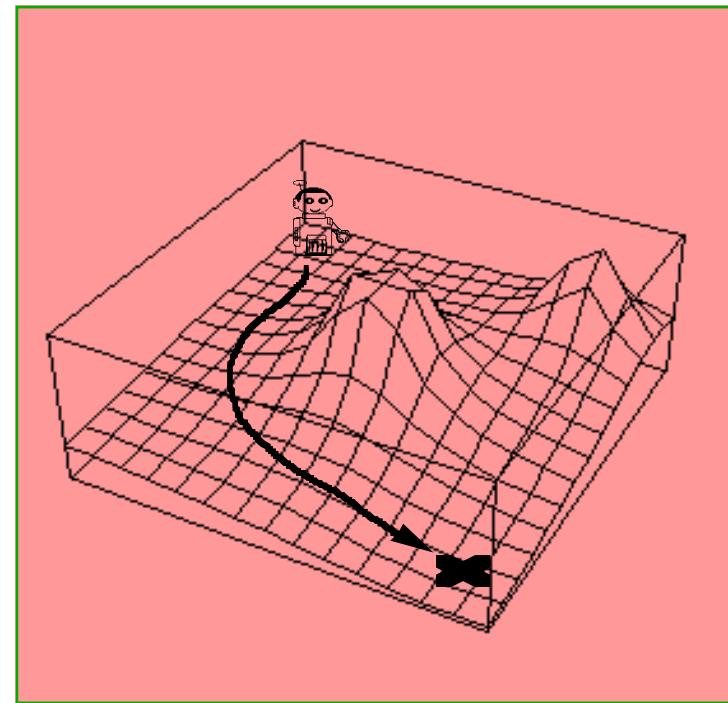
$$p_t = \frac{1}{2} a \cdot \Delta t^2 + v_t \cdot \Delta t$$

$$d_t = \frac{1}{2} r_t \cdot \Delta t^2 + v_t \cdot \Delta t$$

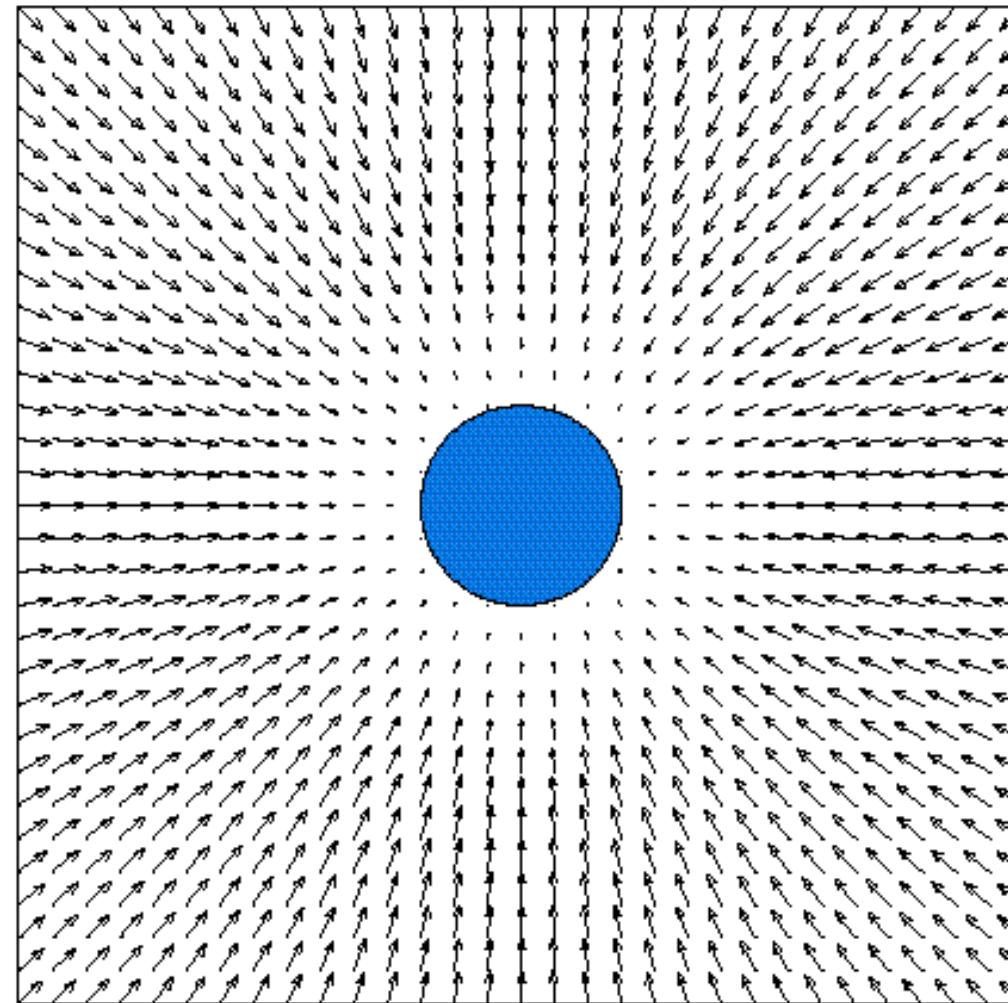
# Champs de forces

*L'agent se déplace dans un champ de forces*

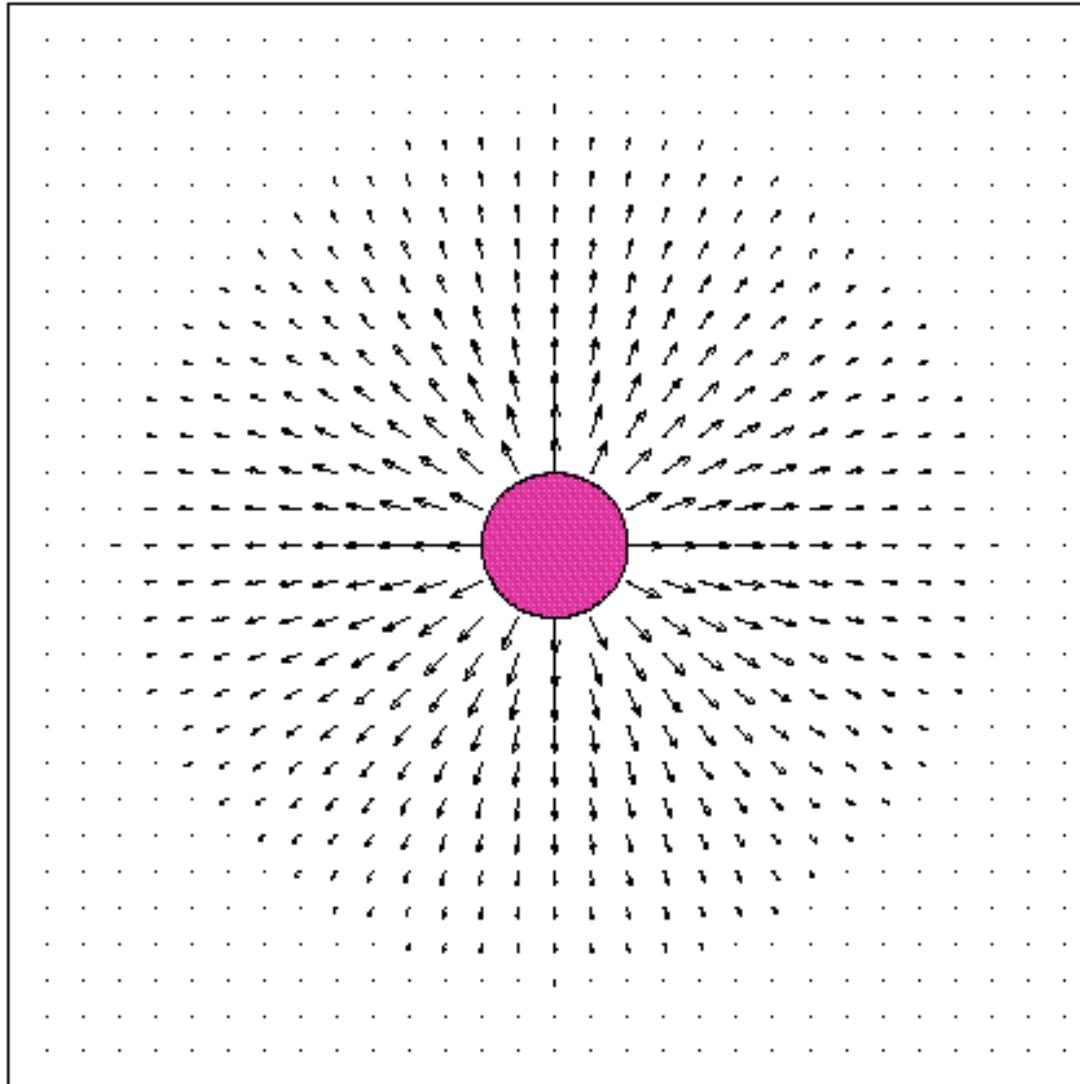
*La position à atteindre est un attracteur, et les obstacles sont des éléments répulsifs*



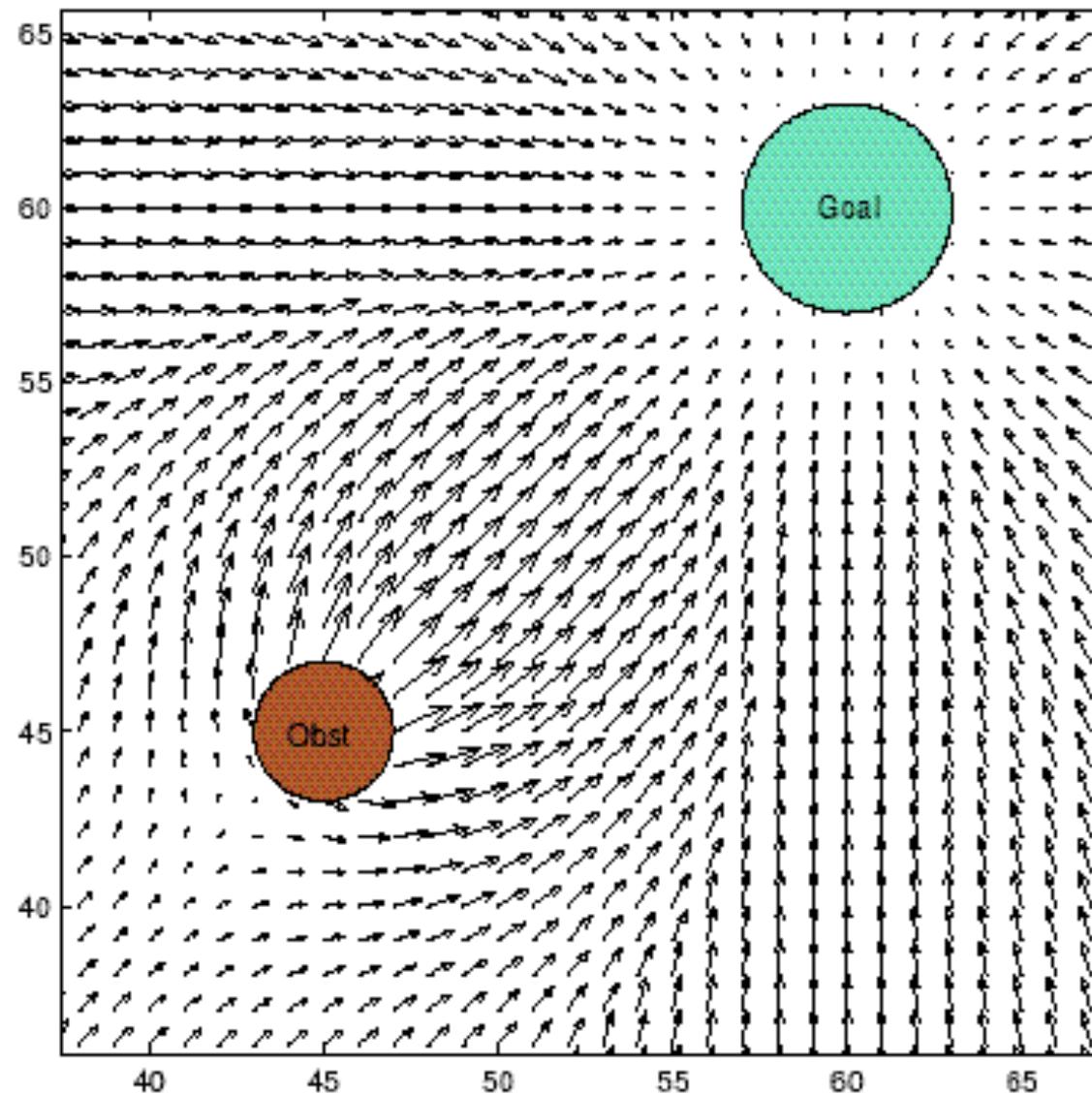
# Champ de force attractif



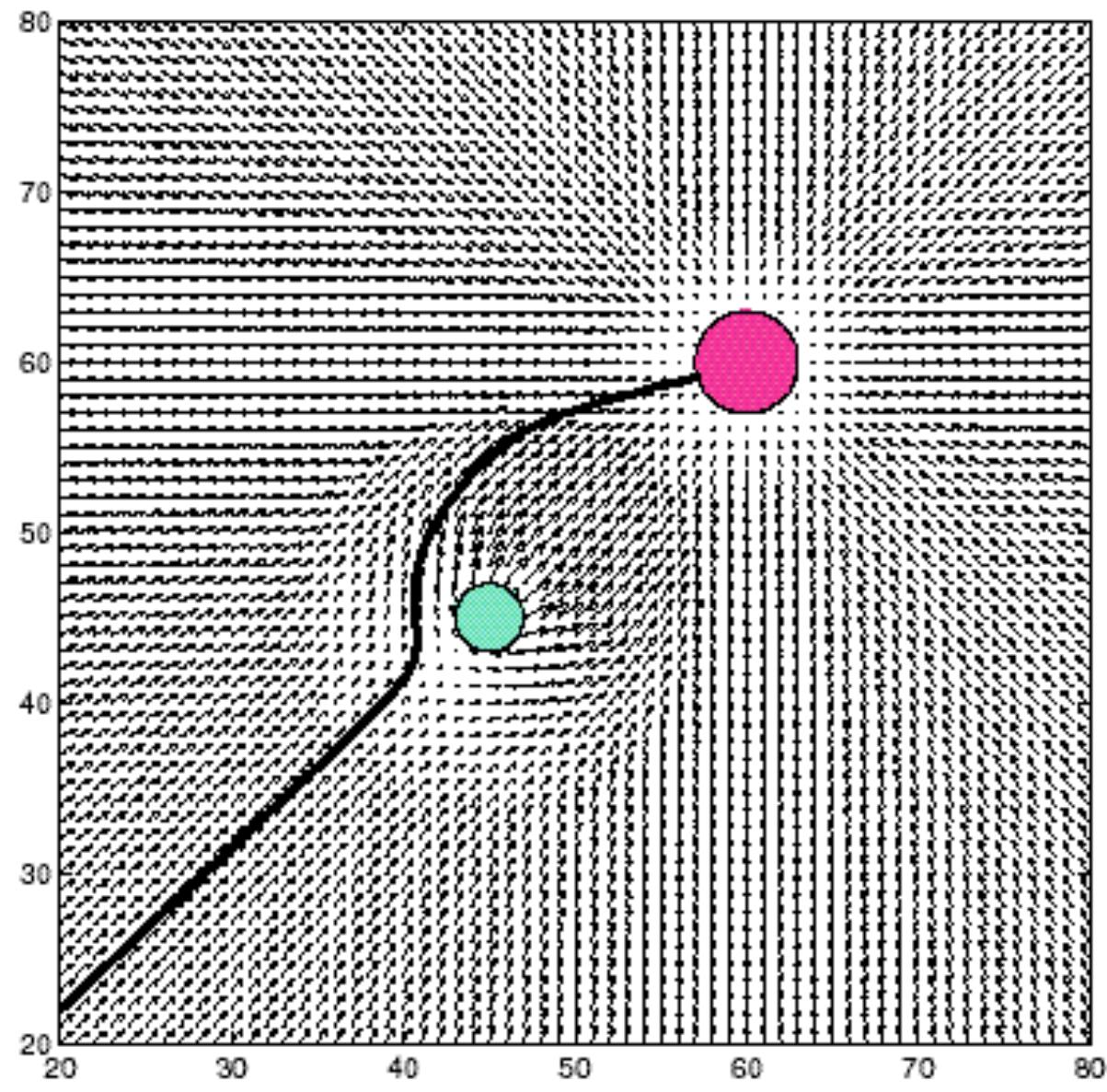
# Champ de force répulsif



# Somme vectorielle des deux champs



## Trajectoire résultante de l'agent



# Champs de potentiel

- ◆ Un champ de potentiel est une fonction qui associe à tout points  $(x,y)$  un nombre considéré comme la valeur du champ en ce point:  $P(x,y)$
- ◆ Le gradient d'un champ de potentiel est un champ vectoriel défini :

$$(x,y) \rightarrow (\Delta x, \Delta y) = \nabla P(x,y)$$

$$(\Delta x, \Delta y) = \nabla P(x,y) = \left( \frac{\partial P}{\partial x}, \frac{\partial P}{\partial y} \right)$$

# Construction du champ: attraction et répulsion

Les forces sont définies comme le gradient d'un champ de potentiel

$$\mathbf{F}(x,y) = -\nabla P(x,y)$$

Les buts sont représentés comme des champs attractifs.

Les obstacles sont représentés comme des champs répulsifs

Le mouvement est obtenu par une combinaison de champs attractifs et répulsifs

$$P(x,y) = U_{attr}(x,y) + U_{repul}(x,y)$$

- ◆ **Mouvement: il suffit de « descendre » le champ en suivant le gradient, la ligne de plus grande pente**

# Problèmes avec les champs de potentiels

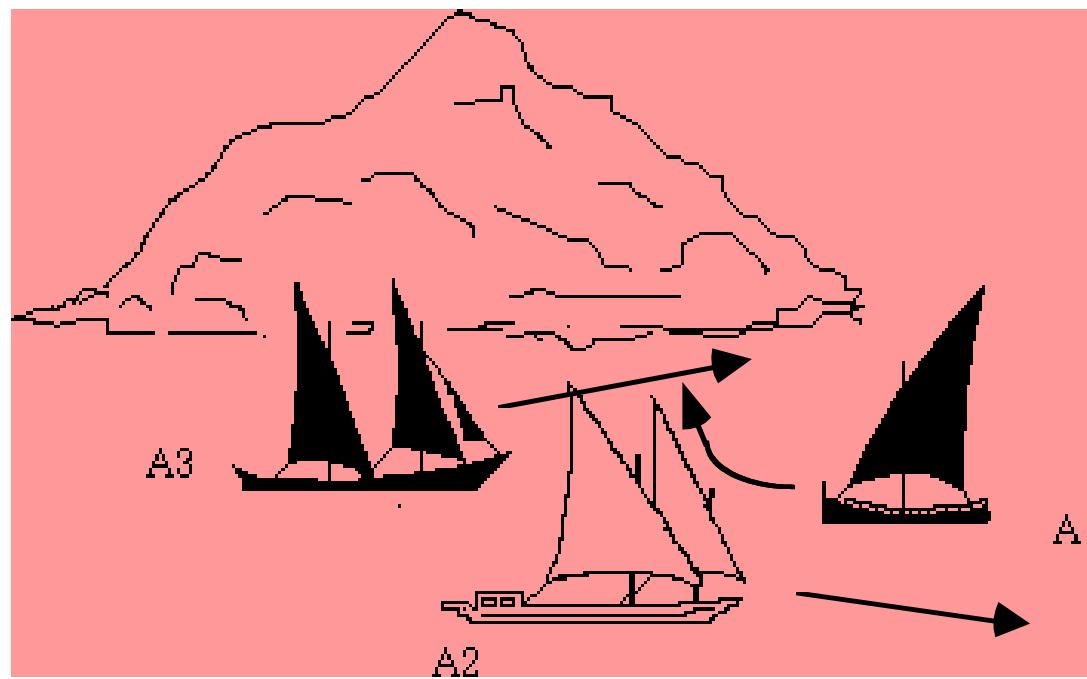
## ◆ *Minimum local*

- Les forces attractives et répulsives peuvent s'annuler
- Les formes convexes constituent des cul de sac

## ◆ Cas dynamique (avec plusieurs agents)

- Les agents s'évitent mais n'arrivent jamais à bon port (cf. suite)

# Evitement de collision



# Evitement d'obstacles statiques

## ◆ 1<sup>ère</sup> solution

- Répulsion

## ◆ 2<sup>ème</sup> solution

- S'écartez de la trajectoire initiale
- Nécessite une répulsion si trop près...

## ◆ Deux approches

- Priorité: d'abord flocking, ensuite répulsion, ou l'inverse
- Combinaison de vecteurs

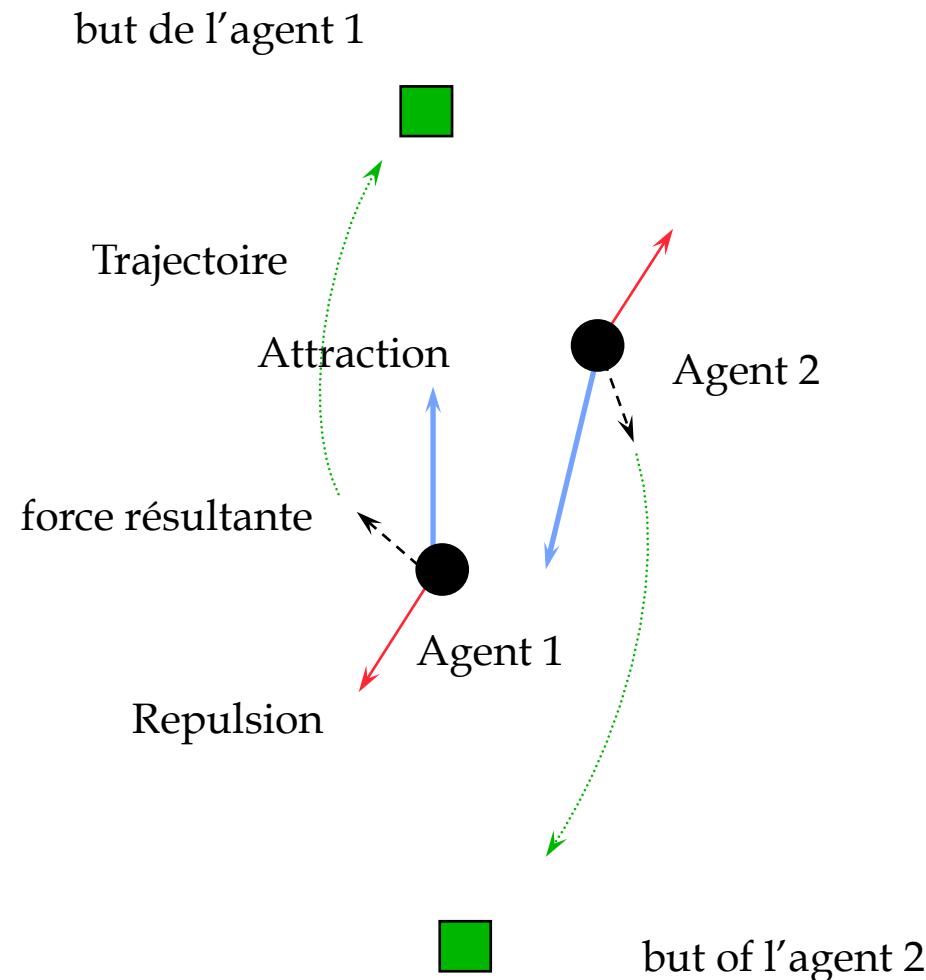
$$\mathbf{D} = a\mathbf{R} + (1 - a)\mathbf{F}$$

Où  $\mathbf{R}$  est le vecteur de répulsion et  $\mathbf{F}$  celui du flocking.

- $a$  est le coefficient de contrôle. Peut varier en fonction inverse de la distance à l'obstacle (quand l'obstacle est droit devant)  $a = k/dist(self, obstacle)$

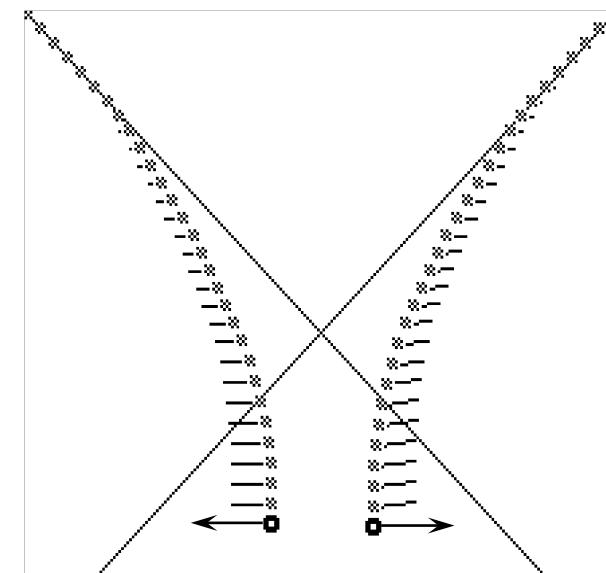
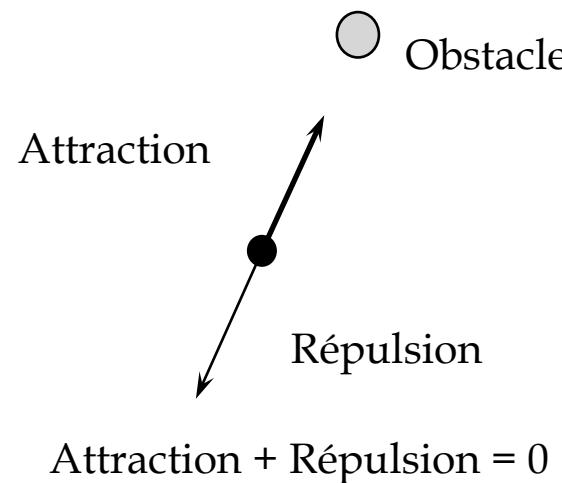
# Approche par champ de force

**Chaque agent est considéré comme étant un obstacle pour l'autre**



# Mais la répulsion n'est pas l'évitement

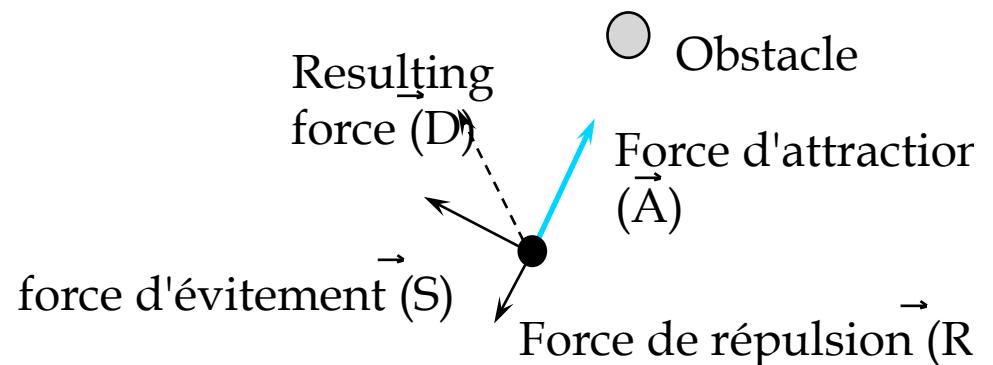
But



# Solution: utiliser des forces d'évitement

But

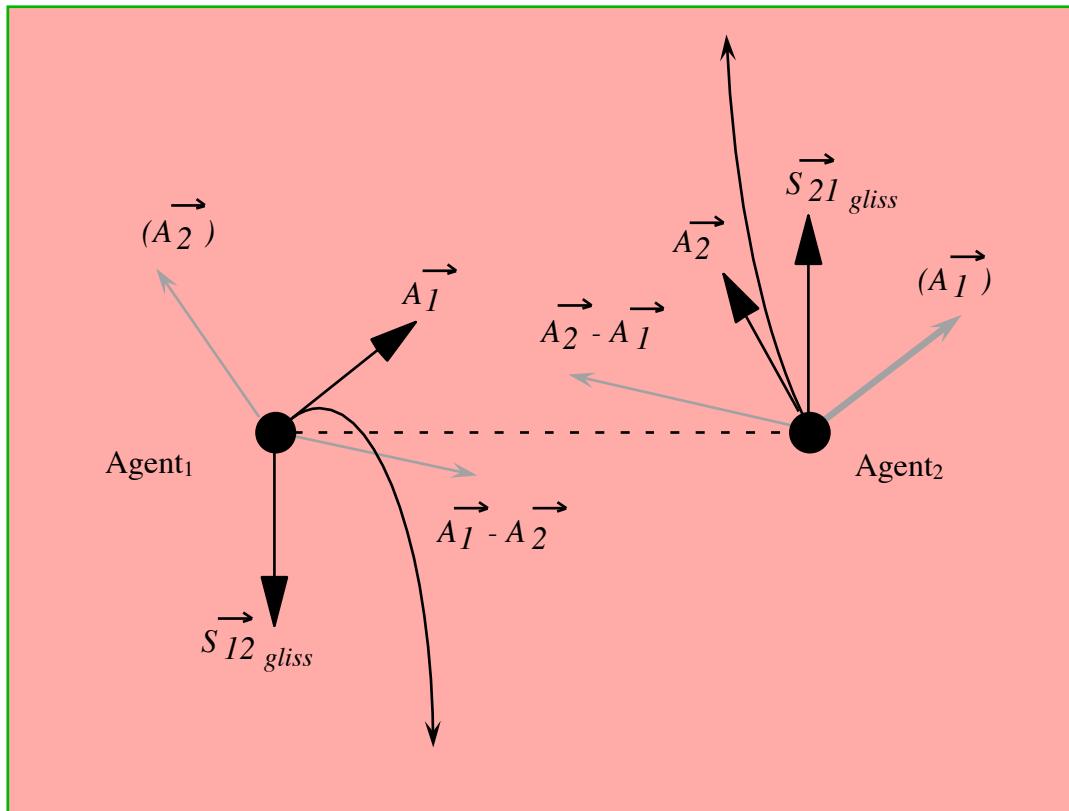
**Eviter** signifie rester à "bonne"  
distance des obstacles en se  
dirigeant vers le but



$S(p)$  est tel que  $\vec{dir}(R(p)) \cdot \vec{dir}(S(p)) = 0$

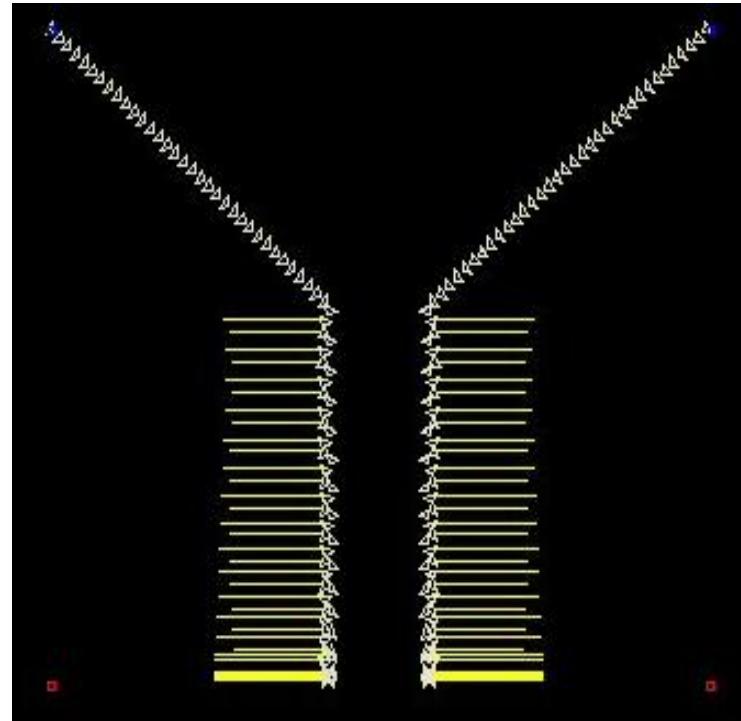
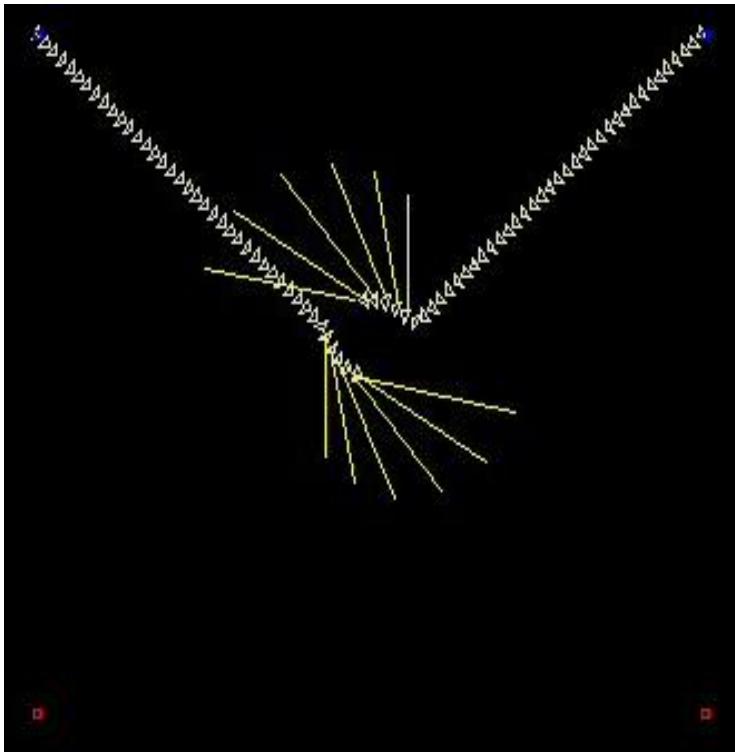
$$\vec{D}(p) = a\vec{A}(p) + b\vec{R}(p) + g\vec{S}(p)$$

# Forces d'évitement symétrique



K. Zeghal, J. Ferber 1992

# Examples



*D'après un projet réalisé à  
l'UTBM sous la dir. d'O. Simonin*

# Emergence de structures dynamiques

