

Programmation orientée agents #2

L'importance de l'environnement

M1 S2 - Université de Montpellier

FMIN108 – Master Informatique

Jacques Ferber

Version 1.3. Oct 2016

Comment programmer les déplacements en NetLogo

◆ Mouvements

- Les tortues ont un mouvement local défini à partir d'une « géométrie tortue »
- fd , rt, lt
 - ☞ cercle : repeat 360 [fd 1 rt 1]

◆ Possible d'aller vers un objet particulier

- NetLogo:
 - ☞ set heading towards x fd 1
 - ☞ Où x est une tortue ou un patch.
- Ex:
 - ☞ set heading towards patch 0 0
 - ☞ face reine 1 ;; va vers la 1^{ère} reine
 - ☞ face one-of reines ;; va vers une reine quelconque...

Problèmes des déplacements directs

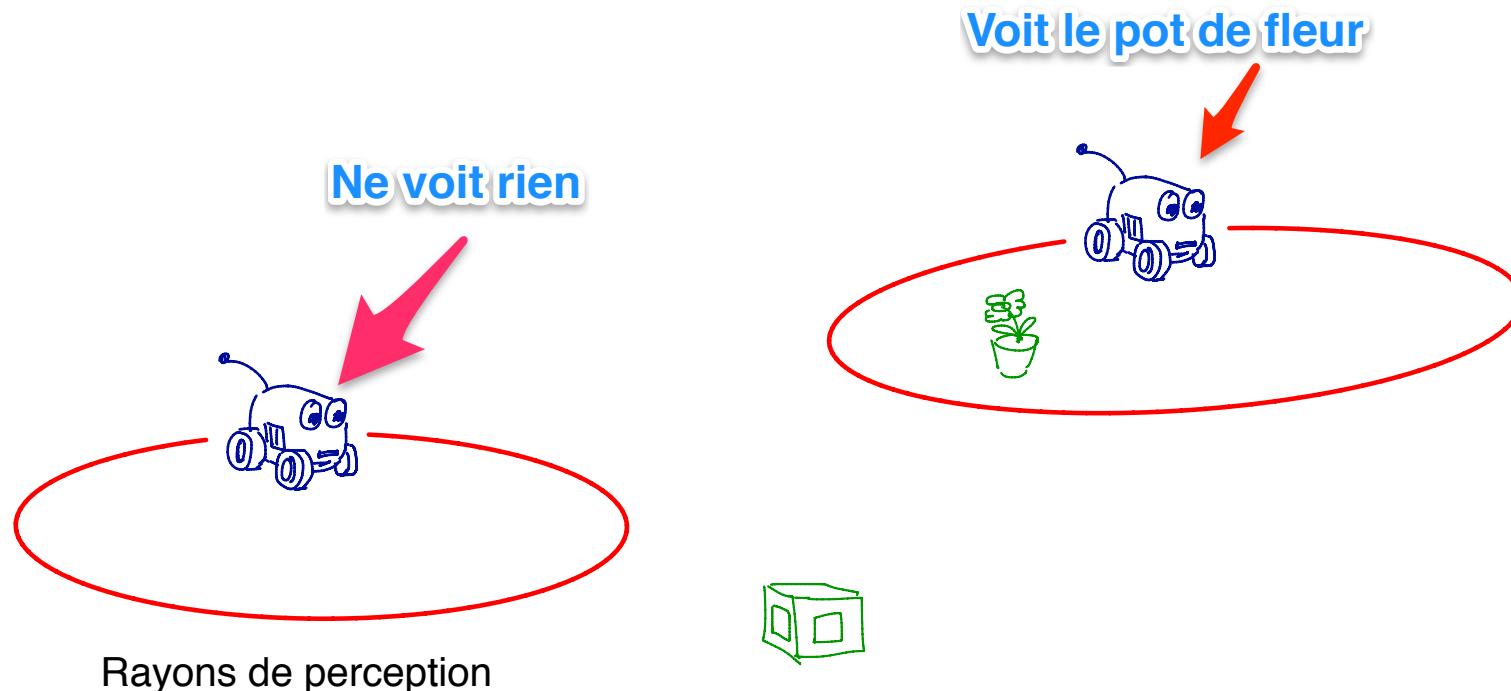
◆ Hypothèses

- Suppose que l'on connaisse les coordonnées (ou tout du moins que l'on puisse avoir la direction vers le but)
- Ne prend pas en compte les obstacles

◆ *Problème: ne prend pas en compte la notion de perception limitée essentielle dans la programmation agent!!*

Les agents ont une perception locale

◆ Perception limitée



Primitives de perception en NetLogo

◆ <trucs> in radius <rayon de perception>

- Où trucs est un agentset (turtles, patches, « breed » ou une restriction d'un de deux là)

◆ Ex:

- Reines in radius 3
 - ☞ Retourne l'agentset de toutes les reines dans un rayon de 3
- in radius 5 with [energy-level > 10]
- Patches with [energy-level > 10] in-radius 5
 - ☞ Attention différence en termes de complexité (et donc de temps de calcul) entre les deux expressions!!

Pour aller vers ce que l'on a perçu

◆ Aller vers l'objet

- `towards <objet perçu>`
- Ex:

```
let p one-of reines in-radius 5
  if p != nobody [
    set heading towards p ;; ou face p
    fd 1 ;; ou faire un « gigoter » wiggle
  ]
```

◆ Aller vers l'objet le plus proche

- `min-one-of <agentset> [distance myself]`
- Ex:
- Let r `min-one-of reines in-radius 5 [distance myself]`
if r != nobody [
 face r
]

L'importance de l'environnement

◆ Mais l'environnement contient plein d'informations:

- Informations naturelles
 - ☞ Végétation, amers, paysage (montagnes, sols, etc.)
- Ajouts d'informations
 - ☞ Marques, balises, phéromones
- Système de communication
 - ☞ Signaux

Aller vers un patch particulier

◆ Pour aller vers le patch avec la plus grande valeur d'un attribut

- max-one-of <patches visible> [<attribut>]
- Ex:
max-one-of patches in radius 8 [hauteur-herbe]
 - ☞ Retourne le patch ayant l'herbe la plus haute dans un rayon de 8
 - ☞ Attention: doit avoir défini l'attribut 'hauteur-herbe' comme attribut de patch:
patches-own [hauteur-herbe]

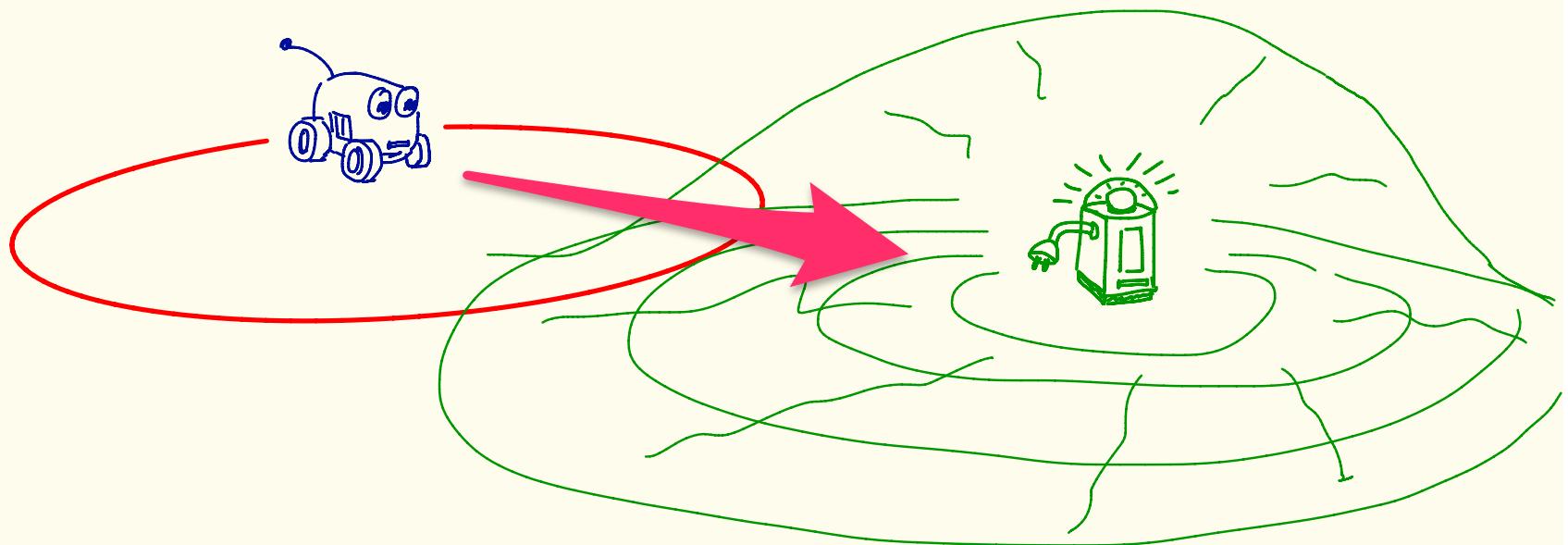
Principe général d'un environnement qui contient des indices

- ◆ **On suit les indices en espérant qu'ils nous conduisent au but en nous faisant éviter les obstacles**
- ◆ **Les indices sont des substituts de ce vers quoi on se dirige**
 - Ex: les traces des animaux pour un prédateur

Se diriger grâce à un signal

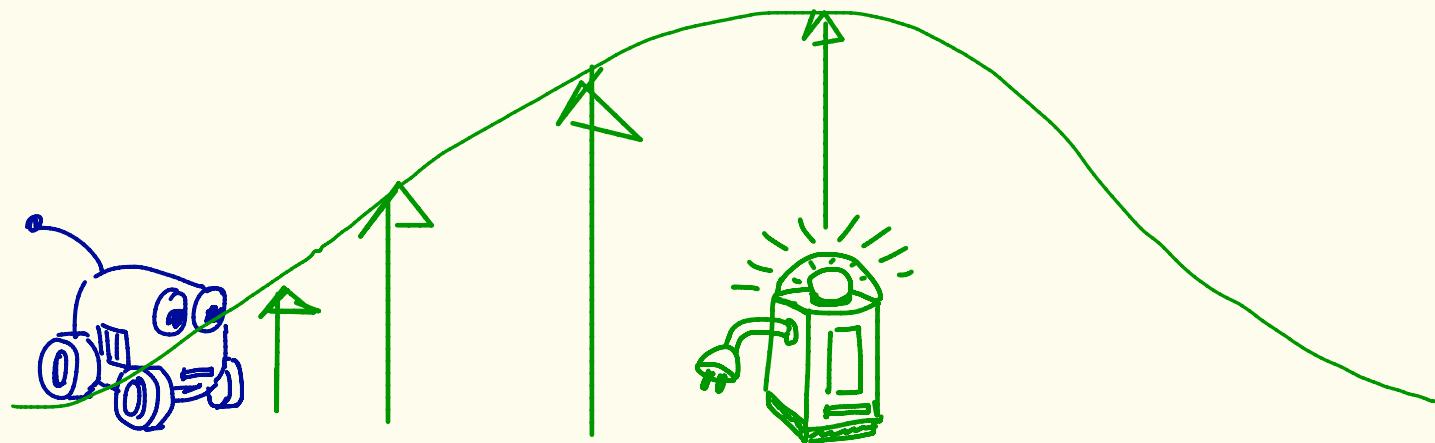
- ◆ les indices et traces sont interprétés comme des signaux pour aller vers un but

L'agent est attiré par le signal émis par la borne



Aller vers les valeurs les plus grandes du champ

= suivre le gradient d'un champ de potentiel



VALEURS DU CHAMP

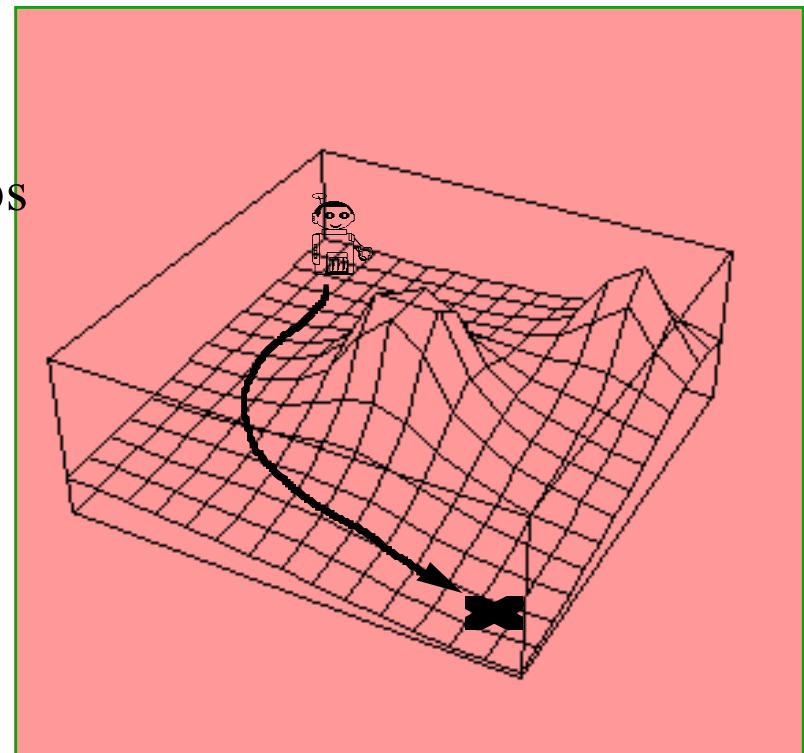
Suivi de gradient de potentiel

◆ Suivre un gradient de potentiel

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

$$\mathbf{F}(\mathbf{p}) = -\nabla U(\mathbf{p})$$

Les buts sont représentés comme des champs attractifs.



Les obstacles

- ◆ **Les obstacles sont des champs qui émettent un signal négatif**

- Les obstacles sont représentés comme des champs répulsifs

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

$$U(p) = U_{\text{attr}(p)} + U_{\text{repul}(p)}$$

- ◆ **Les obstacles sont des champs qui émettent un signal négatif**

- Les obstacles sont représentés comme des champs répulsifs

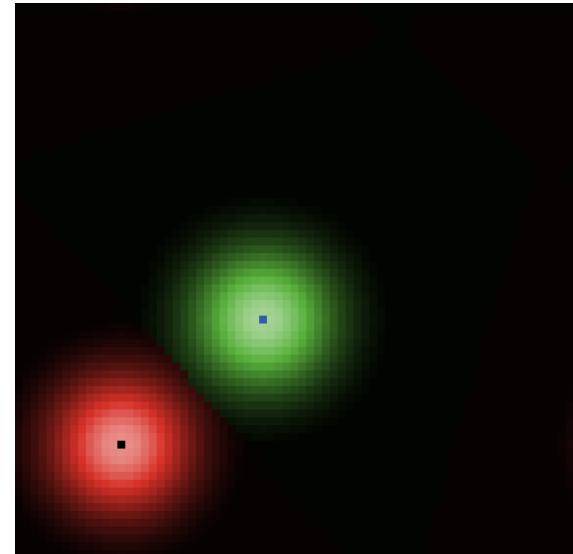
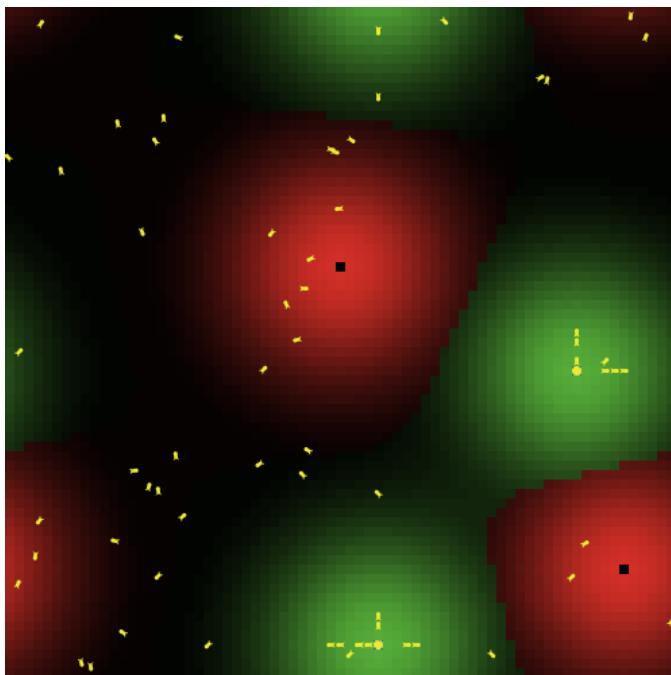
Diffusion

- ◆ Primitive de diffusion: `diffuse <variable> <coeff>` (associée à l'observateur)
 - `diffuse chemical 0.40`
 - chaque patch diffuse 40% de sa variable `chemical` ; à ses 8 patches voisins. Donc, chaque patch voisin reçoit 1/8 de 40% de la variable `chemical` (chaque patch voisin reçoit 5% de la valeur du patch diffusant)

Champs de potentiels en NetLogo

◆ Construction d'un paysage

- Primitive
 - ☞ diffuse <attribut de patche> <coeff>
 - ☞ Partage sa valeur de <coeff> avec ses voisins



◆ Suivi de gradient

- Primitive
 - ☞ uphill <attribut de patch>
 - ☞ Avance la tortue dans le patch dont la valeur de l'attribut est la plus élevée.

Equivalence avec d'autres primitives

- ◆ **Uphill <*variable*>**

Est équivalent à

```
let p max-one-of neighbors [ <variable> ]
  if [<variable>] of p > <variable>
    [ face p
      move-to p ]
```

Préférences

- ◆ On préfèrera:

```
let p max-one-of neighbors [ <variable> ]
  if [<variable>] of p > <variable>
    [ face p
      fd 1]
```

- ◆ à Uphill

Evaporation

◆ La vitesse de disparition des odeurs...

set odeur odeur * (100 – taux) / 100

- A chaque tour, le patch perd taux (en pourcentage) de sa valeur d'odeur.