



# Intelligence artificielle et les défis en robotique mobile et autonome

Éric Beaudry

<http://planiart.usherbrooke.ca/~eric/>

*Étudiant au doctorat en informatique  
Laboratoires Planiart et Laborius  
13 février 2009 – Exposé IFT615*

# Sujets

- Robotique fixe vs mobile
- Applications robotiques mobiles
- Chaîne de traitement (des capteurs aux actionneurs)
- Localisation
- Intégration et architecture décisionnelle

# Robotique fixe vs mobile



# Caractéristiques de la robotique mobile et autonome

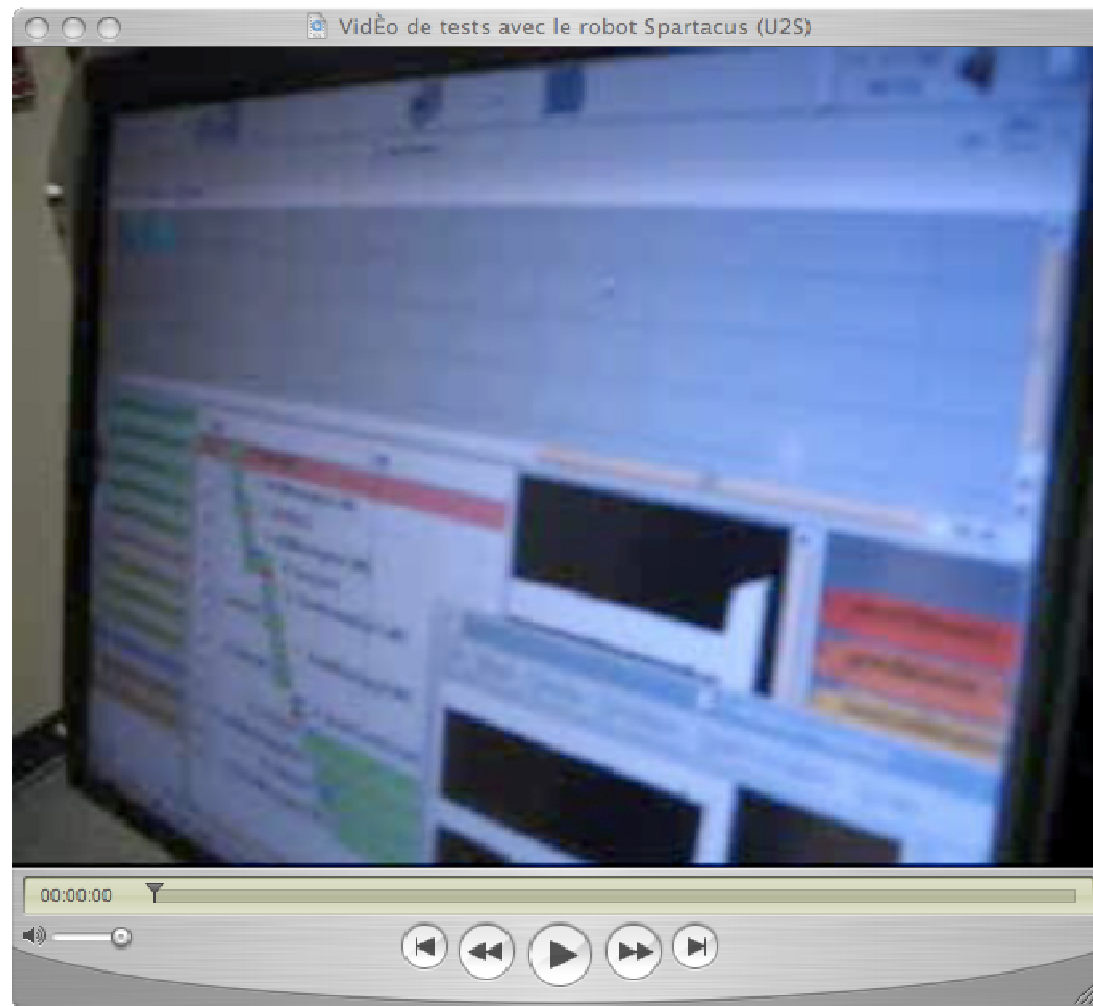
- Environnement partiellement connu
- Environnement dynamique
- Imprécision des capteurs (bruits)
- Les modèles de représentation ne sont que des approximations de la réalité
- Temps réel
- Capacité de calcul limitée
- Autonomie énergétique

# Spartacus (U2S), édition 2006



- Conçu au Laborius (UdeS)
- Planification autonome des tâches (mission)
- Navigation et localisation (CARMEN)
- Localisation et séparation de sources sonores (8 micros)
- Reconnaissance vocale des sources séparées (CSLU)

# Spartacus (U2S)



# Spartacus (U2S)



Plus de 42 000 images en accéléré



# AAAI 2006





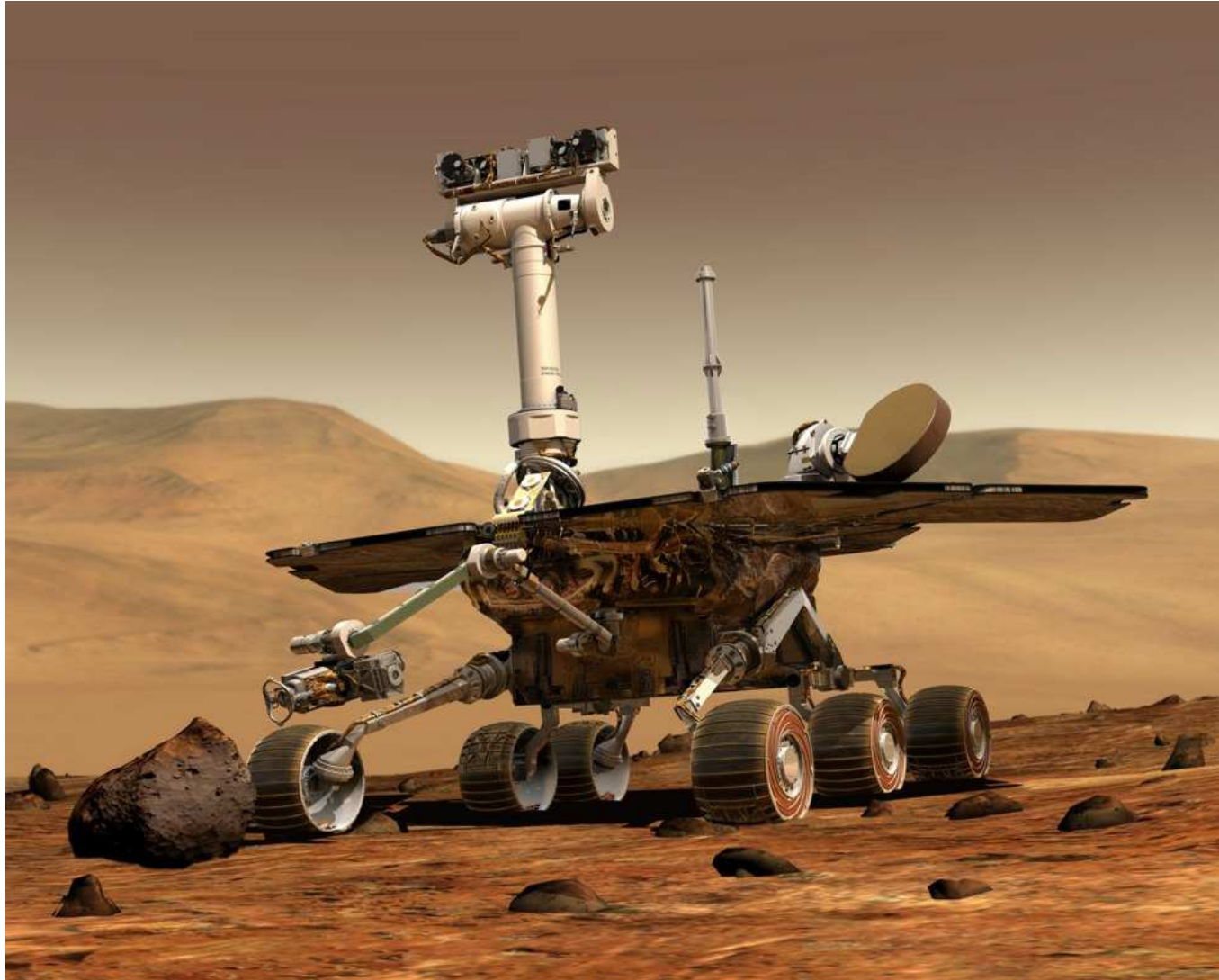
# AAAI\* Robot Competition

- Scavander Hunt
  - Recherche d'objets dans un environnement
- Open Interaction (OI)
  - Interaction avec les participant de la conférences
  - Divertissement
  - Reconnaissance vocale, de visage, de gestes, etc.
  - Capacité de réagir
- AAAI Robot Challenge (intégré à OI en 2006)
  - Robot participant à la conférence
  - Divertir les participants
  - Tâches bénévoles : livrer messages ou objets, surveillance, etc.
  - Prendre des photos
  - Donner une conférence : résumer son expérience et fonctionnement interne
  - Intégration de plusieurs capacités robotiques

\*American Association for Artificial Intelligence : <http://www.aaai.org/>



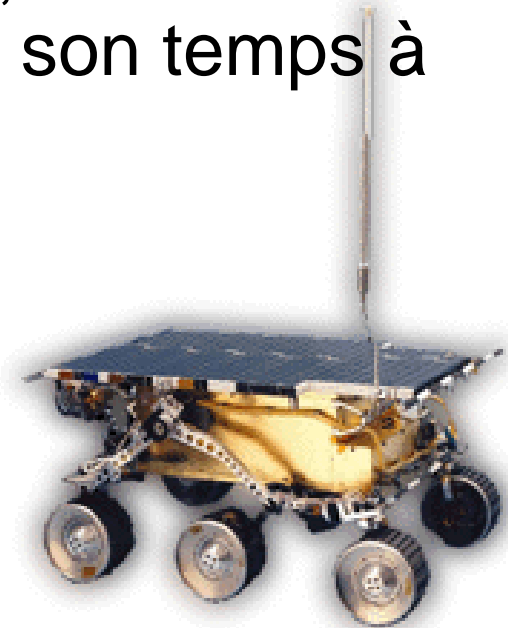
# Robots sur Mars



Source : <http://marsrovers.jpl.nasa.gov/gallery/artwork/hires/rover3.jpg>

# Application: robots sur Mars

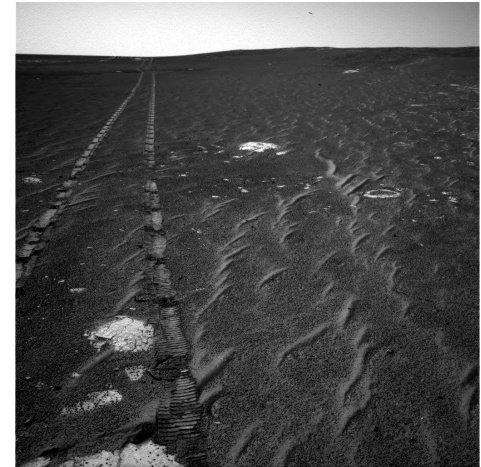
- À la recherche de traces de vie
- Distance Terre - Mars: téléopération très difficile
- Autonomie requise (hautement souhaitée)
- Durant la mission Pathfinder en 1997, le robot Sejourner passait entre 40 et 75% de son temps à ne rien faire [Bresina2002]
- Raison: échecs des plans
- Nécessité de valider les plans
- Planification sur Terre vs «on board»



*Robot Sejourner*

# Mars : contraintes et caractéristiques

- Terrain localement inconnu, relief très accidenté.
- Localisation difficile (pas de systèmes GPS). Structured-Light (Pathfinder) ou Stereovision (MER).
- Énergie (panneaux solaires).
- Batteries.
- Puissance de calcul.
- Espace de stockage.
- Aucune intervention possible.
- Fenêtres de communication.
- Protocole d'utilisation des instruments et capteurs.
- Survie au froid.



# Mars : incertitude

- Une grande partie de l'incertitude est liée aux ressources et au temps:
  - Énergie captée par les panneaux solaires;
  - Énergie consommée par les moteurs;
  - Durée des déplacements;
  - Communications avec la terre;
  - Espace de stockage requis;
- Bref, le plan d'activités du robot doit être soigneusement élaboré et validé avec d'être exécuté. Sinon, le robot risque de «mourir» !



# Véhicules autonomes

- Bilan routier 2006 Québec :
  - 717 décès
  - 46 012 blessés dont 3714 avec blessures sévères
- Traffic Accidents (USA, 2004)
  - 42,636 dead
  - 2,788,000 injured
  - Leading cause of death, age 3-33
- Mission canadienne en Afghanistan
  - Une grande partie des soldats tués au combat sont sur des convois de transport





# DARPA Challenge

- En 2001, le Congrès des É.-U. a voté une motion demandant qu'un tiers des véhicules terrestres de l'armée soit conduit de façon autonome d'ici 2015 (*"A third of all ground vehicles unmanned by 2015"*).
- Suite à cette motion, le DARPA (Defense Advanced Research Projects Agency) a mis sur pied :
  - Grand Challenge (2005)
  - Urban Challenge (2006)



# Tartan Racing (CMU - Carnegie Mellon University, Pittsburgh)



# Stanford Racing Team (Stanford University, Californie)



<http://cs.stanford.edu/group/roadrunner/index.html>



# Laser Range Data Integration



# DARPA Urban Challenge



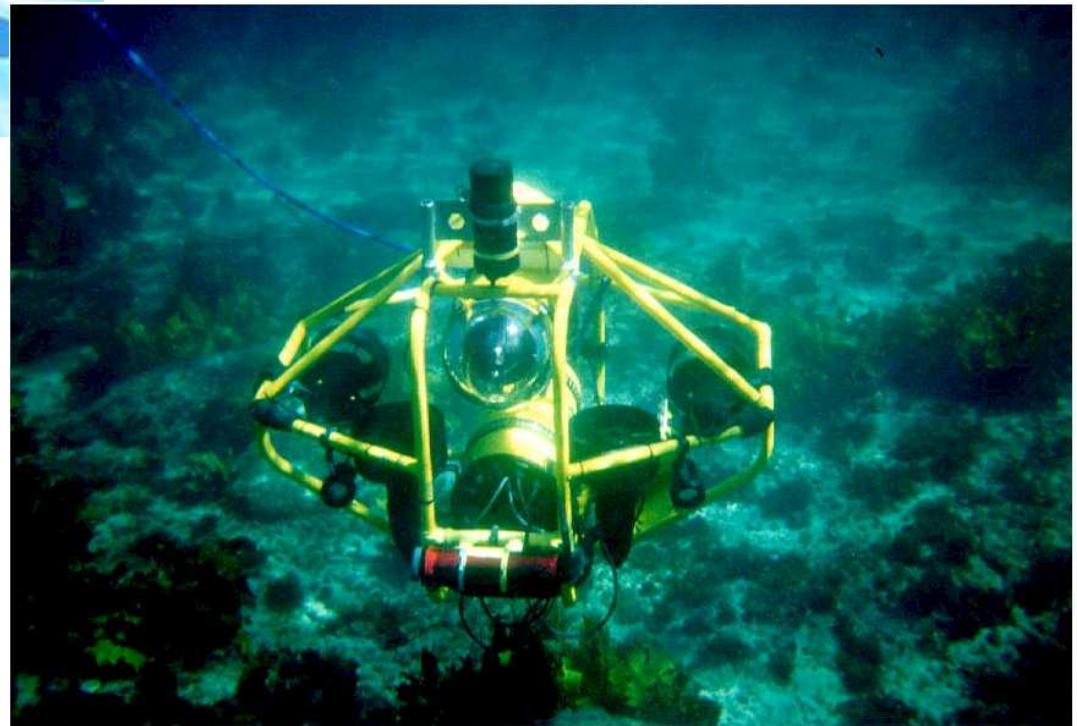
# UAV (Unmanned Aerial Vehicle)

- Missions de reconnaissance et de surveillance.





# Robot sous-marin (Underwater)



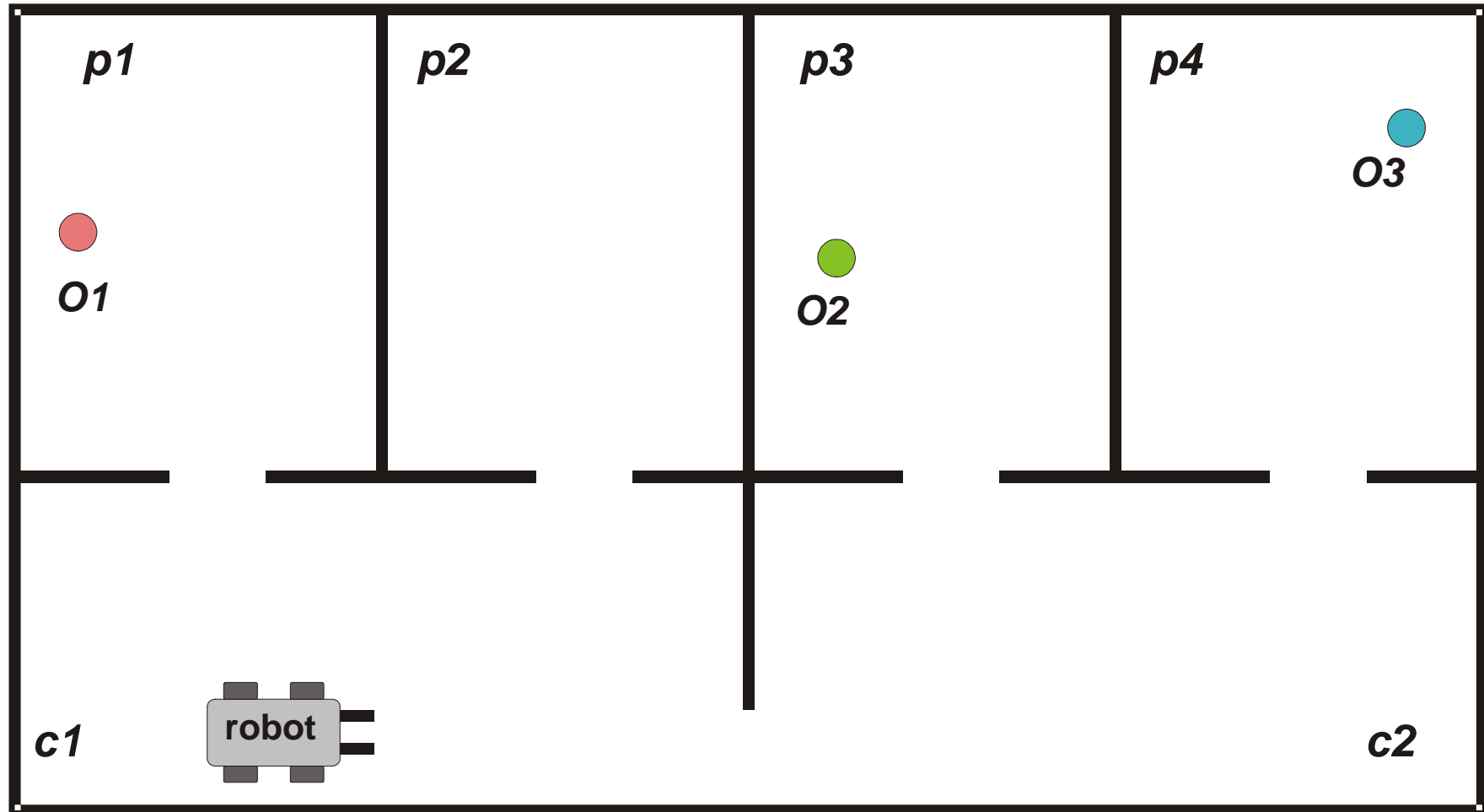
# Application: observation de la Terre

- **Conditions d'acquisition (météo) incertaines** (très problématique pour les données optiques)
- Des requêtes urgentes peuvent survenir
- Les fenêtres de communications sont limitées
- Capacité de stockage limitée sur les satellites
- Les changements d'orbite sont coûteux
- **Volume de données incertain**
- **Besoin de planifier les actions pour optimiser les acquisition de données**

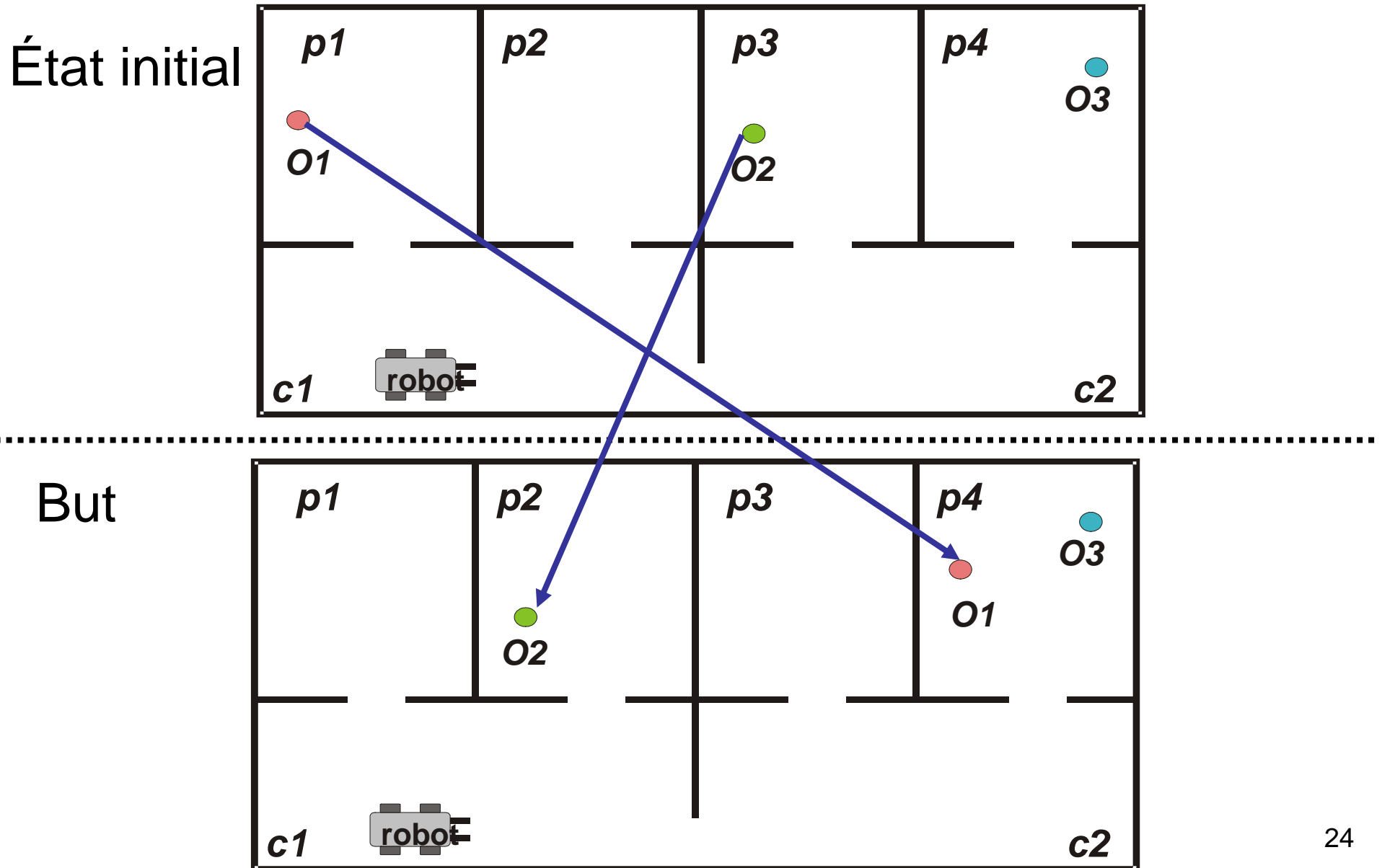


*RadarSat II*

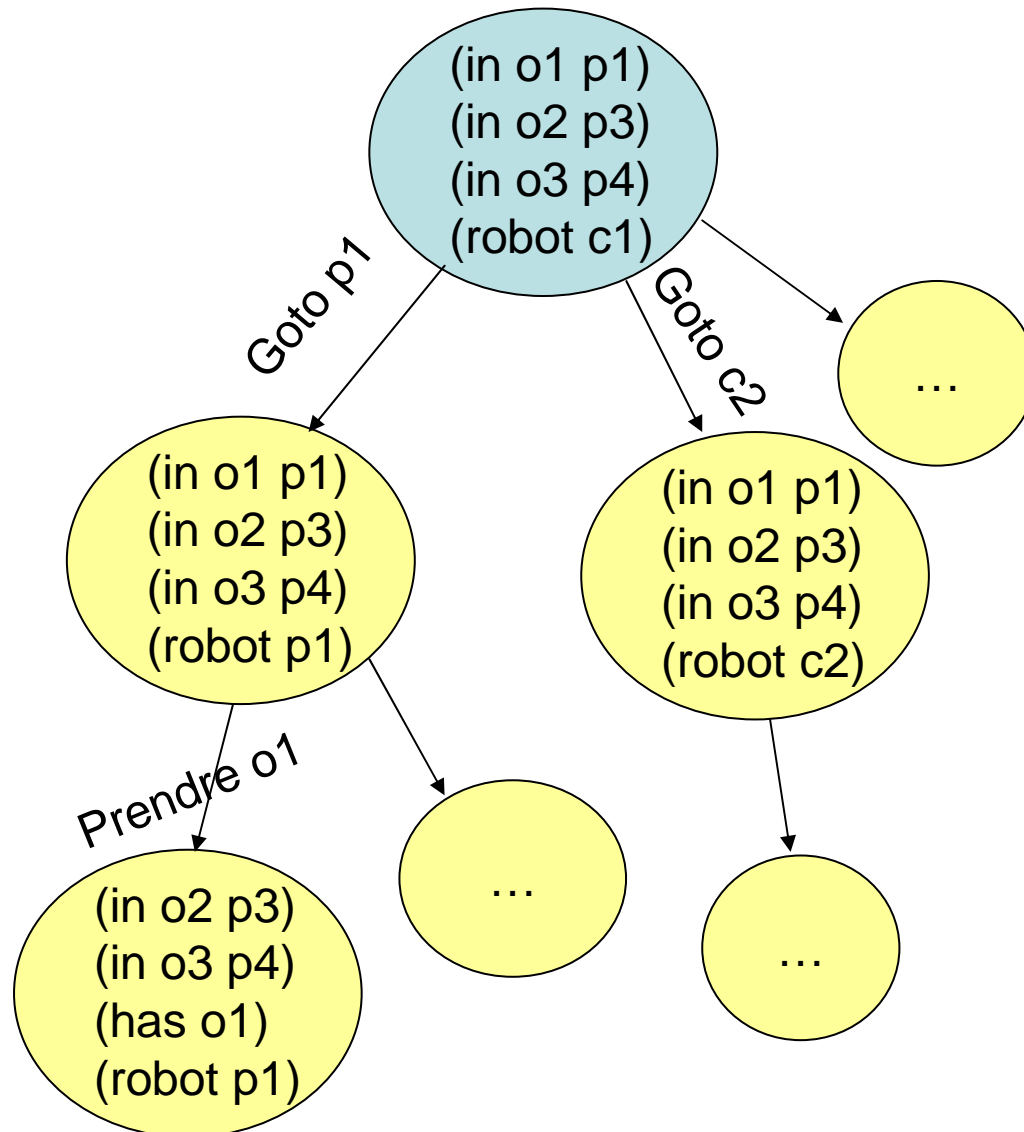
# Application classique: un robot livreur de colis



# Un robot livreur de colis : problème



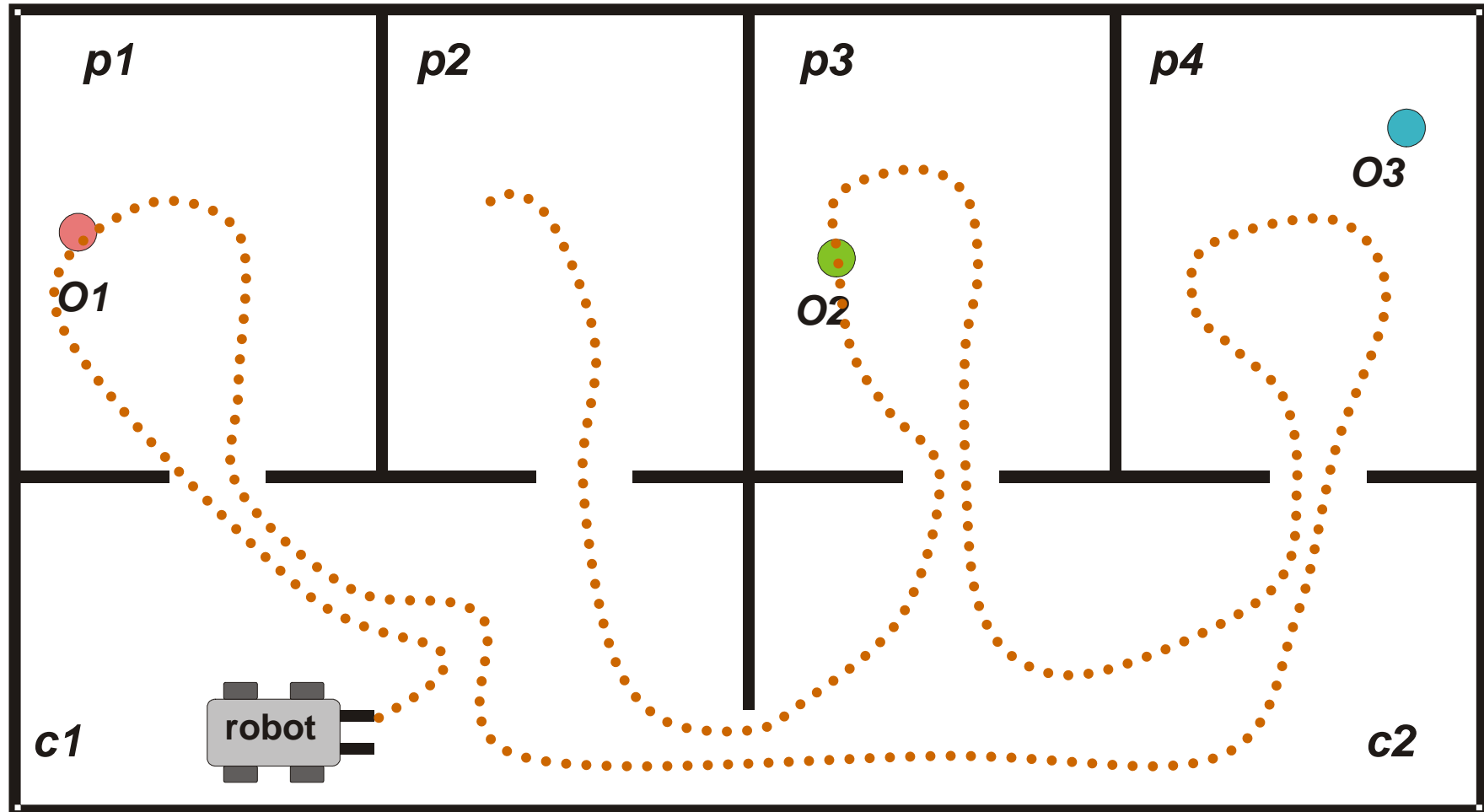
# Un robot livreur de colis : A\*



## Solution:

1. Goto p1
2. Prendre o1
3. Goto c1
4. Goto c2
5. Goto p4
6. Déposer o1
7. Goto c2
8. Goto p3
9. Prendre o2
10. Goto c2
11. Goto c1
12. Goto p1
13. Déposer o2

# Un robot livreur de colis : exécution





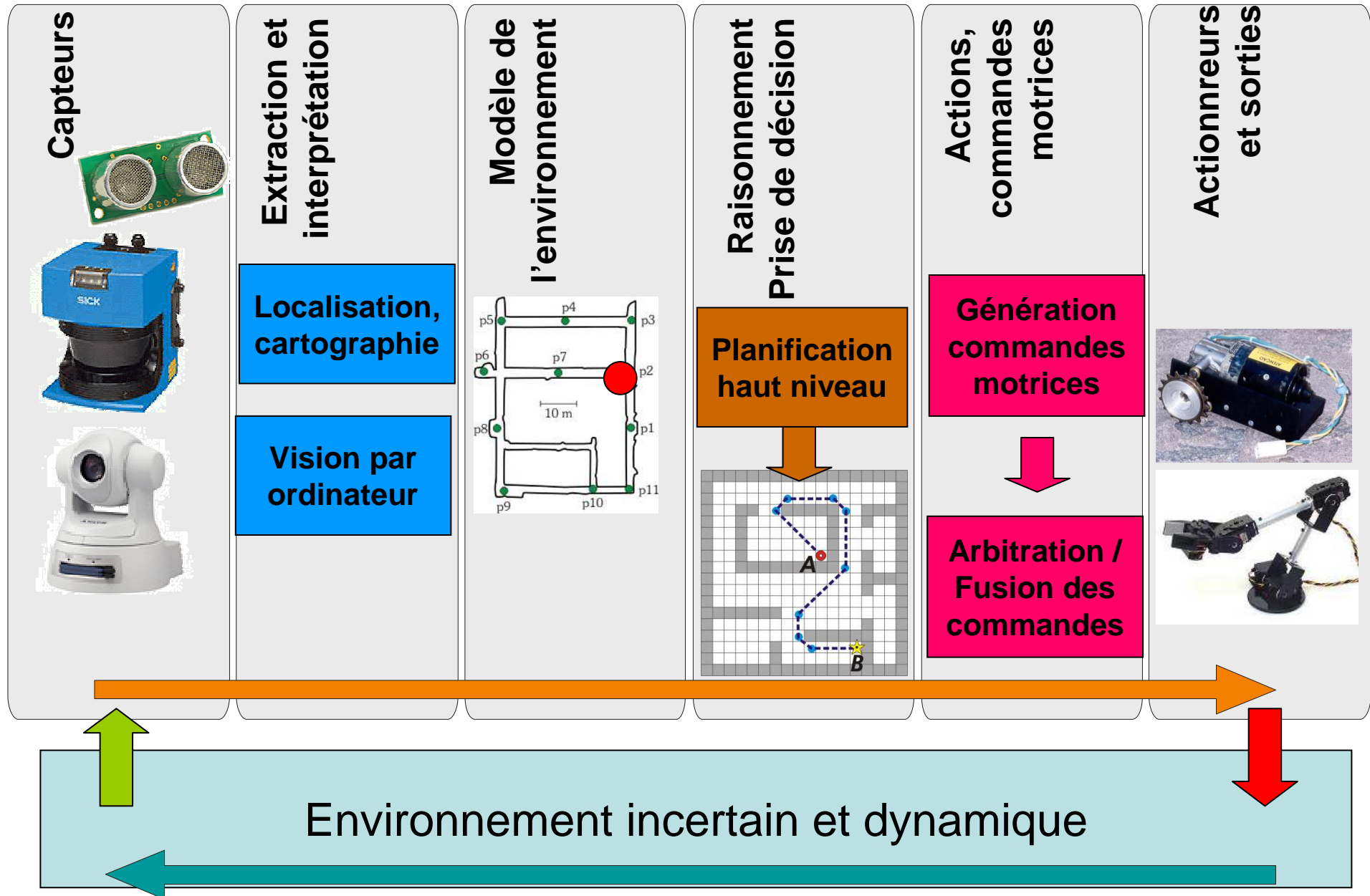
# Incertitude

- Que se passe-t-il si ...
  - Des obstacles se trouvent sur le chemin du robot ?
  - Une porte est fermée ?
  - On ajoute des contraintes de temps ?
  - Chute de tension des batteries ?
  - ...
- Même s'il est pratiquement impossible de tout prévoir, il faut «gérer» l'incertitude.

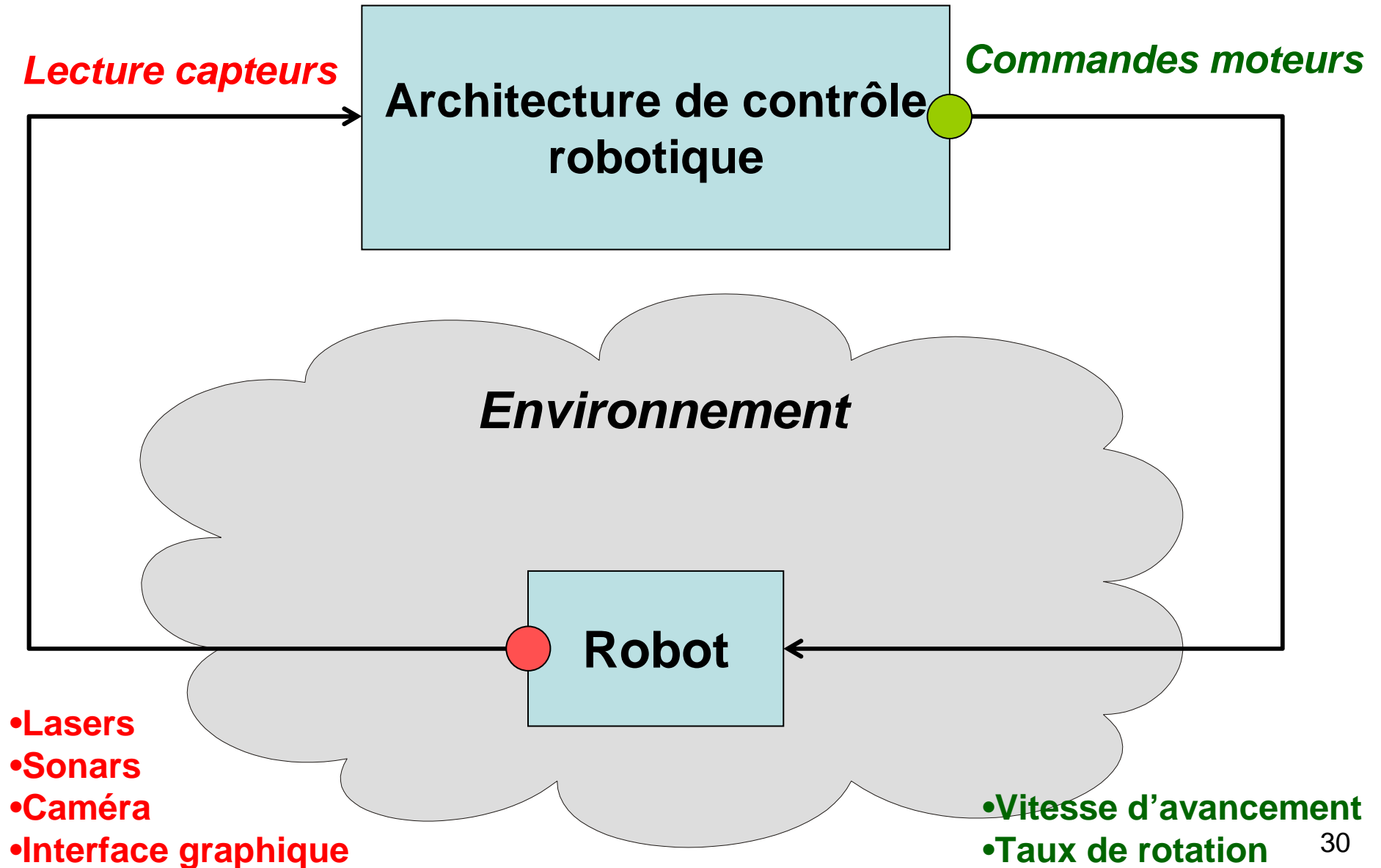
# Approches pour gérer l'incertitude

- Architectures décisionnelles
  - Prendre des décisions à plusieurs niveaux
  - Réactif vs délibératif
- Planification déterministe avec monitoring
  - Surveillance des plans
  - Replanification lors d'échecs perçus ou anticipés
- Planification non déterministe
  - Génération de politiques (*Markov Decision Process*)
  - Génération de plans conditionnels

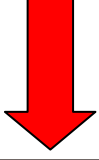
# Chaîne de traitement



# Boucle de contrôle



# Capteurs



**Capteurs**

**Extraction et  
interprétation**

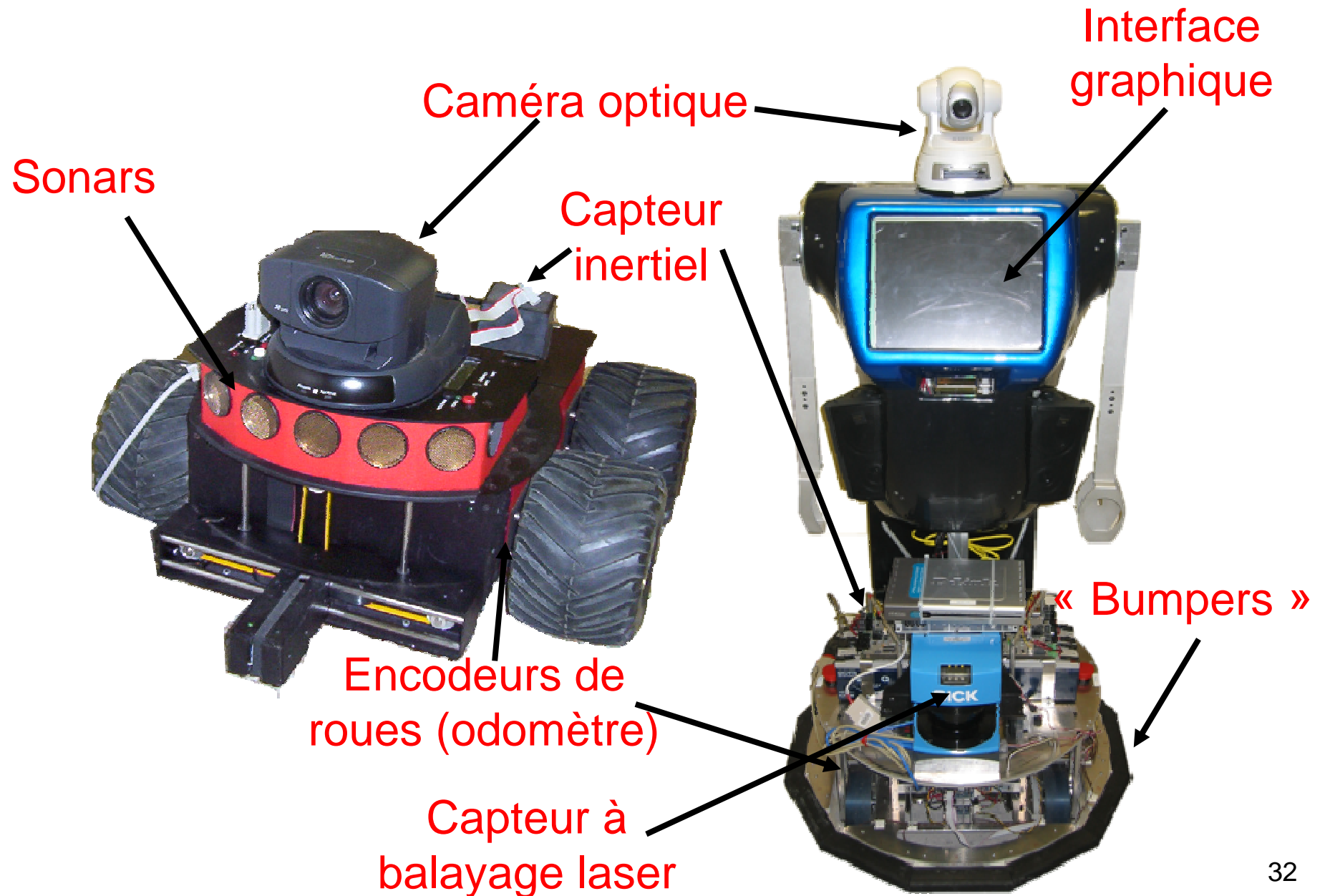
**Modèle de  
l'environnement**

**Raisonnement  
Prise de décision**

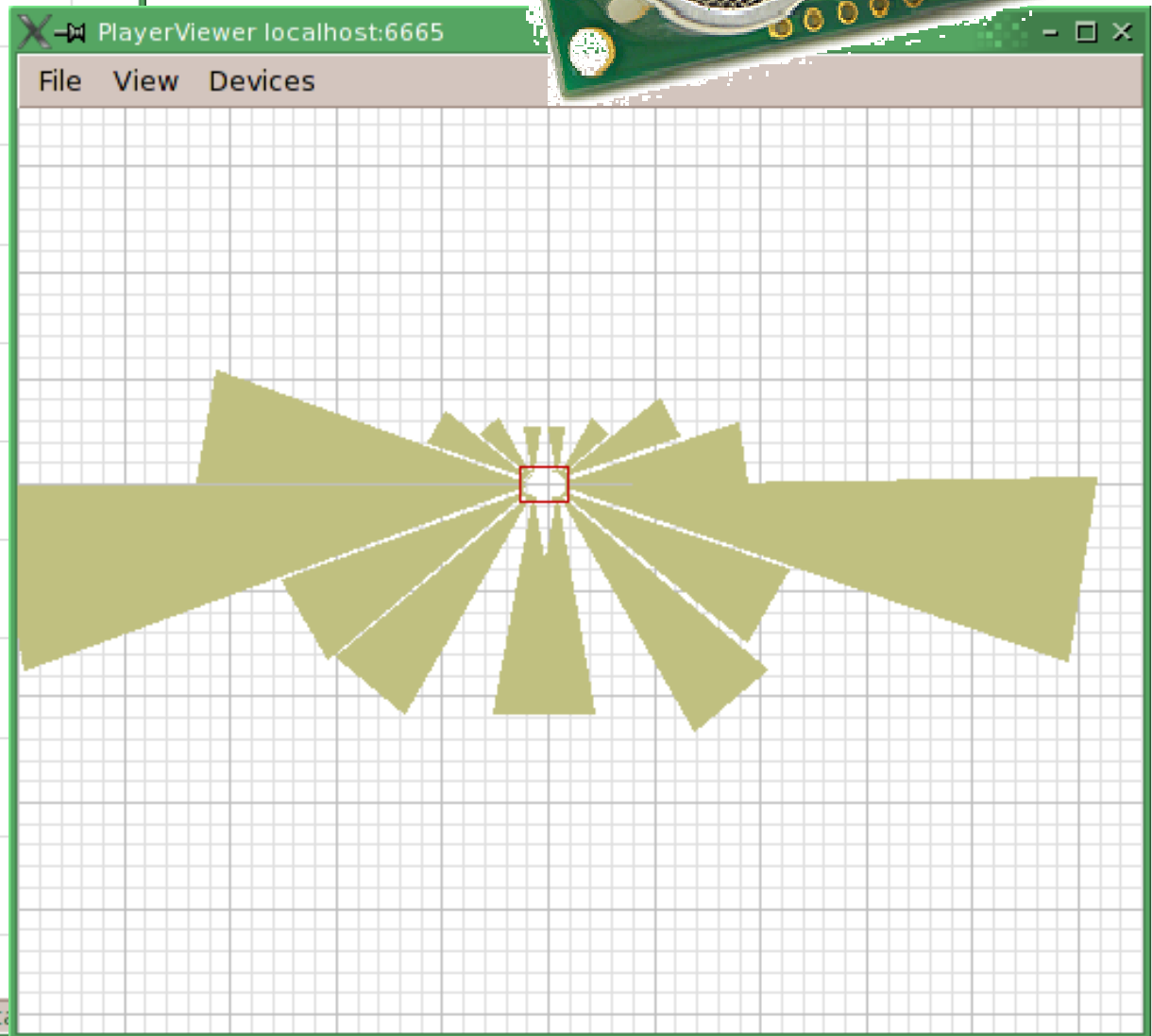
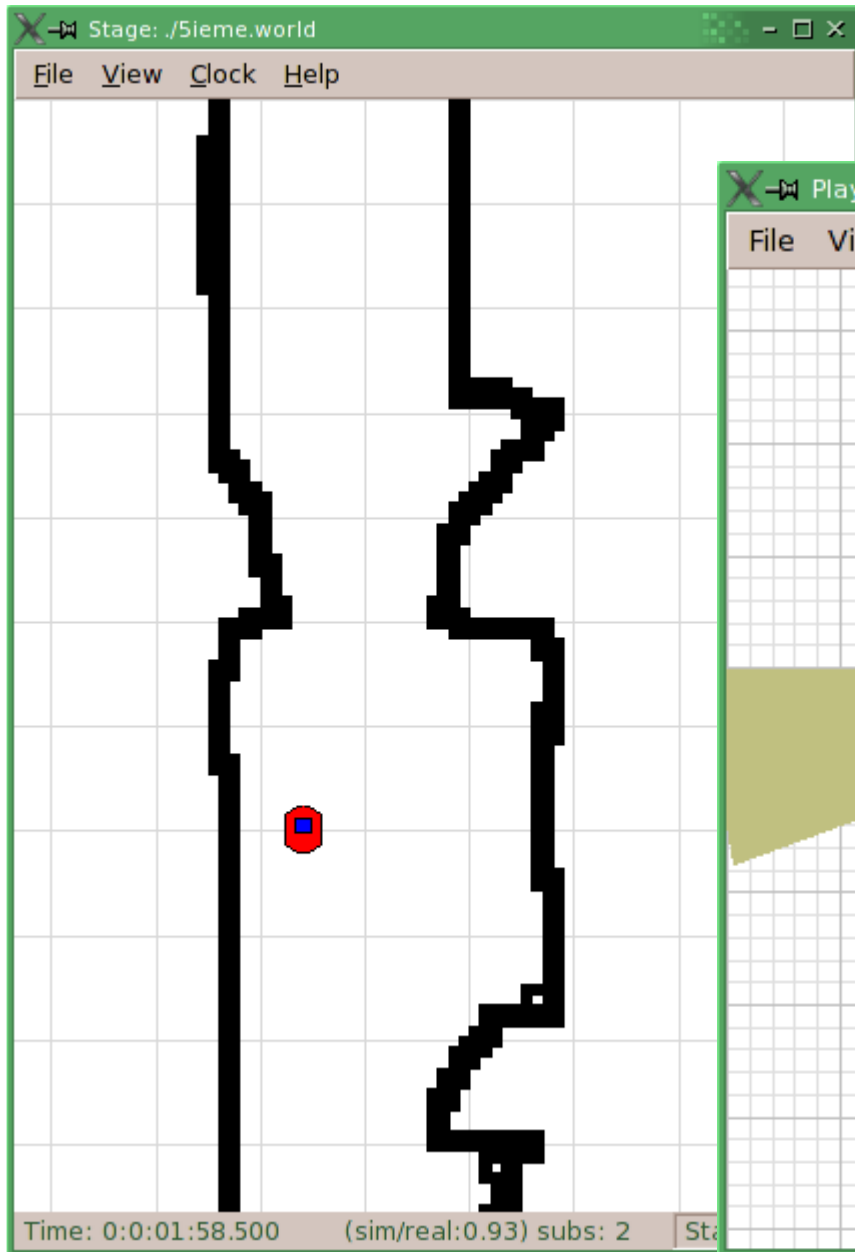
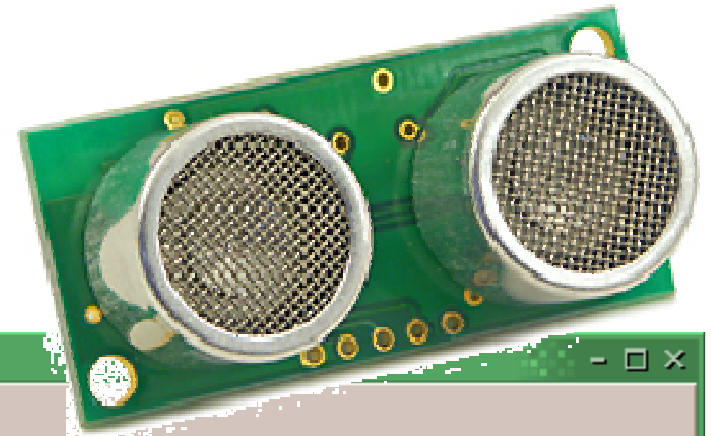
**Actions,  
commandes  
motrices**

**Actionneurs  
et sorties**

# Capteurs d'un robot

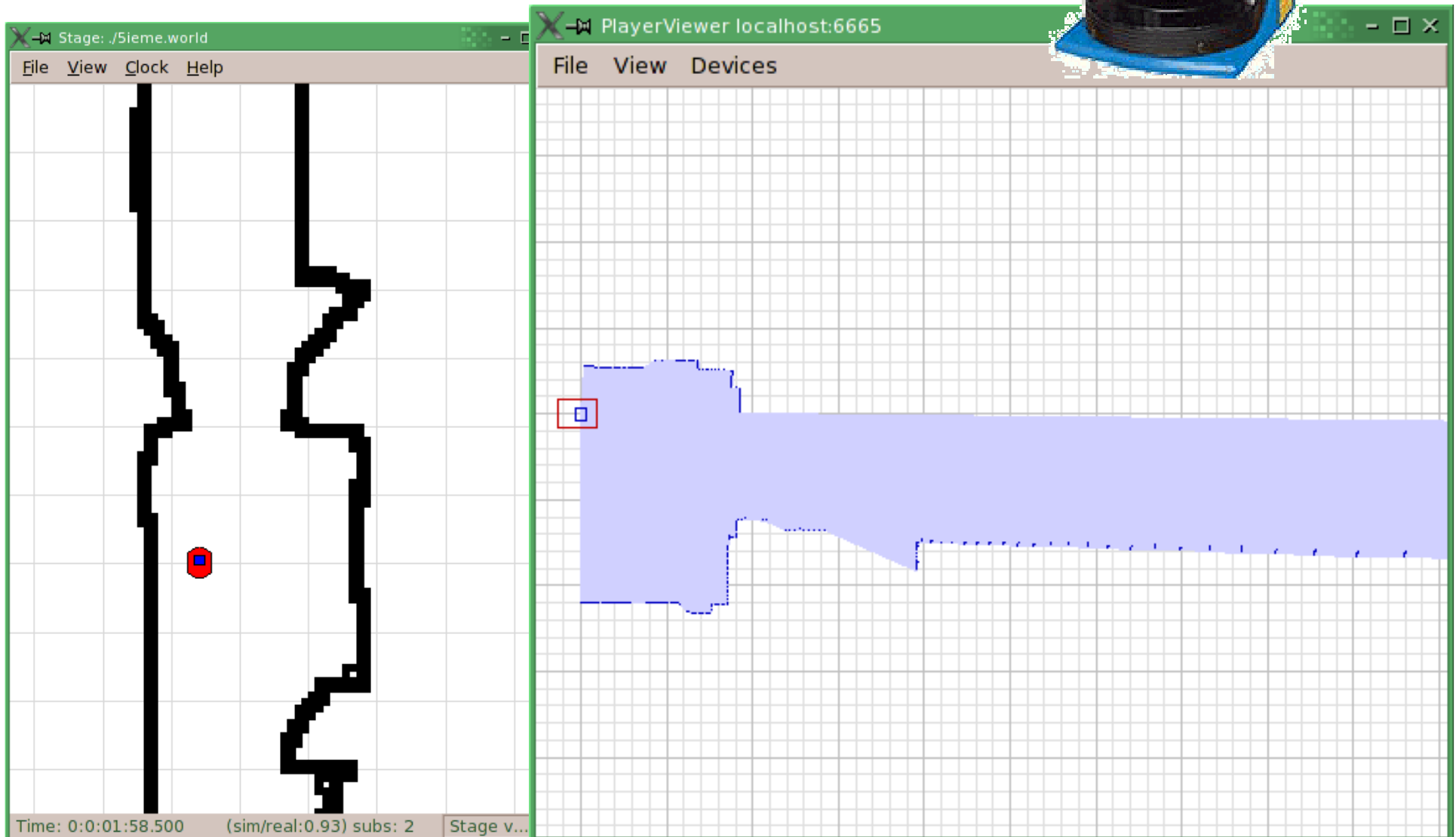


# Sonars à ultrasons





# Capteur de proximité à balayage laser



Stage: ./Sieme.world

File View Clock Help

Time: 0:0:04:33.800 (sim/real:0.92) subs: 2 Stage v...

The Stage window displays a 2D environment on a grid. A light blue polygonal area is bounded by a thick black line. A red diamond with a blue center is located at the bottom of the light blue area. The grid has numerical labels at the bottom: 5, 18, 17, 18, 20, 21, 22, 23.

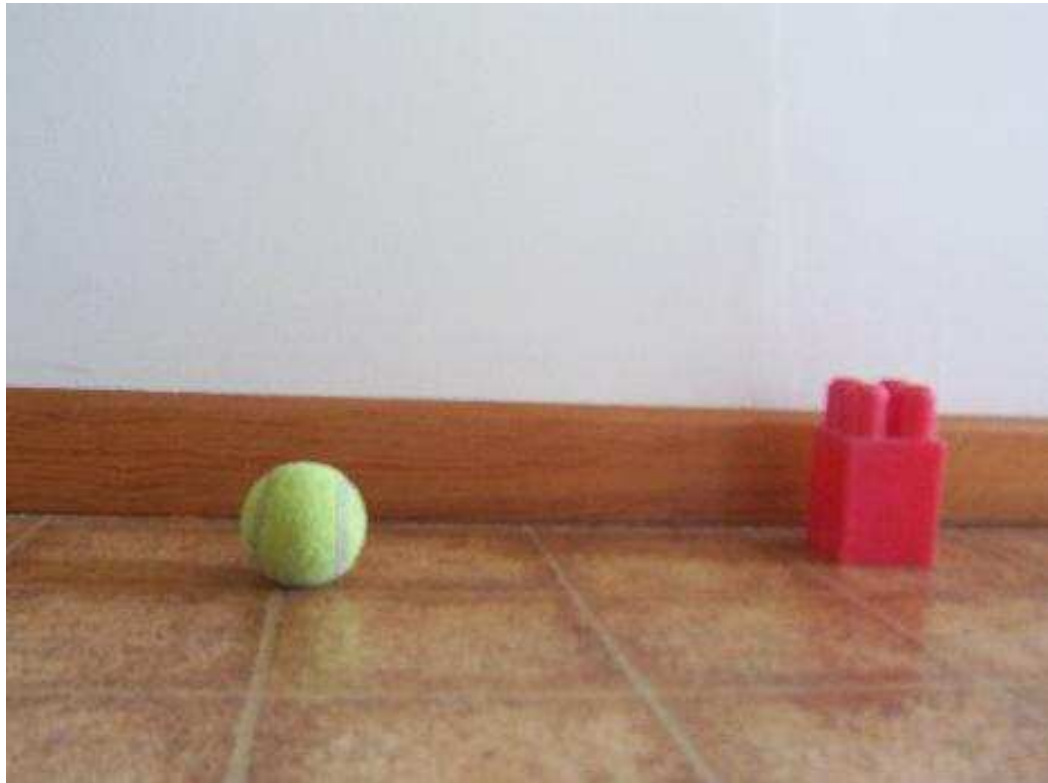
PlayerView localhost:6665

File View Devices

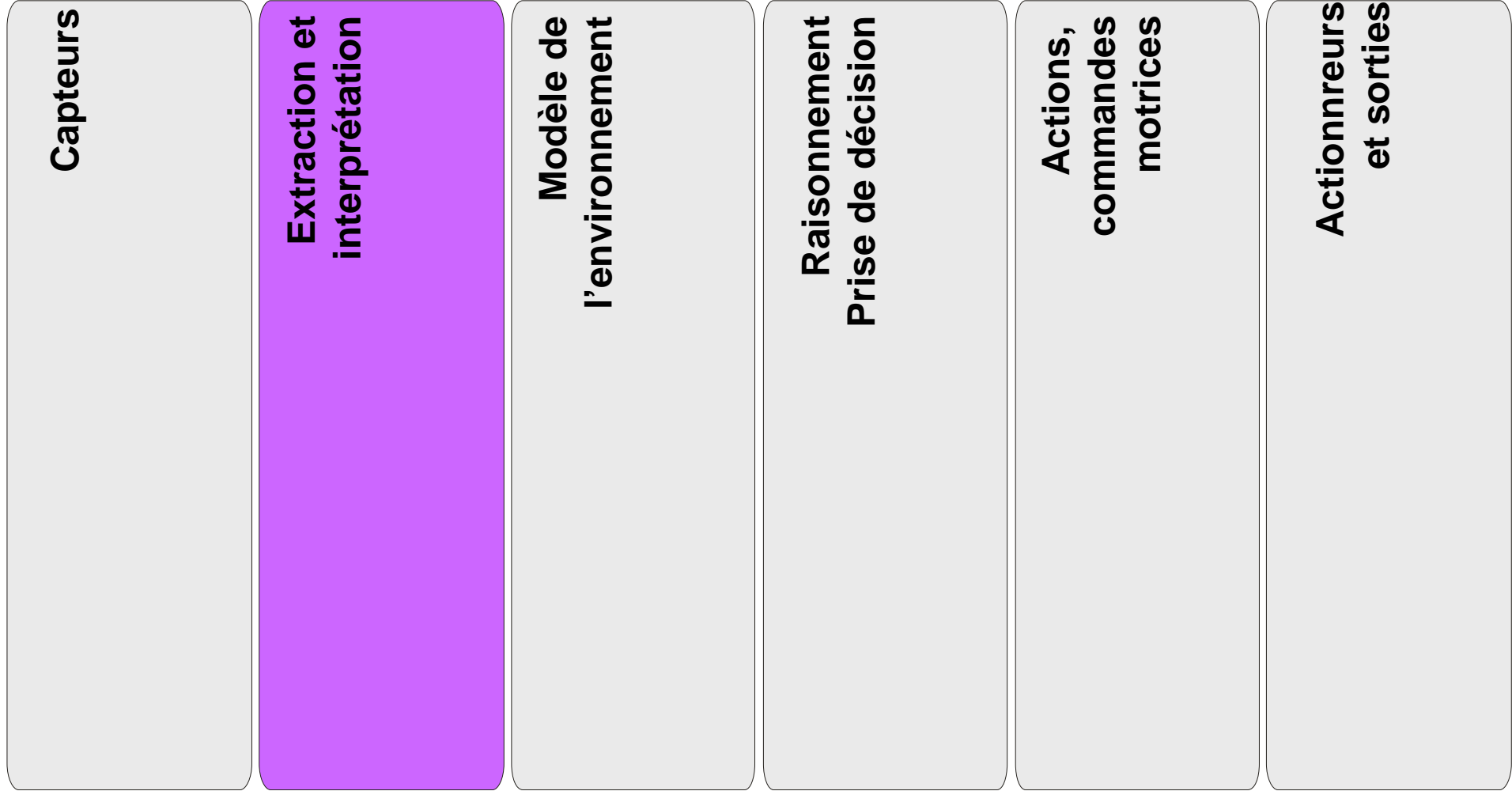
The PlayerView window shows a zoomed-in view of the light blue area from the Stage window. A small red square is positioned on the left side of the light blue area. The grid is visible in the background.



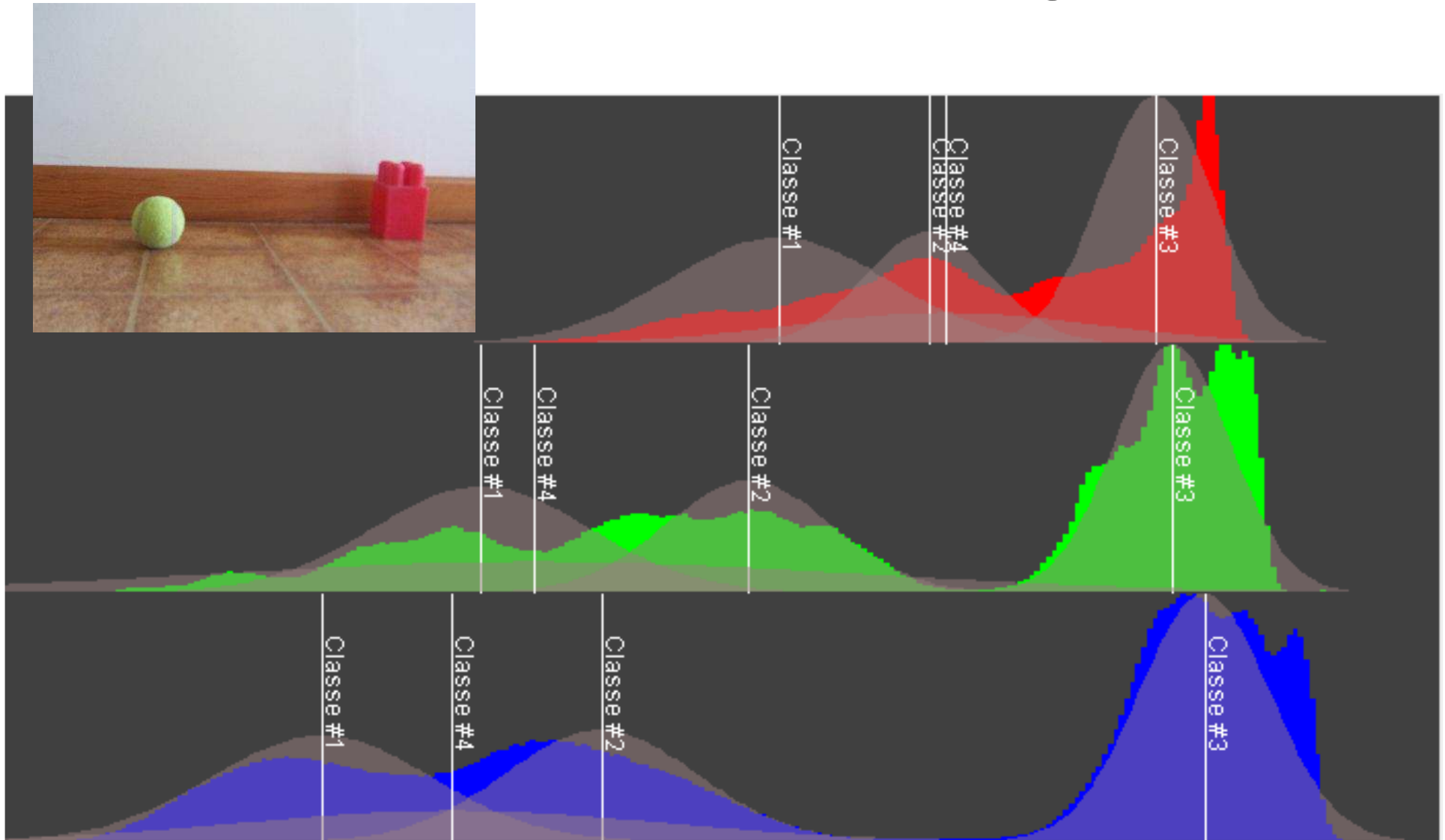
# Caméra optique



# Extraction d'information

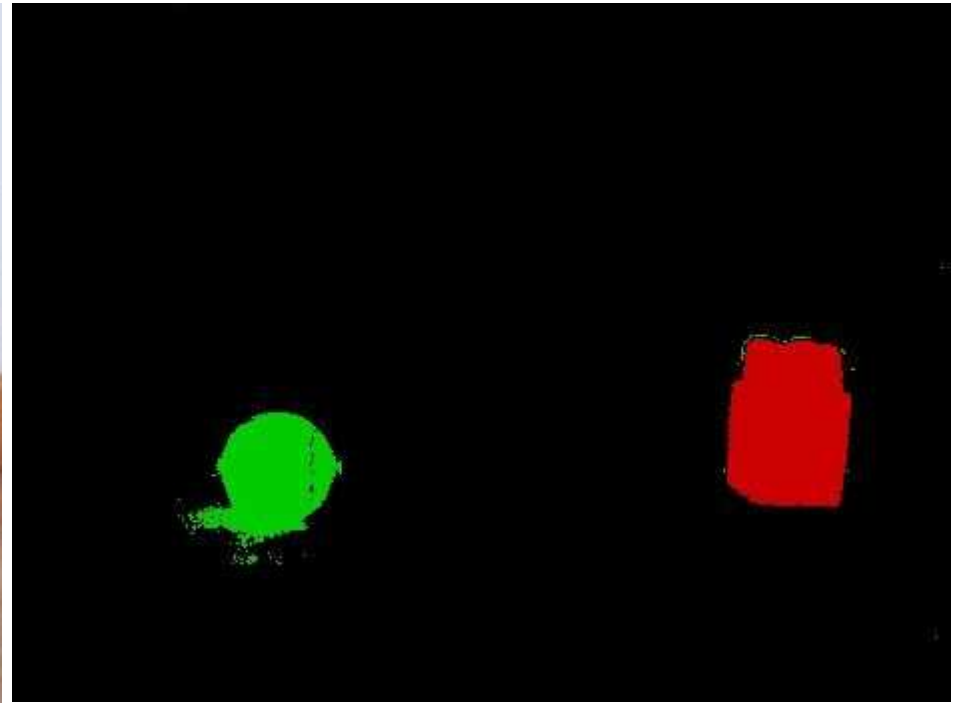
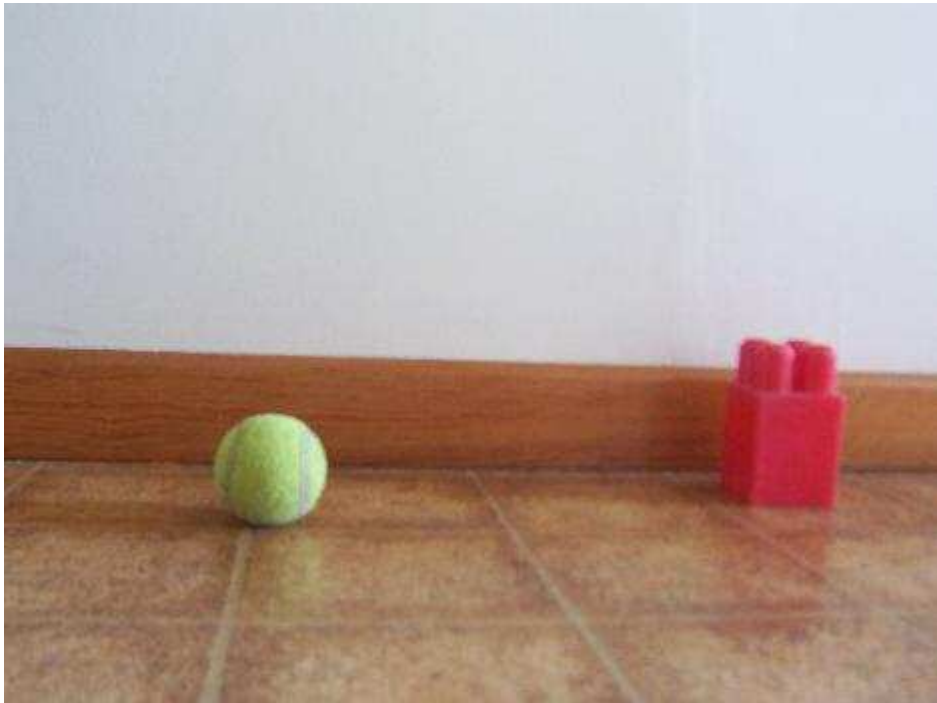


# Classification d'images

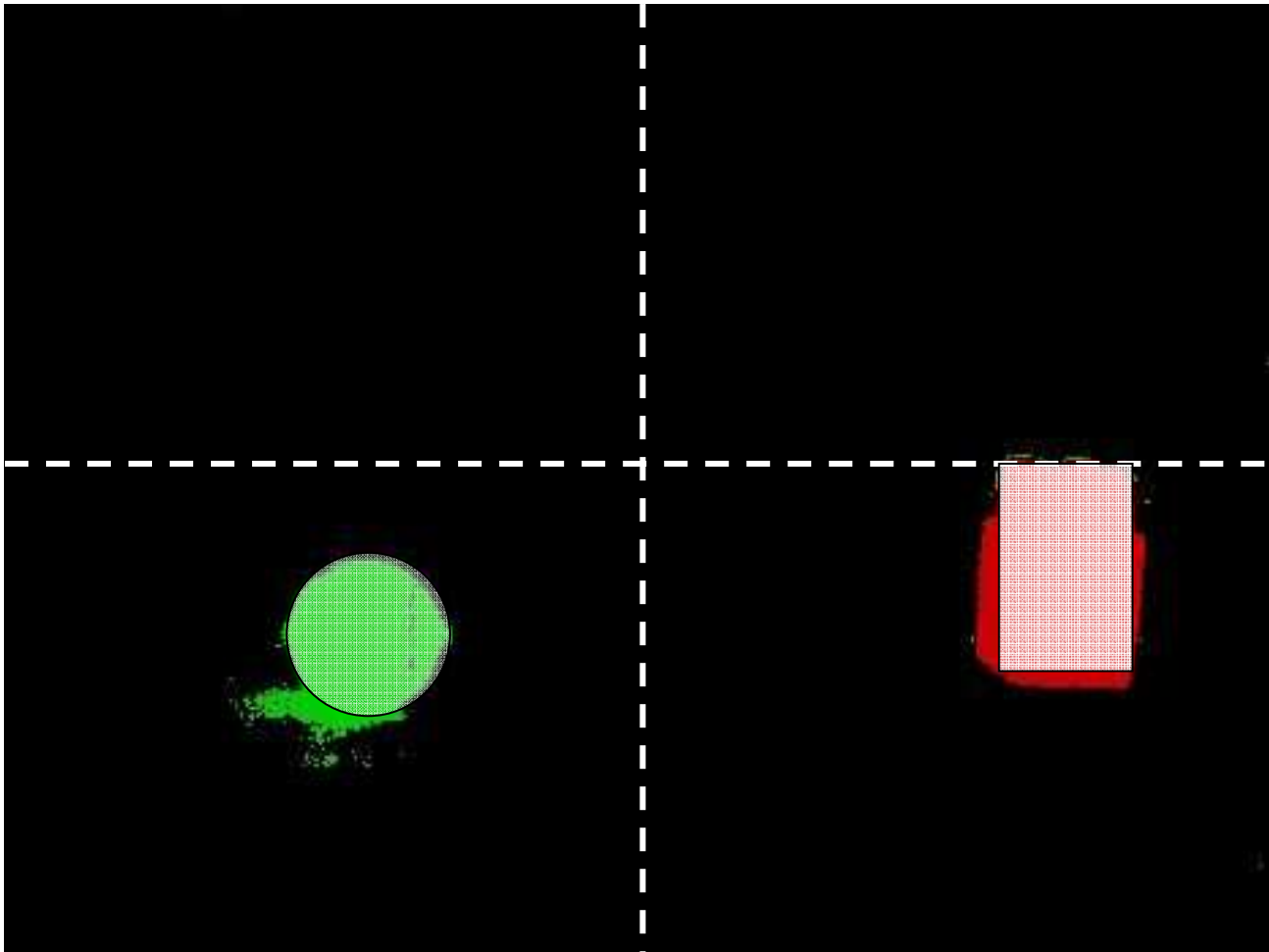




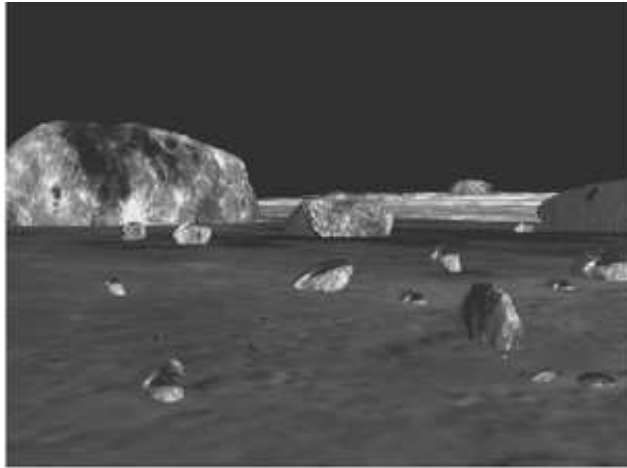
# Classification d'images



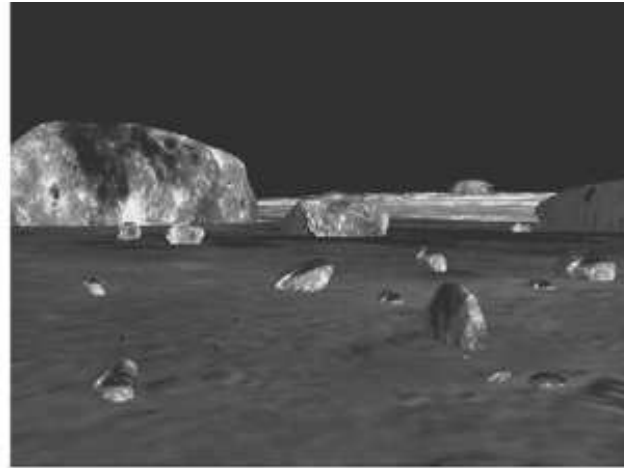
# Segmentation d'images



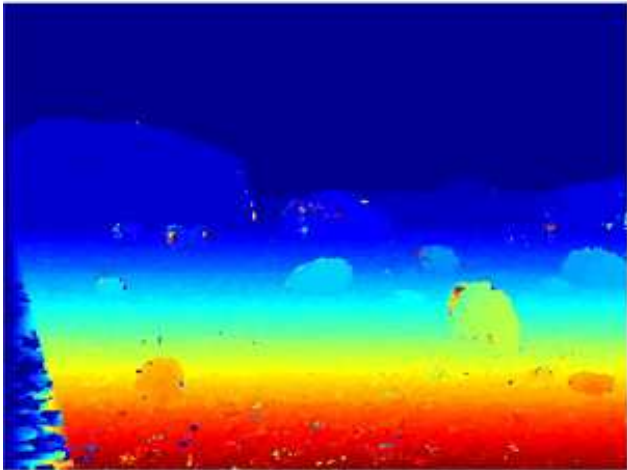
# Indices de profondeur



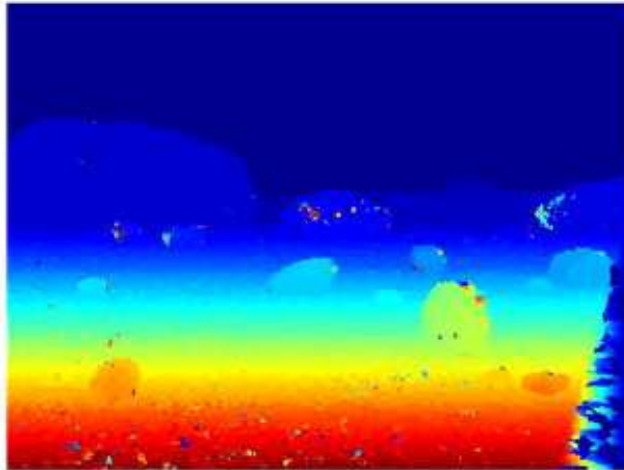
(a)



(b)

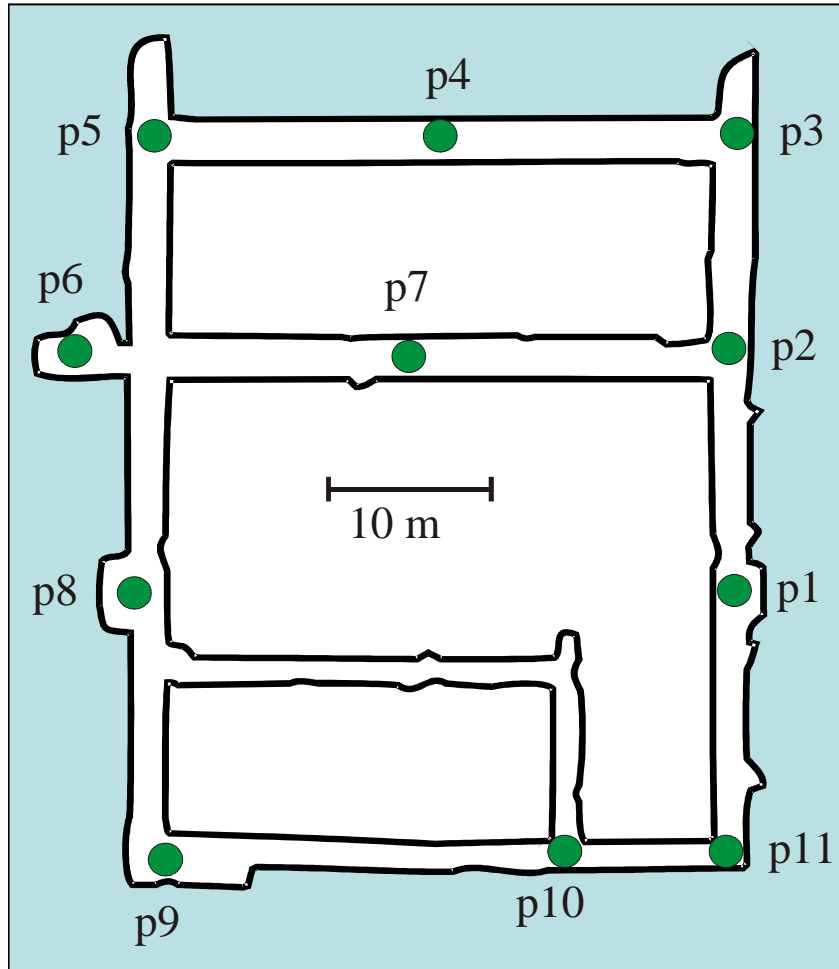


(c)

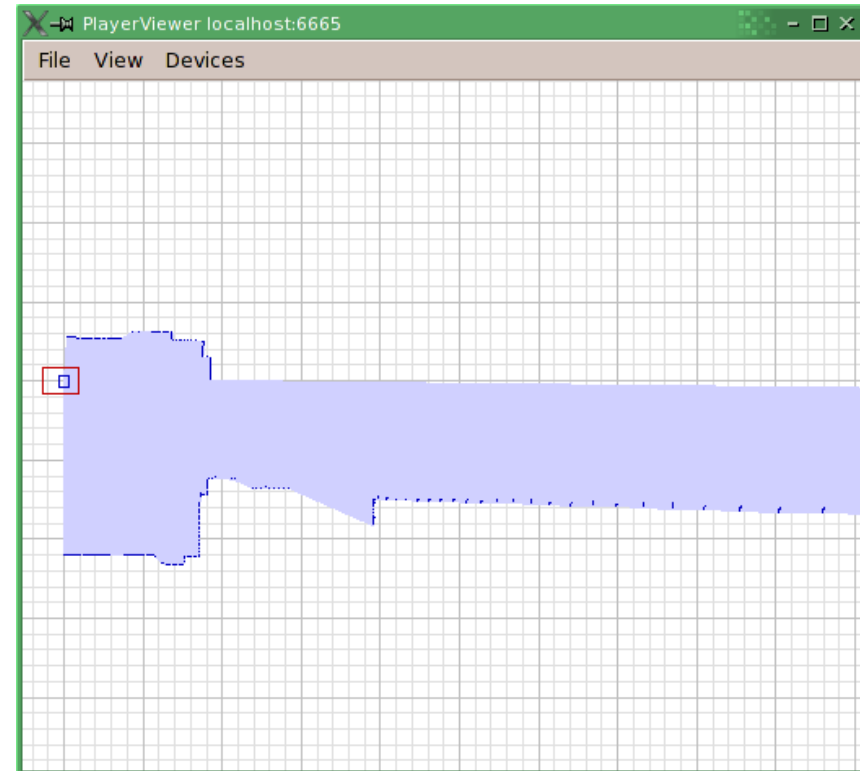


(d)

# Localisation : où suis-je ?



Carte connue



Représentation graphique de la « vue » du laser

# Localisation

- La localisation consiste à trouver ou suivre (*tracking*) la position courante d'un robot.
- Carte connue, **environnement dynamique**.
- Capteurs **imprécis, bruités et sensibles au milieu** (ex: laser et miroir).
- Généralement, on ne peut pas mesurer directement la position. Il faut plutôt l'**estimer**.
- Plusieurs méthodes:
  - Capteur GPS (extérieur)
  - Capteurs infrarouges (requiert d'altérer l'environnement)
  - **Laser + odomètre (Filtres de Kalman / Monte Carlo)**
  - ...



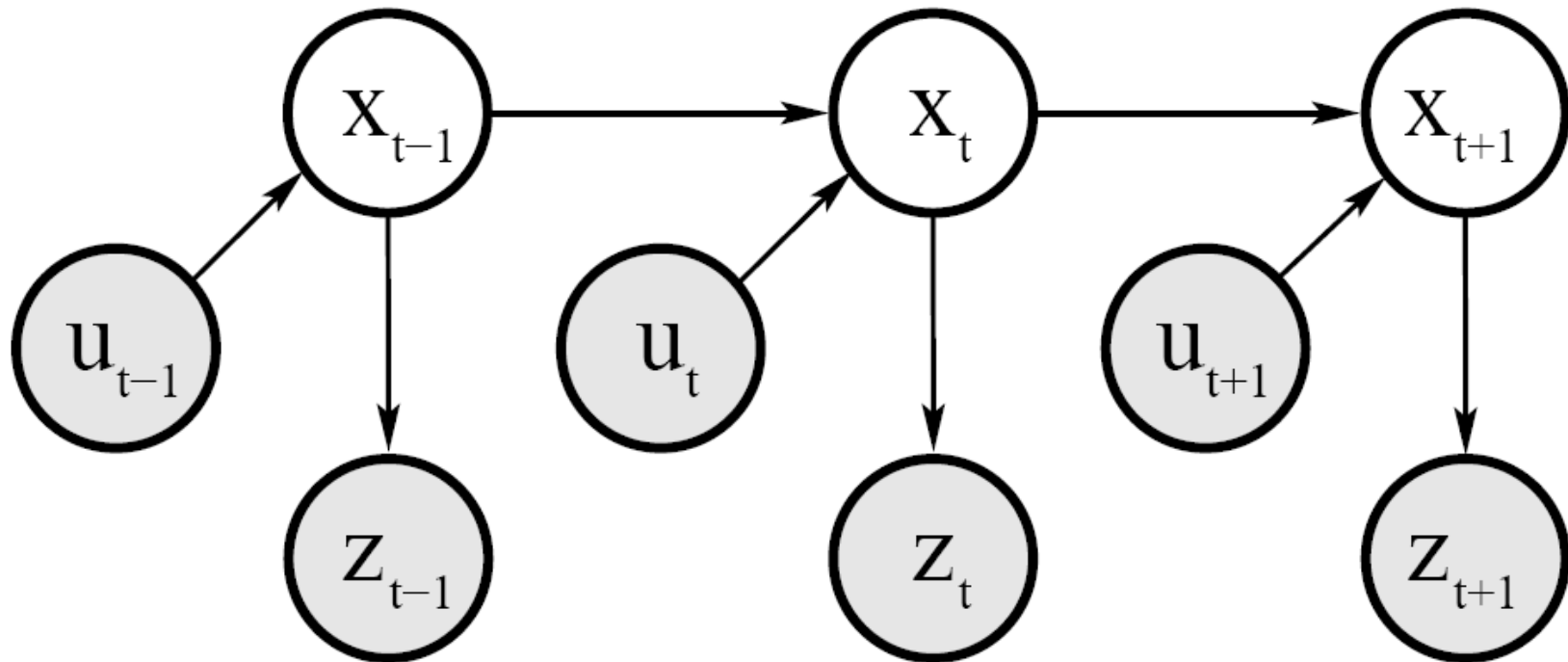
# Localisation

- Odomètre (encodeurs de roues) imprécis : glissement des roues



Source : figures tirées de <http://www.probabilistic-robotics.org/>

# Chaîne de Markov cachée

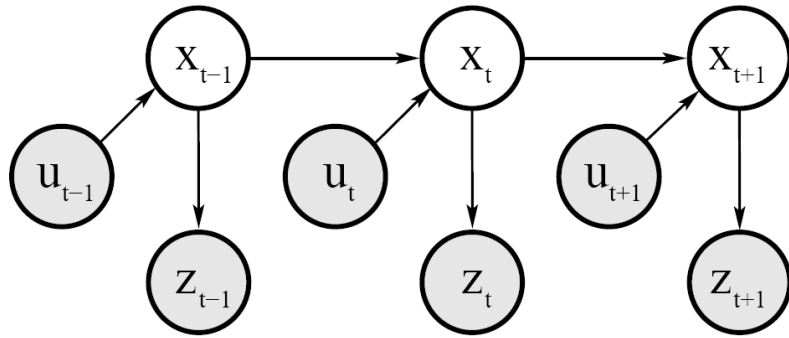


$x$  : position du robot (variable cachée)

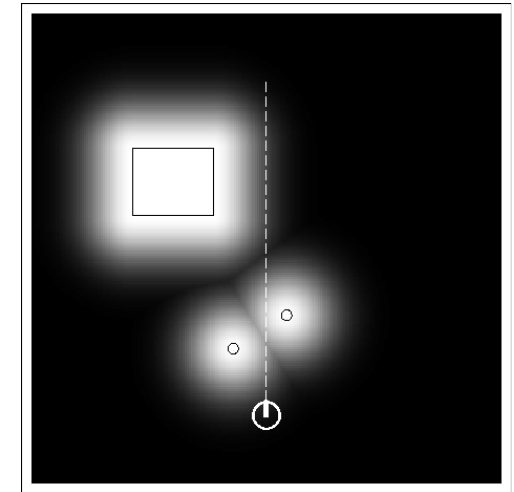
$u$  : actions (ou déplacement perçu)

$z$  : observations (capteurs)

# Modèles probabilistes

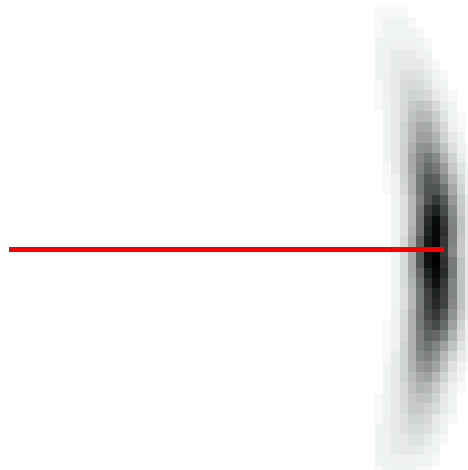


Carte m



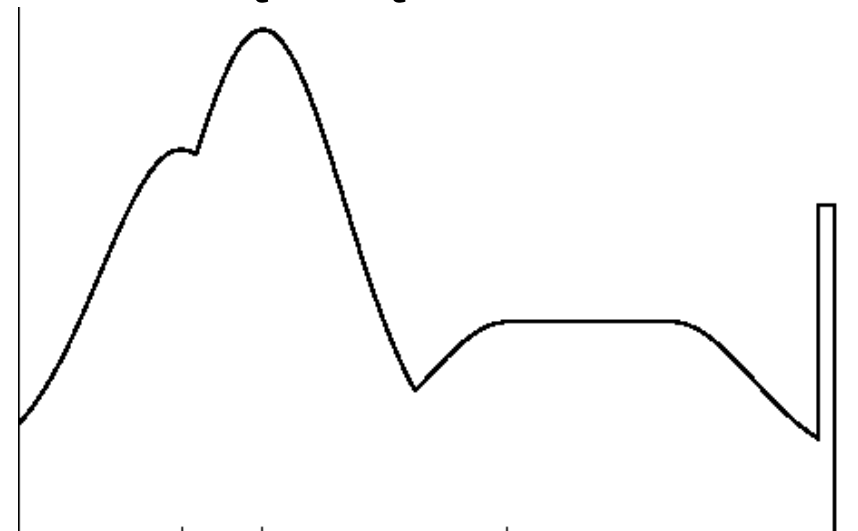
## Déplacement

$$p(x_t | x_{t-1}, u_t)$$



## Observation

$$p(z_t | x_t, m)$$



# Filtre bayésien\*

- Intégration des actions ou de l'odométrie ( $x$ )

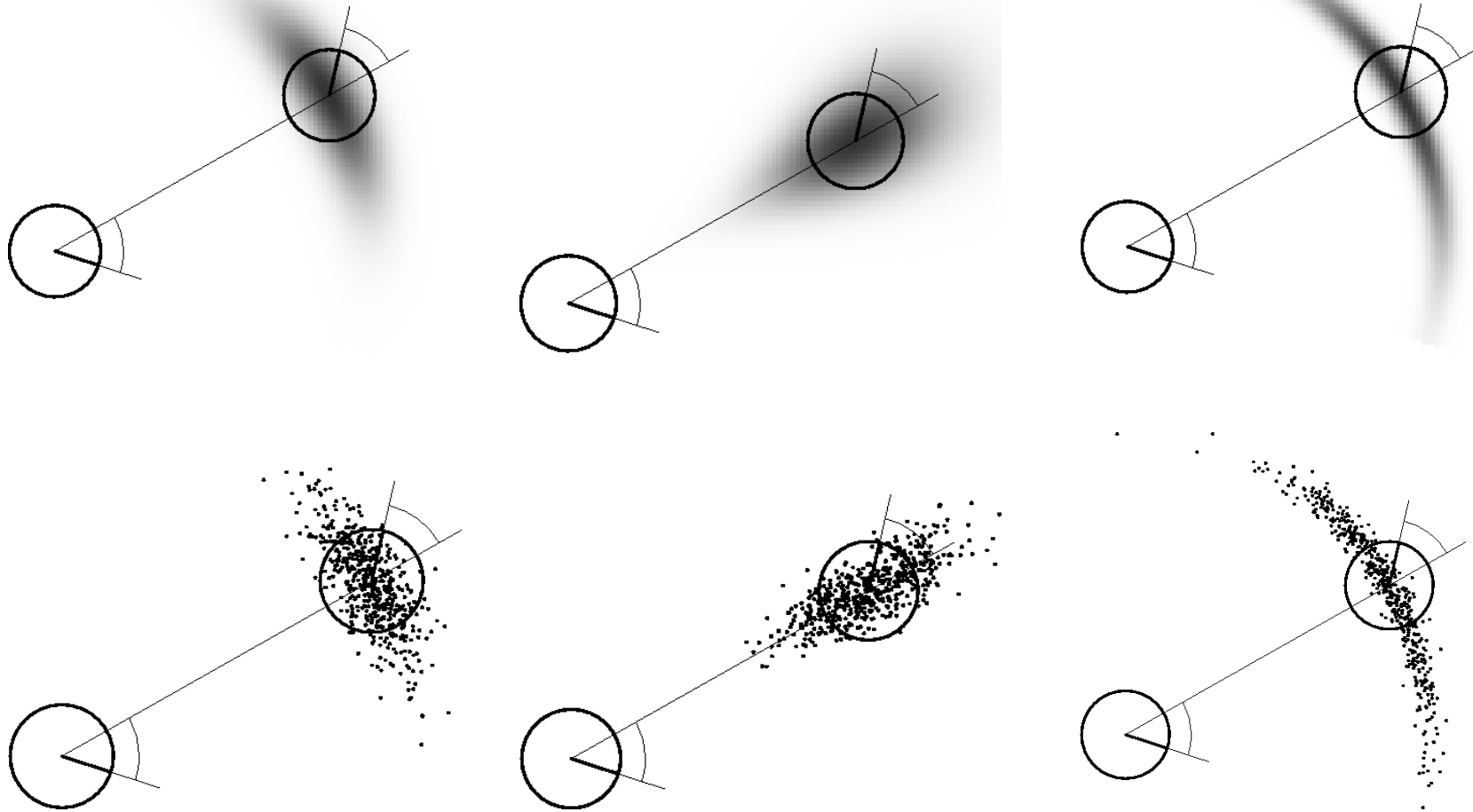
$$\overline{bel}(x_t) = \int p(x_t | u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}$$

- Intégration des observations ( $z$ )

$$bel(x_t) = \eta p(z_t | x_t) \overline{bel}(x_t)$$

*\*Le filtre de Kalman est un cas particulier quand la fonction de densité  $p(x)$  est une fonction gaussienne.*

# Modèles d'odométrie $p(x|u)$



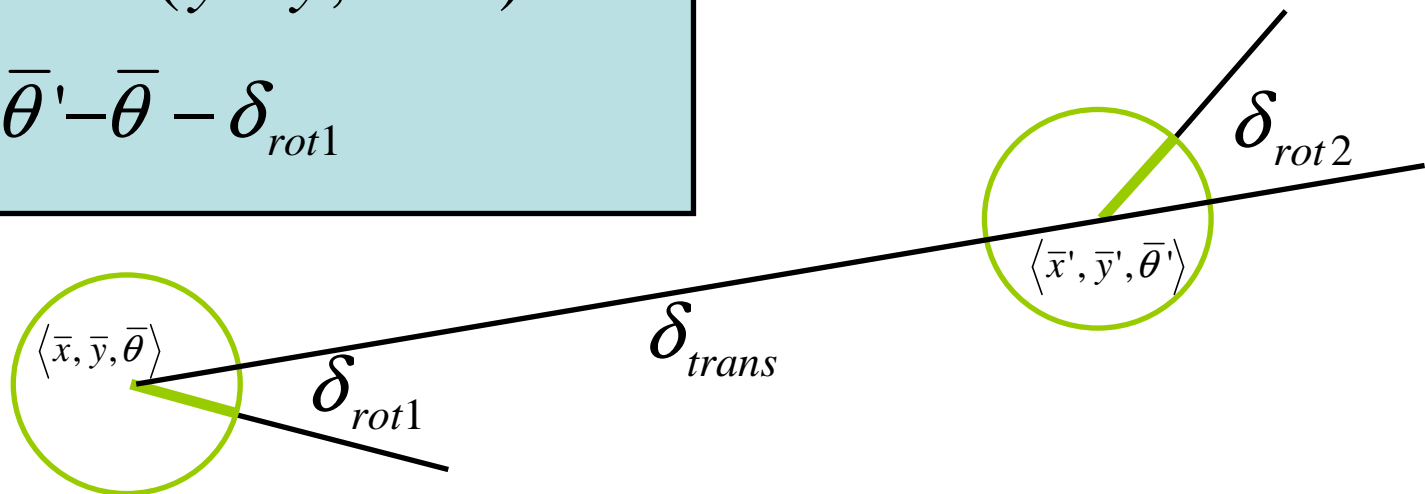
# Odometry Model

- Robot moves from  $\langle \bar{x}, \bar{y}, \bar{\theta} \rangle$  to  $\langle \bar{x}', \bar{y}', \bar{\theta}' \rangle$ .
- Odometry information  $u = \langle \delta_{rot1}, \delta_{rot2}, \delta_{trans} \rangle$ .

$$\delta_{trans} = \sqrt{(\bar{x}' - \bar{x})^2 + (\bar{y}' - \bar{y})^2}$$

$$\delta_{rot1} = \text{atan2}(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta}$$

$$\delta_{rot2} = \bar{\theta}' - \bar{\theta} - \delta_{rot1}$$



# Localisation locale

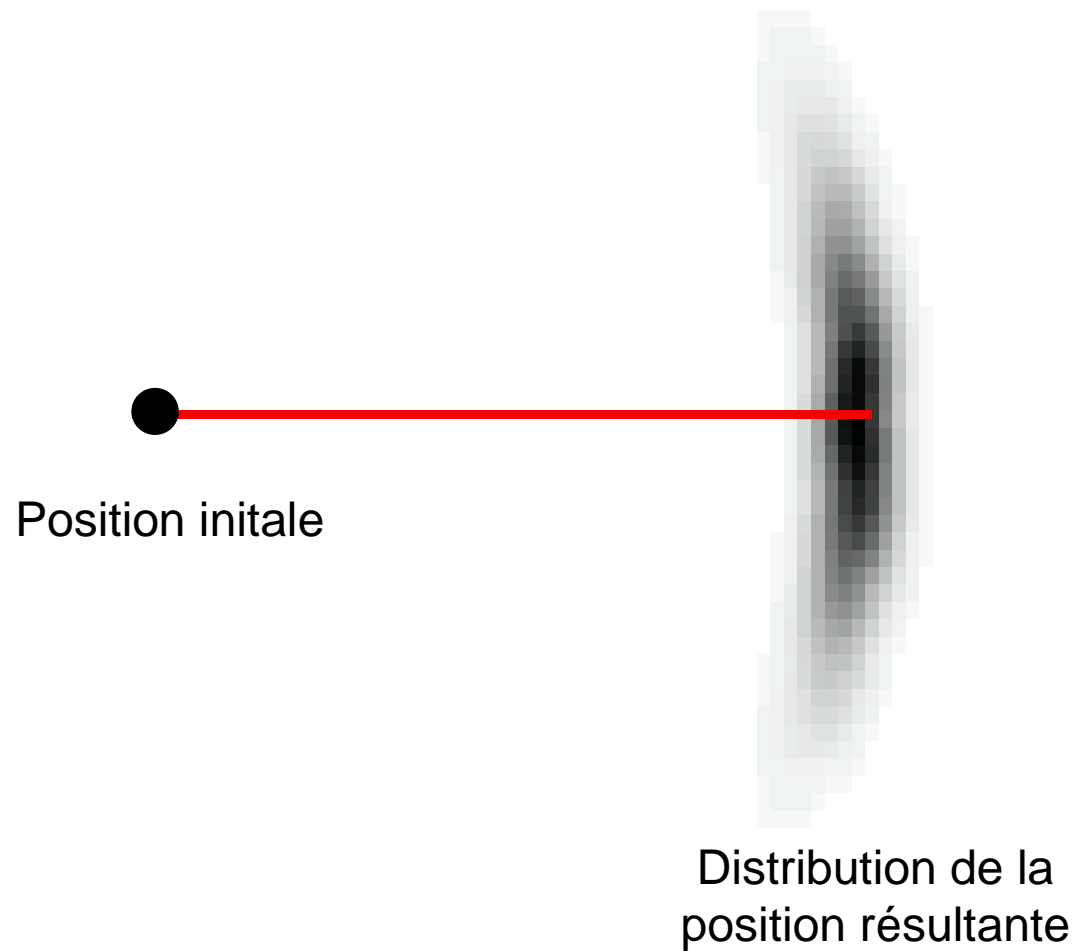
- La position initiale est connue, c'est-à-dire qu'on connaît la fonction de densité  $p(\mathbf{x}_t)$
- Estimations itératives
- Hypothèses (modèle probabiliste) :
  - Dynamique du robot (effet d'une action de déplacement  $a$ )  
$$p(\mathbf{x}_t | \mathbf{x}_{t-1}) = \mathbf{x}_{t-1} + \mathbf{a} + \mathbf{err}$$
  - Modèle d'observation  
$$p(\mathbf{x}_t | \mathbf{o}) = p(\mathbf{o} | \mathbf{x}_t) / p(\mathbf{x}_t) + \mathbf{err}$$
- Basée sur le filtre de Kalman (*Kalman filter*), un estimateur statistique optimal avec un modèle gaussien



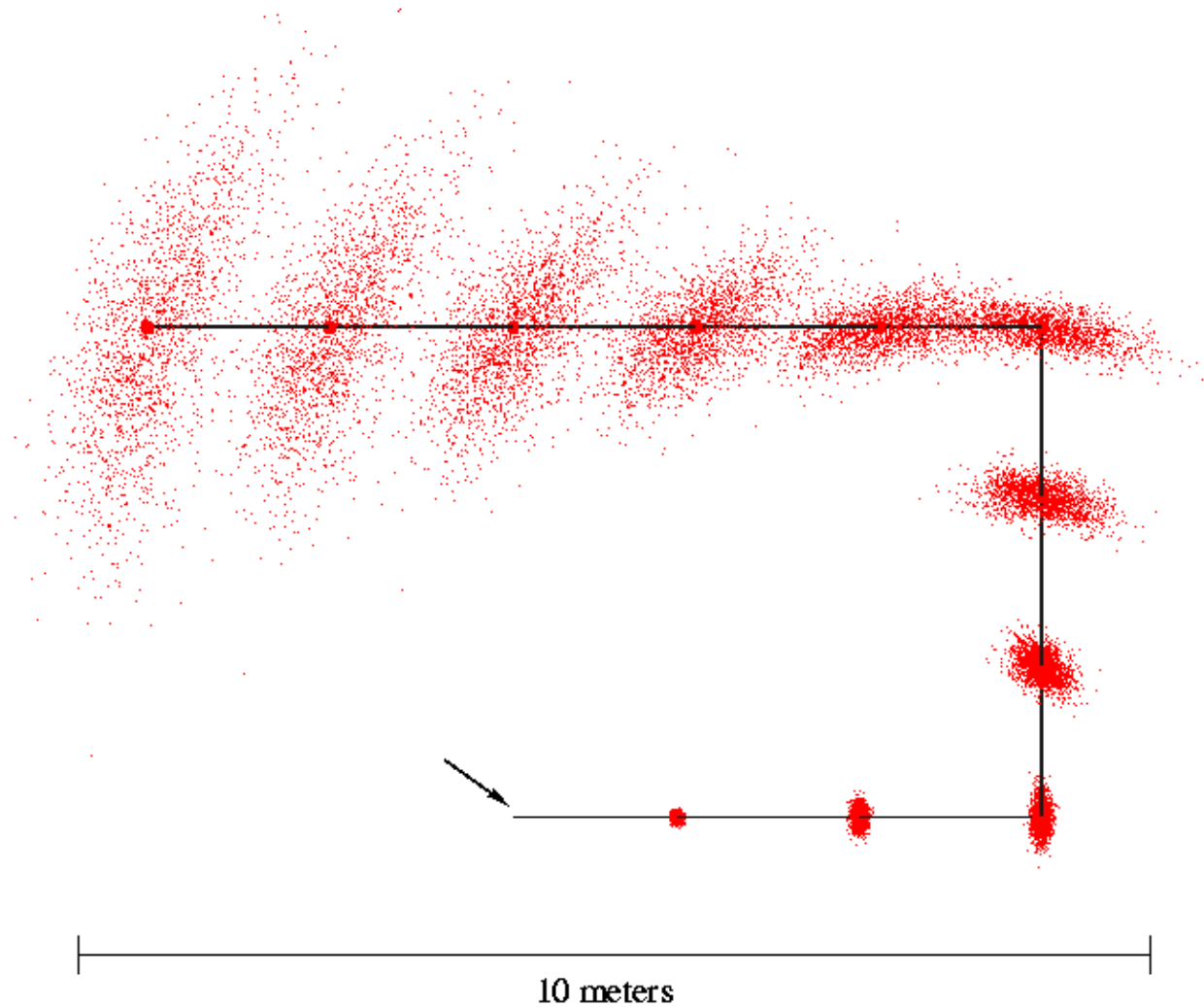
# Localisation locale

- Algorithme itératif
  - Intégrer **dynamique**: prédire la position sachant l'action de déplacement appliquée
  - Intégrer **observations**: corriger la position prédite sachant l'observation réalisée
- Cas continue : Kalman Filter ou Extended Kalman Filter
- Cas discret : Approximation avec des techniques de Monte Carlo (filtre à particules ou « Particles Filter »)

# Modèle de déplacement (continue)



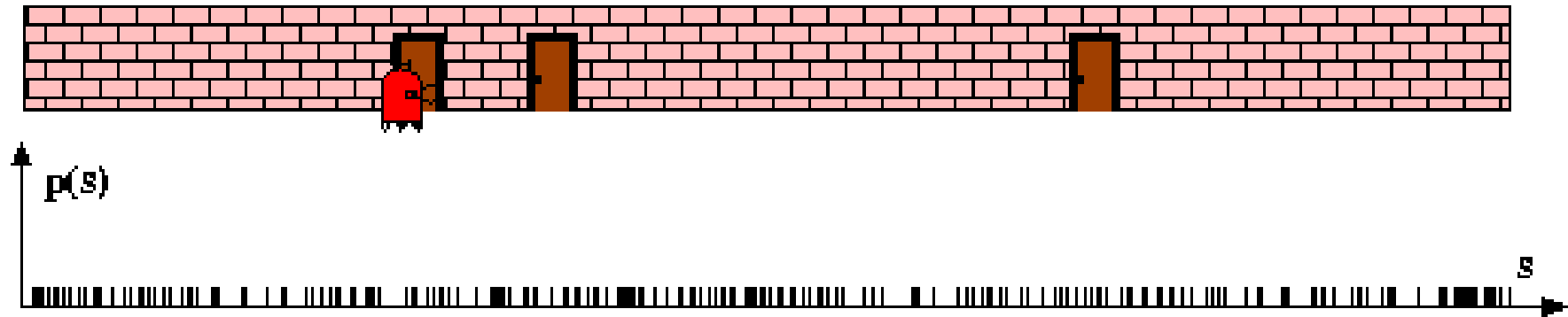
# Modèle de déplacement (discret)



# Localisation globale

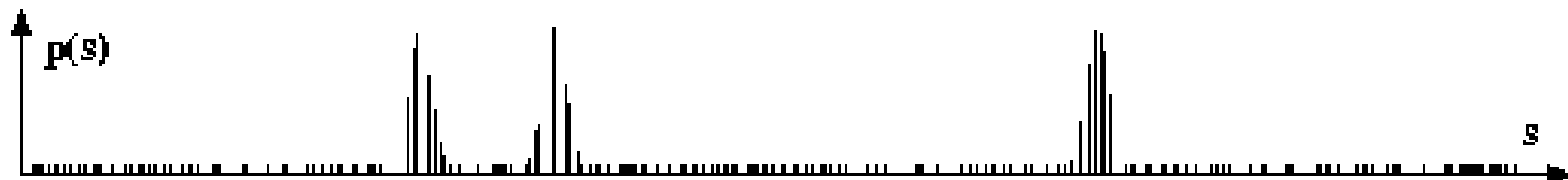
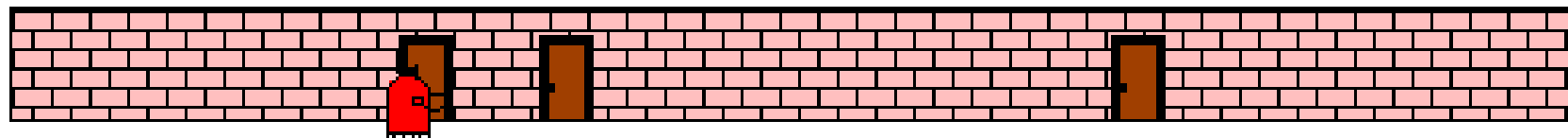
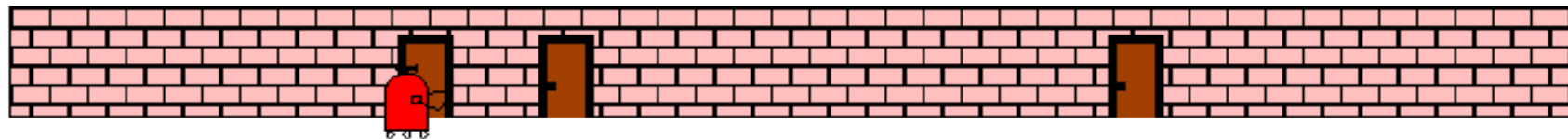
- La position initiale du robot est inconnue.
- Donc, la densité de  $p(x)$  est une fonction de densité uniforme.
- Dans le cas discret (techniques de Monte Carlo), on répartit uniformément les particules sur la carte.

# Particle Filters



# Sensor Information: Importance Sampling

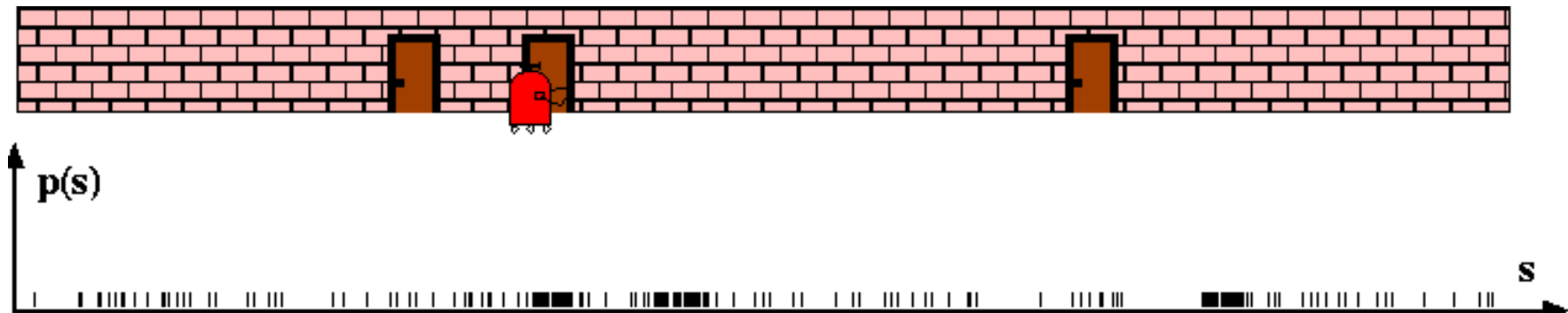
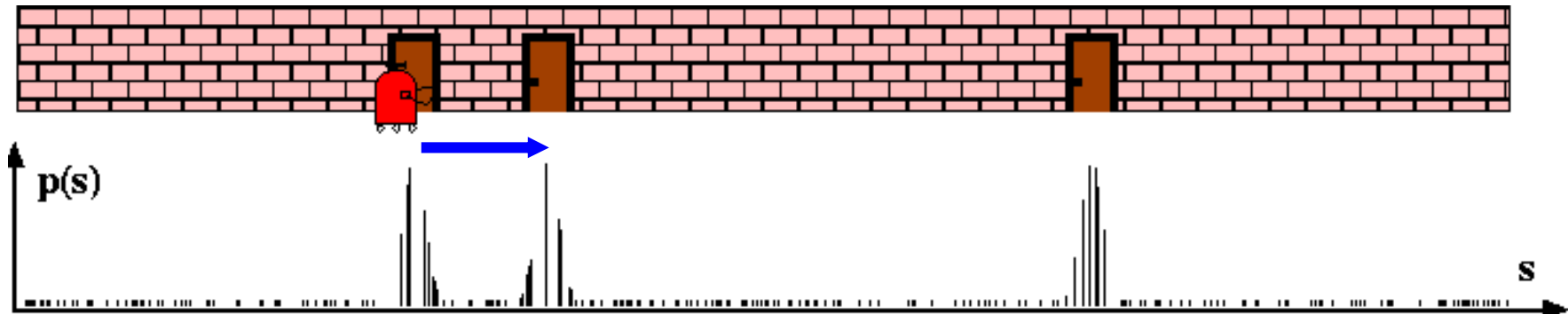
$$\begin{aligned} Bel(x) &\leftarrow \alpha p(z|x) Bel^-(x) \\ w &\leftarrow \frac{\alpha p(z|x) Bel^-(x)}{Bel^-(x)} = \alpha p(z|x) \end{aligned}$$



Source : figure tirée de <http://www.probabilistic-robotics.org/>

# Robot Motion

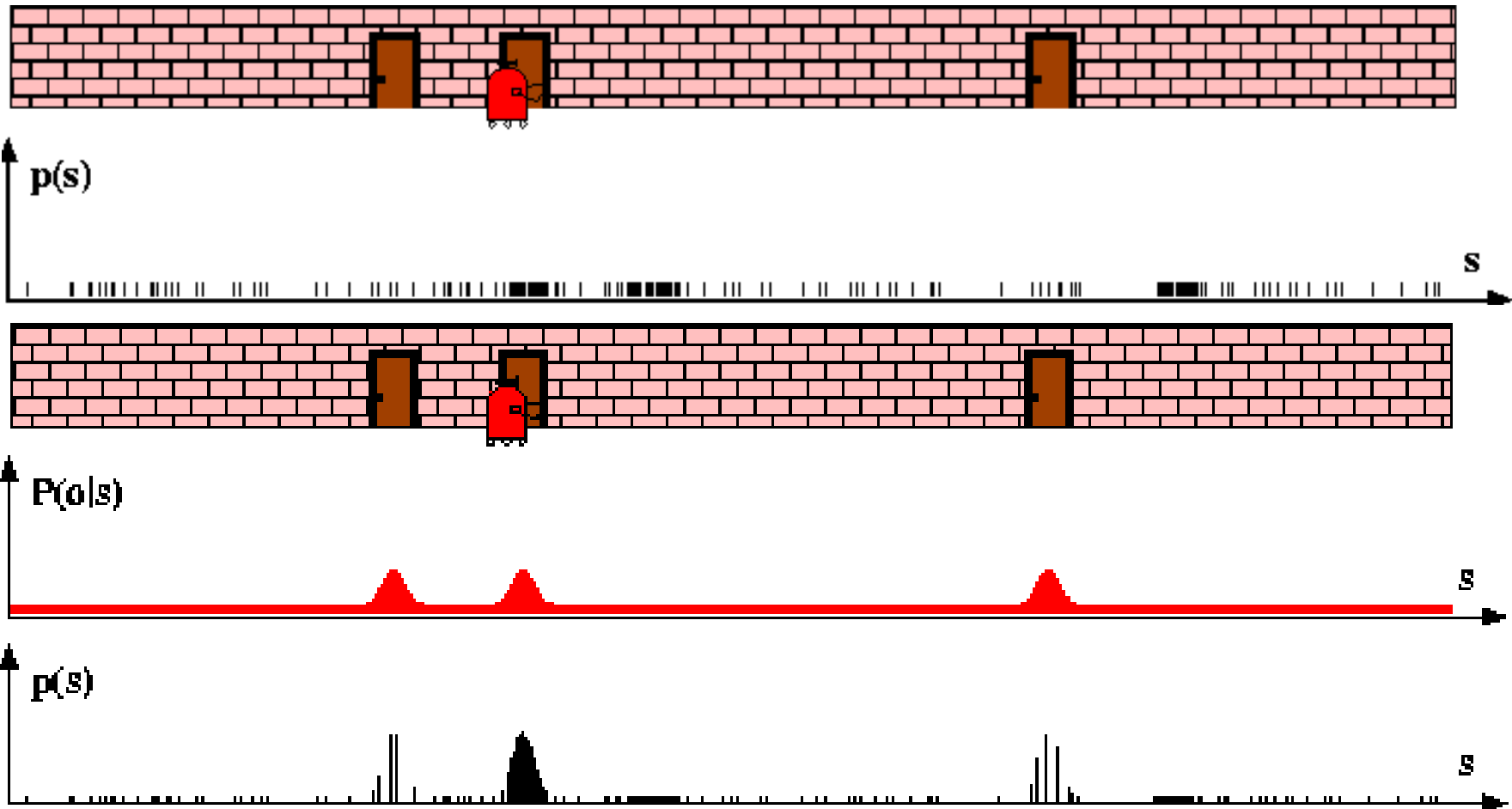
$$Bel^-(x) \leftarrow \int p(x|u, x') Bel(x') dx'$$





# Sensor Information: Importance Sampling

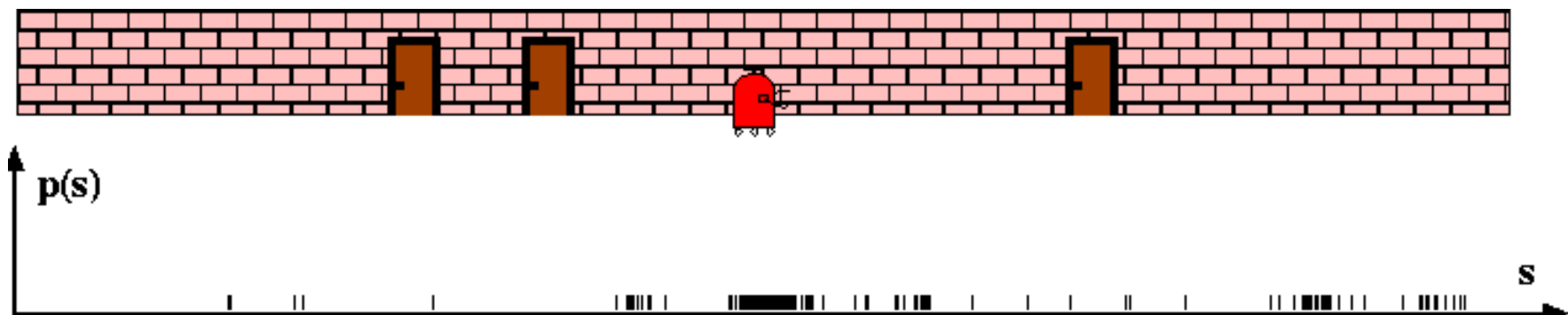
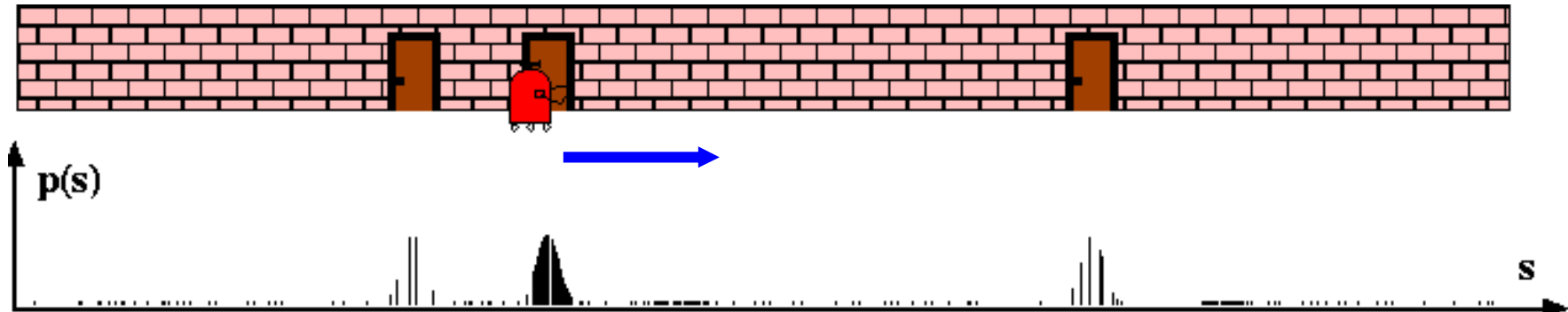
$$\begin{aligned} Bel(x) &\leftarrow \alpha p(z|x) Bel^-(x) \\ w &\leftarrow \frac{\alpha p(z|x) Bel^-(x)}{Bel^-(x)} = \alpha p(z|x) \end{aligned}$$



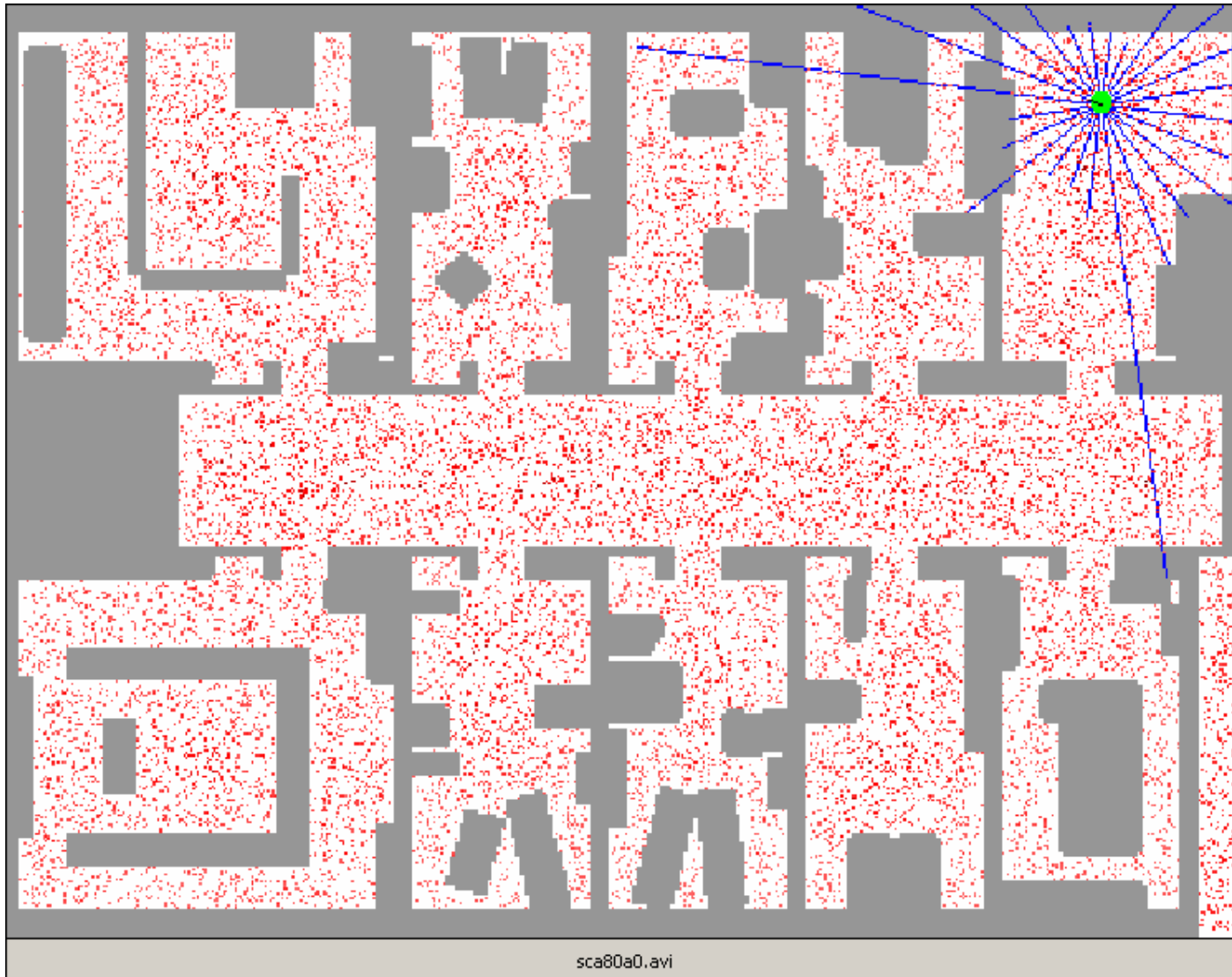
Source : figure tirée de <http://www.probabilistic-robotics.org/>

# Robot Motion

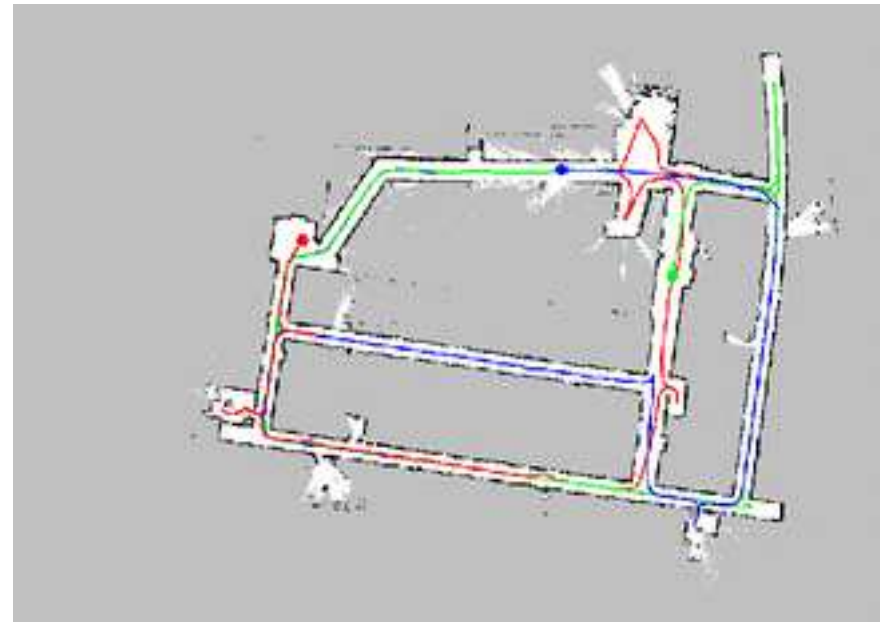
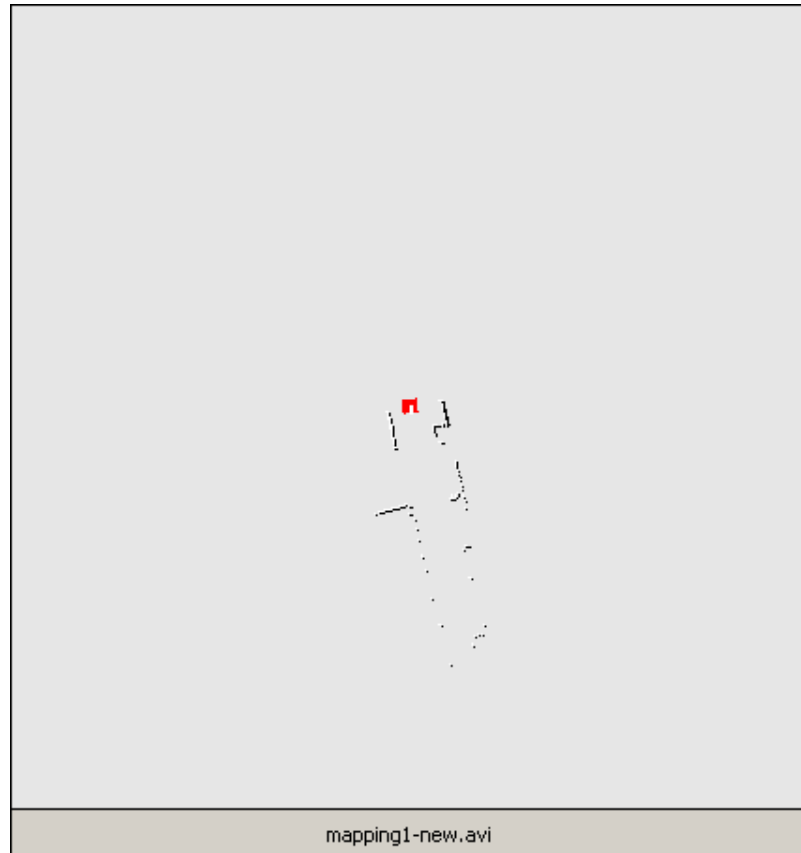
$$Bel^-(x) \leftarrow \int p(x|u, x') Bel(x') dx'$$



# Localisation Monte Carlo



# Construction d'une carte



# Actionneurs

**Capteurs**

**Extraction et  
interprétation**

**Modèle de  
l'environnement**

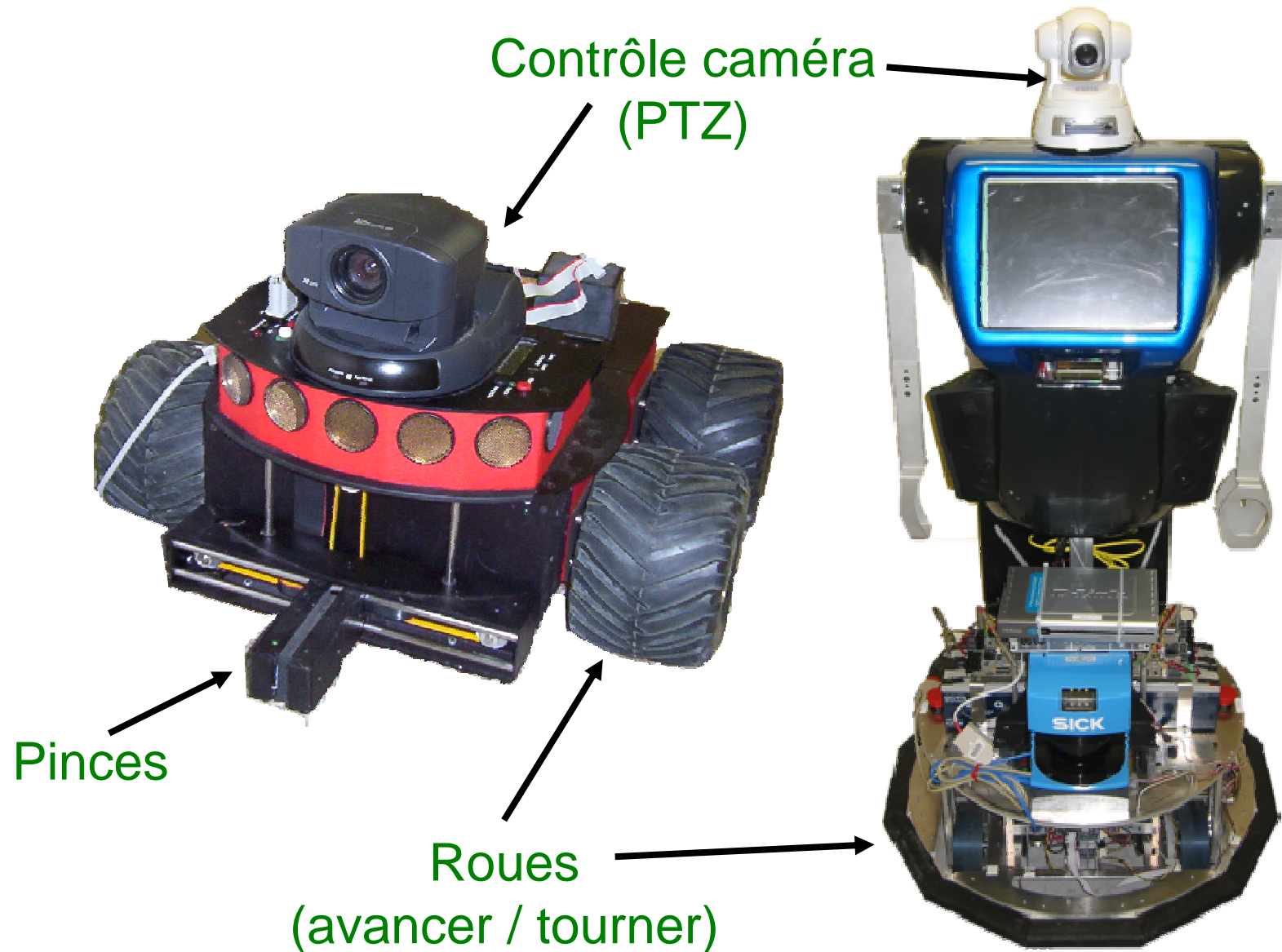
**Raisonnement  
Prise de décision**

**Actions,  
commandes  
motrices**

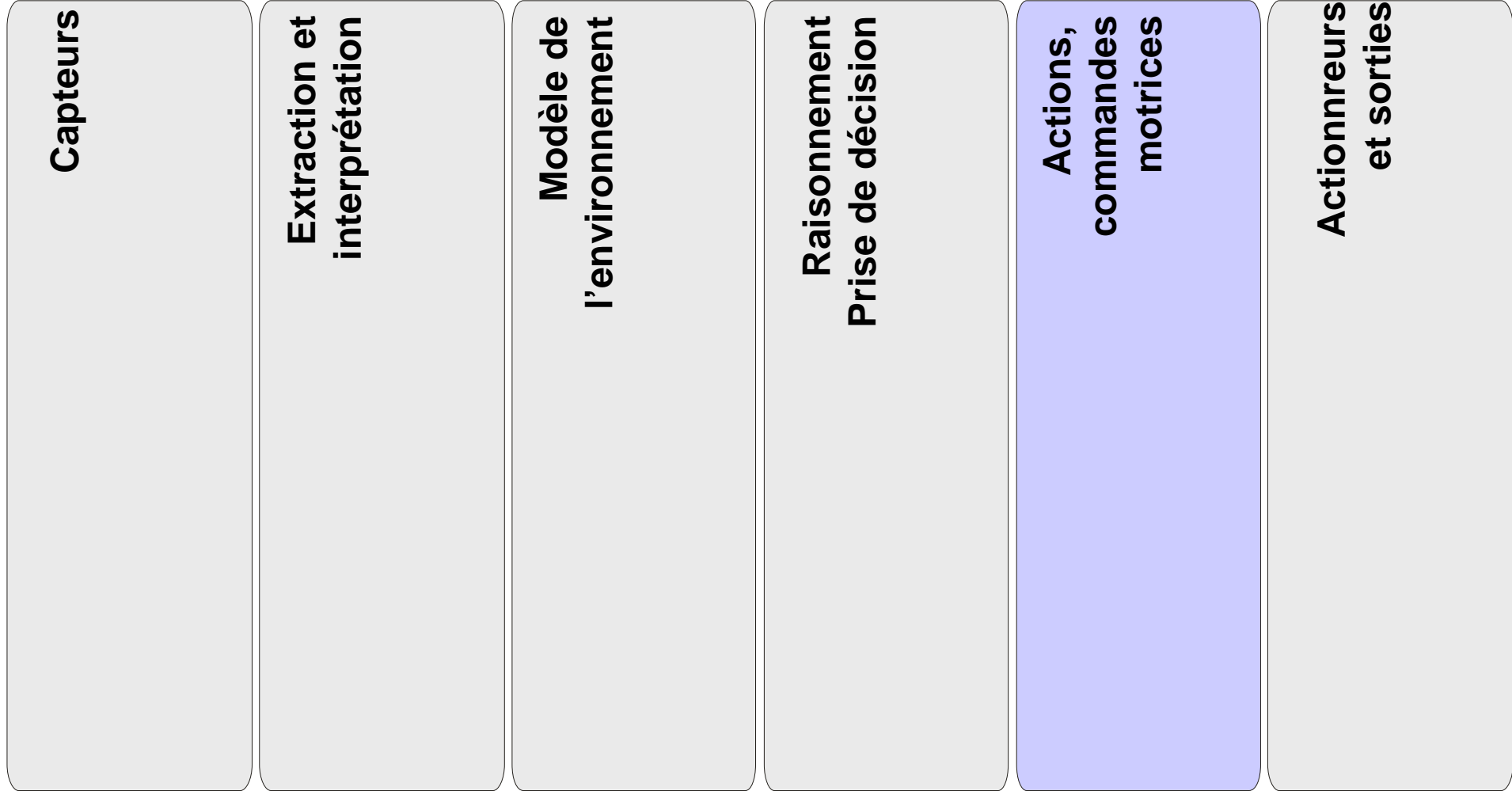
**Actionneurs  
et sorties**



# Actionneurs d'un robot



# Génération des actions





# Modules comportementaux

(« *Behaviors producing modules* »)

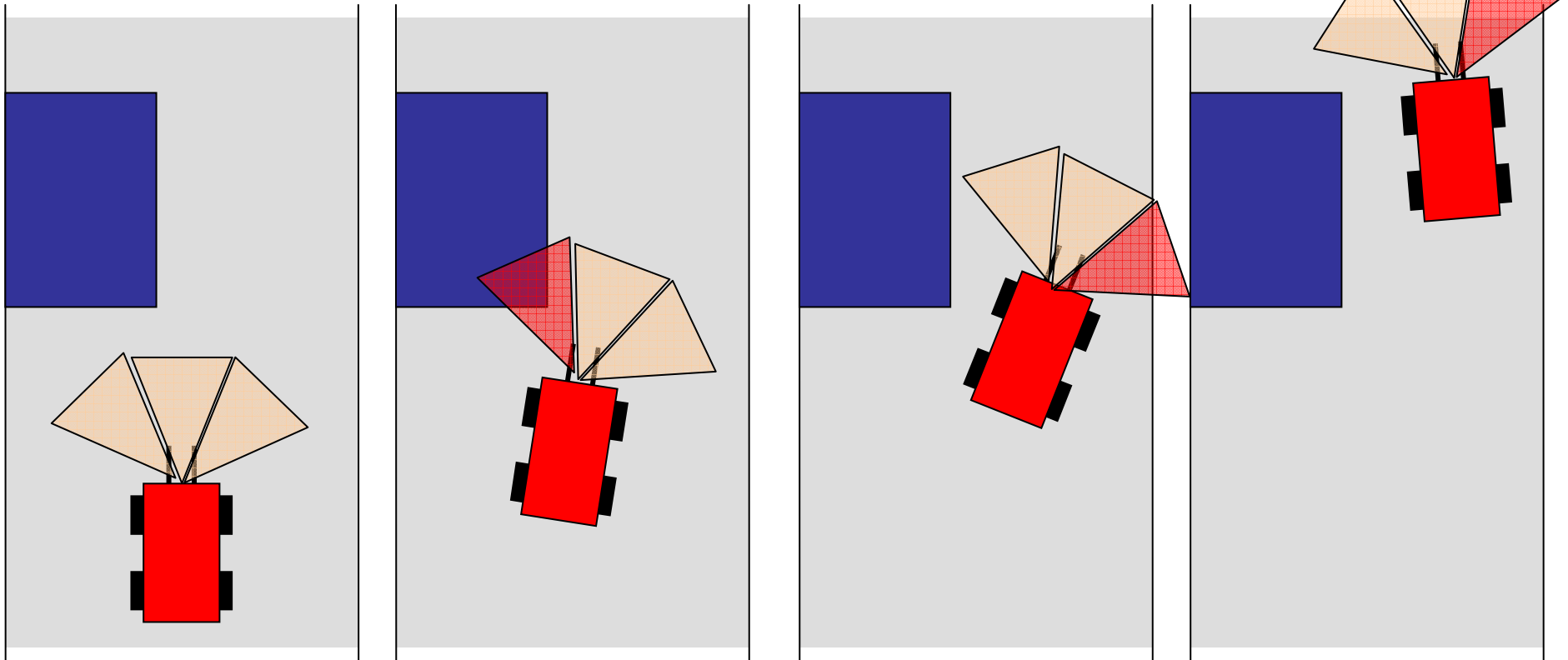
- Un **module comportemental** est une boucle de rétroaction générant des commandes pour 1+ actionneurs à partir des données de 0 ou plusieurs capteurs
- Aussi appelé **comportement**
- Responsable d'accomplir une seule fonction
  - Évitement d'obstacles
  - Suivit de trajectoires
  - Suivit de couleurs, objets, ...
  - Longer des murs
  - ...

# Exemples de comportement :

## évitement d'obstacles (1)

<b>Obstacle</b>	<b>Action</b>
(aucun)	(aucune)
à gauche	tourner à droite
à droite	tourner à gauche
devant	tourner à gauche ou à droite
devant + gauche + droite	faire demi-tour

# Exemples de comportement : évitement d'obstacles (2)



*aucune action*

*tourner à droite*

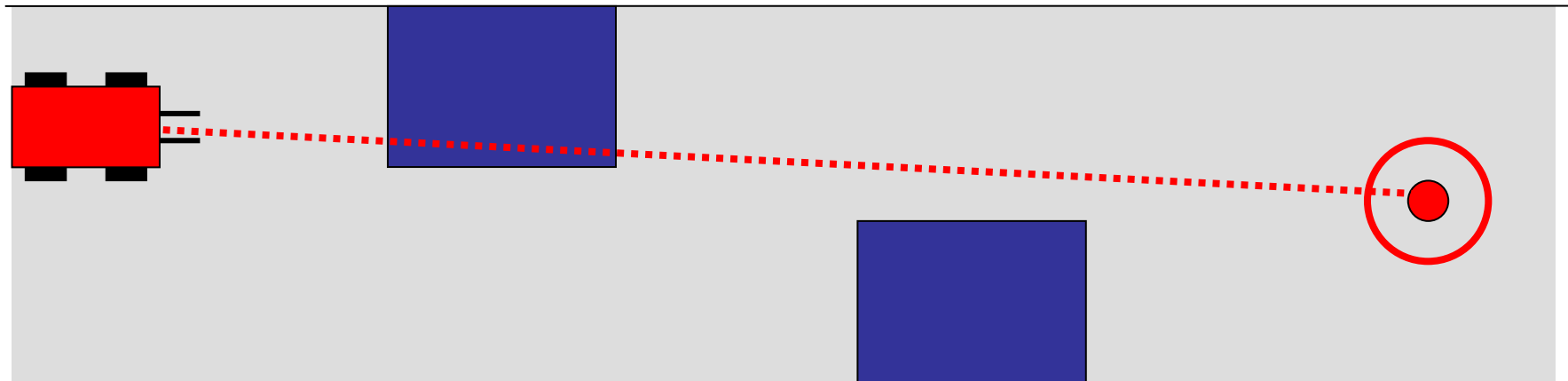
*tourner à gauche*

*légèrement à  
gauche*

# Exemples de comportement :

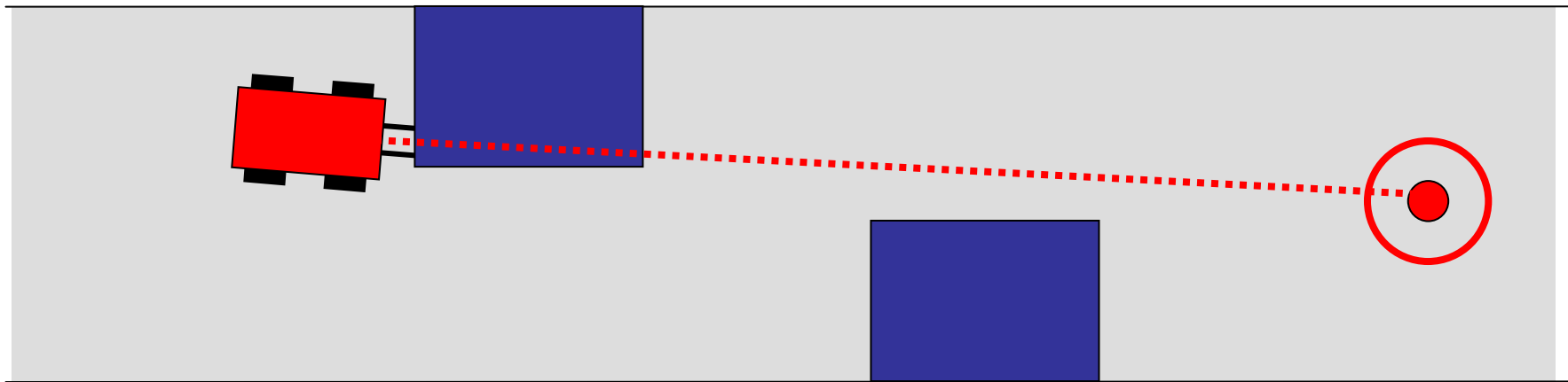
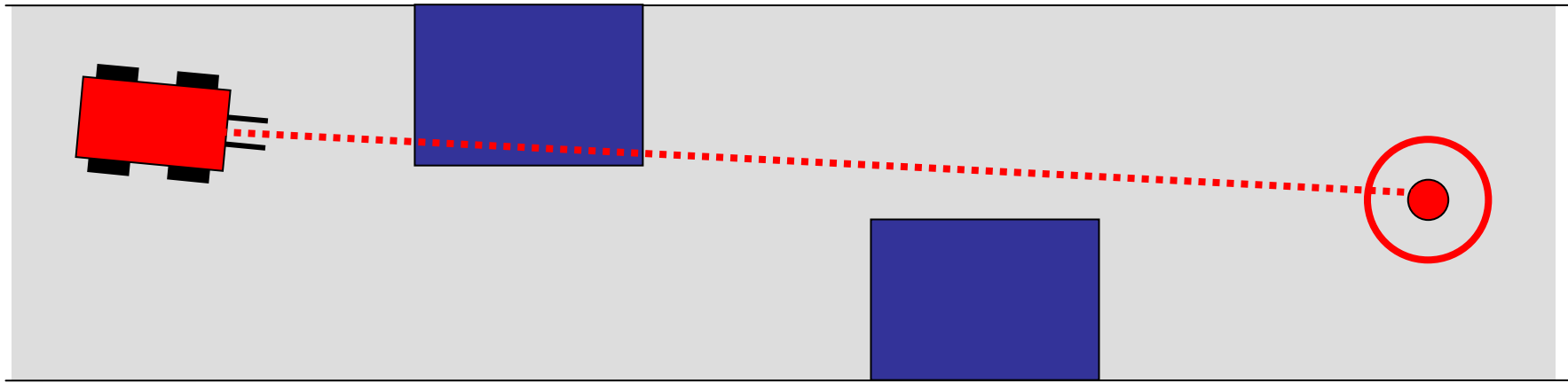
## « GOTO » (1)

- Diriger le robot en ligne droite sur une cible



# Exemples de comportement :

## « GOTO » (2)

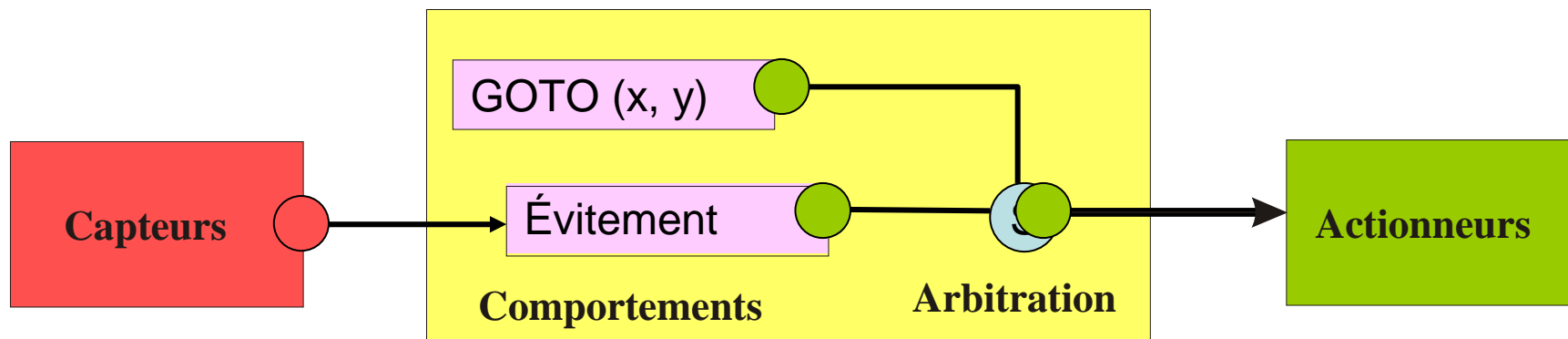


# Architecture comportementale

- Combinaison des comportements
- Arbitration de comportements
  - Priorité
  - Somme pondérée ( *motor schema* )
  - Logique floue
  - Etc.

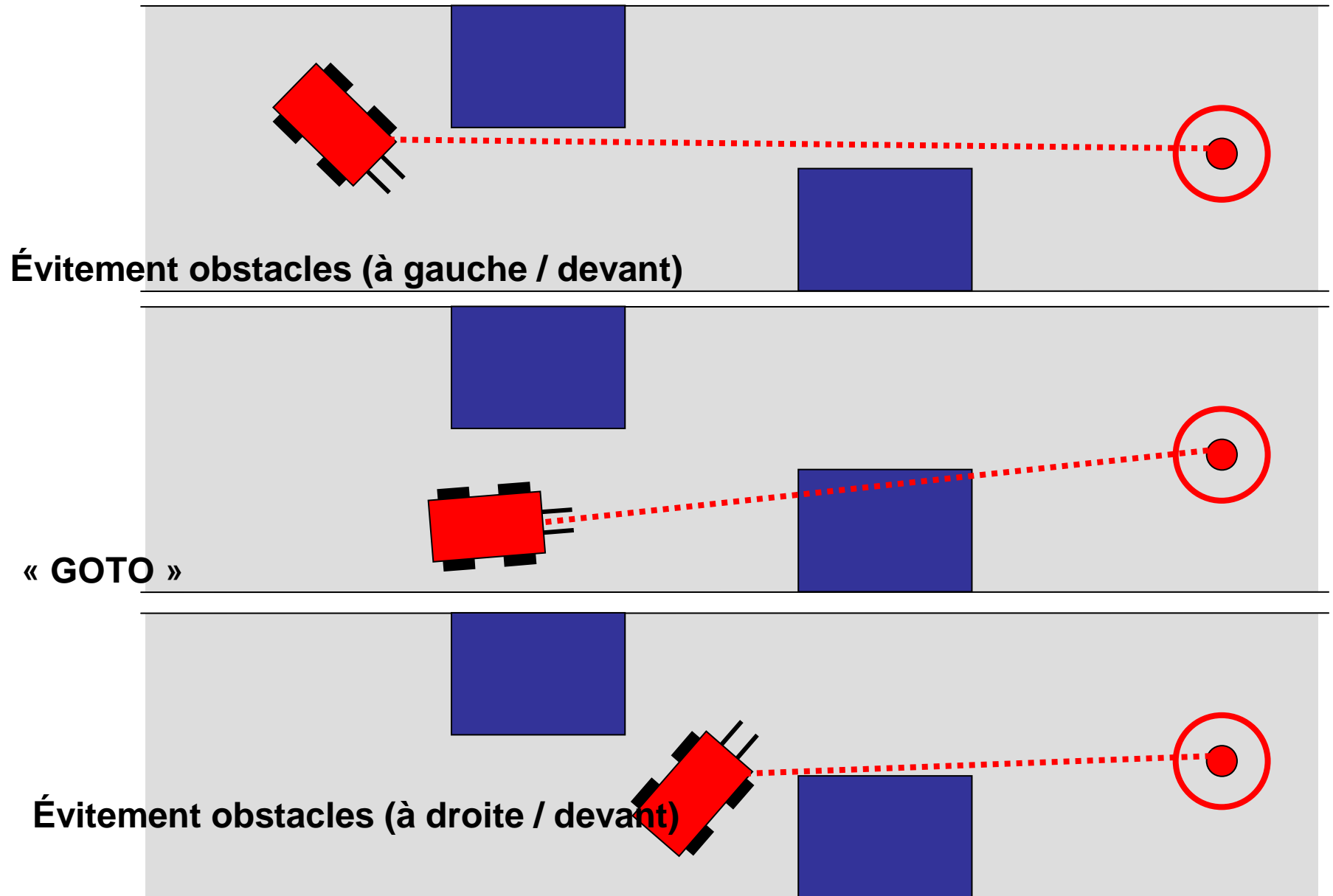
# Architecture comportementale

- L'évitement d'obstacles est prioritaire
- Quand l'évitement génère une commande, il « écrase » celle du suivi de trajectoire





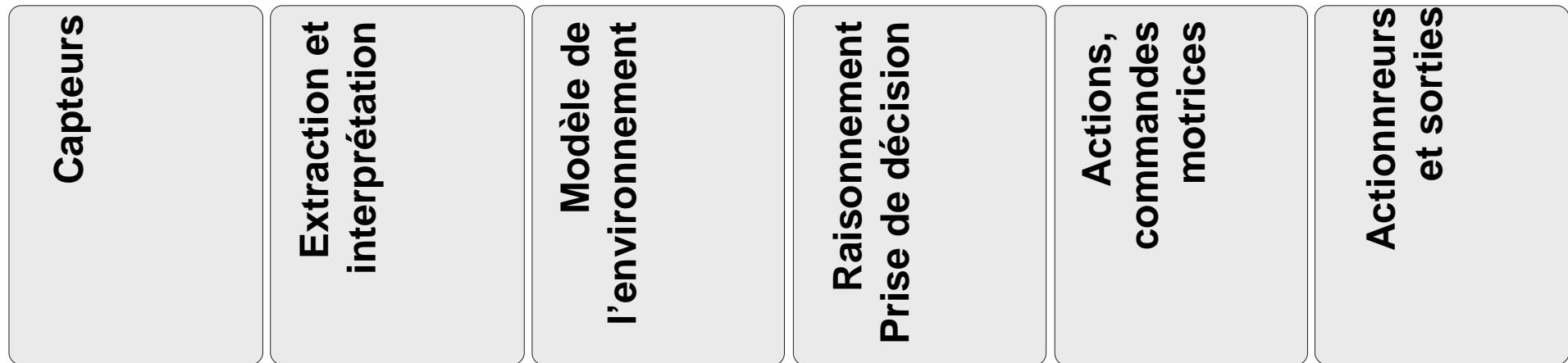
# Arbitration de comportements



# Architecture comportementale

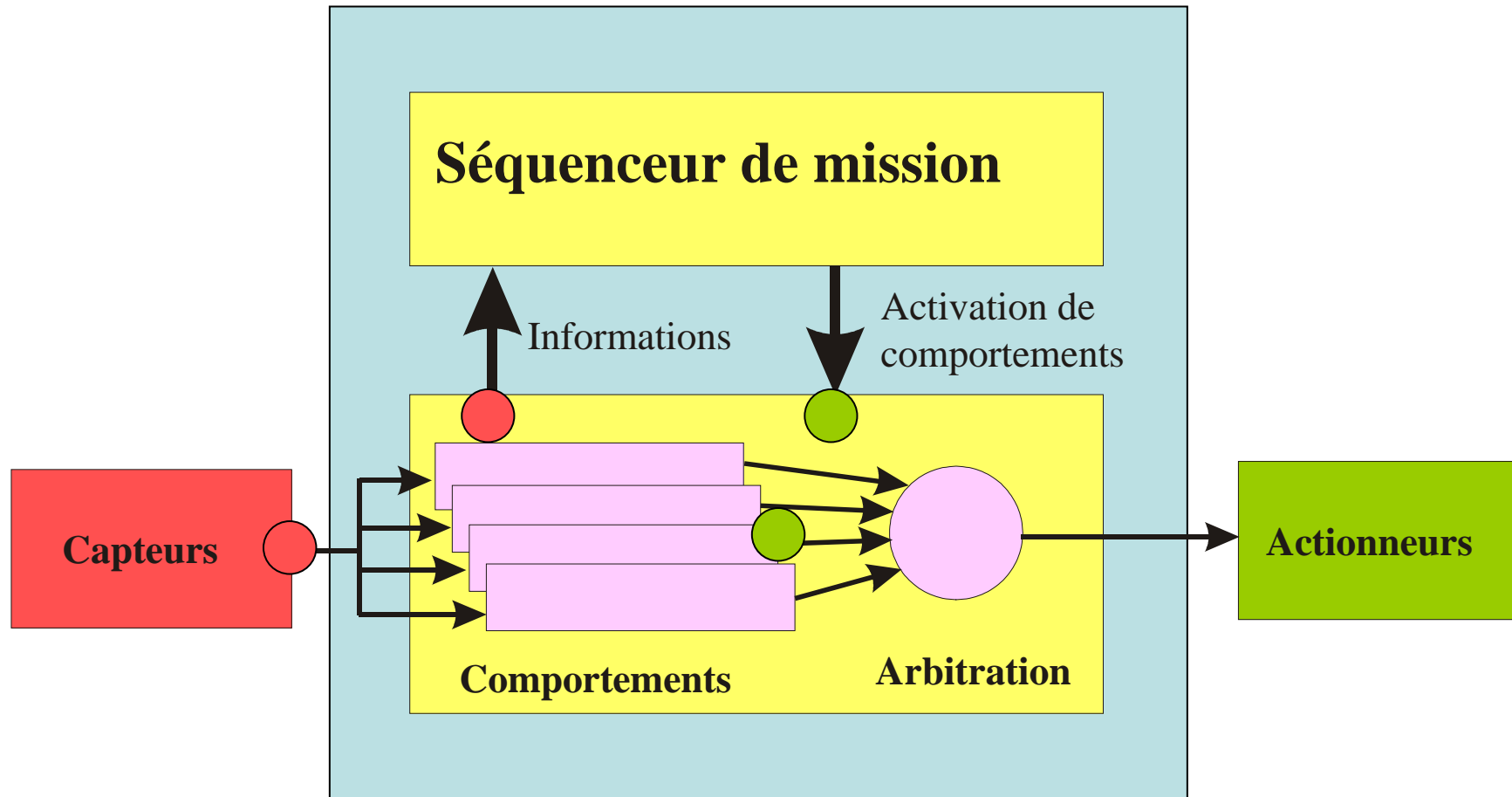
- Problème :
  - Manque de flexibilité pour coordonner l'exécution de tâches complexes

# Intégration

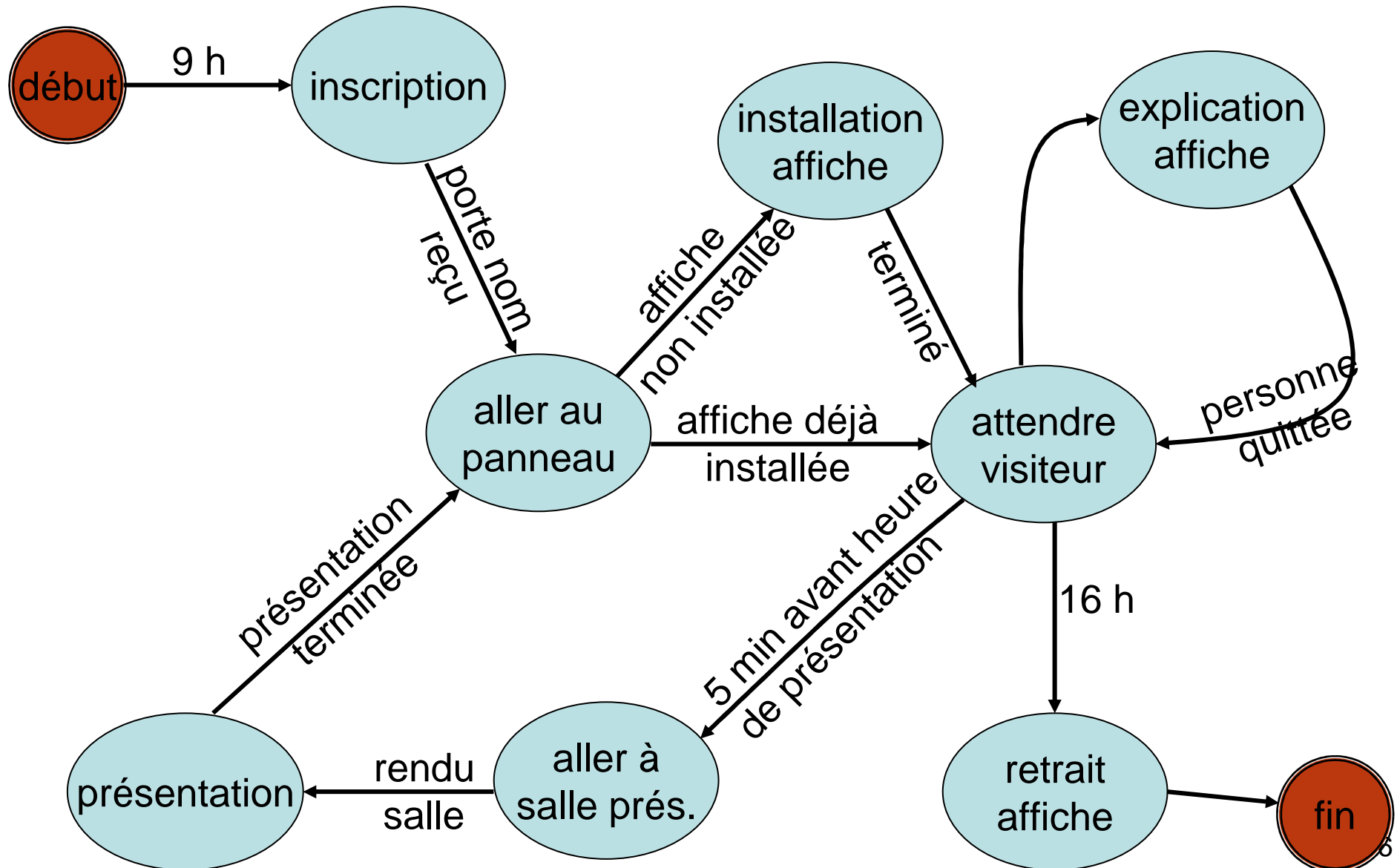


- **Intégration des fonctions : architectures**
- Intégration logicielle
  - Assemblage de modules
  - Programmation distribuée

# Architecture robotique de type hybride



# Exemple de schéma de mission à l'aide d'une « machine à états finis »

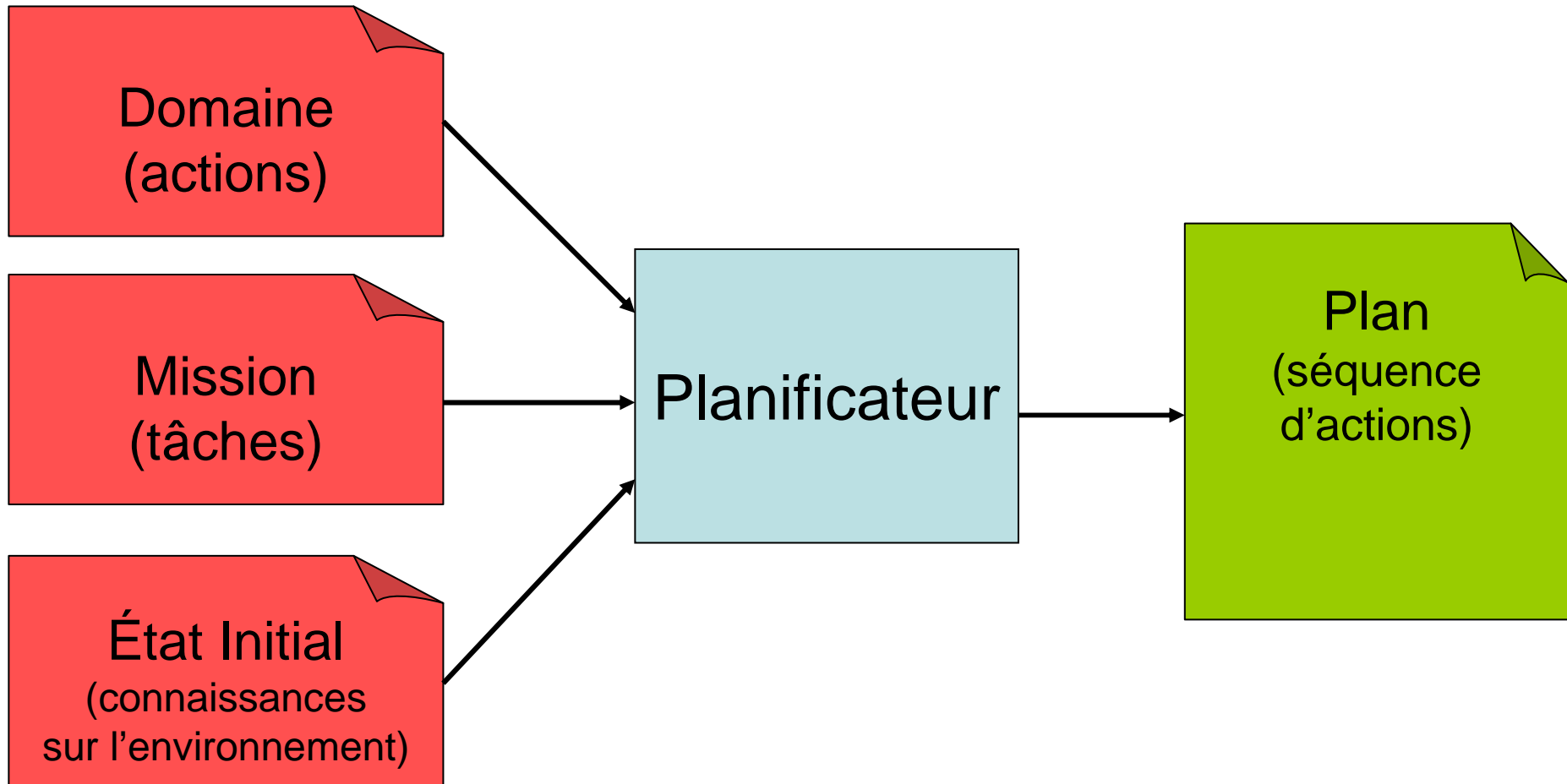


# Limitations

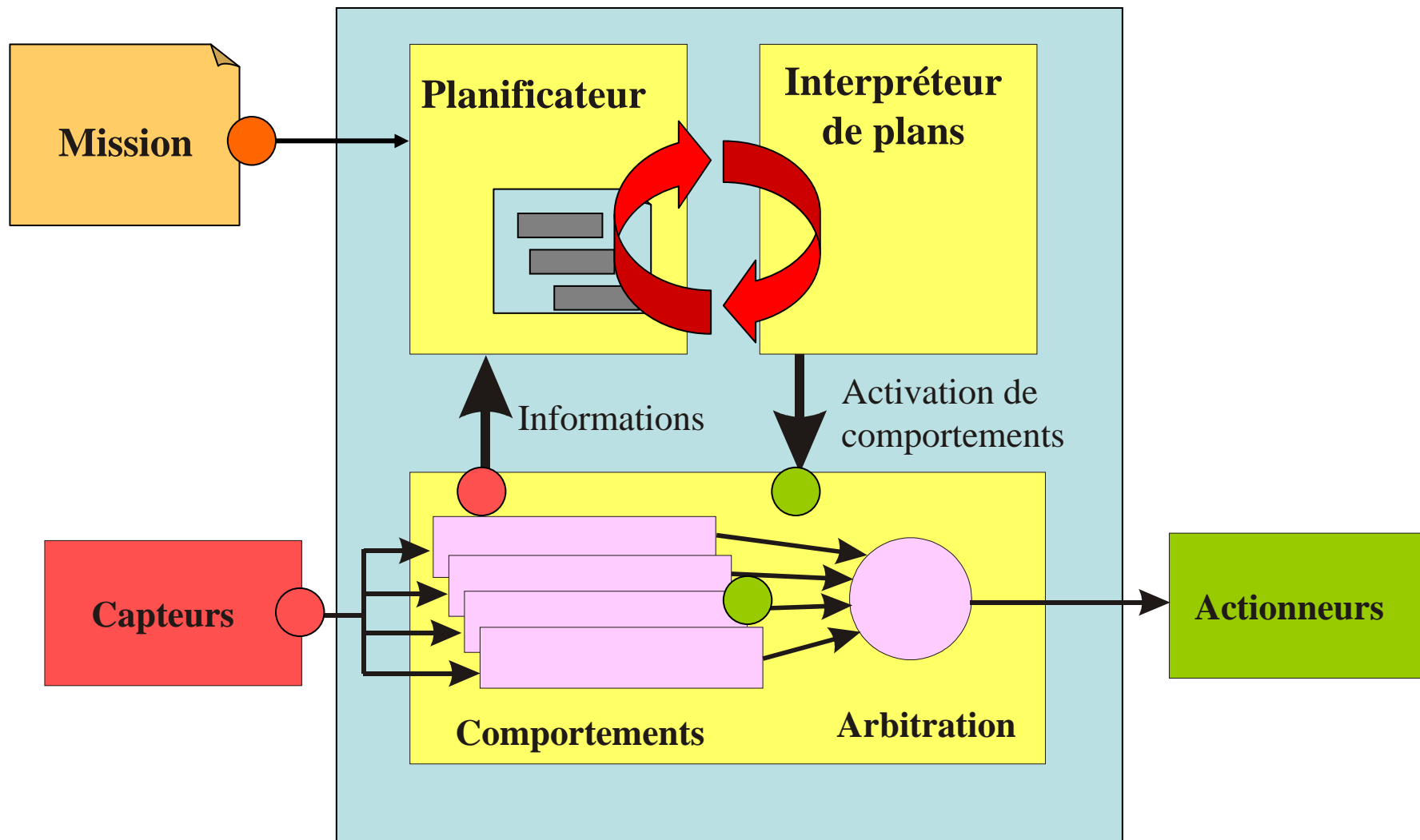
- Demande beaucoup de temps pour programmer les machines à états finis
- Impossible de prévoir tous les cas possibles (nous n'avons pas de boule de cristal)
- Comportement non optimal
- Manque de flexibilité
- Risques d'erreurs



# Modèle de planification



# Boucle réactive

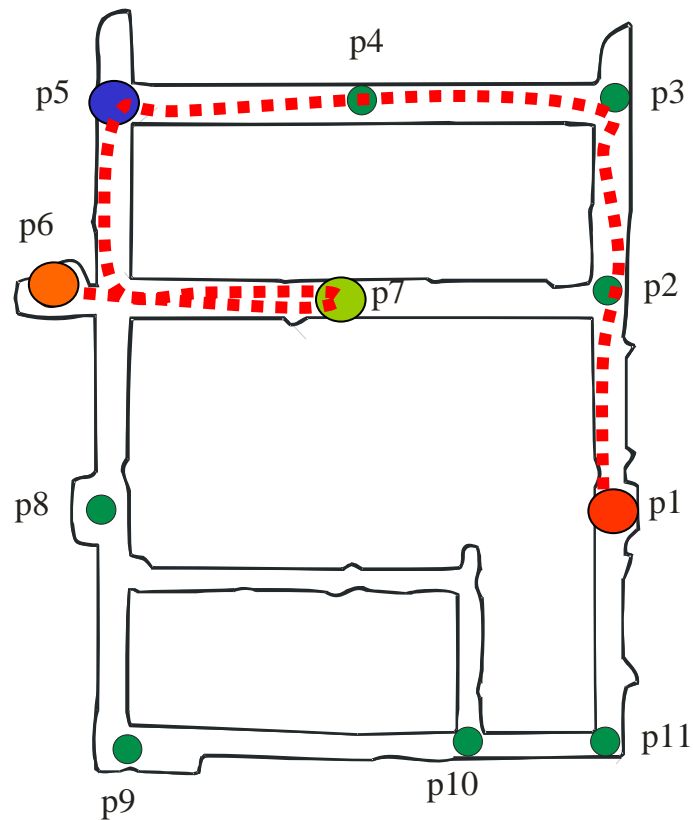




# Exemple

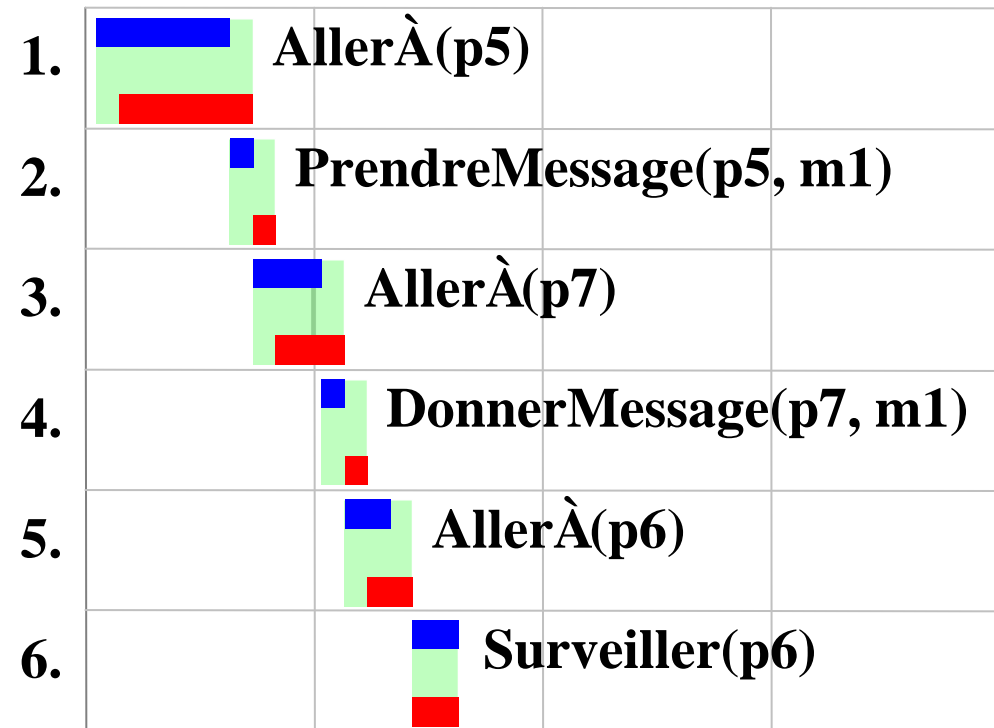
**Mission :**

- Livrer message p5 ● -> p7 ●
- Surveiller p6 à t=0h22m ●

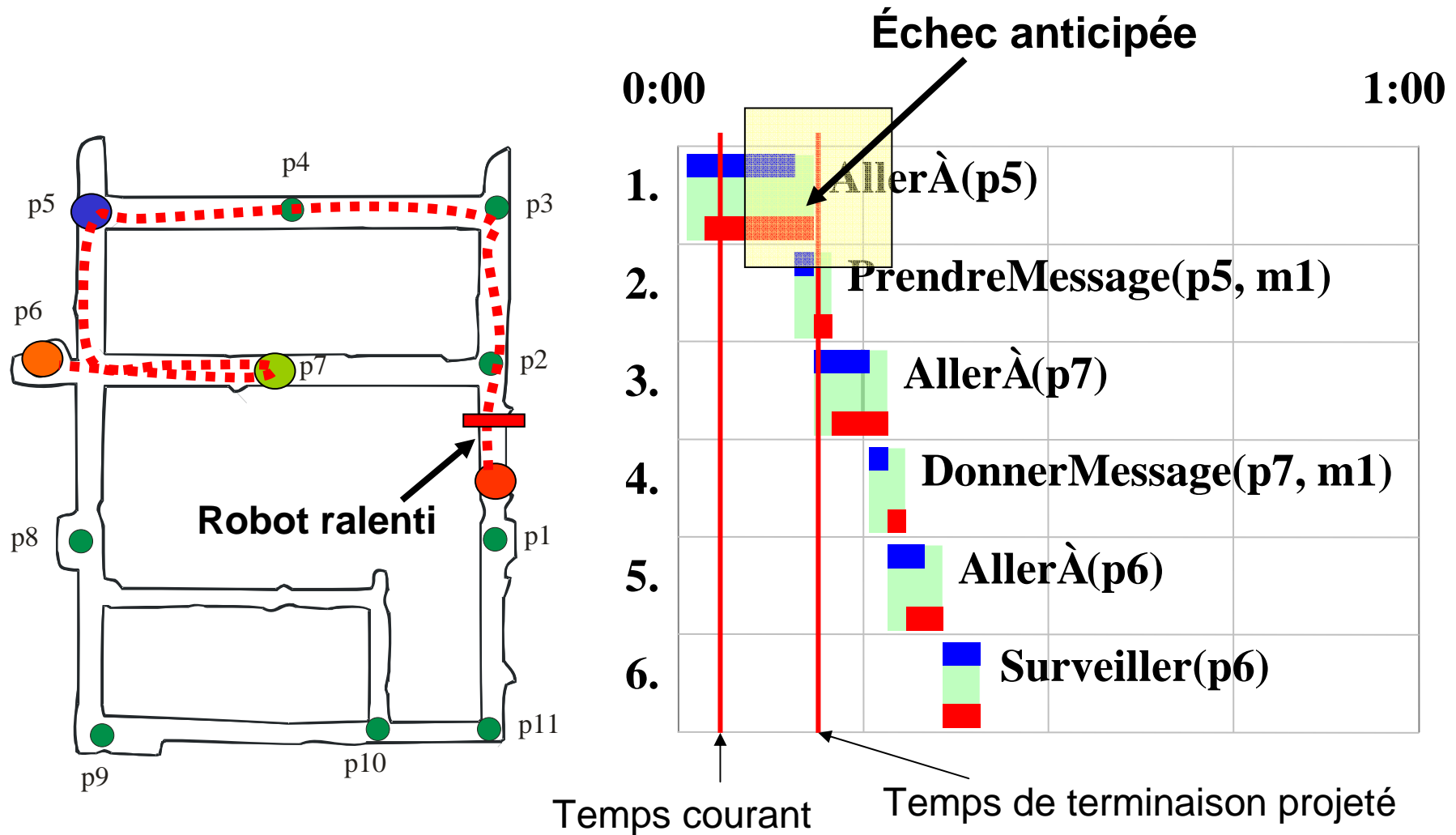


0:00

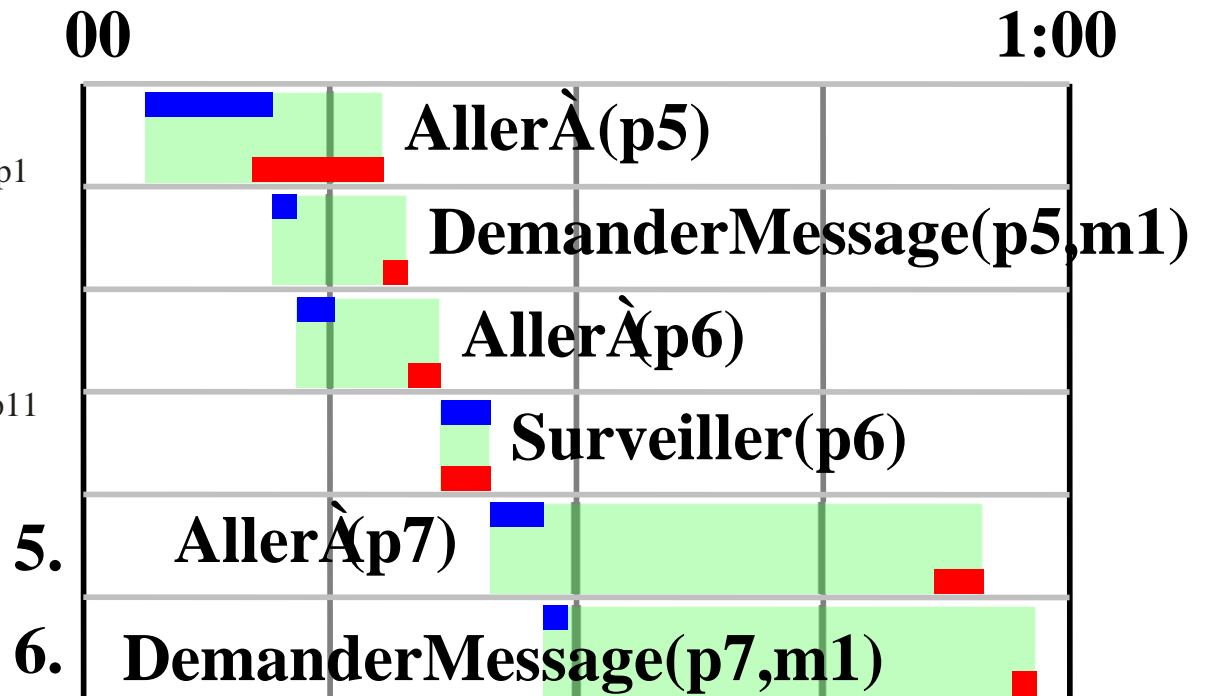
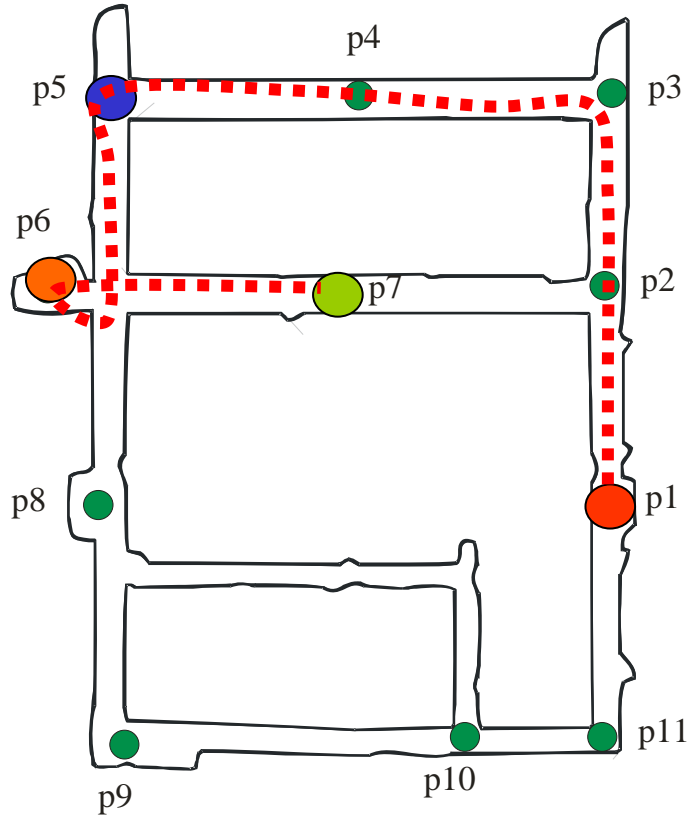
1:00



# Exemple (suite)



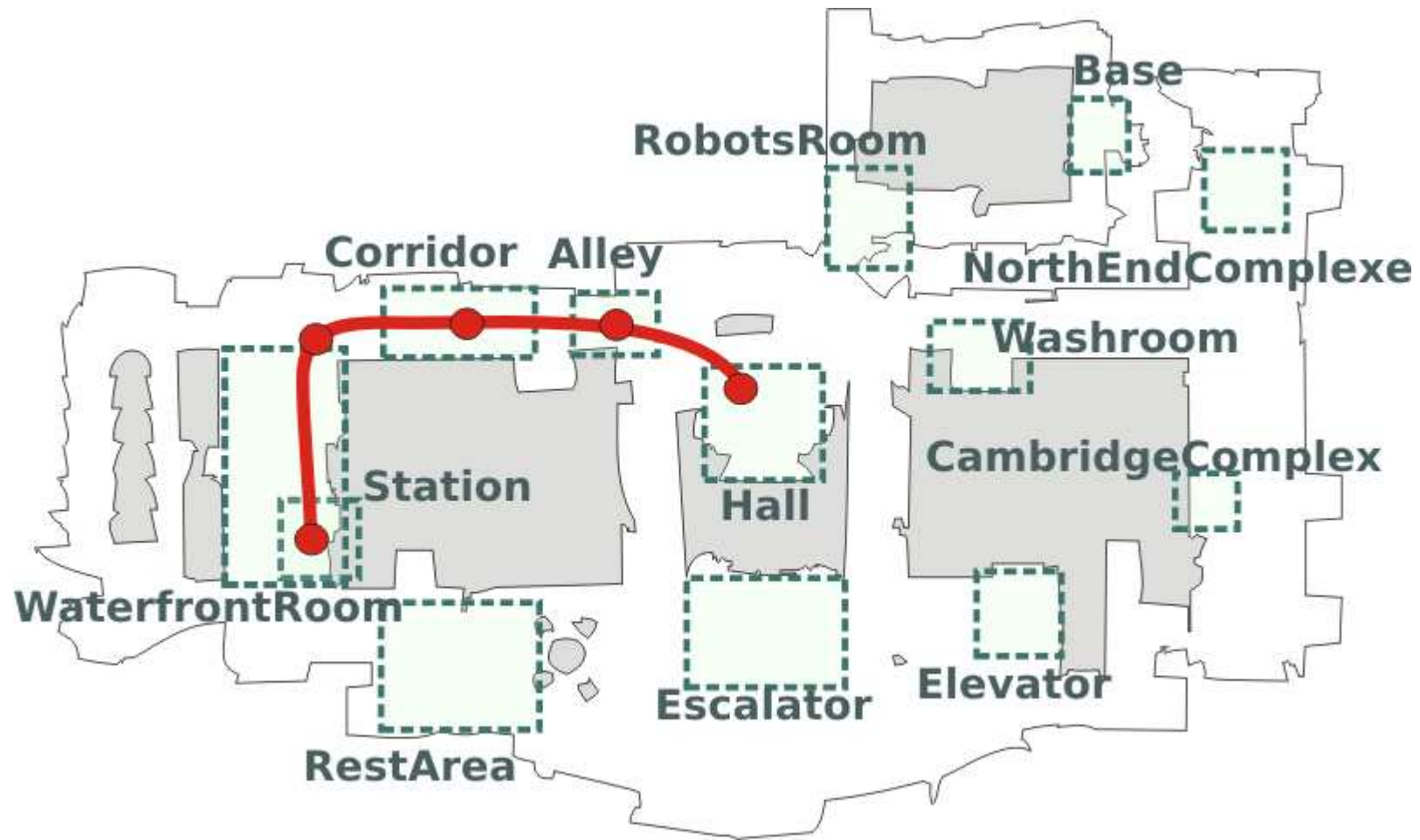
# Exemple (suite)



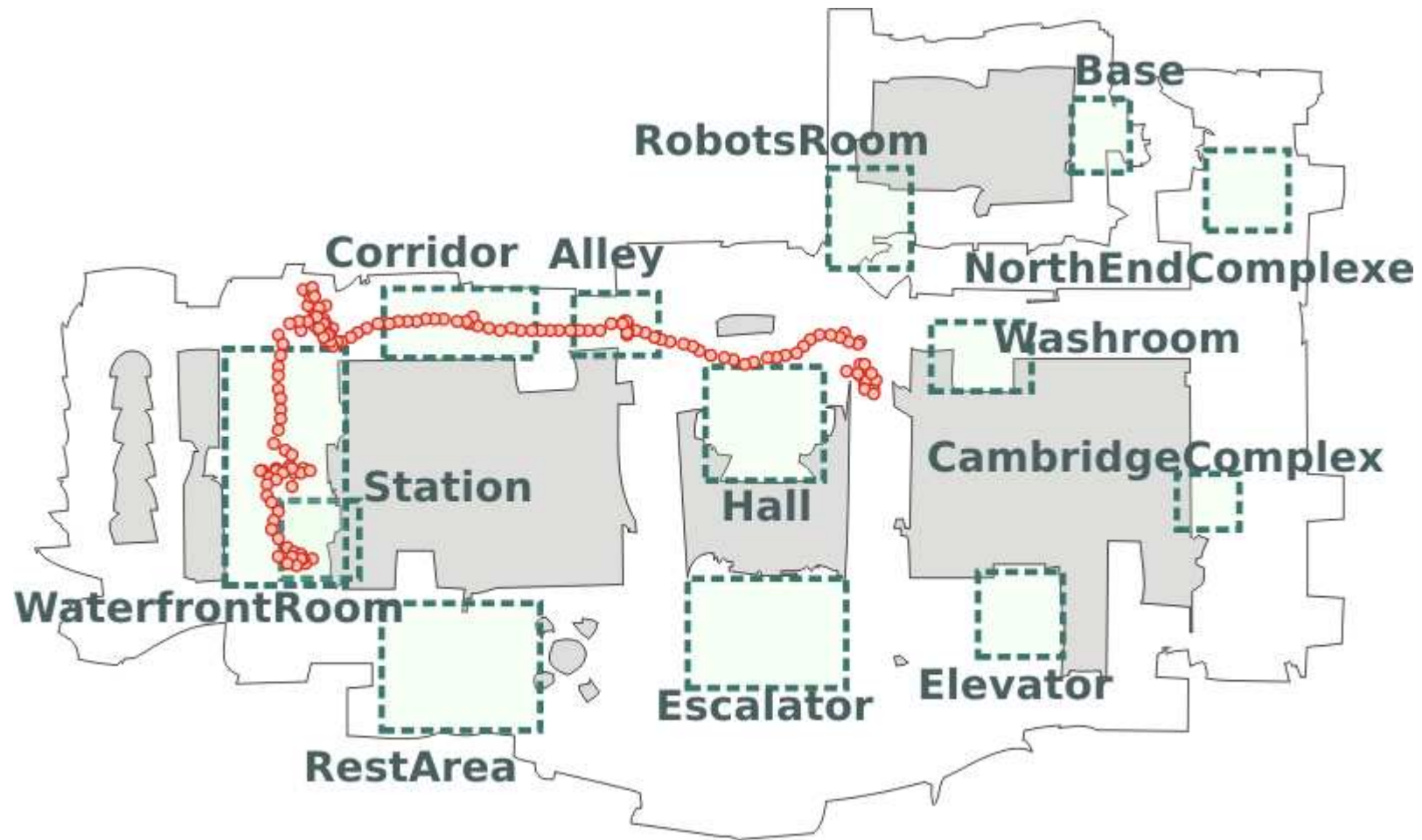
# Notre expérience à AAAI 2006 (Boston)



# Plan de démonstration aux juges

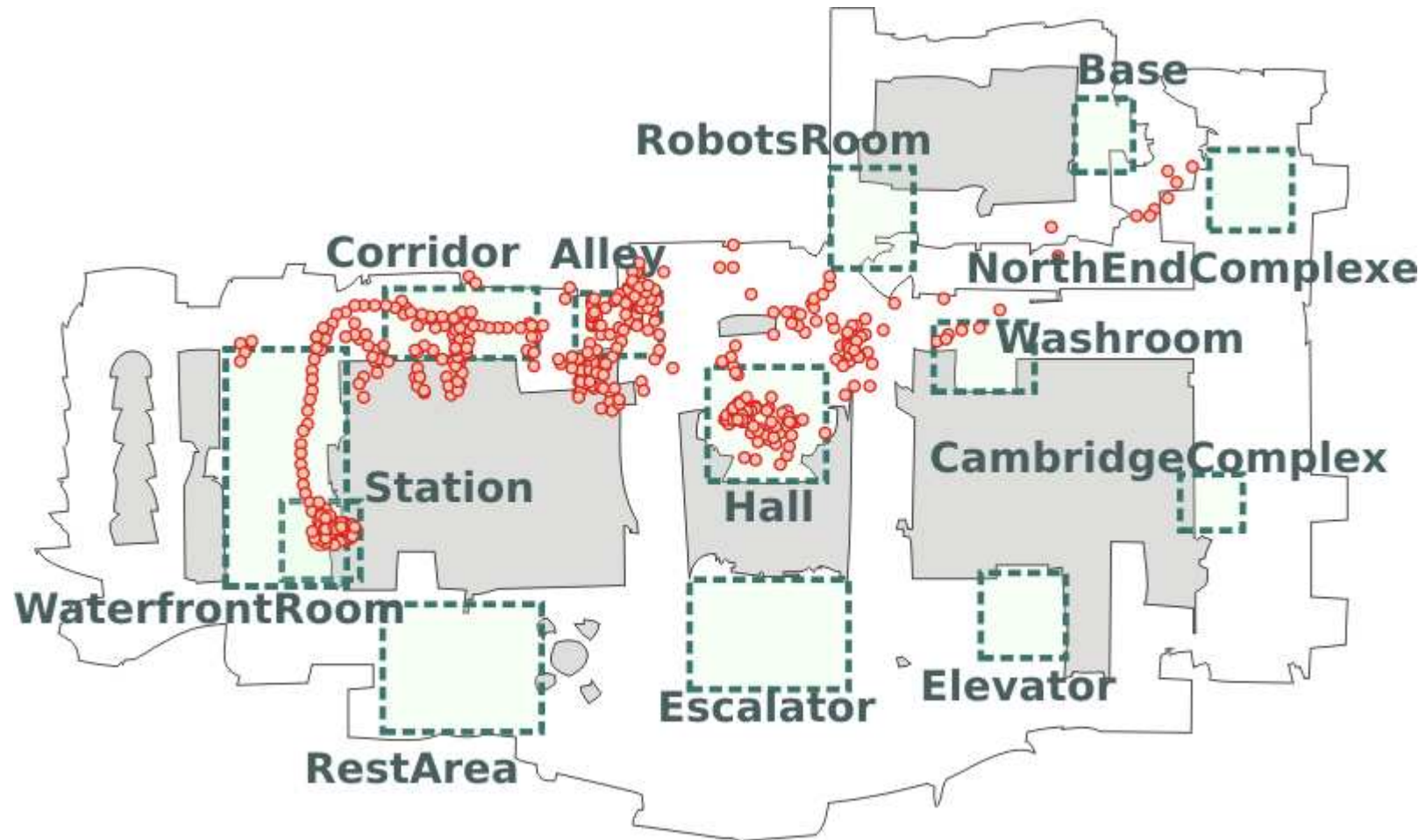


# Expérience de préparation





# Trajet du robot lors de la démonstration



# Problèmes d'intégration

**Capteurs**

**Bruit : des personnes sont devant le robot et obstruent sa vue...**

**Extraction et interprétation**

**Localisation à un mauvais endroit**

**Modèle de l'environnement**

**Modèle inexacte**

**Raisonnement  
Prise de décision**

**Imcapacité à prendre une bonne décision, ou Même juste prendre une décision**

**Actions, commandes motrices**

**Mauvaises actions**

**Actionneurs et sorties**

**Effets indésirés**



# Autre application : jeux vidéo

~~Capteurs~~

~~Extraction et  
interprétation~~

Modèle de  
l'environnement

Raisonnement  
Prise de décision

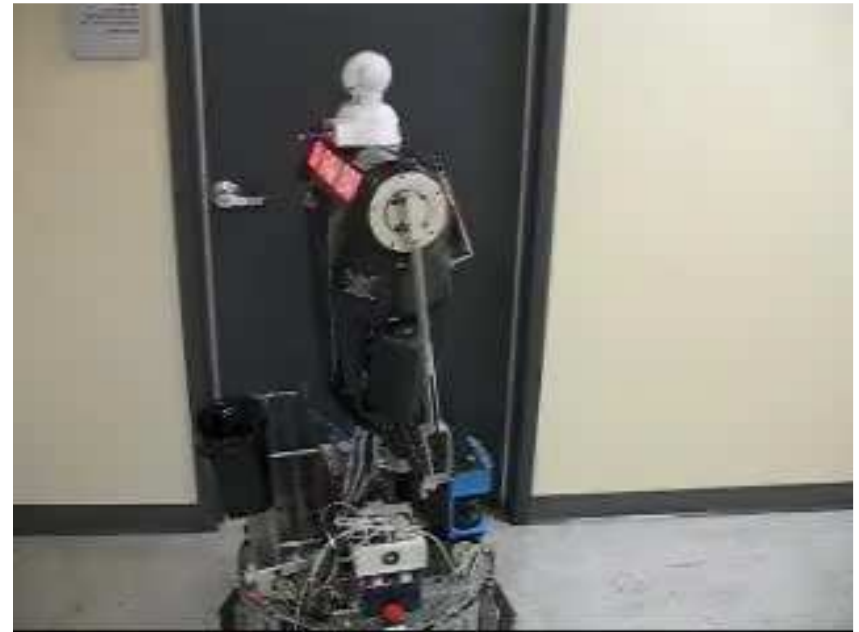
Actions,  
commandes  
motrices

~~Actionneurs  
et sorties~~

# Ces défis vous intéressent ?

- TP3 (projet libre)
- Projets IFT593/693
- Bourse CRSNG 1<sup>er</sup> cycle en recherche
  - Minimum 5625 \$ (bourse non imposable)
  - Durée de 16 semaines
  - Peut être ou ne pas être un stage coop
  - Moyenne de B-
  - Excellente expérience avant la maîtrise
  - [www.crsng.ca](http://www.crsng.ca) ou [eric.beaudry@usherbrooke.ca](mailto:eric.beaudry@usherbrooke.ca)
- Applications
  - Planification de tâche, de chemin ou de trajectoire
  - Robotique et systèmes intelligents
  - Jeux vidéo





# Questions ?

<http://planiart.usherbrooke.ca/~eric/>

Merci au **CRSNG** et au FQRNT  
pour leur support financier

