Robotique autonome

Tâches et architecture

Francis Colas

Introduction

Navigation

- aller à un endroit défini ;
- explorer l'environnement pour en faire une carte ;
- pas de décision de l'endroit où aller.

Introduction

Navigation

- aller à un endroit défini ;
- explorer l'environnement pour en faire une carte ;
- pas de décision de l'endroit où aller.

Planification de haut niveau

- enchainement d'actions simples,
- pour former un comportement plus complexe.

Introduction

Navigation

- aller à un endroit défini ;
- explorer l'environnement pour en faire une carte ;
- pas de décision de l'endroit où aller.

Planification de haut niveau

- enchainement d'actions simples,
- pour former un comportement plus complexe.

Objectif de la séance

- représentation de connaissances ;
- planification de haut niveau ;
- architectures de contrôle.

1

Représentation de connaissances

Connaissances

Représentation de connaissances

- description du robot et de son évolution,
- description de l'environnement et de son évolution,
- description des actions ;
- nécessairement formelle ;
- pour raisonner et planifier.

Connaissances

Représentation de connaissances

- description du robot et de son évolution,
- description de l'environnement et de son évolution,
- description des actions ;
- nécessairement formelle ;
- pour raisonner et planifier.

Langages formels

- plusieurs langages ;
- basés sur la logique (prédicats ou ordre-un).

PDDL

Planning Domain Definition Language

- développé pour une compétition de planification ;
- basé sur des propositions logiques ;
- définition du domaine ;
- notions :
  - concepts (lieu, objet. . .),
  - propositions (être quelque part, . . .),
  - actions :
    - paramètres,
    - préconditions,
    - effets ;

PDDL

Planning Domain Definition Language

- développé pour une compétition de planification ;
- basé sur des propositions logiques ;
- définition du domaine ;
- notions :
  - concepts (lieu, objet. . .),
  - propositions (être quelque part, . . .),
  - actions :
    - paramètres,
    - préconditions,
    - effets ;
- définition de l'instance du problème :
  - ensemble d'objets,
  - état initial,
  - but souhaité ;

PDDL

Planning Domain Definition Language

- développé pour une compétition de planification ;
- basé sur des propositions logiques ;
- définition du domaine ;
- notions :
  - concepts (lieu, objet. . .),
  - propositions (être quelque part, . . .),
  - actions :
    - paramètres,
    - préconditions,
    - effets ;
- définition de l'instance du problème :
  - ensemble d'objets,
  - état initial,
  - but souhaité ;
- syntaxe proche de Lisp.

Exemple de PDDL

Prendre une balle – définitions

```
(define (domain gripper-strips)
    (:predicates (room ?r)
                 (ball ?b)
                 (gripper ?g)
                 (at-robby ?r)
                 (at ?b ?r)
                 (free ?g)
                 (carry ?o ?g))
    ...
)
```

Exemple de PDDL

Prendre une balle – actions

```
(define (domain gripper-strips)
    ...
    (:action move
     :parameters (?from ?to)
     :precondition (and (room ?from) (room ?to)
                        (at-robby ?from))
     :effect (and (at-robby ?to)
                  (not (at-robby ?from))))
    (:action pick
     :parameters (?obj ?room ?gripper)
     :precondition (and (ball ?obj) (room ?room) (at ?obj, ?room)
                        (gripper ?gripper) (at-robby ?room)
                        (free ?gripper))
     :effect (and (carry ?obj ?gripper) (not (at ?obj ?room))
                  (not (free ?gripper))))
    ...
)
```

Exemple de PDDL

Prendre une balle – problème

```
(define (problem strips-gripper1)
    (:domain gripper-strips)
    (:objects room1 room2 ball1 ball2 hand)
    (:init (room room1)
           (room room2)
           (ball ball1)
           (ball ball2)
           (gripper hand)
           (at-robby room1)
           (free hand)
           (at ball1 room1)
           (at ball2 room1))
    (:goal (at ball1 room2)))
```

Extensions

Extensions

- tâches de haut niveau :
  - *Hierarchical Task Network* ;

Extensions

Extensions

- tâches de haut niveau :
  - *Hierarchical Task Network* ;
- représentation du temps :
  - *Linear Temporal Logic* : $\Box$ (*always*), $\Diamond$ (*eventually*)...
  - *Interval Algebra* : {d} (*during*), {p} (*precedes*)...

Extensions

Extensions

- tâches de haut niveau :
  - *Hierarchical Task Network* ;
- représentation du temps :
  - *Linear Temporal Logic* : $\Box$ (*always*), $\Diamond$ (*eventually*)...
  - *Interval Algebra* : {d} (*during*), {p} (*precedes*)...
- représentation de l'espace :
  - *Region Connection Calculus* : {DC} (*disconnected*), {PO} (*partially overlapping*)...
  - *Augmented Rectangle Algebra* : B $\langle$ p[5, 13], p $\rangle$ A.

2

Planification de haut niveau

Planification de haut niveau

Planification de haut niveau

- construction d'une liste d'action,
- pour résoudre un problème défini,
- dans un domaine donné ;
- application de preuve automatique.

Planification de haut niveau

Planification de haut niveau

- construction d'une liste d'action,
- pour résoudre un problème défini,
- dans un domaine donné ;
- application de preuve automatique.

Principes de résolution

Planification de haut niveau

Planification de haut niveau

- construction d'une liste d'action,
- pour résoudre un problème défini,
- dans un domaine donné ;
- application de preuve automatique.

Principes de résolution

- *forward chaining* :
  - génération du graphe des déductions à partir de l'état initial ;

Planification de haut niveau

Planification de haut niveau

- construction d'une liste d'action,
- pour résoudre un problème défini,
- dans un domaine donné ;
- application de preuve automatique.

Principes de résolution

- *forward chaining* :
  - génération du graphe des déductions à partir de l'état initial ;
- *backward chaining* :
  - à partir du but recherché.

Strips

STRIPS

- *STanford Research Institute Problem Solver* (1971) ;
- syntaxe différente de PDDL (plus restreinte) ;
- moteur de résolution basé sur le *backward chaining*.

Strips

STRIPS

- *STanford Research Institute Problem Solver* (1971) ;
- syntaxe différente de PDDL (plus restreinte) ;
- moteur de résolution basé sur le *backward chaining*.

Prendre une balle

```
((pick ball1 room1 hand)
 (move room1 room2)
 (drop ball1 room2 hand))
```

Planification probabiliste

Planification probabiliste

- gestion de l'incertitude ;
- actions : relations probabilistes entre états ;
- planification comme inférence.

Planification probabiliste

Planification probabiliste

- gestion de l'incertitude ;
- actions : relations probabilistes entre états ;
- planification comme inférence.

MDPs et POMDPs

- *Markov Decision Processes* ;
- automate non déterministe ;
- fonction de récompense ;
- construction d'une politique :
  - fonction de l'état vers les actions ;
- *Partially Observable Markov Decision Processes* : incertitude sur l'état.

Conclusion sur la planification de tâches

Représentation de connaissances

- définition du monde et du robot ;
- langage abstrait exprimant les relations ;
- problème d'ancrage des symboles.

Planification

- état de départ et but à atteindre ;
- résolution comme raisonnement.

3

Architectures de contrôle

Architecture de contrôle

Besoin

- intégration perception, décision, action ;

Architecture de contrôle

Besoin

- intégration perception, décision, action ;
- intégration de différents niveaux :
  - planification de tâches,
  - planification de mouvement,
  - évitement d'obstacle,
  - asservissement moteur. . .

Architecture de contrôle

Besoin

- intégration perception, décision, action ;
- intégration de différents niveaux :
  - planification de tâches,
  - planification de mouvement,
  - évitement d'obstacle,
  - asservissement moteur...
- échelles de temps variées :
  - temps réel,
  - milliseconde,
  - seconde,
  - quelques minutes...

Principes structurels

Principes structurels

- modularité :
  - réduction de la complexité,
  - algorithmes et représentations spécialisés pour des rôles particuliers ;

Principes structurels

- modularité :
  - réduction de la complexité,
  - algorithmes et représentations spécialisés pour des rôles particuliers ;
- hiérarchie :
  - couches de comportements de plus en plus complexes,
  - réactivité à bas niveau,
  - difficulté de spécifier la hiérarchie ;

Principes structurels

- modularité :
  - réduction de la complexité,
  - algorithmes et représentations spécialisés pour des rôles particuliers ;
- hiérarchie :
  - couches de comportements de plus en plus complexes,
  - réactivité à bas niveau,
  - difficulté de spécifier la hiérarchie ;
- concurrence :
  - perception et action en parallèle,
  - planification à haut niveau et réaction à l'imprévu. . .

Principes structurels

- modularité :
  - réduction de la complexité,
  - algorithmes et représentations spécialisés pour des rôles particuliers ;
- hiérarchie :
  - couches de comportements de plus en plus complexes,
  - réactivité à bas niveau,
  - difficulté de spécifier la hiérarchie ;
- concurrence :
  - perception et action en parallèle,
  - planification à haut niveau et réaction à l'imprévu. . .
- communication :
  - synchrone ou asynchrone,
  - passage de message,
  - client-serveur,
  - mémoire partagée.

Shakey

Shakey

- *Artificial Intelligence Center of Stanford Research Institute* (1966–1972) ;
- premier robot mobile générique avec raisonnement ;
- développement d'A*, du graphe de visibilité, de la transformée de Hough et STRIPS.

Shakey

Shakey

Shakey

- *Artificial Intelligence Center of Stanford Research Institute* (1966–1972) ;
- premier robot mobile générique avec raisonnement ;
- développement d'A*, du graphe de visibilité, de la transformée de Hough et STRIPS.

Architecture

- *sense-plan-act* (SPA) :
  - les capteurs servent à la perception,
  - la modélisation du monde à la planification,
  - le plan à la définition de la commande ;
- délibérative.

Shakey

Shakey

Shakey

- *Artificial Intelligence Center of Stanford Research Institute* (1966–1972) ;
- premier robot mobile générique avec raisonnement ;
- développement d'A*, du graphe de visibilité, de la transformée de Hough et STRIPS.

Architecture

- *sense-plan-act* (SPA) :
  - les capteurs servent à la perception,
  - la modélisation du monde à la planification,
  - le plan à la définition de la commande ;
- délibérative.

Limitations

- une boucle unique monolithique ;
- la commande n'utilise pas les capteurs.

Shakey

Architecture de subsomption

Architecture de subsomption

- subsomption : inclusion/généralisation de concepts ;
- ensemble de comportements,
- organisés en couches,
- pouvant inhiber les comportements des couches inférieures ;
- basée sur les comportements.

Architecture de subsomption

Architecture de subsomption

- subsomption : inclusion/généralisation de concepts ;
- ensemble de comportements,
- organisés en couches,
- pouvant inhiber les comportements des couches inférieures ;
- basée sur les comportements.

Comportements

- automates à état fini ;
- reliant capteurs et actuateurs ;

Architecture de subsomption

Architecture de subsomption

- subsomption : inclusion/généralisation de concepts ;
- ensemble de comportements,
- organisés en couches,
- pouvant inhiber les comportements des couches inférieures ;
- basée sur les comportements.

Comportements

- automates à état fini ;
- reliant capteurs et actuateurs ;

Limitations

- pas de mémoire ou de représentation ;
- difficulté de gérer un système complexe d'inhibition de comportements.

Architecture de subsomption

Exemple de Brooks (1986)

0 éviter/fuir les obstacles ;

Architecture de subsomption

Exemple de Brooks (1986)

0 éviter/fuir les obstacles ;

1 déambulation ;

Architecture de subsomption

Exemple de Brooks (1986)

0 éviter/fuir les obstacles ;

1 déambulation ;

2 exploration : aller à des endroits observés ;

Architecture de subsomption

Exemple de Brooks (1986)

0 éviter/fuir les obstacles ;

1 déambulation ;

2 exploration : aller à des endroits observés ;

3 construction d'une carte et planification de chemin ;

Architecture de subsomption

Exemple de Brooks (1986)

0. éviter/fuir les obstacles ;
1. déambulation ;
2. exploration : aller à des endroits observés ;
3. construction d'une carte et planification de chemin ;
4. repérage des changements dans l'environnement ;

Architecture de subsomption

Exemple de Brooks (1986)

0 éviter/fuir les obstacles ;

1 déambulation ;

2 exploration : aller à des endroits observés ;

3 construction d'une carte et planification de chemin ;

4 repérage des changements dans l'environnement ;

5 raisonnement en termes d'objets identifiables et actions liés à ces objets ;

Architecture de subsomption

Exemple de Brooks (1986)

0 éviter/fuir les obstacles ;

1 déambulation ;

2 exploration : aller à des endroits observés ;

3 construction d'une carte et planification de chemin ;

4 repérage des changements dans l'environnement ;

5 raisonnement en termes d'objets identifiables et actions liés à ces objets ;

6 planification et exécution pour changer l'état de l'environnement ;

Architecture de subsomption

Exemple de Brooks (1986)

0 éviter/fuir les obstacles ;

1 déambulation ;

2 exploration : aller à des endroits observés ;

3 construction d'une carte et planification de chemin ;

4 repérage des changements dans l'environnement ;

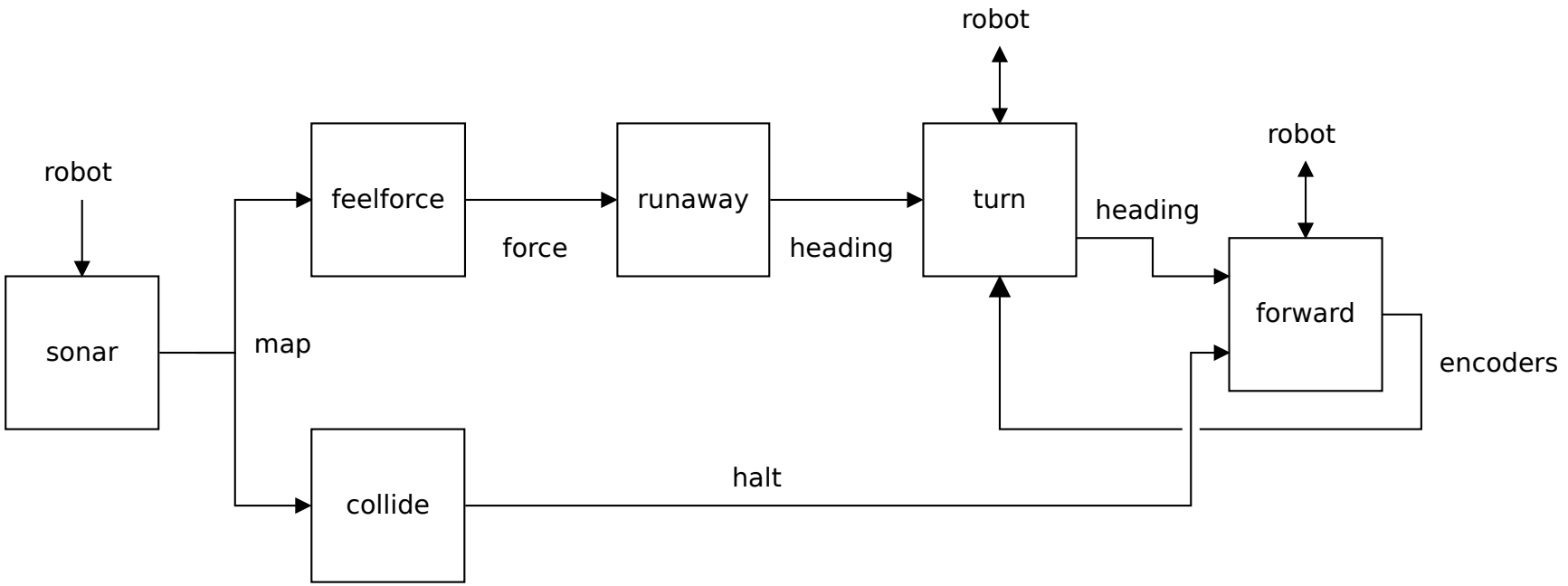
5 raisonnement en termes d'objets identifiables et actions liés à ces objets ;

6 planification et exécution pour changer l'état de l'environnement ;

7 raisonnement à propos du comportement des objets et modification des plans.

Architecture de subsomption

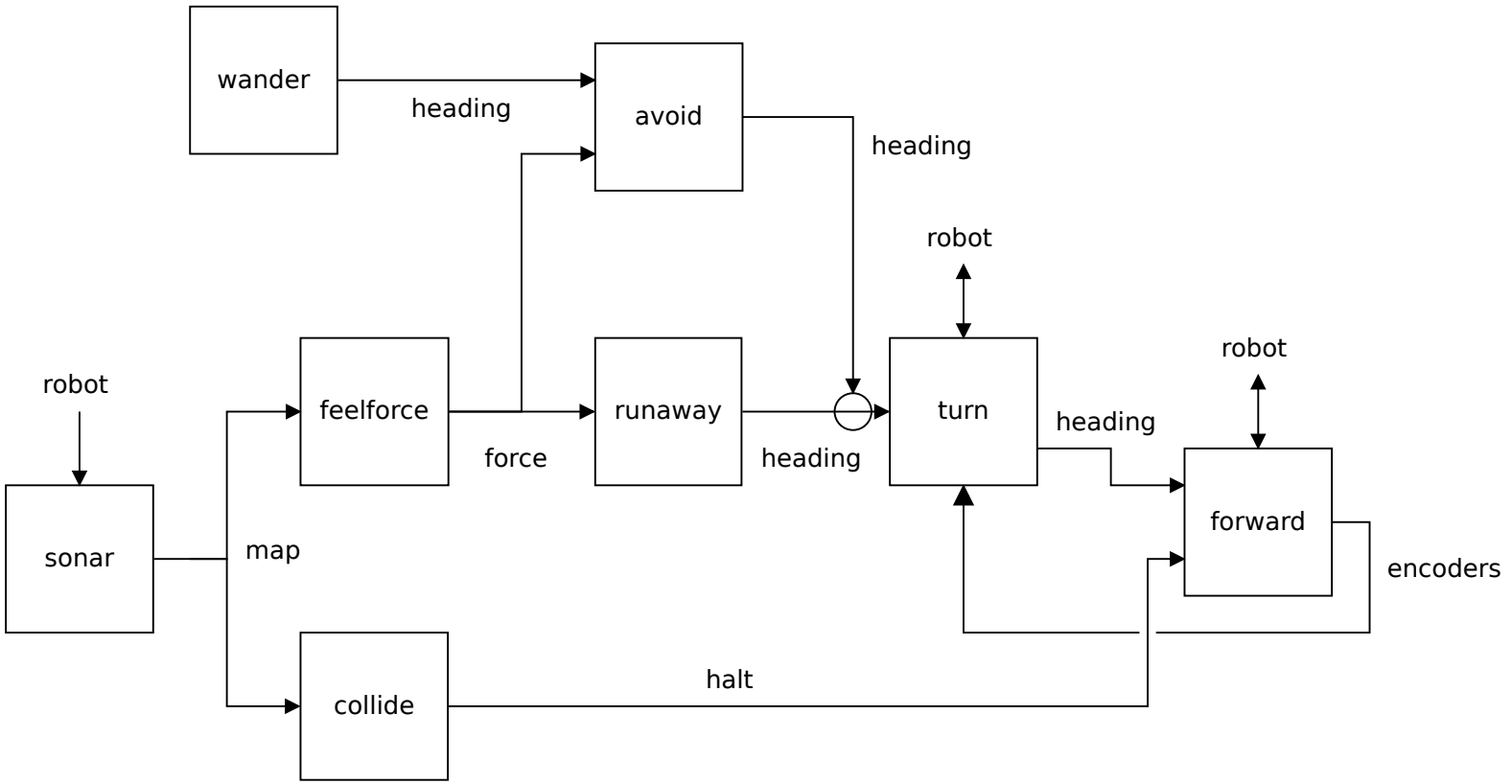
Exemple de Brooks (1986)



Niveau 0 : évitement d'obstacle

Architecture de subsomption

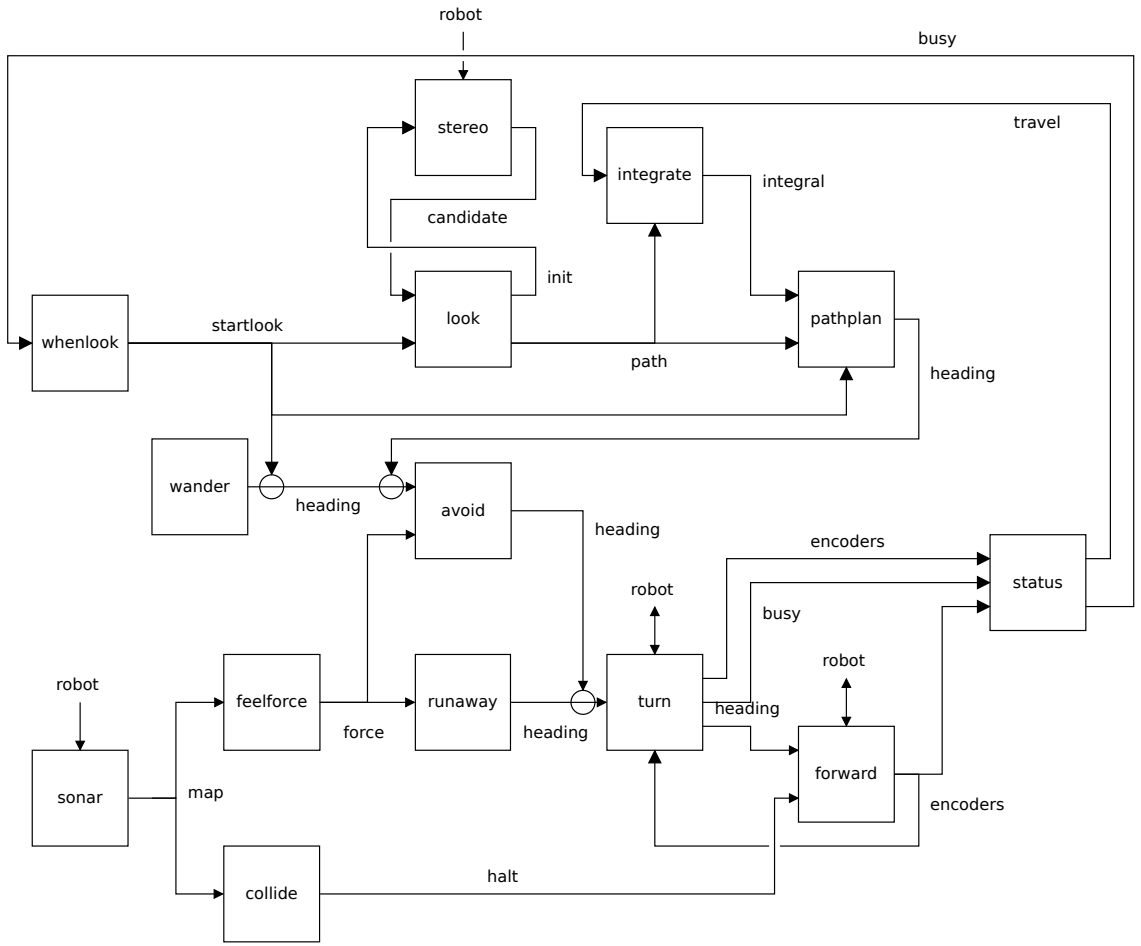
Exemple de Brooks (1986)



Niveau 0 et 1 : déambulation

Architecture de subsomption

Exemple de Brooks (1986)



Niveau 0, 1 et 2 : exploration

Architectures en couches

Couches

- association comportement et représentation ;
- comportement :
  - buts venant de la couche supérieure,
  - commandes vers la couche inférieure ;
- représentation :
  - suivi de la couche inférieure,
  - donnés abstraites pour la couche supérieure ;
- couches supérieures à des fréquences plus faibles.

Architectures en couches

Architecture en trois tiers

- planification :
  - gère les buts haut niveau,
  - maintient une représentation abstraite ;
- *executive* :
  - décomposition des tâches,
  - suivi et synchronisation des tâches,
  - gestion des ressources,
  - mémoire immédiate ;
- comportements :
  - réactifs ou états limités,
  - interaction avec les capteurs et les actuateurs.

Conclusion sur les architectures de contrôle

Shakey

- exclusivement délibératif ;
- lent et rigide ;
- approche descendante (*top-down*).

Conclusion sur les architectures de contrôle

Shakey

- exclusivement délibératif ;
- lent et rigide ;
- approche descendante (*top-down*).

Subsomption

- architecture basée sur des comportements ;
- mémoire limitée et conception complexe ;
- approche ascendante (*bottom-up*).

Conclusion sur les architectures de contrôle

Shakey

- exclusivement délibératif ;
- lent et rigide ;
- approche descendante (*top-down*).

Subsomption

- architecture basée sur des comportements ;
- mémoire limitée et conception complexe ;
- approche ascendante (*bottom-up*).

Architectures en couches

- interaction comportements et représentation ;
- intégration réactif et délibératif.

Conclusion sur les architectures de contrôle

Typologie

- délibérative : *think, then act* ;
- réactive : *don't think, (re)act* ;
- hybride : *think and act concurrently* ;
- basée sur les comportements : *think the way you act*.

4

Conclusion

Conclusion

Fonctions de haut niveau

- représentation des actions et états de haut niveau ;
- raisonnement dans ce domaine ;
- intégration avec le bas niveau.

Bibliographie

STRIPS et PDDL

- Fikes et Nilsson, *STRIPS : A New Approach to the Application of Theorem Proving to Problem Solving*, AI, 1971.
- McDermott *et al.*, *PDDL – The Planning Domain Definition Language*, Yale CVC, 1998.
- Fox et Long, *PDDL2.1 : An Extension to PDDL for Expressing Temporal Planning Domains*, JAIR, 2003.

Subsomption

- Brooks, *A robust layered control system for a mobile robot*, RA, 1986.

Livres

- Siciliano *et al.*, *Springer Handbook of Robotics*, Springer 2016.

Merci de votre attention.

Des questions ?