Navigation autonome en environnement dynamique : une approche par déformation de trajectoire

Thèse de doctorat en Mathématiques et Informatique
présentée par Vivien DELSART
dirigée par Thierry FRAICHARD
INRIA, DGA, CNRS-LIG, Université de Grenoble

UNIVERSITÉ DE GRENOBLE

Robotique – Navigation Autonome

But

- Se déplacer en toute autonomie

Problématiques

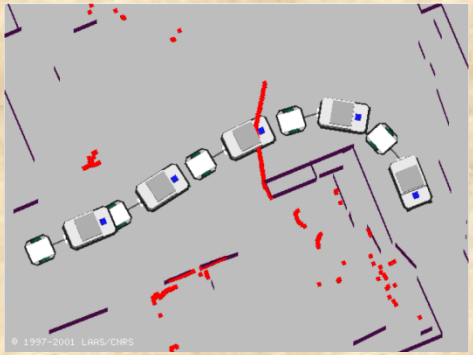
- Où suis et qu'y a-t-il autour de moi?
- Comment rejoindre le but à partir de ma position initiale?
- Comment exécuter ce mouvement?

Détermination du mouvement

Objectifs

- Mener à un but déterminé
- Respecter les contraintes du mouvement du robot
- Eviter toutes collisions

Approche proposée : Déformation de trajectoire



Plan

I. Approches de navigation existantes
II. Déformation de mouvement
III. Génération de trajectoires en temps contraint
IV. Résultats expérimentaux
V. Conclusions et perspectives

Plan

I. Approches de navigation existantes
II. Déformation de mouvement
III. Génération de trajectoires en temps contraint
IV. Résultats expérimentaux
V. Conclusions et perspectives

Approches délibératives

- Calcul d'un plan complet vers le but
- Nécessite un modèle complet de l'environnement
- Détermination du mouvement à priori
- Par graphe

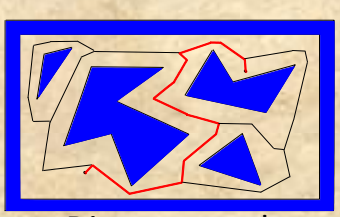
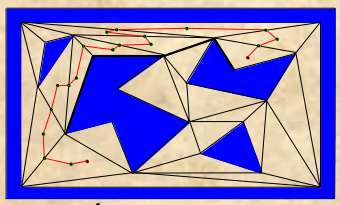
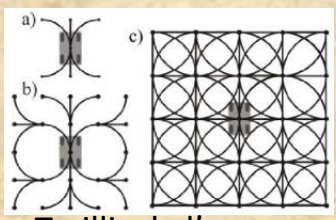


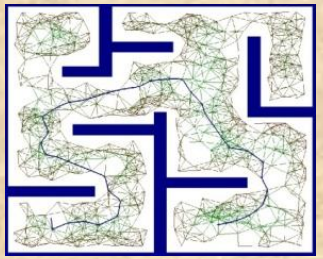
Diagramme de Voronoi [Aurenhammer91]



Décomposition cellulaires [Lingas82]

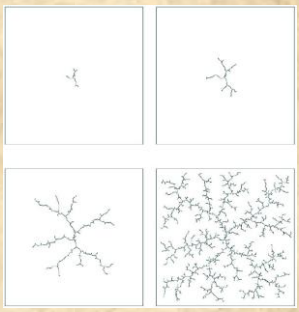


Treillis de l'espace d'état [Pivtoraiko05]

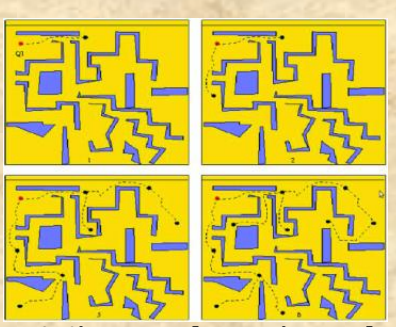


PRM [Kravaki96]

- Par arbre



RRT [Lavalle98]

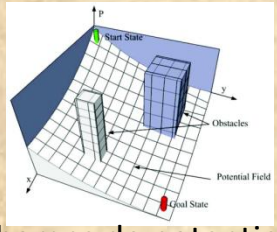


Fil d'Ariane [Bessière93]

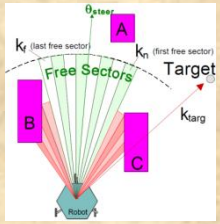
- Problème : complexité
- Limitations de sa mise en œuvre en environnement dynamique

Approches réactives

- Calcul du mouvement pour le pas de temps suivant
- Acquisition des informations sur l'environnement au cours du temps



Champs de potentiels [Khatib86]



VFH [Borenstein91]

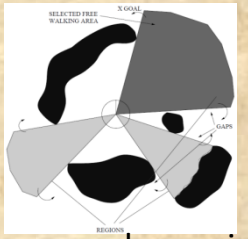
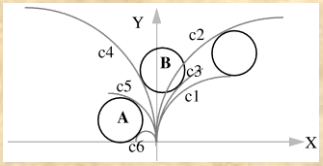
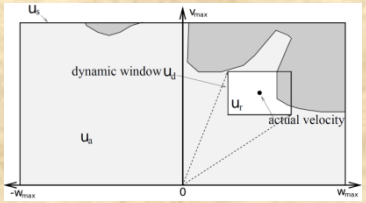


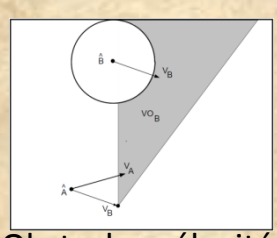
Diagramme de proximité [Minguez00]



Méthode courbure-vélocité [Simmons96]



Fenêtre dynamique [Fox97]



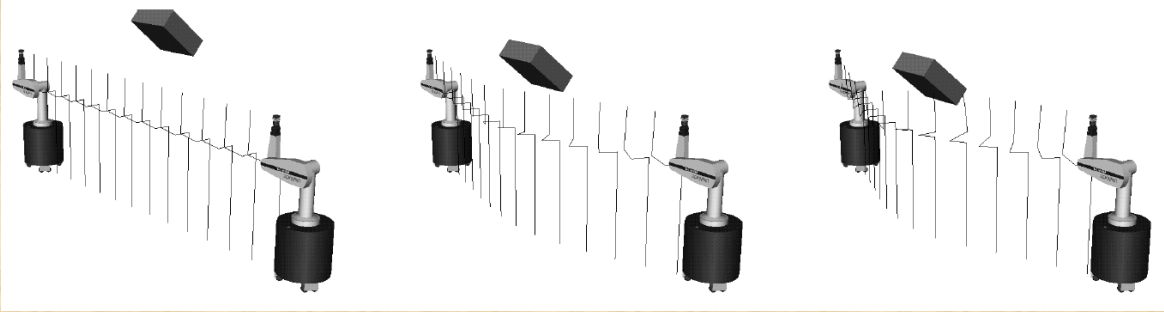
Obstacles vélocité [Fiorini98]

Etats de collision inévitables [Martinez-Gomez08]

- Problème : convergence vers le but

Approches intermédiaires

- Approches hybrides : combinaisons d'approches délibératives/réactives
- Replanification
- Planification partielle de mouvement
- Déformation de mouvement



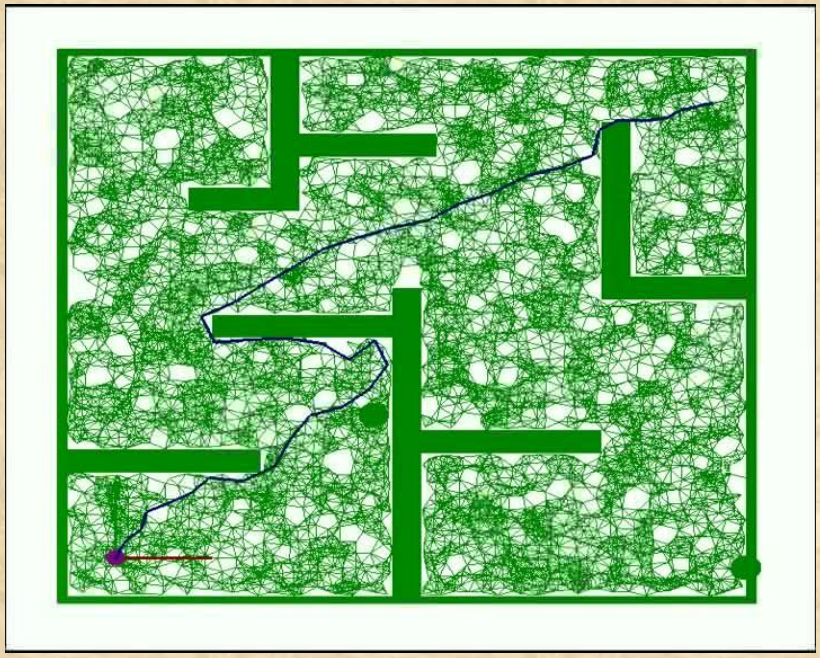
Plan

I. Approches de navigation existantes
II. Déformation de mouvement
III. Génération de trajectoires en temps contraint
IV. Résultats expérimentaux
V. Conclusions et perspectives

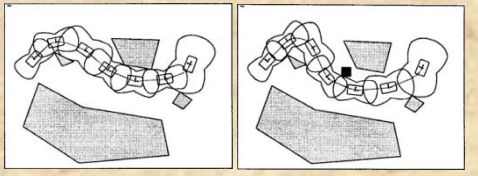
Déformation de mouvement

Principe général

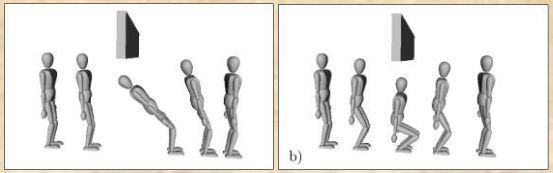
- Mouvement initial complet calculé
- Modifie le mouvement suivi au cours de son exécution
  - Évitement d'obstacles
  - Conserve la convergence vers le but



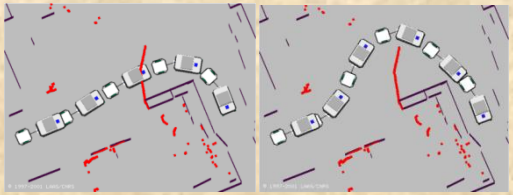
Déformation – approches existantes



Bande élastique NH [Khatib97]



Bande élastique [Brock02]

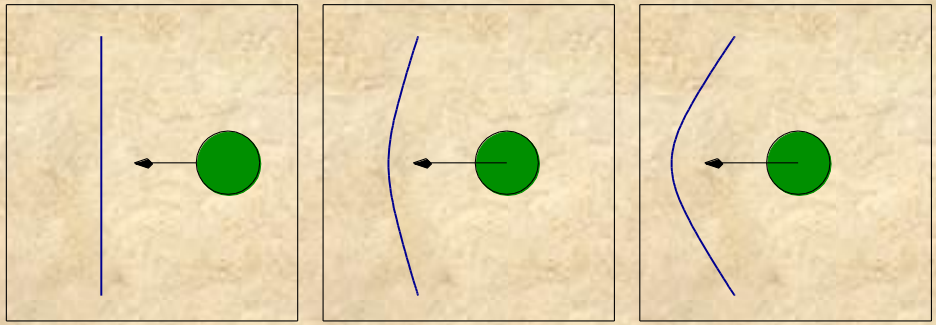


Déformation variationnelle [Lamiraux04]

CHOMP[Ratliff09]

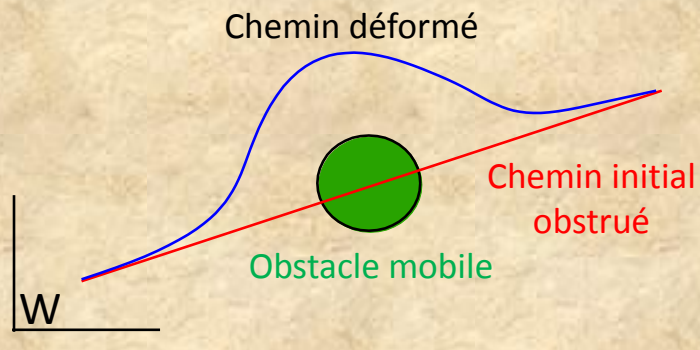
Chemin : courbe géométrique de l'espace

– Déformations à partir des informations sur la position courante des obstacles uniquement

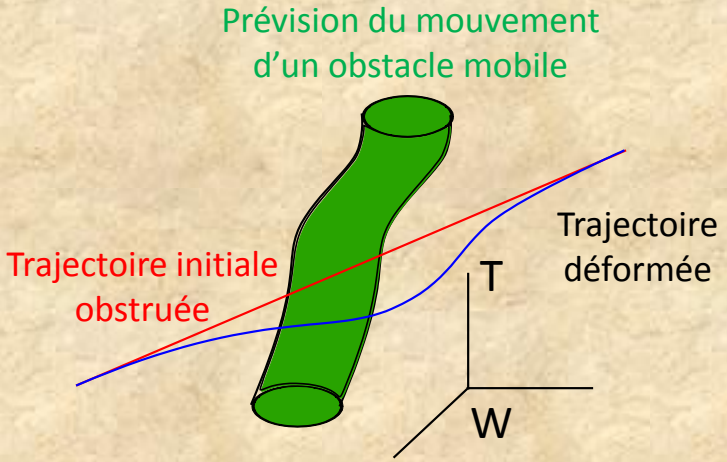


Déformation de trajectoire - Teddy

Trajectoire = chemin géométrique paramétré par le temps



Déformation de chemin

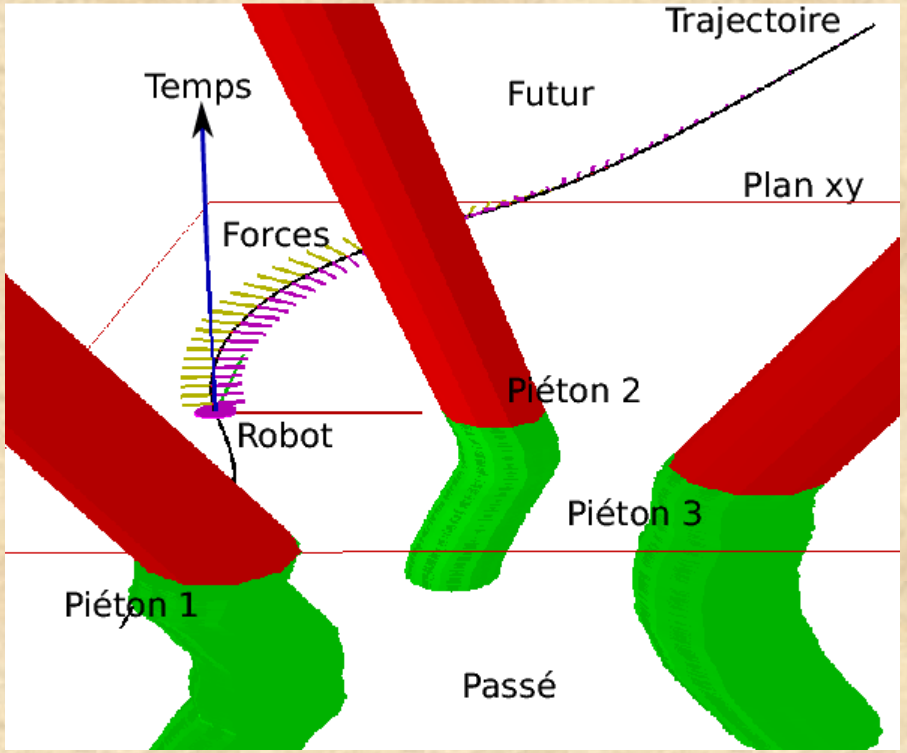


Déformation de trajectoire

- Raisonnement sur l'espace-temps
- Déformations spatiales et temporelles

Comment déformer?

- Discrétisation de la trajectoire en une séquence de nœuds (états-temps)
- Modèle prévisionnel du futur des obstacles mobiles
- Forces appliquées sur la trajectoire
- Décroissance de l'amplitude des forces au cours du temps
- Difficulté : Maintien de la faisabilité de la trajectoire



Comment déformer?

Entrées : $\{n_k, n_{k+1} \dots n_N\}$, modèle prévisionnel $B(t_m, t_h)$

// Applications des forces répulsives et élastiques

pour chaque n_i faire

$$n'_i = n_i + F_{rep}(n_i) + F_{ela}(n_i)$$

fpour

// Restauration de la cohérence temporelle

pour chaque n'_i faire

$$n''_i = n'_i + F_{ct}(n'_i)$$

fpour

// Maintient de la faisabilité par gén. de traj.

pour chaque n''_i faire

$$\hat{n}_i = n''_i + F_{conn}(n''_i)$$

fpour

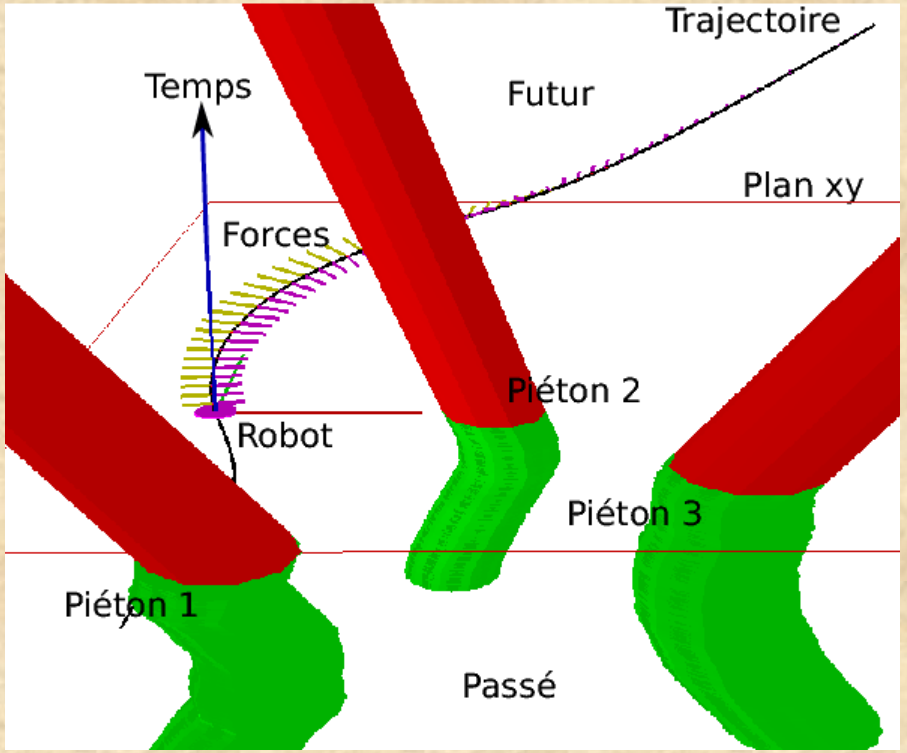
// Vérification de la validité de la trajectoire

si *non valide* $\{\hat{n}_k, \hat{n}_{k+1} \dots \hat{n}_{\hat{N}}\}$ alors

Appel du planificateur global

fsi

Retourner $\{\hat{n}_k, \hat{n}_{k+1} \dots \hat{n}_{\hat{N}}\}$



Forces répulsives

- But : évitement d'obstacles
- Définies dans SxT

En pratique :

- Points de contrôle dans WxT
- Champ de potentiel dans WxT :

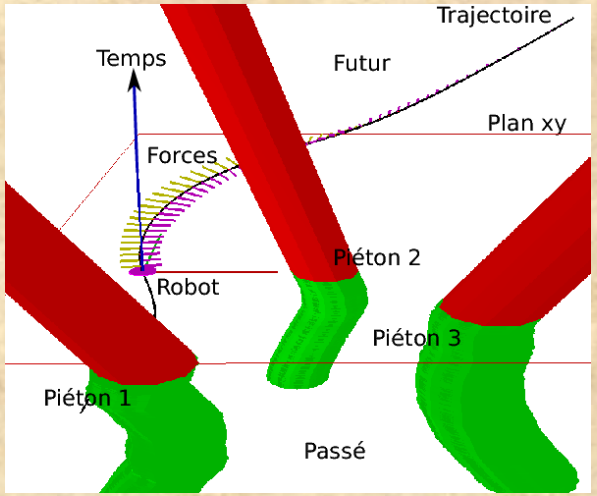
$$V_{rep}(c_i^j) = \begin{cases} k_{rep} k_h(t_i)(d_0 - d_{wt}(c_i^j))^2 & \text{si } d_{wt}(c_i^j) < d_0 \\ 0 & \text{sinon} \end{cases}$$

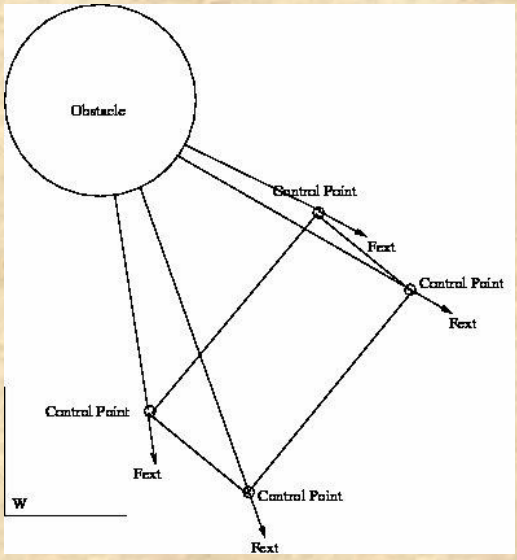
- Dans CxT :

$$F_{rep}^{CT}(q, t_i) = \sum_{j=1}^{r} J_{c_i^j}^T(q, t_i) F_{rep}^{WT}(c_i^j)$$

$$J_{c_i^j}^T(q, t_i) = \begin{pmatrix} \frac{\partial q^1}{\partial p_1^j} & \cdots & \frac{\partial q^1}{\partial p_m^j} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ \frac{\partial q^n}{\partial p_1^j} & \cdots & \frac{\partial q^n}{\partial p_m^j} & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}$$

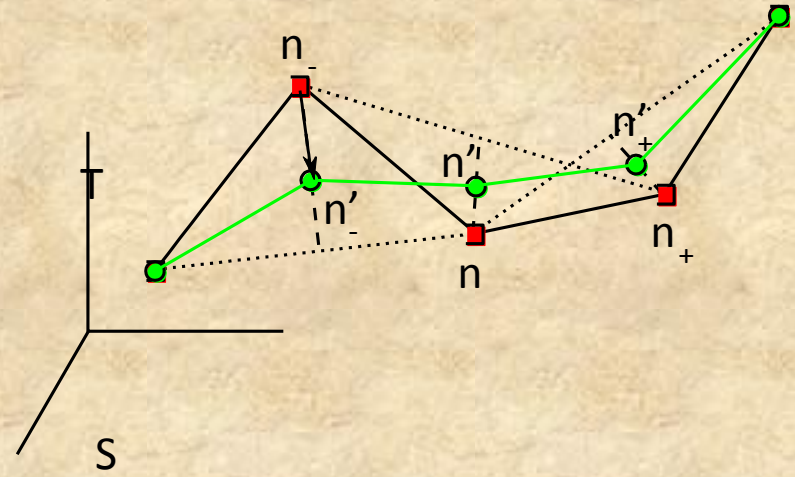
- Dans SxT : paramètres d'état inchangés





Forces élastiques

But : limiter la longueur du chemin et les variations le long de la trajectoire



$$n_- = (s_-, t_-), \quad n = (s, t), \quad n_+ = (s_+, t_+)$$

$$F_{ela}^{WT}(n) = \begin{cases} k_{ela}(\frac{1}{2}n_- + \frac{1}{2}n_+ - n) & \text{si } d_{wt}(c_i^j) \geq d_0 \ \forall j \in [1; r] \\ 0 & \text{sinon} \end{cases}$$

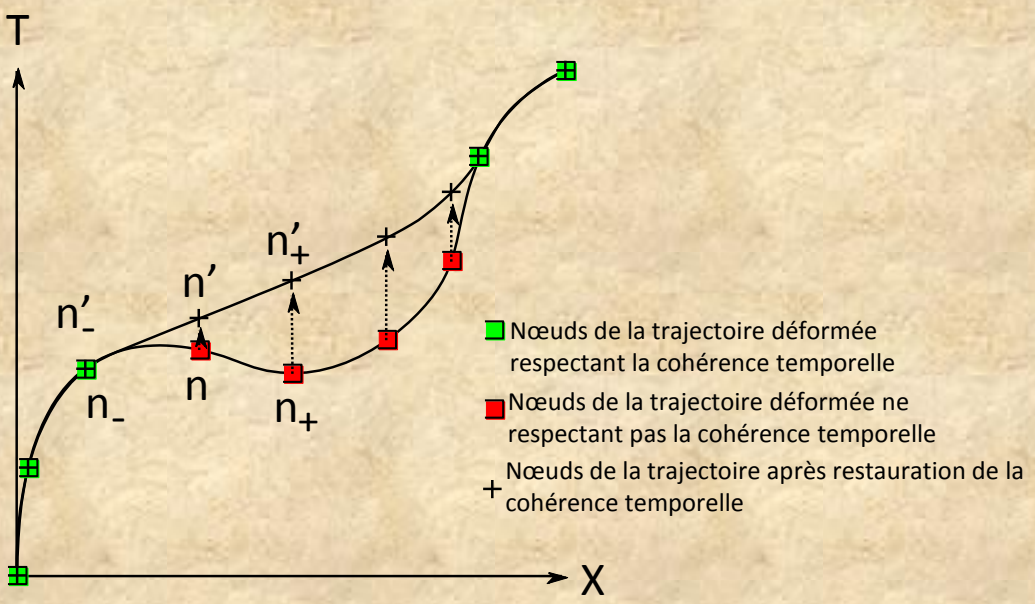
Nœuds de la trajectoire déformée déplacés librement dans SxT

=> Faisabilité du mouvement?

Restauration de la cohérence temporelle

But : s'assurer que chaque nœud de la trajectoire dispose d'un temps minimal pour être atteint à partir de son prédécesseur

- Pour chaque point de contrôle : détermination d'une vitesse max
- Temps de chaque nœud modifié par rapport à son prédécesseur

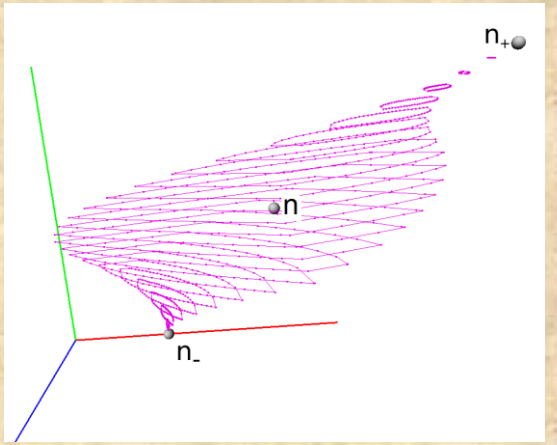


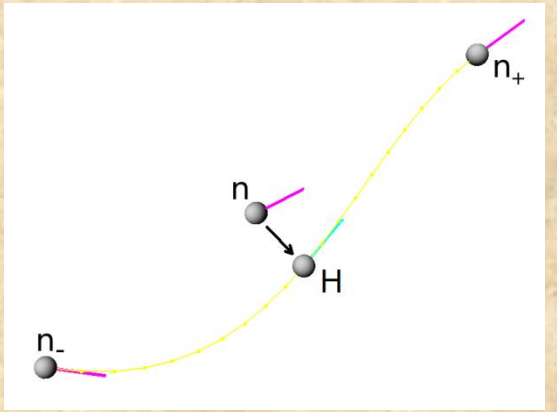
Restauration de la faisabilité par génération de trajectoire (1)

- Critère de faisabilité :
  $n \in R(n_-) \cap R^{-1}(n_+)$
- Calcul d'espace atteignables coûteux

  => génération d'une trajectoire Γ entre n_- et n_+

  Force attractive de n vers l'état-temps défini au temps moyen H = $\Gamma(\frac{t_- + t_+}{2})$
- Problème : si $n_+ \notin R(n_-)$?

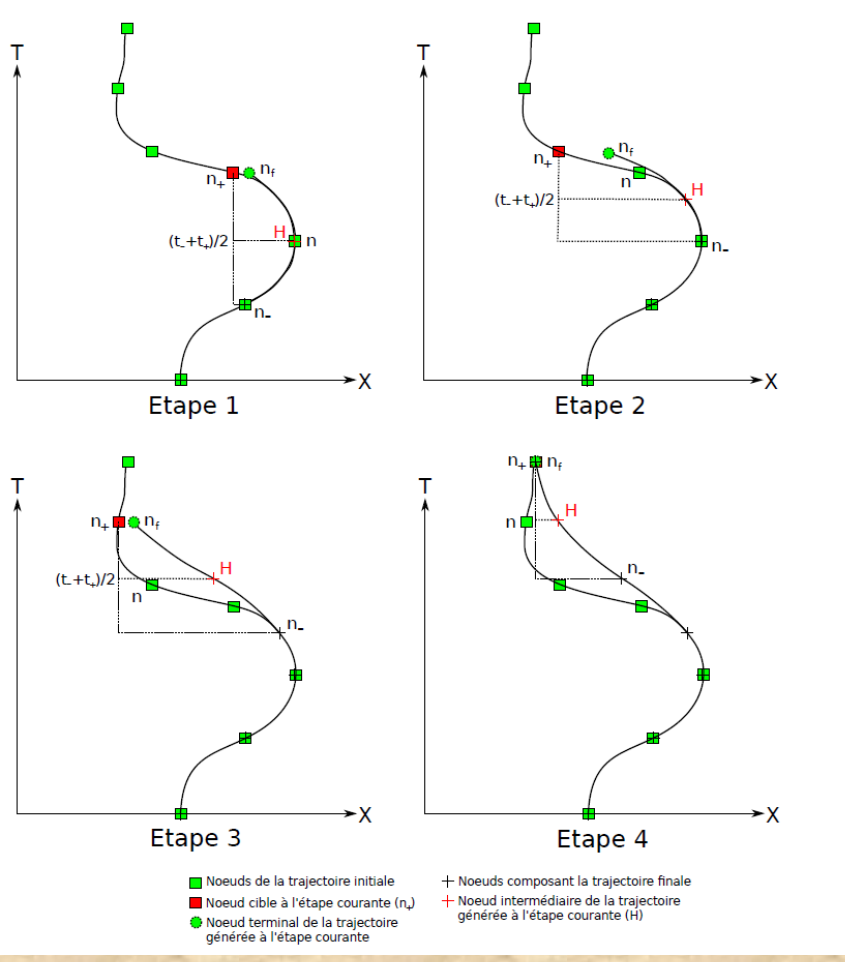




Restauration de la faisabilité par génération de trajectoire (2)

- Calcul d'une trajectoire jusque lorsqu'il est atteignable
- Calcul d'une trajectoire s'arrêtant « aussi près que possible » de n_+ sinon
- Génération de trajectoire entre chaque triplé d'états-temps successifs

Problème : A notre connaissance, un tel générateur de trajectoire n'existait pas



Plan

I. Approches de navigation existantes
II. Déformation de mouvement
III. Génération de trajectoires en temps contraint
IV. Résultats expérimentaux
V. Conclusions et perspectives

Génération de mouvement classique : Objectif

- Déterminer un mouvement pour un système donné
  - Entre deux états s_0 et s_g fixés
  - Respectant les contraintes sur le mouvement du robot
  - En un temps limité

- Dynamique du système robotique :

$$\dot{s} = f(s, u)$$

- Contraint par :

$$\begin{cases} h_s(s) \leq 0 \\ h_u(u) \leq 0 \end{cases}$$

Problème de génération de mouvement :

Trouver : $\tilde{u} : ([0; t_f] \longrightarrow U,\ t \longmapsto u(t))$

Sujet à :

$$\begin{cases} s_f = s_g \\ h_s(s) \leq 0 \\ h_u(u) \leq 0 \end{cases}$$

Ajout de la contrainte sur le temps final - Tiji

Dans le cadre de la déformation :

$\Rightarrow$ Déformation entre deux états s_0 et s_g donnés à des temps t_0 et t_f fixés

Trouver : $\tilde{u} : ([0; t_f] \longrightarrow U,\ t \longmapsto u(t))$

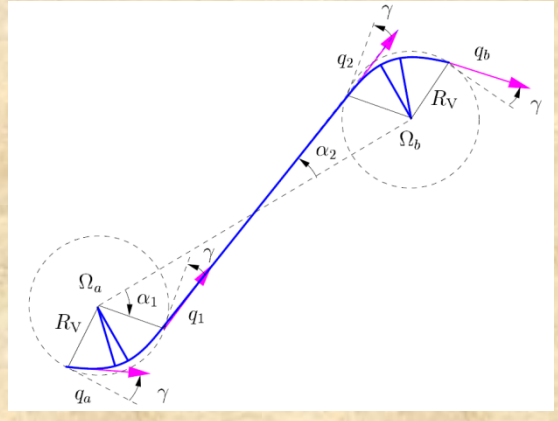
Sujet à : $\begin{cases} s_f = s_g \\ h_u(u) \leq 0 \\ h_s(s) \leq 0 \\ t_f = t_g \qquad (t_{min} \leq t_f \leq t_{max}) \end{cases}$

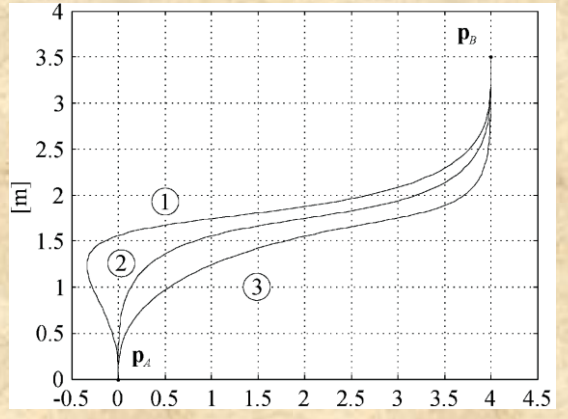
Problème : Aucune garantie de l'existence d'une solution

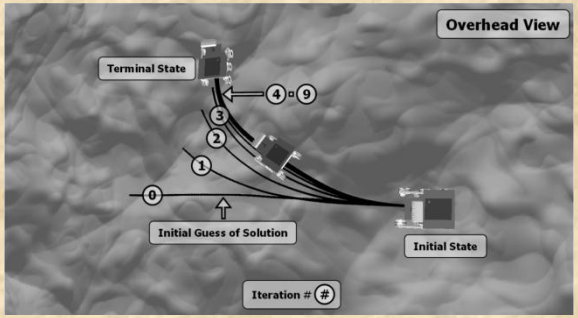
- Calcul d'une trajectoire s'arrêtant « aussi près que possible » du but
- Minimisation de la distance $\|s_f - s_g\|$

Approches de génération de trajectoires existantes

- Combinaisons de primitives simples
- Résolution d'un problème aux limites
- Recherche d'un contrôle optimal





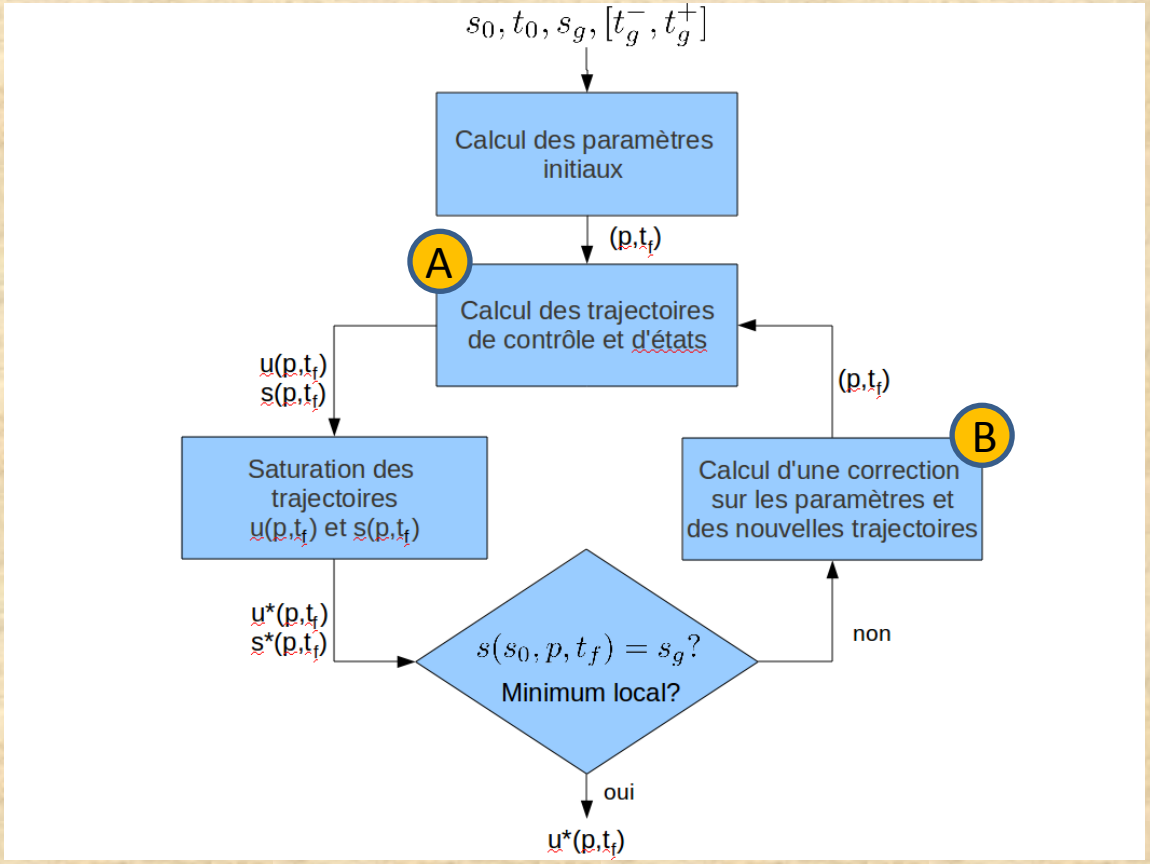


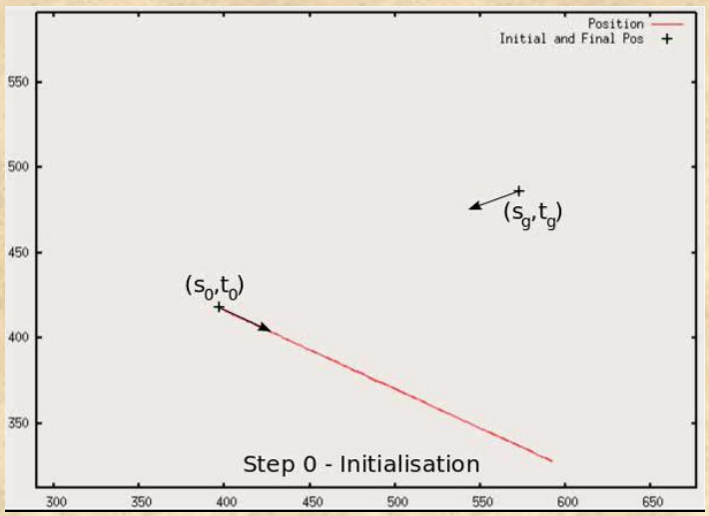
Résolution par méthode variationnelle

Représentation paramétrique des trajectoires de contrôles et d'états

$$u(t) = u(p,t) \quad s(t) = s(p,t)$$

- But : réduction de l'espace de recherche
- Exemples : paramètres d'un polynôme, d'une spline, etc.



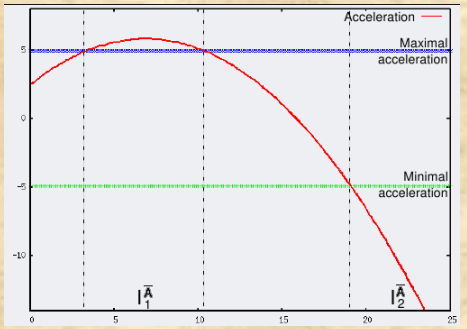


A $$u(t) = u(p,t)$$
$$s(t) = \int_{t_0}^{t_f} f(s,u)dt$$

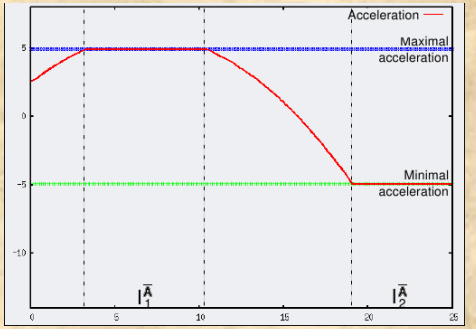
B $$cor_{(p,t_f)} = -\lambda \left[\frac{\partial^2 J}{\partial (p,t_f)^2}\right]^{-1} \left[\frac{\partial J}{\partial (p,t_f)}\right]$$

Saturation et annulation de la trajectoire de contrôles

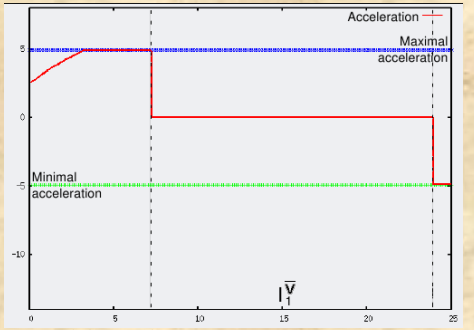
- Détermination des intervalles de temps sur lesquels les contraintes sur le mouvement ne sont pas respectées
- Bornes de contrôles dépassées => saturation du contrôle
- Bornes d'états dépassées => annulation du contrôle
- Profils polynômiaux par morceaux



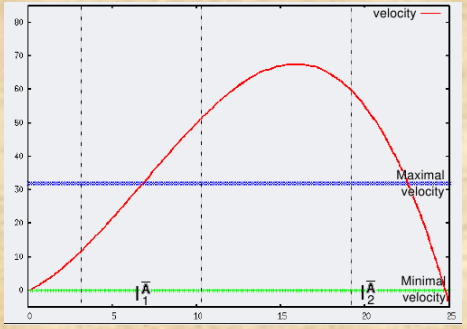
a(t) avant saturation



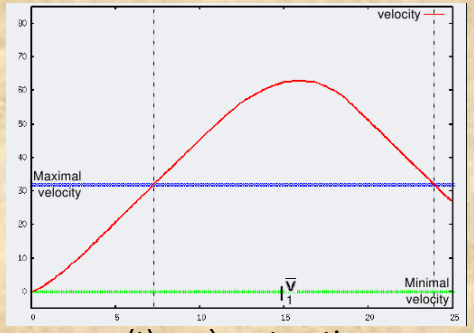
a(t) après saturation de l'accélération



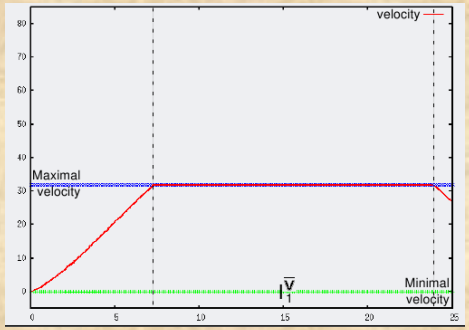
a(t) après saturation et annulation de l'accélération



v(t) avant saturation

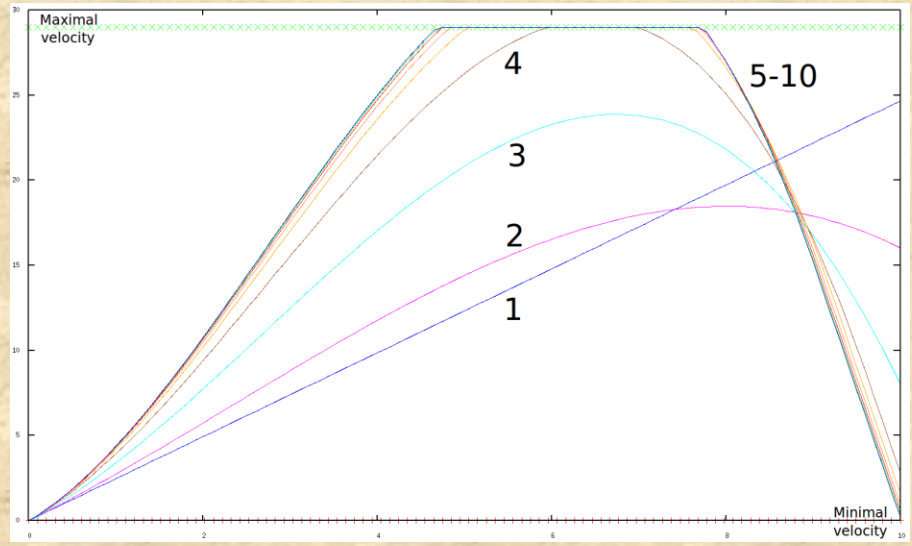


v(t) après saturation de l'accélération

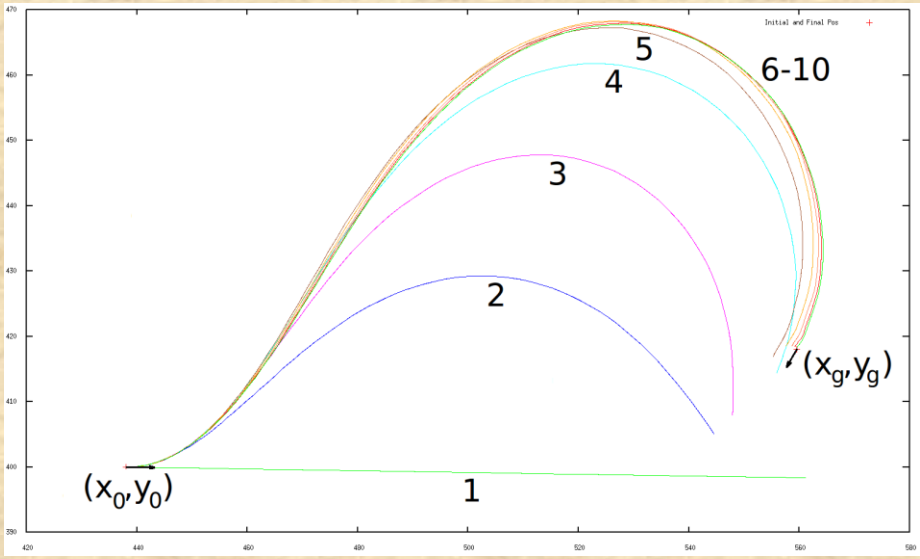


v(t) après saturation et annulation de l'accélération

Exemple



V(t)



XxY

- Application à différents systèmes : robot différentiel, voiture, …

Tiji : Performances

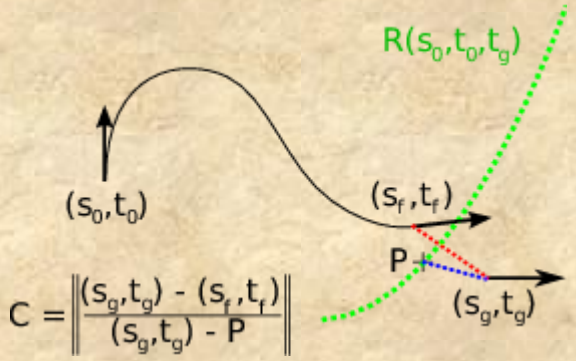
- Cas atteignable

Système	Robot différentiel (sans base IG)	Robot différentiel (avec base IG)	Voiture (avec base IG)
Taux de convergence (%)	94.58	99.80	96.52
Nombre moyen d'étapes requises	6.38	3.23	7.08
Temps moyen de calcul par trajectoire (ms)	4.78	2.84	5.12

- Cas non atteignable

  Nombre d'étapes de convergence max : 20

Système	Robot différentiel	Voiture
Temps moyen requis par trajectoire (ms)	13.33	15.92
C : Coefficient d'évaluation moyen	1.14	1.15



Tiji : Conclusions

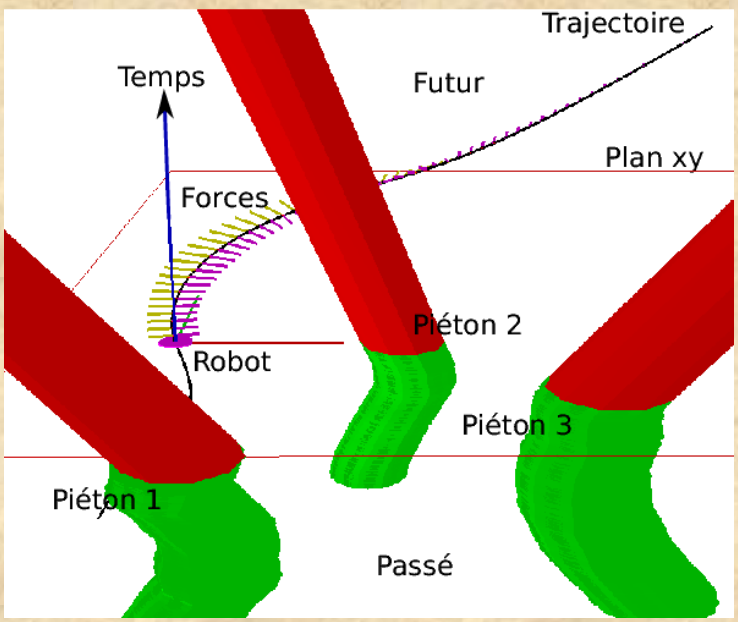
- Calcule une trajectoire entre deux états donnés à des temps fixés
- Méthode variationnelle par optimisation du contrôle d'entrée
- Atteint le but lorsque c'est possible
- Atteint un état « aussi près que possible » du but dans le cas contraire
- Calcul en temps-réel

Plan

I. Approches de navigation existantes
II. Déformation de mouvement
III. Génération de trajectoires en temps contraint
IV. Résultats expérimentaux
V. Conclusions et perspectives

Résultats en simulation

- Core i7 @ 2.6GHz, 6GB RAM, SE : Linux
- Localisation exacte du robot
- Positions et vitesses instantanées exactes des obstacles
- Calcul d'un modèle prévisionnel déterministe en ligne droite



Robot différentiel

- Dynamique :

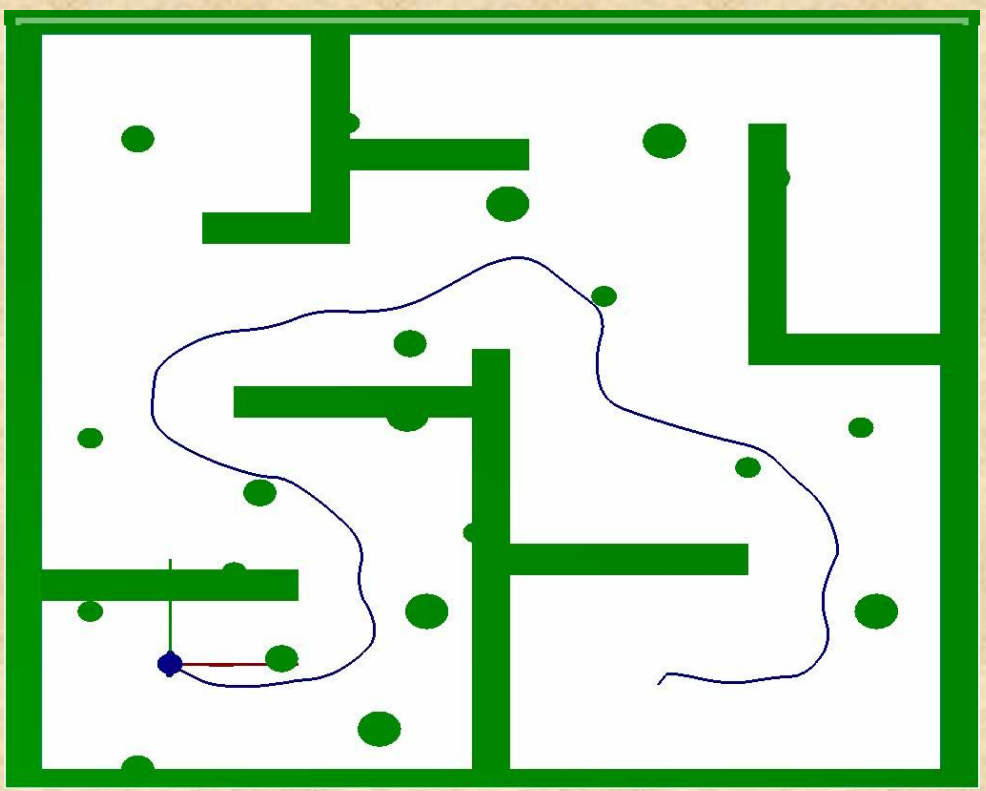
$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\omega}_L \\ \dot{\omega}_R \end{pmatrix} = \begin{pmatrix} v\cos(\theta) \\ v\sin(\theta) \\ \omega \\ \mu_L \\ \mu_R \end{pmatrix}$$

- Contraintes :

$$(\mu_L, \mu_R) \in [-\mu_{max}, \mu_{max}], \quad (\omega_L, \omega_R) \in [-\omega_{max}, \omega_{max}]$$

- Contrôle :

$$(\mu_L, \mu_R)$$



Voiture

- Dynamique :

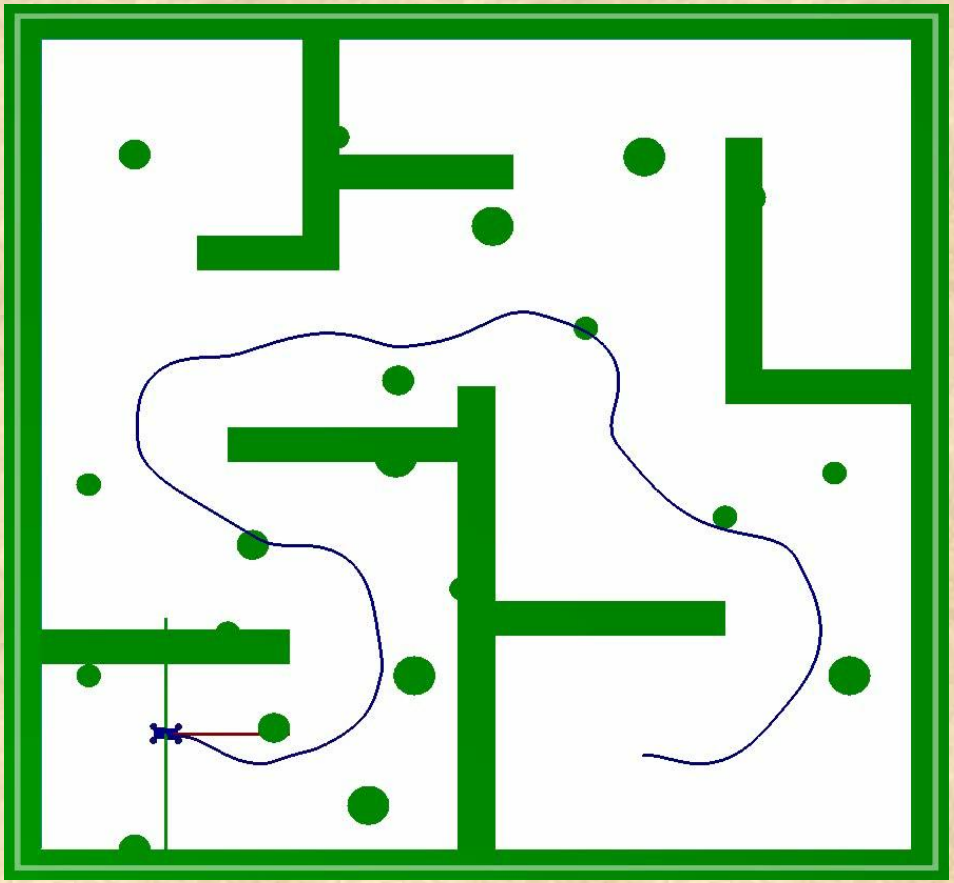
$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \\ \dot{v} \end{pmatrix} = \begin{pmatrix} v\cos(\theta) \\ v\sin(\theta) \\ v\tan(\phi)/L \\ \zeta \\ a \end{pmatrix}$$

- Contraintes :

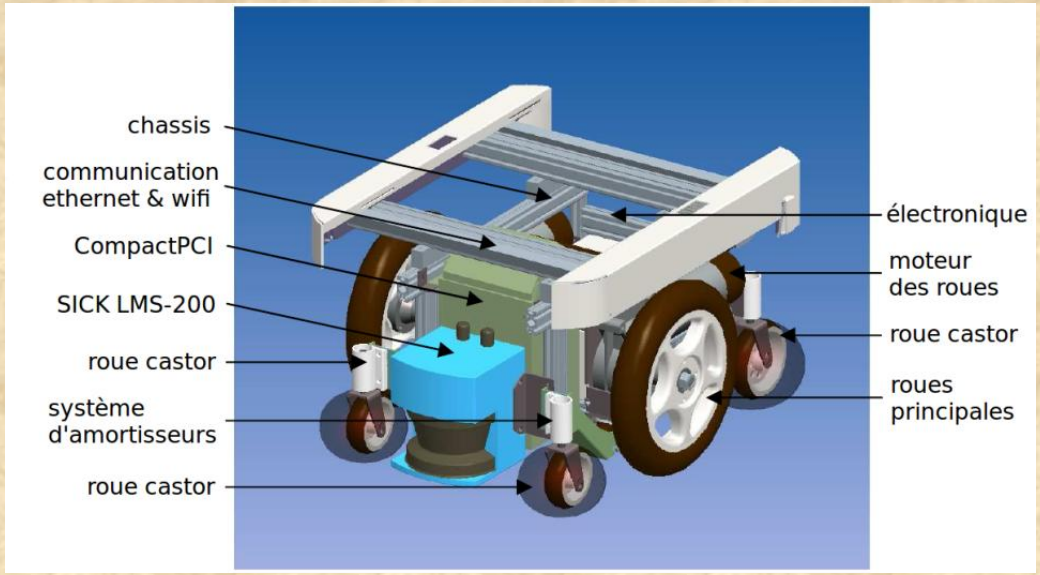
$$v \in [0, v_{max}], \quad |\phi| < \phi_{max}$$
$$|a| < a_{max}, \quad |\zeta| < \zeta_{max}$$

- Contrôle :

$$(a, \zeta)$$

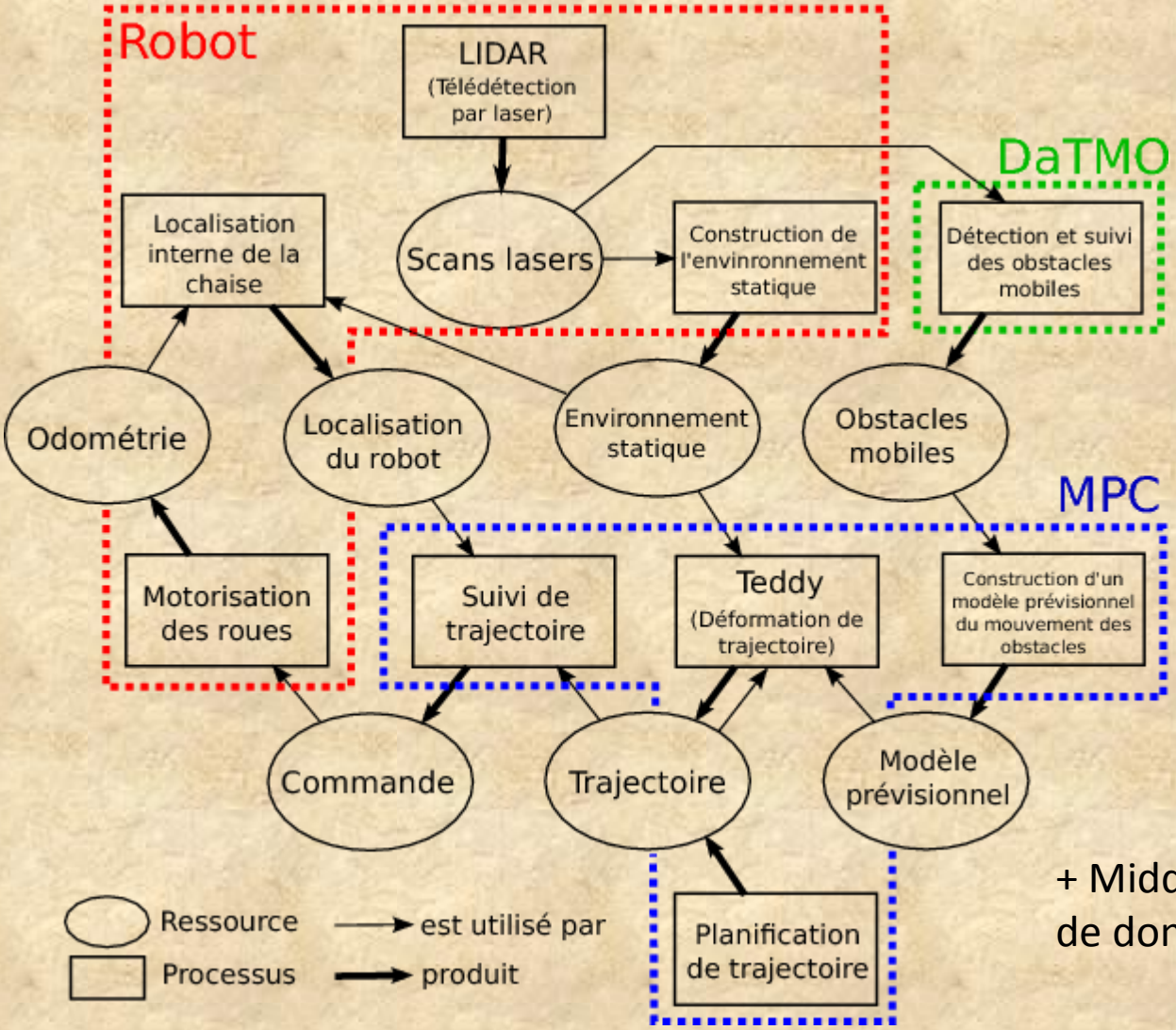


Application à une chaise roulante automatisée



- Dynamique : $\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\omega}_L \\ \dot{\omega}_R \end{pmatrix} = \begin{pmatrix} v\cos(\theta) \\ v\sin(\theta) \\ \omega \\ \mu_L \\ \mu_R \end{pmatrix}$
- Contrôle : (μ_L, μ_R)
- Contraintes : $(\mu_L, \mu_R) \in [-\mu_{max}, \mu_{max}], \quad (\omega_L, \omega_R) \in [-\omega_{max}, \omega_{max}]$

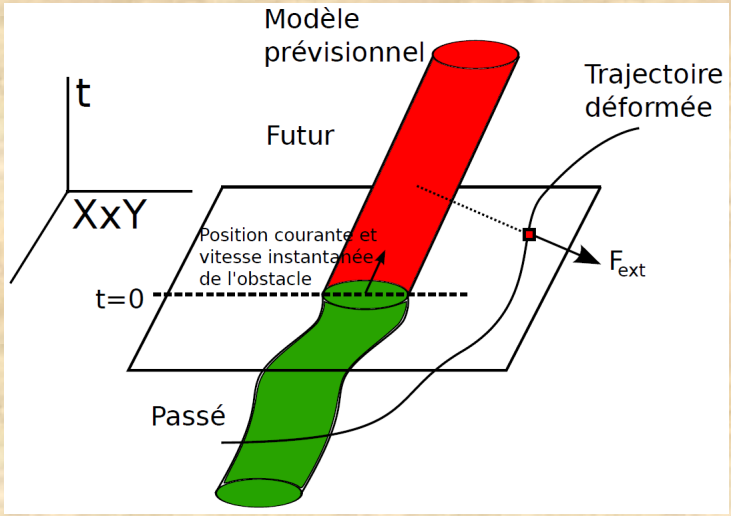
Architecture de navigation

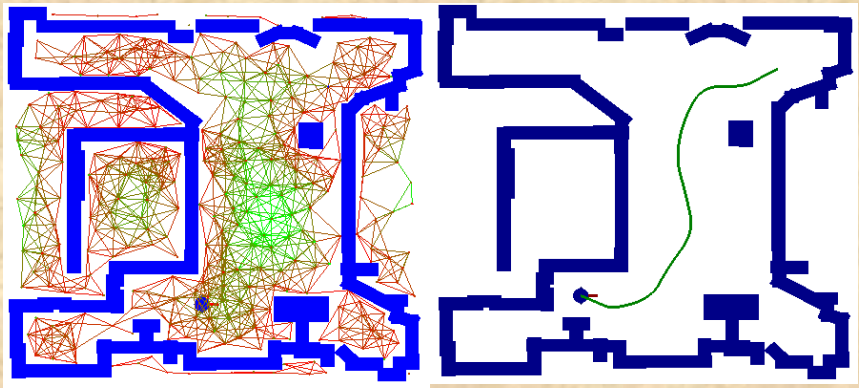


+ Middleware d'échange de données HUGR

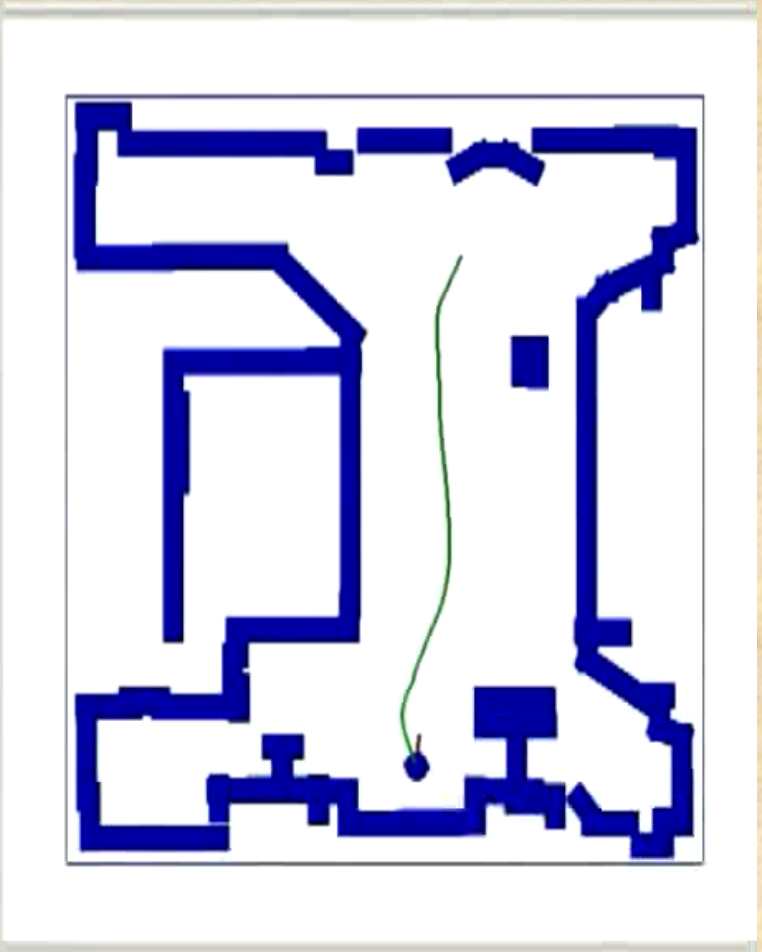
Mise en place des différentes tâches de navigation

- Localisation
  => odométrie + carte
- Environnement statique
  => précalculé
- Environnement dynamique
  - Détection et suivi d'obstacles [VA09]
  - Fournit leur position et leur vitesse instantanée
  - Déduction d'un modèle prévisionnel déterministe
- Planification de trajectoire
  - Recherche de points de passage sur une carte de route
  - Remplacement par une trajectoire faisable générée par Tiji
- Évitement d'obstacles par déformation de trajectoire (Teddy)
- Suivi de trajectoire
  - Recherche d'un contrôle pour rejoindre la trajectoire à suivre basée sur Tiji

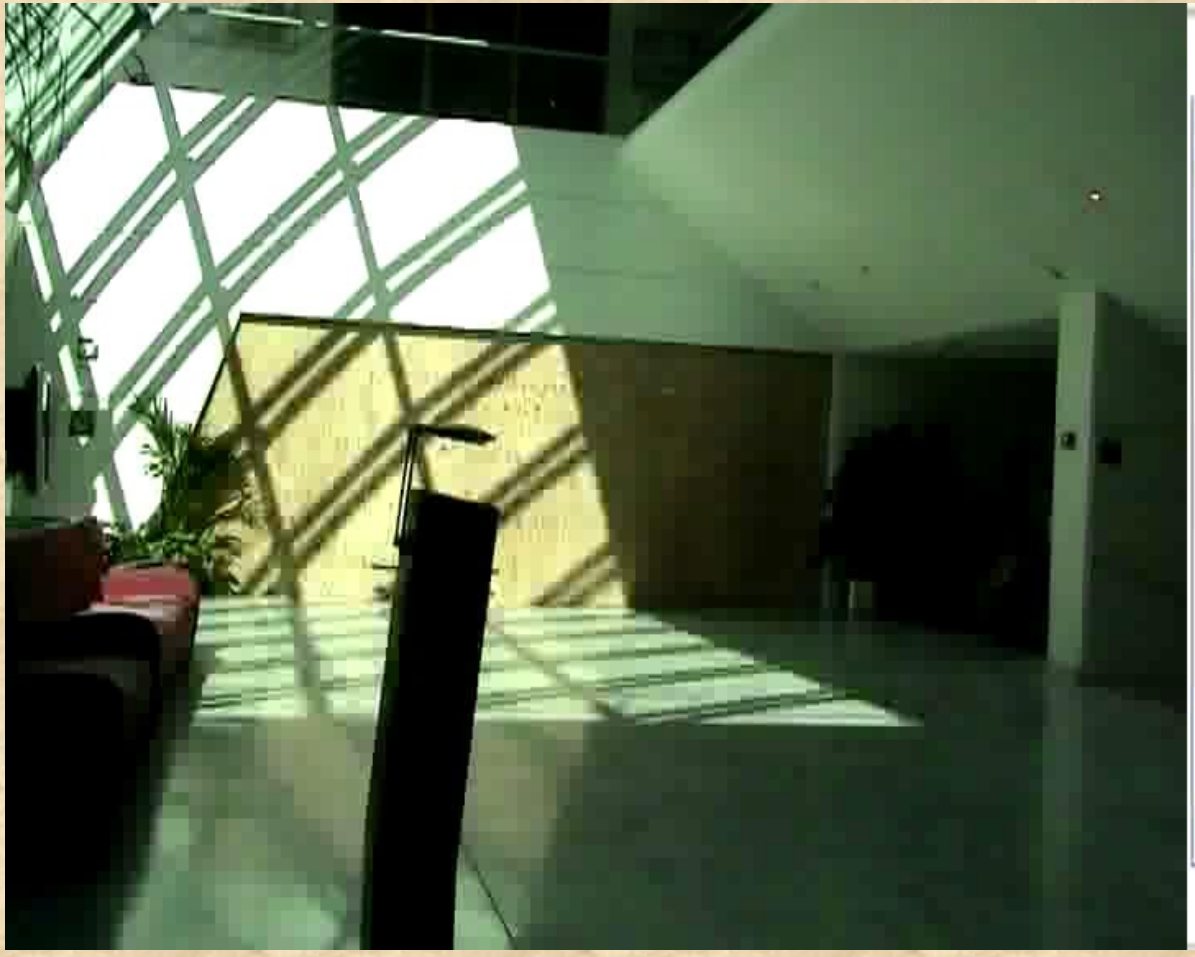


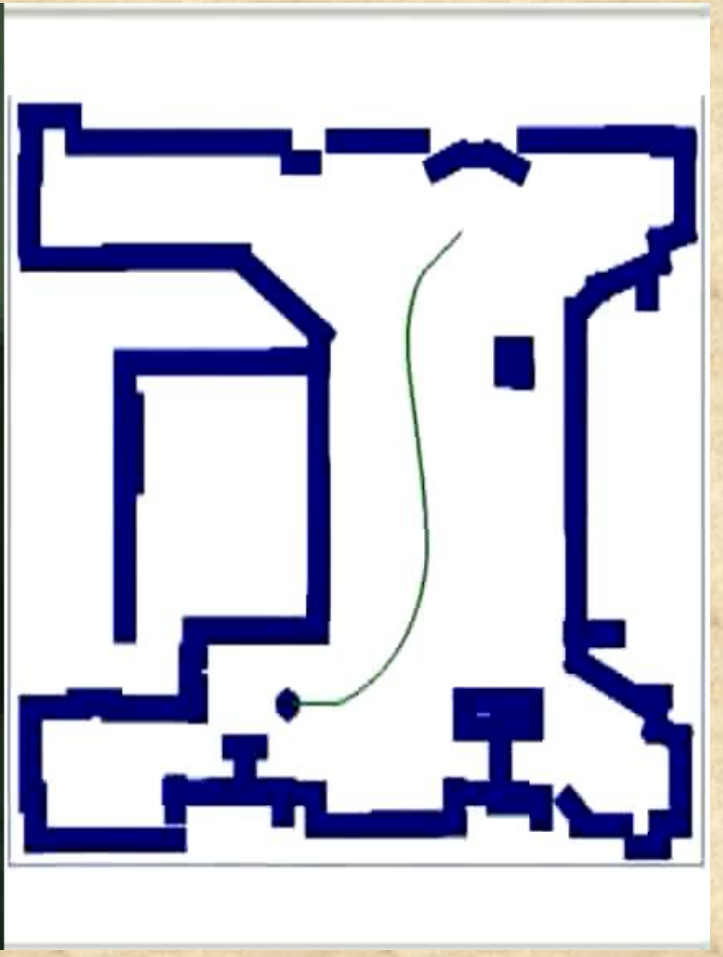


Scénario 1 : Cisaillement



Scénario 2 : Multiple obstacles





Plan

I. Approches de navigation existantes
II. Déformation de mouvement
III. Génération de trajectoires en temps contraint
IV. Résultats expérimentaux
V. Conclusions et perspectives

Résumé des contributions

- Déformation de trajectoire Teddy
  - Conserve les propriétés de la déformation de chemin
  - Utilisation d'un modèle prévisionnel du comportement futur des obstacles mobiles
  - Déformations libres dans SxT : anticipation du mouvement des obstacles mobiles
  - Nécessité d'un générateur de trajectoire en temps contraint
- Générateur de trajectoires Tiji
  - Calcul de trajectoires entre deux états-temps
  - Méthode de génération de trajectoires variationnelle
  - États but non-atteignables : calcul d'une trajectoire s'arrêtant « aussi près que possible » du but
- Validation des résultats sur une chaise roulante automatisée

Perspectives

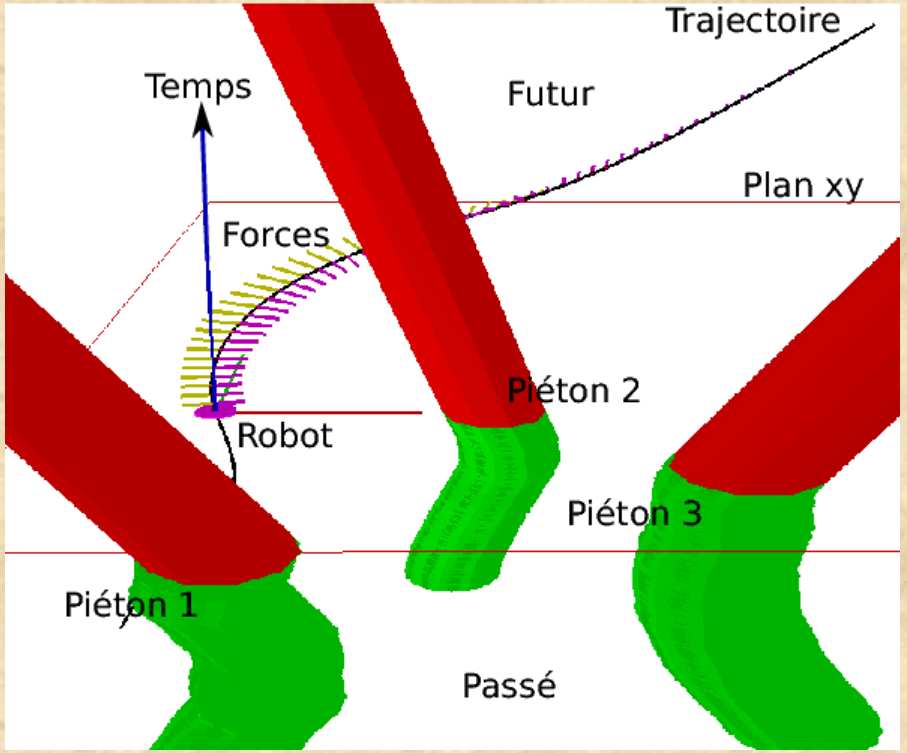
- Amélioration du modèle prévisionnel
- Amélioration des garanties de sécurité (ICS)
- Modélisation de l'incertitude du modèle prévisionnel et du contrôle
- Coordination multi-véhicules par déformation

Questions?

Annexes Teddy

Comment déformer?

Entrées : $\Gamma_{ST}^{m} = \{n_k, n_{k+1} \cdots n_N\}$ **,** modèle prévisionnel $B(t_m, t_h)$
Sorties : $\Gamma_{ST}^{\hat{m}} = \{\hat{n}_k, \hat{n}_{k+1} \cdots \hat{n}_{\hat{N}}\}$
// Applications des forces répulsives et élastiques
$\Gamma_{ST}^{m'} = \emptyset$
pour chaque $n_i \in \Gamma_{ST}^{m}$ **faire**
$n_i^{'} = n_i + \mathbf{F}_{rep}(n_i) + \mathbf{F}_{ela}(n_i)$
$\Gamma_{ST}^{m'} = \Gamma_{ST}^{m'} \cup n_i^{'}$
fpour
// Restauration de la cohérence temporelle
$\Gamma_{ST}^{m''} = RestaureCTemp\ (\Gamma_{ST}^{m'})$
// Maintient de la connectivité de la trajectoire
$\Gamma_{ST}^{m'''} = \emptyset$
pour chaque $n_i^{''} \in \Gamma_{ST}^{m''}$ **faire**
$n_i^{'''} = n_i^{''} + \mathbf{F}_{conn}(n_i^{''})$
$\Gamma_{ST}^{m'''} = \Gamma_{ST}^{m'''} \cup n_i^{'''}$
fpour
// Rééchantillonage de la trajectoire
$\Gamma_{ST}^{\hat{m}} = Rééchantillonne\ (\Gamma_{ST}^{m'''})$
// Vérification de la validité de la trajectoire
si *non valide* $(\Gamma_{ST}^{\hat{m}})$ **alors**
Déterminer un nouveau plan
fsi
Retourner $(\Gamma_{ST}^{\hat{m}})$



Comment déformer?

Entrées : $\{n_k, n_{k+1} \dots n_N\}$, modèle prévisionnel $B(t_m, t_h)$

// Applications des forces répulsives et élastiques

pour chaque n_i **faire**

$$n'_i = n_i + F_{rep}(n_i) + F_{ela}(n_i)$$

fpour

// Restauration de la cohérence temporelle

pour chaque n'_i **faire**

$$n''_i = n'_i + F_{ct}(n'_i)$$

fpour

// Maintient de la faisabilité par gén. de traj.

pour chaque n''_i **faire**

$$\hat{n}_i = n''_i + F_{conn}(n''_i)$$

fpour

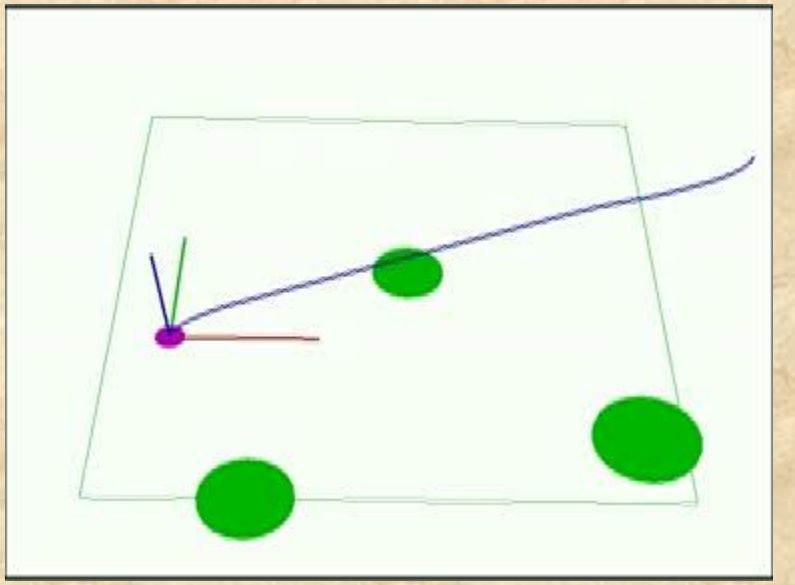
// Vérification de la validité de la trajectoire

si *non valide* $\{\hat{n}_k, \hat{n}_{k+1} \dots \hat{n}_{\hat{N}}\}$ **alors**

Appel du planificateur global

fsi

Retourner $\{\hat{n}_k, \hat{n}_{k+1} \dots \hat{n}_{\hat{N}}\}$



Déformation de trajectoire : Conclusion

- Conserve les propriétés de la déformation de chemin
  - Évitement d'obstacles
  - Convergence vers le but
- Déforme la trajectoire librement dans SxT

  + utilisation d'un modèle prévisionnel du comportement des obstacles

=> permet d'anticiper leur mouvement

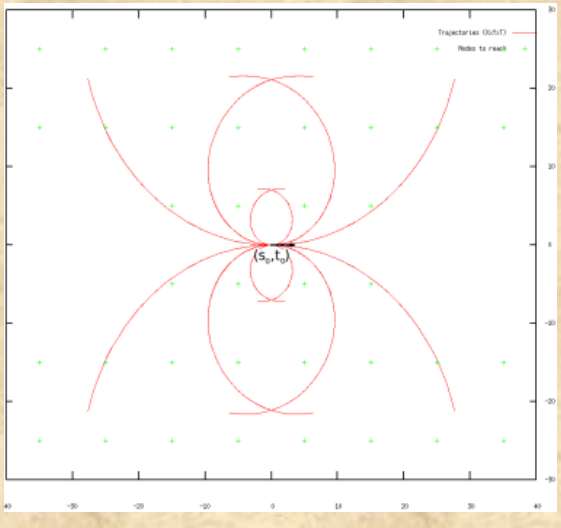
- Difficulté : Nécessité d'un générateur de trajectoire entre deux états donnés à des temps fixés

=> définition d'un nouveau générateur de trajectoires Tiji

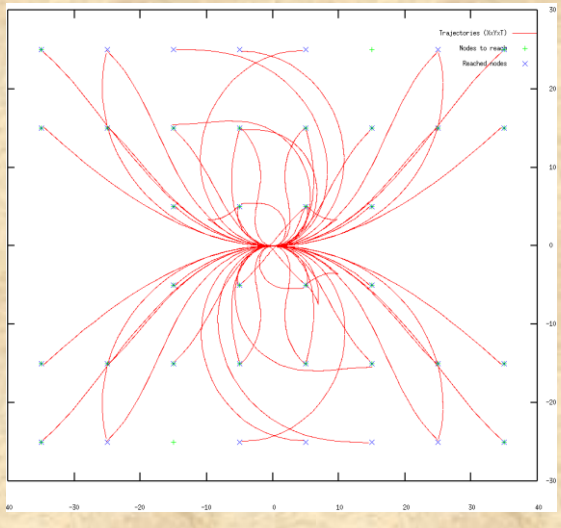
Annexes Tiji

Base de paramètres initiaux

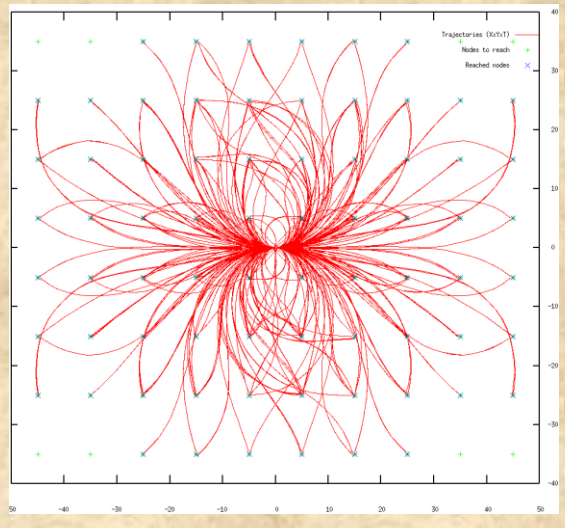
- Échantillonage régulier sur l'espace d'état-temps
- Génération d'un ensemble de trajectoires initiales à partir d'un échantillonage sur le contrôle
- Atteinte de l'ensemble des états-temps de proche en proche



Étape 1

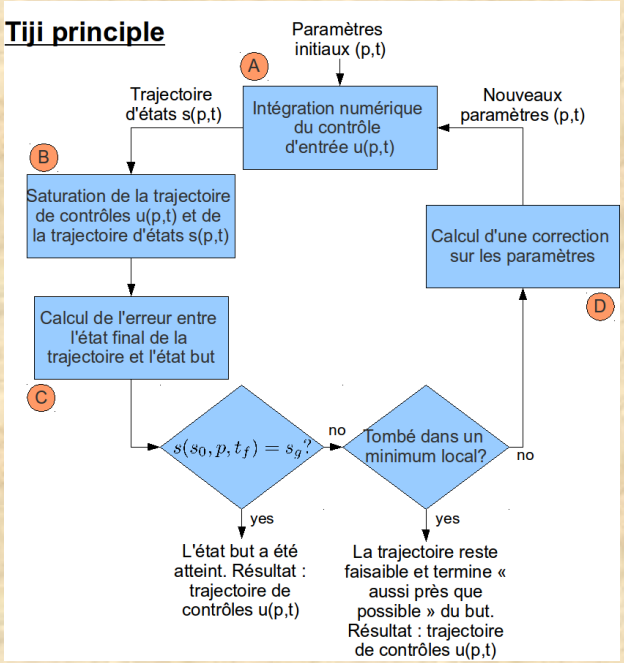


Étape 2



Étape 3

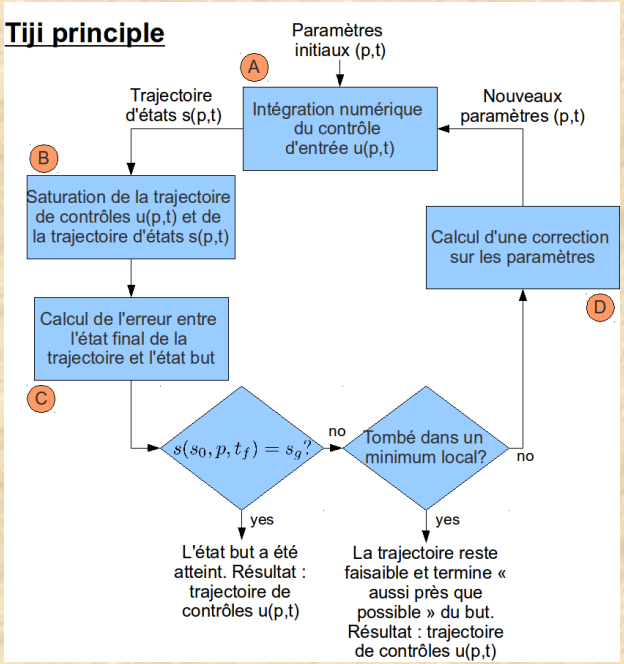
Calcul de l'erreur sur l'état final



$$\Delta s = s(s_0, p, t_f) - s_g$$

$$\Delta s < \epsilon \Rightarrow \ s_g \ atteint$$

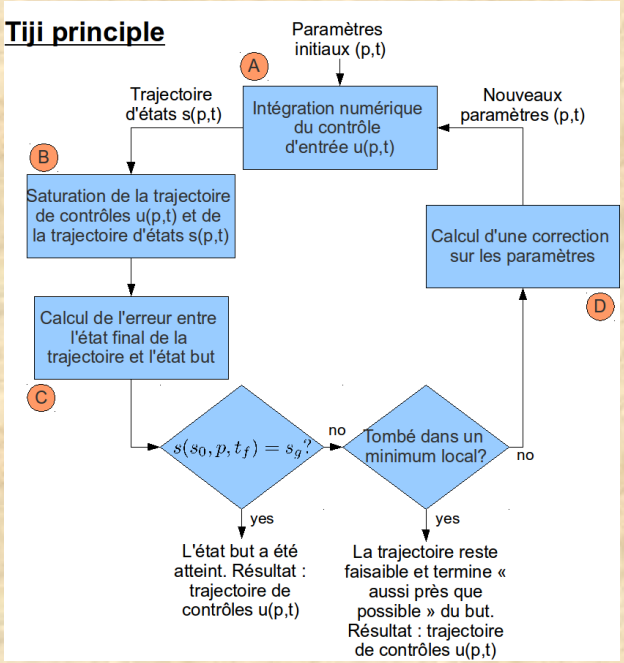
Calcul de la correction sur les paramètres



$$cor_{(p,t_f)} = -\lambda \left[\frac{\partial^2 J}{\partial (p,t_f)^2} \right]^{-1} \left[\frac{\partial J}{\partial (p,t_f)} \right]$$

=> Nouveaux paramètres

Minimum local



Pas de convergence après un nombre fixé d'itérations

⇒Minimum local

Résultat : u(p,t)

- garantie d'être admissible
- « aussi près que possible » du but

Résultats robot différentiel

- Dynamique :

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\omega}_L \\ \dot{\omega}_R \end{pmatrix} = \begin{pmatrix} v\cos(\theta) \\ v\sin(\theta) \\ \omega \\ \mu_L \\ \mu_R \end{pmatrix}$$

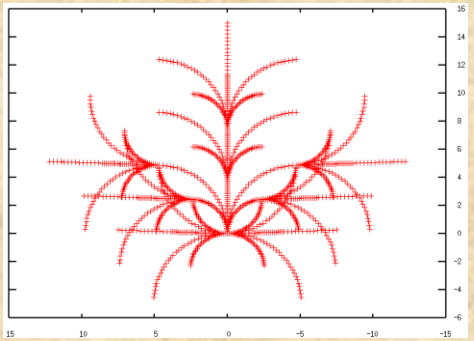
- Contraintes :

$$(\mu_L, \mu_R) \in [-\mu_{max}, \mu_{max}], \quad (\omega_L, \omega_R) \in [-\omega_{max}, \omega_{max}]$$

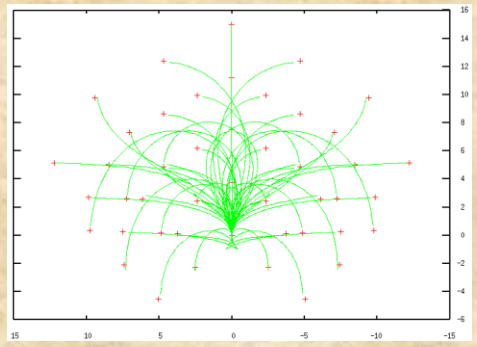
- Contrôle :

$$(\mu_L, \mu_R)$$

Cas atteignable :

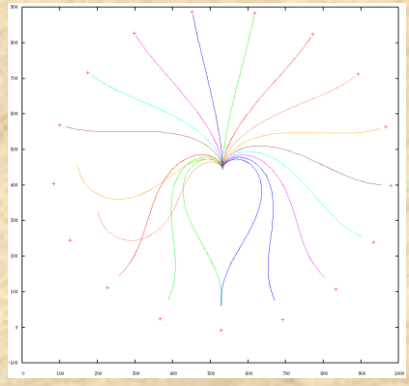


Arbre d'échantillonage

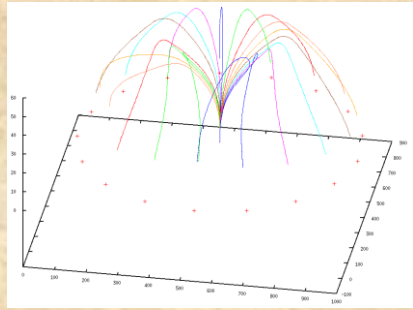


Trajectoires générées

Cas non atteignable :



XxY



XxYxV

Résultats voiture

- Dynamique :

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \\ \dot{v} \end{pmatrix} = \begin{pmatrix} v\cos(\theta) \\ v\sin(\theta) \\ v\tan(\phi)/L \\ \zeta \\ a \end{pmatrix}$$

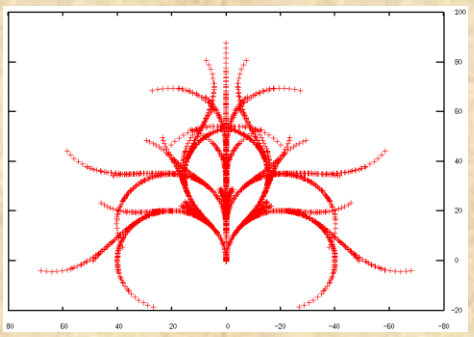
- Contraintes :

$$v \in [0, v_{max}], \quad |\phi| < \phi_{max}$$
$$|a| < a_{max}, \quad |\zeta| < \zeta_{max}$$

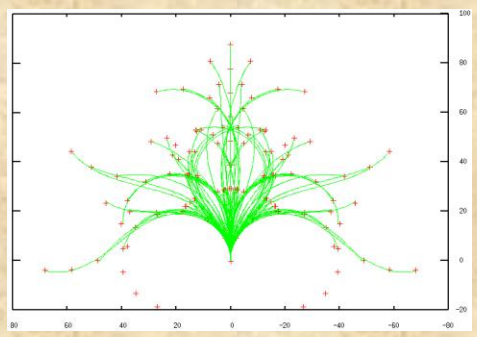
- Contrôle :

$$(a, \zeta)$$

Cas atteignable :

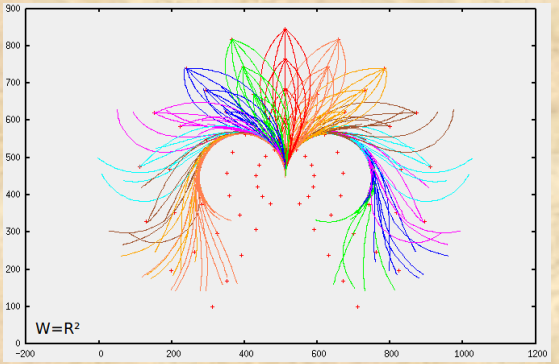


Arbre d'échantillonage



Trajectoires générées

Cas non atteignable :

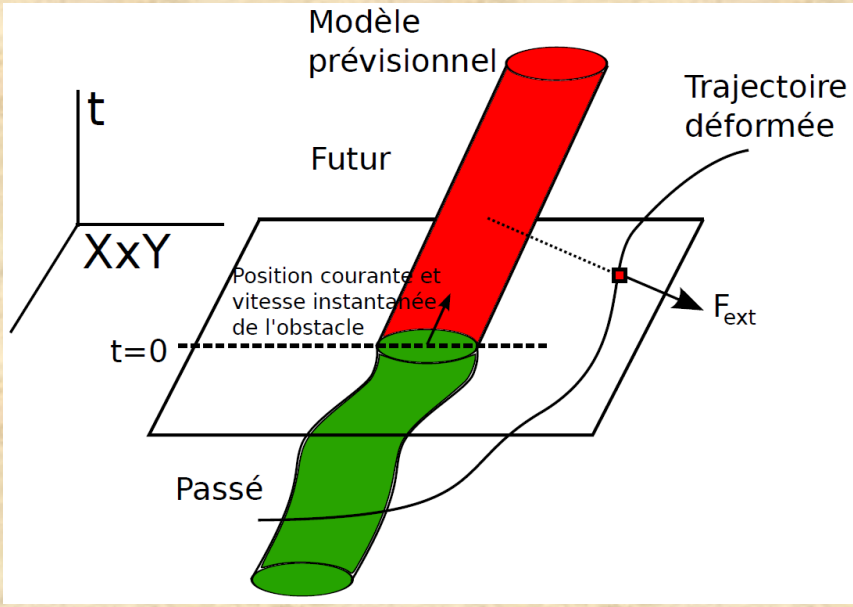


XxY

Annexes XPs

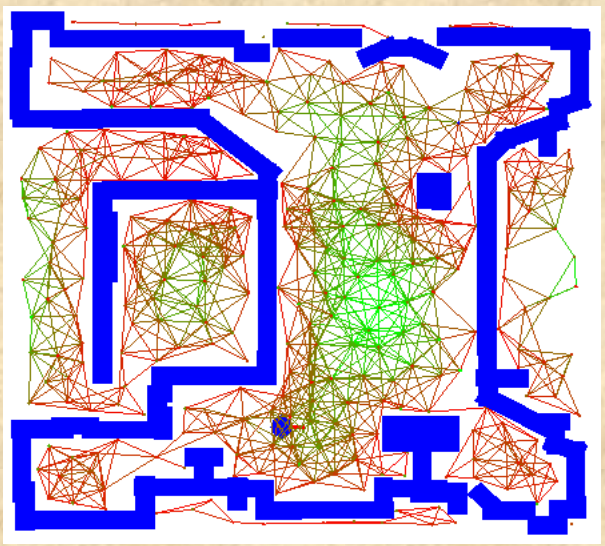
Où suis-je et qu'y a-t-il autour de moi?

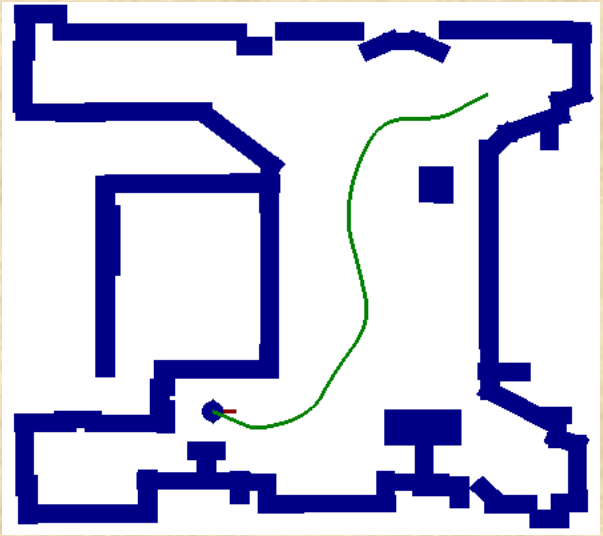
- Localisation
  => odométrie + carte
- Environnement statique
  => précalculé
- Environnement dynamique
  - Détection et suivi d'obstacles [VA09]
  - Fournit leur position et leur vitesse instantanée
  - Déduction d'un modèle prévisionnel déterministe



Comment rejoindre le but à partir de ma position courante?

- Planification à partir d'une carte de route probabiliste





- Évitement d'obstacles par déformation de trajectoire (Teddy)

Comment exécuter ce mouvement?

Suivi de trajectoire utilisant Tiji

- Calcul d'une trajectoire à partir de la localisation relevée
- Possibilité de « rattraper » la trajectoire suivie en cas de déviation