TR Théorie du contrôle - Sujet 2 : Schumacher

Paul Fraenkel

Contents

Part I
Commande en vitesses

1 Présentation du modèle

En première approximation, on considère que le robot est commandé en vitesse horizontale u et en vitesse de lacet Ψ, et en faisant cela on suppose donc qu'il y a roulement sans glissement au niveau du contact avec le sol. On modélise la dynamique du robot par les équations suivantes :

$$\begin{cases} \dot{x} &= ucos(\psi) \\ \dot{y} &= usin(\psi) \\ \dot{\psi} &= r \end{cases}$$

2 Modèle sans collisions

But : se familiariser avec le problème et les outils d'optimisation. Objectif : partir d'une position $(x_0, y_0, \psi_0, u_0, r_0)$ et aller à une position $(x_f, y_f, \psi_f, u_f, r_f)$ en minimisant le temps de déplacement T. Contraintes :

$$\begin{cases} -u_{max} \leq u \leq u_{max} \\ -r_{max} \leq r \leq r_{max} \end{cases}$$

Le problème est non-linéaire mais lisse, on peut donc utiliser un solveur numérique de points intérieurs (justifier pourquoi c'est un des plus adaptés)

2.1 Méthode d'optimisation pour un système régit par des équations différentielles

Il y a deux courants principaux pour l'optimisation d'équations différentielles : "optimize then discretize" et "discretize then optimize". On utilise ici la dernière : On discrétise l'équation différentielle selon un schéma défini (Euler explicite, Euler implicite, trapèzes, etc...), puis on utilise les égalités qui ne découlent comme contraintes de notre problème d'optimisation.

On note l'état $X = \begin{bmatrix} x \\ y \\ \psi \end{bmatrix}$ et la commande $U = \begin{bmatrix} u \\ r \end{bmatrix}$

Cas du schéma d'Euler explicite avec N points :

$$\begin{aligned} \min_{X,U} \quad & T \\ & \dot{X} = B(X) \cdot U \\ \text{s.t.} \quad & X_0 = \bar{X}_0 \\ & X_f = \bar{X}_f \end{aligned} \tag{1}$$

Avec $\delta t = \frac{T}{N-1}$ et $B(X) = B(\psi) = \begin{bmatrix} cos(\psi) & 0 \\ sin(\psi) & 0 \\ 0 & 1 \end{bmatrix}$

3 Test annexe : champ de polygones

Objectif : mettre à l'épreuve le solveur en lui demandant de traverser un espace rempli de polygones convexes.

3.1 Génération des polygones

Pour des soucis de simplicité, les polygones ont tous le même nombre de faces. Dans l'implémentation du fichier `speed_model_random_terrain.jl`, j'ai utilisé des triangles pour avoir des polygones convexes.

Ils sont générés en positionnant de manière uniforme un point central dans une zone rectangulaire délimitée. Les 3 sommets du triangle sont générés uniformément dans un carré autour du point central associé.

Dans la méthode décrite dans l'article [nacofinal.pdf -> mettre la vrai référence], un polygone $P^{(i)}$ est représenté par une matrice $C^{(i)}$ et un vecteur $d^{(i)}$ de la manière suivante :

$$P^{(i)} = \{y \in \mathbb{R}^3 \mid C^{(i)} y \leq d^{(i)}\} \tag{2}$$

Une interprétation est la suivante :
Les lignes de $C^{(i)}$ sont les normales des faces, et les coefficients de $d^{(i)}$ associés sont les distances signées respectives de l'origine à ces faces.

Exemple Soit O l'origine du repère, n la normale à la face j du polygone $P^{(i)}$ et $M^{(i,j)}$ un sommet de $P^{(i)}$ appartenant à la face j.

$$\begin{cases} C_j^{(i)} & = n^T \\ d_j^{(i)} & = OM^{(i,j)} \cdot n \end{cases}$$

Pour obtenir les normales, on considère 2 sommets (consécutifs...), a et b, dont on calcule un vecteur orthogonal :

$$n^\star = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \cdot \frac{ab}{||ab||} \tag{3}$$

Soit c un autre sommet du polygone.

$$n = \begin{cases} +n^\star & \text{si} \quad n^\star \cdot ac \leq 0 \\ -n^\star & \text{sinon} \end{cases} \tag{4}$$

3.2 Observations

Les essais ont été réalisés avec 101 points. L'algorithme de converge généralement assez lentement, sauf dans les cas où le robot peut atteindre l'objectif en ligne droite.
Le modèle est initialisé avec une solution en ligne droite entre le point de départ et le point d'arrivée. L'optimisation privilégie en général les solutions qui en sont proches, parfois au prix de tomber sur des minima locaux infaisables.

4 Test annexe : Parking

Demandons au robot d'effectuer un créneau dans une place de longueur l et de profondeur p.
Remarque : Avec la méthode utilisée pour décrire les collisions, il n'est pas obligatoire d'utiliser des polygones. Un polygone est représenté comme l'intersection de demi-espaces délimités par des droites (les faces du polygone). Il est donc possible de prendre des espaces ouverts comme "polygone".

La place de parking est modélisée par 3