16/20

MTH8408: Description de projet

Remis pour le 5 février, 2017

Professeur Dominique Orban

Très bon mais encore un peu vague. À formaliser pour le rapport intermédiaire.

André Phu-Van Nguyen, 1525972

il vous faut un co-équipier

Description de la problématique choisie

Pour ce projet nous avons choisi de reproduire la méthodologie et les résultats de l'article scientifique intitulé *Minimum Snap Trajectory Generation and Control for Quadrotors* par Daniel Mellinger et Vijay Kumar. Dans cet article, les auteurs avaient entres autres l'objectif de générer des trajectoires qui prennent avantage de la dynamique des quadricoptères plutôt que de voir la dynamique en tant que contrainte sur le système [4].

Lors du suivi d'une trajectoire, une solution triviale est souvent utilisée qui consiste en l'interpolation en ligne droite entre chaque point de cheminement aussi nommé *waypoint*. Ceci est inneficace car la courbure infinie à chaque waypoint oblige le quadricoptère à s'arrêter avant de passer au prochain waypoint. Mellinger propose donc de modéliser une trajectoire optimale par un polynôme défini par parties entièrement lisse à travers les différents waypoints tout en satisfaisant des contraintes sur les vitesses et accélérations possibles du véhicule. Ce problème est résolu en le reécrivant en problème d'optimisation quadratique.

Tout dabord Mellinger démontre que la dynamique d'un quadricoptère a la propriété dêtre différentiellement plat. C'est-à-dire que les états et les entrées peuvent être exprimées par quatre sorties plates et leurs dérivées. Nous avons donc le vecteur de sorties plates

=?

$$\sigma = [x, y, z, \psi]^T$$

où $r = [x, y, z]^T$ est la position du centre de masse dans le système de coordonnées du monde et ψ l'angle de lacet. Une trajectoire est définie comme était une courbe lisse dans l'espace des sorties plates:

Je ne suis pas familier avec ces concepts. Il me faudrait quelques définitions...

$$\sigma(t) : [t_0, t_m] \to \mathbb{R}^3 \times SO(2)$$

où t_0 et t_m sont les temps de début et de fin de la trajectoire, m correspond au nombre d'intervales de temps entre chaque waypoint et $SO(2)$ est le groupe spécial orthogonal. En pratique, une trajectoire est plutôt décrite par un polynôme défini par parties:

$$\sigma_T(t) = \begin{cases} \sum_{i=0}^{n} \sigma_{Ti1} t^i & t_0 \le t < t_1 \\ \sum_{i=0}^{n} \sigma_{Ti2} t^i & t_1 \le t < t_2 \\ \dots & \\ \sum_{i=0}^{n} \sigma_{Tim} t^i & t_{m-1} \le t < t_m \end{cases} \tag{1}$$

ok

où n est l'ordre du polynôme et m est encore le nombre d'intervales de temps.

Formulation du problème de génération de trajectoire

Au final, le but premier de la méthode de Mellinger est d'optimiser la dérivée d'ordre k_r de la position au carré et la dérivée d'ordre k_ψ de l'angle de lacet du véhicule au carré.

$$\min \int_{t_0}^{t_m} \mu_r \left\| \frac{d^{k_r} r_T}{dt^{k_r}} \right\|^2 + \mu_\psi \frac{d^{k_\psi} \psi_T}{dt^{k_\psi}}^2 \tag{2}$$

Vous avez défini r et ψ mais que sont r_T et ψ_T ?

$$\begin{array}{lll} \text{sous contraintes} & \sigma_T(t_i) = \sigma_i & i = 0, \dots, m \\ & \frac{d^p x_T}{dt^p}|_{t=t_j} = 0 \text{ ou libre}, & j = 0, \dots, m; p = 1, \dots, k_r \\ & \frac{d^p y_T}{dt^p}|_{t=t_j} = 0 \text{ ou libre}, & j = 0, \dots, m; p = 1, \dots, k_r \\ & \frac{d^p z_T}{dt^p}|_{t=t_j} = 0 \text{ ou libre}, & j = 0, \dots, m; p = 1, \dots, k_r \\ & \frac{d^p \psi_T}{dt^p}|_{t=t_j} = 0 \text{ ou libre}, & j = 0, \dots, m; p = 1, \dots, k_\psi \end{array}$$

où μ_r et μ_ψ sont des constantes qui rendent l'intégrale non dimensionelle, $\sigma_T = [x_T, y_T, z_T, \psi_T]^T$ et $\sigma_i = [x_i, y_i, z_i, \psi_i]^T$. Mellinger et Kumar choisissent $k_r = 4$ c'est-à-dire la deuxième dérivée de l'accélération (le *snap*) et $k_\psi = 2$. En d'autres mots, les contraintes expriment les waypoints à travers lesquels le véhicule doit voler et la vitesse, l'accélération, le *jerk* et le *snap* désiré à chaque waypoint. Doivent-ils tous être $= 0$?

Le problème peut être formulé en tant que problème d'optimisation quadratique en réécrivant les constantes $\sigma_{T_{ij}} = [x_{T_{ij}}, y_{T_{ij}}, z_{T_{ij}}, \psi_{T_{ij}}]$ en un vecteur c de dimension $4nm \times 1$ avec les variables de décision $\{x_{T_{ij}}, y_{T_{ij}}, z_{T_{ij}}, \psi_{T_{ij}}\}$ pour avoir la forme standard:

$$\begin{aligned} \min \quad & c^T H c + f^T c \\ \text{s. c.} \quad & Ac \leq b \end{aligned} \tag{3}$$

Quelles informations a-t-on sur la structure de H ?

Le vecteur c est donc le vecteur contenant les coefficients des polynômes définissant la trajectoire.

Mellinger démontre aussi une façon de modifier le problème d'optimisation pour ajouter des contraintes de corridor. Autrement dit, c'est une façon de forcer la trajectoire à respecter un corridor de sécurité entre deux waypoints pour éviter une collision.

Formulation du problème d'allocation de temps

Si jamais le temps d'arrivé à chaque waypoint intermédiaire importe peu, il est possible de réallouer les temps assignées entre chaque waypoint. Une fois que la solution $f(T)$ de (2) est trouvée avec un pas de temps égal entre chaque waypoint, on résous un autre problème d'optimisation

$$\min \quad f(T) \tag{4}$$

Qu'y a-t-il dans $f(T)$?

$$\begin{aligned} \text{s. c.} \quad & \sum T_i = t_m \quad & i = 1, \ldots, m \\ & T_i \geq 0 & i = 1, \ldots, m \end{aligned}$$

où $T_i = t_i - t_{i-1}$ sont les temps alloués à chaque segment de la trajectoire. L'optimisation se fait au moyen d'une descente du gradient avec un "backtracking line search". Comment ? Comme vu en classe, la méthode de la plus forte pente s'applique aux problèmes sans contraintes.

Objectifs et plan d'action

Le but général du projet est de reproduire la méthode complète de génération de trajectoire et si le temps le permet, de brancher l'implémentation dans Robot Operating System et le simulateur de physique Gazebo et faire voler une trajectoire par un multirotor en simulation. Plus précisément, les objectifs en ordre de priorité sont:

1. Modéliser le problème et résoudre (3) pour générer une trajectoire lisse préléminaire en Matlab au moyen du solveur *quadprog* en suivant les équations dans l'article
   - La difficulté ici est de formuler une méthode pour automatiquement construire la matrice hessienne H pour $m \in \mathbb{Z}$ et les matrices représentant les contraintes.
2. Résoudre le problème de réallocation des temps (4) avec une implémentation "fait maison" de la descente du gradient en Matlab Encore une fois, ce n'est pas une méthode pour les problèmes avec contraintes.

Le problème (3) est-il convexe? Quadprog s'applique-t-il aux problèmes non convexes ?

3. Ajouter les contraintes de corridor de sécurité à (3) en Matlab
4. Si le temps le permet: Transférer le code à C++ ou Python pour brancher le générateur de trajectoire à une simulation réaliste
   - Dans le cas de C++ il devrait être possible d'utiliser la librairie d'algèbre linéaire Eigen et le solveur OOQP. Je ne pense pas que OOQP fonctionne avec Eigen. OOQP ne traite que les problèmes convexes.

En ce qui concerne la dernière étape du projet, le simulateur de drones développé par Furrer et al. au moyen de Robot Operating System et du simulateur Gazebo sera utilisé [2]. Cette simulation inclut entres autres plusieurs modèles de multirotors et une implémentation d'un contrôleur de trajectoires basé sur le contrôleur gémétrique proposé par Lee et al. [3] qui sera utilisé au lieu du contrôleur décrit dans l'article et Mellinger.

Je dispose d'un solveur pour (3) en Python mais il faudrait savoir si le problème est convexe ou non.

Impact attendu

Le laboratoire de robotique mobile et des systèmes autonomes est souvent appelé à faire des présentations grand public à fin de représenter Polytechnique Montréal lors des évènements portes ouvertes ou lors des visites de partenaires industriels. Dans l'idéal, la complétion de ce projet permettrait de générer de belles trajectoires (lisses) pouvant être volées par l'un des véhicules multirotors du laboratoire lors de ces présentations.

Bien qu'il existe maintenant des méthodes plus performantes et plus complexes pour la génération de trajectoires par optimisation tel que [5] et [1], elles reposent toujours sur une forme d'optimisation quadratique similaire à celle de Mellinger dans [4]. L'implémentation de cette méthode est donc un moyen idéal d'étudier la formulation la plus simple d'un problème d'optimisation quadratique dans le contexte de la génération de trajectoires pour multirotors dans le but d'aussi comprendre les méthodes modernes.

Semble très intéressant et important. Bon travail !

Références

[1] J. Chen, T. Liu, and S. Shen. Online generation of collision-free trajectories for quadrotor flight in unknown cluttered environments. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1476–1483, May 2016.

[2] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart. *Robot Operating System (ROS): The Complete Reference (Volume 1)*, chapter RotorS—A Modular Gazebo MAV Simulator Framework, pages 595–625. Springer International Publishing, Cham, 2016.

[3] T. Lee, M. Leoky, and N. H. McClamroch. Geometric tracking control of a quadrotor uav on se(3). In *49th IEEE Conference on Decision and Control (CDC)*, pages 5420–5425, Dec 2010.

[4] D. Mellinger and V. Kumar. Minimum snap trajectory generation and control for quadrotors. In *2011 IEEE International Conference on Robotics and Automation*, pages 2520–2525, May 2011.

[5] C. Richter, A. Bry, and N. Roy. *Polynomial Trajectory Planning for Aggressive Quadrotor Flight in Dense Indoor Environments*, pages 649–666. Springer International Publishing, Cham, 2016.