Rapport d'article de robotique avancée

# A Kinodynamic steering-method for legged multi-contact locomotion

Pierre Fernbach, Steve Tonneau, Andrea Del Prete, Michel Taïx Soumis le 26 Juillet 2017

S9 - 2017/2018

**Yoann Fleytoux**
**Chris IBRAHIM**
Présenté le 15 Janvier 2018

**À l'attention de F. Ingrand, S. Tonneau, M. Taïx**

# Table des matières

# Définition du problème

La planification de mouvement en robotique, consiste à décomposer un mouvement pour une tâche désirée, en plusieurs mouvements réalisables par le robot et satisfaisant diverses contraintes. Calculer cette trajectoire revient donc à trouver une succession de configurations du robot permettant à un robot de se déplacer d'un point de départ à un point d'arrivée.

En robotique et en planification de mouvement, le terme *kynodynamic planning* désigne une classe de problèmes de planification de mouvement qui consiste à satisfaire des contraintes, non seulement sur la cinématique du système, mais aussi sur sa dynamique, c'est-à-dire les vitesses et les accélérations. [1]

# Solution mise en place

Le prototype présenté dans ce travail réalisé par Pierre Fernbach, Steve Tonneau, Andrea Del Prete et Michel Taïx, est basé sur une formulation efficiente de la dynamique des robots à pattes. Leur contribution est directement applicable sur des méthodes liées à des déplacements multi-contacts.

La méthode consiste à utiliser un DIMT ou Double Minimum Integration Time à partir de bornes symétriques d'accélération et de vitesses définies par l'utilisateur pour calculer une trajectoire reliant deux états consécutifs, sans regarder les contraintes de collision. Le principal problème est que, pour un robot à pattes, ces contraintes ne sont ni symétriques, ni constantes au cours du déplacement.

Pour remédier à cela, une phase de validation de trajectoire est réalisée pour vérifier l'équilibre dynamique et les contraintes de non-collision tout le long de la trajectoire. La trajectoire retournée correspond à la portion de trajectoire qui satisfait toutes les contraintes.

De plus, le DIMT est également utilisé en amont pour calculer des bornes réalistes sur les accélérations. Cela est essentiel dans la mesure où elle permet d'augmenter significativement le taux de succès de la *steering method*. Le calcul retourne une valeur d'accélération et une direction d'accélération pour chaque phase du mouvement. En appliquant un deuxième DIMT et utilisant l'accélération trouvée en tant que nouvelles bornes, on trouve une nouvelle accélération dont on vérifiera la colinéarité (même direction) avec la précédente. Si elle est colinéaire à la précédente, les bornes sont exactes et la trajectoire est dynamiquement valide dans le voisinage de l'état de départ.

# Résultats: application à la planification multi-contact

## Avantages

Leur méthode a été intégré au sein du planificateur **RB-RRT**, qui découple le problème de planification en trois phases (P1, P2, P3) [2][3], afin de générer des mouvements dynamiques multi-contact, puis elle a été testée dans différents scénarios [4].

Leur méthode permet de transformer la partie planification de chemin du planificateur, en problème de planification de trajectoire, ce qui supprime les contraintes imposées par les configurations en contact pour être en équilibre statique. Cela permet de traiter différents problèmes nouveaux tels des scénarios

incluant des sauts, des descentes de pentes raides, et retrouver l'équilibre après une poussée. Chaque scénario met en évidence une propriété spécifique du planificateur.

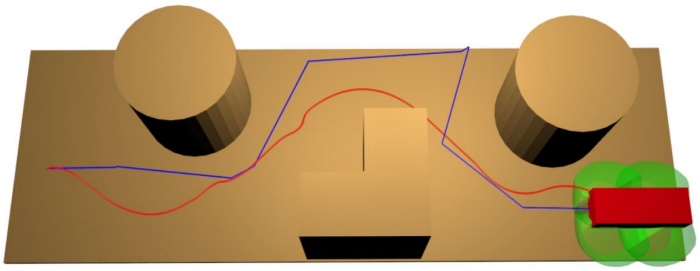

**Fig. 1 :** Les trajectoires calculés par la version kynodynamique du planificateur, en rouge, est visuellement plus intéressant que sa version quasi-statique, en bleue

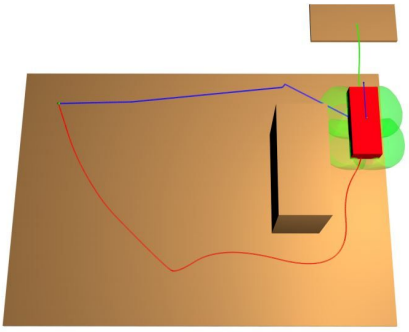

**Fig.6 :** Trajectoire de la méthode quasi-statique en bleue et celle de leur planificateur en rouge

D'après la figure 6, la trajectoire bleue obtenue par la méthode quasi-statique, bien qu'elle soit le chemin le plus court, ne permet pas d'atteindre la vitesse requise pour effectuer le saut qui suit, contrairement à la trajectoire obtenue avec leur planificateur, en rouge. Un planificateur géométrique ou une approche locale ne serait pas capable d'échapper à la trajectoire bleue, contrairement à la méthode de leur planificateur.

Leur méthode est capable de calculer des mouvements fortement dynamiques, tout en échappant aux minima locaux, dans des cas où cela n'a jamais été réalisé dans la littérature. Des méthodes existantes [5],[6] pourraient s'en servir pour effectuer des initialisations efficaces , ce qui leur permettrait de traiter des environnements encombrés et converger plus vite.

Ils ont développé un critère permettant de vérifier efficacement l'équilibre dynamique du robot ainsi que l'accélération maximum de son **COM**.

Leur méthode, dans le pire des cas, est aussi efficace que les approches dynamiques existantes, et est, dans le meilleur des cas, bien plus performantes.

| | Path Planning | | | | Contact Generation | | Total |
|---|---|---|---|---|---|---|---|
| Scenario | Valid traj. (%) | Valid dir. (%) | num. nodes | time (% tot. time) | time (% tot. time) | success (%) | time (s) |
| Slalom (HyQ) | 95.72 % | 65.6 % | 491 | 69.56 % | 30.43 % | 90 % | 85.55 |
| Local minima (HyQ) | 95.25 % | 69.8 % | 315 | 69.14 % | 30.86% | 75% | 38.31 |
| Steep slope (25°) (HRP-2) | 82.4 % | 58.3 % | 3895 | 77.36 % | 22.63% | 85 % | 165.61 |
| Steep slope (38°) (HRP-2) | 80.75 % | 56.5 % | 5533 | 87.35 % | 12.64% | 90 % | 679.73 |
| Push recovery (HRP-2) | 100 % | 100 % | 2 | 0.08 % | 99.92% | 100 % | 5.78 |
| Inclined planes (HyQ) | 100 % | 100 % | 2 | 0.20 % | 99.80% | 100 % | 2.55 |

**Tableau I :** Temps de calculs et taux de succès par scénarios moyennés sur 20 essais

En termes de performances temporelles, la tableau I montre que, selon les scénarios, des variations de quelques secondes jusqu'à quelques minutes peuvent se produire. Ces durées s'avèrent comparables à celles que l'on retrouve dans la littérature, à la différence que les scénarios ici sont plus longs et plus contraints.

| Scenario : Slalom (HyQ) | | | |
|---|---|---|---|
| acc. bounds | time (s) | valid trajectories | solution length (s) |
| LP (ours) | 59.5 | 95.7% | 12.2 |
| 6 | time-out | 0.8% | X |
| 4 | 913.3 | 69.1% | 13.4 |
| 2 | 109.7 | 90.3 % | 22.3 |
| Scenario : Slope (38°) (HRP-2) | | | |
| acc. bounds | time (s) | valid trajectories | solution length (s) |
| LP (ours) | 593.7 | 80.75 % | 11.64 |
| 6 | time-out | 0.2% | X |
| 4 | 10992.6 | 7.6% | 16.98 |
| 2 | time-out | 8.8% | X |

**Tableau II** : Comparaison des performances de planification obtenue avec des bornes sur l'accélération définies par l'utilisateur

Le tableau II montre l'intérêt de leur steering method en mettant en évidence l'efficacité de leur approche par rapport à la méthode DIMT traditionnelle avec des bornes définies par l'utilisateur. Leur trajectoires sont plus courtes et sont plus rapidement calculées grâce à leur meilleur taux de rejet. De plus, le calcul automatique de seuils permet de ne pas avoir régler minutieusement les paramètres.

## Inconvénients

L'intégration de la *steering method* dans leur "*reachability planner*" a requis l'introduction de certaines heuristiques, inhérents à son architecture : les contacts sont considérés sans glissement et toujours actifs, ce qui n'est pas vrai dans certains cas.

De plus, la sortie désirée de la première phase $P_1$ du **RB-RRT** n'est pas la trajectoire racine finale mais plutôt une très bonne estimation initiale. En effet, les approximations ne sont correctes que lorsque les contacts sont générés, et à la phase d'interpolation ($P_2$ et $P_3$).

Leur planificateur, bien qu'il ne tombe pas dans des minima locaux, l'algorithme **RRT** ne garantie pas une solution optimale globale.

# Conclusion

Finalement, l'intérêt de la *steering method* proposée dans ce papier est mis en évidence grâce à son intégration dans le **RB-RRT**, leur planificateur multi-contact discret, et à son utilisation dans des scénarios jusqu'ici impossible à traiter avec des planificateurs existants.

Des travaux futurs se focaliseront sur des méthodes d'optimisation de trajectoire pour leur planificateur ou utiliser un planificateur asymptotiquement optimal tel que **RRT** [7]. Les temps de calculs respectant les contraintes temps-réel constituent un objectif ultérieur. Et un autre point qu'ils souhaiteraient traiter est de proposer un algorithme direct pour les trajectoires calculées par leur planificateur.

Néanmoins, un point sombre dans l'article réside dans les performances réelles des autres méthodes où les tableaux de comparaisons ne les mentionnent pas, étant donné que les autres articles ne parlent pas de leurs ressources de calculs.

# Réference

[1] Donald, B.; Xavier, P.; Canny, J.; Reif, J. (1993), "Kinodynamic motion planning"

[2] S. Tonneau, N. Mansard, C. Park, D. Manocha, F. Multon, and J. Pettre, “A reachability-based planner for sequences of acyclic ´ contacts in cluttered environments,” in Int. Symp. Robotics Research (ISRR), Sestri Levante, Italy, 2015.

[3] S. Tonneau, A. D. Prete, J. Pettre, C. Park, D. Manocha, and N. Mansard, “An efficient acyclic contact planner for multiped robots,” in Technical report, 2016.

[4] Video : kinodynamic steering method for legged multicontact locomotion
https://www.researchgate.net/publication/318702205_Video_kinodynamic_steering_method_for_legged_multicontact_locomotion

[5] I. Mordatch, E. Todorov, and Z. Popovic, “Discovery of complex ´ behaviors through contact-invariant optimization,” ACM Trans. on Graphics, vol. 31, no. 4, pp. 43:1–43:8, 2012.

[6] R. Deits and R. Tedrake, “Footstep planning on uneven terrain with mixed-integer convex optimization,” in IEEE-RAS Inter. Conf. on Humanoid Robots (Humanoids), Madrid, Spain, 2014.

[7] S. Karaman and E. Frazzoli, “Sampling-based Algorithms for Optimal Motion Planning,” Int. Journal of Robotics Research, vol. 30, pp. 846–894, 2011.

Intérêts de l'article (AKA solution mise en oeuvre)

Commencons par rappeler qu'ici nous entendons par équilibre dynamique, la contrainte sans glissement en accélération non nulle.

Le problème de planification de mouvements est ici considéré comme la recherche d'une trajectoire sans collision dynamique réalisable notée S(t) entre deux état $S_{goal}$ et $S_{init}$. Le problème est abordé en utilisant un algorithme RRT où sont ajoutés des méthodes de steering ou *steering methods* en anglais et de validation de trajectoires afin de respecter la dynamique du robot.

La steering méthode permet de générer des trajectoires réalisables au voisinage de l'état initial, tandis que celle de validation de trajectoire permet déterminer précisément la portion de trajectoire effectivement réalisable afin que tous les états du graphe généré soit connecté par des trajectoires sans collision dynamiquement réalisable.

Un état se définit ainsi, $\mathbf{S} = < \mathbf{X,P,N,Q} >$ où :

- $\mathbf{X} = < c,c,c >$ représente la position, la vitesse et l'accélération du **COM**
- $\mathbf{P} = [p_1,..,p_k]$ les positions des k points de contact
- $\mathbf{N} = [n_1,..,n_k]$ les normales respectives à ces points de contacts.
- **q** est la configuration du robot décrit par les n degrés de liberté et la position de base

Et notons que chaque état peut avoir un nombre différents k de contacts et que toutes les positions sont exprimées dans le repère monde selon les 3 axes **x,y,z**.

A chaque étape de l'algorithme de planification, la *steering method*, qui est une approche heuristique permettant de connecter deux état par une trajectoire réalisable, est réalisée. Les entrées de cette *steering method* sont des états initial et final $\mathbf{S_0} = < \mathbf{X_0,P_0,N_0,Q_0} >$ et $\mathbf{S_1} = < \mathbf{X_1,P_1,N_1,Q_1} >$ et retourne trajectoire optimale en temps de durée $t_f$ *:*

$$\mathbf{X} : t \in [0,t_f] \rightarrow < c(t),c(t),c(t) >$$

Tel que $\mathbf{X}(0) = \mathbf{X}_0$ et $\mathbf{X}(t_f) = \mathbf{X}_1$ et $\mathbf{X}(t)$ est contraint à satisfaire les seuils d'accélération du **COM** imposées par la contrainte sans glissement en $\mathbf{S_o}$.

La trajectoire est ensuite validée selon plusieurs contraintes dont, ici, la non-collision, des bornes sur la position définie par l'utilisateur, et l'équilibre statique. Les bornes supplémentaires sur la vitesse et l'accélération du **COM** qui peuvent être définies par l'utilisateur ont une garantie d'être respectées par la *steering method* et ne requiert donc pas de vérification.

$$\mathbf{X'} : t \in [0,t'_f \leq t_f] \rightarrow \mathbf{X}(t)$$

Et s'il n'y a aucune portion de trajectoire valide, $t'_f = 0$ et la planificateur rejette la trajectoire et commence une nouvelle itération.

Intérêts de l'article (AKA solution mise en oeuvre)

Le but du papier est de présenter une nouvelle méthode de synthèse de comportement de déplacement dynamique et sans collision tels que des sauts, des descentes de pentes raides et se rétablir d'une poussé effectuée par un bras du robot.

Le travail réalisé concerne notamment une nouvelle *steering method* de faible dimensionnalité, très efficient en temps de calculs et comprenant des applications dans de nombreux problèmes liés aux locomotions à pattes ou *legged locomotion* en anglais. Ainsi, en l'intégrant au sein d'un planificateur de contact discret, ils ont proposé le premier planificateur de contacte pour robots à pattes.

Ils présentent ici deux contributions théoriques et une concrète:

- Une extension du test d'équilibre statique proposé pour des cas dynamiques plus véloces que des approches précédentes dans notre cas.
- Un Linear Program (LP) efficient permettant de déterminer les intervalles d'accélération sur le **Centre de Masse** (**COM**) du robot sachant les forces de contact et une direction d'accélération désirée
- L'un des premiers planificateurs kynodynamiques capable de synthétiser des mouvement multi-contacts qui sont réellement dynamiques pour les robots à pattes.

Gaited motion
Path planning
Multi contact locomotion
Linear problem
Local planning methods
Kinodynamic planning
Steering method: heuristic approach to try to connect two states with a feasible trajectory
Trajectory: “
Dynamic Equilibrium
Sampling-based approaches
Configuration
Centroidal dynamics of the robot
Dynamic constraints
RRT connect algorithm
Motion planning algorithm
Double integrator for minimum time (dimt)
Jerk constraint
RB-RRT planner
root-path
Cinématique
Dynamique
planificateur quasi-statique

https://www.youtube.com/watch?v=EtH5I5xlyU8
Résoudre un prolème de motion planning for multi contact locomotion
Kynodynamic planning
(steering methode  generates trajectory between states)
(trajectory validation)
2LP pour les contraintes dynamic du robot à pattes

Etat de lart

Multi contact planning:
Calculer la trajectoire
Prévoir une séquence de configuration en équilibre pour la trajectoire
Si on calcul la trajectoire comme un LP problème risque de minimum local
Sampling based method n’ont pas se probleme, mais risque de trop de calcul
Les autres méthode 15 ont besoint de point de contact en équilibre statique, ce qui rendez des movement dynamics impossible

Kynodynamic planning
Besoin d’avoir la vitesse et l’accélération dans le systeme
Besoin de gérer les bornes d’accélérations variable en fonction des points de contacts et de la position du com alors que les planner existant demande que ces contraintes soient constantes
On réduitt la dimention du problème en ce concentrant sur la centroidal dynamics du robot, avec une steering method qui prend en compte les bornes variables d’accélérations

Computing dynamic constraints:
Résolu avec des LP

Contribution
2 theorique 1 pratique
Dynamic static equilibrium test (vérifie l'etat d'équilibre statique du robot dans des cas non coplanaire et dans des cas dynamiques)
LP pour déterminer les bornes d'accélération du COM en fonction des points de contacts et de la direction d'accélération désirée
Un des premier kinodyamic planner pour synthetiser dynamic multi contact motions pour robbot à pattes

Résoudre le motion planning problem
On choppe la trajectoire avec un RRT-connect algo, en respectant la dynamic du robot avec les nouveaux steering et path validation

La steering methode: genere une trajectoire réalisable, dans le voisinage de l'état initial
La path validation methode détermine quel parties du graphes sont réalisable par le robot

Steering method:
Dérivé du dimt -> boîte noire
Given user define symetrics bounds on the com
Output a minimum time trajectory that connects x0 and X1 without considering avoidance

Problem: for legged robot com acceleration bounds neither constant nor symmetric
Solution: …

Trajectory validation
Inputs: 2 etats et une trajectoire les reliant
Un set de configuration de point de contacts le long de cette trajectoires
Output : une nouvelle trajectoire qui respect les contraintes dynamiques, et sans collisions

Computing acceleration bounds for the steering method
Plutot que de calculer avec une double description method
Ils utilisent une method de LP