

Motion planning problem : création de séquences de points

Florence Clerc

February 8, 2011

Ce document est basé sur l'article "On the relationship between classical grid search and probabilistic roadmaps" de S.M.Lavalle, M.S.Branicky et S.R.Lindemann et étudie les différentes réponses qui peuvent être apportées au problème de création de feuilles de route.

On rappelle brièvement la définition du problème de création de feuille de route : étant donné un espace $\mathcal{C} = [0; 1]^d$ dont l'espace \mathcal{C}_{free} est libre d'obstacles et deux positions q_{init} et q_f , existe-t-il un chemin continu allant de q_{init} à q_f dans \mathcal{C}_{free} ?

La solution aujourd'hui employée est basée sur l'utilisation de séquences aléatoires de points. Cette méthode a toutefois un inconvénient intrinsèque : par définition, on ne connaît par avance rien sur les relations entre points et construire une telle séquence est difficile. Cet article questionne donc l'utilité de l'aléatoire dans la recherche de feuille de route et cherche des alternatives à cette méthode.

Quelques critères à prendre en compte

Ensemble fini ou séquence infinie

Dans le cas d'un ensemble fini, on connaît avant de construire le graphe le nombre de points qui seront testés. Dans le cas d'une séquence infinie, les points sont ajoutés un par un.

On peut toutefois définir des séquences d'ensemble qui sont utiles pour étendre des notions définies pour les séquences. Remarquons toutefois que dans une telle séquence d'ensembles (P_N) les ensembles P_N et P_{N+1} peuvent être disjoints.

Single ou Multi-query

Certaines cartes de route peuvent être construites dans l'optique de ne les interroger qu'une seule fois, alors que d'autres peuvent être interrogées plusieurs fois. La phase de précompilation est donc légèrement changée.

Deux outils mathématiques

La divergence est définie normalement pour les séquences mais peut être étendue aux ensembles comme précédemment décrit. La définition mathématique est $D(P, \mathcal{R}_{aar}) = \sup_{R \in \mathcal{R}_{aar}} \left| \frac{|P \cap R|}{N} - \mu(R) \right|$ où \mathcal{R}_{aar} est l'ensemble des sous-ensembles de la forme $[a_i; b_i]^d$ avec d la dimension et a_i, b_i dans $[0; 1]$ et $a_i < b_i$. La divergence mesure l'erreur que l'on fait en prenant une distribution plutôt qu'une autre.

On définit également la dispersion : il s'agit du diamètre de la plus grande sphère qu'on puisse placer dans la séquence ou l'ensemble de points sans qu'elle en contienne un seul. La définition mathématique est : $\delta(P, \rho) = \sup_{x \in X} \min_{p \in P} \rho(x, p)$.

On cherche à obtenir une faible dispersion qui assure qu'on couvre correctement tout \mathcal{C} soit en $O(N^{-1/d})$ pour les séquences et les ensembles. On vise aussi une faible divergence qui est en $O\left(\frac{\log^d N}{N}\right)$ pour les séquences et en $O\left(\frac{\log^{d-1} N}{N}\right)$ pour les ensembles.

Les méthodes probabilistes

la réponse probabiliste : PRM (probabilistic roadmap) Dans un premier temps, on construit un graphe qui constitue la feuille de route et qu'on pourra interroger dans un deuxième temps. Ce principe est général à toutes les méthodes développées dans la suite.

On commence par tirer au hasard un certain nombre de points dans \mathcal{C}_{free} qui vont constituer les nœuds du graphe qui constitueront la feuille de route. Pour choisir quelles arrêtes sont créées à partir d'un nœud M , on choisit un voisinage de M . Deux méthodes sont possibles :

- on peut choisir les n points les plus proches de M (n est fixé pour tous les nœuds)
- on peut choisir les points situés à une distance inférieure ou égale à r (r est fixé pour tous les nœuds)

Pour chacune des arrêtes, avant de la créer, on vérifie qu'elle n'est pas interdite en vérifiant un certain nombre de points sur la future arrête. Tout ceci constitue la phase de précompilation. Comme on s'en rend compte, elle est longue pour cette méthode.

Au moment d'interroger cette carte, on introduit q_{init} et q_f dans la carte et on suit par exemple l'algorithme A^* pour s'orienter dans le graphe. Cette phase est donc très simple.

une adaptation : lazy PRM La différence par rapport à l'algorithme précédent est que le gros du travail se fait dans la partie de recherche dans le graphe. La partie de précompilation se borne à choisir des points dans l'espace \mathcal{C}_{free} de manière aléatoire, puis à les relier par une des deux méthodes définies précédemment. C'est dans la partie recherche que va s'effectuer la recherche de collision : un premier itinéraire est choisi, et la recherche de collision l'amène à évoluer. La deuxième phase est donc beaucoup plus lourde que dans le cas du PRM.

Avantages et inconvénients Un très gros inconvénient est lié au fait que la structure nous est totalement inconnue par avance, par conséquent, la recherche de voisinage est quelque chose de très long (85% du temps de compilation dans le cas lazy PRM est consacré à la phase de précompilation). Toujours lié à la construction de la séquence, la dispersion et la divergence ne sont pas optimales.

Un autre extrême : les grilles

Les grilles Le principe est de créer une grille, on vérifie ensuite que tous les nœuds sont dans \mathcal{C}_{free} et on supprime ceux qui ne sont pas dedans. Comme on connaît la grille par avance, il y est très simple de définir un voisinage.

Cette méthode présente toutefois un gros problème : si on fixe une résolution cherchée, on fixe en même temps un nombre de points par axe N . Si la dimension est d , le nombre total de points est N^d . L'augmentation du nombre de points est donc exponentielle en la dimension.

Un ajustement très simple permet dans un premier temps de diminuer sensiblement le nombre de points de la grille. Au lieu de choisir toutes les arrêtes de la grille qui sont reliées à des nœuds dans \mathcal{C}_{free} , on vérifie ces arrêtes comme dans le cas probabilistes pour vérifier si elles intersectent un obstacle ou non. Ainsi beaucoup moins de points sont nécessaires.

Grilles de Sukharev Ce qu'on a à y gagner est plus sensible en forte dimension et en faible résolution. L'idée est qu'en mettant la grille en (0,0), on "dépense" des points qu'on pourrait garder pour accroître la résolution. Si par exemple en dimension 1 on souhaite obtenir une résolution 0.5, pour une grille il faut trois points : à 0, à 0.5 et à 1. Dans le cas d'une grille de Sukharev, il suffit de 2 points : à 0.25 et à 0.75.

Avantages et inconvénients Le premier avantage qui apparaît immédiatement en comparaison avec l'algorithme PRM, c'est qu'on connaît le voisinage. De plus, la dispersion est optimale. Malheureusement, la divergence ne l'est pas. De plus, il s'agit d'un ensemble auquel il est particulièrement compliqué de rajouter un point.

D'autres méthodes possibles

Halton et Hammersley

L'idée de Van der Corput Il s'agit d'écrire les nombres décimaux entre 0 et 1 ($\frac{1}{2}, \frac{1}{4}, \frac{3}{4}, \frac{1}{8}, \frac{5}{8}, \frac{3}{8}$ etc) en binaire puis d'inverser l'écriture. Le bit de poids fort devient ainsi le bit de poids faible.

Pour un nombre quelconque i et un nombre premier p , on donne la décomposition de i dans la base p : $i = a_0 + a_1 \times p + a_2 \times p^2 + \dots$. On définit alors $r_p(i) = \frac{a_0}{p} + \frac{a_1}{p^2} + \frac{a_2}{p^3} + \dots$ qui correspond à l'inversion des "bits". L'idée est d'étendre l'algorithme de Van der Corput aux nombres premiers.

Séquence de Halton On choisit d nombres premiers p_1, p_2, \dots, p_d où d est la dimension de \mathcal{C} . Le i^e élément de la séquence est défini comme $(r_{p_1}(i), r_{p_2}(i), \dots, r_{p_d}(i))$.

Ensemble de Hammersley Pour faire un ensemble de N éléments, on choisit $d-1$ nombres premiers : p_1, p_2, \dots, p_{d-1} . Le i^e élément de la séquence est défini comme $\left(\frac{i}{N}, r_{p_1}(i), r_{p_2}(i), \dots, r_{p_{d-1}}(i)\right)$.

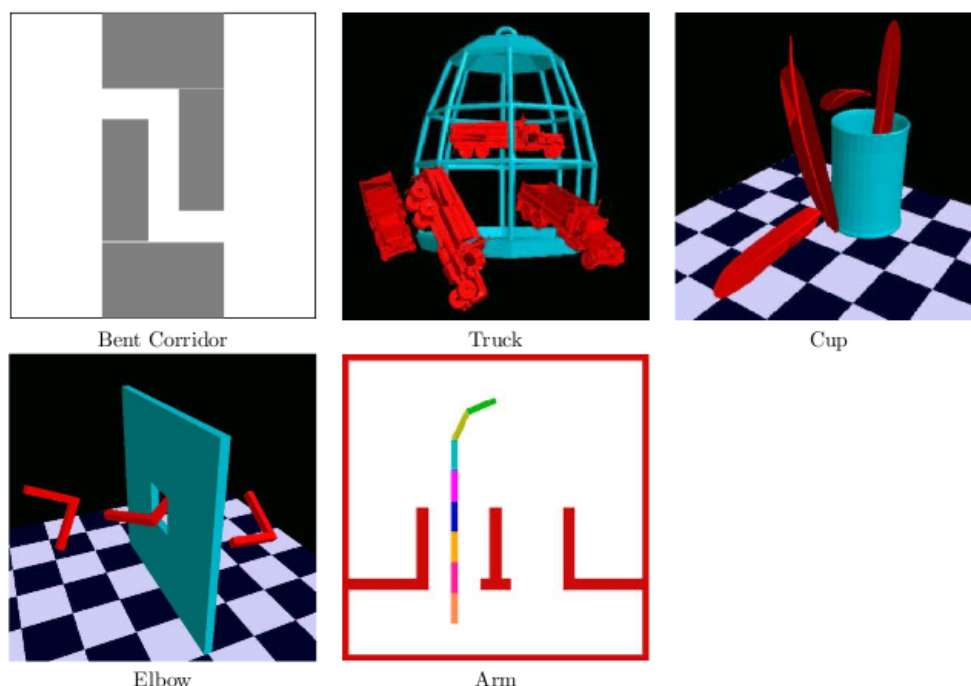
Avantages et inconvénients La divergence et la dispersion sont optimales (malgré une constante élevée). Par contre, le nombre de points nécessaires augmente super-exponentiellement en la dimension.

Les réseaux (lattice)

On pourrait comparer cet ensemble à une grille dont les axes ne seraient pas orthogonaux entre eux. En dimension d , on choisit $d-1$ réels $\alpha_1, \dots, \alpha_{d-1}$. Pour obtenir un ensemble de N points, on définit le i^e élément comme $\left(\frac{i}{N}, \{i \times \alpha_1\}, \dots, \{i \times \alpha_{d-1}\}\right)$ où $x \equiv \{x\}[1]$ et $\{x\} \in [0, 1]$.

Les réseaux ont une dispersion optimale et une divergence quasi-optimale.

Les résultats obtenus



Détail des expériences La première expérience est de passer à travers un couloir coudé en le moins de points possible, on a fait varier le diamètre et la longueur du couloir et la dimension. En faible dimension, on a également vérifié les résultats en tournant les ensembles ou séquences par rapport à \mathcal{C} .

La deuxième expérience consiste à sortir un camion d'une cage en le moins de points possible. Dans cette expérience et la suivante, on teste également les deux méthodes de connexion mentionnées.

La troisième expérience consiste à mettre une plume dans une tasse en le moins de points possible.

La quatrième expérience consiste à faire passer une forme en L à travers le trou d'un mur. Le mur peut bouger entre les expériences parallèlement au plan dans lequel il est. Le trou peut bouger dans le mur entre deux mesures.

La cinquième expérience consiste en un robot articulé dans le plan.

Sur ces cinq expériences ont été testées les grilles de Sukharev, la séquence de Halton, l'ensemble de Hammersley et le PRM.

Résultats Les grilles de Hammersley et de Halton sont dans tous les cas meilleurs que les grilles de Sukharev et que les méthodes aléatoires. PRM est plus efficace que la grille de Sukharev pour le camion et la forme en L. Il faut noter qu'un des gros inconvénients des grilles de Sukharev est qu'en augmentant la résolution d'un point par axe, on augmente considérablement le nombre de points, ce qui permet à PRM d'être meilleur que les grilles de Sukharev dans le cas du camion.

Les méthodes déterministes atteignent toutefois leurs limites en forte dimension (10 dans le cas du couloir) où PRM redevient efficace, mais où la grille Sukharev perd totalement l'avantage.

Les résultats obtenus en changeant les méthodes de connexion ne changent pas fondamentalement : on a tout de même besoin du même ordre de grandeur de points pour résoudre le problème. Ceci n'est plus vrai pour les grilles de Sukharev dans le cas de la tasse où l'avantage est clairement en faveur de la méthode de connexion par nombre de voisins.

Il faut noter que pour les grilles de Sukharev, le premier nombre de points tel que le problème soit résolu n'est pas forcément celui à partir duquel le problème est toujours résolu. Dans le cas du couloir notamment, les résultats sont très différents, mais deviennent égaux pour les fortes dimensions.

On a également souhaité comparer le lazy PRM au réseau dans le cas du camion, de la tasse et du coude en L. Le nombre de points nécessaires dans le cas du réseau est toujours inférieur à celui en moyenne nécessaire pour le lazy PRM, même si ce n'est remarquable que pour le camion et la tasse. Pour le coude en L, les résultats sont à peu près équivalents. La principale différence est ailleurs : le temps de compilation est beaucoup plus important dans le cas du lazy PRM que dans le cas des réseaux. Notons également que pour lazy PRM, 85% du temps de compilation est consacré à la précompilation, soit générer les points et chercher leurs voisinages.

Conclusion

Cet article a l'avantage de proposer une nouvelle idée en montrant que les méthodes déterministes ne sont forcément désavantagées par rapport aux méthodes aléatoires en robotique. Ces pistes devront être testées plus profondément, notamment dans les grandes dimensions. De plus, il reste un travail à faire pour passer d'un ensemble à N éléments à un ensemble à $(N+1)$ éléments. Des recherches ont cependant l'air en cours sur ce problème.