Troisième Projet RO 2015-2016
Déformation de trajectoire dans un environnement dynamique pour la navigation autonome d'un robot

Sawssen Jalel et Philippe Marthon

4 janvier 2016

FIGURE 1 – Une voiturette autonome évitant un vélo !

1 Introduction à la planification de trajectoire

La planification de trajectoire [1] d'un ou de plusieurs robots est un problème fondamental de la navigation autonome des robots. "Où dois-je me déplacer ?" est une question récurrente que se pose un robot. Du point de vue de la Recherche Opérationnelle, le robot poursuit un objectif qui consiste à atteindre un certain endroit. En général, il y a un critère d'optimisation à satisfaire : il faut que le chemin pour s'y rendre soit le plus court possible ou encore que le temps pour s'y rendre soit le plus court possible. D'autre part, le robot doit éviter de se

1. On veillera à ne pas confondre le mot *chemin* qui désigne une courbe géométrique représentant la suite des positions que le robot doit prendre afin d'atteindre son but avec le mot *trajectoire* qui est un chemin paramétré par le temps. Une trajectoire nous informe non seulement où le robot doit être mais aussi quand et avec quelle vitesse

heurter à tout un ensemble d'obstacles immobiles ou en mouvement.
On supposera dans la suite que la "scène" dans laquelle se déplace le robot est une pièce rectangulaire délimitée par quatre murs. Les obstacles fixes seront connus a priori.

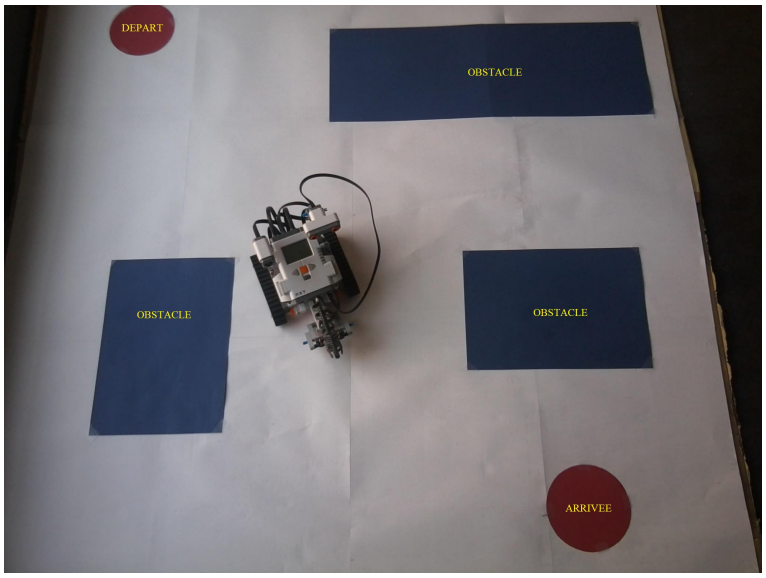

FIGURE 2 – Image de la Scène dans laquelle se déplace un robot LEGO. Les obstacles fixes sont en bleu

2 Description sommaire de la méthode de planification de trajectoire

Cette méthode se divise en deux parties offline et online :

1. La **planification offline** consiste à calculer a priori [2] un chemin permettant au robot d'aller d'un point de départ à un point d'arrivée en évitant les obstacles fixes (trajectoire réalisable) et en minimisant la longueur du chemin parcouru. Deux types de chemin sont ainsi calculés.
   (a) Un chemin formé d'une suite de segments de droite. Les extrémités de ces segments de droite seront appelés "Critical Curve Points" (CCP) (cf. 7). Ils marquent les positions où le robot change de direction.
   (b) Un chemin optimisé sous la forme d'une courbe NURBS (Non Uniform Rational B-Spline) qui est une courbe "lisse" dont la courbure est contrainte. **Cette première étape a déjà été programmée en MATLAB et le code, nommé "*Chemin_Initial*", vous est fourni.**
2. La **planification online** va consister à déformer en temps réel la trajectoire initiale de façon a éviter aussi les obstacles mobiles. **C'est cette**

2. ceci est possible car l'on suppose que les positions des obstacles fixes sont connus ainsi que le point de départ et le point d'arrivée du robot

planification online qui fait l'objet de ce troisième projet.Pour cela, deux stratégies de déformation seront développées et testées :

(a) La stratégie **TEDDY** (pour Trajectory Deformer) , proposée par V. Delsart et T. Fraichard. La trajectoire représentée par une courbe espace-temps est soumise à des forces de déformation à la fois externes et internes (pour maintenir la faisabilité et la connectivité de la trajectoire). Les forces externes sont des forces répulsives exercées par les obstacles de l'environnement pour éviter les collisions. Les forces internes sont introduites pour s'assurer que la trajectoire reste connexe, i.e. qu'il existe une trajectoire vérifiant la dynamique du robot entre deux noeuds consécutifs de la trajectoire.

(b) La stratégie **MICKEY** (pour ... Mickey qui échappe toujours à gros minet ...) , proposée par S. Jalel et P. Marthon. La déformation de la trajectoire est solution d'un problème d'optimisation. Les variables de ce problème sont les points de contrôle, les poids de la NURBS représentant la trajectoire courante et la vitesse. La fonction objectif à minimiser comprend quatre termes : le nombre de collisions, la faisabilité du chemin (nombre de points de la courbe qui dépasse la valeur limite de la valeur absolue de la courbure), la longueur du chemin et l'écart-type des courbures le long du chemin). Pour résoudre ce problème, on choisira une métaheuristique (algorithme génétique, essaim particulaire ou recuit simulé).

3 Planification offline

3.1 Modélisation de la scène

Le modèle de la scène est une image numérique im. Chaque pixel de cette image est repéré par son numéro de ligne i et son numéro de colonne j. Il représente une surface carrée d'aire fixée (par exemple un carré de 5 cm de côté). Par convention, la valeur im(i,j) sera égale à 7 si la surface du pixel est occupée par un obstacle, différente de 7 si elle est libre de tout obstacle

3.2 Modélisation du robot

Le robot est modélisé par un cercle de diamètre d (mesuré en pixels). On peut encore définir ce diamètre comme celui du plus petit cercle contenant le robot.

3.3 Couloirs

L'espace libre à l'intérieur duquel va se déplacer notre robot peut être divisé en "couloirs". Un couloir est un passage délimité par deux obstacles. Une CNS pour que notre robot puisse s'engager dans un couloir est que son diamètre d soit inférieur à la plus petite largeur de ce couloir. Pour savoir si notre robot peut s'engager dans un couloir on place le centre du cercle circonscrit au robot en un point quelconque de la ligne médiane ou "squelette" du couloir et on regarde si ce cercle intersecte ou non un obstacle. Ainsi, on voit la nécessité de mettre en oeuvre une squelettisation de l'espace libre.

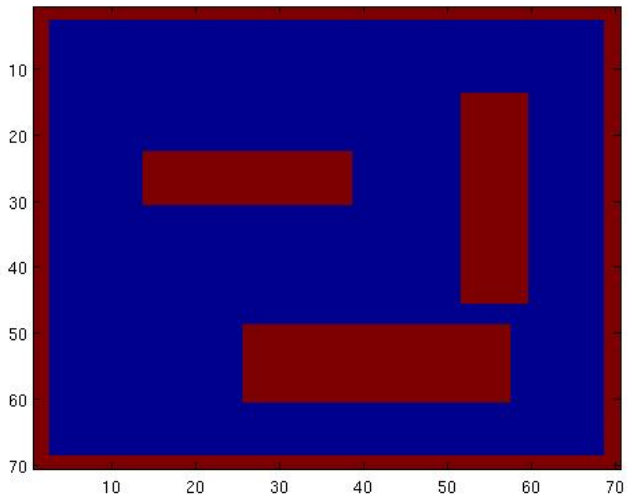

FIGURE 3 – Modèle de la Scène (image numérique imscene(70,70). Les obstacles fixes en rouge, l'espace libre en bleu

3.4 Squelettisation

La squlettisation est une opération morphologique classiquement utilisée en traitement d'images (par exemple en reconnaissance des caractères). Plusieurs méthodes existent. Nous utiliserons ici la méthode proposée dans le cours de traitement d'images (voir aussi http ://www.tsi.telecom-paristech.fr/pages/enseignement/ressources/beti/ske 4_8/compare.html) et qui consiste à effectuer une succession d'amincissements de la forme jusqu'à l'obtention de son squelette. Le robot va maintenent se déplacer sur le squelette (comme un train sur ses rails).

3.5 Raccordement des points de départ et d'arrivée au squelette. Création d'un squelette étendu

Dans le cas général, les points de départ et d'arrivée du robot ne font pas partie du squelette. Il faut donc raccorder le point de départ A et le point d'arrivée B au squelette. Cette opération doit nous fournir un nouveau squelette (étendu) sur lequel le robot va se déplacer.

3.6 Analyse du squelette

3.6.1 Post-squelettisation

Une fois le squelette étendu obtenu, on effectue une post-squelettisation qui permet de classer les pixels du squelette. Par convention :

1. si im(i,j) = 0 ou 1 alors (i,j) est un pixel "simple" du squelette
2. si im(i,j) = 2 (i,j) est un pixel "complexe" , 8-voisin d'un point de jonction
3. si im(i,j) = 3 (i,j) est une extrémité

4. si im(i,j) = 4 (i,j) est un point de jonction
5. si im(i,j) = 7 (i,j) est un pixel du fond contenant un obstacle.

Un pixel du squelette est une **extrémité** s'il ne possède qu'un seul pixel 8-voisin appartenant au squelette. Un pixel du squelette est un **point de jonction** s'il possède au moins 3 pixels 8-voisins appartenant au squelette.

3.6.2 Analyse

Nous sommes maintenant en mesure de faire une analyse de la forme du squelette. Lors de cette analyse, on approche chaque "branche" du squelette par une succession de segments de droite (approximation dite polygonale). Une **branche** est un morceau du squelette dont les deux extrémités sont soit un point de jonction soit une extrémité du squelette. Les extrémités des segments de droite qui marquent des changements significatifs de direction seront appelées **points critiques** (CCP). On ajoutera la convention :

6. si im(i,j) = 5 alors (i,j) est un point critique

Outre cette approximation polygonale, notre analyseur va construire un "graphe de sommets" représentatif de la morphologie du squelette. Dans ce graphe, chaque sommet est soit un point critique, soit un point de jonction, soit une extrémité. D'autre part, à chacun de ces sommets est associé ses coordonnées en ligne et en colonne. Chaque arc de ce graphe représente un couloir dont les extrémités sont les sommets du graphe.

3.6.3 Filtrage du graphe

Pour chaque arc-couloir, on doit tester si le couloir est suffisamment large pour que notre robot puisse le traverser. Si ce n'est pas le cas, un tel arc devra être supprimé. A l'issue de ce filtrage on a obtenu un graphe partiel dont chque arc-couloir peut être emprunté par notre robot. On décide maintenant que le robot ne pourra se déplacer que sur les arcs-couloirs du graphe G'.

3.7 Recherche du plus court chemin entre deux sommets

Grâce aux traitement précédents, on peut reformuler notre problème initial comme suit : trouver le plus court chemin joignant un sommet A (le point de départ) à un sommet B (le point d'arrivée) dans le graphe partiel G'. Pour cela, on peut utiliser l'algorithme de Moore-Dijkstra , de Bellman ou encore de Ford-Fulkerson pour les tensions.

3.8 Représentation du chemin initial par une NURBS

Le chemin précédemment trouvé est formé par une succession de segments de droite. Pour parcourir un tel chemin, le robot se déplace de façon rectiligne puis effectue une rotation sur place pour suivre le segment suivant. Cette rotation sur place outre le fait qu'elle n'est pas toujours physiquement possible, fait perdre beaucoup de temps. Pour surmonter cette difficulté, on cherche une nouvelle représentation du chemin sous la forme d'une courbe lisse. Le choix qui nous semble le mieux adapté est celle d'une courbe NURBS (Non Uniform Rational B-Spline)

3.8.1 Définition d'une NURBS

Une courbe NURBS de degré p est définie par :

$$C(u) = \frac{\sum_{i=0}^{n} N_{i,p}(u) w_i P_i}{\sum_{i=0}^{n} N_{i,p}(u) w_i} \quad 0 \leq u \leq 1$$

où les P_i sont les points de contrôle (formant un polygone de contrôle) , les w_i) sont les poids positifs et les $N_{i,p}(u)$ sont les fonctions de base B-spline de degré p définies sur le vecteur nodal :

$$U = (\underbrace{0, \ldots, 0}_{p+1}, u_{p+1}, \ldots, u_{m-p+1}, \underbrace{0, \ldots, 1}_{p+1})$$

Les fonctions de base B-spline de degré p, $N_{i,p}(u)$, sont ainsi définies :

$$N_{i,0}(u) = \begin{cases} 1 & \text{si } u_i \leq u < u_{i+1} \\ 0 & \text{sinon} \end{cases}$$

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u)$$

Le degré p , le nombre de points de contrôle $n + 1$ et le nombre de nœuds $m + 1$ sont reliés par l'équation :

$$m = n + p + 1$$

3.8.2 Calcul du chemin initial sous la forme d'une NURBS

A partir du chemin polygonal précédemment obtenu, on construit une courbe NURBS dont les points de contrôle sont les extrémités de chaque segment de ce chemin polygonal. Le problème est d'obtenir une coube NURBS "réalisable" sans collision et qui soit telle que le rayon de courbure en chaque point de cette courbe soit supérieur à une valeur minimale fixée. Ceci est fait en résolvant un problème d'optimisation dont les variables sont les poids associés aux points de contrôle.

4 Planification online ! A FAIRE !

Une fois le chemin initial trouvé, il est transmis au robot. Lors de son déplacement, la partie du mouvement restant à être exécutée est déformée continuellement en réponse aux informations sur l'environnement récupérées par les capteurs. Le robot peut ainsi modifier son parcours en fonction du déplacement d'obstacles, de l'imprécision et de l'incomplétude de sa connaissance de l'environnement. Les deux approches proposées (TEDDY et MICKEY) déforment la trajectoire du robot, modifiant le mouvement suivi à la fois dans l'espace et dans le temps. Pour cela, on raisonne sur le futur en utilisant une estimation du mouvement futur des obstacles mobiles.

1. La stratégie **TEDDY** (pour Trajectory Deformer) , proposée par V. Delsart et T. Fraichard. La trajectoire représentée par une courbe espace-temps est soumise à des forces de déformation à la fois externes et internes (pour maintenir la faisabilité et la connectivité de la trajectoire). Les forces externes sont des forces répulsives exercées par les obstacles de l'environnement pour éviter les collisions. Les forces internes sont introduites pour s'assurer que la trajectoire reste connexe, i.e. qu'il existe une trajectoire vérifiant la dynamique du robot entre deux noeuds consécutifs de la trajectoire.
2. La stratégie **MICKEY** (pour ... Mickey qui échappe toujours à gros minet ...) , proposée par S. Jalel et P. Marthon. La déformation de la trajectoire est solution d'un problème d'optimisation. Les variables de ce problème sont les points de contrôle, les poids de la NURBS représentant la trajectoire courante et la vitesse. La fonction objectif à minimiser comprend quatre termes : le nombre de collisions, la faisabilité du chemin (nombre de points de la courbe qui dépasse la valeur limite de la valeur absolue de la courbure), la longueur du chemin et l'écart-type des courbures le long du chemin). Pour résoudre ce problème, on choisira une métaheuristique : algorithme génétique (AG), essaim particulaire (EP) ou recuit simulé (RS) ; ces trois méthodes sont décrites succinctement dans le diaporama "Programmation dynamique , Programmation linéaire , Métaheuristiques" disponibles sur Moodle-N7-RO (diapos 172-173 pour RS , diapos 182-193 pour AG et diapos 205-208 pour EP).

5 Travail demandé

Programmation en Matlab des deux stratégies TEDDY et MICKEY et comparaison de leurs performances sur des exemples variés.

La date limite de remise du rapport de ce troisième projet est fixé au **vendredi 29 janvier 2016 à 18h**.

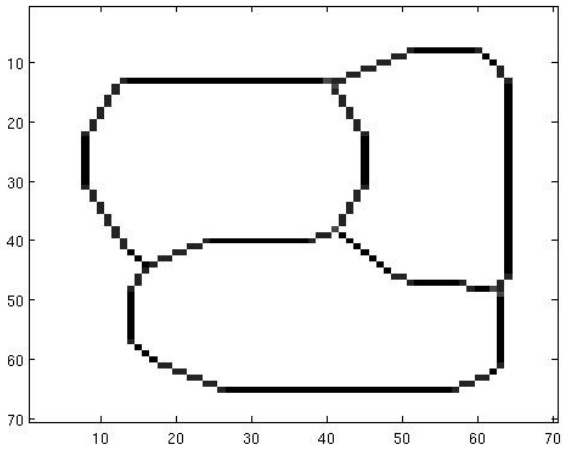

FIGURE 4 – Squelette

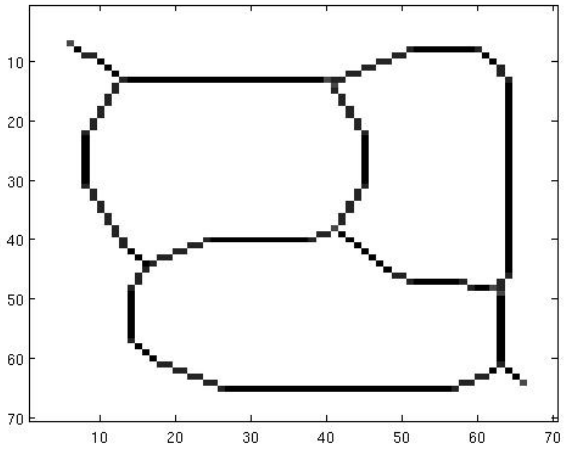

FIGURE 5 – Squelette étendu

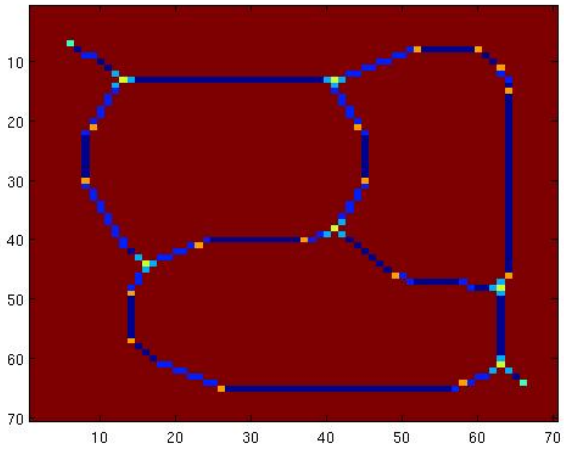

FIGURE 6 – Analyse du squelette. En jaune, sont les points de jonction et en orange, les points critiques

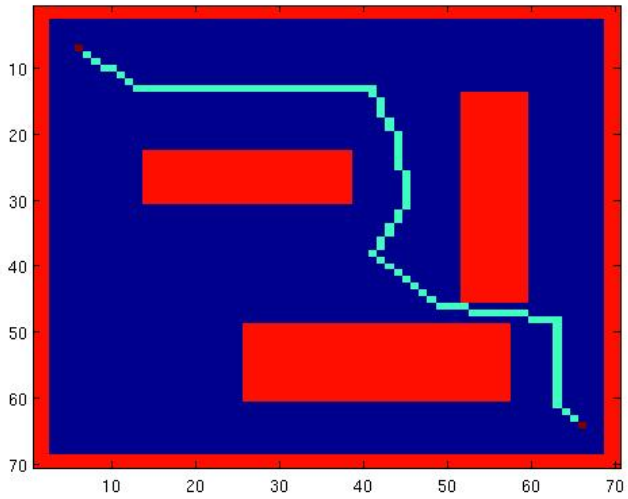

FIGURE 7 – Chemin initial. Diamètre du robot = 1 pixel

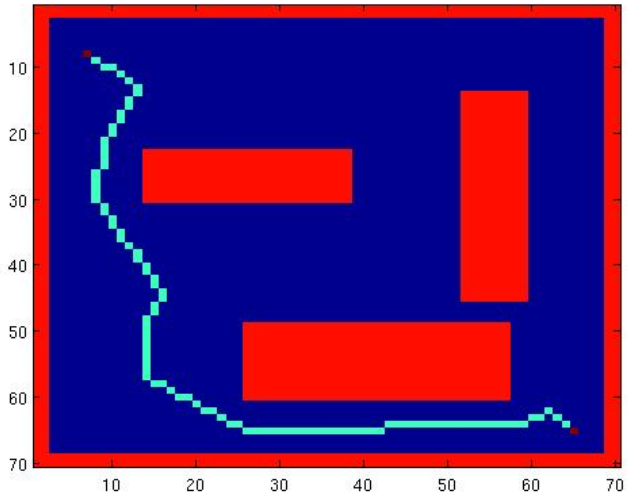

FIGURE 8 – Chemin initial. Diamètre du robot = 4 pixels

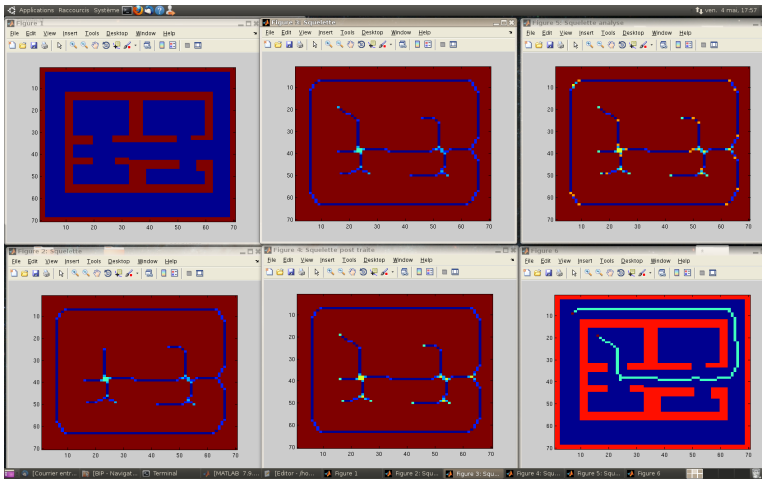

FIGURE 9 – Une autre expérience !

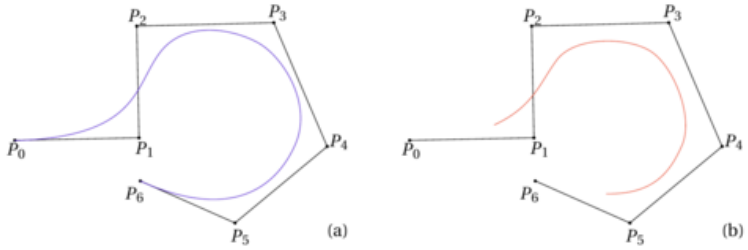

FIGURE 10 – Courbes NURBS possédant le même polygone de contrôle et des vecteurs nodaux différents (a) $U = (0, 0, 0, 0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1, 1, 1, 1)$ (b) $U = (0, \frac{1}{10}, \frac{1}{5}, \frac{3}{10}, \frac{2}{5}, \frac{1}{2}, \frac{3}{5}, \frac{7}{10}, \frac{4}{5}, \frac{9}{10}, 1)$

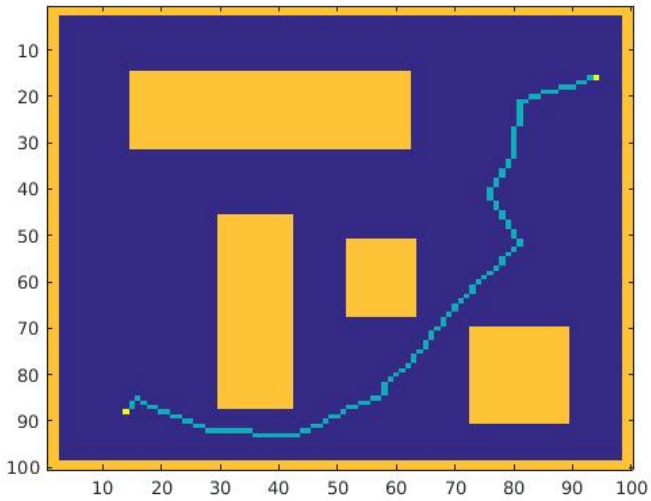

FIGURE 11 – Chemin initial sous forme d'une polyligne

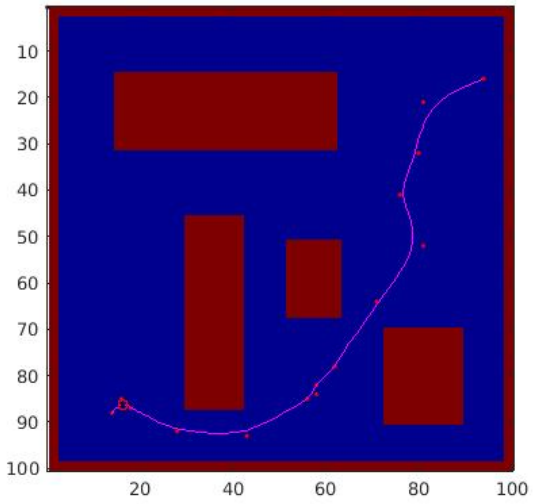

FIGURE 12 – Chemin initial sous forme d'une NURBS. Le cercle rouge entoure le point ne satisfaisant pas la contrainte de courbure.

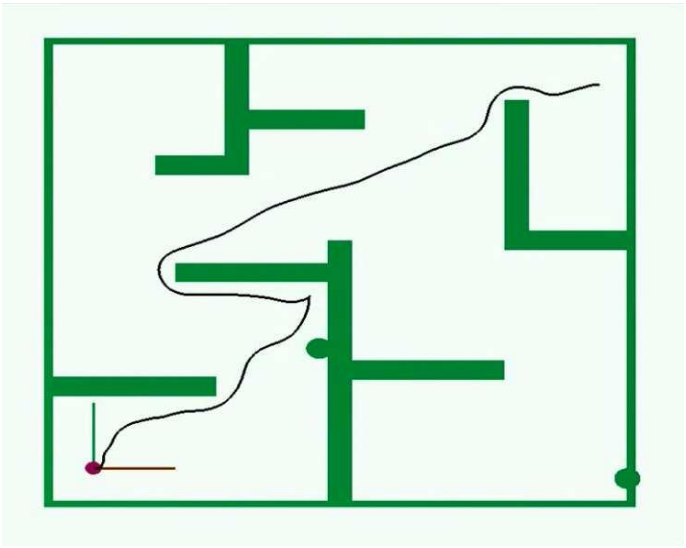

FIGURE 13 – Trajectoire TEDDY à t=0

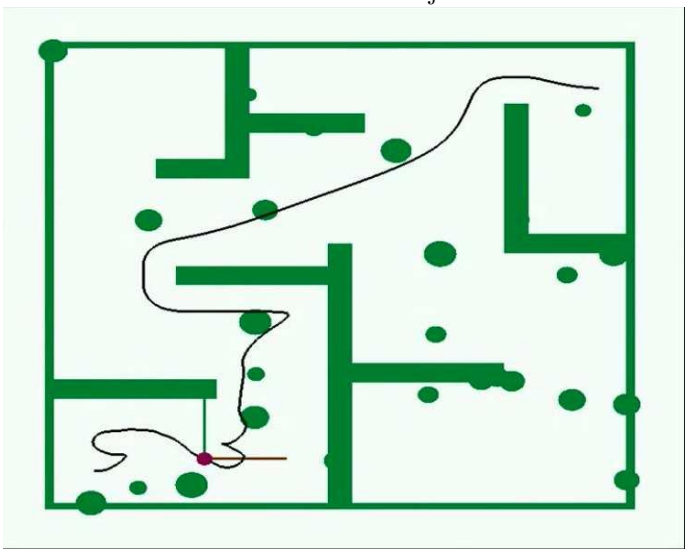

FIGURE 14 – Trajectoire TEDDY à t=5

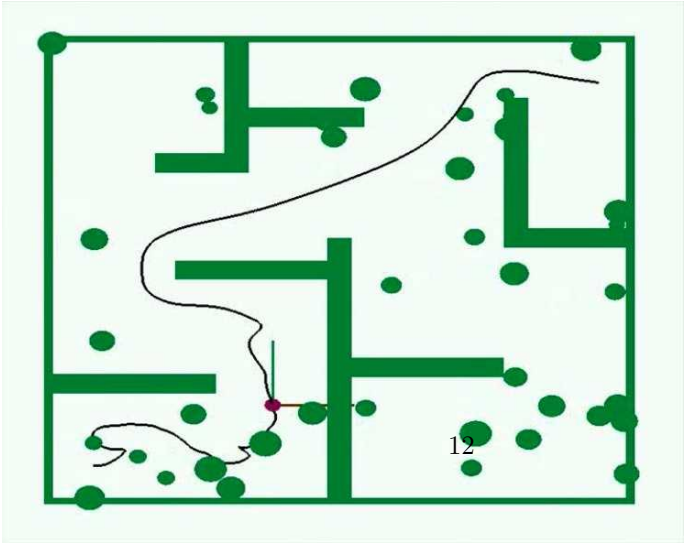

FIGURE 15 – Trajectoire TEDDY à t=10

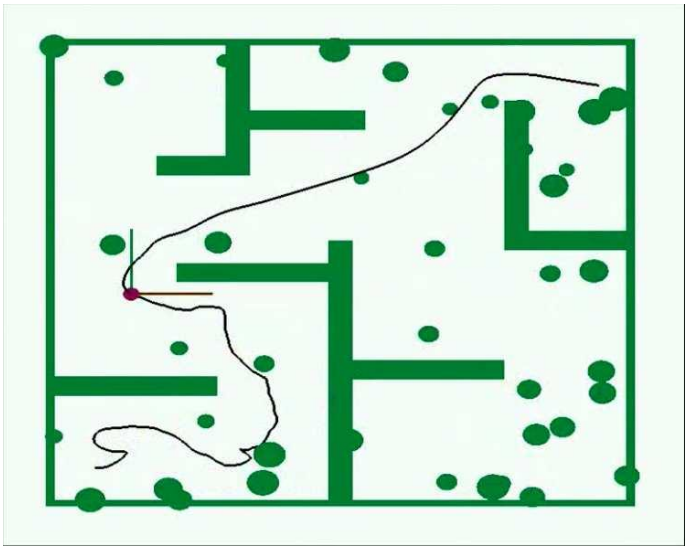

FIGURE 16 – Trajectoire TEDDY à t=20

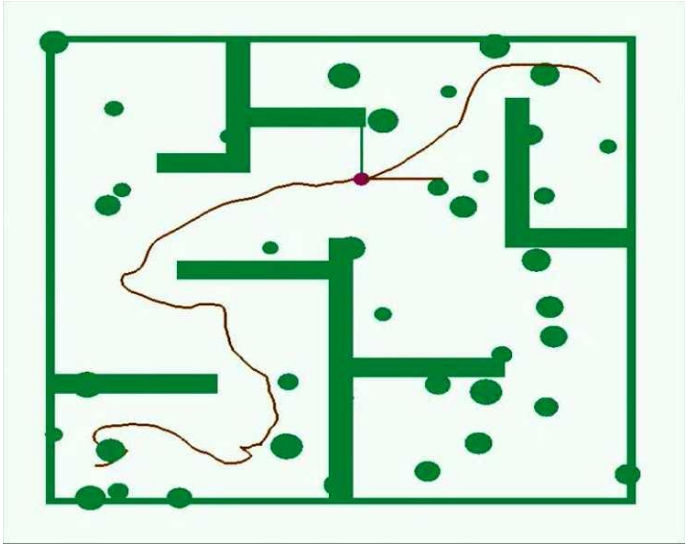

FIGURE 17 – Trajectoire TEDDY à t=25

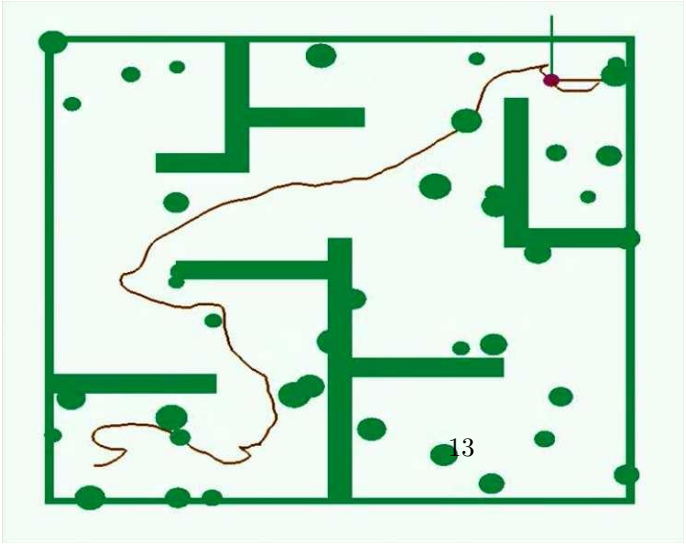

FIGURE 18 – Trajectoire TEDDY à t=30