Introduction à la robotique mobile **GLO-4001/GLO-7021**

Philippe Giguère A14

Laboratoire N^{o}5

Modèle de déplacement

Travail en équipe de 2

Matériel fourni : feuilles de papier, règle en aluminium de 1m, règle en plastique de 30 cm, ruban adhésif.

Partie 1 – Contrôle du robot

Le robot *iRobotCreate* se contrôle en envoyant des commandes sur son port série. Ces commandes sont sous la forme d'octets, dont les valeurs sont propres à certaines actions. À titre informatif, le protocole de communication complet du robot est décrit dans le document *CreateOpenInterface_v2.pdf*, disponible sur le site web du cours.

Pour vous faciliter la tâche, nous vous fournissons des fonctions Matlab programmées pour les principales actions du robot (*RoombaControl)*. Branchez le fil usb du robot sur l'ordinateur portable, démarrez le *iRobotCreate* et démarrez l'ordinateur sur l'image de la clé USB. Par courtoisie assurez-vous de vous connecter au réseau wifi *eduroam* avec votre idul@ulaval.ca et votre mot de passe avant de lancer « ./start » dans la console.

Démarrez Matlab sur votre station de base et assurez-vous d'être d'avoir ajouté les dossiers `ros4mat` ainsi que `RoombaControl` grâce à la fonction `addpath`. Assurez-vous aussi que `ros4mat` est compilé (la commande de compilation à partir de ros4mat/matlab est : `mex ros4mat.c wsock32.lib`). Vous pouvez maintenant vous connecter au robot à distance avec la commande `ros4mat('connect', IP, 1)`, où IP est la variable d'adresse IP de l'ordinateur connecté au robot.

Vous êtes maintenant prêt à commander le robot !

Attention! Il est suggéré de fixer l'ordinateur sur le robot afin qu'il ne tombe pas durant les déplacements parfois brusques.

Attention! Nous vous suggérons fortement de placer le robot par terre pour toutes les expérimentations dans ce laboratoire, à moins que vous le surveilliez très attentivement en tout temps. Il serait dommage de voir des robots s'affaler contre le sol…

Attention! Si vous donnez une vitesse au robot en utilisant la fonction roomba`SetFwdVelRadius`, celui-ci n'arrêtera pas tant que vous n'enverrez pas une commande de vitesse nulle (0).

Voici des exemples de fonctions que vous pouvez utilisez :

Nom de la fonction	Description
`roombaInit()`	*Initialise la connexion avec le robot*
`roombaBeep()`	*Émet un son*
`roombaSetFwdVelRadius(vitesse,rayonCourbure)`	*Fait avancer le robot à une vitesse donnée*
`roombaTravelDist(vitesse,distance)`	*Fait avancer le robot d'une distance donnée*
`roombaTurnAngle(vitesse,angle)`	*Fait tourner le robot sur lui-même d'un angle donné*

Vous devez d'abord utiliser la fonction Testez les fonctions pour vous familiariser avec leur utilisation. En particulier, testez `roombaTravelDist` et roomba`TurnAngle`. La vitesse des roues donnée en paramètre doit être comprise entre 0.025 m/s et 0.5 m/s pour les translations et entre 0.025 m/s et 0.2 m/s pour les rotations. Les distances sont en mètres et les angles en degrés.

Partie 2 – Modèle de déplacement p(x_{t+1}| x_t, u_t) : mesure du bruit

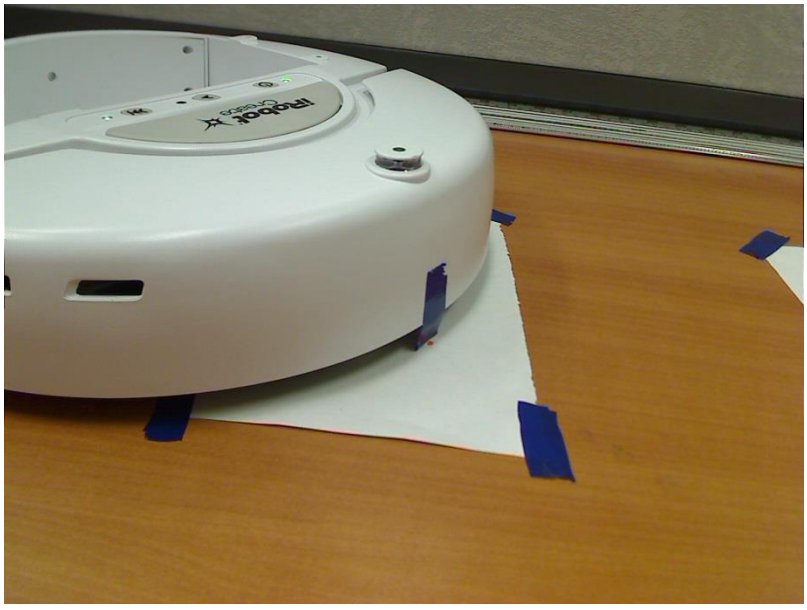
2.1 - En translation

Fixez un point de départ pour le robot, que vous pouvez identifier à l'aide de ruban adhésif. Collez trois feuilles de papier à 30, 60 et 90 cm du point de départ. Le centre des feuilles doit être approximativement à la distance demandée.

Vous pouvez utiliser plusieurs morceaux de ruban adhésif pour marquer la position de départ du robot. Assurez-vous d'être en mesure de le replacer à la même position précisément (au mm près), de même qu'à une orientation très similaire.

Faites avancer le robot par pas de 30cm à une vitesse de 0.05 m/s. Après chaque déplacement, marquez la position d'arrivée du robot sur la feuille correspondante à l'aide d'un stylo. Lorsque le robot est arrivé sur la 3^{e} feuille, replacez-le à position de départ. Refaites ces trois déplacements entre 10 et 15 fois.

Notez que pour vous aider à marquer précisément la position du robot, vous pouvez coller un bout de ruban adhésif frôlant le sol à l'avant du robot, au centre.



Lorsque vous avez terminé ces essais, mesurez, sur chaque feuille, la distance x_t de chaque point par rapport à la position de départ.

Entrez ces données dans Matlab et calculez la valeur moyenne μ_{pas} et la variance σ_{pas}^2 de la position sur chaque feuille. Pour ce faire, utilisez les commandes `mean`, `std` et/ou `var` qui représentent respectivement la moyenne, l'écart-type et la variance. Par exemple, si votre robot a les positions suivantes après un pas :

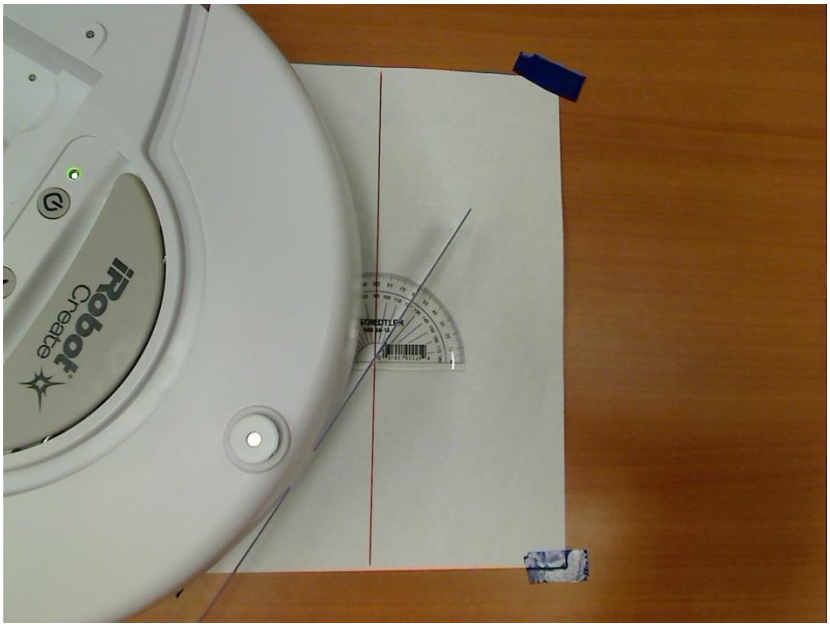
```
x1 = [30.2 30 29.8 30.3 30.1 29.9 30 30.4]
mu1 = mean(x1)
sigma1 = std(x1)
```

Utiliser la commande `hist(x1)` pour visualiser la distribution des distances après un certain nombre de pas. Est-elle approximativement gaussienne? Aussi, comment est-ce que l'erreur croît en fonction de la distance et du nombre de pas effectués?

Recommencez cette expérience en utilisant cette fois une vitesse de déplacement de 0.3 m/s. Comparez les la nouvelle moyenne μ_{pas} et la variance σ_{pas}^2 obtenues.

2.2 - En rotation

En vous servant d'une feuille de papier ou des lignes entre les tuiles sur le plancher, définissez une ligne correspondant à une orientation de départ (0°). Collez une règle sur le devant du robot, qui servira à mesurer l'angle du robot par rapport à la ligne de 0°. La figure suivante montre le robot placé à son orientation de départ.



Faites tourner le robot par pas de 30° avec une vitesse de rotation des roues de 0.05 m/s. Cette vitesse de rotation des roues correspond à une vitesse de rotation angulaire d'environ 22 °/s. Après chaque rotation, mesurez l'angle réel du robot à l'aide d'un rapporteur d'angle.

Lorsque le robot a atteint environ 90°, soit après trois rotations, replacez-le à son orientation de départ. Recommencer ces étapes entre 10 et 15 fois. Vous pouvez faire les essais en faisant tourner le robot dans les deux sens (en passant un angle négatif en paramètre).

Servez-vous encore une fois de Matlab pour calculer la moyenne μ_{rot} et la variance σ_{rot}^2 des orientations du robot après un, deux et trois pas. Comment est-ce que l'erreur sur l'orientation croît en fonction du nombre de pas effectués? Utiliser la commande `hist(x)` pour visualiser la distribution des orientations. Est-elle approximativement gaussienne?

Recommencer l'expérience avec une vitesse des roues de 0.2 m/s, ce qui correspond à une vitesse de rotation angulaire de près de 90 °/s. Comparez la moyenne μ_{rot} et la variance σ_{rot}^2 obtenues.

Partie 3 – Navigation à l'aveugle

Pour cette expérience, vous allez faire suivre au robot une trajectoire carrée. Pour ce faire, vous devez alterner entre des commandes de rotation et des commandes de translation, séparées par des pauses permettant au robot de terminer chaque déplacement.

Définissez une trajectoire carrée sur le plancher. Ceci peut être fait en utilisant les lignes entre les tuiles. Mesurez précisément la longueur des déplacements à effectuer pour suivre la trajectoire choisie. Placez ensuite le robot au point de départ et exécutez les déplacements jusqu'à ce qu'il revienne à son point de départ. Mesurez l'écart entre la position de départ et la position finale (position + angle). Continuez à faire quelques tours, jusqu'à ce que vous observiez une dérive dans la trajectoire effectuée par rapport à la trajectoire initiale. Après un certain temps, la trajectoire carrée devrait en effet se retrouver un peu décalée. Si le temps le permet, mesurez les coordonnées x-y (en cm) du robot, par rapport à la position de départ, après chaque déplacement et affichez la trajectoire réelle dans Matlab.

```
positions = [x1 y1; x2 y2; x3 y3; …];
figure(1); clf; hold on
plot(positions(:,1), positions(:,2), '.', 'LineWidth', 2, 'MarkerSize', 6)
plot(positions(:,1), positions(:,2))
```

Que concluez-vous sur la précision de la navigation à l'aveugle à court terme et à long terme?

Note : une fois le laboratoire terminé, n'oubliez pas d'enlever vos information personnelle de connexion au réseau sans fil.