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

Philippe Giguère A14

Laboratoire N^{o}1

Connexion et utilisation de la plateforme robotique

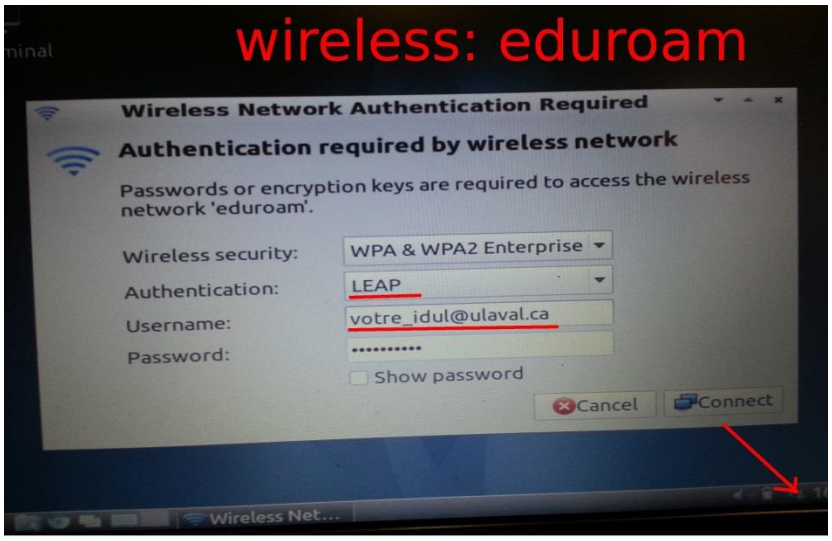
Travail en équipe

PARTIE 1 – DÉMARRAGE ET CONNEXION

Le robot fonctionne avec un ordinateur portable installé sur le dessus, connecté au Roomba et aux différents capteurs. Cet ordinateur doit exécuter l'image Linux prévue à cette fin, que nous fournissons sur les clés USB rouge. Un deuxième ordinateur, que nous appellerons ordinateur de travail, utilise une interface de communication dans Matlab pour contrôler le robot et accéder aux données des capteurs.

DÉMARRAGE DU ROBOT

Branchez le câble USB du robot dans l'ordinateur portable installé sur le robot. Démarrez l'image Linux sur cet ordinateur et assurez-vous que vous êtes connectés au réseau sans-fil de l'université. À titre informatif, pour vous connecter vous devez cliquer sur l'icône réseau dans la barre des tâches en bas à droite, sélectionner eduroam puis choisir LEAP comme protocole d'identification.



Ensuite, ouvrez un terminal en cliquant sur le raccourci placé sur le bureau ou encore en appuyant sur les touches Ctrl+Alt+T. Assurez-vous que la plate-forme *Roomba* est ouverte en appuyant sur le bouton du dessus (une lumière verte devrait s'allumer). Pour démarrer le programme du robot, tapez dans le terminal :

```
./start
```

L'adresse IP du robot devrait apparaître quelques secondes dans le terminal, avant que la suite du démarrage affiche l'état des différents processus. Vous pouvez aussi à tout moment obtenir l'adresse IP avec la commande `ifconfig`. Notez cette adresse IP.

Si une erreur concernant l'ADC apparaît en boucle, tentez de débrancher et de rebrancher le câble USB de l'ADC. Lorsque tout semble en bon état de marche, vous pouvez baisser le capot de l'ordinateur.

INSTALLATION DE L'INTERFACE ROS4MAT ET CONFIGURATION DE L'ORDINATEUR DE TRAVAIL ET CONNEXION AU ROBOT

Les communications entre Matlab sur votre desktop et le robot mobile se fait à travers une interface appelée ros4mat. Cette interface est contenue dans le fichier `ros4mat.zip` , disponible sur le site web du cours. Aussi, téléchargez le toolbox de contrôle du Roomba `RoombaControl.zip` sur le site web du cours. Créez un dossier sur votre ordinateur à un endroit approprié pour travailler avec le robot. Déposez-y le dossier de toolbox du roomba ainsi que le dossier ros4mat.

Avant de pouvoir utiliser l'interface de communication sans-fil `ros4mat` avec le robot, vous devez la compiler. Vous ne devrez faire cette étape qu'une seule fois. Dans la fenêtre principale de Matlab, exécutez la commande suivante :

```
mex -setup
```

Choisissez le compilateur à utiliser. Par défaut, il se peut que vous n'ayez qu'un seul choix. Sinon, il est recommandé d'utiliser GCC sur Linux et LCC sur Windows. Dans le cas où la compilation donne une erreur, vous pourrez revenir choisir un compilateur différent. Pour compiler l'interface de communication ros4mat sur Linux, exécutez les commandes suivantes :

```
cd ros4mat/matlab/
mex ros4mat.c
cd ../..
```

Sur Windows, utilisez plutôt la commande suivante :

```
cd ros4mat/matlab/
mex ros4mat.c wsock32.lib
cd ../..
```

Après la compilation, qui devrait normalement s'effectuer très rapidement, vous pourrez exécuter les commandes pour vous connecter au robot. Vous devez réussir cette étape avant de continuer plus loin.

Dans Matlab, changez le répertoire courant (« CurrentFolder », en haut de la fenêtre) pour votre dossier de travail. Créez un nouveau fichier script, enregistrez-le dans votre dossier et ajoutez les lignes suivantes :

```
clearvars
addpath('ros4mat/matlab/')
addpath('RoombaControl/')
ros4mat('connect', IP, 1)
```

Une bonne habitude est de commencer tout fichier script avec la commande `clearvars`, qui efface toutes les variables en mémoire. Les commandes `addpath` indiquent à Matlab de rechercher également dans ces deux sous-dossiers les fonctions que vous appelez. Finalement, la dernière ligne

établit la connexion avec le robot. Vous devez remplacer ***IP*** par l'adresse IP du robot que vous avez notée précédemment, écrite entre guillemets.

À titre informatif, le paramètre ayant la valeur 1 dans la commande `ros4mat` ci-haut active la compression des données pour leur envoi par le réseau. Cela est utile pour transférer des images de la caméra par exemple. Cette fonctionnalité peut être désactivée en mettant un 0 à la place.

Pour terminer une connexion avec le robot, exécutez la commande :

```
ros4mat('close')
```

PARTIE 2 – ACCÈS AUX DONNÉES DES CAPTEURS

Voici la liste des capteurs disponibles sur le robot :

- Caméra
- Gyroscope
- Capteur infrarouge moyenneportée (10-80 cm)
- Capteur infrarouge longue portée (20-150 cm)
- Hokuyo (optionnel)

FONCTIONS D'ACCÈS AUX DONNÉES DES CAPTEURS

Pour accéder aux données d'un capteur dans Matlab, vous devez d'abord vous abonner à ce capteur :

```
ros4mat('subscribe','Capteur',FreqAcquisition,ExtraInfo,TailleBuffer,FreqPolling,ExtraInfo2)
```

Les détails de chaque paramètre sont donnés plus bas. Ensuite, vous pouvez récupérer les données avec la commande :

```
[donnees, timestamps] = ros4mat('Capteur');
```

Le vecteur `timestamps` contient les valeurs d'horodatage, c'est-à-dire le temps précis auquel chacune des données a été mesurée. Les valeurs d'horodatage ont une résolution d'environ 0.1 ms.

Lorsque vous savez que vous n'utiliserez plus un capteur, vous pouvez vous désabonner à ce capteur pour libérer les ressources :

```
ros4mat('unsubscribe', 'Capteur')
```

DESCRIPTION DES PARAMÈTRES

Capteur

Nom du capteur auquel vous désirez accéder. Les valeurs possibles sont (entre guillemets simples) : `'camera'`, `'adc'`, ou `'hokuyo'`.

FreqAcquisition

Fréquence à laquelle le capteur prend les mesures (en Hz). En général, plus la fréquence est élevée mieux c'est. Cependant, une fréquence élevée signifie plus de données à transférer et à traiter. Par ailleurs, il ne sert à rien de mettre une fréquence plus élevée que la fréquence maximale supportée par le capteur.

ExtraInfo

Propre à chaque capteur. Pour la caméra, il s'agit de la résolution de l'image. Pour le convertisseur analogique-numérique (ADC), il s'agit des canaux à activer. Pour les autres capteurs, ce paramètre n'est pas utilisé (vous devez alors mettre 0).

TailleBuffer

Taille du tampon circulaire. L'interface de communication avec le robot utilise un tampon circulaire pour chaque capteur afin d'éviter les débordements mémoire. La taille de ce tampon peut être configurée selon l'application voulue. Ce paramètre revient à préciser le nombre de lectures à conserver en mémoire avant de commencer à supprimer les plus vieilles.

FreqPolling

Fréquence à laquelle l'ordinateur du robot lit les données du capteur (Hz). Cette variable permet de spécifier la fréquence de mise à jour du tampon circulaire. Choisir une fréquence différente de la fréquence d'acquisition permet d'éviter de surcharger l'unité lorsque la fréquence d'acquisition est très rapide, en lisant plusieurs valeurs du capteur à la fois. Par exemple, si la fréquence d'acquisition de l'ADC est de 2 kHz, mais que la fréquence de polling est de 100 Hz, alors le processus responsable de l'ADC lira 20 valeurs à la fois, réduisant ainsi le temps de processeur nécessaire. Dans tous les cas, les valeurs obtenues par Matlab seront les mêmes.

ExtraInfo2

Propre à chaque capteur. Pour la caméra, il s'agit de l'indice de la caméra à utiliser.

DÉTAILS D'UTILISATION DES CAPTEURS

ADC

Le paramètre ExtraInfo indique les canaux du convertisseur à activer. Il s'agit d'un nombre où chaque bit correspond à un canal. Les valeurs à indiquer pour activer chacun des canaux sont :

Capteur	Numéro du canal (bit)	Valeur de ExtraInfo
Infrarouge 20 à 150 cm	1	1
Infrarouge 10 à 80 cm	2	2
Gyro X	5	16
Gyro Z	6	32
Gyro 4X	7	64
Gyro 4Z	8	128

Pour activer plusieurs canaux, il suffit d'additionner les valeurs (c'est-à-dire effectuer un « ou » binaire). Par exemple, l'opération 1 + 16 (ou 1 | 16), soit la valeur 17, active les canaux 1 et 5. (L'opérateur | est le ou binaire).

L'ADC est capable de supporter une fréquence d'acquisition pouvant aller jusqu'à 10000 Hz lorsqu'un seul canal est activé et jusqu'à 2000 Hz lorsque plusieurs canaux sont activés. Cependant, la fréquence maximale de mesure du gyroscope est de 140 Hz et celle des capteurs infrarouges d'environ 25 Hz.

Le paramètre ExtraInfo2 n'est pas utilisé.

Caméra

La résolution de l'image, mentionnée par le paramètre ExtraInfo, doit être cohérente avec la fréquence d'acquisition. Les options disponibles pour la caméra du robot sont les suivantes :

Valeur de ExtraInfo	Résolution de l'image (pixels)	Valeur de FreqAcquisition correspondente (images/sec)
`'160x120'`	160x120	5, 10, 15, 20, 25 ou 30
`'320x240'`	320x240	5, 10, 15, 20, 25 ou 30
`'640x480'`	640x480	5, 10, 15, 20, 25 ou 30
`'800x600'`	800x600	5, 10, 15, 20 ou 25
`'960x720'`	960x720	5, 10 ou 15
`'1600x1200'`	1600x1200	5

L'indice de la caméra à utiliser doit être mentionné avec le paramètre ExtraInfo2. Si l'ordinateur du robot possède une webcam intégrée, il est possible par exemple que l'indice de cette dernière soit 0 et que l'indice de la caméra du robot soit 1.

Hokuyo

Le capteur Hokuyo est un cas particulier. Le seul paramètre utile lors de l'abonnement à ce capteur est TailleBuffer. Les paramètres FreqAcquisition, ExtraInfo, FreqPolling et ExtraInfo2 sont ignorés. Autrement dit, vous pouvez vous abonner à ce capteur en ne spécifiant que la taille du tampon désirée :

```
ros4mat('subscribe', 'hokuyo', 0, 0, 10, 0, 0)
```

La fréquence d'acquisition du Hokuyo est de 10 scans par seconde. Dans le cas de l'exemple précédent, le tampon peut donc conserver une seconde de données.

Également, dans le cas précis de ce capteur, trois variables sont retournées (au lieu de deux) lors de la récupération des données :

```
[donnees, timestamps, params] = ros4mat('hokuyo');
```

Le vecteur `params` contient les éléments suivants, dans l'ordre : angle min, angle max, incrément d'angle entre les mesures, portée min, portée max, incrément de temps entre les mesures. Ces valeurs sont primordiales pour construire une carte de l'environnement à partir des données brutes du Hokuyo.

EXEMPLES D'UTILISATION

ADC

L'exemple suivant permet de recevoir 5 secondes de données du gyro Z à 140 Hz :

```
ros4mat('subscribe', 'adc', 140, 32, 500, 10)  % 500/100Hz = 5 secondes
pause(5);    % Donne le temps au buffer de se remplir
[donnees, timestamps] = ros4mat('adc')
ros4mat('unsubscribe', 'adc') % Optionnel
```

Pour recevoir des données en continu, vous pourriez faire :

```
ros4mat('subscribe', 'adc', 140, 32, 500, 10)  % 500/100Hz = 5 secondes
while true
    [donnees, timestamps] = ros4mat('adc')
    ...
end
ros4mat('unsubscribe', 'adc') % Optionnel
```

Caméra

L'exemple suivant permet de capturer et afficher une image de la caméra :

```
ros4mat('subscribe', 'camera', 5, '640x480', 1, 5, 1)
image = []; maxTry = 0;
while(isempty(image)  & maxTry < 5)
   [image, timestamp] = ros4mat('camera');
    maxTry += 1;
end
imagesc(image(:,:,:,1))% 3 premières dimensions = RGB. La 4e =indice de l'image.
ros4mat('unsubscribe', 'camera') % Optionnel
```

Pour recevoir un flux vidéo continu vous pourriez faire :

```
ros4mat('subscribe', 'camera', 30, '640x480', 1, 30, 1)
pause(0.5);    % Donne le temps à la caméra de prendre une image
figure(1)
while true
    [image, timestamp] = ros4mat('camera');
    if ~isempty(image) % si l'image est vide, c'est parce que la caméra n'a
                       % pas eu le temps de prendre une nouvelle image
                       % depuis la dernière boucle. Dans ce cas on passe.
        imagesc(image(:,:,:,1))
        drawnow
    end
end
ros4mat('unsubscribe', 'camera') % Optionnel
```

PARTIE 3 – CONTRÔLE DU ROBOT

Les fonctions de contrôle du Roomba, utilisées au laboratoire précédent, ont été modifiées afin de pouvoir contrôler le robot à distance, en utilisant l'interface de communication sans-fil. Les fonctions disponibles sont présentées au tableau suivant. Notez que vous n'avez plus besoin de passer en paramètre une variable `serPort`. Celle-ci est gérée automatiquement.

Fonction	Description
roombaInit	Initialise la connexion avec le Roomba (aucun paramètre)
roombaBeep	Émet un son (aucun paramètre)
roombaDemo(type_demo)	Effectue une démonstration de déplacement
roombaSetLEDs(LED, couleur, intensité)	Change la couleur et la luminosité des LEDs
roombaSetFwdVelRadius(vitesse, rayon_courbure)	Fait avancer le robot en tournant selon le rayon de courbure donné
roombaTravelDist(vitesse, distance)	Fait avancer le robot d'une distance donnée
roombaTurnAngle(vitesse, angle)	Fait tourner le robot d'un angle donné

Pour recourir à l'aide concernant chacune de ces fonctions, ouvrez le fichier de la fonction ou encore appuyez sur F1 lorsque le curseur clignote sur le nom de la fonction.