Documentation MINDSTORMS

TER 2023/2024

Yassine DERGAOUI - Mamadou cire CAMARA

Mise en place du serveur MQTT

1. Serveur MQTT

Le but de cette tâche est de créer un serveur écoutant les échanges et les connexions de messagerie par protocole MQTT, cette approche est très souvent utilisée pour la communication de machine à machine dans le même réseau et sera utile pour nos expérimentations.

Pour cela, **Eclipse Mosquitto** a été choisie comme service permettant la mise en place du serveur. Il suffisait de l'installer sur une machine qui va servir comme hébergeur du serveur tout au long des expérimentations.

Voici le guide pour installer Eclipse Mosquitto et pour tester les différentes communications possibles (messages, topics, json, xml...) par la mise en place d'un client MQTT. Assurez-vous de bien lancer le serveur en spécifiant sa configuration.
La procédure n'est pas aussi simple que sur Windows, il pourrait y avoir des problèmes de pare-feu qui empêchera les connexions distantes.

- Pour résoudre ce problème, on peut ajouter une règle au pare-feu Windows pour permettre les connexions entrantes sur le port 1883 en utilisant le protocole TCP.
- Pour des raisons de sécurité, et dans le cadre de notre environnement de développement, il est recommandé d'ajouter l'option **enable=yes.** Cela va permettre la suppression de cette règle au prochain démarrage de la machine de développement.

```
netsh advfirewall firewall add rule name="MQTT Allow incoming TCP 1883" protocol=TCP
dir=in localport=1883 action=allow enable=yes
```

Pour éviter les accès non autorisés au serveur MQTT, Il est recommandé de définir l'authentification pour celui-ci.

Maintenant que le serveur est prêt pour recevoir les connexions client depuis les machines connectées au même réseau que celui du serveur, on peut mettre en place le client MQTT sur les robots Mindstorms.

Guide/Ref: Mosquitto Authentification

NOTE: Dans le cadre de ce TER et pour faciliter la collaboration, le serveur MQTT est hébergé dans une machine virtuelle dans le cloud.

2. Client MQTT:

Il existe plusieurs bibliothèques Java qui peuvent servir comme client MQTT, on utilise dans notre cas Eclipse Paho.

Le framework LeJOS installé sur le logiciel des robots accepte uniquement les programmes compilés en Java 1.7, ce qui n'a pas permis à la bibliothèque de s'exécuter sur le programme puisque celle-ci a été forcement compilée en une version Java plus récente.

La solution était de chercher dans les archives de la bibliothèque et d'utiliser cette version. Il doit être mis dans le build path du projet Eclipse pour être utilisé.

Il ne reste qu'à écrire le programme nécessaire pour réaliser la communication entre le serveur et le robot. Assurez-vous de connecter le robot au même réseau. Notez aussi l'adresse IP de la carte réseau à laquelle elle est connectée.

Liste des programmes réalisés pour l'expérimentation:

- Un programme qui permettra de se connecter au serveur MQTT ainsi que la création d'un topic ou le robot va émettre le message **" Salut, de EV3 !! "** (Trello).
- Un programme qui permet l'écoute des messages reçus depuis le serveur et les utilise dans l'EV3. L'exemple que nous avons mis en place consiste à faire avancer ou reculer le robot en fonction d'un signal reçu depuis le serveur.

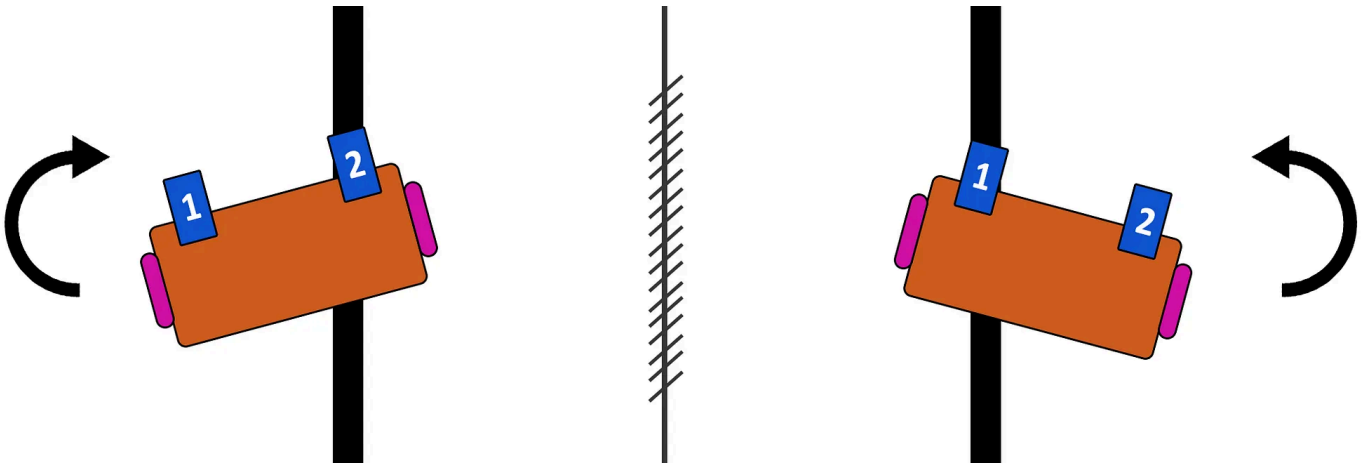
Mise en place de la Conduite Autonome

1. Recherche sur des algorithme

Algorithme Line Following :

Nous utiliserons un robot avec 2 capteurs de couleur de chaque côté. Alternativement, des capteurs de lumière/ultraviolets peuvent également fonctionner.

Case 1

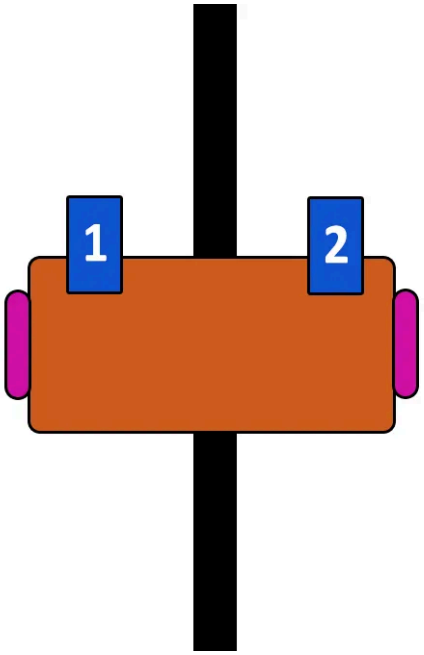


Dans ce cas, seul le capteur droit (Capteur 2) ou le capteur gauche (Capteur 1) détecte une ligne, c'est-à-dire qu'ils détectent la couleur noire.
Si seul le capteur droit détecte une ligne noire , alors le robot doit se diriger vers la droite pour suivre la ligne. Alternativement, si seul le capteur gauche détecte une ligne noire , le robot doit alors se diriger vers la gauche pour suivre la ligne.

```
    if right_sensor detects black
      // Move Right
          set left motor ON
          set right motor OFF
```

```
    else if left_sensor detects black
     // Move Left
    set left motor OFF
    set right motor ON
   endif
```

Case 2



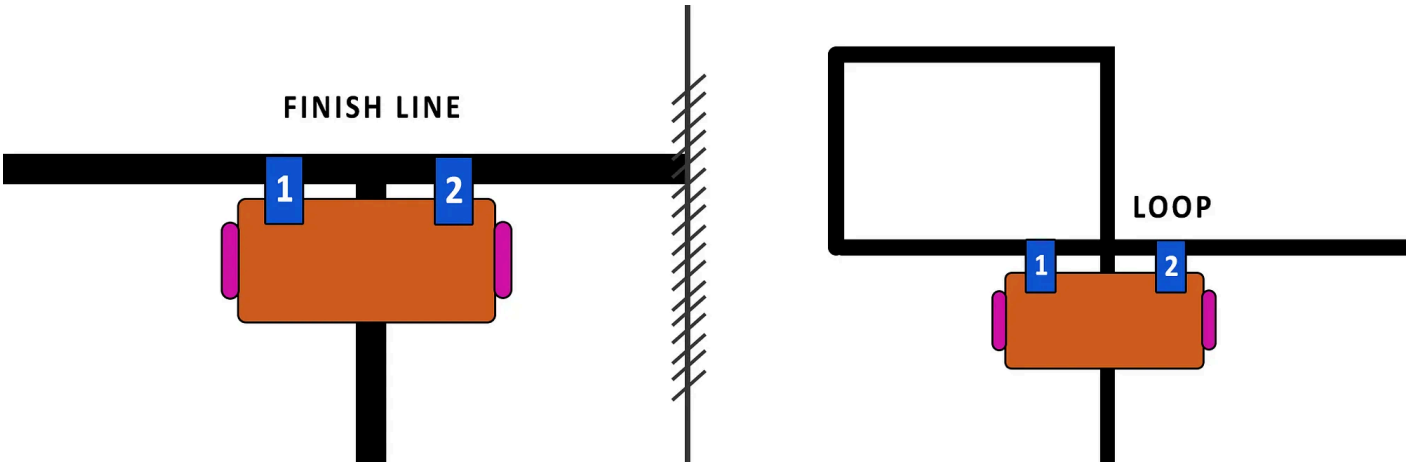
Dans ce cas, les deux capteurs ne détectent pas de ligne noire , le robot doit donc continuer à avancer dans la même direction et maintenir sa trajectoire actuelle.

```
if right_sensor and left sensor do not detect black

   set left motor ON

   set right motor ON
```

Cas 3:



Ce cas est un peu délicat. En fonction de votre scénario de jeu, les deux capteurs détectant une ligne noire peuvent signifier soit qu'il s'agit d'une ligne d'arrivée (image de gauche) , soit simplement d'une boucle qui fait partie d'un problème plus vaste de suivi de ligne (image de droite).
Si cela signifie une ligne d'arrivée, alors le robot doit s'arrêter et terminer le programme.
Sinon, si cela signifie une boucle, alors :
1. Le robot doit s'arrêter pendant une durée de 1 à 2 secondes pour se stabiliser,
2. Déplacez le robot un peu vers l'avant *(environ 1/4ème de rotation)* pour dépasser le noir. ligne puis,
3. Continuez l'algorithme de suivi de ligne normal.
La signification de la détection d'une ligne noire par les deux capteurs doit être prédéterminée avant le début du programme.

```
// Black line is the finish line
if left_sensor and right_sensor detect black
   set left motor OFF
   set right motor OFF

// Black line is a part of a loop
if left_sensor and right_sensor detect black
   set both motors OFF for 2 seconds (Stabalise)
   set both motors ON for 1/4th rotation (Move past the black line)
   continue to normal line-following algorithm
```

Complete Algorithm:

```
if right_sensor detects black
   // Move Right
   set left motor ON
   set right motor OFF

else if left_sensor detects black
   // Move Left
   set left motor OFF
   set right motor ON

else if right_sensor and left sensor do not detect black
   set left motor ON
   set right motor ON

else if left_sensor and right_sensor detect black
   set left motor OFF
   set right motor OFF
endif
```

Line Follower Robot Algorithm - Written by Jay Gupta

Algorithme Filtrage de Kalban:

Cet algorithme utilise des systèmes et des capteurs que nous ne possédons pas.

ref :

https://projets-ima.plil.fr/mediawiki/images/5/52/Rapport_Projet_P24_Belhouachi.pdf

2. Recherche sur les Capteurs

Capteur UltraSons:

Le bloc Capteur à ultrasons reçoit les informations du capteur à ultrasons. Vous pouvez mesurer la distance en pouces ou en centimètres et obtenir une valeur de sortie numérique. Vous pouvez également comparer la distance à une valeur de seuil pour obtenir une valeur de sortie logique (Vrai ou Faux). Vous pouvez aussi détecter d'autres ultrasons en mode Présence.

Pour plus d'informations sur le fonctionnement du capteur à ultrasons, les informations qu'il fournit et des exemples de programmation, consultez la section Utilisation du capteur à ultrasons.

Capteur Couleur:

Le bloc Capteur de couleur reçoit les informations du capteur de couleur. Vous pouvez mesurer la couleur ou l'intensité de la lumière et obtenir une valeur de sortie numérique. Vous pouvez également comparer les données du capteur à une valeur d'entrée et obtenir une valeur de sortie logique (Vrai ou Faux).

Pour plus d'informations sur le fonctionnement du capteur de couleur, ses modes et les données fournies, ainsi que des exemples de programmation, consultez la section Utilisation du capteur de couleur.

> VALEURS D'ENTRÉE ET DE SORTIE

Les valeurs d'entrée disponibles pour le bloc Capteur de couleur varient selon le mode sélectionné. Vous pouvez saisir les valeurs d'entrée directement dans le bloc. Les valeurs d'entrée peuvent également être définies par les Fils de données des valeurs de sortie d'autres blocs de programmation.

Valeur d'entrée	Type	Valeurs autorisées	Remarques
Liste de couleurs	Tableau de nombres	Chaque élément : 0 – 7	Couleur(s) sélectionnée(s) en mode Comparer – Couleur : 0 = Aucune couleur 1 = Noir 2 = Bleu 3 = Vert 4 = Jaune 5 = Rouge 6 = Blanc 7 = Marron
Type de comparaison	Numérique	0 - 5	0 : = (Égal à) 1 : ≠ (Différent de) 2 : > (Supérieur à) 3 : ≥ (Supérieur ou égal à) 4 : < (Inférieur à) 5 : ≤ (Inférieur ou égal à)
Valeur de seuil	Numérique	Tout nombre	Valeur à laquelle comparer les données du capteur
Valeur	Numérique	0 – 100	Intensité lumineuse pour les modes Étalonner

Les valeurs de sortie disponibles varient selon le mode sélectionné. Pour utiliser une valeur de sortie, connectez-la à un autre bloc de programmation avec un Fil de données.

Valeur de sortie	Type	Remarques
Couleur	Numérique	Valeur de couleur détectée : 0 = Aucune couleur 1 = Noir 2 = Bleu 3 = Vert 4 = Jaune 5 = Rouge 6 = Blanc 7 = Marron
Résultat de comparaison	Logique	Résultat Vrai/Faux en mode Comparer.
Lumière	Numérique	Intensité lumineuse (0 à 100).

Capteur Tactile(Contact):

Le bloc Capteur tactile reçoit les informations du capteur tactile. Vous pouvez tester si le capteur tactile est enfoncé,

relâché ou heurté, et obtenir une valeur de sortie logique (Vrai ou Faux).

Pour plus d'informations sur le fonctionnement du capteur tactile, les informations qu'il fournit et des exemples de programmation, consultez la section Utilisation du capteur tactile.

> VALEURS D'ENTRÉE ET DE SORTIE

Les valeurs d'entrée disponibles pour le bloc Capteur tactile varient selon le mode sélectionné. Vous pouvez saisir les valeurs d'entrée directement dans le bloc. Les valeurs d'entrée peuvent également être définies par les Fils de données des valeurs de sortie d'autres blocs de programmation.

Valeur d'entrée	Type	Valeurs autorisées	Remarques
État	Numérique	0 - 2	État à tester en mode Comparer. 0 = Relâché 1 = Enfoncé 2 = Heurté

Les valeurs de sortie disponibles varient selon le mode sélectionné. Pour utiliser une valeur de sortie, connectez-la à un autre bloc de programmation avec un Fil de données.

Valeur de sortie	Type	Remarques
État	Logique	Utilisée en mode Mesure. Vrai si le capteur tactile est enfoncé, Faux s'il ne l'est pas.
Résultat de comparaison	Logique	Valeur de l'état de capteur sélectionné en mode Comparer.
Valeur mesurée	Numérique	État actuel du capteur en mode Comparer. 0 = Relâché 1 = Enfoncé 2 = Heurté

Ref : Bloc Capteur à ultrasons