Document générale

Suivit

Ce document a le but de vous présenter le sujet choisi pour cette option de spécialisations. Tout d'abord, je vous présente le sujet que j'ai choisi ainsi que la raison de ce choix. Puis je vous présente les différents les différentes étapes de développement de la solution. Ensuite, je vous mets en avant les problèmes que j'ai rencontrés durant le développement de ce projet avec les solutions qui m'ont permis à résoudre ces problèmes et je termine par mettre à votre disposition le manuel d'utilisation pour ce projet.

Présentation du sujet et pourquoi ai-je choisi ce sujet ?

Suivit

Pour pouvoir valider cette option de spécialisations, j'ai choisi le projet numéro 1 ce qui est appelé "Suivit". Le but dans ce projet est d'implémenter une solution multi-agents qui permet de suivre un agent leader par tous les autres agents qui existent dans l'application.

Pourquoi ce projet?

Les cours que j'ai suivis pour cette option de spécialisations étaient mes premiers cours dans le domaine de robotique. Ces cours m'ont permis de découvrir les différents techniques pour mieux comprendre le domaine de robotique. Au départ, mon choix s'est porté vers le projet 5 qui est "Formule 1" (un projet de Netlogo) car ce projet me semblait plus simple à réaliser et l'environnement ROS (Robot operating system) me paraissait compliqué en tant que débutant dans ce domaine. Mais je voulais profiter de mes connaissances en C++. En effet, j'ai commencé à découvrir le monde de ROS en lisant la documentation de ROS sur son site officiel et en consultant les vidéos de ce cours disponibles sur la plateforme dédiée. Après avoir acquis les bases en ROS, mon choix s'est porté cette fois-ci vers un projet ROS qui est "Suivit".

Étapes pour réaliser le projet

Les étapes que j'ai suivi à construire ce projet se décomposent ainsi :

1. Analyser le projet pour mieux comprendre les différentes problématiques
2. Préparer l'environnement de travail
3. Acquérir les connaissances nécessaires
4. Création des schémas UML (Unified Modeling Language)
5. Réaliser le projet en commentant le code

Analyser le projet

Une bonne analyse de ce projet m'a permis à repérer les différentes problématiques de ce projet et à déterminer les connaissances nécessaires et les outils à utiliser pour réaliser ce projet.

Les différentes problématiques que j'ai pu constater sont ainsi :

1. Comment choisir l'agent leader qui sera suivi par les autres ?
2. Comment les autres agents peuvent communiquer avec l'agent master afin de recevoir sa position en temps réel ?

Les connaissances et les outils qui étaient nécessaires à réaliser ce projet que j'ai pu constater sont :

1. Comprendre l'environnement ROS
2. Avoir des bases (Terminal, Comprendre l'anatomie de linux) pour travailler sous Ubunutu
3. Créer les schémas UML
4. Des bonnes connaissances soit en C++ soit en Python

Les Problèmes rencontrés

Pendant la réalisation de ce projet, j'ai rencontré plusieurs problèmes dont certains étaient résolus.

Problèmes résolus

J'ai entamé ce projet avec le langage de programmation C++ car c'est le seul langage qu j'ai connaissais avec lequel je pouvais réaliser ce projet. Après avoir défini les processus dans les schémas UML, j'ai commencé à les implémenter avec C++. En plein milieu de projet, pour obtenir le résultat par rapport un processus dans l'UML je devais avoir besoin du chemin de package "suivit". Pour cela ROS fournissait bien un package appelé "roslib" qui contient une fonction qui est "getPath" pour obtenir le chemin absolu d'un package. Par contre, Ce package ne fonctionnait pas avec la distribution utilisée par moi qui est "Kinetic".

Résolution

Pour résoudre ce problème, j'ai choisi de travailler avec Python, un autre langage de programmation qu'on peut utiliser avec ROS. En effet, je devais convertir tout mon code C++ en Python ainsi que apprendre Python car ce langage n'était pas maîtrisé par moi. Cette résolution m' permis à découvrir un nouveau langage de programmation et comprendre le fonctionnement de ROS avec Python.

Problèmes non résolus

Après avoir terminé le projet, j'ai constaté qu'il y a un décalage de suivi entre le leader agent et les agents qui suivient. Le problème c'est que, les agents qui suivent ne réagissent pas aux déplacements de quelques secondes effectués par le leader agent.

Par rapport à ce problème j'ai cherché plusieurs solutions sur internet mais malheureusement je n'ai trouvé aucune solution pour résoudre ce problème.

La mise en place et le manuel d'utilisation

Pour utiliser ce projet sur votre environnement, tout d'abord vous avez besoin de télécharger soit par la plateforme de supinfo soit depuis mon github qui est https://github.com/RamnSingh/RosSuivit.git.

Après avoir téléchargé ce projet, je vous liste les étapes à suivre pour l'exécuter :

1. Placez-vous dans le dossier "src" de package suivit qui se trouve dans le projet téléchargé qui est également un espace de travail (workspace). Sur mon pc, le projet se trouve à la racine donc pour moi le chemin est : /ros/src/suivit/src.
2. Dans ce dossier vous allez trouver 4 fichiers de Python qui sont ainsi :
   - customizeFrames.py
   - main.py
   - publishLeaderPosToFollowers.py
   - publishLeaderPosToLeaderPosHolder.py
3. Pour chacun de ces fichiers, vous allez exécuter la commande suivante pour les exécutables : sudo chmod +x <nom-du-fichier-avec-extension>
4. Après vous allez vous retourner à la racine de ce projet pour charger les dépendances pour ce projet. La commande à exécuter est : catkin_make
5. Puis en restant toujours à la racine, vous allez exécuter la commande suivante afin d'utiliser les outils fournis par ROS afin de mieux trouver les packages ROS sur votre système d'exploitation.
   - source devel/setup.bash
6. Et ensuite pour lancer le projet, saisissez la commande suivante :

   roslaunch suivit main.launch agents:=5

   Dans la commande au-dessus, j'ai saisi 5 à la fin de la commande. Ce chiffre 5 représente le nombre des agents à générer.