AIP PRIMECA

Manuel d'utilisation

Solution de contrôle et de supervision du bras robot Stäubli RX60 sous ROS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 732287.
The RIO project starts in Jan 2020 and ends in Dec 2020.

RIO ROSin

Nathan MORET


RIO


ROSin

Table des matières

Initialisation de ROS

1. Création de l'espace de travail

- Choisir le nom de l'espace de travail. (ici « catkin_ws »)
- Ouvrir le fichier .bashrc :
  gedit .\bashrc
- Copier /coller les lignes suivante à la fin du fichier :
  source /opt/ros/melodic/setup.bash
  source $HOME/catkin_ws/devel/setup.bash
- Sauvegarder et fermer le fichier .bashrc
- Lancer dans un terminal la commande suivante :
  source /opt/ros/melodic/setup.bash
  source $HOME/catkin_ws/devel/setup.bash
- Créer l'espace de travail :
  mkdir -p ~/catkin_ws/src
  cd ~/catkin_ws/src
  catkin_init_workspace
- Création du répertoire build et devel :
  cd ~/catkin_ws/
  catkin_make

2. Installation des packages

- Copier/coller les dossiers *com_arm* et *ros_rx60_workspace* dans le répertoire catkin_ws/src
- Rendre exécutable le programme du serveur :
  cd catkin_ws/src/com_arm/src
  chmod +x serveur1_tcp.py
- Compiler le projet
  cd catkin_ws
  catkin_make

Installation du driver Val3

- Télécharger le dossier *ros_com_rx60b* sur une clé USB
- Connecter la clé USB sur la baie du contrôleur CS8
- Passer en mode automatique
- Lancer le programme *ros_com_rx60b* depuis le pendant qui se trouve dans l'espace mémoire USB
- Entrer l'adresse IP du PC master où est hébergé le serveur

Utilisation de la solution

.

1. Lancement des nœuds

- Lancer dans un terminal ROS :
  roscore
- Lancer le serveur dans un autre terminal :
  rosrun com_arm serveur1_tcp.py
- Lancer le nœud de supervision dans un autre terminal
  rosrun com_arm robot
- Lancer dans un autre terminal l'application
  rosrun com_arm application

2. Créer une application

- Créer un fichier nom.cpp et copier /coller dedans le modèle de base d'une application
- Souscrire aux topics dont l'utilisation est nécessaire à l'application que l'on veut développer
- Déclarer les variables du programmes (points, joints ou paramètre de mouvement)
- Utiliser les fonctions que propose la classe Control pour programmer l'application dans l'espace dédié.
  Fonctions disponibles : control.moveJoint()
  control.moveCartJ()
  control.moveCartL()
  control.moveCartC()
  control.paramSpeed()
  control.doTool()
  control.paramTool()
  control.paramConfig()

3. Application de planification de mouvement (Moveit)

- Lancer l'application *app_test*
  rosrun com_arm app-test
- Lancer moveit
  roslaunch rx60_moveit_config demo.launch