27/08/2020
ROLLAND
Killian

Notice de prise en main des développements

Ce document est la notice de prise en main du développement, celui-ci va se focaliser principalement sur ROS et le déploiement de l'intelligence d'une machine à une autre.

En effet, mes codes sont adaptés à une utilisation du système sur un TurtleBot3 Waffle Pi, pour voir comment implémenter ces codes sur ce robot spécifique, veuillez-vous référer au fichier *Document_deploiement_TurtleBot3.pdf* présent dans la partie *Documentation* de mon Google Drive.
(lien :https://drive.google.com/drive/folders/10cN4ByttwTIIeknRdFRT4O2PrBo6WrrW?usp=sharing).

Pour ce qui est d'utiliser ce fonctionnement sur un autre robot, quelques adaptations seraient à faire.
Premièrement la gestion de la batterie du robot ici relève de l'équipement du TurtleBot. LA tension de la batterie de ce dernier est relevée en temps réel par un capteur présent sur l'OpenCR et relevé par un code Arduino. Ainsi cette tension est ensuite mise sous forme d'information dans un topic ROS (« battery_state »), sous le nom anglais « voltage ».
Pour obtenir une gestion de la batterie comme celle du TurtleBot il faut donc avoir un système capable d'avoir la ten sion d'une batterie en temps réel et de la communiquer via un topic ROS, il suffirait ensuite de modifier les codes de simple_navigation_goals.cpp et de battery_monitoring.cpp (codes trouvables dans le Drive, rubrique *Sauvegardes/ROS_Master_PC/src*) afin d'indiquer le nom du topic qui fourni cette tension.
Dans le même ordre d'idée, il faudrait revoir le calcul de pourcentage de batterie restant (expliqué et calculé dans battery_monitoring.cpp) soit en modifiant ce calcul, soit en ayant un système qui fournit lui-même le pourcentage de batterie restante.

Une autre adaptation serait celle du cas où le robot doit se rendre à son dock. Sur le TurtleBot, le code utilisé est un code développé par Robotis spécialement pour ce robot qui permet de rejoindre un marqueur visuel. Il faudrait alors soit adapter ce code, soit changer la façon dont le robot rejoint son dock. Cela pourrait se faire par exemple par un déplacement en coordonnées (comme les autres points que le robot doit rejoindre), du type le robot se rapproche du dock, se tourne (pour présenter sa partie qui doit rentrer en contact avec le dock), puis recule jusqu'à ce que le contact s'établisse et que le robot soit bien en situation de charge.
Il faudrait aussi changer la source de la caméra ROS dans le code webui.js trouvable dans le drive : *Sauvegardes/ROS_Master_PC/src/Web_App.* En effet, à la ligne 121 de ce code, on doit indiquer le nom du topic où est publiée l'image diffusée par la caméra, pour le turtlebot c'est /raspicam_node/image, c'est donc cette partie qu'il faudrait modifie rpour un autre robot/ une autre caméra.