



Plan de développement

Projet Intelligence Artificielle

Eva DROSZEWSKI
Sara IKAN
Anesie MARTINIANI
Alix PRATABUY

Classe Actions

tourner_de(angle) : permet au robot de tourner d'un angle donné (spécifié dans les paramètres de la fonction).

ouvrir_pince : permet au robot d'ouvrir ses pinces quand il détecte un palet.

fermer_pince : permet au robot de fermer ses pinces lorsque le palet est entre les pinces (ou pas)

recherche_palet : permet au robot de rechercher un palet en effectuant un tour complet sur lui-même. Il se dirige ensuite vers la discontinuité détectée et réinitialise la valeur de l'angle.

mouvement_aleatoire : permet au robot de faire un mouvement aléatoire en choisissant une valeur d'angle de rotation et une distance à parcourir au hasard dans une liste prédéfinie.

avancer_sans_palet : permet au robot d'avancer les pinces ouvertes lorsqu'il détecte un palet.

sarreter_sans_palet : Le robot s'arrête lorsqu'il détecte un palet. Il commence à avancer en mesurant la distance. Dans le cas où la distance mesurée passe sous le seuil défini, cela signifie que l'obstacle est un mur, donc il s'arrête pour se réorienter.

attraper_palet : permet au robot d'attraper le palet lorsque la distance est en-dessous du seuil fixé, il ouvre d'abord les pinces, il s'arrête puis ferme les pinces.

tourner_vers_nord : grâce à la boussole, le robot s'oriente vers le nord pour avancer vers la ligne blanche et pouvoir déposer le palet.

avancer_avec_palet : permet au robot d'avancer en ayant le palet entre les pinces (fermées).

eviter_obstacle : lorsque la distance est inférieure à une distance seuil (distance qui correspond à celle où le palet n'est plus dans le champ de détection du capteur ultrason du robot, environ 32 cm) le robot adopte la stratégie d'évitement qui est de tourner à droite puis tourner à gauche et continuer tout droit (pour l'instant on testera un autre méthode par la suite).

avancer_de(distance) : permet au robot d'avancer d'une distance donnée (spécifiée dans le code).

sarreter_avec_palet : Lorsqu'il a le palet dans les pinces, tant qu'il n'a pas détecté la ligne blanche, grâce à **get_couleur**, mais qu'il détecte un obstacle (avec la fonction **detecter_discontinuite**), il fait une stratégie d'évitement en utilisant la fonction **eviter_obstacle**. Lorsqu'il l'a franchie, il avance de 10 cm en appelant la fonction **avancer_de(distance)**, puis il ouvre les pinces et recule de 10 cm à l'aide de la fonction **reculer_de(distance)**.

reculer_de(distance) : permet au robot de reculer d'une distance donnée (spécifiée dans le code).

Classe Capteurs

get_distance : permet de récupérer la distance qui le sépare d'un objet à un instant t.

stocker_distance : permet de stocker dans une liste toutes les valeurs de distance à intervalles de temps régulier.

detecter_discontinuite : permet de détecter une discontinuité en comparant la dernière et l'avant dernière valeur de la liste renvoyée par la fonction **stocker-distance**. Si leur écart est assez important, la fonction renvoie les deux valeurs.

get_pression : permet de récupérer la valeur de la pression à un instant t.

get_couleur : permet de récupérer la valeur de la couleur à l'instant t. Renvoie une chaîne

de caractères correspondant à la couleur détectée.

get_position : permet au robot de tourner une fois sur lui-même tout en récupérant les valeurs de distance aux murs et de trouver les deux minimums correspondant aux deux coordonnées de sa position.

get_angle_trigo : permet de récupérer l'angle de rotation du robot. Pour cela on récupère la distance au mur avant de commencer à tourner et celle une fois la rotation terminée. Ensuite on calcule avec une formule de trigonométrie et la fonction renvoie l'angle correspondant en degrés.

get_angle_points : permet également de récupérer l'angle de rotation du robot. Dans ce cas, on suppose que l'on connaît le nombre de points de mesure de distance que le robot effectue lorsqu'il fait un tour de 360°. Le calcul de l'angle se fait ici avec un produit en croix, et de même la fonction renvoie l'angle en degrés.

choisir_methode_mesure_dangle : permet de dire quelle technique il faudrait utiliser afin de mesurer l'angle. En effet, dans certains cas un objet peut se trouver face au robot avant le début de sa rotation ou à la fin de celle-ci. Dans ce cas la distance mesurée n'est pas celle au mur, et l'angle renvoyé par la méthode **get_angle_trigo** est faux. Pour choisir on compare les deux valeurs renvoyées par les fonctions. Si celles-ci sont trop différentes on privilégiera la méthode **get_angle_points** car plus fiable. Dans ce cas la méthode renvoie l'angle renvoyé par cette méthode et le chiffre 2 pour signaler que la méthode utilisée est **get_angle_points**. Dans le cas contraire, la première étant plus précise, la fonction renvoie l'angle déterminé par la méthode et le chiffre 1 pour signaler que la méthode utilisée est **get_angle_trigo**.

get_orientation : permet de mettre à jour l'orientation du robot en ajoutant l'angle de rotation et renvoie la valeur actualisée.

stocker_orientation : permet de stocker dans un tableau toutes les valeurs d'orientation.

Classe Main

boussole : permet de définir le nord, sud, est et ouest en fonction des différentes valeurs d'angle.

programme général : code principal permettant de traduire les choix effectués par le robot en fonction de la perception de son environnement

[illegible]