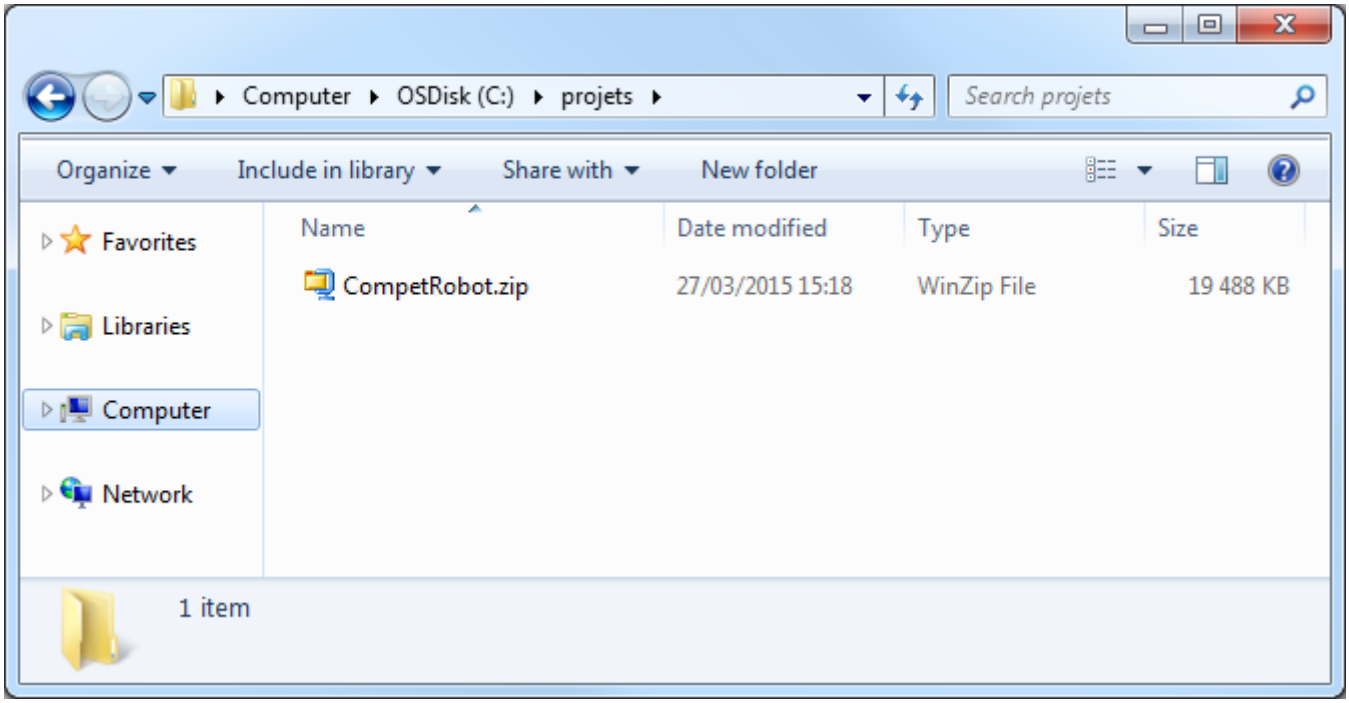
Mission on Mars Robot Challenge : Prise en main

Récupération du projet

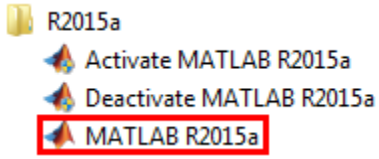
Le projet servant de base à la compétition se trouve sur le site internet MATLAB central. Téléchargez l'archive CompetRobot.zip contenant le projet avec les éléments techniques dont vous aurez besoin (modèle de simulation du robot, documentation…). Une fois l'archive téléchargée, copiez-la dans le répertoire de votre choix. Dans les captures d'écran de ce document, ce répertoire est `C:\projets` mais vous pouvez utiliser le dossier qui vous convient.



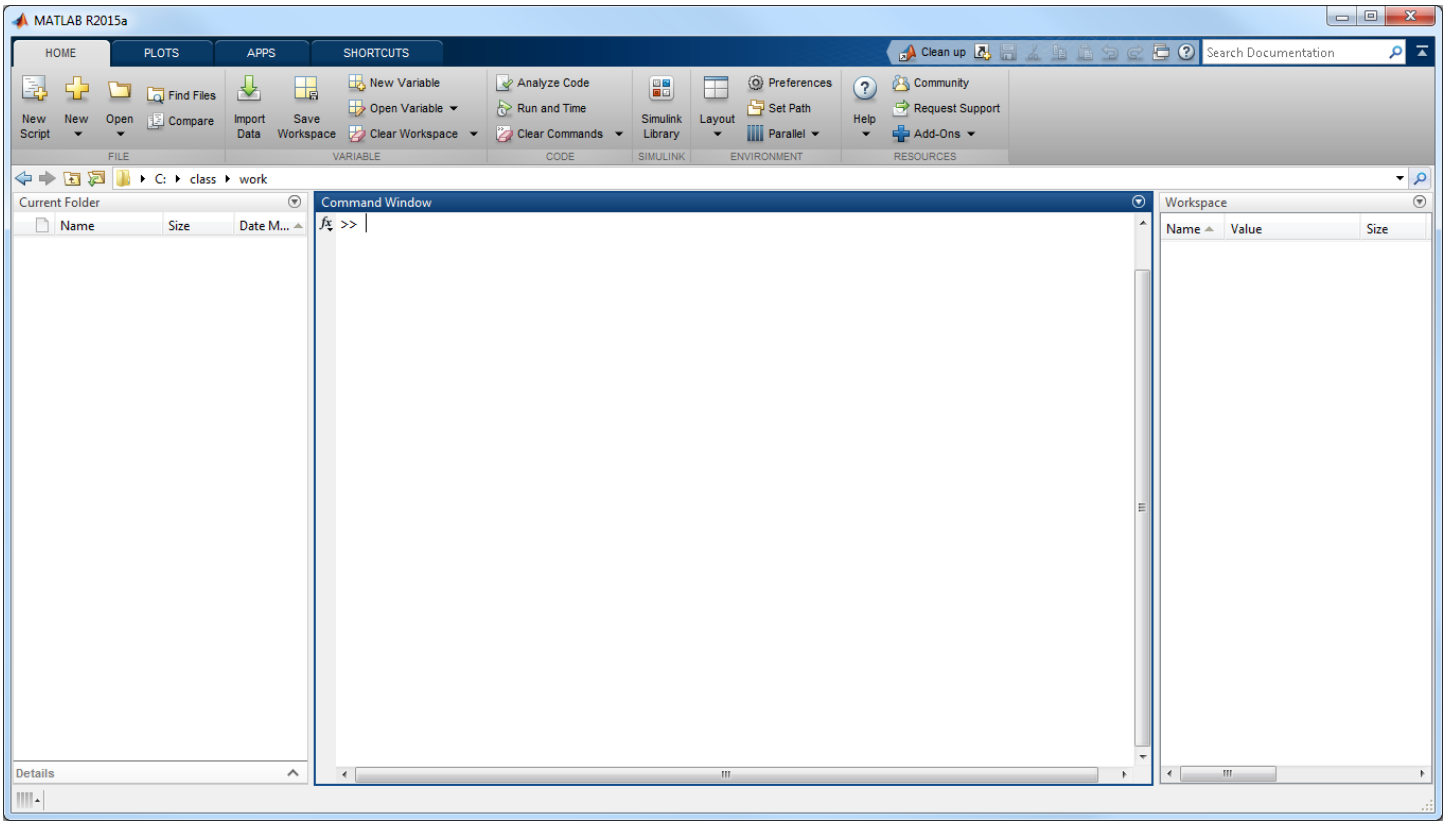
Lancement de MATLAB

Une fois installé, MATLAB se lance comme tous les logiciels. Vous pouvez par exemple aller dans le menu démarrer, puis cliquer sur « Tous les programmes », et enfin sélectionner le répertoire dans lequel vous avez installé MATLAB.

Vous devriez alors voir un sous-répertoire R2015a contenant l'exécutable MATLAB R2015a qui sert à démarrer MATLAB :

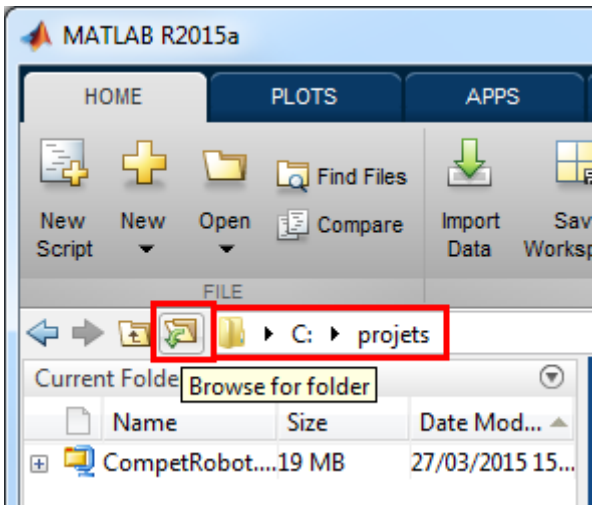


Une fois que vous aurez cliqué sur cette icône, le « splash screen » (ou fenêtre de chargement) de MATLAB devrait s'afficher pendant quelques minutes, le temps que le logiciel démarre. Puis, vous devriez voir apparaître la fenêtre d'interface MATLAB ressemblant à ceci :

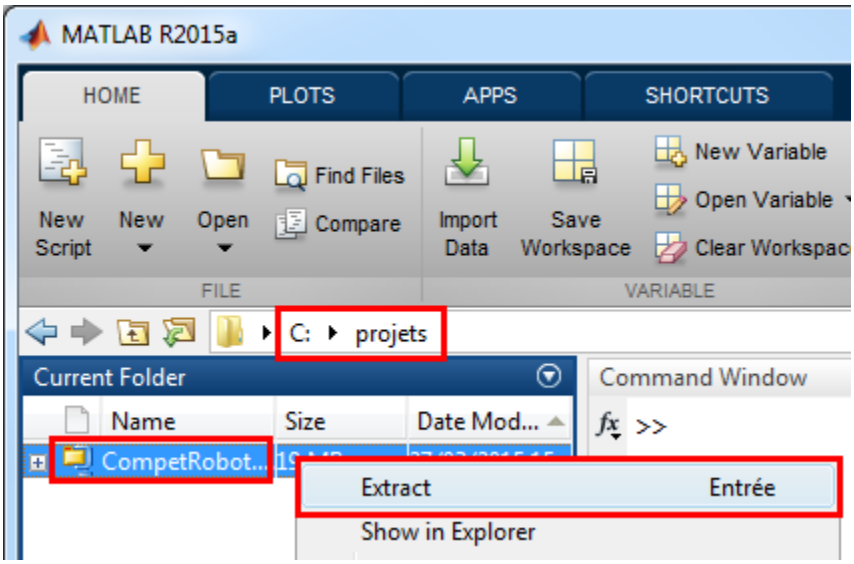


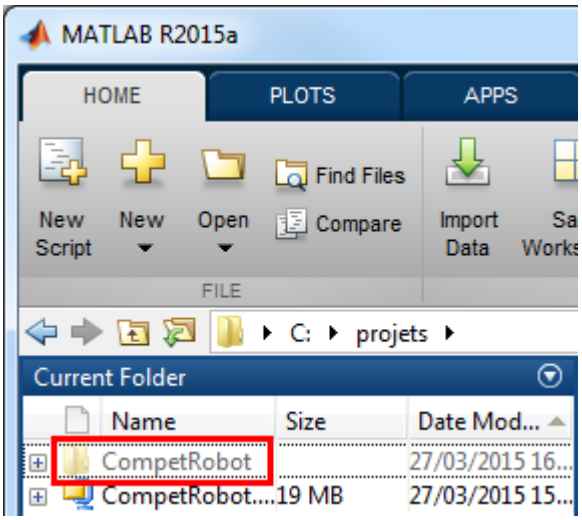
Extraction du projet Simulink

Une fois que MATLAB est correctement démarré, placez-vous dans le répertoire où vous avez copié l'archive `RobotCompet.zip`, à l'aide par exemple du bouton indiqué ci-dessous :



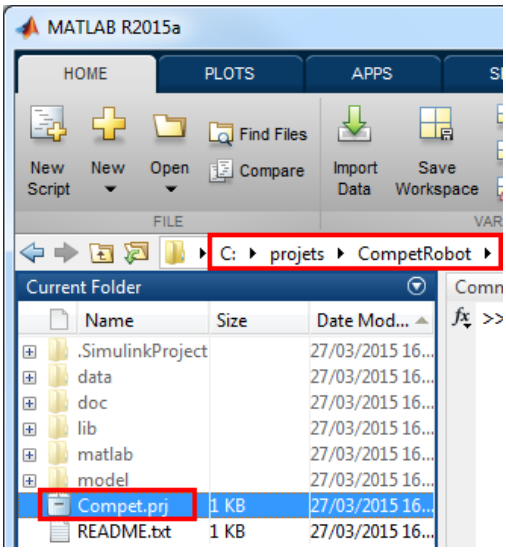
Une fois que vous êtes dans le bon répertoire, vous devriez voir dans la fenêtre `Current Folder` de MATLAB l'archive `CompetRobot.zip` que vous avez copié antérieurement. Décompressez alors l'archive en effectuant un clic-droit et en sélectionnant `Extract` dans le menu contextuel, ce qui devrait faire apparaître le dossier `CompetRobot`, comme vous pouvez le voir ci-dessous :



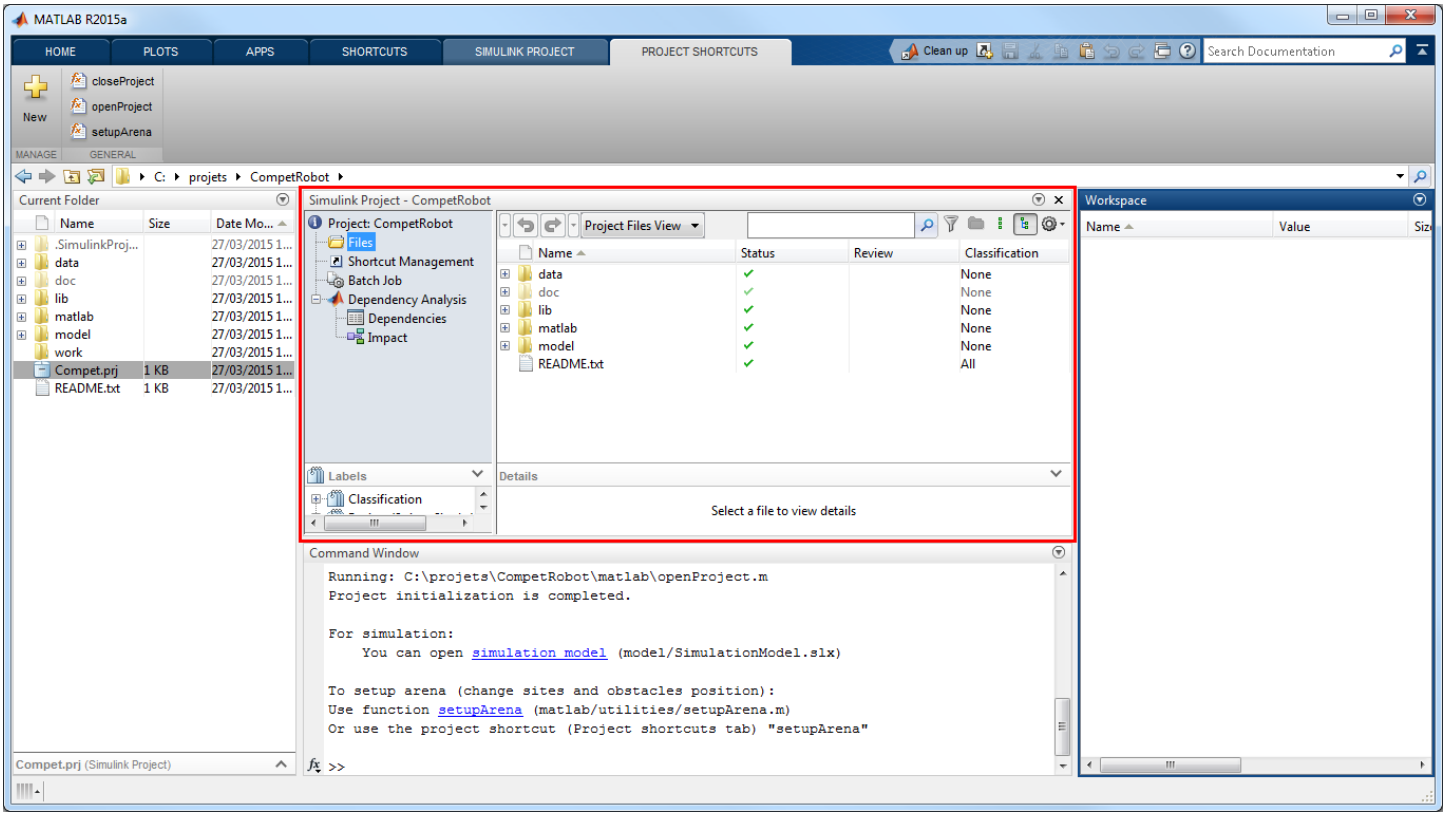


Ouverture du projet Simulink

Une fois le dossier CompetRobot décompressé, ouvrez ce dossier, puis double-cliquez sur le fichier de projet Simulink `Compet.prj` pour l'ouvrir :



La fenêtre du projet Simulink `CompetRobot` devrait alors s'ouvrir dans l'interface MATLAB. Dans notre exemple, elle s'ouvre au milieu de l'interface mais ce ne sera pas forcément son emplacement chez vous :



Une fois le projet ouvert, vous allez alors pouvoir faire deux choses :

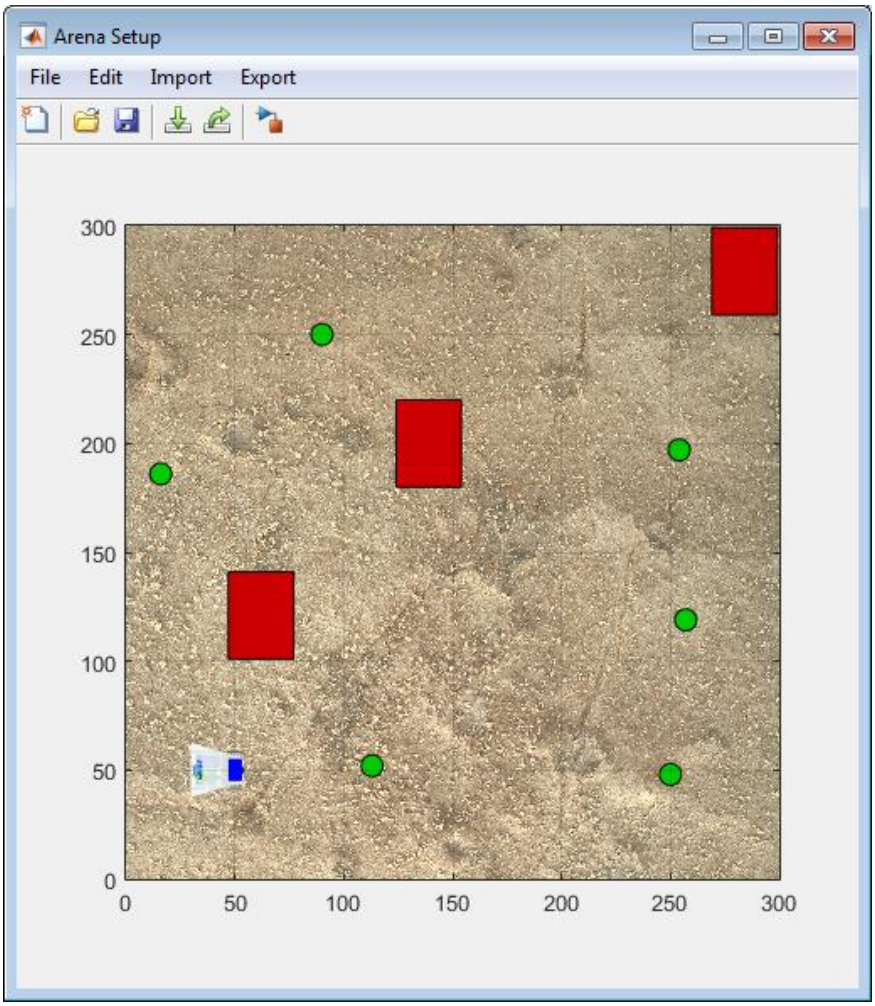
- Gérer différentes configurations de l'arène dans laquelle le robot va évoluer,
- Améliorer l'algorithme de contrôle du robot dans le modèle Simulink.

Ces deux étapes sont décrites dans la suite de ce document.

Gestion des configurations de l'arène

Un outil vous est fourni pour pouvoir facilement modifier la position des sites et des obstacles dans l'arène.

En voici un aperçu :



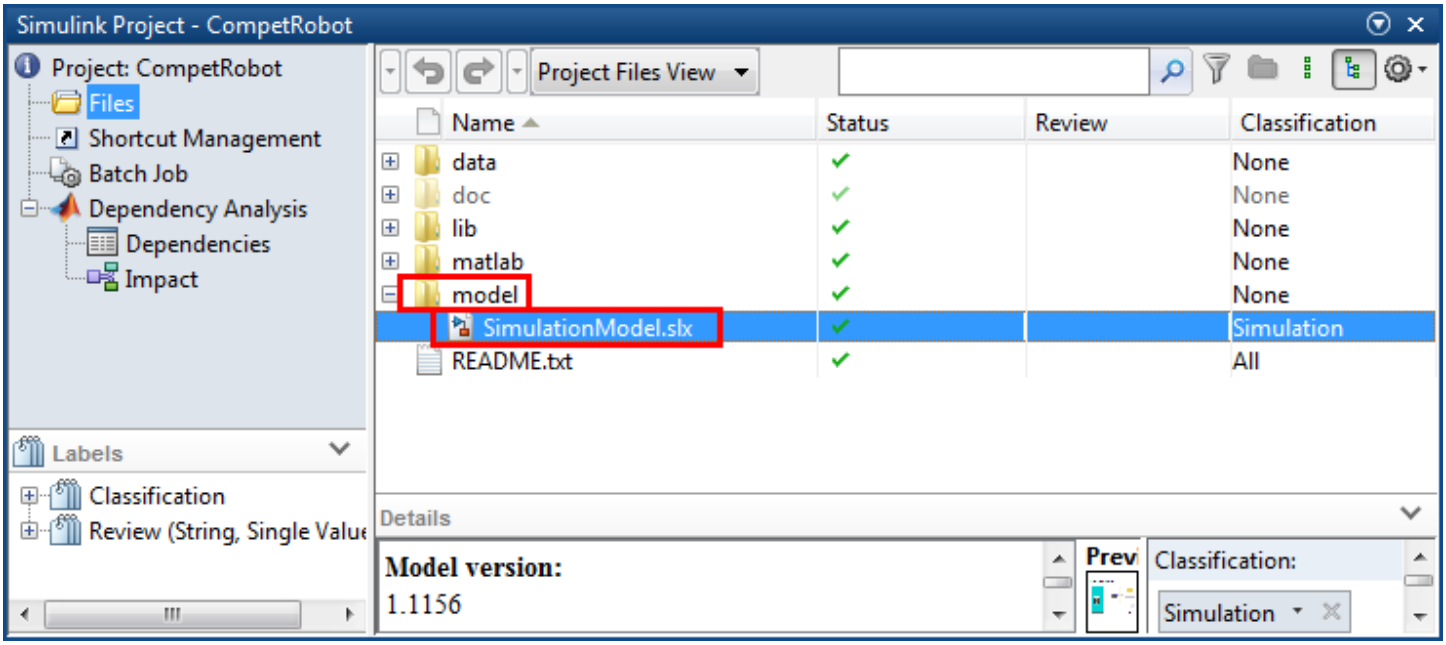
Pour gérer les configurations de l'arène, il suffit d'utiliser le raccourci `setupArena` (visible dans le bandeau d'outils, onglet `Project Shortcuts`).

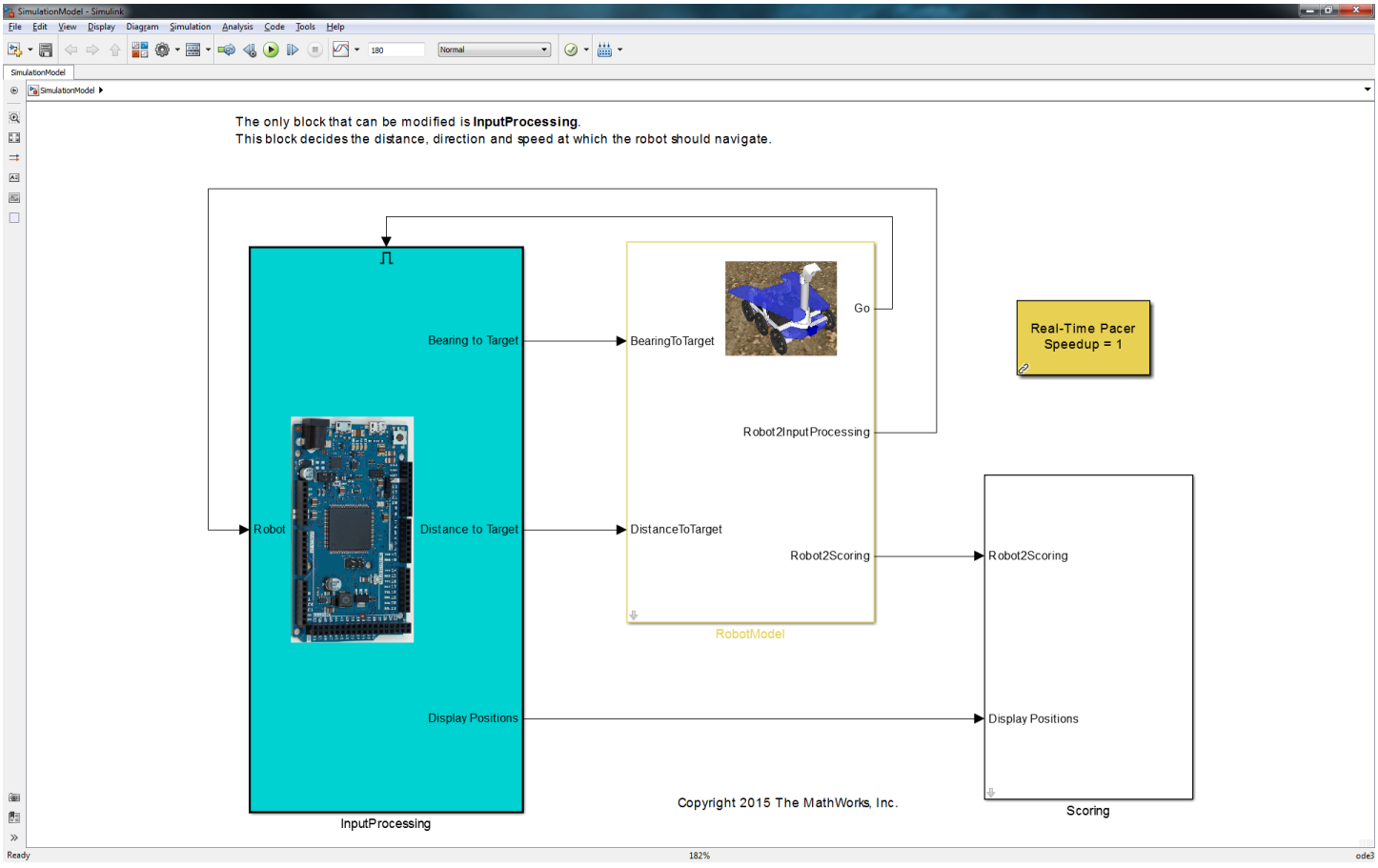
Une documentation complète est disponible dans le répertoire Doc\SetupArena.html .

Nous vous rappelons que vous ne connaîtrez pas les emplacements des sites et des obstacles le jour de la compétition.

Ouverture du modèle de simulation du robot

Le modèle de simulation se trouve dans le répertoire `model`, et se nomme `SimulationModel.slx`. Vous pouvez double-cliquer dessus pour ouvrir le modèle dans l'interface Simulink.





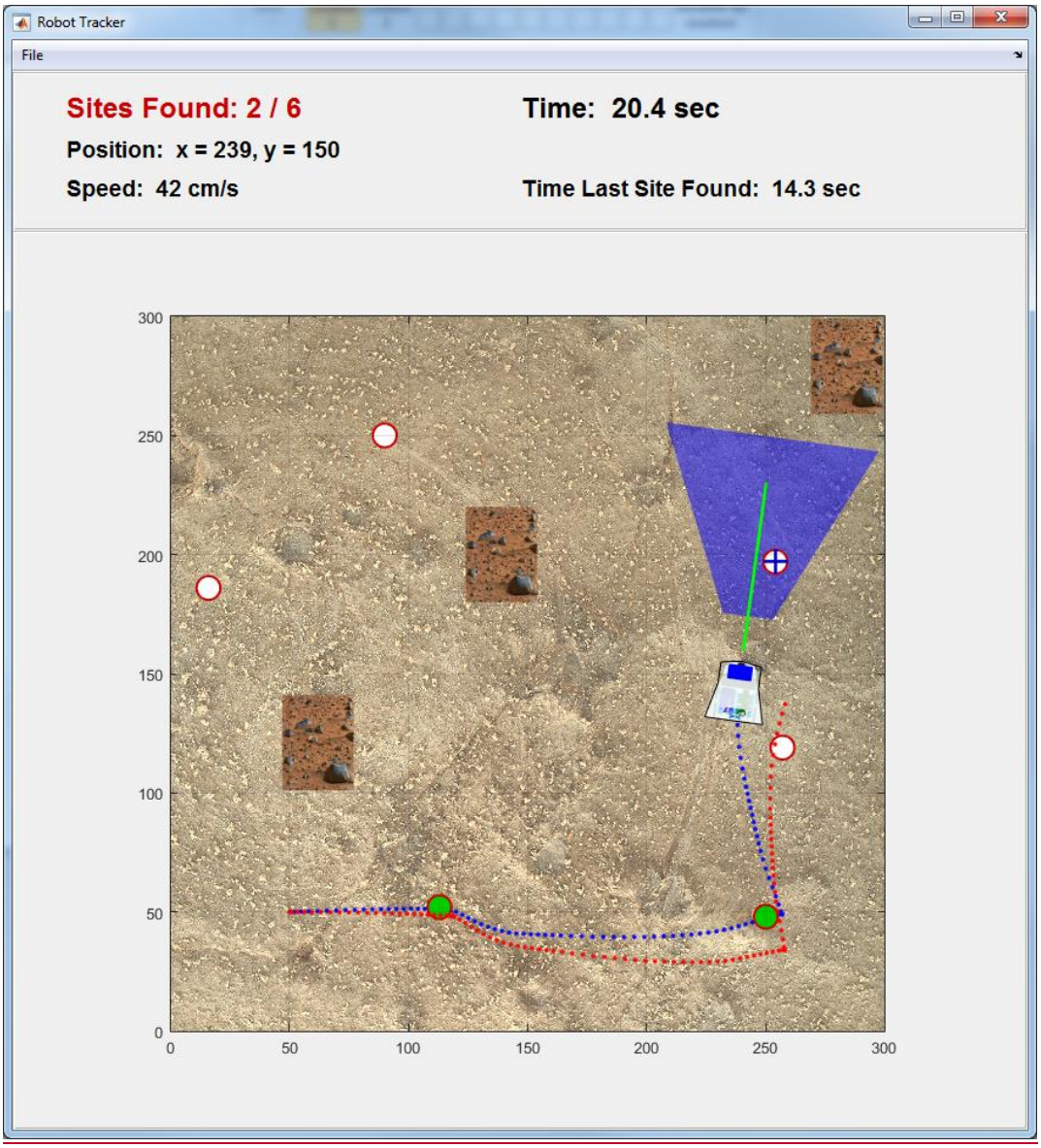
Cette fenêtre contient le modèle de simulation.

Lancer la simulation

Avant d'examiner et de modifier le modèle, il est important de comprendre comment lancer la simulation, afin de pouvoir observer les résultats de ses modifications ou expérimentations.

Pour lancer la simulation, il suffit de cliquer sur le bouton « Run » illustré dans la figure ci-dessous. Ce bouton vérifie les éventuelles erreurs dans le modèle, et lance la simulation dans le cas où aucune erreur n'est rencontrée.

Aussi après quelques instants correspondant à la vérification et la compilation du modèle, la fenêtre suivante devrait apparaitre.



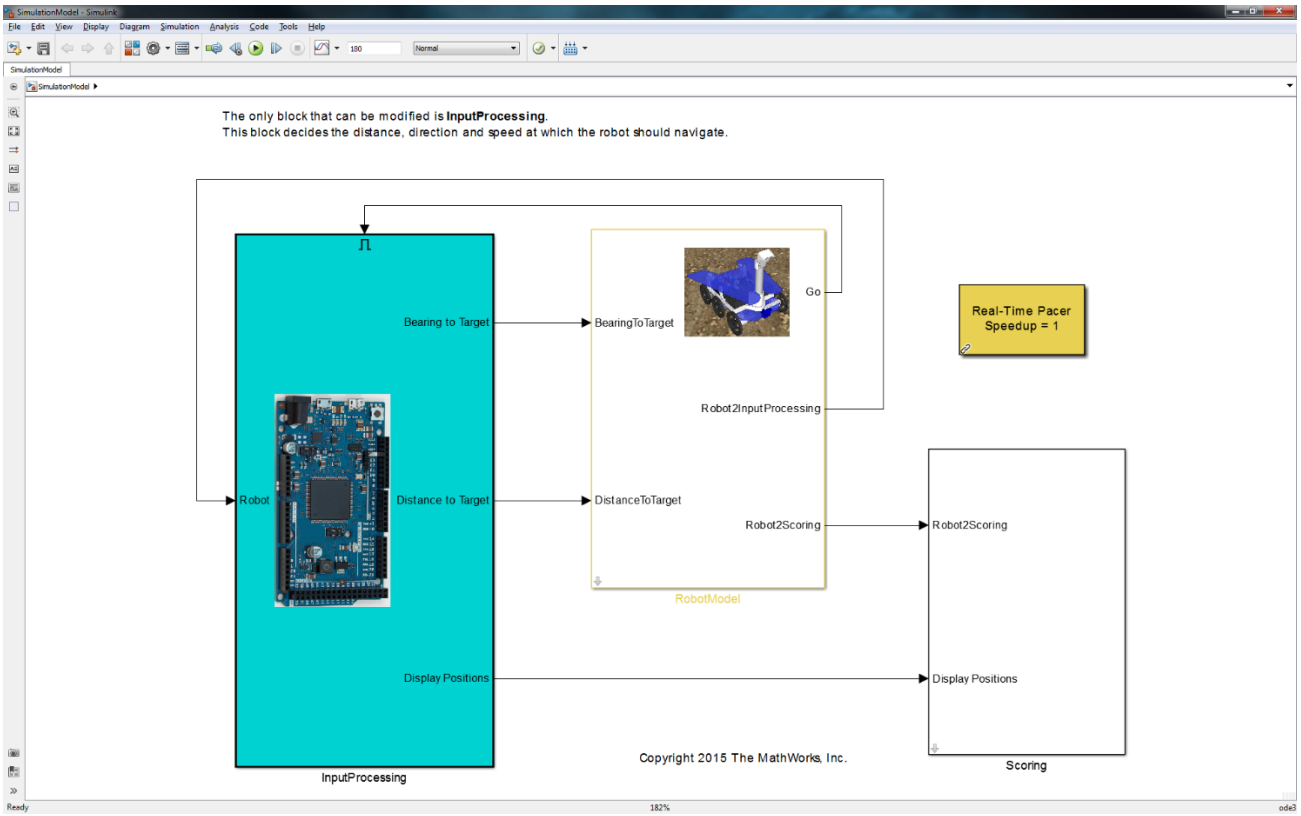
Description de la fenêtre de simulation:

- Les sites à visiter sont représentés par des cercles :
    - Les sites sont blancs lorsqu'ils n'ont pas été trouvés par le robot.
    - Ils sont verts lorsqu'ils ont été trouvés par le robot et que ce dernier s'est arrêté au-dessus pendant au moins 3 secondes.
- Les obstacles sont représentés par des pierres. Si le robot touche l'un des obstacles, la mission s'arrête…
- La trajectoire du robot est représentée par :
    - Le tracé pointillé bleu correspond au déplacement réel du robot sur l'arène.
    - Le tracé en pointillé rouge correspond à la position à laquelle le robot pense se situer en utilisant les encodeurs sur les roues. Les deux tracés ne sont pas confondus car le robot à tendance à glisser sur la surface.
- Le quadrilatère bleu devant le robot représente la zone de vision de la caméra embarquée. Lorsqu'un ou plusieurs sites sont vus par la caméra, leurs distances et angles sont disponibles pour traitement.
- La ligne devant le robot correspond à la position du capteur de distance. Elle est verte si aucun obstacle n'est détecté, c'est-à-dire qu'il n'y a pas d'obstacle entre 10 et 80cm. Si un objet se présente dans cet intervalle, elle devient rouge avec affichage de la distance mesurée. Le capteur de distance fonctionne comme un radar et fait en permanence des allers-retours pour mesurer son environnement.
- Cette fenêtre indique également :
    - Le nombre de sites trouvés par le robot.
    - La position actuelle du robot ainsi que sa vitesse.
    - Le temps écoulé depuis le départ du robot.
    - Le temps écoule entre le départ et le dernier site visité.

Vous pouvez bien sûr modifier le modèle et le simuler autant de fois que vous le souhaitez tout en observant immédiatement l'effet de vos modifications sur le comportement du robot.

Vue d'ensemble du modèle

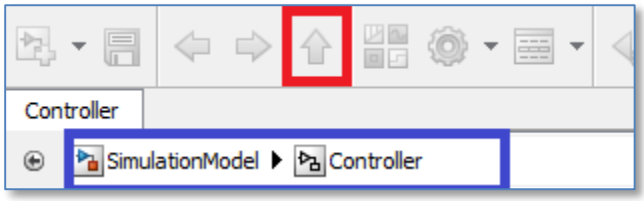
La figure ci-dessous représente le modèle de simulation du système tel qu'il apparait lorsque vous l'ouvrez. Il comprend à la fois le programme qui fonctionnera sur le robot, et une modélisation de l'environnement.



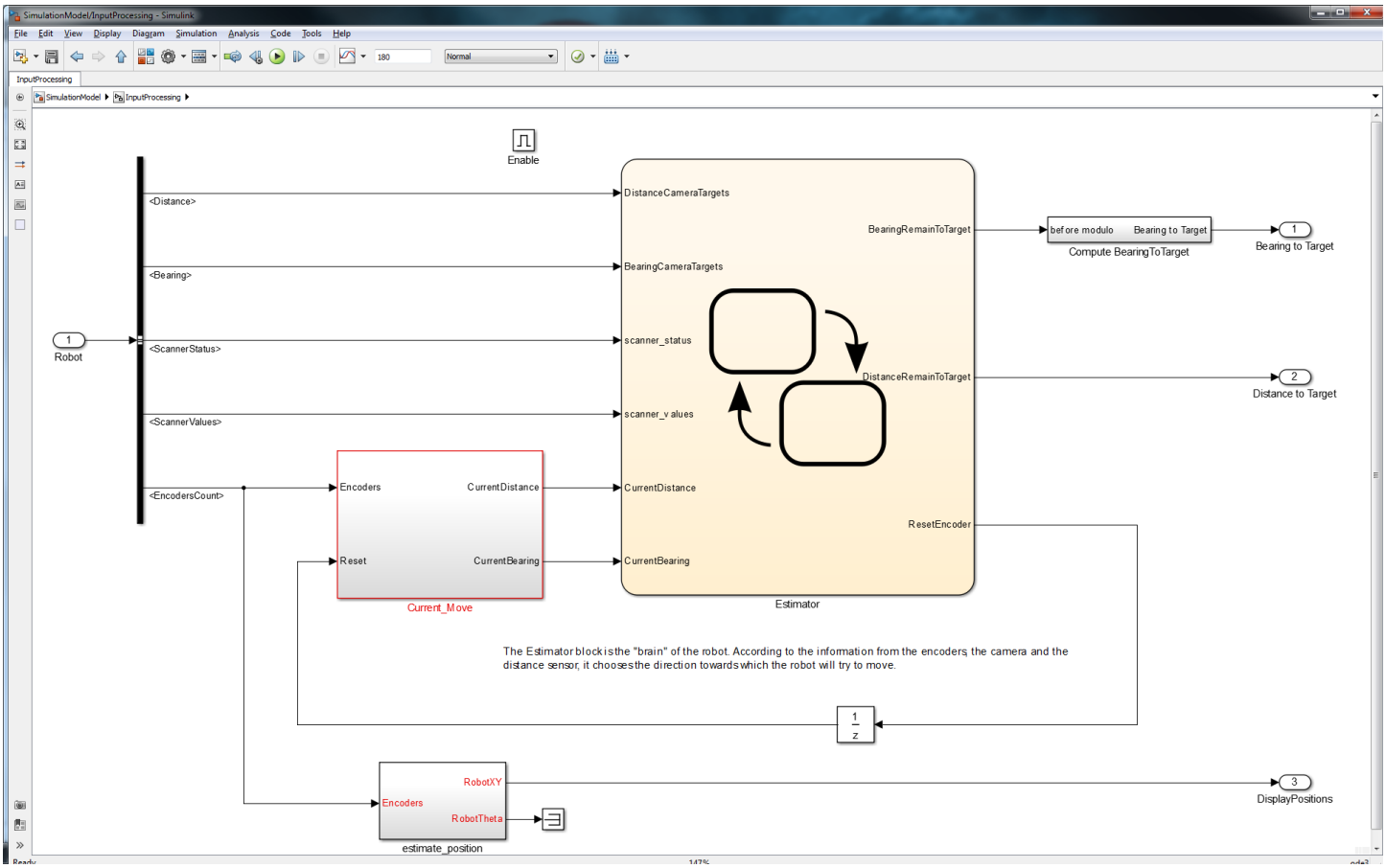
Ce modèle se compose de 3 grandes parties :

1. **Le bloc bleu « InputProcessing »** : Il implémente la stratégie de déplacement du robot. Etant données la position estimée du robot, les informations de la caméra et les distances mesurées, ce bloc permet de décider de la direction et de la distance que le robot va emprunter. C'est dans ce bloc uniquement que vous devrez apporter vos améliorations.
2. **Le bloc « Robot Model »** : Il modélise le comportement physique du robot ainsi que ses capteurs. Ce bloc ne doit pas être modifié.
3. **Le bloc « Scoring »** : Il modélise le juge électronique et gère la fenêtre de simulation graphique. Ce bloc ne doit pas être modifié.

Pour examiner le contenu des blocs, il suffit de double-cliquer dessus. Lorsqu'un sous-bloc est ouvert, il est possible de remonter au niveau supérieur à l'aide de la flèche haut de la barre d'outils (en rouge sur la figure suivante), ou de la barre de navigation (en bleu sur la figure suivante).



Le bloc « InputProcessing »



La logique de parcours du robot se situe dans le bloc « Estimator » qui est un diagramme d'état Stateflow. Le produit MathWorks Stateflow est un environnement de modélisation et de simulation de logique de décision à partir de machines d'état.

Ce bloc prend en entrée :

- Les informations fournies par la caméra : un tableau contenant les distances et angles des sites visibles depuis la caméra par rapport à l'avant du robot.
- Les informations des encodeurs situés sur les 2 roues avant du robot.
- La position et les distances mesurées par le capteur de distance.

Les informations des encodeurs (rotation des roues gauches et droites) sont utilisées pour estimer la position actuelle du robot dans les blocs « Current_Move » et « Estimate_Position ». Il faut cependant garder à l'esprit qu'il s'agit d'une estimation. Ainsi, si la taille réelle des roues n'est pas exactement égale à la taille des roues estimées, ou si les roues glissent, cette estimation va présenter des erreurs qui vont s'accumuler.

La logique implémentée dans le modèle qui vous est fourni est simple :

- Si la caméra détecte un site, le robot naviguera "à vue" vers celui-ci.
- Si aucun site n'est présent dans le champ de vision de la caméra, le robot se déplace en ligne droite.
- Si lors d'un déplacement, le capteur de distance détecte un obstacle à droite ou à gauche, une correction de la trajectoire dans le sens opposé est appliquée. Si le capteur de distance détecte un obstacle devant lui alors le robot s'arrête et tourne jusqu'à ce qu'il trouve un espace sans obstacle puis repart.

Le robot

Le robot « Mars Rover » avec lequel vous allez concourir sera fourni par MathWorks le jour de la finale de la compétition, le 3 Juillet. Vous ne pouvez pas concourir avec votre propre robot.
Chaque équipe finaliste testera pour la 1ère fois son modèle sur un robot le jour de la compétition.

C'est un robot type 3 roues avec 2 roues motrices et une roue libre. Ce qui lui permet de se déplacer en ligne droite mais également de tourner sur lui-même.
Le robot Mars Rover mesure 24 cm de large, 24 cm de long pour une hauteur de 22cm.

Le robot est composé des éléments suivants :

- Une carte Arduino DUE
  - C'est sur cette carte que vos algorithmes vont être utilisés.
  - La carte Arduino DUE est en charge de la navigation et de la stratégie de déplacement vers les sites.
  - Elle est équipée d'une carte fille pour la partie puissance moteur : DFRobot motor Shield.
- Une carte Raspberry Pi
  - A partir du flux d'images de la caméra connectée, cette carte va identifier la présence ou non d'un ou plusieurs sites dans son champ de vision. Ceux-ci sont représentés par des marqueurs de couleur au sol.
  - Après des calculs, elle va déterminer la distance en centimètre et l'angle (0° en face, angle positif à gauche et angle négatif à droite) des sites détectés.
  - Ces informations sont ensuite envoyées à la carte Arduino sous forme d'un tableau par I^2C.
  - Il n'y a aucune modification à apporter à cet ensemble qui vous est fourni.
- Une Webcam
  - Celle-ci est reliée directement au Raspberry Pi pour la partie vision.
  - Le flux d'image est continu mais le temps de traitement est d'environ 200-300ms.
- Deux moteurs à courant continu
  - Ces moteurs sont équipés de roues de 70mm de diamètre.
  - Ils sont également équipés d'encodeurs avec une résolution de 636 pas par tour de roue.
  - La distance entre les roues est 165mm.
  - Avec tous ces renseignements, la carte Arduino est capable d'estimer la distance parcourue et l'angle du robot.
  - Il est à noter que les roues peuvent glisser et que le robot n'est pas parfait.
- Un capteur de distance
  - Celui-ci est monté sur un servomoteur qui permet de le faire tourner. Les mesures s'effectuent en continue : centre, droite, gauche, centre,... Le temps entre 2 mesures varie entre 200 et 400ms.
  - Ce capteur fourni une information de position angulaire qui correspond à la dernière valeur lue et un tableau contenant les 3 valeurs de distances.
  - Il fournit une information de distance en centimètre des obstacles et du bord de l'arène.
  - La plage de détection est 10 à 80cm.
- Une batterie de 6000mAH permettant une journée complète de fonctionnement

La planète Mars est représentée par une arène de 3 mètres sur 3 mètres. Nous allons placer dans cette arène :

- Des points verts : ils représentent les sites sur Mars sur lesquels vous devez passer et vous arrêter 3 secondes pour valider leur exploration.
- Des obstacles à éviter.

Conclusion

Le modèle fourni est une base de départ que vous devrez améliorer.

Voici quelques pistes qui s'offrent à vous pour modifier le modèle, de la plus simple à la plus complexe:

- Optimiser les paramètres de vitesse du robot.
- Améliorer l'algorithme de recherche existant pour qu'ils trouvent plus de sites en prenant par exemple en compte des délais de la caméra dans les déplacements.
- Créer un nouvel algorithme de navigation qui se sert de la position estimée absolue du robot pour construire une carte de son environnement.

Mais bien sûr, vous êtes libres d'inventer d'autres solutions. Gardez cependant à l'esprit que la puissance de calcul du robot est très limitée, et que des calculs complexes risquent de prendre trop de temps. Ainsi, il est recommandé de ne pas utiliser de nombres en double précision ou un trop grand nombre d'opérations trigonométriques.