Cette bibliothèque *DarwinOP library for Matlab Simulink* propose des blocs Matlab Simulink pour créer des modèles interagissant avant le robot DarwinOP de Robotis[1]. Ces blocs permettent de commander et de contrôler les mouvements faits par le robot ainsi que d'obtenir un retour sur son état actuel.

Deux modes de fonctionnement sont possibles :

- La simulation logicielle (« software in the loop ») permet l'exécution du modèle Matlab Simulink sur le PC avec envoi/réception des informations vers/depuis le robot via une liaison TCP/IP (Ethernet filaire ou wifi) ;
- La programmation matérielle (« hardware in the loop ») génère un programme en C++ à partir du modèle Matlab Simulink et lance sa compilation et son exécution directement sur le robot.

1. Historique

Cette version 1.0 a été développée et testée avec Matlab R2013a. Elle nécessite sur le robot la présence de la bibliothèque darwinop-ens[2] en remplacement de la bibliothèque officielle darwinop[3] préinstallée sur le robot. La procédure de remplacement nécessite plusieurs étapes[4].

Contacter l'auteur : ce logiciel est encore en phase de développement. Si vous obtenez un comportement non désiré ou que vous avez des problèmes pour mettre en œuvre les blocs de cette bibliothèque, n'hésitez pas à me contacter sur mon adresse email florent.ouchet@ens-rennes.fr.

[1] http://www.robotis.com/xe/darwin_en
[2] https://github.com/darwinop-ens/darwin-op
[3] http://sourceforge.net/projects/darwinop/
[4] https://github.com/darwinop-ens/darwin-op/wiki/Hack-guide

2. Guide d'installation

L'installation de la bibliothèque *DarwinOP library pour Matlab Simulink* requiert peu d'opérations sur le PC. Ces opérations doivent être exécutées en mode « administrateur », si vous n'avez pas accès au compte « administrateur », vous ne pourrez pas utiliser les blocs dans Matlab Simulink.

La première étape est de télécharger l'archive[5] et d'en extraire le contenu sur votre disque local, par exemple dans `c:\darwinop-ens\Simulink`. **Il est recommandé de ne pas extraire ces fichiers dans le répertoire de Matlab. Une fois la bibliothèque installée, il ne faut pas supprimer ces fichiers.**

Après avoir lancé Matlab sur l'ordinateur, ouvrir et exécuter le script `c:\darwinop-ens\Simulink\darwinoplib\darwinoplib_setup.m`. Ce script recense les chemins de la bibliothèque dans les chemins à parcourir de Matlab. Si votre utilisateur ne dispose pas des privilèges « administrateur », le script vous demandera de vous connecter avec le compte « administrateur ». Cette première opération se termine normalement par l'affichage du message :

```
DarwinOP Target Path Setup Complete.
```

L'étape suivante consiste en la sélection de la langue pour l'interface graphique, actuellement seuls l'anglais et le français sont supportés :

```
Select library language:

  [1] english

  [2] français

1-2:
```

Vous pouvez sélectionner la langue en sélectionnant « 1 » ou « 2 » puis en validant par « entrée ».

Pour changer la langue ultérieurement, vous pouvez exécuter à nouveau ce script.

Il n'existe pas actuellement de procédure de désinstallation de la bibliothèque, une désinstallation manuelle est possible en supprimant toutes les entrées commençant par `c:\darwinop-ens\Simulink` dans l'utilitaire « Set Path » de Matlab (Figure 1 et Figure 2 ci-dessous).

Figure 1 Utilitaire "Set Path" de Matlab

[5] https://github.com/darwinop-ens/simulink/archive/master.zip

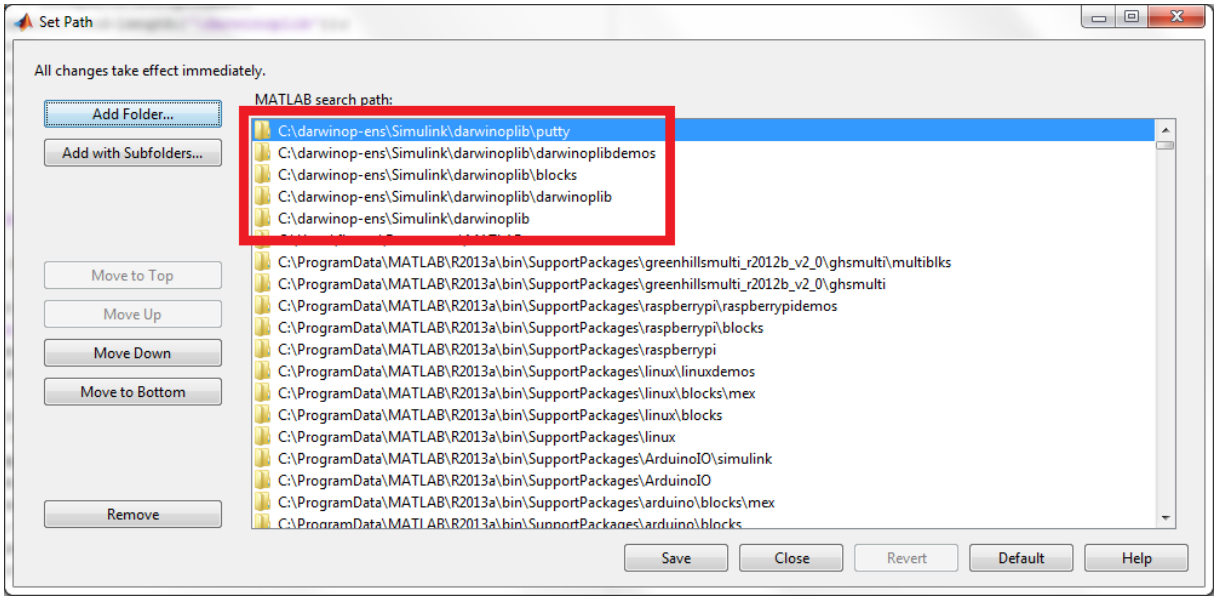

Figure 2 Chemins relatifs à la bibliothèque

3. Création d'un modèle

a) Présentation des blocs de la bibliothèque

Trois nouveaux blocs sont installés dans cette bibliothèque (Figure 3) :

- « DarwinOP communication » est le bloc principal gérant la communication avec les actionneurs du robot. Il est recommandé de n'utiliser qu'un seul de ces blocs lors des simulations car plusieurs blocs impliquent des latences plus élevées et augmentent le temps d'échantillonnage minimal pour la simulation. En mode génération de code, vous pouvez utiliser plusieurs blocs sans pénalité sur le temps d'exécution ;
- « Real-time simulation » permet la synchronisation de l'horloge de simulation avec l'horloge de l'ordinateur. Ce bloc n'est utile que lors des simulations et il ne faut en placer qu'un sur le modèle. Ce bloc n'a aucune fonctionnalité lors de la programmation matérielle et peut être conservé sur le modèle ;
- « 12-bit signed conversion » permet la représentation naturelle des nombres renvoyés par le bloc « DarwinOP communication » principalement pour toutes les mesures de vitesses de rotation et de charge sur les actionneurs MX-28.

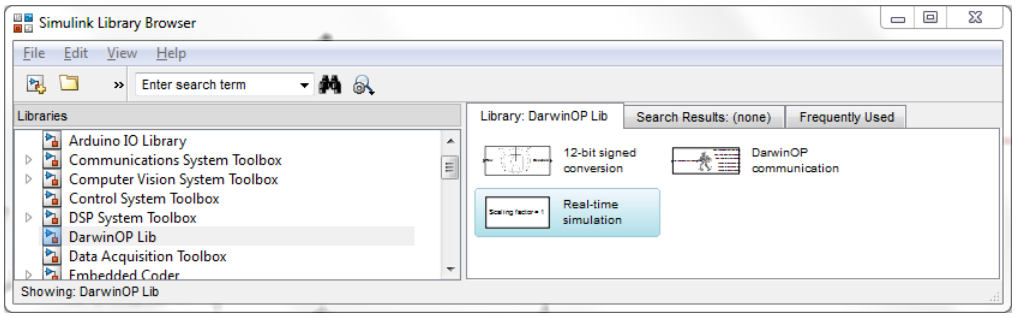

Figure 3 blocs installés par la bibliothèque DarwinOP Lib

b) Création et paramétrage d'un modèle

La création d'un modèle se base sur une feuille de dessin Simulink vierge. Simulink peut être lancé à l'aide de l'icône dans la barre d'outils de Matlab ou en exécutant la commande `simulink`. Une feuille vierge peut être créée à l'aide du menu « File/New/Model ».

La première opération est la modification des paramètres du modèle, accessibles depuis le menu « Simulation/Model Configuration Parameters ». Il faut accéder aux paramètres « Code Generation » et changer le « System target file » en « darwinoplib.tlc » à l'aide du bouton « Browse » (Figure 4).

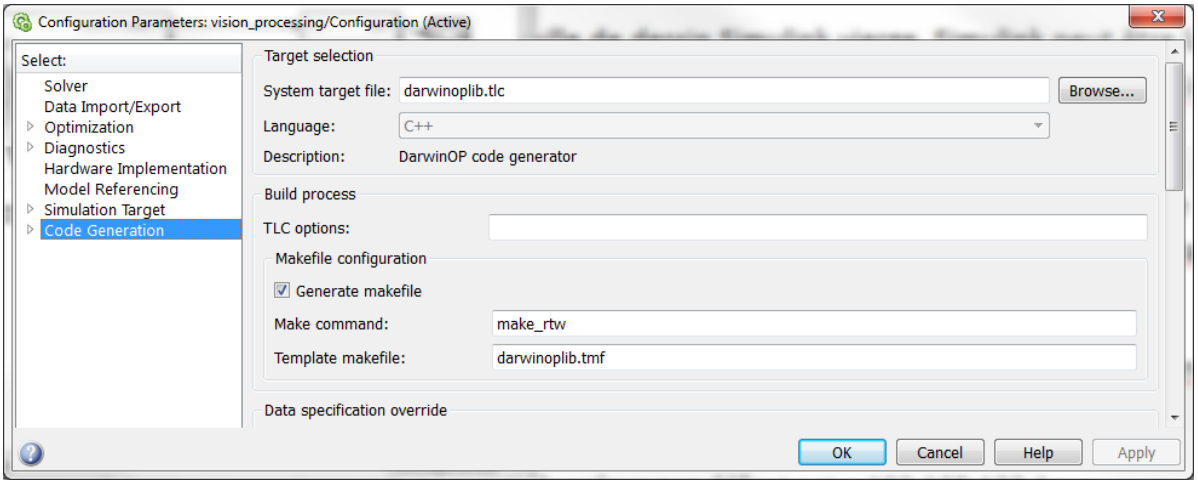

Figure 4: Paramétrage du générateur de code

Une nouvelle catégorie d'options est automatiquement ajoutée dans « Code Generation/DarwinOP options ». Elle permet le paramétrage des options spécifiques au robot (Figure 5).

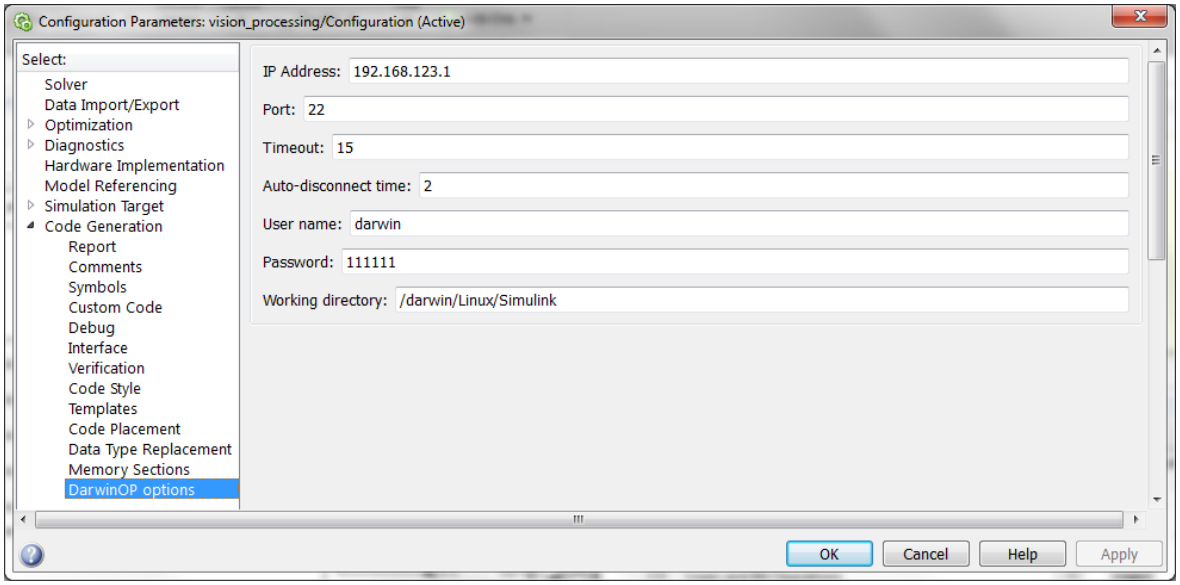

Figure 5: Paramétrage des options de connexion au robot

Les valeurs par défaut correspondent à la configuration d'origine du robot, elles sont à adapter si besoin à la configuration courante de votre robot :

- « IP Address » : adresse IP du robot, la valeur par défaut est « 192.168.123.1 » ;
- « Port » : port de connexion au robot lors de la programmation matérielle, ce port correspond au serveur SSH présent nativement sur le robot sur le port « 22 » ;
- « Timeout » : temps limite de connexion au serveur et d'exécution des commandes, normalement « 15 » secondes suffisent dans la plupart des cas d'utilisation ;
- « Auto-disconnect time » : après le lancement du programme sur le robot, temps pendant lequel l'état du robot est affiché dans la console Matlab et temps à partir duquel vous pouvez à nouveau utiliser pleinement Matlab. « 2 » secondes permettent au programme de démarrer ;

- « User name » : nom d'utilisateur pour se connecter au serveur SSH du robot, par défaut le nom d'utilisateur employé est « darwin ». Ce nom est sensible à la casse ;
- « Password » : mot de passe utilisé pour se connecter au serveur SSH du robot, le mot de passe par défaut est « 111111 » ;
- « Working directory » : répertoire de travail sur le robot, par défaut « /darwin/Linux/Simulink » (sensible à la casse comme tout nom de fichier sous Linux). Attention, le changement de ce répertoire requiert des adaptations dans les fichiers générés par la bibliothèque.

Le paramétrage suivant est dans la catégorie « Solver » toujours dans la configuration du modèle (Figure 6). Le solveur utilisé doit être du type « Fixed-step » et le « Solver » utilisé doit être « discrete (no continuous states) ». La valeur du paramètre « Fixed-step size » est le temps d'échantillonnage du modèle. Elle est à ajuster en fonction des besoins mais est contrainte par des limitations matérielles :

- En mode « simulation », les latences et le jitter sur le réseau ethernet rendent très aléatoires les simulations dont le temps d'échantillonnage est inférieur à « 0.02 » secondes, ce temps limite est multiplié par le nombre de blocs « DarwinOP communication » présents dans le modèle ;
- En mode « programmation matérielle », la limite est imposée par la complexité du modèle et par le bus Dynamixel aux alentours de quelques millisecondes en fonction du volume d'informations échangées.

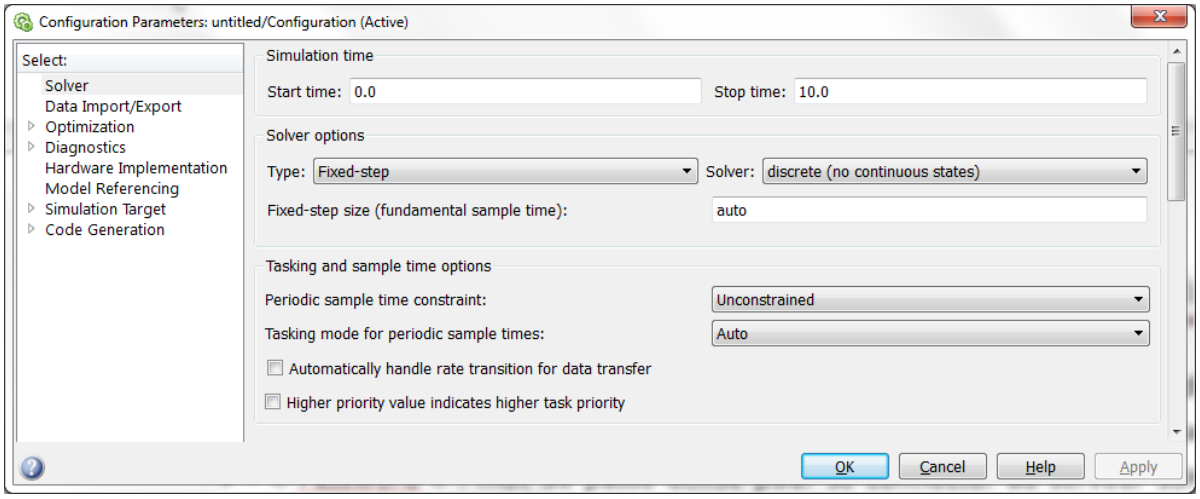

Figure 6 Configuration du solveur

Vous pouvez ensuite déposer les composants « DarwinOP communication » et « Real-time simulation » sur la feuille de dessin pour obtenir le modèle de la Figure 7. Un seul bloc « Real-time simulation » est nécessaire. Il est possible de placer plusieurs blocs « DarwinOP communication » si nécessaire (par exemple pour simuler avec deux robots ou pour avoir plusieurs fréquences d'échantillonnage) bien que cela ne soit pas recommandé car chaque nouveau bloc « DarwinOP communication » augmente la limite inférieure sur le temps d'échantillonnage en mode « simulation ».

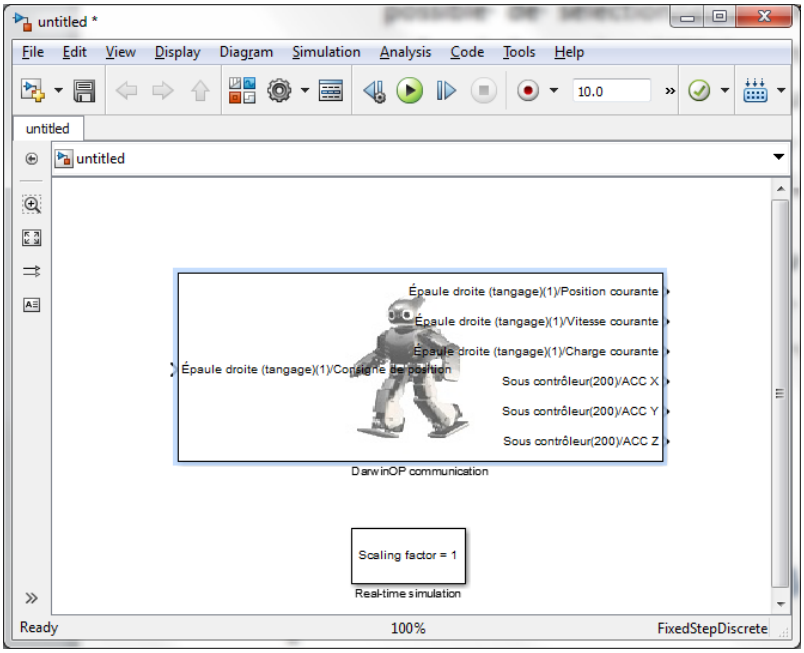

Figure 7 Début de modèle avec les blocs DarwinOP

c) Paramétrage du bloc « DarwinOP communication »

Le bloc « DarwinOP communication » peut être reconfiguré en fonction des entrées/sorties désirées. La fenêtre de configuration de la Figure 8 est obtenue en double-cliquant sur le composant.

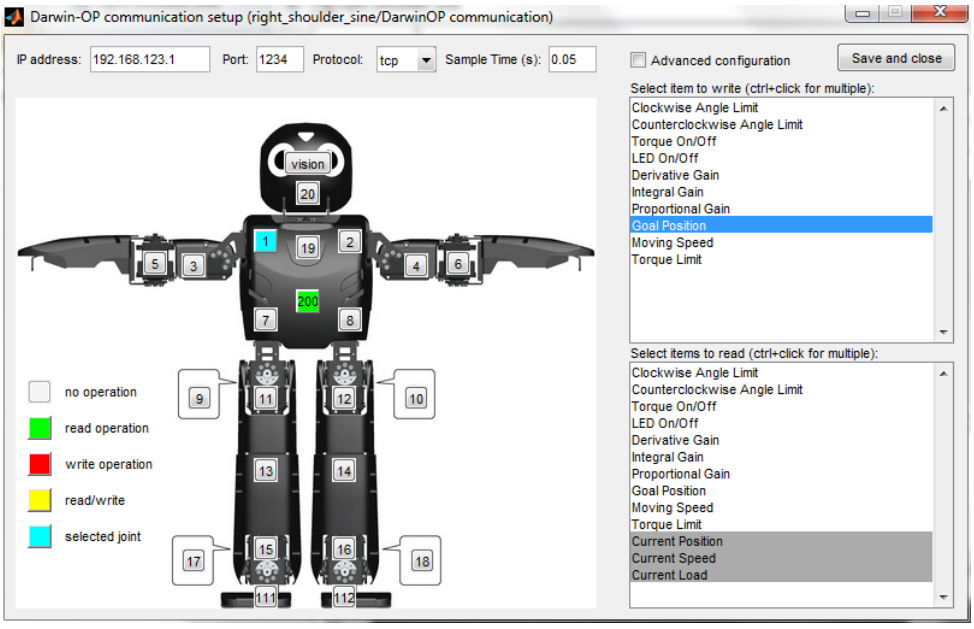

Figure 8 Fenêtre de configuration du bloc "DarwinOP communication"

Elle se décompose en trois zones aux fonctionnalités différentes :

- Les lignes de saisies en partie supérieure paramètre le robot utilisé en mode « simulation ». L'adresse IP par défaut est « 192.168.123.1 » et le port utilisé « 1234 » selon le protocole

« TCP ». La période d'échantillonnage fixe l'intervalle de rafraîchissement de toutes les sorties ;

- La partie gauche résume les opérations effectuées sur chacune des actionneurs/capteurs du robot. Le modèle exécute des opérations de lecture sur les actionneurs/capteurs colorés en vert, des opérations d'écriture sur les actionneurs/capteurs colorés en rouge. Les actionneurs/capteurs colorés en jaune ont à la fois des données lues et écrites. L'actionneur/capteur en bleu est en cours de configuration et ses détails sont affichés dans la partie droite ;
- Les deux listes dans la partie droite de la fenêtre résument les données écrites (en haut) et lues (en bas) pour l'actionneur/capteur sélectionné dans la partie gauche (coloré en bleu). Il est possible de sélectionner/désélectionner plusieurs données à lire/écrire en maintenant enfoncée la touche « CTRL » pendant le clic. Par défaut, seules les données les plus courantes sont représentées dans ces deux listes, la case à cocher située au-dessus permet d'afficher toutes les données disponibles dans les actionneurs MX-28 (identifiants 1 à 20), dans le sous-contrôleur CM-730 (identifiant 200), dans les pieds à retour d'effort (identifiants 111 et 112) et dans le capteur de vision (webcam non reliée sur le bus Dynamixel).

Une fois la fenêtre fermée à l'aide du bouton « Valider et fermer », les entrées/sorties du bloc correspondant sont automatiquement ajustées en fonction de la configuration. Pour annuler les changements en cours dans la fenêtre de configuration, il faut la fermer à l'aide de la croix rouge « Windows » en haut à droite.

Pour chaque entrée/sortie, le nom affiché chacune des entrées/sorties correspond aux informations suivantes : « nom de l'actionneur (identifiant)/nom de la donnée » ou « nom du capteur (identifiant)/nom de la donnée ». Deux blocs « 12-bit signed conversion » peuvent être ajoutés pour convertir les données de vitesse et de charge depuis le format brut (« raw ») dans le format numérique utilisé par Matlab (« decoded »). Les différentes entrées/sorties peuvent ensuite être reliées à des blocs « sources » et « sinks » classiques (Figure 9).

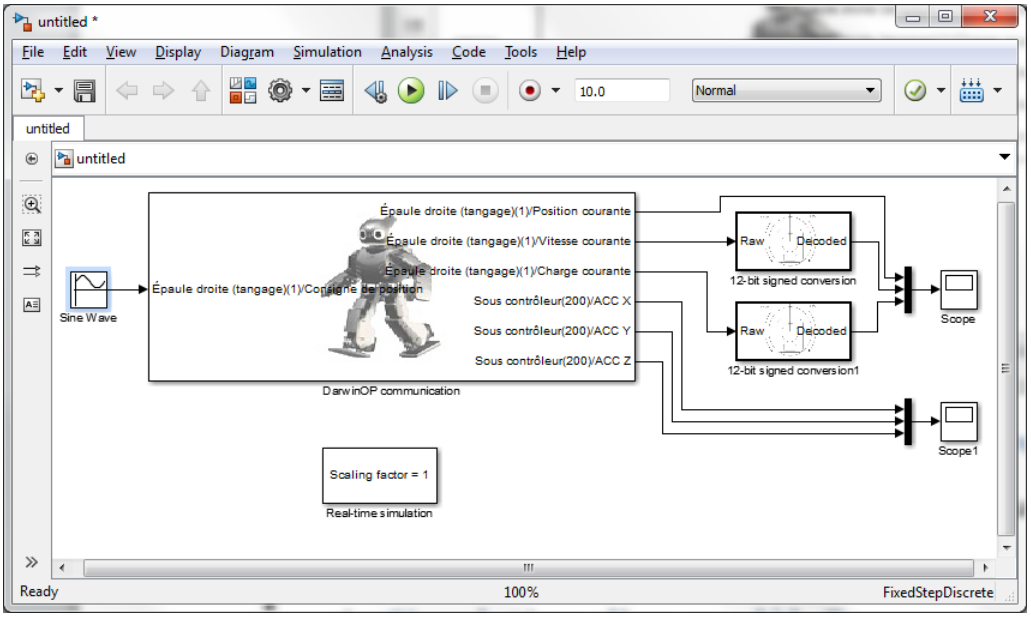

Figure 9 Modèle complet

d) Paramétrages des blocs de stimulation

Le générateur Sinus est paramétré pour créer un signal numérique entre les valeurs 0 et 4095 avec une période de 2s et une période d'échantillonnage de 0.1s (Figure 10).

Pour les deux blocs « DarwinOP communication » et « Sine Wave », la période d'échantillonnage doit être égale ou un multiple de la fréquence d'échantillonnage configurée dans les paramètres du modèle (Figure 6).

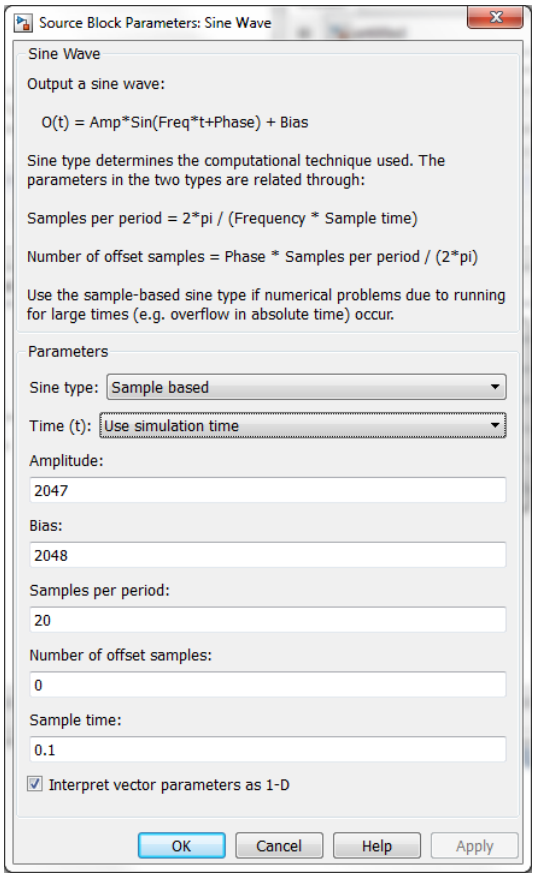

Figure 10 Confiuguration du générateur Sinus

4. Simulation d'un modèle

La simulation d'un modèle est possible à l'aide de la flèche verte « Run » dans la barre d'outils ou par le menu « Simuation/Run » ou par le raccourci clavier « CTRL+T ». La simulation est possible dans les modes « Normal » et « Accelerator » (menu déroulant dans la barre d'outils à droite du temps de simulation). Les modes « Software-in-the-loop », « Hardware-in-the-loop » et « External » ne sont pas opérants en mode simulation.

La simulation s'exécute pour le temps spécifié dans la ligne de saisie « Simulation stop time » dans la barre d'outils. Il est possible de faire une simulation à temps infini en spécifiant « inf » à cet endroit. La simulation doit alors être arrêtée à l'aide du bouton stop de la barre d'outils.

La simulation nécessite l'exécution sur le robot d'un programme d'instrumentation situé à l'adresse « /darwin/Linux/project/instrumentation/instrumentation ». Il est possible de lancer ce programme à la main sur le robot à l'aide des commandes suivantes exécutées dans une console connectée au robot :

```
$ /darwin/Linux/project/instrumentation/instrumentation
```

Si ce programme n'est pas en exécution lors du démarrage d'une simulation, le simulateur tentera de lancer automatiquement ce programme sur le robot.

5. Programmation matérielle

a) Sans retour d'information

La programmation matérielle est lancée à l'aide du bouton « Build Model » situé à droite de la barre d'outils ou du menu « Code/C/C++Code/Build Model » ou du raccourci clavier « CTRL+B ». Pendant cette phase, le modèle Simulink est traduit automatiquement en un code C/C++, téléchargé sur le robot, compilé à l'aide des outils du robot et enfin exécuté sur cette cible matérielle.

Le code C/C++ est généré dans le répertoire « nom du modèle_darwinop_rtw » situé dans le répertoire de travail courant de Matlab. Ce code est envoyé automatiquement dans le répertoire « /darwin/Linux/Simulink/nom du modèle » sur le robot et compilé à cet endroit.

Pendant la phase de programmation matérielle, les blocs « Real-time simulation », les blocs de simulation tels que les oscilloscopes, ainsi que la durée de simulation sont ignorés. L'exécution du programme commence automatiquement quelques secondes après l'exécution de la commande, l'état d'avancement est visible dans la console Matlab. **Le programme s'exécute pendant un temps infini : le temps de simulation tel que spécifié dans le modèle n'est pas pris en compte.**

Par défaut le mode « programmation » ne permet pas l'observation des différents signaux du modèle (les oscilloscopes et les exports ne sont pas implémentés). Il est néanmoins possible d'instrumenter et d'observer le modèle en paramétrant le générateur de code. Deux retours d'instrumentation sont possibles :

- Enregistrement en temps réel de signaux sous la forme de fichiers « .mat ». Ces fichiers peuvent ensuite être téléchargés sur le PC et chargés dans Matlab.
- Communication réseau pour récupérer les informations en temps différé.

b) Enregistrements des données

Le bloc « To File » est situé dans la page « Simulink/Sinks ». Il faut le déposer sur le modèle et le relier au(x) signal/signaux à enregistrer. Les paramètres de ce bloc permettent de spécifier le nom du fichier d'enregistrement et le nom de la variable stocké à l'intérieur de ce fichier (ces deux noms sont arbitraires et à votre choix). Le paramètre « Save format » doit être paramétré en « Array ». Pour réduire le volume des données enregistrées, il est possible d'enregistrer 1 point sur « n » en spécifiant ce « n » dans le paramètre « Decimation ».

L'enregistrement de fichier sur le robot nécessite du code supplémentaire à générer et à lier dans le programme s'exécutant sur le robot. L'ajout de ce code supplémentaire doit être paramétré dans les options du modèle dans la page « Code Generation/Interface » (Figure 11) : il faut cocher la case « MAT-file logging ».

Les fichiers générés sont accessibles une fois la simulation terminée dans le répertoire « /Darwin/Linux/Simulink/nom du modèle ». Il est possible de les télécharger depuis le robot sous Linux vers le PC sous Windows à l'aide de l'outil winSCP.

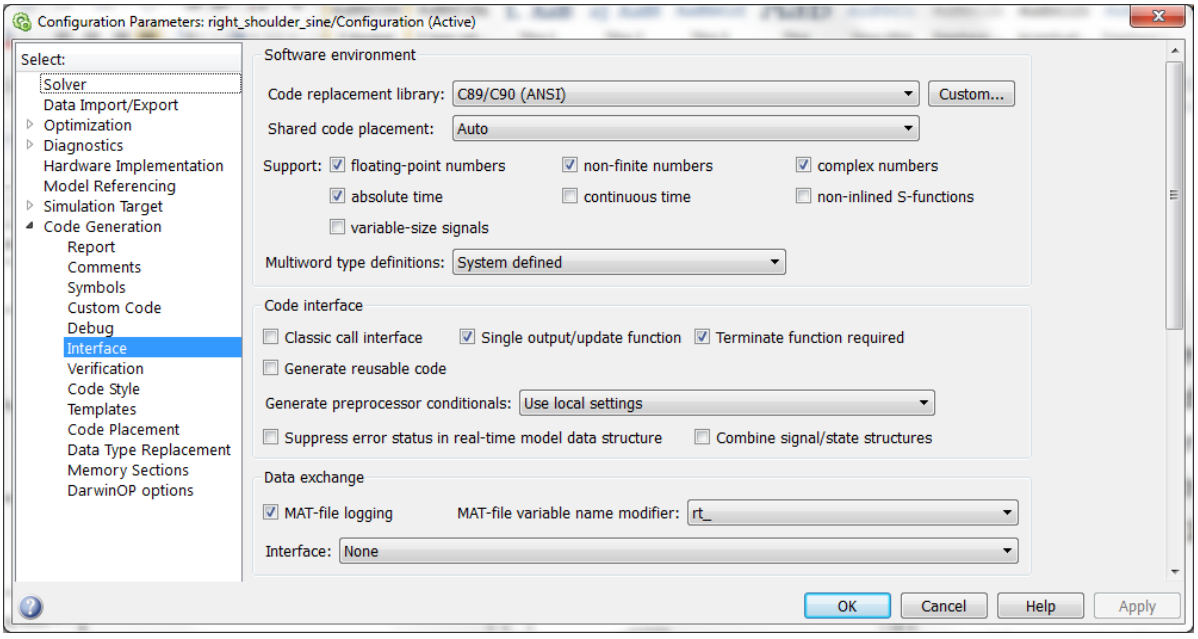

Figure 11: Paramétrage de l'enregistrement

Si cette case n'est pas cochée, les blocs « To File » peuvent être laissés sur le modèle, ils sont alors inopérants et ne sont pas générés dans le code lors de la programmation matérielle, ils restent néanmoins actifs en mode simulation.

c) Retour d'information par le réseau

Le programme s'exécutant sur le robot peut aussi communiquer avec Matlab Simulink afin d'afficher en temps différé les données sur les oscilloscopes et les blocs d'export tels que « To workspace ». Cette communication nécessite un ajout de code dans le programme généré qui peut être activé dans les options du projet dans la page « Code Generation/Interface » () : il faut régler la liste déroulante « interface » à « External mode ». Plusieurs champ de configuration apparaissent alors : le « Transport layer » doit être réglé à « tcpip » et le paramètre « MEX-file arguments » est généré automatiquement par la bibliothèque lors du démarrage de la simulation. Ce dernier contient normalement la valeur « '192.168.123.1' 1 17725 » où '192.168.123.1' correspond à l'adresse IP du robot, « 1 » à l'affichage d'informations détaillées dans la console Matlab et « 17725 » au port TCP/IP sur lequel Matlab Simulink va se connecter sur le robot pour récupérer les données.

La case à cochée « Static memory allocation » doit être décochée : le programme sur le robot s'exécute dans un environnement Linux disposant d'un gestionnaire mémoire supportant les allocations dynamiques.

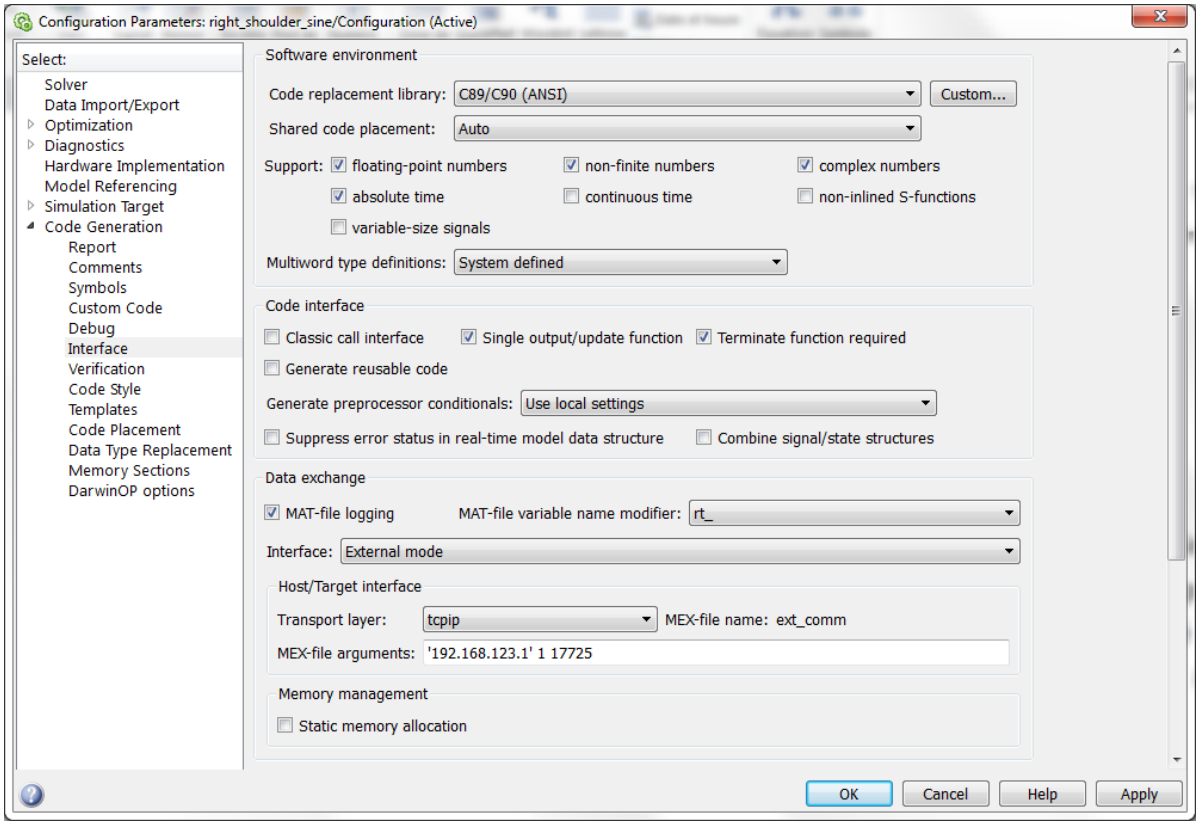

Figure 12: Paramétrage de la communication en mode externe

Une fois cette configuration effectuée, il faut changer le mode de simulation à « External » dans la barre d'outils du modèle (Figure 13). Il faut ensuite reconstruire le modèle (CTRL+B). L'exécution du programme ne commence pas automatiquement sur le robot. Il faut se connecter sur la cible à l'aide du menu « Simulation/Connect to Target » ou du bouton « Connect to Target » sur la barre d'outils.

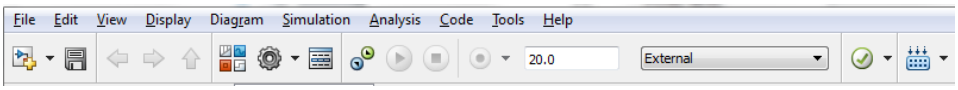

Figure 13: Paramétrage du modèle en mode externe

La console Matlab donne alors des informations sur l'avancement de la connexion. Une fois connecté au programme s'exécutant sur le robot, la flèche verte (ou le menu « Simulation/Run ») permet de lancer son exécution pendant le temps spécifié dans la barre d'outils. A la fin de l'exécution du programme, les données sont envoyées au PC et affichées dans les oscilloscopes du modèle.

Le programme s'exécutant sur le PC est fortement lié au modèle s'exécutant sur le PC, il ne faut pas faire de modifications sur le modèle entre la construction et la connexion au robot. En cas de modification, Matlab déclenchera une erreur et refusera la connexion. Il faut alors reconstruire le programme.

6. Détails de la bibliothèque

La bibliothèque comporte de nombreux fichiers nécessaires à son fonctionnement, ils ne doivent pas être supprimés. Cette partie contient une description sommaire des fonctions réalisées dans chacun d'eux :

- blocs/ : répertoire contenant la bibliothèque Matlab Simulink « DarwinOP Lib » et les fichiers propres à chacun des blocs ;
- blocs/darwin.jpg : image affichée dans la fenêtre de configuration (Figure 8) ;
- blocs/darwin-blk.jpg : image affichée sur le bloc « DarwinOP communication » (Figure 7) ;
- blocs/darwin-conversion.jpg : image affichée sur le bloc « Speed/Load conversion » (Figure 7) ;
- blocs/darwinop_communication.m : code exécuté lors de la simulation du composant « DarwinOP communication » ;
- blocs/darwinop_communication.tlc : squelette de code généré pour le composant « DarwinOP communication » ;
- blocs/darwinop_communication_setup.m : code de la fenêtre de configuration du composant « DarwinOP communication » (Figure 8) ;
- blocs/darwinop_consts.m : définition de tous les textes affichés dans les interfaces graphiques en anglais ;
- blocs/darwinop_consts_fr.m : définition de tous les textes affichés dans les interfaces graphiques en français ;
- blocs/darwinop_realtime.m : code exécuté lors de la simulation du composant « Real-time simulation » ;
- blocs/darwinop_realtime.tlc : squelette de code généré pour le composant « Real-time simulation » (aucun code puisque ce bloc est inactif lors de la génération de code » ;
- blocs/lib_darwinop.slx : définition des composants graphiques de la bibliothèque Matlab Simulink « DarwinOP Lib » ;
- blocs/slblocks.m : fonction d'enregistrement de la bibliothèque Matlab Simulink « DarwinOP Lib » ;
- darwinoplib/ : répertoire contenant le code générant le code source C++ envoyé et compilé sur le robot ;
- darwinoplib/darwinoplib.tlc : « System target file » définition des règles de génération de code (Figure 4) et les paramètres propres au robot DarwinOP (Figure 5) ;
- darwinoplib/darwinoplib.tmf : « Template Make File » squelette de fichier de commandes exécuté sur le robot pour compiler et lier l'application ;
- darwinoplib/darwinoplib_main.tlc : squelette de code du fichier principal du programme C++ généré (contient le point d'entrée « main ») ;
- darwinoplib/darwinoplib_make_rtw_hook.m : commande additionnelle de construction pour envoyer et compiler le programme sur le robot ;
- darwinoplib/darwinoplib_select_callback.m : commande à executer après la selection de « darwinoplib.tlc » dans la Figure 4 (force l'utilisation du langage C++ par exemple) ;
- darwinoplib/darwinoplib_settings.tlc : vérifie lors de la construction du programme les options supportées et non-supportées par le générateur de code ;

- darwinoplib/darwinoplib_setup.m : fonction d'enregistrement des chemins de la bibliothèque ;
- darwinoplib/rtiostream_interface.c : fichier modifié de Matlab intervenant dans le transfert des données entre le programme s'exécutant sur le robot et Matlab Simulink en mode « programmation matérielle avec retour d'information par le réseau » (Page 12) ;
- darwinoplib/sl_customization.m : programme enregistrant le « transport layer tcpip » pour communiquer avec le programme sur le robot (Figure 12)
- darwinoplibdemos/ : répertoire contenant les démos de la bibliothèque (Page 16) ;
- putty/ : répertoire contenant les programmes permettant à Matlab d'envoyer des fichiers sur le robot et d'exécuter des commandes à distance. Ces programmes font partie de la suite « Putty »[6] ;
- putty/licence.txt : licence d'utilisation de ces programmes tiers (type BSD donc sans limitation particulière) ;
- putty/plink.exe : programme permettant l'envoi de commandes à distance ;
- putty/pscp.exe : programme permettant l'envoi de fichiers vers le robot ;
- putty/readme.txt : fichier faisant référence à la source de ces programmes ;
- .gitignore : gestion des fichiers ignorés (inutile pour l'utilisateur final) ;
- manual-fr.pdf : manuel d'utilisation de la bibliothèque en français (ce fichier) ;
- manual.pdf : manuel d'utilisation de la bibliothèque en anglais ;
- readme.txt : informations générales sur la bibliothèque.

[6] http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

7. Détails des démos fournies

Plusieurs démos sont disponibles dans le répertoire « darwinoplibdemos », chacune met en évidence une utilisation possible des blocs de la bibliothèque. Ces démos fonctionnent toutes en simulation et en génération de code.

a) gravity.slx

Cette démo illustre l'utilisation des accéléromètres internes à la centrale inertielle du robot dans le but de détecter la direction de la pesanteur. Le robot bouge alors ses bras gauche et droit pour indiquer la direction verticale, quelle que soit l'orientation du robot. Les coordonnées cartésiennes récupérées depuis l'accéléromètre « 3-axes » sont converties en coordonnées sphériques et envoyées aux articulations des deux bras en fonction de la direction de la pesanteur.

Attention lors de l'utilisation de cette démo : les deux bras bougent et peuvent pincer « fort », il est préférable de tenir le robot par les jambes, par la sangle arrière ou par le torse (avec appuis sur le devant et l'arrière).

b) Right_shoulder_sine.slx

L'épaule droite est actionnée (dans son degré de liberté « tangage ») d'après une consigne de position « sinus ». La position, la vitesse et la charge réelles sont relevées à l'oscilloscope, de même que les trois axes de l'accéléromètre.

c) vision_processing.slx

Le robot repère les taches de couleurs rouges/bleues/jaunes à l'aide de sa webcam. Ces positions sont relevées grâce à des oscilloscopes en mode XY. Le robot désigne également grâce à ses bras la position de la balle vue par la webcam.