Le but du projet est de réaliser un compilateur**TORCC** pour un langage impératif permettant de simplifier l'écriture de codes de robots pour TORCS, le simulateur de courses automobiles. Le projet est à réaliser par binôme, et sera à rendre en janvier (date à fixer en fonction des partiels).

1- TORCS et objectif du projet

TORCS est un simulateur de courses automobiles dont les voitures peuvent être pilotées par des robots. On trouvera la procédure d'installation de TORCS sur le site web. Les robots pilotant les voitures sont programmés en C++/C et un tutorial est disponible ici. Pour le pilotage de la voiture, les robots disposent d'information correspondant à ce que verrait un pilote:

- compteur de tour,
- vitesse,
- position sur la route,
- orientation de la route,
- position du levier de vitesse

Ses actions consistent à :

- Tourner le volant
- Appuyer plus ou moins fort sur l'accélérateur, la pédale de frein, l'embrayage
- Changer de vitesse

Il a par ailleurs une connaissance de paramètres de la voiture, comme la vitesse de déplacement de roues, de leur rotation (pour déterminer si les roues glissent), de divers paramètres du moteur...

L'objectif du projet est de pouvoir simplifier l'écriture de code de robots. Cela passe par l'utilisation des seules caractéristiques disponibles à un pilote (vitesse, compteur de tour, ...). Les paramètres plus fins concernant le fonctionnement du moteur ou la motricité seront ignorés. En revanche, on pourra définir des mécanismes d'aide à la conduite comme

- le freinage d'urgence, en cas de collision imminente
- l'ABS
- la boite automatique
- ...

qui eux peuvent prendre en compte ces paramètres. Le code du robot sera décrit par un langage dédié dont il faut réaliser le compilateur. Le code généré sera de la représentation intermédiaire LLVM. Le code pour les aides à la conduite sera en C/C++ et sera fourni sous forme de bibliothèque et ne fait pas partie du projet. En revanche, ces mécanismes auront un impact sur le code généré du robot.

Pour le fonctionnement du projet, il est recommandé d'installer (en local) le simulateur TORCS sur votre machine. Le compilateur pourra néanmoins être testé indépendamment de l'installation de TORCS.

2- Installation et test de TORCS

Deux installations sont possibles. Sur votre machine personnelle, il est recommandé d'utilisé la dernière archive de TORCS:

http://sourceforge.net/project/torcs/

ou, si vous testez sur les machines de l'école, le plus simple est d'utiliser la version installée appelée torcs-local que vous pouvez télécharger du dossier à partir de moodle.

Dans ce dernier cas, décompressez l'archive dans la racine de votre repértoire. Vous avez alors un repertoire **bin** et **lib** qui sont créés et qui contiennent l'essentiel de la distribution de TORCS. Pour l'utiliser, il faut vous logger sur **travail32**

> ssh -X travail32

> bin/torcs

Choisissez alors "Race","Practice","Configure Race","Accept" pour accepter la piste par défaut, puis selectionnez le robot pilot enseirbot dans la liste, faites "Accept", dans les options faites "Accept" puis "New race". Apres un certain temps, vous verrez "Ready" puis la course demarre avec un pilote très simple.

3- Le langage d'entrée

Le langage d'entrée est basé sur du C. En cas d'ambiguité sur la façon d'interpréter le langage, on se basera sur le C. Les types autorisés sont void, int, float et les tableaux à une dimension. Ces tableaux ne pourront être déclarés que statiquement. Le langage permet en outre de lire certaines variables qui représentent l'état de la voiture. Ces variables, définies dans TORCS dans car.h, seront ici préfixées par $. Ces variables ne sont qu'en lecture seule et le compilateur devra vérifier qu'elles ne sont jamais écrites. Ces variables sont:

- $x,$y,$z: les coordonnées de la voiture (flottant). Correspondent à _posx, _posy, _posz en C.
- $speedx,$speedy,$speedz: la vitesse de la voiture (flottant). Correspondent à _speed_x, _speed_y, _speed_z en C.
- $accelx,$accely,$accelz: l'accélération de la voiture (flottant). Correspondent à _accel_x,_accel_y,_accel_z en C.
- $rpm: les tours/minute du moteur. Correspond en C à _enginerpm
- $gear: la position du boitier de vitesse. Correspond en C à _gear.
-

Par ailleurs, les variables suivantes permettent de modifier le comportement de la voiture et peuvent être écrites et lues:

- $steer: entre -1 et 1, définit la position du volant. Correspond en C à _steerCmd
- $accel: entre 0 et 1: définit la position de la pédale d'accélérateur
- $brake: entre 0 et 1, idem pour le frein
- $clutch: entre 0 et 1, idem pour l'embrayage
- $gear: entre -1 et 6, pour les vitesses

La grammaire et l'analyseur lexical sont fournis.

4- Génération de code

Le langage précédent devra être compilé en la représentation intermédiaire de LLVM. La documentation de cette page est également en ligne.

Le compilateur **TORCC**devra vérifier que les variables et noms de fonctions utilisés sont bien du bon type.

5- A livrer

Il y a deux livrables à faire, le premier avant les vacances de Noel et le second au mois de janvier.

- Pour le 20 décembre: rendre un compilateur minimal. Il doit juste permettre, à partir d'un fichier d'entrée décrivant une fonction drive définissant une valeur d'accélération (**$accel = 1;** par exemple), de générer une bibliothèque pour un robot dans TORCS.
- Pour fin janvier: rendre le compilateur complet. Une soutenance de 5 min est sera faite, permettant de décrire le projet réalisé.