Session S8 Électrique & Informatique

APP

Unité 2

Semaines 9, 10 et 11

INTELLIGENCE ARTIFICIELLE BIO-INSPIRÉE : LOGIQUE FLOUE, RÉSEAUX DE NEURONES ARTIFICIELS ET ALGORITHME GÉNÉTIQUE - APPRENTISSAGE MACHINE II

Documentation des outils

Département de génie électrique et de génie informatique
Faculté de Génie
Université de Sherbrooke

Automne 2017

Note : En vue d'alléger le texte, le masculin est utilisé pour désigner les femmes et les hommes.

Document GIA792A17DocumentInformationComplementaireProbleme.pdf
Version 4, 23 octobre 2017
Rédigé par Simon Brodeur, Msc.A. et Jean Rouat, Ph.D.en collaboration avec

Tuteur: Jean Rouat, Ph.D.
Réalisé avec TeXshop et LaTeX.

Table des matières

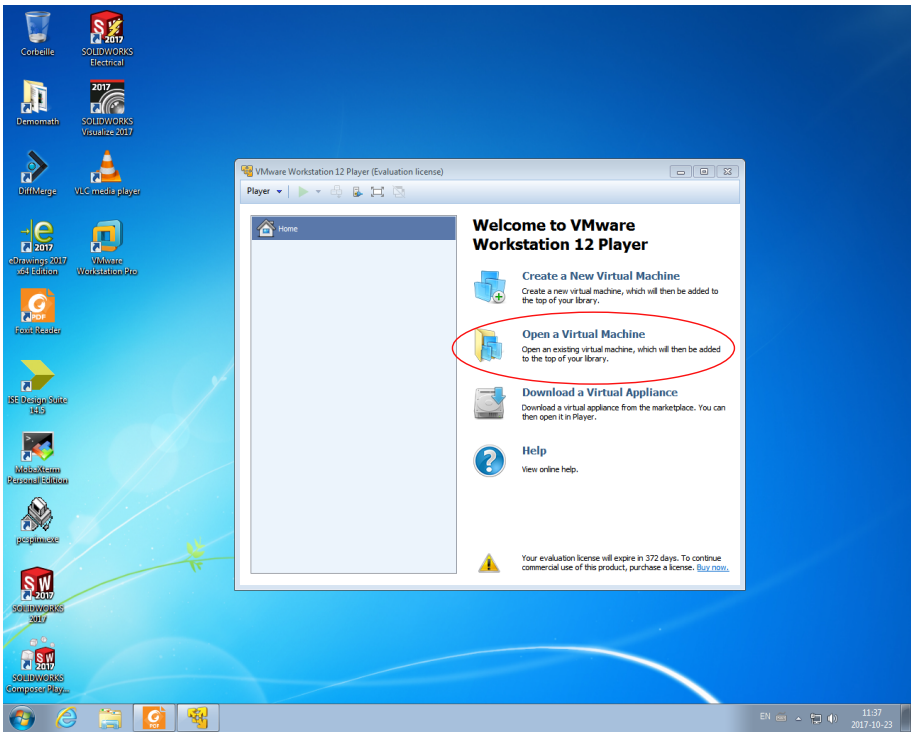
1 Introduction

Une machine virtuelle Ubuntu LINUX comprenant l'ensemble des logiciels (simulateur de voiture et outils d'intelligence artificielle) nécessaires pour la résolution du problème est fournie. Cette machine virtuelle inclut les environnements Octave 4.0 et TORCS – "Open Racing Car Simulator". L'installation des environnements LINUX, Octave, TORCS ainsi que l'écriture des codes d'interface entre Octave et le simulateur TORCS ont été réalisées par Simon Brodeur. Veuillez vous référer à Simon pour toute question sur l'environnement logiciel de l'APP.

2 Comment aborder les outils ?

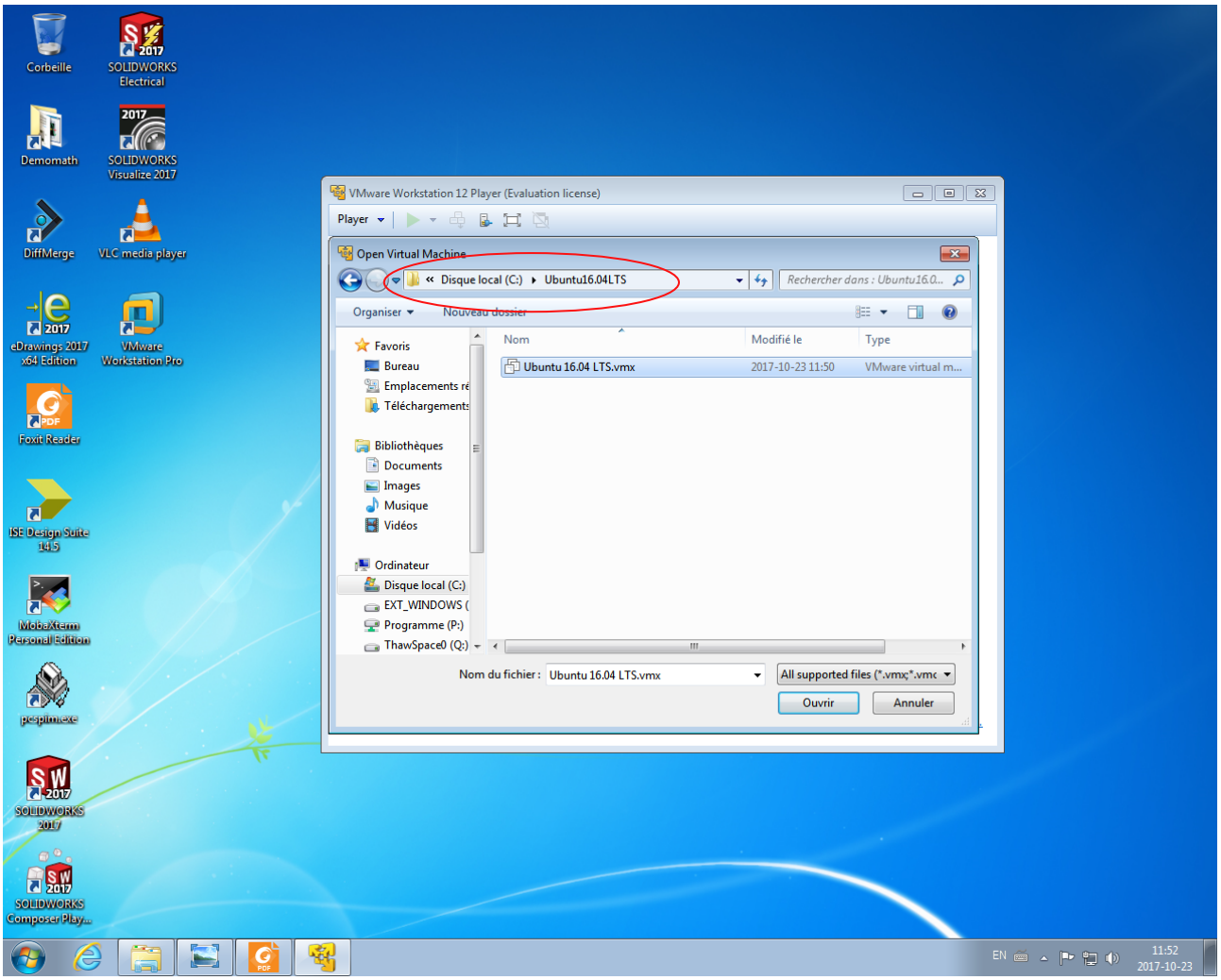
2.1 Machine virtuelle

La machine virtuelle est portable et peut être installée sur votre ordinateur personnel. Il est possible de la récupérer sur les postes des laboratoires dans le dossier *C :/Ubuntu16.04LTS/* ou sur le site web de l'APP. Le logiciel VMware Player est requis pour exécuter la machine virtuelle [1]. La procédure pour lancer la machine virtuelle est la suivante :

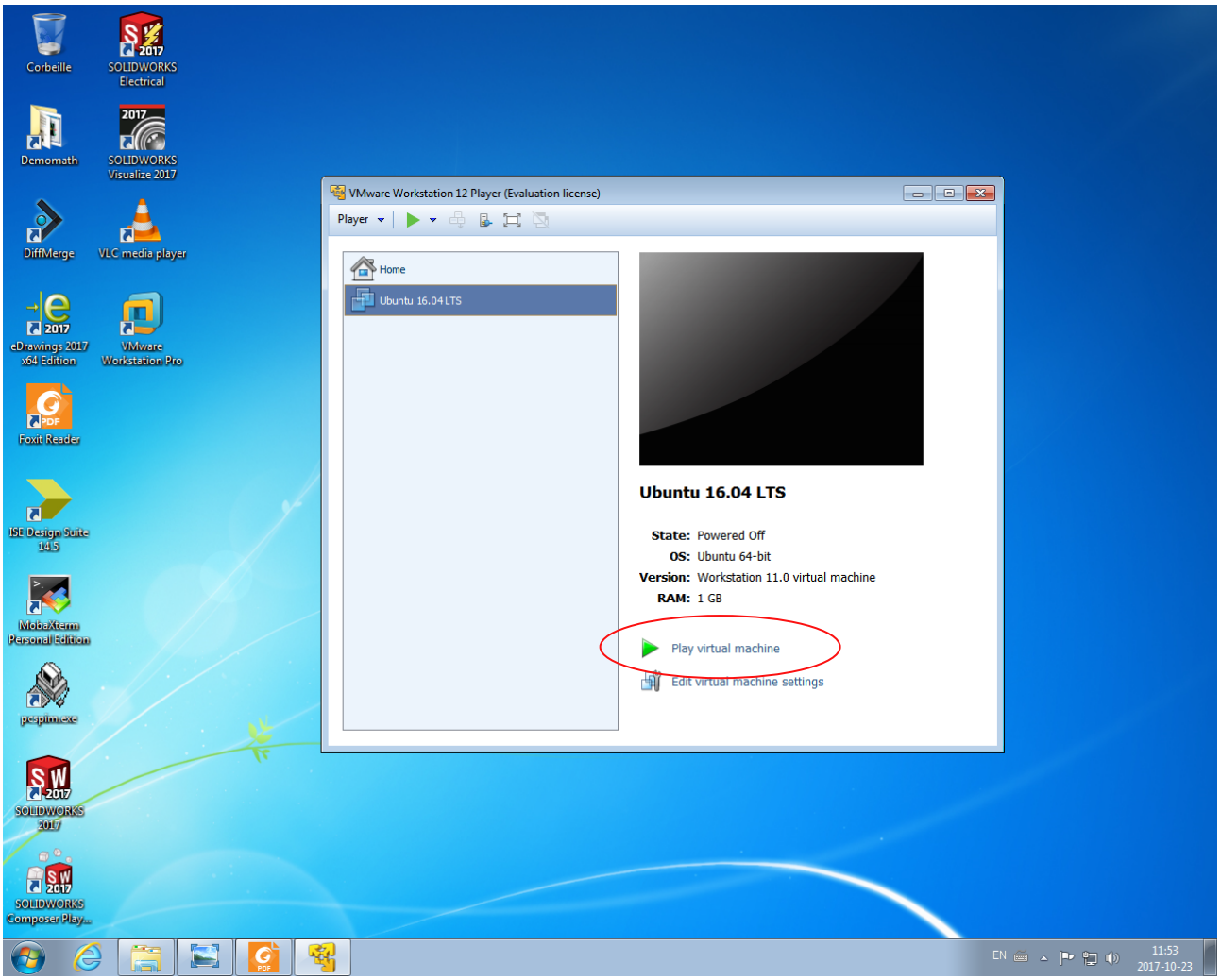


Étape 1 - Ouverture du logiciel VMware Player.

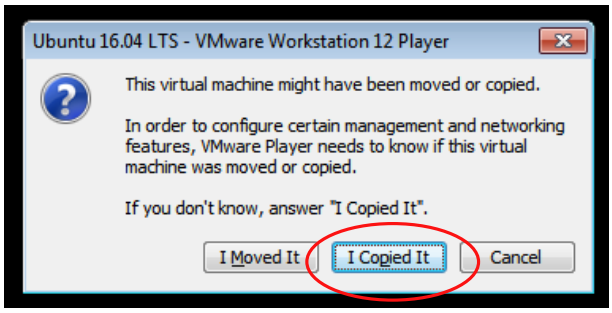
1. La machine virtuelle peut aussi être importée par le logiciel libre VirtualBox, mais certains problèmes peuvent apparaître relatifs à l'accélération 3D et au support OpenGL pour les simulations avec rendu graphique. Ainsi, aucun support ne sera fourni aux étudiants quant à l'utilisation de ce logiciel.



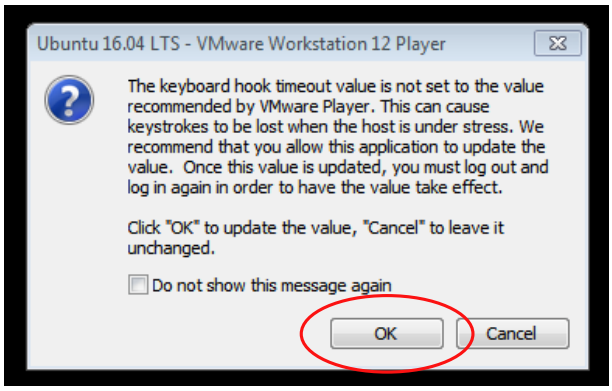
Étape 2 - Importation de la machine virtuelle à partir du répertoire *C :/Ubuntu16.04LTS/*.



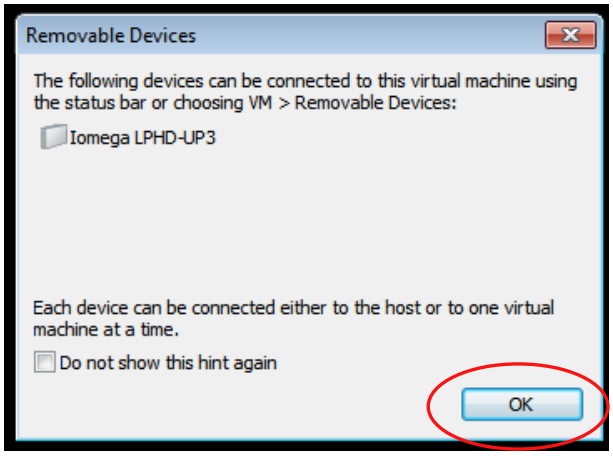
Étape 3 - Exécution de la machine virtuelle par le bouton "Play virtual machine".



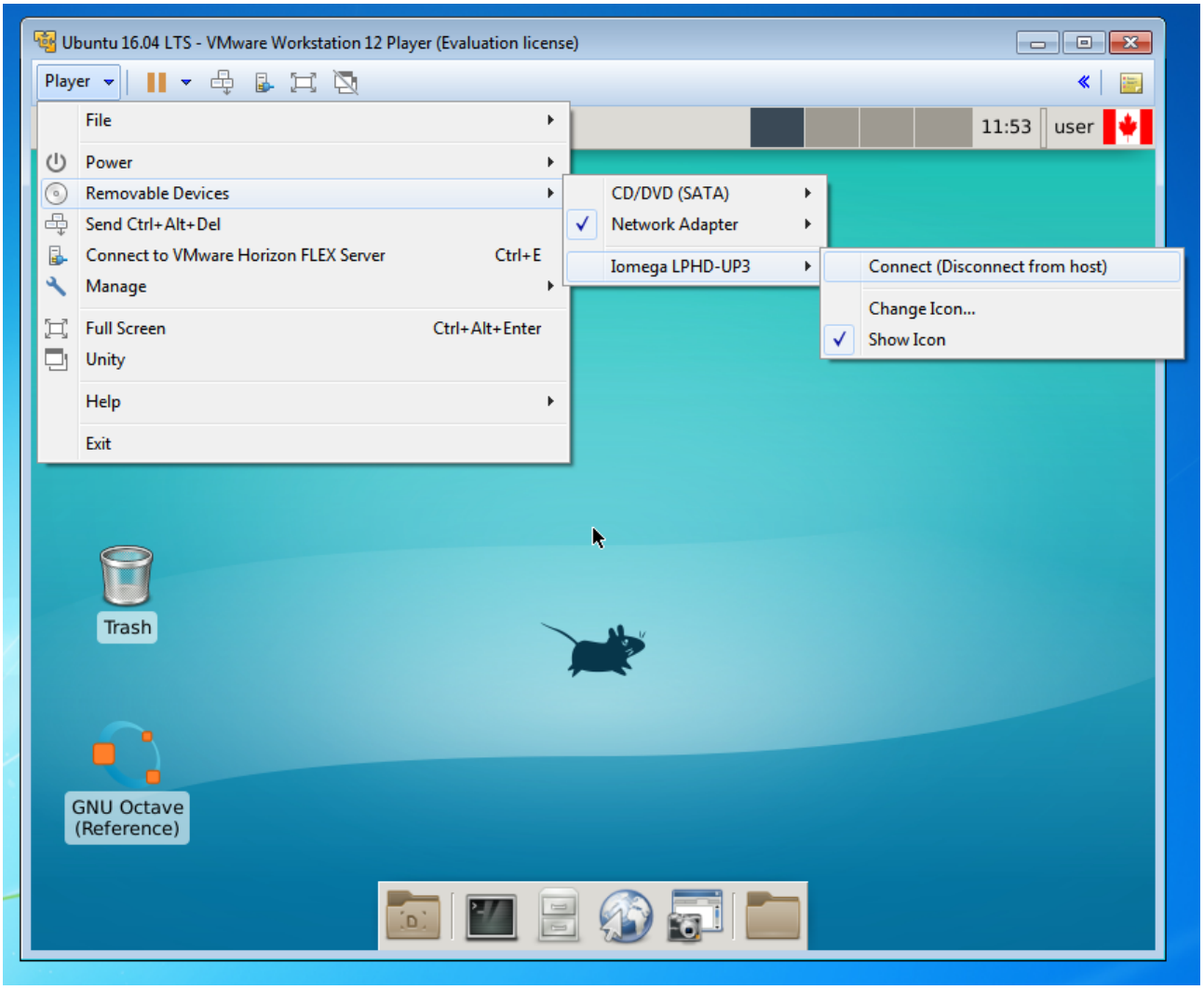
Si vous recevez ce message, appuyer sur le bouton "I copied it".



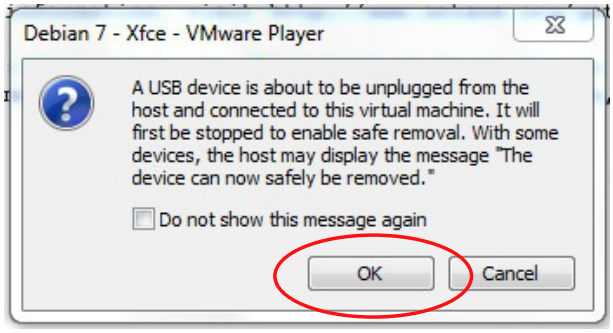
Si vous recevez ce message, appuyer sur le bouton "OK".



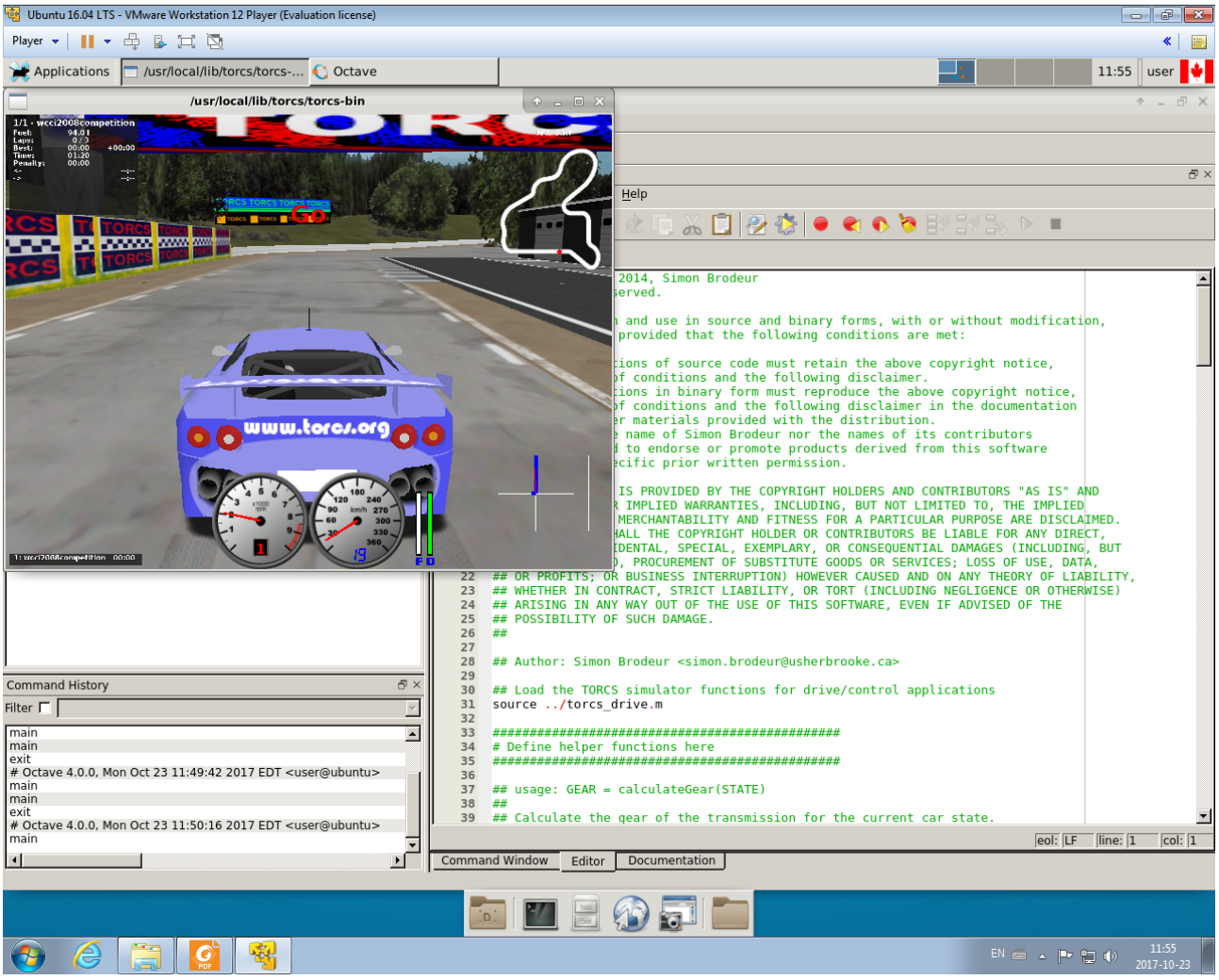
Si vous êtes avertis que certains périphériques USB peuvent être connectés ultérieurement à la machine virtuelle, appuyer sur le bouton "OK". Par exemple, si une clé USB est connectée à l'ordinateur, vous devriez la voir dans la liste des périphériques.



Étape 4 - Pour connecter votre clé USB contenant le code, allez dans l'onglet "Removable Devices" et sélectionnez "Connect (Disconnect from host)" dans le sous-onglet de votre clé USB. Le répertoire de la clé USB apparaîtra sur le bureau de la machine virtuelle, et sera aussi disponible sous le chemin */media/user*. Pour ouvrir un fichier de code *.m* avec l'éditeur Octave, vous devez utiliser ce chemin direct. ATTENTION : Il est important de ne jamais travailler dans un répertoire interne à la machine virtuelle (e.g. un dossier créé sur le bureau), car vous perdrez votre code et vos données lorsque les ordinateurs du laboratoire seront réinitialisés chaque nuit. Travaillez toujours directement à partir d'un dossier sur une clé USB.



Si vous recevez ce message, appuyer sur le bouton "OK".



Étape 5 - Lancez votre script (ici l'exemple *drive-simple/main.m*) par l'éditeur Octave. Dépendamment du mode de simulation, vous devriez voir une fenêtre apparaître avec le rendu graphique de la simulation. Pour arrêter la simulation, fermer cette fenêtre ou appuyer sur CTRL-C dans la fenêtre de commande d'Octave. Il est préférable d'utiliser l'éditeur Octave pour écrire ou modifier du code.

Si vous recevez un message similaire, appuyer sur le bouton "Change Directory".

2.2 Simulateur de voiture et interface Octave

Commencez par vous familiariser avec le simulateur de voiture et l'interface en examinant le fichier *main.m* situé dans le répertoire *drive-simple* du code fournit. Ce fichier fait parcourir un circuit complet à la voiture tout en enregistrant à chaque instant l'état de la voiture. Les données enregistrées sont stockées dans le répertoire *drive-simple/data*. Vous pouvez vous inspirer de la fonction *showData.m* située dans le répertoire *record* pour examiner les variables d'entrées et de sortie du simulateur.

ATTENTION NE PAS MODIFIER les fichiers **torcs_drive.m** ainsi que **torcs_opt.m**. Ces fichiers réalisent l'interface entre le simulateur et Octave. Il en est de même des fichiers XML de configuration **race-config-gui.xml** et **race-config-nogui.xml**.

2.3 Description des structures de données

2.3.1 Mode optimisation

En mode optimisation, votre algorithme implémenté sous Octave transmet un vecteur de 5 paramètres au simulateur (voir Tableau 1 pour description). Physiquement, les paramètres correspondent aux ratios (ou rapports) de vitesse de la transmission pour les 5 dernières vitesses, comme illustré à la figure 2. Le simulateur roule le véhicule avec ces paramètres pour un temps défini. Il retourne ensuite des mesures de performance sur la voiture. Ce mécanisme est illustré à la figure 1. Vous devrez exploiter ce mécanisme pour implémenter la méthode d'optimisation par algorithme génétique.

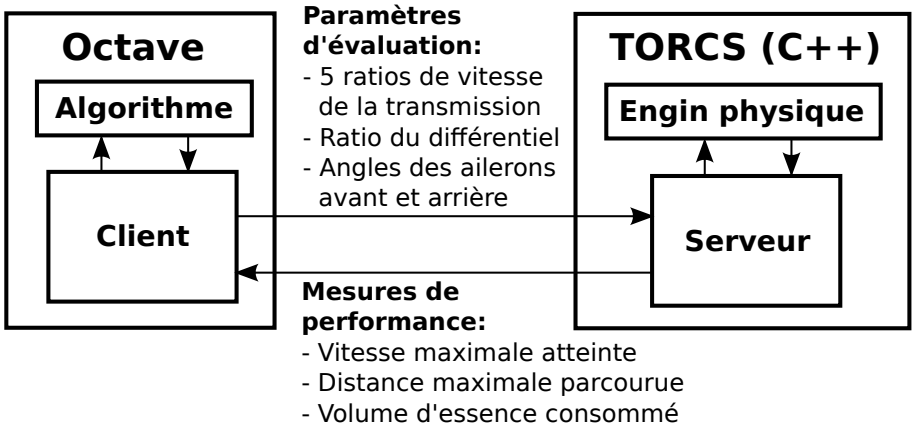


FIGURE 1 – Mécanisme de communication entre le code Octave et TORCS pour la simulation en mode optimisation.

Index	Description	Valeur typique
0	Ratio de la 2e vitesse	2.5
1	Ratio de la 3e vitesse	1.5
2	Ratio de la 4e vitesse	1.5
3	Ratio de la 5e vitesse	1.5
4	Ratio de la 6e vitesse	1.0

TABLE 1 – Description des paramètres à optimiser sur la voiture. Tous les rapports de vitesse sont en unités SI et définis dans l'intervalle [0,5].

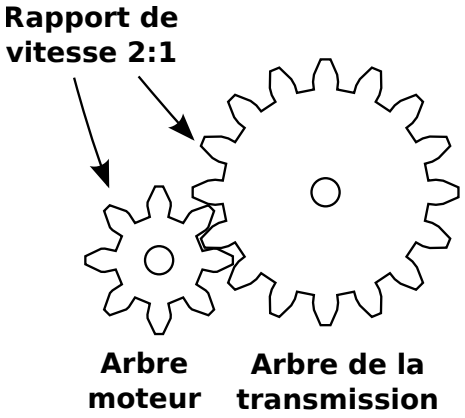


FIGURE 2 – Exemple de rapport de vitesse 2 :1 pour la transmission de la voiture.

2.3.2 Mode contrôle

En mode contrôle, votre algorithme implémenté sous Octave récupère l'état de la voiture dans la simulation et transmet en retour une commande à exécuter. Ce mécanisme est illustré à la figure 3. Vous devrez exploiter ce mécanisme pour implémenter le contrôle de la direction, de l'accélérateur, des freins et de la transmission par logique floue et/ou réseau de neurones artificiels. La structure Octave accessible et décrivant l'état de la voiture comporte les champs décrits au tableau 2. La structure Octave permettant le contrôle de la voiture requiert quant à elle les champs décrits au tableau 3.

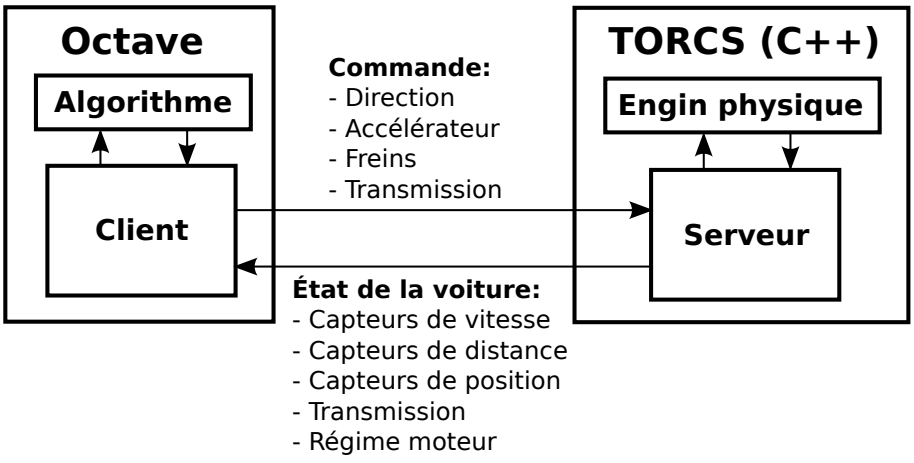


FIGURE 3 – Mécanisme de communication entre le code Octave et TORCS pour la simulation en mode contrôle.

Nom	Description
angle	Angle entre la direction de la voiture et la direction de l'axe de la piste. Intervalle de $[-\pi, \pi]$. [rad]
damage	Dommage actuel sur la voiture (plus la valeur est élevée, plus le dommage est élevé). [points]
curLapTime	Temps écoulé depuis le dernier tour. [s]
distFromStart	Distance parcourue de la voiture à partir de la ligne de départ selon la piste. [m]
fuel	Niveau actuel d'essence dans le réservoir. [litres]
distRaced	Distance parcourue de la voiture depuis le début de la course. [m]
gear	Vitesse actuelle de la transmission. -1 si en reculons, 0 si neutre et de 1 à 6 pour les vitesses avants.
lastLapTime	Temps enregistré pour compléter le dernier tour de piste. [s]
rpm	Nombre de rotations par minute du moteur de la voiture, dans l'intervalle [2000, 7000]. [rpm]
speedX	Vitesse de la voiture selon l'axe longitudinal (i.e. vitesse avant). [km/h]
speedY	Vitesse de la voiture selon l'axe transversal (i.e. vitesse de côté). [km/h]
track	Vecteur de 19 capteurs de distance : chaque capteur représente la distance entre la bordure de la piste (i.e. l'asphalte) et la voiture. Les capteurs sont orientés à chaque 10 degrés de $-\pi/2$ et $+\pi/2$ à l'avant de la voiture. Les capteurs ont une portée de 100 m. Quand la voiture est en dehors de la piste, les valeurs des capteurs ne sont pas fiables ! [m]
trackPos	Distance entre la voiture et l'axe de la piste. La valeur est normalisée par rapport à la largeur de la piste : 0 quand la voiture est sur l'axe (i.e. au centre de la piste), -1 quand la voiture est à la bordure droite de piste et +1 lorsque sur la bordure gauche. Les valeurs supérieures à 1 ou inférieures à -1 indiquent que la voiture se situe à l'extérieur de la piste !
wheelSpinVel	Vecteur de 4 capteurs représentent la vitesse de rotation de chacune des roues. [rad/s]

TABLE 2 – Description des variables décrivant l'état de la voiture durant la simulation. Tableau adapté de Software Manual of the Car Racing Competition @ WCCI2008

Nom	Description
accel	Pédale de gaz virtuelle (0 ne signifiant aucun gaz, 1 plein gaz), dans l'intervalle [0,1].
brake	Pédale de frein virtuelle (0 ne signifiant aucun freinage, 1 plein freinage), dans l'intervalle [0,1].
gear	Vitesse actuelle de la transmission. -1 si en reculons, 0 si neutre et de 1 à 6 pour les vitesses avants.
steer	Angle de la direction. -1 et +1 correspondent respectivement à une direction pleinement à gauche et à droite, correspondant à un angle de 0.785398 rad.

TABLE 3 – Description des variables décrivant une commande de la voiture durant la simulation. Tableau adapté de Software Manual of the Car Racing Competition @ WCCI2008

En mode de contrôle, les données provenant du simulateur sont enregistrées sur le disque en format de tableaux de structure. Pour manipuler ces structures, voir les exemples de code fournit ou l'aide en ligne d'Octave sur ce sujet (i.e. Structure Arrays). Du code est fourni dans le répertoire *record* pour accumuler des données d'entraînement et de test pour l'approche par réseau de neurones artificiels. Il est possible de conduire soi-même ou de faire conduire la voiture par un algorithme optimal sur plusieurs pistes (voir *record/main.m*). À partir de l'enregistrement des données de la voiture pour chacune des pistes, une base de données peut être construite (voir *record/makeDataset.m*). Pour visualiser l'ensemble des données de la voiture pour un exemple de piste, voir le script *record/showData.m*.