Charlotte BRICOUT
Nathan MARTIN
IMA 4ème année

Rapport de projet de 4ème année

Guidage et surveillance d'un véhicule à travers un cockpit de conduite

POLYTECH LILLE

16/04/2014

TABLE DES MATIERES

REMERCIEMENTS

Nous tenons à remercier, tout d'abord, M. Rochdi MERZOUKI et M. Xavier REDON pour la qualité de leur encadrement.

Nous remercions, ensuite, M. Vincent COELEN pour son aide précieuse et le temps qu'il nous a consacré.

Nous remercions, enfin, M. Laurent ENGELS pour sa disponibilité et la qualité de notre vidéo de fin de projet.

INTRODUCTION

Dans le cadre de notre 4ème année à Polytech Lille en spécialité Informatique-Microélectronique-Automatique, nous avons réalisé un projet permettant le guidage et la surveillance d'un train de véhicule par un cockpit de conduite. Ce projet s'inscrit dans la continuité du projet InTraDE, résumé en première partie de ce rapport.

Nous souhaitons relier le cockpit de conduite de Polytech Lille à un véhicule autonome Robucar. Le cockpit qui fonctionne sous le logiciel de simulation temps réel SCANeR Studio, communiquera directement en WiFi avec le Robucar. Tout d'abord, nous modéliserons le parking de Polytech Lille sur le logiciel SCANeR Studio afin de créer un environnement pour la simulation. Ensuite, nous élaborerons un scénario dans lequel nous intègrerons un véhicule de type Robucar, pouvant être piloté par une API (Application Programming Interface) que nous programmerons sur le PC supervision du cockpit de conduite. La simulation se jouant sur le PC supervision.

L'objectif de ce projet est de récupérer la vitesse et l'angle de braquage des roues du Robucar et de les envoyer au PC Supervision où ils seront récupérés par une API. L'envoi de ces consignes se fera par le biais d'un serveur UDP. L'API, constituant un client pour le serveur, prendra en charge l'application des consignes reçues au cockpit de conduite et au modèle dynamique Robucar de la simulation.

De ce fait, nous verrons évoluer le Robucar en temps réel dans la simulation et simultanément, le volant du cockpit suivra les mouvements du véhicule réel. L'utilisateur pourra ainsi suivre le déplacement de la Robucar dans la simulation et observer, parallèlement, les mouvements du volant du cockpit. Cette application est utile pour la surveillance mais aussi pour le guidage d'un véhicule.

I. CONTEXTE ET OBJECTIFS DU PROJET

1. LE PROJET INTRADE

Le projet InTraDE (Intelligent Transportation for Dynamic Environment) est un projet européen dans le cadre du programme INTERREG North West Europe. Il contribue à la gestion du trafic dans des ports de taille considérable tels que les ports de Rotterdam, Düsseldorf ou encore Hambourg. Il participe également à l'optimisation des espaces dans les zones confinées de ceux-ci en développant un système de transport intelligent et écologique, offrant une meilleure sécurité. Ce système est apte à s'adapter aux exigences spécifiques de l'environnement et pourrait être transféré à des ports et terminaux quelles que soient leurs dimensions. Le système de transport évolue dans le site en collaboration avec un logiciel de simulation virtuel : SCANeR Studio. Ce logiciel permet une supervision en temps réel de l'opération de manutention des conteneurs. Pour ce faire, chaque port doit être modélisé dans le logiciel de simulation SCANeR Studio. Cette simulation interagit alors avec un système intelligent de transport routier pour l'acheminement du fret.

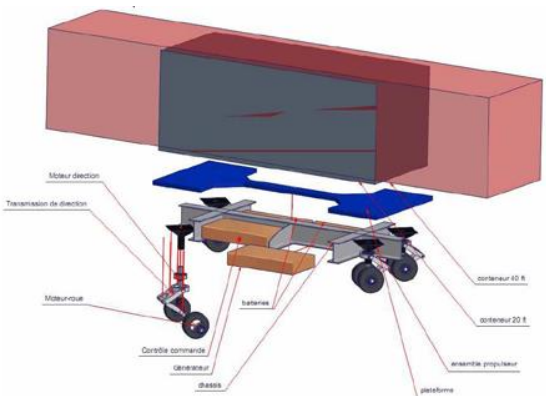


Figure 1: Suivi de véhicule

2. STAGE DE 4EME ANNEE CHEZ OKTAL-JAPON

A) LE GROUPE OKTAL

OKTAL est une filiale du groupe SOGECLAIR, spécialisé dans l'ingénierie de haute technologie. Il constitue le pôle d'activité simulation en concevant des simulateurs ferroviaires, automobiles et aéronautiques. Le groupe est un acteur majeur de la simulation depuis 1989. Ils interviennent en tant que maître d'œuvre de simulateurs complets ainsi qu'éditeur de logiciels de simulation. SCANeR Studio est un logiciel de simulation développé par le groupe.

B) SUJET DU STAGE CHEZ OKTAL-JAPON

Dans le cadre de la 4ème année de cycle ingénieur à Polytech Lille, nous effectuerons un stage de 3 mois dans la société OKTAL-JAPON située à Tokyo. Lors de ce stage, nous travaillerons respectivement sur une plateforme de simulation à 3 et à 5 degrés de liberté. Ces deux simulateurs fonctionnent sous le logiciel SCANeR Studio. Nous serons chargés de préparer le simulateur à 3 degrés de liberté pour le présenter au JSAE 2014 (Japanese Society of Automotive Engineers) à Yokohama au Japon. A l'issue de ce salon, nous apporterons des améliorations techniques à ce simulateur et travaillerons sur la version à 5 degrés de liberté. Afin de préparer au mieux ce stage, nous nous formerons à SCANeR Studio durant ce projet et travaillerons sur une simulation Hardware-in-the-loop qui est une méthode de simulation caractérisée par l'association de véritables composants, connectés à une partie temps-réel simulée. Ici, le composant connecté à la simulation est le véhicule intelligent et autonome Robucar.

DEFINITION DES OBJECTIFS DU PROJET

Pour réaliser notre projet, nous suivrons la chronologie suivante :

- Modéliser le parking de Polytech Lille dans le logiciel SCANeR Studio
- Créer un serveur pour rendre possible des échanges de paquets entre le PC intégré au Robucar et le PC Supervision
- Récupérer, stocker et envoyer des consignes de vitesse et de braquage des roues provenant du Robucar
- Concevoir une API qui permet de récupérer et d'appliquer ces consignes au véhicule dynamique simulé dans SCANeR Studio.

II. DESCRIPTION DES OUTILS DE TRAVAIL

1. LE LOGICIEL SCANER STUDIO

Pour ce projet, nous avons utilisé le logiciel SCANeR Studio qui est un logiciel de simulation Temps réel modulaire offrant la possibilité d'ajouter et d'utiliser différents modules tels que des véhicules, des terrains, différents types de routes etc.

Les principales caractéristiques du logiciel sont les suivantes:

- Architecture distribuée : toutes les ressources ne sont pas situées au même endroit
- Communication par Ethernet
- Conçu pour de nombreux environnement (Windows XP / 7 / Vista, RT OS)
- Applications de supervision et de conduite
- Possibilité d'intégrer des API (Application Programming Interface)
- Simulation de fichier XML

Le logiciel comporte 5 modes:

- Terrain : permet de modéliser un environnement
- Véhicule : permet de sélectionner, de créer ou d'importer un véhicule dans la simulation
- Scénario : permet d'établir un scénario avec un Terrain et des Véhicules et de choisir des API
- Simulation : permet de simuler le Scénario en Temps réel
- Analyse : permet de récupérer des données de la simulation

OKTAL propose deux versions du logiciel dédiées respectivement à la conduite et aux tests.

2. LE VEHICULE ROBUCAR

Le Robucar est un véhicule automatique électrique, intelligent et autonome commercialisé par la société française ROBOSOFT. Il est destiné à évoluer dans des milieux sains ou hostiles pour réaliser des opérations spécifiques telles que des déminages ou des transports dangereux. Le véhicule comporte un ordinateur intégré et une architecture parallèle permettant le développement d'applications temps réel. En effet, il comprend une carte d'acquisition Temps Réel DSPACE qui récupère les données du véhicule.

Figure 2: Véhicule Robucar

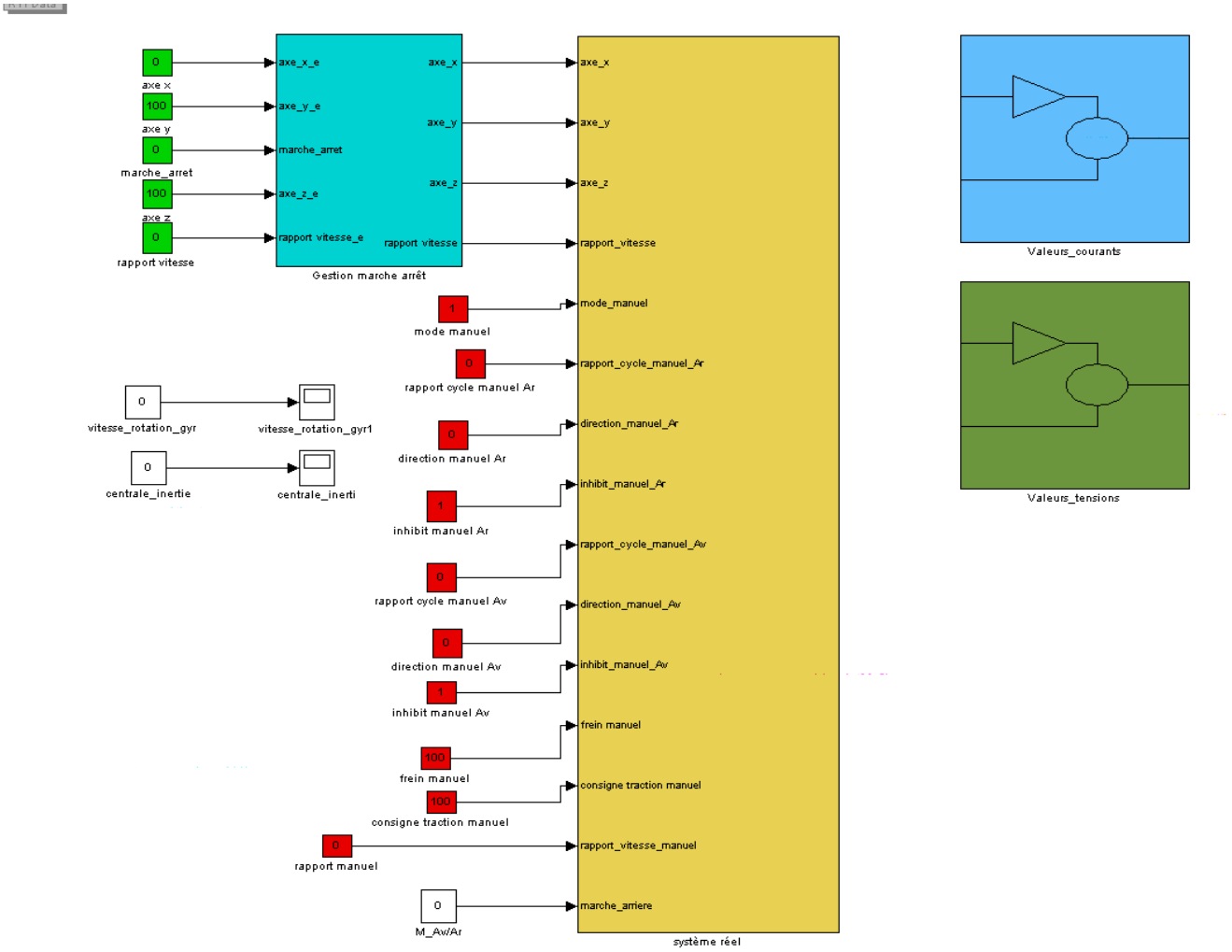
Ses principales caractéristiques techniques sont les suivantes :

Caractéristiques	Robucar RV100	Robucar RV200
Vitesse maxi de déplacement	15 km/h	18 km/h
Vitesse maxi dans un virage à 90°	6 km/h	8 km/h
Poids à vide (avec batteries)	3400 N	3400 N
Poids en charge maxi	5400 N	6400 N
Garde au sol à vide	180 mm	180 mm
Garde au sol en charge maxi	100 mm	80 mm
Vérin électrique de direction	DA12-05A65	?

Le véhicule se compose d'un châssis à quatre roues motrices et directrices pilotables séparément. Pour chaque roue, il y a, par conséquent, deux paramètres de commande : la vitesse et l'angle de braquage. Ce sont ces deux grandeurs que nous chercherons à récupérer et à introduire dans la simulation sous SCANeR Studio.

Pour piloter le véhicule, il faut un programme MATLAB associé à un schéma Simulink. Nous utiliserons le schéma global suivant :

Figure 3: Schéma Simulink du Robucar



Dans le bloc central, on trouve plusieurs sous blocs dont un réalisant la commande de la vitesse (Annexe 1) et un autre celle de l'angle de braquage des roues (Annexe 2). On y retrouve intégré un régulateur de type PID pour corriger les deux consignes à envoyer.

Pour exécuter le programme MATLAB relié au schéma Simulink ci-dessus, nous utilisons le logiciel ControlDesk permettant de lancer l'application de contrôle du véhicule. Sur cette application, on voit changer en temps réel les différentes données du véhicule. Voici un aperçu de deux onglets de l'application où l'on peut voir se modifier l'angle de braquage et la vitesse du Robucar :

Figure 3 : Poste de pilotage du Robucar dans Control Desk

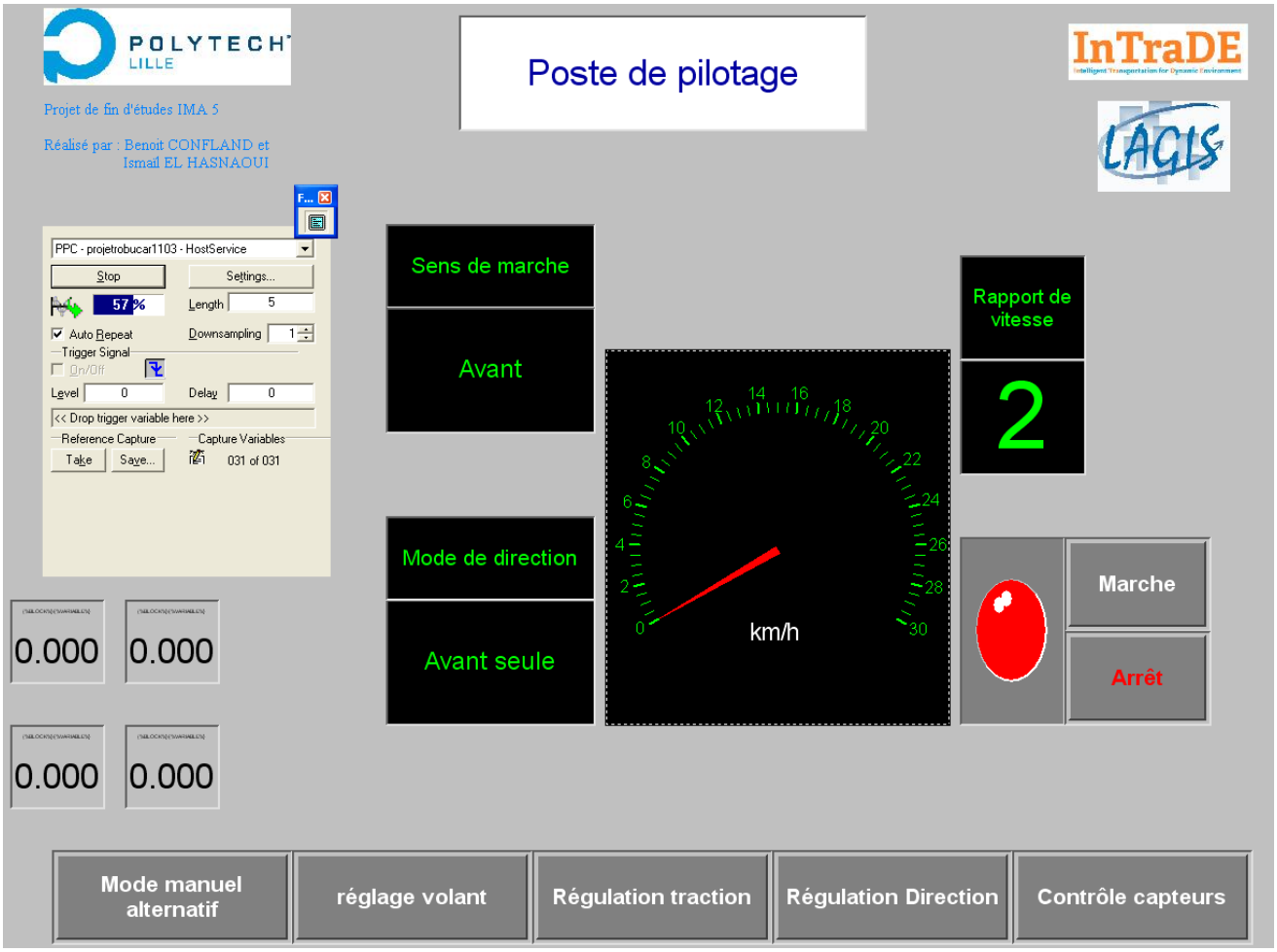
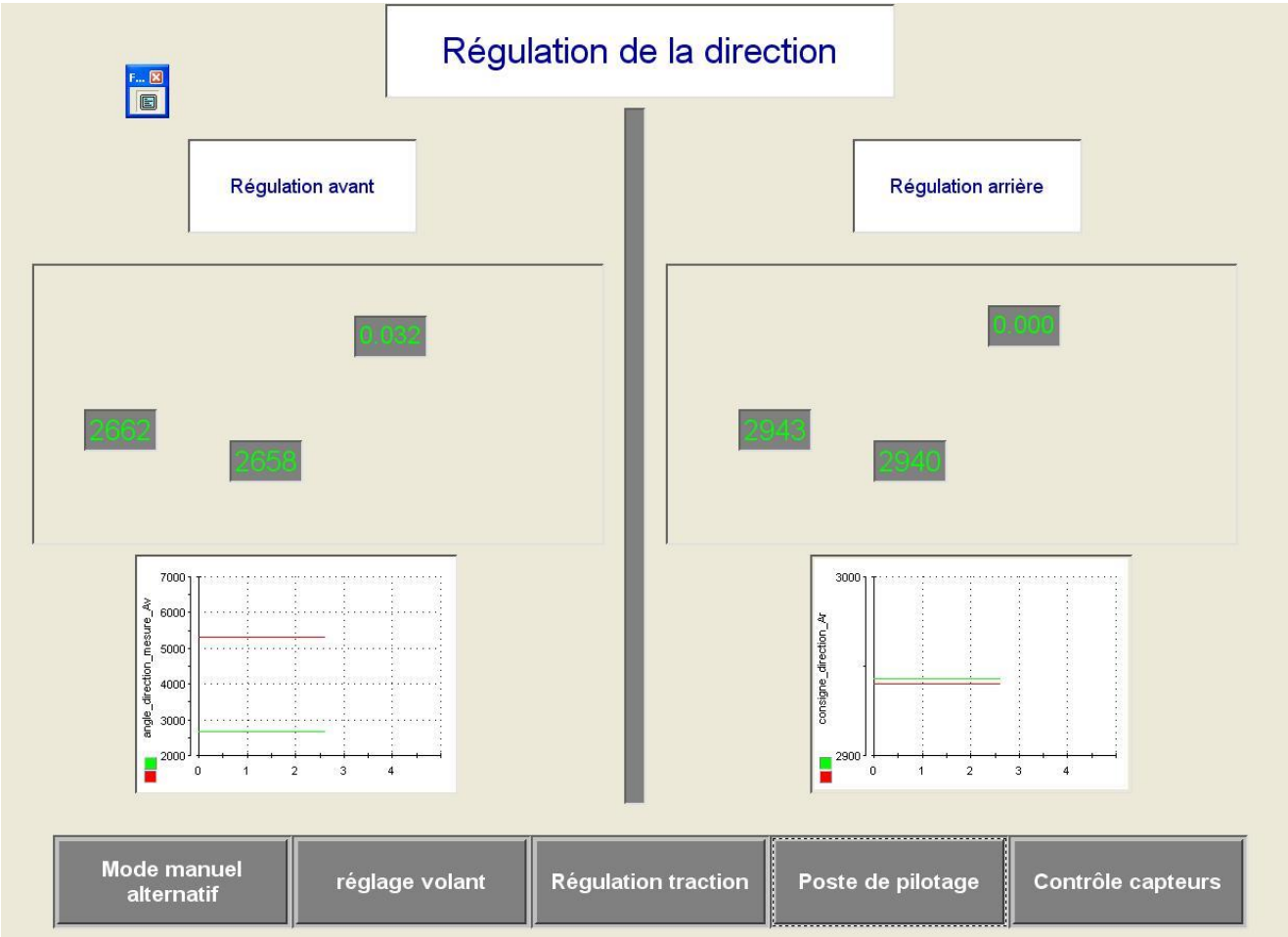
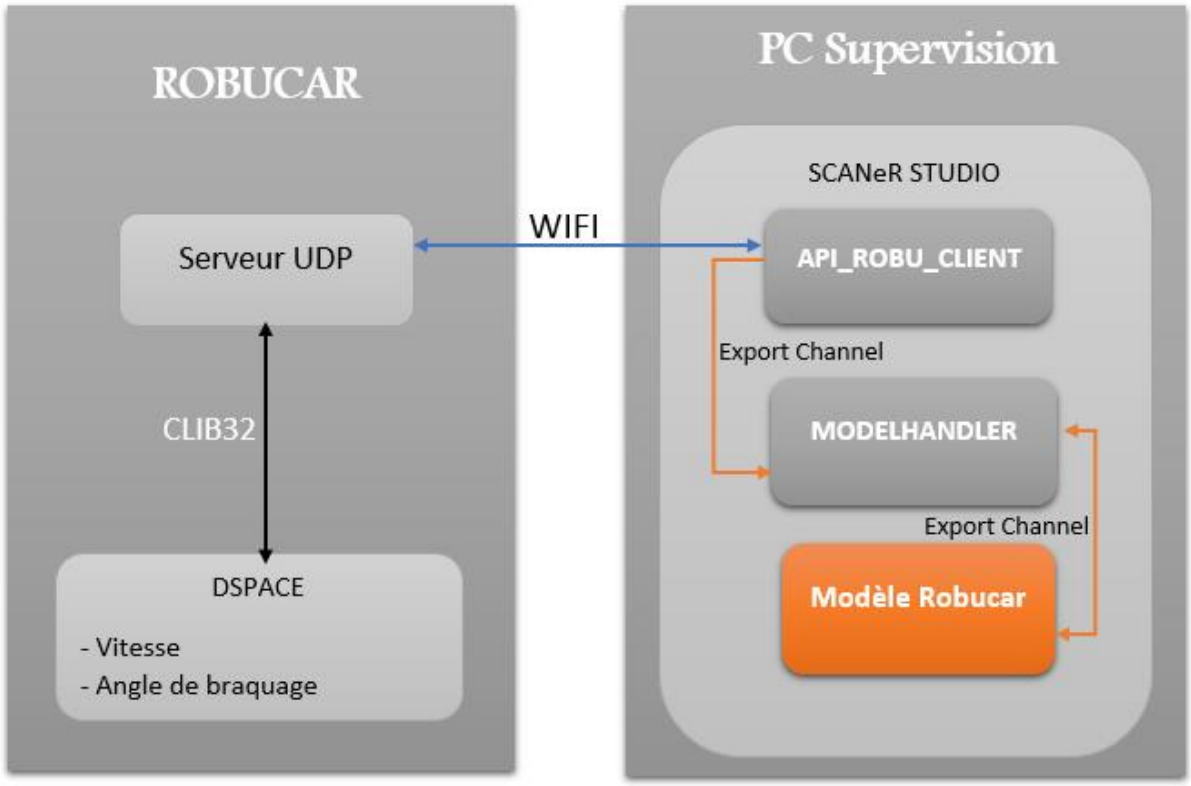


Figure 4 : Logiciel Control Desk: régulation de la vitesse



III. REALISATION DU PROJET

Voici le schéma de synthèse pour la réalisation de notre projet :



Celui-ci met en évidence les deux principales composantes de notre système et le lien qui existe entre elles.

1. MODELISATION DU PARKING DE POLYTECH LILLE

Pour notre simulation, il nous faut créer un environnement dans le logiciel. Afin de modéliser avec précision les routes du parking de l'école, nous utilisons une image satellite de celui-ci que nous insérerons en arrière-plan de notre terrain.

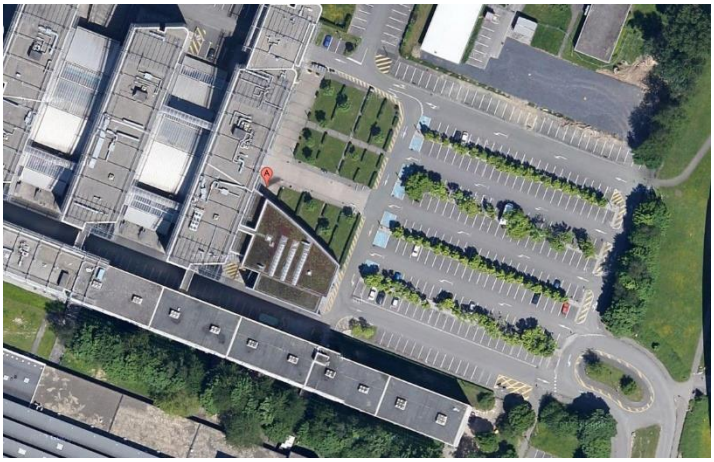


Figure 5: Image satellite du parking

Figure 6: Modélisation du parking

Le mode Terrain met à disposition un certain nombre de routes différentes permettant de définir les caractéristiques telles que le sens de circulation, le nombre de voies, des Cédez-le-Passage etc. Le comportement des véhicules dépend de ces caractéristiques. Dans ce terrain, on peut ajouter des éléments en 3 Dimensions tels que des arbres, des piétons ou des bâtiments. Nous n'avons pas priorisé l'aspect esthétique de ce terrain donc nous nous sommes concentrés sur de simples routes offrant une vue dégagée, optimale pour notre application.

2. LE MODE VEHICULE DE SCANER STUDIO

Dans SCANeR Studio, le mode véhicule propose un certain nombre de véhicules de tous types (Bus, voiture, poids lourds, 2 roues …) et contrôlés selon différents modes (Simple, Advanced etc). Dans notre environnement, nous avons inséré un véhicule de type Advanced et plusieurs véhicules de type Simple. Les véhicules Avanced sont pilotés par l'API « ModelHandler » et les véhicules simples sont pilotés par l'API « Traffic ». L'API « Traffic » permet de commander des véhicules de façon autonome : c'est le logiciel qui gère le trafic. Avec l'API « Modelhandler », les véhicules sont pilotés par un élément extérieur qui peut être, à titre d'exemple, un cockpit de simulation, un joystick ou, dans notre cas, un véhicule réel. Ici, nous n'utiliserons pas le modèle de simulation du Robucar déjà implémenté car celui-ci n'est pas actuellement opérationnel. Nous choisissons le véhicule Advanced « LargeFamilyCar ».

3. REALISATION D'UN SERVEUR ET D'UNE API CLIENT

A) CREATION D'UN SERVEUR UDP

Afin d'établir une connexion entre le Robucar et le PC Supervision du Cockpit de simulation où sera lancé SCANeR Studio, nous avons créé un serveur UDP permettant l'échange de paquets. Nous avons préféré le protocole UDP au protocole TCP car il est plus rapide bien qu'il soit moins fiable. SCANeR Studio étant un logiciel de simulation Temps Réel, nous avons choisi de porter la priorité sur l'aspect temps réel et non sur la précision.

Ce serveur est chargé d'envoyer à un client, les consignes du Robucar en les récupérant de la DSPACE. Pour pouvoir lire ces données dans le programme MATLAB, nous devons utiliser un parser. Ce parser prend en paramètre les noms des variables du schéma Simulink et nous renvoie leur nom système correspondant, exploitable en C++. Pour cela, il faut indiquer au parser le chemin (path) et le nom exact des variables dans Simulink. Dans le serveur UDP, on récupère donc grâce à la librairie « Clib32 » les valeurs de la vitesse et l'angle de braquage des roues du Robucar.

Une fois ces informations récupérées, le serveur implémente une structure de données comportant les champs « Vitesse » et « angle de braquage ». Il initialise une « socket » et envoie un paquet contenant les deux champs de la structure à un client qui les réceptionne. Le serveur comptera également le nombre de paquets perdus ce qui permettra d'arrêter la simulation si la connexion est perdue ou si le manque de précision devient trop important. Le client, ici, n'est autre que l'API que nous avons développé et implanté dans SCANeR Studio. Elle se charge de récupérer les données reçues et de les appliquer.

B) CREATION D'UNE API CLIENT POUR SCANER STUDIO

Nous avons créé une API « API_ROBU_CLIENT » qui envoie des consignes de vitesse et d'angle de braquage des roues à l'API ModelHandler. Cette dernière se charge de les appliquer au véhicule modélisé. Pour donner des consignes à ModelHandler, nous devons écrire sur des canaux : les « Export Channels ». Ici, nous utilisons six canaux correspondants à six informations et consignes à transmettre. Pour connaître l'emplacement ou écrire les consignes, on se réfère au fichier Shm.html qui liste tous les canaux utilisables, numérotés et désignés par un nom.

Ici, nous irons écrire sur les canaux suivants :

- Accelerator : Accélération des roues
- SteeringWheel : Angle de braquage des roues
- GearBoxAutoMode : Mode automatique de la boîte de vitesse
- WantedGear : Numéro de vitesse désirée
- IgnitionKey : Etat du moteur
- TimeOfUpdate : Synchronisation des 2 API

0	CabToModel	From Cabin to model	Accelerator	double		throttle pedal position (range: from 0 to 1, 0 is depressed)
			Brake	double	N	brake force.
			Clutch	double		clutch pedal position (range: from 0 to 1, 0 is depressed)
			ParkingBrake	double	N	parking brake force.
			Fowa	boolean		exhaust brake (for truck only) (range: false : off, true : on)
			Telma	double		electroMagnetic brake position (for truck only) (typical range: from 0 to 4)
			WantedGear	short		wanted gear (range: 0 to max_gear), used if GearBoxAutoMode is 0
			GearBoxAutoMode	short		gear box mode (range: 0 (TAutoModeManual), 1 (TAutoModeReverse2), 2 (TAutoModeReverse1), 3 (TAutoModeNeutral), 4 (TAutoModePark), 5 (TAutoModeLimit1), 6 (TAutoModeLimit2), 7 (TAutoModeLimit3), 8 (TAutoModeLimit4), 9 (TAutoModeRace), 10 (TAutoModeDrive), 11 (TAutoModeOverDrive))
			IgnitionKey	short		ignition key state (range: 0 (ENGINE_OFF), 1(ENGINE_CONTACT), 2(ENGINE_ON), 3 (ENGINE_START))
			Indicators	short		indicators position (range: -1 (RIGHT), 0 (OFF), 1 (LEFT))
			Lights	short		lights state (range: 0 (off), 1(position lights ON), 2 (dipped lights ON), 3 (full lights ON))
			Wipers	short		wipers state (range: 0 (off), 1(sometimes), 2 (low speed), 3 (high speed))
			Horn	boolean		horn state
			Warnings	boolean		warnings lights state
			FrontFogLights	boolean		front fog lights state
			RearFogLights	boolean		rear fog lights state
			TrailerBrake	double		trailerBrake state (for truck only) (range: from 0 to 1, 0 is released)
			RVLVState	short		speed limiter and cruise control (range: 0 (RV_ON), 1 (RVLV_OFF), 2 (LV_ON)
			RVLVSpeed	double		speed limiter and cruise control value
			TimeOfUpdate	double	s	Time when the value in this struct was written
5	CabToSteering	From Cabin to model	IsTorqCommand	boolean		if true apply torque value
			SteeringWheel	double		steering wheel angle (in rad, positive to the left)
			SteeringWheelSpeed	double		steering wheel angular speed (in rad/sec)
			SteeringWheelAccel	double		steering wheel angular acceleration from the wheel(rad/s2)
			SteeringWheelTorque	double		steering wheel torque from the wheel(N.meters, SAE convention : positive to turn left)
			DistanceOnTraj	double	m	Distance on trajectory
			CurrentTrajError	double	m	Trajectory offset
			NearestTrajCurvature	double	m-1	Ideal curvature
			NearestPosOnTrajX	double	m	x value of the nearest point of the ideal trajectory
			NearestPosOnTrajY	double	m	y value of the nearest point of the ideal trajectory
			NearestPosOnTrajZ	double	m	z value of the nearest point of the ideal trajectory
			NearestTrajDirectionX	double		Nearest Traj Direction X
			NearestTrajDirectionY	double		Nearest Traj Direction Y
			NearestTrajDirectionZ	double		Nearest Traj Direction Z
			ExtrapolatedPositionX	double	m	x value of the pilot prediction point
			ExtrapolatedPositionY	double	m	x value of the pilot prediction point
			ExtrapolatedPositionZ	double	m	x value of the pilot prediction point
			TimeOfUpdate	double	s	Time when the value in this struct was written

Figure 7: Extraits de Shm.html

Les commandes envoyées sur les canaux sont :

```
Com_setDoubleData(CabToModelOutput,"Accelerator", OutputData.accelerator);
Com_setDoubleData(CabToSteeringOutput,"SteeringWheel",OutputData.vehiculewheelangle);
Com_setShortData(CabToModelOutput, "GearBoxAutoMode", 0);
Com_setShortData(CabToModelOutput, "WantedGear", 1);
Com_setShortData(CabToModelOutput, "IgnitionKey", 3);
Com_setDoubleData(CabToModelOutput, "TimeOfUpdate",Process_GetTime());
```

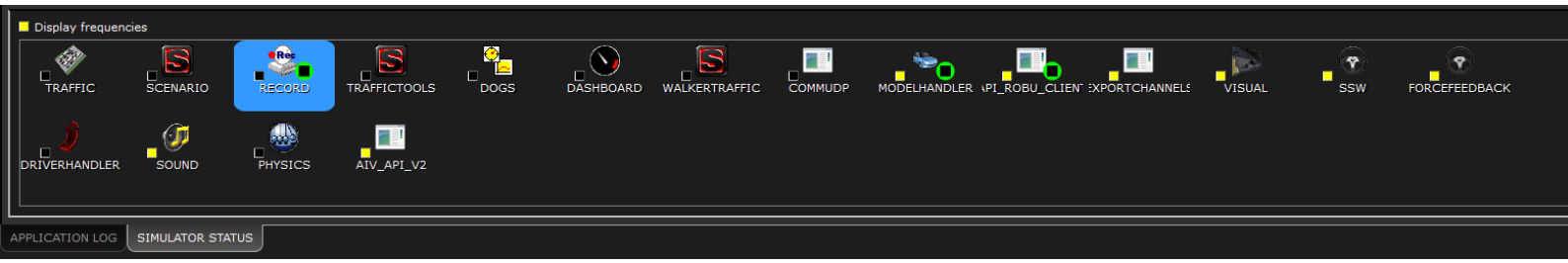
On impose 0 sur “GearBoxAutoMode” pour gérer la boîte de vitesse manuellement.
On impose 3 sur “IgnitionKey” pour démarrer le moteur (Start Engine)
On impose 1 sur “WantedGear” pour démarrer en vitesse numéro 1
On met à jour “TimeOfUpdate” pour permettre la synchronisation de l'envoi de données entre ModelHandler qui fonctionne à 500Hz et notre API qui fonctionne à 100Hz.

4. LE MODE SCENARIO DE SCANER STUDIO

Dans le mode scénario, nous allons indiquer au logiciel quelles API nous intéressent pour notre application. Ici, nous utiliserons « ModelHandler », « EXPORTCHANNELS », «RECORD» et notre propre API : API_ROBU_CLIENT.

L'API « EXPORTCHANNELS » permet de visualiser les données circulant dans les canaux utilisés par la simulation. L'API « RECORD » permet d'enregistrer certaines informations du véhicule modélisé telles que l'évolution de la vitesse et de l'accélération ou encore la variation de la boîte de vitesse. Cette API, comme ExportChannels, est facultative mais elles nous sont utiles pour comprendre le comportement du modèle dans la simulation. Une fois les API sélectionnées, on passe à l'étape de la simulation.

Figure 8 : Sélection des API



5. LE MODE SIMULATION DE SCANER STUDIO

Dans ce mode, les différentes API s'exécutent et on peut voir le modèle évoluer en temps réel. Dans la simulation, on peut se placer selon différents angles de vue permettant ainsi de superviser facilement notre véhicule. Nous adopterons les modes Driver View et Top View matérialisant respectivement le point de vue du conducteur et la vue du dessus du véhicule.

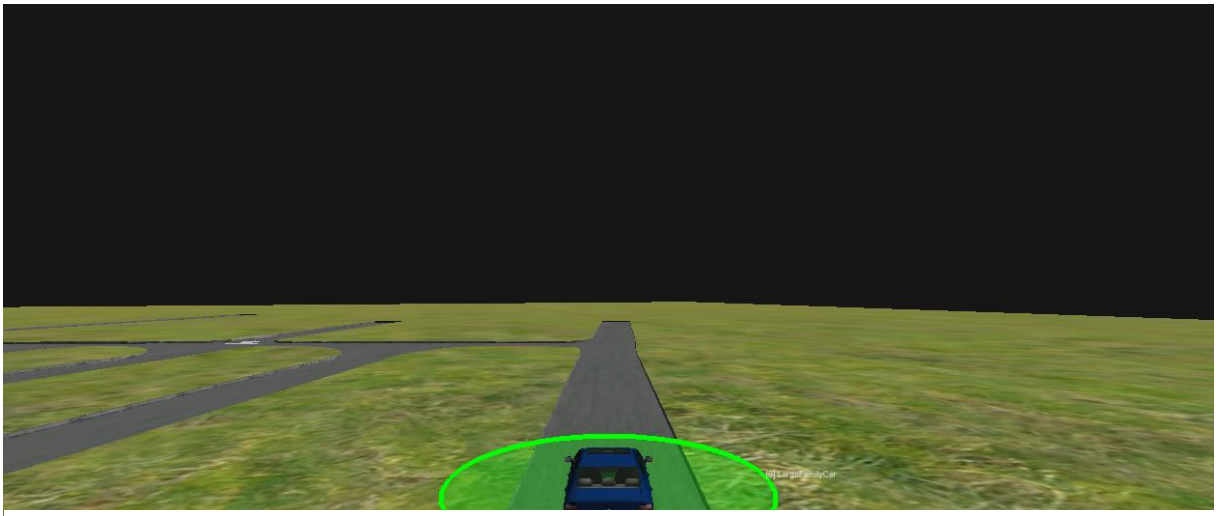


Figure 9 : Driver View dans SCANeR Studio

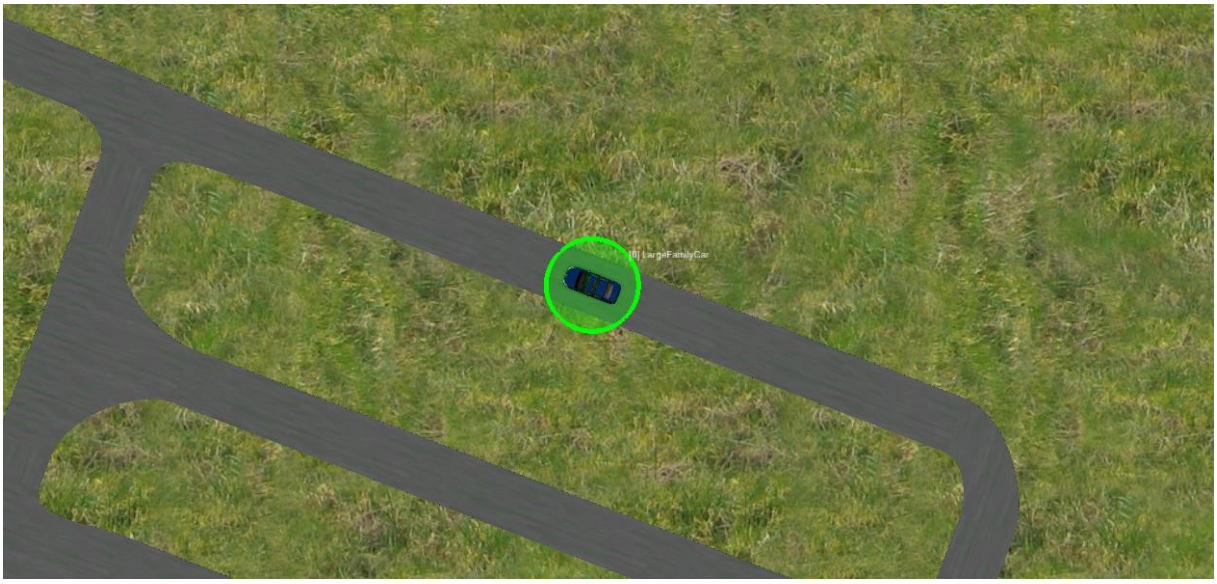


Figure 10 : Top View dans SCANeR Studio

Comme les API sont en cours de fonctionnement dans ce mode, on peut, par exemple, observer en temps réel les données circulant sur les canaux grâce à l'API Export Channels. Une fois la simulation arrêtée, on peut observer les enregistrements des données du véhicule durant la simulation avec l'API Records. Ici, on prêtera attention à l'évolution de la vitesse et à la direction des roues. Ici, le véhicule se déplace dans la simulation (cf vidéo de présentation) lorsque l'on pilote le Robucar avec une vitesse et/ou un angle de braquage.

Les paramètres de vitesse et d'angle des roues doivent être réajustés dans la simulation par rapport aux consignes enregistrées dans le modèle MATLAB. Le Robucar fourni un angle compris entre 2242 (angle de 30° à gauche) et 3002 (angle de 30° à droite). De ce fait, le déplacement tout droit correspond à une valeur de 2622. Dans la simulation, la direction tout droit correspond à zéro. Pour tourner à gauche, on veut un angle en radian positif et pour tourner à droite un angle négatif. Dans la mesure où la variation de l'angle en fonction de la position du volant est sensible, on choisit un intervalle $I \in [2600; 2670]$ pour lequel on ordonnera au véhicule simulé d'aller tout droit. Le but étant de palier une position neutre du volant pas toujours identique. Le calcul de l'angle pour tourner à gauche ou à droite est réalisé selon une règle de 3.
Nous avons également réajusté la vitesse du véhicule dans la simulation. En effet, la consigne de vitesse envoyée par le Robucar représente la moyenne des vitesses des roues avant et arrière. Or les roues arrière ont une vitesse plus faible que les roues avant. Notre véhicule est en effet utilisé en traction avant et étant sur cales pour des raisons pratiques, les roues arrière tournent très lentement. De ce fait, nous avons calculé un coefficient permettant d'ajuster la vitesse uniquement selon la vitesse des roues avant. Ici : $Coefficient = 1{,}34$.

VI. PROBLEMES RENCONTRES

1. PROBLEMES DE CONNECTIVITE

Initialement, nous avons créé un réseau sans fil Ad-hoc pour ouvrir la connexion entre le Robucar et le PC de supervision. Cependant, même à très courte distance entre le véhicule et le PC, la connexion n'était pas stable et se coupait très rapidement, empêchant une simulation convenable. De ce fait, nous avons configuré un routeur pour connecter les deux éléments. La connexion était alors stable mais le débit lent ce qui occasionnait des erreurs de direction très importante dans la simulation. Nous avons, de ce fait, relier le Robucar au routeur par un câble Ethernet et les résultats sont nettement plus satisfaisants. Nous conserverons donc cette configuration car nous privilégions l'aspect temps réel de la simulation à l'aspect pratique bien que la liaison Ethernet entre le véhicule et le routeur n'est pas gênante.

2. ABSENCE DE COCKPIT DE SIMULATION

L'intérêt du projet reposait en parti sur l'utilisation du cockpit de simulation de la salle C002 pour permettre la surveillance du véhicule par son biais. Cependant, le cockpit a été envoyé entre temps à l'une des sociétés françaises OKTAL pour y mettre à jour entre autre la version de SCANeR Studio. Malheureusement, il n'a pas été ramené à temps pour la fin du projet donc nous n'avons pas pu l'utiliser. Par ailleurs, l'API de contrôle du Cockpit étant disponible sur le PC Supervision, nous avons pris connaissance du programme et nous avons réfléchi à l'amélioration de notre API pour y intégrer l'envoi de consignes de vitesse et de braquage à celui-ci.

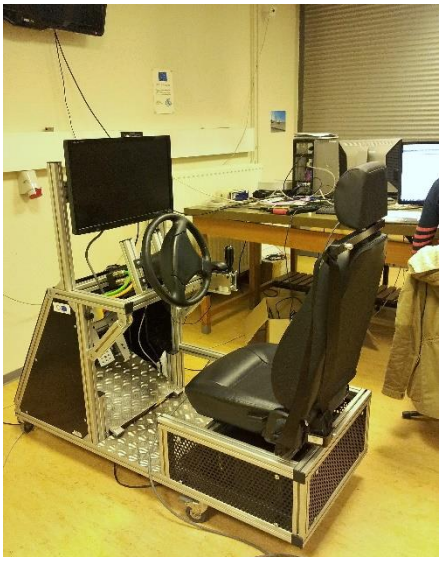
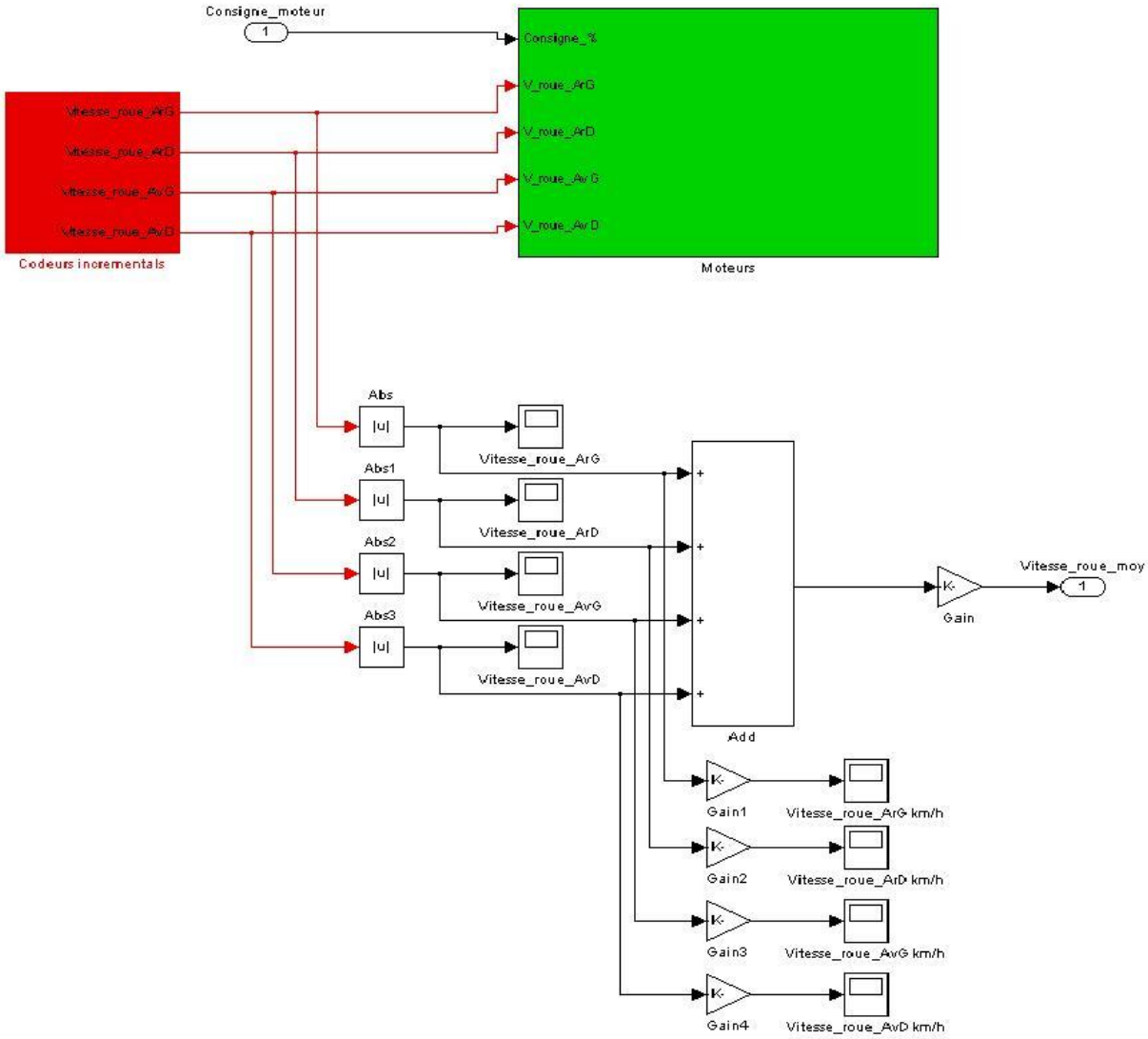


Figure 11 : Cockpit de conduite en C002

CONCLUSION

En utilisant un cockpit de conduite tel que celui de la salle C002 à Polytech Lille, la surveillance d'un véhicule est possible à courte distance. En effet, la connectivité actuelle ne permet pas la communication entre le cockpit (ou le PC supervision ici) et le Robucar. Par ailleurs, si on palie à ce problème, on pourrait imaginer le guidage du véhicule dans des zones confinées où la perception d'obstacles ou détection de collision par le conducteur est difficile. On pourrait également utiliser le télémètre laser pour la détection d'obstacles et envoyer, de la même façon, les informations à l'opérateur sur le cockpit de simulation.

ANNEXE1 : SCHEMA BLOC DE LA VITESSE DES ROUES



ANNEXE 2: SCHEMA BLOC POUR L'ANGLE DE BRAQUAGE DES ROUES