Contribution à la sécurisation du véhicule autonome: Modélisation comportementale avec AltaRica

Mohamed Tlig, Mathilde Machin, Romain Kerneis, Emmanuel Arbaretier, Linda Zhao, Florent Meurville, Jean van Frank

Contribution à la sécurisation du véhicule autonome: Modélisation comportementale avec AltaRica

Contribution to the safety of autonomous vehicle: Behavioral modeling with AltaRica

Mohamed TLIG*, Mathilde MACHIN†, Romain KERNEIS*, Emmanuel ARBARETIER†, Linda ZHAO‡, Florent MEURVILLE§, Jean VAN FRANK*

* *IRT SystemX – dept. transport autonome* – Paris-Saclay – Email: firstname.lastname@irt-systemx.fr
† *APSYS AIRBUS Group* – Blagnac – Email: firstname.lastname@apsys-airbus.com
‡ *SECTOR – dept. Functional Safety* – Villebon-sur-Yvette – Email: firstname.lastname@sector-group.net
§ *VALEO– dept. Functional Safety and Modeling* – Créteil – Email: firstname.lastname@valeo.com

Résumé

L'objectif d'insérer des véhicules autonomes dans la circulation en 2020 représente un véritable défi technologique. Afin d'arriver à un véhicule autonome sûr, de nouvelles approches de conception, de validation et de sécurité sont nécessaires. Au sein de l'IRT SystemX[2] et du projet SVA[1] (Simulation pour la Sécurisation du Véhicule Autonome), nous développons une approche afin de dépasser les limitations des méthodes existantes.

L'objectif est de fournir aux concepteurs les méthodes et les outils pour analyser la sécurité de leur système pendant les phases de conception et de validation des fonctions autonomes.

Dans ce papier, nous utilisons le MBSA (*Model Based Safety Analysis*) sur un ADAS (*Advanced Driver-Assistance System*). Nous décrivons les différentes activités de modélisation et de simulation nécessaires. Les résultats de cette approche sont présentés et discutés.

Mots-clés

Système de conduite autonome, Système autonome critique, Sécurité fonctionnelle, ISO26262, MBSA, AltaRica

Summary

The objective to introduce autonomous vehicles by 2020 on the roads represents a real technological challenge. It requires dismiss with traditional design, security and validation processes to achieve a safe system. As part of the SVA[1] (Simulation of Autonomous Vehicle Safety) project, we present the process under development at the Institute for Technological Research SystemX[2], in order to optimally address the limitations of existing methods.

The objective is to provide designers methods and tools to support safety considerations during the design and the validation phases of autonomous vehicles functions.

In this paper, we apply a Model Based Safety Analysis methodology (MBSA) to an Advanced Driver-Assistance System (ADAS). We describe the different activities carried out during each stage. Finally, The results of this approach are presented and discussed.

Index Terms

Autonomous Driving Systems, Critical Autonomous Systems, Functional Safety, ISO 26262, MBSA, AltaRica

1 Introduction

Etudier, argumenter et valider un certain niveau de sûreté de fonctionnement et de sécurité d'un véhicule autonome est un problème ouvert. La norme automobile ISO 26262 [1] traite essentiellement des fautes internes (modes de défaillance aléatoires matériels et fautes systématiques logicielles). A l'opposé, elle ne fournit pas de pistes pour gérer les problèmes liés aux interactions avec l'environnement et à l'interprétation des données issues des capteurs, qui restent des sujets de préoccupation majeurs pour tout véhicule autonome. De plus, la criticité des situations redoutées est évaluée, entre autres critères, à travers la contrôlabilité du conducteur (humain), ce qui n'est plus pertinent dans le cadre des véhicules autonomes. C'est pourquoi nous proposons d'adapter une méthode particulièrement utilisée en l'aéronautique appelée MBSA (*Model Based Safety Analysis*) pour pour traiter des phénomènes concomitants internes et externes, notamment en ce qui concerne la perception de l'environnement et toutes les interactions possibles.

Le MBSA requiert de construire un modèle du système avec un haut niveau d'abstraction et permet de générer toutes les séquences minimales menant à une situation dangereuse,

Ces travaux ont été effectués à l'IRT SystemX et donc financés dans le cadre du programme "Investissements d'avenir". Au sein du projet SVA, les partenaires industriels sont : Apsys, Sector, Valeo, Continental, Renault, PSA, AV Simulation, Assystem, All4Tec, Optis. Les partenaires académiques sont : CEA, Université Paris-Saclay, Université de Versailles Saint Quentin en Yvelines, LNE.

[1] https://www.irt-systemx.fr/en/project/sva/

[2] https://www.irt-systemx.fr/en/

ce qui permet d'avoir une approche plus riche et plus détaillée que les démarches par coupes minimales habituellement utilisées en Sûreté de Fonctionnement. De ce fait, le résultat recherché consiste à utiliser le MBSA pour générer des scénarios potentiellement critiques liés à des problèmes d'interaction ou d'interprétation de l'environnement. Ces scénarios sont des ensembles ou séquences de défaillances internes et d'événements représentant des variations issues de l'environnement au sens large. Comme dans toute approche de modélisation, le choix d'un niveau d'abstraction et de granularité oriente la génération des scénarios critiques. Néanmoins, notons que la démarche traditionnelle d'exploitation de simulations basées sur des modèles de conception et couplées avec un environnement de roulage se déroule de la manière suivante : les simulations sont effectuées selon une logique d'exploration prescrite, en vue d'illustrer et de vérifier les propriétés de comportement prévues; cependant, cette logique ne permet pas de favoriser la recherche la plus complète possible en matière de scénario critique pour la sécurité (phénomènes rares).

L'intérêt d'une approche MBSA est de se focaliser sur des classes de scénarios critiques, à travers la sélection de facteurs ayant un impact direct sur des situations dangereuses définies de manière macroscopique. Les algorithmes de calculs remontent ainsi de manière systématique les enchaînements d'événements de comportements susceptibles de produire cette situation dangereuse.

Il y a donc plusieurs avantages :

- la nature « scénario critique » est caractérisée dès le début du traitement;
- l'exploration combinatoire est assurée à travers un cheminement qui « inverse » en quelque sorte la flèche du temps et balaye tous les faisceaux de trajectoires amont possibles;
- et enfin, les résultats de traitements ne sont donc pas «parasités » par l'identification de scénarios non pertinents par rapport aux enjeux de sécurité.

C'est pourquoi, dans un contexte de recherche de "scénarios critiques" , cette méthode apparaît particulièrement efficace pour, d'une part adresser de larges familles de scénarios potentiellement critiques, et d'autre part décrire précisément la trajectoire temporelle de chaque scénario afin d'être capable de les valider un à un en simulation multi-physique. En effet, compte tenu du nombre et de la complexité des facteurs d'influence, cette identification "*manuelle*" faite habituellement dans le cadre de référentiels méthodologiques de type "*APR*[3]" s'avère fastidieuse, voire infaisable. De plus, chaque scénario généré par le MBSA représente en réalité une classe d'instanciations concrètes entièrement définies, telles qu'elles puissent être simulées, ce qui permet de définir des bornes précises dans lesquelles le scénario est effectivement critique, ainsi qu'une efficacité du système conçu par rapport à ce scénario.

Pour illustrer cette méthode, nous l'appliquons à la fonction d'autonomie TJC (*Traffic Jam Chauffeur*), qui peut commander le véhicule dans une situation d'embouteillage, à une vitesse maximum de 70 km/h et sur une voie à chaussée séparée.

Après la présentation de travaux dans les domaines de la sécurité fonctionnelle et de la conduite autonome dans la section II, nous décrivons la modélisation du TJC dans la section III. La section IV présente les résultats de notre approche, en termes de coupes et de séquence.

2 Travaux connexes

Dans ce papier, nous traitons du problème de la conduite autonome et des ADAS. Dans la classification du SAE J3016 [2] décrivant les différents niveaux d'autonomie, nous nous plaçons au niveau 3 : dans certaines conditions de circulation, le conducteur peut céder le contrôle complet du véhicule au système automatisé qui sera alors chargé des fonctions critiques de sécurité. Ces systèmes sont encore peu répandus et les retours d'expérience ne permettent pas d'identifier tous leurs modes de défaillance, ou plus précisément l'identification des scénarios critiques. De plus, les analyses classiques de sécurité comme celles définies dans l'ISO 26262 [1] ne sont plus suffisantes. En effet, elles ne prennent pas en compte les violations des concepts de sécurité qui ne sont pas dues aux défaillances, comme les fautes liées aux limites de fonctionnement des capteurs.

Ces difficultés ont conduit les constructeurs automobiles et leurs ingénieurs à travailler sur le développement de nouvelles approches qui dépassent les limitations des analyses de sécurité habituelles. Par exemple, le PAS 21448 [3] (aussi appelé SOTIF[4]) est une initiative pour gérer les dysfonctionnements de perception en l'absence de fautes matérielles ou logicielles. Ces dysfonctionnements ont pour principales causes l'interprétation des données (par exemple, une reconnaissance de forme erronée), l'interaction avec l'environnement (par exemple l'éblouissement causé par la neige) si ce n'est la combinaison des deux.

Par ailleurs, l'amélioration de la sécurité par la quantification des incertitudes sur les valeurs de sortie des composants a fait l'objet de travaux récents. La plupart se basent sur les outils bayésiens [4] et leurs variantes comme le *deep learning* [5].

Dans la suite, nous introduisons le MBSA, une approche formelle conçue pour l'aéronautique [5] [6].

3 Modélisation

L'objectif de la modélisation est de représenter le TJC et tous les éléments liés (par exemple le véhicule de devant) dans un langage de MBSA afin de générer des scénarios critiques. En particulier, nous cherchons des scénarios qui comprennent des problèmes de perception.

[3] Analyse Préliminaire des Risques

[4] Safety Of The Intended Functionality

[5] Cette méthode a notamment été utilisée pour étudier la sécurité du système hydraulique de la famille des Airbus A320

3.1 Outils utilisés

Pour modéliser le TJC nous utilisons Simfia, développé par Apsys (groupe-Airbus). Cet outil offre une interface graphique conviviale pour construire des modèles dans le langage AltaRica [7], initialement développé au LaBRI[6].

Un modèle est un ensemble de briques dont la structure interne est décrite en figure 1. Les briques ont des connecteurs d'entrée et de sortie par lesquels elles communiquent entre elles. La valeur d'un connecteur de sortie est déterminée par une assertion logique, fonction des connecteurs d'entrée et des variables d'état de la brique. Une variable d'état est une variable interne à la brique qui évolue grâce à des événements. La brique contient donc une machine à état, dont les états sont définis par les valeurs des variables d'état et dont les transitions sont déclenchées par les événements. Enfin, les événements apparaissent selon une loi de probabilité.

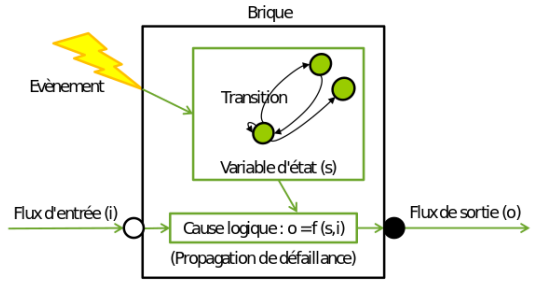

Fig. 1: Une brique AltaRica

Dans l'exemple de la figure 2, un événement "*défaillance*" fait passer la variable d'état *s* de la valeur "*nominal*" à la valeur "*endommagé*". Par l'intermédiaire de la cause logique, le connecteur de sortie prend alors la valeur "*ko*", même si l'entrée de la brique est correcte.

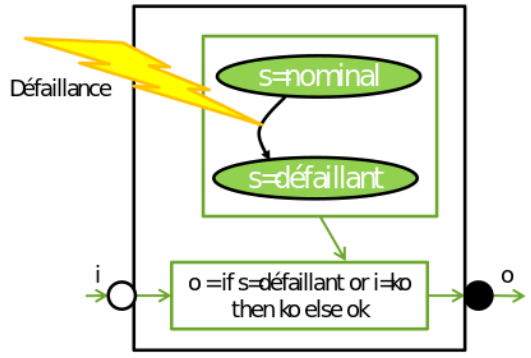

Fig. 2: Occurrence d'une défaillance

Le langage AltaRica permet d'organiser les briques dans une arborescence de décomposition hiérarchique: une brique composite contient d'autres briques (composites ou non).

Etant donné un point d'observation (typiquement une situation redoutée), un compilateur d'AltaRica génère toutes les coupes, c'est-à-dire les ensembles d'événements qui conduisent à la situation redoutée. Le compilateur AltaRica peut également générer des séquences, c'est-à-dire un ensemble d'événement avec un ordre d'occurrence déterminé. Par exemple dans le modèle de la figure 3, où chaque brique a une défaillance simple:

- Les coupes sont {C}, {D} et {A,B};
- Les séquences sont (C), (D), (A,B) et (B,A).

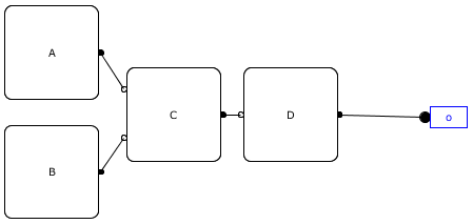

Fig. 3: Exemple de modèle

La modélisation AltaRica est orientée pour la sûreté de fonctionnement. Elle applique un très haut niveau de granularité (par exemple *nominal* ou *défaillant*) mais permet de générer tous les ensembles d'événements critiques sans avoir à simuler chaque scénario indépendamment.

3.2 Structure du modèle

Dans le but de modéliser l'ADAS, nous prenons en compte le véhicule autonome (appelé *véhicule ego* par la suite) et son environnement (les autres véhicules, les conditions météorologiques, etc.). La structure du modèle est la suivante :

- Environnement : tous les éléments de l'environnement intervenant dans le fonctionnement du TJC;
- Perception : capteurs du véhicule ego utilisés par le TJC;
- Fusion : combinaison de différentes informations renvoyées par différents capteurs);
- Commande : décision du mouvement du véhicule ego;
- Conditions d'activation du TJC.

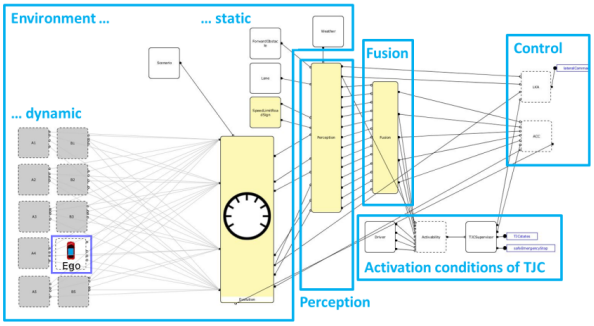

Fig. 4: Vue générale du modèle

La situation redoutée à laquelle on s'intéresse est la collision du véhicule ego avec un autre véhicule. La collision est modélisée par le fait que le véhicule ego et un autre véhicule aient la même position (telle que modélisée dans la section III-D3)

[6]http://www.labri.fr

3.2.1 Environnement: Les éléments de l'environnement considérés sont ceux qui sont utilisés par le TJC ou qui l'influencent, et qui se trouvent dans la limite de la portée de capteurs. L'environnement statique comprend les lignes de marquage, les conditions météorologiques, la signalisation et, possiblement, des obstacles. L'environnement dynamique est constitué de la circulation routière. Le modèle peut contenir jusqu'à deux véhicules dans l'environnement du véhicule autonome. Ce point sera détaillé dans la partie III-D.

3.2.2 Perception: La partie Perception contient les capteurs d'environnement et des capteurs internes (pour la vitesse longitudinale et vitesse de lacet du véhicule ego). Le véhicule ego a pour capteurs d'environnement une caméra et un radar à l'avant. Ils sont dits "intelligents" en ceci qu'ils ne renvoient pas une image (pour la caméra par exemple) mais la position et la nature d'un objet (par exemple une voiture à 10m).

Les obstacles et le véhicule précédant le véhicule ego sont perçus à la fois par la caméra et le radar. La caméra détecte également les lignes de marquage et la signalisation.

3.2.3 Fusion: La caméra et le radar ne sont pas toujours d'accord sur la présence, la position d'un élément d'environnement comme un obstacle. La partie fusion va donc choisir quelle information doit prendre en considération le système, celle venant du radar ou celle venant de la caméra. Dans les systèmes embarqués classiques, la fusion est habituellement effectuée par la sélection d'une source d'information, par priorité ou vote. Pour les systèmes autonomes, la fusion prend en général en compte des niveaux de confiance sur les informations des capteurs, voire des probabilités, approchées qui ne peuvent être modélisées en AltaRica. En conséquence, nous avons choisi de suivre une approche conservative : la fusion prendra toujours la "pire décision" du point de vue de la "safety", la décision la plus dangereuse. Par exemple, si la caméra "voit" le véhicule de devant comme étant proche et le radar le perçoit comme étant loin, la fusion conclura que le véhicule est loin. De la même manière si la caméra perçoit un obstacle que le radar ne détecte pas, la fusion conclura à l'absence d'obstacle.

Comme les algorithmes de fusion ne sont pas modélisés en eux-mêmes, nous avons choisi de modéliser une fusion exempte de défaillance. En d'autres termes, nous considérons que la fusion est validée par une autre méthode et nous ne nous intéressons qu'aux scénarios impliquant l'environnement et la perception.

3.2.4 Commande: De même la commande ne peut pas défaillir. Le modèle contient deux modules de commande :

- L'ACC (Adaptive Cruise Control) commande l'accélération et donc la vitesse longitudinale.
- Le LKA (Lane Keeping Assist) commande la trajectoire latérale.

La partie Commande porte les logiques de décision. Ces logiques sont assez complexes à concevoir car elles doivent assurer la stabilité du modèle, c'est-à-dire :

- lorsqu'il n'y a pas de changement ou dans l'état du système, la consigne doit rester identique,
- lorsqu'il y a un changement, la consigne doit réagir et remettre le système dans un état stable.

Par exemple, si la distance au véhicule de devant se réduit, l'ACC commande de freiner. Les actionneurs ne sont pas considérés car leur technologie est indépendante de l'autonomie et bien maîtrisée.

3.2.5 Conditions d'activation du TJC: Le TJC peut être activé par le conducteur à certaines conditions. Par exemple, le conducteur doit avoir les mains sur la volant et sa ceinture bouclée. Dès qu'une de ces conditions n'est plus remplie, le TJC demande au conducteur de reprendre la main. Si le conducteur ne reprend pas la main après 30s, le TJC déclenche un freinage d'urgence.

3.3 Modéliser la perception

Modéliser l'environnement et sa perception est indispensable pour évaluer la sécurité du TJC. Dans cette partie, nous présentons les artefacts de modélisation que nous utilisons. Ils sont caractéristiques de l'autonomie et suffisamment génériques pour être réutilisés dans d'autres systèmes autonomes.

3.3.1 Types de données pour les éléments d'environnement: Pour les modéliser les éléments de l'environnement statique, qui sont divers et nombreux, nous proposons 3 types de données génériques.

a) Objet sans paramètre: Les objets sans paramètre sont des objets qui peuvent être présents ou non dans l'environnement. Ils sont "sans paramètre" au sens où aucun paramètre n'est pas modélisé même s'ils en ont en réalité. Par exemple, un obstacle sur la voie du véhicule Ego est modélisé comme un objet sans paramètre alors qu'il a une position, une taille, etc. Le fait de ne pas modéliser les paramètres peut avoir différentes explications : les paramètres ne sont pas pertinents pour la sécurité, ou trop de paramètres sont à prendre en compte. Une altération d'un paramètre par la perception est donc modélisée comme un "erroné" global sur l'objet plutôt que sur un paramètre en particulier.

La perception a donc trois modes de défaillance.

- omission : l'objet est présent mais pas détecté,
- commission : il n'y a pas d'objet présent mais un objet est détecté,
- erroné : l'objet présent est détecté avec des paramètres faux.

b) Flux: Un flux est une donnée qui est toujours présente. Par exemple, la vitesse du véhicule ego ou le degré d'hygrométrie de l'air existent toujours. Les modes de défaillance considérés sont alors l'omission et l'erroné.

c) Objet avec paramètre: Pour modéliser les paramètres, nous utilisons un objet sans paramètre et, pour chaque paramètre, un flux. La fusion, pour un objet avec paramètre, se décompose en deux étapes : 1) fusion du signal de présence de l'objet, 2) si la fusion conclut à la présence de l'objet, fusion de chaque flux de paramètre. Pour la détection de l'objet, les seuls modes de défaillance considérés sont l'omission et la commission.

3.3.2 Capteurs, support matériel et fonctions: Pour les véhicules autonomes, les erreurs de perception sont principalement dues à des erreurs d'interprétations des données ou d'interaction avec l'environnement. Dans ces cas, la caméra peut identifier correctement un panneau de signalisation tout en omettant un obstacle. Pour modéliser les capteurs, nous séparons donc les fonctions de perception du capteur (une par objet détectable) de son support matériel. Chaque fonction peut défaillir indépendamment des autres et transitoirement (une réparation est donc modélisée). Nous modélisons également le support matériel du capteur, sa perte causant la perte de toutes les fonctions. Le modèle permet ainsi d'évaluer l'architecture de capteurs et les possibles redondances.

3.4 Modéliser l'effet de la commande et l'environnement dynamique

Nous proposons ici une méthode de modélisation qui prend en compte la circulation.

3.4.1 Les véhicules externes: Dans l'environnement du véhicule ego, nous modélisons deux autres véhicules qui peuvent changer de vitesse et de voie librement. Pour éviter des scénarios irréalistes, nous appliquons néanmoins quelques restrictions sur leurs comportements :

- un véhicule ne peut pas se rabattre sur un autre (Ego compris);
- un véhicule ne peut se rabattre devant ou derrière un autre véhicule que si le véhicule suiveur a une vitesse inférieure ou égale à son prédécesseur;
- un véhicule ne peut pas modifier sa vitesse si cela la rend supérieure à celle du véhicule le précédant ou inférieure à celle du véhicule le suivant.

Etant données ces restrictions, qui impliquent des dépendances entre véhicules, modéliser plus de deux véhicules rendrait le modèle très complexe. C'est pourquoi nous limitons à deux le nombre de véhicules externes considérés dans cette étude.

3.4.2 Bouclage de la consigne ACC et de la vitesse: Pour modéliser la dynamique du véhicule ego inséré dans la circulation, nous avons bouclé la consigne ACC et la vitesse longitudinale du véhicule ego. Lorsqu'un changement apparaît dans l'environnement ou l'état du système, l'ACC fournit une nouvelle consigne et un événement est déclenché pour mettre à jour la valeur de la vitesse. Ainsi nous pouvons visualiser l'effet des logiques de l'ACC et les différentes étapes du mouvement.

3.4.3 Scène routière: Pour permettre la représentation graphique de la scène routière, nous ajoutons des briques d'affichage au modèle. Comme le montre la figure 5, ces briques forment une fenêtre glissante autour du véhicule ego. La fenêtre se déplace à la même vitesse que le véhicule ego qui occupe une case fixe dans la scène (B4).

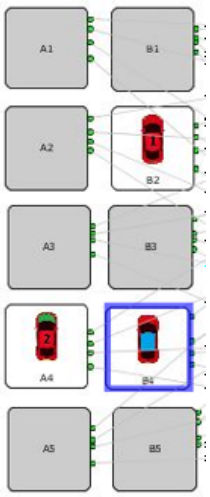

Fig. 5: Exemple de scène routière, le véhicule ego est en bleu, case B4

Cet affichage permet à la fois de mettre au point le modèle, de le valider par la simulation de différents scénarios et enfin de le présenter à un public non-expert en modélisation.

4 RÉSULTATS

Dans cette partie, nous présentons le processus que nous avons construit pour extraire l'information pertinente du modèle, à savoir, des scénarios critiques. Simfia génère les coupes ou les séquences amenant le véhicule dans la situation redoutée de la collision. Une coupe est un ensemble d'événement, dans lequel l'ordre d'occurrence n'est pas considéré, alors qu'une séquence spécifie un ordre d'occurrence. Nous utiliserons les deux générations pour avoir deux niveaux d'interprétation des résultats.

Au sein d'une coupe, le nombre d'événements présents dans la coupe est appelé *ordre*. Plus la coupe comporte d'événements, moins elle est probable donc dangereuse.

4.1 Analyse des coupes

Après 60 heures de calcul, nous obtenons les coupes d'ordre 4 menant à la collision (voir table I).

Afin d'exploiter les résultats de Simfia, nous appliquons trois étapes de post-traitement :

- Filtrage des coupes comprenant un événement de reprise en main pour se concentrer sur les seuls cas où le TJC n'est pas capable de détecter sa propre incapacité à gérer la situation;
- Suppression des événements de changement de position, complètement déterminés par les décisions de changement de vitesse;
- Minimalisation des coupes, c'est-à-dire suppression des coupes incluses dans des coupes d'ordre inférieur.

Ainsi, nous obtenons 39 coupes (comme mentionné dans la table I) que nous classons en 6 catégories :

1) Vitesse de lacet Ego erronée (ordre 1);
2) Détection erronée de la ligne de marquage et perte de la vitesse de lacet (ordre 2) ;

Ordre	Généré par Simfia	Après post-traitement
1	1	1
2	29	6
3	153	19
4	283	13
Total	466	39

TABLE I: Nombre de coupes obtenues pour chaque ordre avant et après post-traitement

3) Vitesse longitudinale Ego erronée et ralentissement du véhicule de devant (ordre 2);
4) Fonction de distance du véhicule de devant erronée sur la caméra et le radar (ordre 2) ;
5) Fonction de distance erronée sur 1 capteur et l'autre capteur en panne ou confiance 0 (ordre 2) ;
6) Commission (détection inadvertante) de ligne par la caméra, fin de ligne de marquage et perte de la vitesse de lacet (ordre 3).

Remarque : La météo n'apparaît pas explicitement dans ces catégories. Cependant, des coupes contenant de la météo sont comprises dans ces catégories. Par exemple, pour la coupe numéro (2), la détection erronée de la ligne de marquage peut être dûe à du brouillard. Les treize coupes d'ordre 4 sont dues à la météo.

La coupe (3) contient un événement qui ne dépend pas du système (ralentissement du véhicule de devant). De ce fait, du point de vue de la conception du système, cette coupe doit être considérée comme une coupe d'ordre 1. De même la coupe (6) doit être considérée comme une coupe d'ordre 2 car la fin de ligne ne dépend pas du système mais de l'environnement.

Au vu de ces éléments, nous constatons que la vitesse de lacet et la vitesse longitudinale de l'Ego sont des éléments critiques du système (car ce sont des coupes d'ordre 1). Il faut donc veiller à la fiabilité de l'ESP qui estime ces informations : l'électrostabilisateur programmé (en anglais Electronic Stability Program, ESP, aussi appelé « ESCo » pour « Electronic Stability Control ») ou encore correcteur électronique de trajectoire, qui est un équipement de sécurité active d'anti-dérapage destiné à améliorer le contrôle de trajectoire d'un véhicule automobile [8].

Dans un deuxième temps, la détection de la distance du véhicule de devant est importante (ordre 2). On note que la détection de la vitesse du véhicule de devant n'est pas représentée dans les coupes car si la distance entre les deux véhicules est faible, peu importe leurs vitesses, l'ACC commande de ralentir.

Enfin, l'importance de la détection des lignes de marquages est mise en évidence. En effet, une non-détection est censée arrêter le fonctionnement du TJC. Cependant, une détection erronée ou une commission et une fin de ligne ne permettent plus la détection de l'erreur. Ces éléments combinés avec une perte de vitesse de lacet (toutes les détections deviennent erronées sur les capteurs) entraînent une collision.

[7]http://www.avsimulation.fr

4.2 Analyse des séquences

Lors de l'analyse des séquences, nous nous demandons si, pour chaque coupe trouvée précédemment, les permutations des événements forment toutes des séquences critiques. Par exemple, si on a une coupe {A,B}, il est intéressant de savoir si les permutations (A,B) **et** (B,A) mènent toutes deux à la collision (donc sont générées par Simfia). On peut alors déterminer si l'ordre d'occurrence des événements est important ou pas. Ainsi, nous constatons que l'ordre d'occurrence intervient dans les séquences suivantes :

- Détection erronée de la ligne de marquage et perte de la vitesse de lacet;
- Commission de ligne par la caméra, fin de ligne de marquage et perte de la vitesse de lacet.

En effet, une perte de la vitesse de lacet en premier entraîne directement une demande de reprise en main. Par contre, une détection erronée de ligne suivi de la perte de la vitesse de lacet n'est pas détectée. La commission de la ligne doit avoir lieu avant la disparition de la ligne pour que le système ne la détecte pas. Et de la même façon, si vient ensuite une perte de la vitesse de lacet, le système n'est pas en mesure de la détecter et cela entraîne une collision.

4.3 Simulations avec SCANeR Studio

Notre approche MBSA, reposant sur un modèle haut-niveau du système, permet de déterminer des scénarios critiques, eux aussi décrits de manière haut-niveau. Afin de les préciser, nous les simulons dans un simulateur plus détaillé, qui permet la représentation précise des positions et des vitesses par exemple. Nous utilisons SCANeR Studio[7].

Comme le montre la figure 6, SCANeR permet de visualiser aisément les situations complexes.

Fig. 6: Simulation avec le module visualisation de SCANeR Studio

5 Conclusion

Dans ce travail, nous présentons un exemple de génération de scénario critique sur un véhicule autonome avec un modèle comportemental fondé sur le MBSA en utilisant le langage AltaRica et le logiciel Simfia. Le modèle produit concerne un système complexe, le TJC développé et utilisé pour les tests dans le cadre du projet SVA. A travers cette approche, il a été nécessaire de l'améliorer pour obtenir des résultats pertinents et exploitables en simulation.

Le véhicule autonome est représenté dans un environnement avec deux autres véhicules, le marquage au sol, les obstacles et les panneaux de limitation de vitesse. Un affichage séparé de la scène est mis en place afin d'interpréter facilement les états des véhicules du modèle. Ces éléments facilitent l'adaptation et l'enrichissement du modèle. C'est ainsi que les coupes et les séquences sont générées par rapport à la situation "collision" et que les résultats sont analysés en identifiant les éléments critiques du système. Enfin, quelques exemples de séquences sont simulés pour les besoins de démonstration. L'approche est synthétisée par la figure 7.

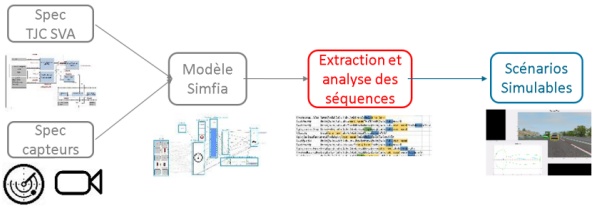

Fig. 7: Vue générale de la méthode

Une perspective de valorisation de ces travaux consiste à développer la notion de post-traitement ou traitement simultané pour obtenir les scénarios pertinents et s'affranchir de quantités de scénarios non pertinents, voire impossible à exécuter qui alourdissent les sorties et brouillent la lisibilité des résultats. Une autre continuation de ces travaux consisterait à analyser les cas de collisions avec demande de reprise en main. Plus largement, on peut imaginer de modéliser d'autres environnements, par exemple des environnements de rues avec piétons ou de routes de campagne, et d'autres situations redoutées.

References

[1] ISO 26262, "Road Vehicle – Functional Safety, Working Group TC22 SC32," International Organization for Standardization, Standard, 2011.

[2] SAE J3016, "Taxonomy and Definitions for Terms Related to Driving Automation Systems for on-Road Motor Vehicle," On-Road Automated Driving (Orad) Committee, SAE International, 2016.

[3] ISO/WD PAS 21448, "Road vehicles – Safety of the intended functionality," International Organization for Standardization, Standard, Under development.

[4] D. Geronimo, A. M. Lopez, A. D. Sappa, and T. Graf, "Survey of pedestrian detection for advanced driver assistance systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 7, pp. 1239–1258, July 2010.

[5] M. Rowan, G. Yarin, K. Alex, v. d. W. Mark, S. Amar, C. Roberto, and W. Adrian, "Concrete problems for autonomous vehicle safety: Advantages of bayesian deep learning," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 2017, pp. 4745–4753.

[6] P. Bieber, C. Castel, and C. Seguin, "Combination of fault tree analysis and model checking for safety assessment of complex system," in *Proceedings of the 4th European Dependable Computing Conference on Dependable Computing*, ser. EDCC-4. Springer-Verlag, 2002, pp. 19–31.

[7] A. Arnold, G. Point, A. Griffault, and A. Rauzy, "The AltaRica formalism for describing concurrent systems," *Fundamenta Informaticae*, vol. 40, no. 2, 3, pp. 109–124, 1999.

[8] Esp : Electronic stability program. Car Engineer. [Online]. Available: car-engineer.com