Rapport Projet d'intégration

Abdelmounaim EL AMRANI

Détection et évitement d'obstacles pour le véhicule autonome

G3
Encardré par :
Mr. Gerald DHERBOMEZ
Mr. Alexandre KRUSZEWSKI

Ecole Centrale De Lille
Lille
Mars 2021

14 mars 2021

Table des matières

1 Contexte

Dans le cadre du projet d'intégration en G3, J'ai opté pour le projet qui tourne autour du développement d'un véhicule autonome. Les objectifs de ce projet sont de tester les nouvelles méthodes de détection d'obstacles se basant sur les travaux de la littérature, notamment en Intelligence Artificielle et Deep Learning et de les intégrer et les tester dans un véhicule autonome dans des conditions de roulage réelles.

Ce projet d'intégration est réalisé en collaboration avec le laboratoire Cristal et Renault.

2 Objectif

L'objectif de ce travail est de montrer les résultats/conclusions de mes travaux/recherches durant les 6 mois derniers sur les nouvelles théories et application du Deep Learning pour les véhicules autonomes, ainsi qu'expliquer les différents sous-projet de la 'stratégie d'évitement'. Notamment l'approche d'apprentissage par renforcement pour développer une politique de conduite (Path-planning) qui va permettre au véhicule de se rendre d'un point A vers un point B.

Ce rapport est présenté dans l'ordre suivant : Une bréve introduction sur le développement des véhicules autonomes et leur écosystéme dans la Section 3. Dans la Section 4, je détaillerai les approches de perception, notamment la "Bird's-eye-view" et comment est généré cette représentation. Dans la section 5, on discutera la stratégie d'évitement via l'apprentissage par renforcement. Puis une conclusion/Perspectives/ressenti dans la section 6 et 7.

3 Introduction

Le véhicule autonome est un sujet de recherche en pleine actualité et de grande influence sur le développement économique et social. Aujourd'hui les véhicules autonomes Waymo développés par Google ont dépassé les 3.2 millions de Km de tests devenant ainsi les proches d'une voiture ordinaire. De plus, la grande demande du marché a invité plusieurs nouveaux fabricants tels que : Apple, Nvidia, Tencent, Alibaba, etc à s'investir dans le domaine et à participer au développement de ses technologies.

FIGURE 1: Tesla Autopilot

En effet, l'immense progrès technologique témoigné ces dernières années au niveau du domaine des véhicules autonomes est principalement dû au développement de l'intelligence artificielle, et notamment des méthodes de Deep Learning de plus en plus performantes, et ce grâce à la grande communauté qui s'y intéresse.

Le but des recherches faites pour le développement des véhicules autonomes est d'atteindre un niveau d'autonomie où le système ne nécessite plus aucune interaction de la part du conducteur. On distingue 5 niveaux d'autonomies différents (figure 2).[8]

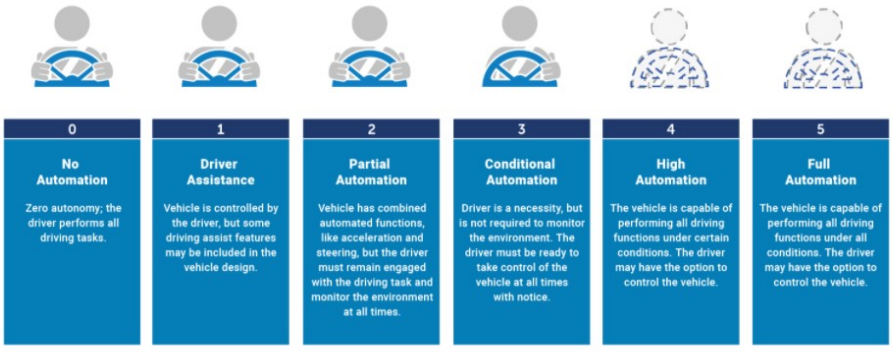


FIGURE 2: Les 5 niveaux d'autonomie [8]

Pour un véhicule autonome donc de niveau supérieur au niveau 3, l'architecture informatique peut généralement être répartie en 4 grands systèmes principaux : Un système de perception d'environnement, un Système de décision autonome, un Système de contrôle et un Système de supervision.[5]

Dans ce qui suit je vais principalement me pencher sur la perception d'environnement et le systéme de décision.

4 Système de perception d'environnement

Le but du système de perception d'environnement est de générer un modèle d'environnement entourant la voiture, c'est-à-dire les positions des obstacles à proximité du véhicule et leurs natures, les positions des feux de circulations des signaux routiers et leurs types, la nature de la route ainsi que toutes autres informations telles que la localisation. L'acquisition de ce type de données est faite grâce à des caméras, des LIDARs et GPS.[2]

4.1 LIDAR Vs Caméra

On distingue principalement 2 approches différentes pour cartographier l'environnement : Vision caméra et Lidar.

LIDAR Ou télédétection par LASER, a pour rôle de situer la voiture dans son environnement, afin de permettre des fonctions telles que le planning du chemin et la facilitation de la navigation. Généralement, les LIDARs sont réputés pour leur prise en compte des obstacles dynamiques et leur courte marge d'erreur par rapport à la localisation satellite.[3]

Disposer de cartes précises est essentiel au succès de la conduite autonome pour l'acheminement, la localisation ainsi que pour faciliter la perception. Il est possible d'obtenir des cartes contenant divers degrés d'information en s'abonnant au service de cartographie disponible dans le commerce. Toutefois, dans les régions où les cartes ne sont pas disponibles, les véhicules autonomes doivent compter sur leur propre capacité de création de cartes pour assurer la fonctionnalité et la sécurité de la conduite autonome.

FIGURE 3: Mapping à partir du Lidar[2]

4.2 Offline/Online Mapping :

Il existe deux types de cartographie : Offline et Online.

— Dans un scénario de *mapping offline*, les données des capteurs sont regroupées dans un lieu centralisé. Les données peuvent être soit des images satellites, soit des données collectées par des capteurs embarqués tels que des caméras ou des lidars. Elles peuvent provenir de plusieurs passages du même véhicule au même endroit ou être fournies par une flotte de voitures. Un rendu de la map est construite hors ligne, et des annotateurs humains sont nécessaires pour annoter les structures sémantiques sur la carte et examiner les résultats finaux. Le service de cartographie traditionnel fonctionne de cette manière hors ligne, puis les cartes annotées sont ensuite servies aux véhicules sur la route. cet approche est hors le spectre de ce rapport.

— Le *mapping Online* se fait à bord du véhicule, en temps-réel, et où il n'y a pas d'intervention humaine. L'approche SLAM (la localisation et la cartographie simultanées) est l'exemple typique. En outre, il existe des approches de mapping basées sur des réseaux de neurones qui fusionnent des séquences d'images monoculaires provenant de plusieurs caméras en une 'Bird's-eye-view'(BEV). Cet BEV donne une représentation pseudo-Lidar, puissante et globale de l'environnement du véhicule.[4]

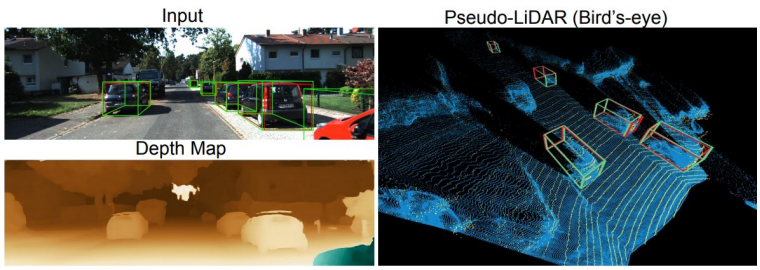


FIGURE 4: Perception pseudo-Lidar [4]

4.2.1 Sensor Fusion

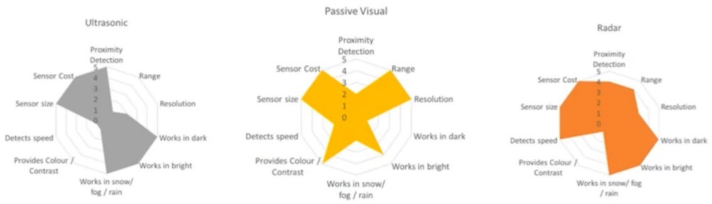


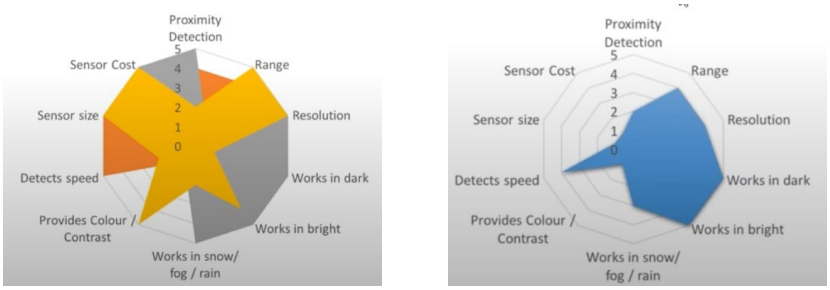
FIGURE 5: Performances séparées [2]

Deux Approches : Computer Vision Vs LIDAR[1]

— **Les LIDARs** : En prenant un ensemble particulier et restreint de routes, avec un mapping très spécifique et de manière à les comprendre de façon complète(sous différents climat etc)... Les lidars permettent de se localiser de manière efficace.

— **Les caméras** ont un aspect de vision de haute résolution, une texture riche d'information qu'on peut pas avoir avec du Lidar, ainsi qu'un énorme jeu de données pour l'entraînement des réseaux de neurones qui permettront de couvrir tous les cas marginaux et les cas limites.

Fusion de données issues de : Caméra+radar+ultrasonics(CRU). [2]
La figure suivante permet d'offrir une comparaison entre les performances des deux approches.



(a) performance CRU (b) performance Lidar

FIGURE 6: comparaison performances Fusion CRU / LIDAR[1][4]

On se retrouve alors avec :

— L'approche coûteuse par LIDAR, non basée sur le machine learning. Explicable, consistante et précise.

— Approche non-coûteuse basée sur la combinaison CRU+DL, qui nécessite beaucoup de données et qui n'est pas explicable et est relativement fiable.[2]

4.3 Génération de la 'Bird's-eye-view'(BEV)

4.3.1 Intérêt de la BEV

La conduite autonome nécessite une représentation d'état précise de l'environnement qui entoure le véhicule. L'environnement comprend des éléments statiques tels que le tracé de la route et la structure des voies, mais aussi des éléments dynamiques tels que les autres voitures, les piétons et les autres types d'usagers de la route. Les éléments statiques peuvent être saisis par une map HD contenant des informations sur le niveau des voies.

Dans les endroits où il n'y a pas de support cartographique et où le véhicule autonome n'est jamais allé, la cartographie en ligne serait indispensable. Parmi les différents types de capteurs couramment utilisés pour comprendre l'environnement,les caméras sont populaires en raison de leur faible coût et des techniques de vision par ordinateur. d'où l'intérêt de la BEV. il s'agit d'une approximation de la scéne qui entoure le véhicule en appliquant des transformation de perspectives des caméras, ce qui donne un vue d'en haut.[5]

la prévision et la planification sont généralement effectuées en se basant sur la BEV, car la plupart des informations dont un véhicule autonome aurait

besoin peuvent être représentées de manière concise avec la BEV.

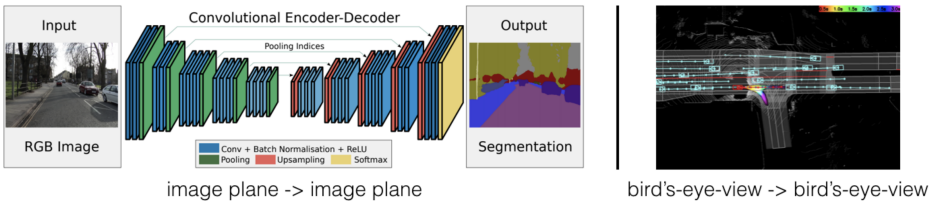


FIGURE 7: Stack de vision

4.3.2 Principe

La méthode utilisée pour transformer les images issues des caméras en BEV est communément appelée *Inverse Perspective Mapping*(IPM). l'hypothése faite étant que le monde est plat. tout ce qui est tridiemensionnel ou le moindre changement de hauteur risque de violer cette hypothése et génére des distortions du . Et c'est pour cette raison que l'IPM ne sert que d'input pour des algorithms de détection des voies ou l'espace libre calcul, pour lesquelles l'hypothèse du monde plat est souvent raisonnable.[cam2Bev]

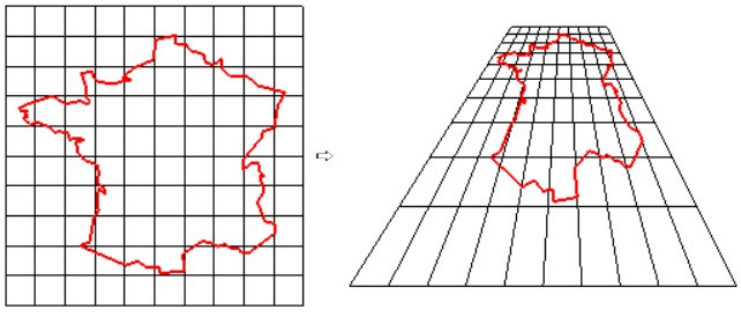


FIGURE 8: Homographie projective

Afin d'incorporer la transformation projective, il convient de déterminer la matrice d'homogrpahie $\mathbf{P}$ qui représente les paramétres intrinsèques (par exemple la Distance focale...), et extrinsèques (rotation $\mathbf{R}$ et translation $\mathbf{t}$ par rapport au plan d'en haut).

$$\boldsymbol{x}_i = \mathbf{P}\boldsymbol{x}_w$$

$\boldsymbol{x}_i$ étant les coordonnées image, et $\boldsymbol{x}_w$ les coordonnées au plan d'en haut.

$$\mathbf{P} = \mathbf{K}[\mathbf{R} \mid \boldsymbol{t}]$$

Afin de déterminer P pour les caméras du monde réel, des méthodes d'étalonnage peuvent être utilisées.[4] En supposant qu'il exite une transformation $\mathbf{M} \in R^{4\times3}$ à partir du plan de la route $\boldsymbol{x}_r \in R^3$ au plan d'en haut

$$\boldsymbol{x}_w = \mathbf{M}\boldsymbol{x}_r$$

En obtient une transformation des coordonnées de l'image au plan de la route.[6]

$$\boldsymbol{x}_r = (\mathbf{PM})^{-1}\boldsymbol{x}_i$$

le but du réseau de neurones peut se résumer en l'approximation de la fonction qui permet de fusionner ces images en corrigeant les effet de l'IPM.

4.3.3 Architecture Réseau de Neurones

Le réseau de neurones prend en entrée 4 images(avant, gauche, droite, arrière), applique l'IPM sur chaque image, les fusionnent et donne en sortie la BEV.

l'architecture de base consiste en un Auto-Encoder à convolution, lui même inspiré de l'architecture U-Net qui permet d'étendre le nombre d'input en **n** images selon le besoin.

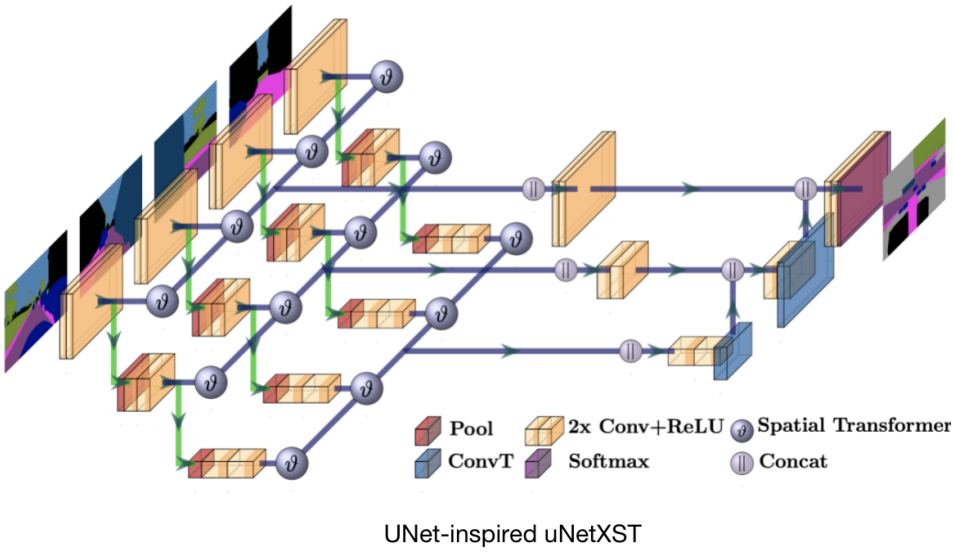


FIGURE 9: Architecture U-NetXST [6]

Cet architecture est fourni/implémenté par des chercheurs de l'université *RWTH Aachen* et est fourni sur Github[cam2bev].

4.3.4 Configuration Expérimental

Les données utilisées pour entraîner et évaluer la performance de cet architecture proviennent de l'environnement de simulation VTD(Virtual test drive), Qui permet de générer un série d'images labellisé. La BEV(target) est généré par un drone virtuel et dispose d'un camp de vision 70 m $\times$ 44 m.

Les images d'entrée et la BEV sont enregistrées à une résolution de 964×604. Toutes les caméras virtuelles produisent à la fois des images réalistes et sémantiquement segmentées. Pour la ségmentation sémanique, neuf classes sémantiques différentes sont prises en compte(route, trottoir, personne, voiture, camion, bus, vélo, obstacle, végétation).

ces classes ont été considéré et dispatchés en fonction du comportement que notre véhicule prendra selon la nature de ces classes. parceque le comportement du véhicule va changer s'il existe un camion dans notre environnement ou non.

En total, le dataset d'entraînement contient 33k échantillons. et 3.7k pour la validation. Où chaque échantillon correspond aux 4 images (avant, arrière, droite et gauche) et la BEV (target). et afin de réduire le temps d'entraînement, les images d'entrées et la *target* sont recadrés et redimensionnés à une résolution 512 x 256.

La métrique d'évaluation est principalement *l'indice de Jaccard*(IoU). Les scores IoU sont calculés pour chaque classe, et moyenné pour obtenir un score globale(MIoU).

$$(IoU)\ J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Ce choix tient en grande partie au fait que l'on souhaite mesurer dans quelle mesure le processus de fusion arrive à chevaucher les différents classes(sematiquement ségmentés), entre l'image constuite et la *vérité terrain*.

4.3.5 Traitement des Occultations

Une difficulté apparaît immédiatement lorsqu'on essaye de constuire la BEV : les autres véhicules et les obstacles statiques peuvent masquer certaines parties de l'environnement, ce qui rend pratiquement impossible la prédiction de ces zones dans une image BEV.

Afin de pallier à ce probléme, une autre classe est ajoutée pour spécifier les zones invisibles par les caméras. Pour chaque caméra du véhicule, des rayons virtuels sont diffusés depuis sa position vers les bords des autres classes qui invisibilisent ces régions. Les classes qui invisiblisent notre véhicule sont principalement : *(camion, bus)*.

ces zones invisibles sont également ajoutés aux données d'entraînement lors de l'étape du pré-traitement[6].

4.3.6 Résultats

Plusieurs variations du modéle U-NetXST ont été entrainées et comparés entre eux. Et on peut remarquer les améliorations/corrections apportées par rapport à une représentation homographique : 71,92 % contre 30 %.

MIoU Scores (%) on the Validation Set

Model	MIoU
uNetXST	**71.92**
DeepLab Xception	71.35
DeepLab MobileNetV2	66.60
DeepLab Xception*	60.13
DeepLab MobileNetV2*	55.09
uNetX*	45.95
Homography	30.17

Le signe (*) à la fin du nom du modéle est censé indiquer que la transformation projective n'a pas été appliqué.

TABLE II
Class IoU Scores (%) on the Validation Set

Model	Road	Sidewalk	Person	Car	Truck	Bus	Bike	Obstacle	Vegetation	Occluded
uNetXST	**98.10**	93.36	**13.56**	**80.90**	65.82	62.10	**32.43**	88.99	**97.27**	**86.62**
DL Xception	98.06	**94.02**	6.93	80.21	**65.94**	**65.98**	30.80	**89.05**	97.09	85.42

le tableau précédent nous permet d'analyser la performance par class. les 2 modéles (U-NetXST et Xception) détectent trés bien les régions qui sont grands/statique, à titre d'exemples*(la route, végétation)*. Des scores relativement bon pour *(Véhicules, camions, bus)*, qui sont participant au traffic/dynamique. Cependant, le modéle à un score trés faible pour les classes *(Personne, vélo)*, et qui s'explique par le fait que ces classes représentent des petits objets, et la moindre déviation par rapport à la vérité terrain résulte en un score trés faible.

L'image suivante présente les 4 images issues des caméras, l'homographie, reconstruction par le modéle DeepLab et le modéle U-NetXST

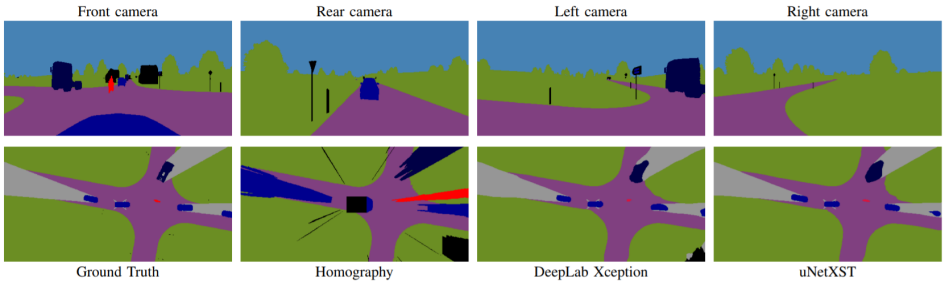


FIGURE 10: Exemple de résultat de la fusion des 4 images[6]

Comme il est visible, le U-NetXST arrive à approximer de maniére efficace la *vérité terrain*. Et les erreurs de l'homographie sont relativement corrigées, qui part du principe que le *monde est plat*.

4.3.7 Limitations & Ecart Réalité/Simulation

L'usage du dataset synthétique issus de la plateforme de simulation VTD a permis d'entraîner le modèle sans avoir l'obligation de passer par une grande étape d'annotation des données (dans ce cas, ça sera un drone qui permettra de capturer les images BEV). Cependant, il se peut que le modèle ne soit pas aussi performant sur des images réelles.

En revanche, l'écart entre réalité et simulation peut être rétréci en appliquant une segmentation sémantique aux images que l'on souhaite fusionner. Cette segmentation sémantique permet de supprimer des détails qui n'apporte pas beaucoup d'informations utiles au modelé de décision.

5 Système de Décision Autonome

Le rôle principal du Système de décision autonome est l'exploitation des informations acquises par le Système de perception d'environnement pour la prise de décisions du véhicule ; la planification de trajectoire tout en évitant les obstacles. Le Système de décision autonome décidera donc le trajet réel de la voiture en combinant deux trajets : un trajet global généré en se basant sur la localisation du véhicul et de sa destination finale, et un trajet local donné par l'unité de perception.[5]

Une grande partie de mes recherches se sont articulées autour du développement d'une stratégie d'évitement d'obstacles. L'approche que j'ai décidé de suivre se base sur la création d'un agent via des méthodes d'apprentissage par renforcement. L'avantage de cette approche c'est qu'elle est *model-free*. Elle part de l'hypothèse qu'il existe un modèle de transition d'état qui décrit la dynamique du système et qui reste fixe. Et par conséquent, il n'est pas nécessaire que la forme exacte de ce modèle soit connue a priori. (typiquement, un tel modèle est considéré comme inconnu).[8]

Dans cette partie je présenterai les principaux défis que j'ai rencontré en essayant de suivre cette approche.

5.1 Formulation du probléme

Notre objectif globale est de trouver une trajectoire locale et sans collision pour se rendre d'un point A vers un point B dans un environnement dynamique. Pour cela, nous devrions entraîner un agent RL(reinforcement learning), via une méthode de tâtonnements/erreurs, afin de trouver la meilleure *politique de conduite*. cette politique fait un mapping entre les meilleurs actions à prendre à partir d'un état donné.

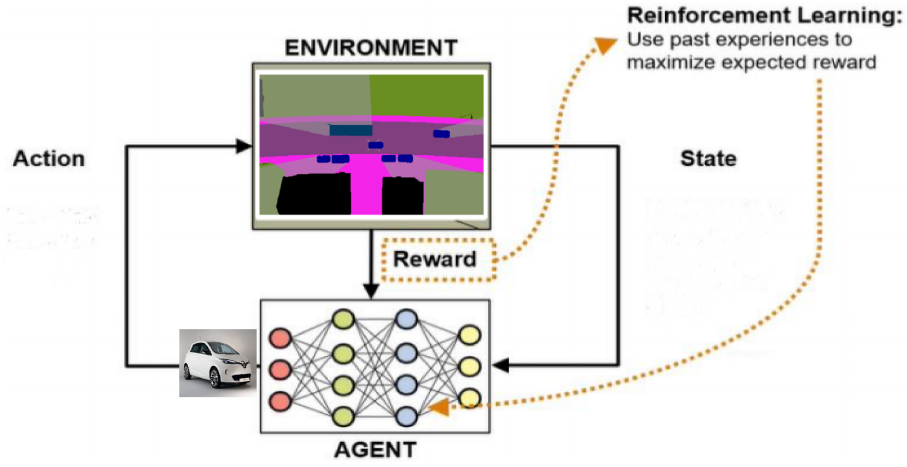


FIGURE 11: Schéma explicatif du probléme

Descriptif des entités :

— Environnement : BEV généré par fusion.
— Agent : Réseau de neurones, qui va jouer le rôle de controlleur. c'est lui qui prend les décisions à partir de l'état de notre véhicule.
— State : Les *inputs* du réseau de neurones seront : (BEV, vitesse, position, dérivés, mesures des capteurs)
— Actions : Les *outputs* du réseau de neurones seront : (Angle de braquage, accélération, frein)

5.2 Process

Le processus séquentiel du Reinforcement Learning peut être décrit comme le processus de décision Markovien(MDP). En général, un MDP se compose d'un ensemble d'états $s \in S$, d'un ensemble d'actions $a \in A$, une fonction de récompense R et un modéle de Transition P. Le véhicule autonome(agent) perçoit l'environnement courant avec l'état s_t et performe une action a_t, puis transite vers l'état s_{t+1} en fonction de la fonction de transition $P\left(s_{t+1} \mid s_t, a_t\right)$. Et ceci continue comme un systéme de feedback, et à chaque timestep, l'environnement évalue cet action et donne une récompense par rapport à l'action que l'agent a pris.

On peut par la suite obtenir une politique $\pi : \mathcal{S} \rightarrow \mathcal{A}$, qui cartographie chaque états $_t$ à l'action a_t. L'objectif en RL est de maximiser la récompense accumulée Rt, qui est calculée par : $R_t = \sum_{i=t}^{T} \gamma^{i-t} r\left(s_i, a_i\right)$

Il existe beaucoup d'algorithmes pour qui traitent des *state-action space* à valeurs continue, Notamment l'algo DDQN(Double Deep Q-Network)[8]. La récompense future maximale pour une paire (state a, action a)est représentée par la fonction action-valeur optimale $Q^*(s, a)$. la sélection d'une action a est représentée par la fonction action-valeur optimale Q(s, a), qui est définie comme suit :

$$Q^*(s, a) = \max_{\pi} E\left[R_t \mid s_t = s, a_t = a, \pi\right]$$

La fonction action-valeur optimale obéit à une identité très importante connue sous le nom d'équation de Bellman, et est donné par sa formule de récurrence[8] :

$$Q^*(s, a) = E_{s_{t+1}}\left[r_t + \gamma \max_{a_{t+1}} Q^*\left(s_{t+1}, a_{t+1}\right) \mid s_t = s, a_t = a\right]$$

5.3 Défis

De nombreux défis à relever, pour en citer quelques-uns :

— **Design de la fonction de récompense :**
Le plus gros challenge c'est de designer une fonction de récompense qui indique la qualité de notre contrôle. On est en face d'un probléme avec des récompense qui sont éparses(*sparse*) : nous savons seulement que notre véhicule a fait du bon travail lorsqu'il s'est rendu du point A à B sans collision sous certaines contraintes. nous n'obtenons pas de récompenses concrètes à des intervalles intermédiaires, ce qui ne va pas vraiment aider à apprendre plus rapidement.

— **Problème d'affectation de crédit**(*Credit Assignment Problem CAP*) :
Dans ce contexte, le mot crédit est synonyme de valeur. En RL, une action qui conduit à une récompense cumulative finale plus élevée devrait avoir plus de valeur (donc plus de "crédit" devrait lui être attribué) qu'une action qui conduit à une récompense finale plus faible.

— **Data-efficiency** :
malgré de nombreuses avancées récentes, l'apprentissage profond est encore très gourmand en données, de sorte que la plupart limitent leur formation à des simulations où ils peuvent recueillir gratuitement de nombreuses données, puis essayer de les transférer dans le monde réel.

— **Sécurité** :
Les algorithmes de RL apprennent généralement par essais et erreurs. C'est très bien dans une simulation où ils sont libres de se planter autant qu'ils le souhaitent pendant l'apprentissage, mais dans le monde réel, cela pose de nombreux problèmes logistiques pour essayer de mettre en place un système sûr et rentable.

6 Conclusion

Le long de ce rapport on a vu comment la représentation d'état et le mapping de l'environnement sont des éléments critiques pour le pipeline d'autonomisation du véhicule. Ainsi que dans quelle mesure cela peut être beaucoup plus approprié de générer une BEV que traiter les images issues des caméras de manière indépendante.

Cela étant dit, il est avantageux pour la perception à travers les modalités d'utiliser la représentation BEV. Tout d'abord, elle est interprétable et facilite le débogage des modes de défaillance inhérents à chaque modalité de détection. Elle est également facilement extensible à d'autres nouvelles modalités et simplifie la tâche de la fusion tardive. En outre, comme mentionné ci-dessus, les résultats de la perception dans cette représentation peuvent être facilement consommés par les systèmes de prédiction, Décision et de planification.

Cependant, quelques corrections devraient être mis en place, notamment développer une méthode qui permet de mieux détecter les *(personne, vélo)* en fusionnant d'autres données de capteurs ultrasonique/radar à titre d'exemple.

Pour la partie Planification de trajectoire, l'approche discutée permettra de générer une trajectoire locale, mais plusieurs hypothèses sont supposées être vraies, ce qui rend l'extrapolation vers le monde réel compliqué. En revanche, plusieurs défis sont à relever, notamment le design de la fonction de récompense qui est dans notre *use-case* très éparse. Pour pallier à ce problème un expert métier peut mettre en place une *proxy-reward* pour obtenir des récompenses intermédiaires et permettre à l'agent d'apprendre plus rapidement .

7 Retour d'expérience

A travers ce projet d'intégration, j'ai pu consolider/explorer des thémes scientifiques traitant des technologies de pointe. Ainsi que travailler sur un des plus grands sujets au niveau mondial.

En effet, les missions auxquelles j'étais confronté m'ont permis de toucher à plusieurs stades du machine learning : de la récupération à l'interprétation des résultats en passant par le nettoyage, la structuration et le traitement des données mais aussi de ce qui se passera après déploiement du modèle.

Par ailleurs, dans le cadre de ce projet, j'ai été confronté à la lecture, l'étude et la compréhension de publications scientiques et avoir une veille technologique continue, primoridiale dans un domaine caractérisé par sa dynamique vive et permanente. De plus, j'ai dû m'autoformer sur des sujets tels que le *Deep Reinforcement Learning* et *Computer-vision* . Comme je le laisse transparaître tout au long de la rédaction de ce rapport, je suis adepte des solutions à long terme et généralisable ainsi que de la rigueur et la justification théorique de concepts utilisés. C'est pour respecter cela que j'ai amélioré mon effort d'abstraction de concepts métier et leur transcription en équations mathématiques afin de se donner une idée claire sur les points auxquels il faut faire attention lors du développement d'algorithme ou de modèles.

8 Bibliographie

[1] Automated Vehicles for Safety : article par : **National Highway Traffic Safety Administration**

[2]Lex Fridman's MIT lectures

[3] A Survey on Theories and Applications for Self-Driving Cars Based on Deep Learning Methods : **Jianjun Ni**, **Yinan Chen**, **Yan Chen**, **Jinxiu Zhu**, **Deena Al**, and **Weidong Cao**

[4] Pseudo-LiDAR from Visual Depth Estimation : Bridging the Gap in 3D Object Detection for Autonomous Driving : **Yan Wang, Wei-Lun Chao**

[5] MultiPath : Multiple Probabilistic Anchor Trajectory Hypotheses for Behavior Prediction : **Yuning Chai, Benjamin Sapp**

[6] Cam2BEV : A Sim2Real Deep Learning Approach for the Transformation of Images from Multiple Vehicle-Mounted Cameras to a Semantically Segmented Image in Bird's Eye View, **Lennart Reiher, Bastian Lampe, Lutz Eckstein**, ITSC 2020

[7] Waymo, ChauffeurNet : Learning to Drive by Imitating the Best and Synthesizing the Worst ; **Mayank Bansal**

[8] Deep Reinforcement-Learning-Based Driving Policy For Autonomous Vehicles, **Konstantinos Makantasis, Maria Kontorinaki, Ioannis Nikolos**, Department of Statistics and Operations Research, Faculty of Science, University of Malta, Msida, Malta. 2020