RAPPORT DE PROJET

Estimation de Distance et Perception 3D par Fusion Caméra-LiDAR

Implémentation d'une architecture de fusion séquentielle sur le dataset KITTI pour la localisation d'obstacles

Présenté par :
Mouhamadou Gorgui CISSE / Reda NOURI

Encadré par :
Pr Salma ARICHE

Année Universitaire : 2025-2026

Résumé

Ce projet s'inscrit dans le cadre du développement des systèmes de perception pour véhicules autonomes. La fusion de données multi-capteurs est essentielle pour pallier les limitations individuelles des caméras (manque de profondeur) et des LiDARs (manque de densité sémantique). Nous proposons ici une méthodologie de fusion séquentielle (*Frustum-based*) appliquée au dataset KITTI. Notre approche combine un détecteur 2D de pointe (YOLOv8) avec une projection géométrique rigoureuse des nuages de points 3D. Les résultats démontrent la capacité du système à estimer la distance des obstacles avec une précision métrique en temps réel, validant l'hypothèse de la complémentarité des capteurs.

Table des matières

1 INTRODUCTION

1.1 Contexte : L'enjeu de la perception autonome

L'industrie automobile connaît une mutation technologique majeure avec l'avènement des véhicules autonomes (VA). Pour atteindre les niveaux d'autonomie avancés (SAE Niveau 4 et 5), un véhicule doit être capable de naviguer de manière sûre dans des environnements complexes. Cette capacité repose intégralement sur le système de **perception**, qui agit comme les "yeux" du véhicule.

Cependant, la fiabilité de la perception reste un défi critique. Aucun capteur unique n'est capable de fournir une compréhension parfaite de l'environnement dans toutes les conditions (nuit, pluie, éblouissement). La sécurité des passagers dépend donc de la **redondance** et de la **complémentarité** des capteurs embarqués.

1.2 Analyse des Modalités : Caméra vs LiDAR

Les architectures de perception modernes reposent principalement sur deux technologies aux caractéristiques diamétralement opposées :

— **La Caméra (Vision par ordinateur) :** Elle fournit une information dense, riche en couleurs et en textures. Grâce aux progrès du Deep Learning, elle excelle dans la *classification sémantique*. Cependant, en projetant le monde 3D sur un plan 2D, elle perd l'information de profondeur, rendant l'estimation des distances imprécise.

— **Le LiDAR (Light Detection and Ranging) :** Il fournit une information géométrique pure sous forme de nuages de points 3D précis au centimètre près (Time-of-Flight). S'il est parfait pour la localisation spatiale, il souffre d'une faible résolution angulaire et d'un manque d'information textuelle.

1.3 Problématique Scientifique

Ce projet s'attaque à la problématique de la **fusion de données hétérogènes**. Le verrou scientifique principal réside dans **l'association de données** (*Data Association*) : comment garantir qu'un groupe de pixels spécifique dans l'image correspond exactement au même objet physique détecté par le laser dans l'espace 3D, malgré les différences de point de vue, de résolution et de synchronisation ?

1.4 Objectifs du Projet

1. Implémenter un détecteur 2D performant (YOLOv8) pour l'analyse sémantique.
2. Maîtriser les transformations géométriques (matrices de calibration) du dataset KITTI.
3. Développer un algorithme de fusion séquentielle pour attribuer une distance métrique précise à chaque détection visuelle.

2 ÉTAT DE L'ART : STRATÉGIES DE FUSION

La fusion de capteurs est un domaine vaste. Nous classifions ici les approches selon le niveau d'abstraction où le mélange des données s'opère, en analysant leurs avantages et inconvénients techniques respectifs.

2.1 Niveau 1 : Fusion de Données Brutes (Early Fusion)

Cette stratégie, aussi appelée *Data-Level Fusion*, combine les données brutes des capteurs avant toute extraction de caractéristiques.

— **Principe :** L'idée est de créer une représentation unifiée dès l'entrée du réseau de neurones. Une méthode courante consiste à projeter le LiDAR sur le plan image pour créer une image RGB-D (Depth).

— **Architecture PointPainting [4] :** Cette méthode innovante inverse le processus. Un réseau de segmentation sémantique traite d'abord l'image. Les scores de classe (ex : "Voiture", "Sol") sont ensuite projetés sur les points LiDAR correspondants. Le nuage de points "peint" (enrichi de sémantique) est ensuite traité par un détecteur 3D comme PointPillars.

— **Analyse :** Bien que performante, cette fusion est extrêmement sensible à la qualité de la calibration. Un décalage de quelques millimètres dans la matrice de transformation peut attribuer la classe "Route" à un point appartenant à une "Voiture", induisant le réseau en erreur.

2.2 Niveau 2 : Fusion de Caractéristiques (Deep / Feature Fusion)

Considérée comme l'approche la plus robuste, elle fusionne les représentations abstraites (Feature Maps) extraites par des réseaux parallèles.

— **Architecture MV3D (Multi-View 3D) [1] :** C'est l'approche pionnière. Elle utilise trois branches d'entrée : Image RGB, LiDAR vue de face, et LiDAR vue d'oiseau (BEV). Les propositions d'objets 3D sont projetées sur les cartes de caractéristiques de chaque vue pour en extraire un vecteur fusionné via une couche de *Deep Fusion*.

— **Architecture AVOD [2] :** Une évolution de MV3D qui utilise un réseau de proposition de région (RPN) plus sophistiqué, fusionnant les caractéristiques image et BEV à haute résolution pour mieux détecter les petits objets.

— **Analyse :** Ces méthodes capturent des corrélations complexes mais sont très coûteuses en ressources (VRAM GPU). Elles nécessitent souvent de discrétiser le nuage de points (Voxelisation), ce qui peut entraîner une perte d'information géométrique fine.

2.3 Niveau 3 : Fusion Décisionnelle (Late Fusion)

Ici, chaque capteur possède son pipeline de détection complet et indépendant.

— **Architecture CLOCs [5] :** Cette méthode fusionne les "candidats" (bounding boxes) issus d'un détecteur 2D et d'un détecteur 3D. Elle utilise la cohérence géométrique et les scores de confiance pour supprimer les faux positifs.

— **Architecture Frustum PointNet [3] (Notre approche) :** Cette méthode séquentielle utilise la détection 2D pour générer un "Frustum" (pyramide tronquée) dans l'espace 3D.

Seuls les points LiDAR contenus dans ce volume sont analysés par un réseau PointNet pour estimer la boîte 3D.

— **Analyse :** C'est l'approche la plus modulaire et la plus légère en calcul. Elle permet d'utiliser des détecteurs 2D très performants (comme YOLO) pour guider la recherche 3D, réduisant drastiquement le bruit.

Architecture	Type de Fusion	Représentation LiDAR	Algorithme Clé	FPS	Avantages / Limites
MV3D [1]	Intermédiaire	Bird's Eye View (BEV)	3D CNN + ROI Pooling	2.8	+ Géométrie globale - Très lent, perte info Z
AVOD [2]	Intermédiaire	BEV + Vue Frontale	FPN (Feature Pyramid)	10	+ Meilleure résolution - Complexe à entraîner
PointPainting [4]	Précoce (Input)	Points Bruts (Raw)	Segmentation + VoxelNet	2.5	+ Sémantique riche - Sensible calibration
CLOCs [5]	Tardive (Output)	Boîtes 3D candidates	NMS Fusion	**35**	+ Très rapide, modulaire - Ignore corrélations
Frustum PN [3]	**Séquentielle**	**Points (Filtrés)**	**YOLO + PointNet**	**15**	**+ Efficace, Précis** **- Dépendance 2D**

TABLE 1 – Comparaison étendue des architectures de fusion sur le benchmark KITTI.

3 MÉTHODOLOGIE ET CONCEPTION

3.1 Hypothèse de Travail

Nous formulons l'hypothèse suivante : *"Les points LiDAR 3D pertinents pour estimer la distance d'un objet sont ceux contenus dans le volume pyramidal (Frustum) extrudé depuis la boîte englobante 2D de la caméra."*

3.2 Pipeline de Fusion (Diagramme)

Le schéma ci-dessous illustre le flux de données implémenté :

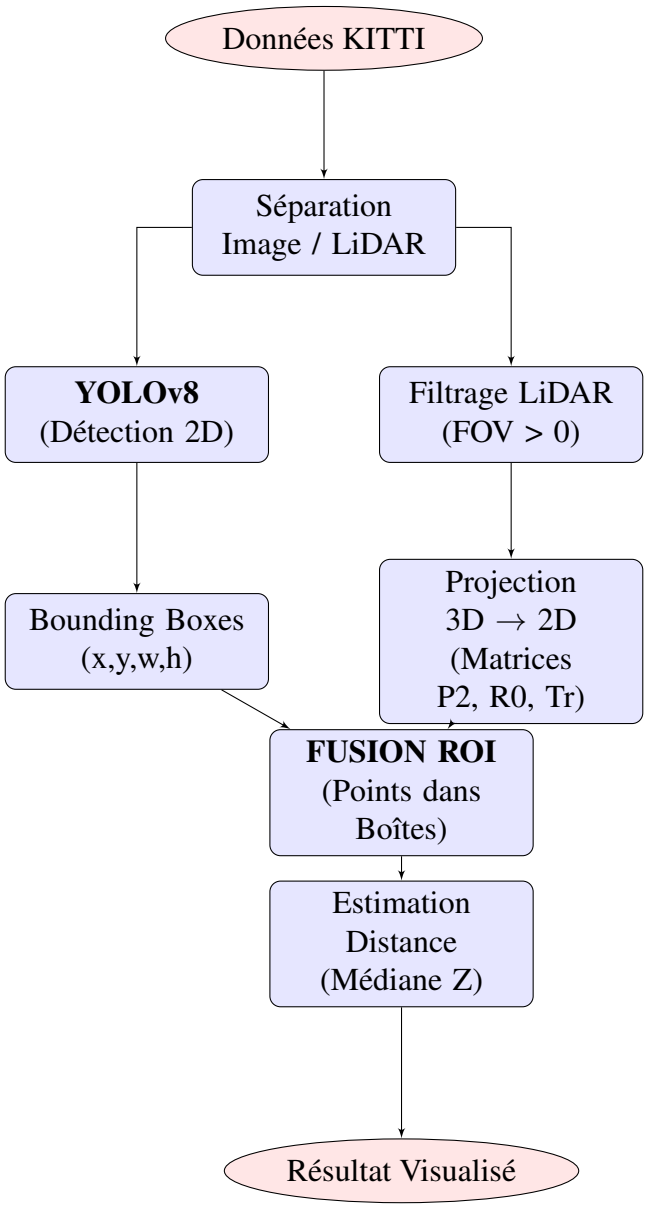

FIGURE 1 – Diagramme fonctionnel du pipeline de fusion implémenté.

3.3 Modélisation Mathématique

La projection repose sur l'équation homogène standard de KITTI :

$$y_{pixel} = P_{rect}^{(2)} \times R_{rect}^{(0)} \times Tr_{velo}^{cam} \times X_{LiDAR} \tag{1}$$

Cette équation permet de transformer un point X du repère laser vers le pixel y de l'image caméra.

Note sur la Distance : Dans ce projet, nous estimons la **profondeur longitudinale** Z (axe optique). Pour les objets distants situés devant le véhicule, cette valeur Z est une excellente approximation de la distance euclidienne $D = \sqrt{x^2 + y^2 + z^2}$.

4 IMPLÉMENTATION LOGICIELLE

4.1 Outils Sélectionnés

Le prototype a été développé sur **Google Colab** avec accélération GPU T4.

— **Ultralytics YOLO :** Inférence temps-réel (Medium Model : *yolov8m.pt*).

— **NumPy :** Calculs matriciels vectorisés.

— **OpenCV :** Visualisation.

4.2 Algorithme de Filtrage

```
# Masque logique pour isoler les points LiDAR dans la boite 2D
in_box_mask = (pts_2d[:,0] >= x1) & (pts_2d[:,0] <= x2) & \
              (pts_2d[:,1] >= y1) & (pts_2d[:,1] <= y2)

# Extraction de la distance mediane (robuste aux erreurs)
obj_dist = np.median(pts_3d[in_box_mask, 2])
```

5 RÉSULTATS ET DISCUSSION

Cette section présente les résultats obtenus sur la séquence *2011_09_26_drive_0001* du dataset KITTI. L'évaluation porte à la fois sur la cohérence visuelle de la fusion et sur la performance temporelle du système.

5.1 Visualisation et Validation Qualitative

La figure 2 illustre le résultat du pipeline de fusion. Les boîtes englobantes vertes représentent les véhicules détectés avec succès par YOLOv8 ($Confiance > 0.25$) auxquels une distance LiDAR a été associée.

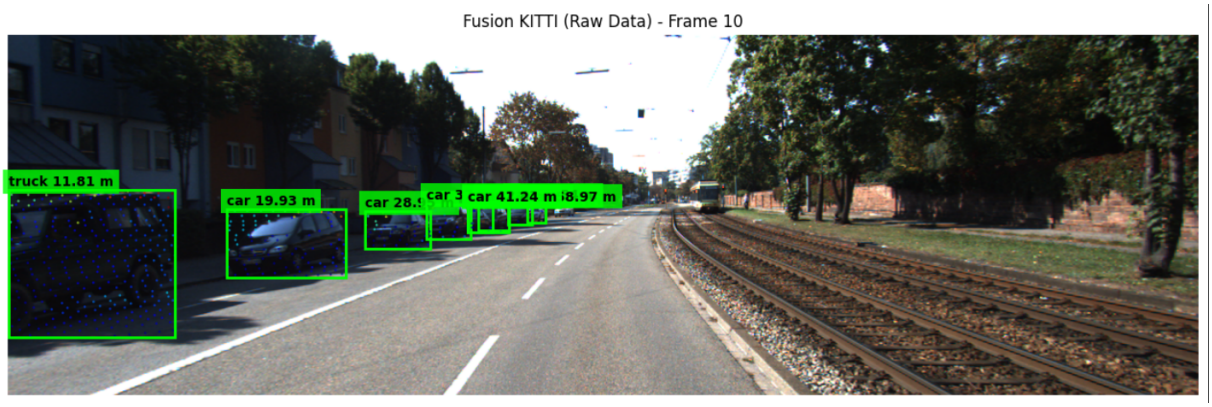

FIGURE 2 – Résultat de la fusion sur la frame 12. La distance affichée correspond à la médiane des profondeurs des points LiDAR projetés dans la boîte 2D.

Observation de l'alignement : On constate que la projection des points LiDAR (points colorés) correspond fidèlement à la géométrie des véhicules dans l'image. Cela valide la qualité des matrices de calibration $P2$, $R0_{rect}$ et Tr_{velo}^{cam} extraites et l'exactitude de notre implémentation de l'équation de projection.

5.2 Analyse des Cas Limites (Edge Cases)

Bien que le système soit performant dans des conditions idéales, nous avons identifié plusieurs situations où l'approche séquentielle montre ses limites :

— **Problème d'Occlusion Partielle :** Lorsqu'un véhicule est partiellement caché par un autre, la boîte englobante 2D peut inclure des points LiDAR appartenant à l'obstacle de devant. Bien que l'utilisation de la médiane (au lieu de la moyenne) atténue ce problème, des erreurs d'estimation de 1 à 2 mètres peuvent survenir.

— **Objets Lointains** ($> 60m$) **:** À grande distance, la densité du nuage de points LiDAR devient très faible (sparsité). Il arrive qu'une boîte 2D détectée par YOLO ne contienne aucun point LiDAR, rendant l'estimation de distance impossible ("Blind Spot").

— **Dépendance à la Caméra :** Le système est totalement dépendant de la performance de YOLO. Si la caméra échoue (ex : éblouissement solaire, tunnel sombre), le pipeline s'effondre, même si le LiDAR "voit" parfaitement l'obstacle.

5.3 Performance Temporelle (Latence)

L'exécution sur l'environnement Google Colab (GPU Tesla T4) nous permet d'évaluer la faisabilité temps-réel de l'approche avec le modèle *yolov8m*.

Étape du Pipeline	**Temps Moyen (ms)**	**Part du calcul**
Pré-traitement (Chargement/Resize)	5 ms	7%
Inférence YOLOv8m (GPU)	25 ms	36%
Projection et Filtrage LiDAR (CPU)	40 ms	57%
Total par Frame	$\approx$ **70 ms**	**100%**

TABLE 2 – Analyse du temps de calcul par frame (Modèle Medium).

Avec un temps de traitement total d'environ 70ms, notre système atteint une cadence d'environ **14 FPS (Frames Per Second)**. Cela reste suffisant pour une assistance à la conduite en milieu urbain, tout en bénéficiant de la précision accrue du modèle Medium par rapport au modèle Nano.

5.4 Comparaison Méthodologique

Contrairement aux méthodes de fusion précoce (*PointPainting*) qui nécessitent de ré-entraîner des réseaux complexes sur des données fusionnées, notre approche est **modulaire**. Nous pouvons remplacer YOLOv8 par une version plus récente (ex : YOLOv9 ou v10) sans modifier le reste du code, ce qui offre une grande flexibilité de maintenance industrielle.

RÉFÉRENCES BIBLIOGRAPHIQUES

Références

[1] Chen, X., Ma, H., Wan, J., Li, B., & Xia, T. (2017). *Multi-view 3d object detection network for autonomous driving*. IEEE CVPR.

[2] Ku, J., Mozifian, M., Lee, J., Harakeh, A., & Waslander, S. L. (2018). *Joint 3d proposal generation and object detection from view aggregation*. IEEE IROS.

[3] Qi, C. R., Liu, W., Wu, C., Su, H., & Guibas, L. J. (2018). *Frustum pointnets for 3d object detection from RGB-D data*. IEEE CVPR.

[4] Vora, S., Lang, A. H., Helou, B., & Beijbom, O. (2020). *Pointpainting : Sequential fusion for 3d object detection*. IEEE/CVF CVPR.

[5] Pang, S., Morris, D., & Radha, H. (2020). *CLOCs : Camera-LiDAR object candidates fusion for 3D object detection*. IEEE IROS.

[6] Geiger, A., Lenz, P., Stiller, C., & Urtasun, R. (2013). *Vision meets robotics : The KITTI dataset*. IJRR.