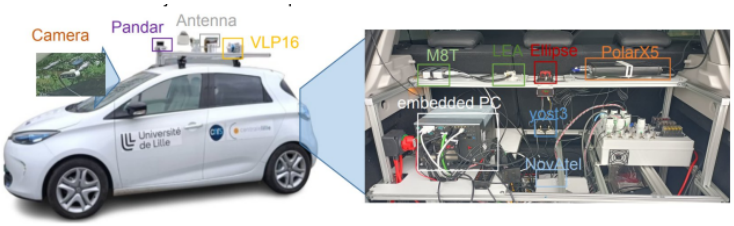
Université de Lille

Master SMaRT — ToSyMA

Évaluation de modules de localisation pour véhicule autonome

ROS 2 & CARLA : `robot_localization` (EKF/UKF)



Auteurs :

DAOU Sekou

Touwende OUEDRAOGO

Encadrants :

Zaynab EL MAWAS

Maan EL BADAOUI EL NAJJAR

Janvier 2026

Résumé

Ce rapport évalue la localisation d'un véhicule autonome en simulation (ROS 2/CARLA) au moyen des filtres EKF et UKF du package `robot_localization`. Les configurations de fusion étudiées s'appuient sur l'odométrie, l'IMU et le GNSS, et sont comparées à la vérité terrain fournie par CARLA via des indicateurs de position et d'orientation (erreur 2D, RMSE, erreur yaw) et des visualisations standardisées. Les résultats montrent que, sur une séquence simulée courte et peu bruitée, les métriques de position différencient faiblement les configurations, tandis que l'orientation (yaw) met davantage en évidence l'impact du modèle de fusion et du paramétrage des incertitudes. Une expérimentation complémentaire avec odométrie dégradée confirme que les performances deviennent fortement sensibles aux réglages dès que les mesures s'éloignent du cas idéal.

Mots-clés : localisation, fusion multi-capteurs, EKF, UKF, ROS 2, CARLA.

Abstract

This report evaluates vehicle localization in simulation (ROS 2/CARLA) using the EKF and UKF filters from the `robot_localization` package. Sensor fusion configurations based on odometry, IMU, and GNSS are assessed against CARLA ground truth using position and orientation metrics (2D error, RMSE, yaw error) and standardized plots. Results indicate that, on a short low-noise sequence, position metrics provide limited discrimination between configurations, while heading (yaw) better reflects the influence of the fusion model and uncertainty tuning. A complementary test with degraded odometry confirms that performance becomes highly sensitive to filter parameters when measurements depart from ideal conditions.

Keywords : localization, sensor fusion, EKF, UKF, ROS 2, CARLA.

Remerciements

Nous remercions nos encadrants, Zaynab EL MAWAS et Maan EL BADAOUI EL NAJJAR, pour leurs conseils et leur suivi tout au long du projet.

Table des matières

Table des figures

Liste des tableaux

Table des matières

Table des figures

Liste des tableaux

1 Introduction

1.1 Contexte et motivation

La localisation constitue une brique centrale de la mobilité autonome : elle conditionne le suivi de trajectoire, la planification et, plus largement, la prise de décision. Sur un véhicule réel, aucune source n'est parfaitement fiable ni suffisamment robuste seule : l'odométrie dérive avec le temps (erreurs d'intégration, glissements), l'IMU est bruitée et sujette aux biais, tandis que le GNSS dépend fortement de l'environnement (masques, multipaths, pertes de signal). La pratique industrielle et académique repose donc sur la **fusion multi-capteurs**, souvent implémentée via des filtres bayésiens de la famille de Kalman.

Ce projet s'inscrit dans les travaux de la plateforme PRETIL et de l'équipe ToSyMA autour d'un **jumeau numérique** d'un véhicule Renault Zoé robotisé et de son équivalent simulé sous **CARLA/ROS 2**. L'objectif global est de **mettre en place, tester et évaluer** des modules de localisation multi-capteurs (issus notamment d'Autoware et de dépôts open-source) afin d'étudier leur précision, leur latence et leurs limites, dans une logique de transférabilité entre simulation et véhicule réel.

1.2 Capteurs disponibles et périmètre de fusion retenu

Le rosbag exploité provient de la chaîne CARLA/ROS 2 et contient plusieurs modalités : **odométrie**, **IMU** et **GNSS**, ainsi que des capteurs de perception tels que **caméras** (RGB, profondeur, segmentation), **LiDAR** (standard et sémantique) et **radar**. Ces capteurs de perception constituent une base intéressante pour des approches plus riches (VO/VIO, SLAM LiDAR/vision), mais ils ne sont pas directement fusionnables par `robot_localization` sans une étape intermédiaire produisant une mesure de pose/odométrie assortie de covariances.

Dans ce rapport, le périmètre de fusion se concentre donc sur **odométrie + IMU + GNSS**, car ce sont les sources *directement exploitables* par `robot_localization`. Ce choix permet de prioriser une analyse claire (et reproductible) du comportement des filtres **EKF/UKF**, et de l'impact des réglages d'incertitudes (Q, R, P_0) sur les erreurs observées.

1.3 Objectifs et contributions

Le travail présenté vise à :

— mettre en place une chaîne expérimentale reproductible basée sur un rosbag CARLA rejoué en temps simulé

— configurer et exécuter des filtres de `robot_localization` (EKF puis UKF) sur différentes combinaisons de capteurs

— définir une méthode d'évaluation par comparaison à la vérité terrain disponible dans CARLA et produire des métriques (erreurs e_x, e_y, e_{2D}, RMSE, erreur yaw) et des figures associées

— analyser l'apport relatif des capteurs et relier les résultats au fonctionnement interne des filtres (prédiction/correction, rôle de Q, R et P_0).

1.4 Organisation du rapport

Le rapport présente d'abord l'environnement expérimental et les données (CARLA/ROS 2, topics disponibles), puis rappelle le principe des filtres EKF/UKF et la structure prédiction/correction. La méthodologie d'évaluation (appariement temporel, métriques, figures) est ensuite détaillée. Enfin, les résultats sont présentés *run par run* et synthétisés afin de dégager des tendances et des limites dans le cadre étudié.

2 Environnement et données

2.1 Données utilisées

Les expérimentations reposent sur un rosbag issu de CARLA, rejoué en temps simulé (`/clock`). Les topics utilisés sont :

— vérité terrain : `/carla/hero/odometry` ;

— IMU : `/carla/hero/imu` ;

— GNSS : `/carla/hero/gnss` ;

— sortie filtre : `/odometry/filtered` ;

— repères et temps : `/tf /clock`.

2.2 Remarque sur la simulation

Dans ce scénario simulé, l'odométrie peut être très cohérente avec la vérité terrain. Les performances observées dans ce cadre ne doivent pas être extrapolées directement au cas réel, où l'odométrie dérive généralement. Cela motive l'analyse multi-capteurs et la discussion sur le réglage des incertitudes.

3 Filtres de localisation : EKF, UKF et capteurs

3.1 Principe général (prédiction / correction)

Les filtres EKF et UKF alternent :

— **Prédiction** : propagation de l'état via un modèle de mouvement ; l'incertitude est décrite par la covariance de processus Q.

— **Correction** : mise à jour à la réception d'une mesure ; la correction est pondérée par la covariance de mesure R.

La covariance initiale P_0 influence principalement le transitoire au démarrage.

3.2 Vecteur d'état (mouvement plan)

Les configurations sont utilisées en mode `two_d_mode`. Le filtre estime principalement la position (x, y) et l'orientation yaw ψ, ainsi que des composantes de vitesse nécessaires à la propagation. L'interprétation des résultats dépend des composantes effectivement corrigées par les capteurs.

3.3 Capteurs et variables corrigées

Les fichiers YAML définissent, via les vecteurs `*_config`, les composantes corrigées par chaque capteur :

— **Odométrie** : corrige principalement (x, y) et des variables dynamiques (yaw, vitesses) selon la configuration.

— **IMU** : agit surtout sur l'orientation (yaw) et la dynamique angulaire ; son influence dépend fortement de la pondération (covariances).

— **GNSS** : après conversion (via `navsat_transform_node`), contraint essentiellement la position globale (x, y).

3.4 Schéma prédiction/correction : corrections par capteur et lien avec `/odometry/filtered`

Afin d'interpréter les courbes d'erreurs, il est essentiel de préciser *quelles composantes de l'état* sont corrigées par chaque capteur dans `robot_localization`. Dans la suite, l'état est représenté (forme simplifiée en 2D) par :

$$\mathbf{x} = \begin{bmatrix} x & y & \psi & v_x & v_y & \dot{\psi} \end{bmatrix}^\top, \tag{1}$$

où (x, y) est la position dans le plan, ψ le yaw, (v_x, v_y) les vitesses linéaires et $\dot{\psi}$ la vitesse angulaire.

Prédiction (modèle interne). Entre deux mesures, le filtre propage l'état selon un modèle générique de type *vitesse quasi constante*. L'incertitude augmente en fonction de la covariance de processus Q (erreurs de modèle, dynamique non modélisée). Ce rapport étudie donc principalement l'effet de la *fusion probabiliste* et du paramétrage (Q, R, P_0), et non l'évaluation d'un modèle cinématique véhicule détaillé (ex. Ackermann).

Corrections successives (selon l'arrivée des mesures). À l'arrivée d'une mesure, le filtre effectue une correction uniquement sur les composantes observées par ce capteur. Dans `robot_localization`, ce choix est réalisé dans les fichiers YAML via les vecteurs `odom0_config`, `imu0_config` (et `odom1_config` pour une source GPS convertie). Concrètement, cela explique pourquoi certains capteurs influencent fortement le yaw alors qu'ils modifient peu la position.

Capteur	x	y	ψ	v_x	v_y	$\dot{\psi}$
Odométrie (`/carla/hero/odometry`)	✓	✓	✓*	✓*	(opt.)	✓*
IMU (`/carla/hero/imu`)			✓/ (opt.)			✓
GNSS (`/carla/hero/gnss` → `/odometry/gps`)	✓	✓				

Table 1. Rôle des capteurs et composantes de l'état principalement corrigées (principe). Les composantes exactes dépendent des vecteurs `*_config` des fichiers YAML.

* dépend du paramétrage `odom0_config`.

Lecture pas à pas de la sortie `/odometry/filtered`. Sur une fenêtre temporelle typique, la sortie du filtre s'obtient par l'enchaînement suivant :

1. **Prédire** l'état (propagation) : $\mathbf{x}_{k|k-1}$ et sa covariance $P_{k|k-1}$.
2. **Corriger** avec l'odométrie lorsque disponible : mise à jour principalement de (x, y) et, selon configuration, de ψ et de composantes dynamiques.
3. **Corriger** avec l'IMU lorsque disponible : mise à jour des composantes angulaires (typiquement $\dot{\psi}$ et éventuellement ψ).
4. **Corriger** avec le GNSS (après conversion) : recadrage de la position globale (x, y).

Ainsi, si l'IMU est fortement pondérée (covariances trop faibles), la correction sur ψ peut dominer et provoquer une augmentation de l'erreur yaw, tandis que le GNSS recadre surtout la position sans agir directement sur l'orientation.

3.5 Synthèse : sources et corrections

Source	Topic	Rôle principal dans la correction
Odométrie	`/carla/hero/odometry`	position (x, y) et dynamique (selon config)
IMU	`/carla/hero/imu`	orientation (yaw) et vitesse angulaire
GNSS	`/carla/hero/gnss`	position globale (x, y) (via navsat)

Table 2. Synthèse des sources et variables corrigées (principe).

3.6 EKF vs UKF

Les deux filtres sont disponibles dans `robot_localization` :

— l'**EKF** gère les non-linéarités via une approximation locale (linéarisation)

— l'**UKF** propage des points représentatifs (sigma points) à travers les modèles non linéaires.

L'objectif de la comparaison est d'évaluer si l'UKF apporte un gain de stabilité, notamment sur l'orientation lorsque l'IMU est fusionnée.

4 Méthodologie d'évaluation

4.1 Principe de comparaison

L'évaluation consiste à comparer la vérité terrain `/carla/hero/odometry` à la sortie filtrée `/odometry/filtered`. Une association temporelle est effectuée pour former des paires GT–estimation.

4.2 Erreurs et métriques

Pour chaque paire, on calcule :

$$e_x = x_{est} - x_{gt}, \tag{2}$$

$$e_y = y_{est} - y_{gt}, \tag{3}$$

$$e_{2D} = \sqrt{e_x^2 + e_y^2}, \tag{4}$$

$$e_\psi = \mathrm{wrap}(\psi_{est} - \psi_{gt}) \in [-\pi, \pi]. \tag{5}$$

Les indicateurs principaux sont RMSE_x, RMSE_y, RMSE_{2D} et RMSE_{yaw}.

4.3 Visualisations

Chaque run est documenté par :

— une trajectoire 2D (GT vs estimation)

— des courbes d'erreurs (e_x, e_y, e_{2D})

— une courbe d'erreur yaw (e_ψ)

— un histogramme de e_{2D}.

4.4 Protocole expérimental

Pour chaque configuration (EKF ou UKF), le protocole est identique :

1. rejouer le rosbag en temps simulé
2. lancer le filtre avec la configuration considérée
3. enregistrer les paires GT–estimation
4. calculer automatiquement métriques et figures.

5 Résultats expérimentaux (RUN1 à RUN3)

Cette section présente les résultats *run par run* en suivant le même canevas : (i) contexte et objectif du test, (ii) configuration capteurs/filtre, (iii) analyse des figures et interprétation. Les figures associées sont celles générées automatiquement (trajectoire 2D, erreurs temporelles, yaw, histogramme).

5.1 RUN1 : EKF avec odométrie seule

5.1.1 Contexte et objectif

RUN1 sert de **référence** : le filtre EKF estime la pose du véhicule en ne s'appuyant que sur l'**odométrie** fournie par CARLA. L'objectif est double :

— valider la chaîne de bout en bout (rosbag $\rightarrow$ EKF $\rightarrow$ paires GT/EST $\rightarrow$ métriques/figures)

— établir un point de comparaison avant d'ajouter l'IMU et le GNSS (RUN2/RUN3).

5.1.2 Configuration (capteurs et principe EKF)

Le filtre produit une estimation `/odometry/filtered` comparée à la vérité terrain `/carla/hero/odometry`. Dans RUN1, l'odométrie joue le rôle de **mesure de correction principale** : à chaque message d'odométrie, l'EKF corrige l'état estimé en fonction de la confiance accordée à l'odométrie (covariance R_{odom}). Entre deux mesures, le filtre réalise une **prédiction** via son modèle de mouvement et fait croître l'incertitude selon Q. En pratique, avec une seule source de mesure dominante, la sortie filtrée a tendance à **suivre de très près** l'odométrie.

5.1.3 Résultats et interprétation

Trajectoire 2D (GT vs estimation). La Figure 1 montre une superposition quasi complète entre la trajectoire estimée et la vérité terrain. Sur cette séquence CARLA, l'odométrie reste fortement cohérente avec la référence simulée ; cela explique que RUN1 apparaisse très performant sur la position.

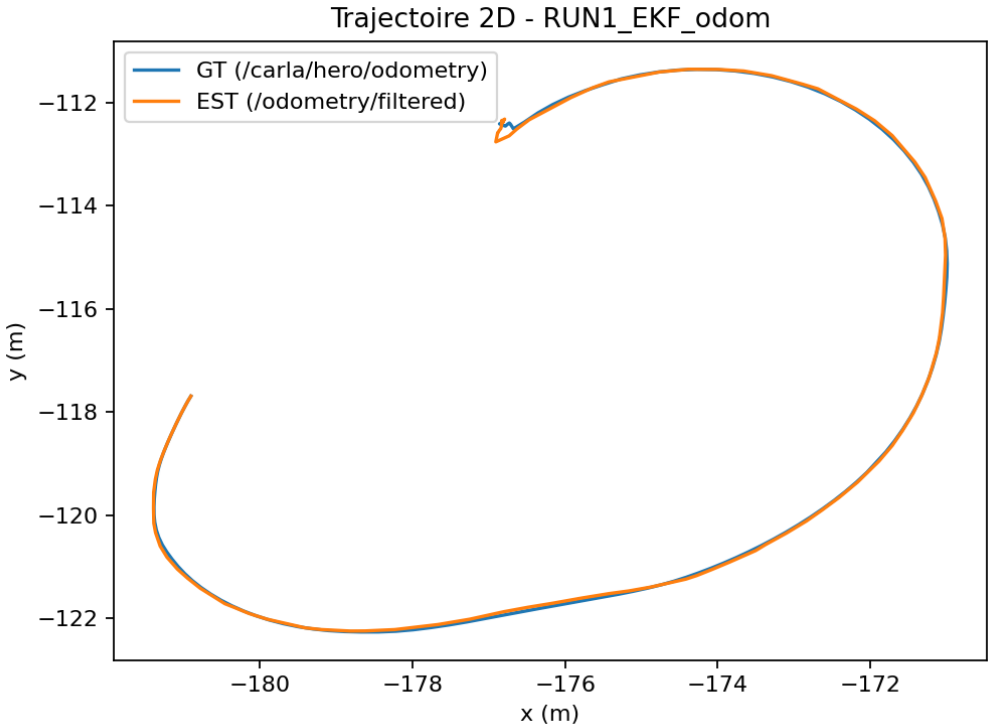


Figure 1. Trajectoire 2D : GT vs estimation EKF

Erreurs de position (e_x, e_y, e_{2D}). La Figure 2 indique que les erreurs restent globalement faibles, avec quelques variations/pics ponctuels. Ces pics ne signifient pas nécessairement une "instabilité" du filtre : ils peuvent aussi provenir de l'**appariement temporel** entre flux GT et estimation. L'histogramme de la Figure 3 confirme que l'essentiel des erreurs e_{2D} est concentré près de zéro.

Erreur d'orientation (yaw). La Figure 4 présente une erreur yaw relativement contenue sur l'ensemble de la séquence. Ce comportement est cohérent avec RUN1 : sans IMU, le yaw est principalement porté par l'odométrie (et le modèle interne), ce qui évite d'introduire une source inertielle potentiellement mal pondérée.

5.1.4 Conclusion RUN1

RUN1 valide la chaîne expérimentale et fournit une baseline solide : sur des données CARLA propres, l'odométrie est très cohérente avec la vérité terrain, ce qui conduit à de faibles erreurs pour l'EKF. Ce résultat doit toutefois être interprété avec prudence : en conditions réelles, l'odométrie dérive généralement au cours du temps. L'intérêt de RUN2/RUN3 est précisément d'évaluer l'apport (et les limites) des capteurs supplémentaires lorsque la situation devient plus discriminante (biais IMU, GNSS bruité, ou odométrie dégradée artificiellement).

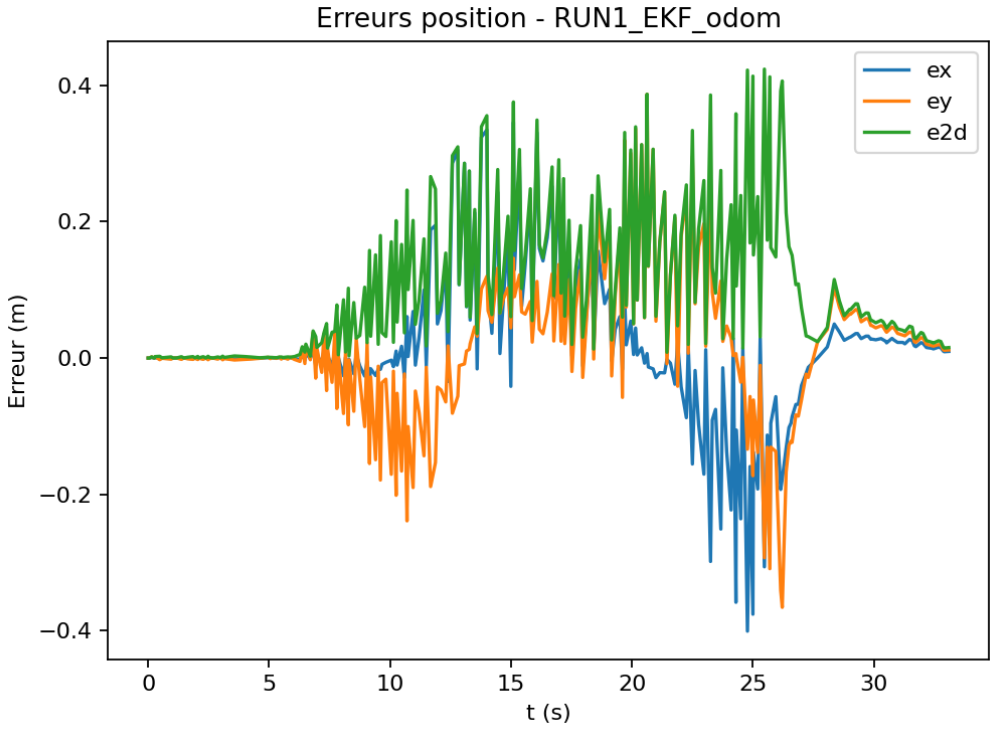


FIGURE 2. Erreurs temporelles de position : $e_x(t)$, $e_y(t)$ et $e_{2D}(t)$.

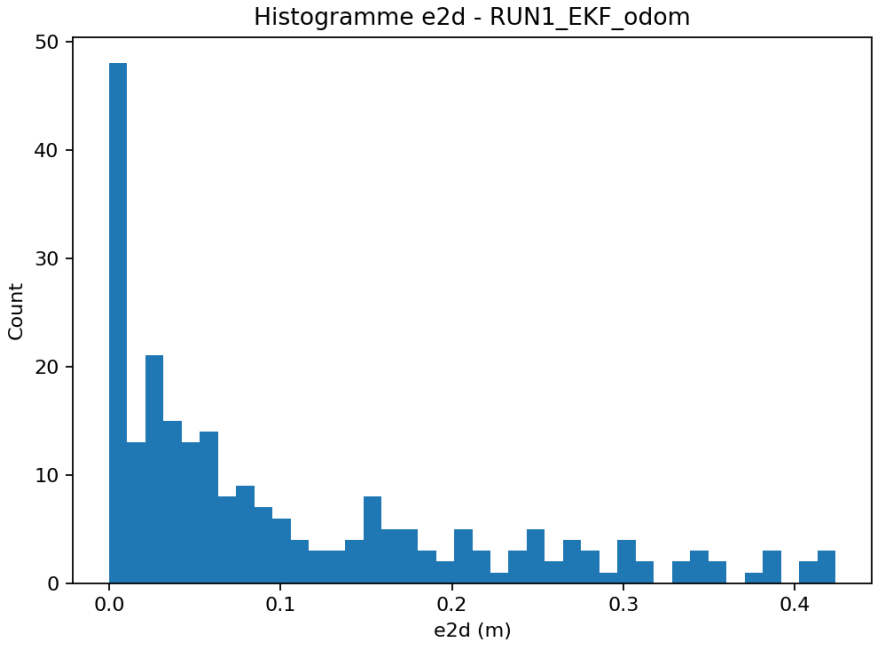


FIGURE 3. Histogramme de l'erreur 2D e_{2D}. La distribution est majoritairement concentrée vers les faibles erreurs.

5.2 RUN2 : EKF avec odométrie + IMU

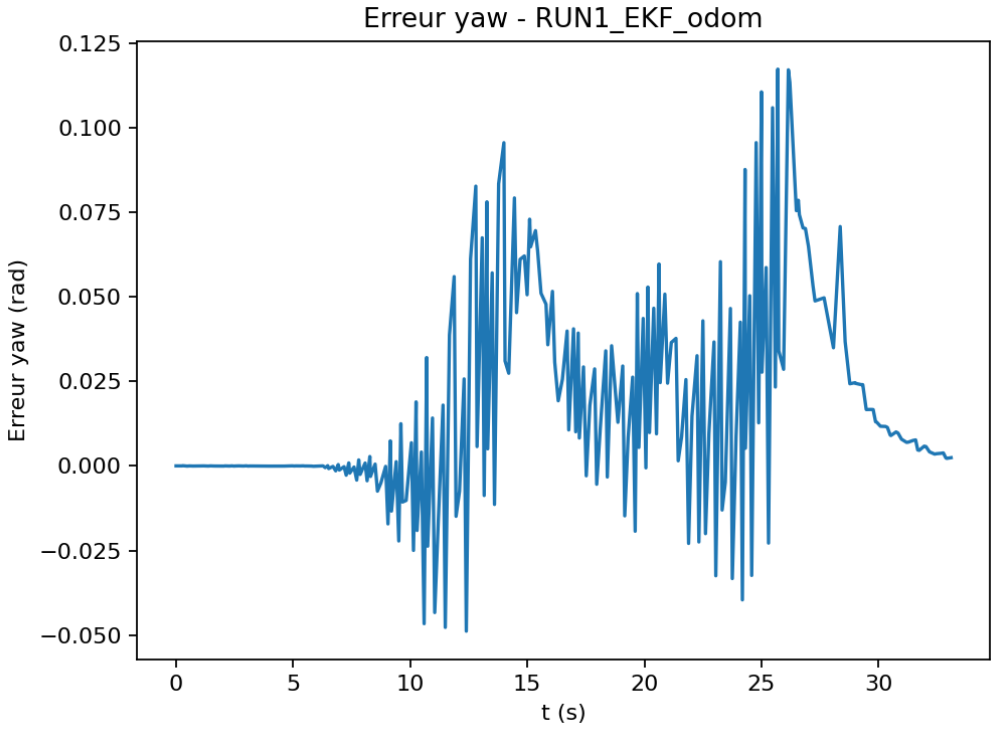


Figure 4. Erreur yaw $e_\psi(t)$ (différence d'orientation entre estimation et GT).

5.2.1 Contexte et objectif

RUN2 évalue l'apport de l'**IMU** en complément de l'odométrie. L'idée est d'améliorer la robustesse de l'estimation, en particulier sur la dynamique et l'orientation (yaw), en injectant des informations inertielles (orientation/vitesse angulaire et éventuellement accélération). RUN2 permet aussi d'identifier si l'IMU est correctement pondérée (covariances) et si les repères sont cohérents.

5.2.2 Configuration (capteurs et principe de fusion)

Le filtre EKF fusionne :

— l'odométrie `/carla/hero/odometry` (contrainte principale sur la position),

— l'IMU `/carla/hero/imu` (orientation/dynamique angulaire selon configuration).

Dans la configuration retenue, l'IMU influence directement l'orientation (yaw) et la vitesse angulaire yaw. L'estimation reste publiée sur `/odometry/filtered` et comparée à la vérité terrain.

5.2.3 Résultats et interprétation

Trajectoire 2D (GT vs estimation). La Figure 5 montre une trajectoire estimée très proche de la vérité terrain, comparable à RUN1. Sur cette séquence simulée, l'odométrie étant déjà très cohérente, l'ajout de l'IMU ne modifie pas fortement l'erreur de position.

Erreurs de position (e_x, e_y, e_{2D}). Les erreurs de position (Figure 6) restent du même ordre de grandeur que RUN1, avec des variations et quelques pics, en partie imputables à la synchronisation et à la dynamique de la trajectoire. L'histogramme de la Figure 7 confirme une distribution toujours majoritairement concentrée sur de faibles valeurs de e_{2D}.

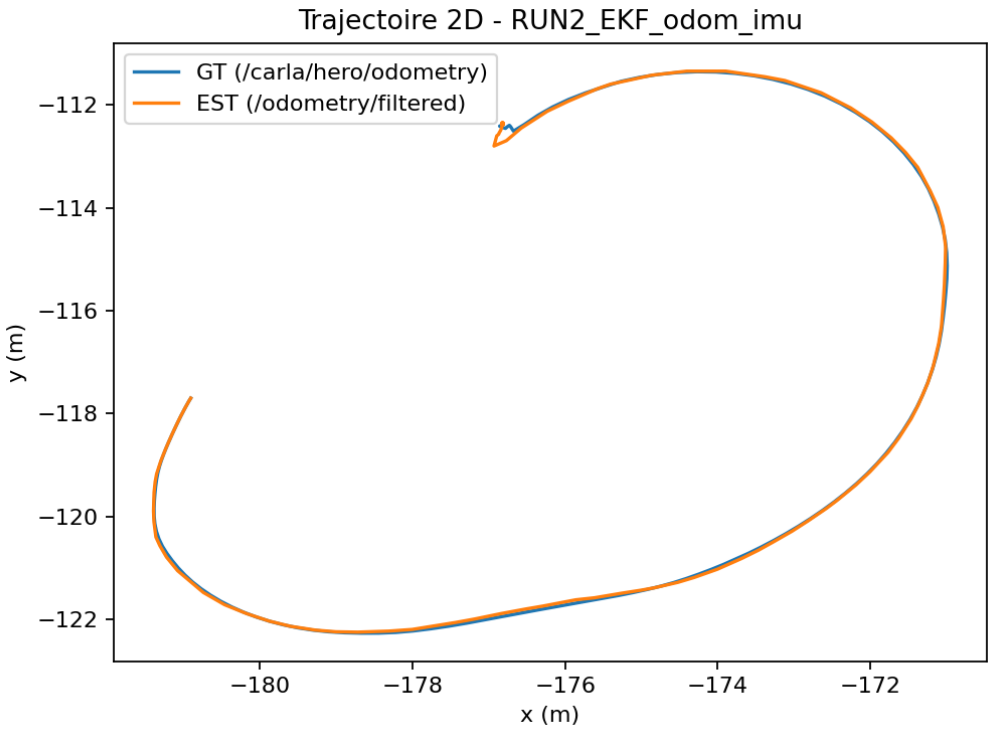


Figure 5. Trajectoire 2D : GT vs estimation EKF (odom + IMU).

Figure 6. Erreurs temporelles de position : $e_x(t)$, $e_y(t)$ et $e_{2D}(t)$.

Erreur d'orientation (yaw) : effet de l'IMU. La principale différence avec RUN1 apparaît sur l'orientation (Figure 8). L'erreur yaw est nettement plus élevée et présente des phases où l'estimation s'écarte significativement de la GT. Ce comportement suggère que, dans la

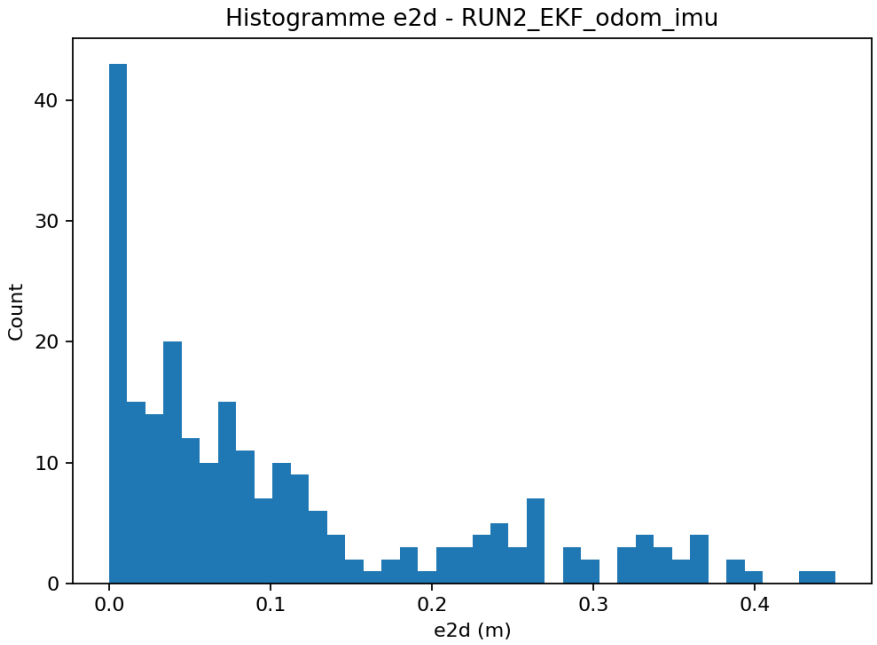


Figure 7. Histogramme de l'erreur 2D e_{2D}.

configuration actuelle, l'IMU (ou sa pondération) influence fortement le yaw et peut introduire un écart. Deux causes sont généralement à considérer :

— **pondération/covariances** : si les covariances IMU sont trop faibles (capteur considéré trop fiable), le filtre peut sur-corriger le yaw ;

— **cohérence repères/mesures** : une incohérence de repères ou une orientation IMU peu réaliste en simulation peut dégrader la composante yaw.

Ainsi, RUN2 met en évidence que l'ajout d'un capteur n'est bénéfique que si son intégration est cohérente (choix des champs fusionnés et réglage des incertitudes).

5.2.4 Conclusion RUN2

Sur cette séquence CARLA, RUN2 conserve des performances en position proches de RUN1, mais révèle une **sensibilité marquée du yaw** lorsque l'IMU est fusionnée. Ce résultat souligne l'importance du paramétrage (covariances R) et du choix des composantes corrigées par l'IMU. La suite (RUN3) permet de compléter l'analyse en ajoutant une contrainte GNSS, principalement sur la position globale.

5.3 RUN3 : EKF avec odométrie + IMU + GNSS

5.3.1 Contexte et objectif

RUN3 étudie l'apport d'une contrainte **GNSS** en complément de l'odométrie et de l'IMU. Dans une architecture classique, le GNSS sert principalement à **recadrer la position globale** et à limiter la dérive sur des durées longues. L'objectif de RUN3 est donc d'observer si l'ajout GNSS modifie la précision en position et la stabilité globale de la trajectoire, tout en vérifiant la cohérence de la chaîne (transformation GNSS et fusion).

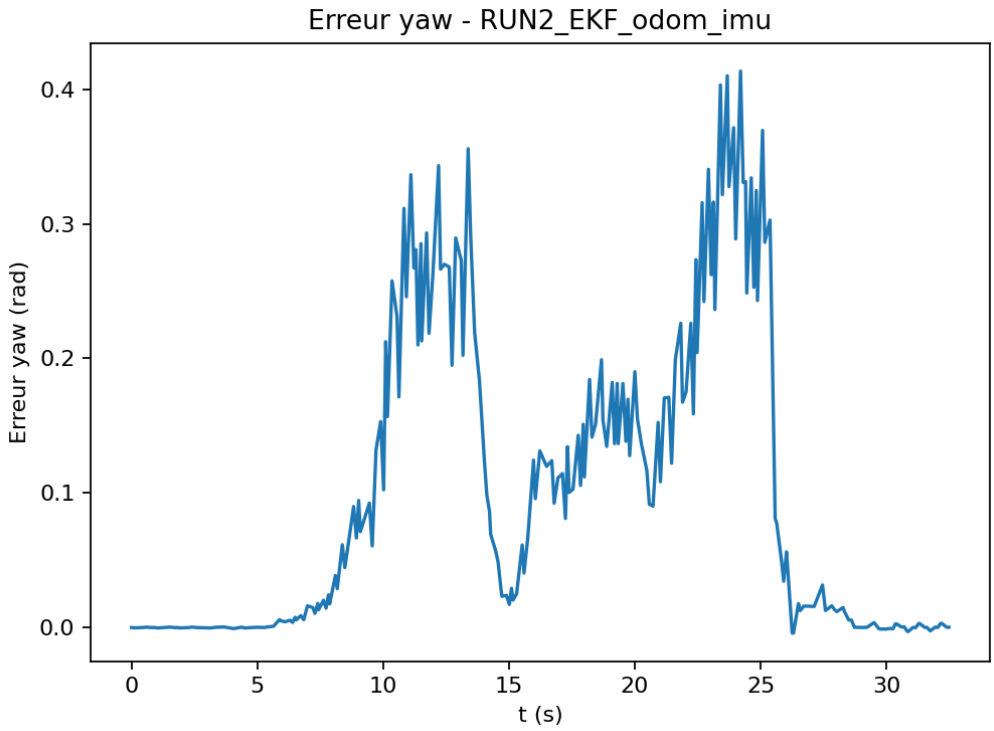


FIGURE 8. Erreur yaw $e_\psi(t)$: l'ajout de l'IMU augmente l'erreur d'orientation dans la configuration actuelle.

5.3.2 Configuration (pile GNSS et fusion)

Le GNSS (`/carla/hero/gnss`) est d'abord converti en une odométrie exploitable via `navsat_transform_node`, produisant un topic de type odométrie (ex. `/odometry/gps`). Cette information est ensuite fusionnée dans un filtre (local ou global selon la configuration retenue) avec l'odométrie et l'IMU. Dans la configuration utilisée, l'apport GNSS porte principalement sur la **position** (x, y), tandis que l'orientation yaw reste majoritairement contrainte par l'odométrie et/ou l'IMU.

5.3.3 Résultats et interprétation

Trajectoire 2D (GT vs estimation). La Figure 9 montre une superposition très proche entre la trajectoire estimée et la vérité terrain, similaire à RUN1 et RUN2. Sur une séquence courte et peu bruitée, le bénéfice attendu du GNSS sur la position reste limité, car l'odométrie simulée est déjà fortement cohérente avec la GT.

Erreurs de position (e_x, e_y, e_{2D}). Les erreurs de position (Figure 10) restent du même ordre de grandeur que RUN2. L'histogramme (Figure 11) confirme une distribution concentrée sur de faibles erreurs. Dans ce contexte, l'ajout du GNSS ne produit pas une amélioration spectaculaire car le scénario est relativement facile : faible durée et capteurs simulés cohérents.

Erreur d'orientation (yaw). La Figure 12 indique que l'erreur yaw reste comparable à RUN2. Cela est cohérent avec la configuration : le GNSS contraint principalement (x, y) et n'apporte pas directement d'information sur l'orientation. Par conséquent, si l'IMU dégrade le yaw dans RUN2, l'ajout GNSS dans RUN3 ne corrige pas ce point.

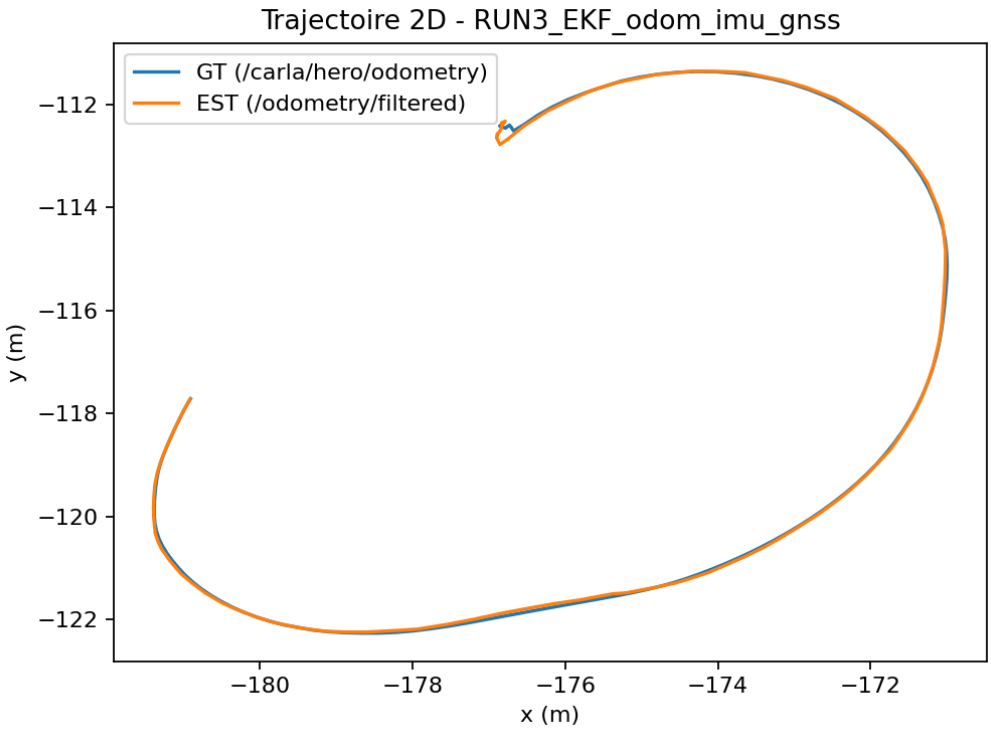


FIGURE 9. Trajectoire 2D : GT vs estimation EKF (odom + IMU + GNSS).

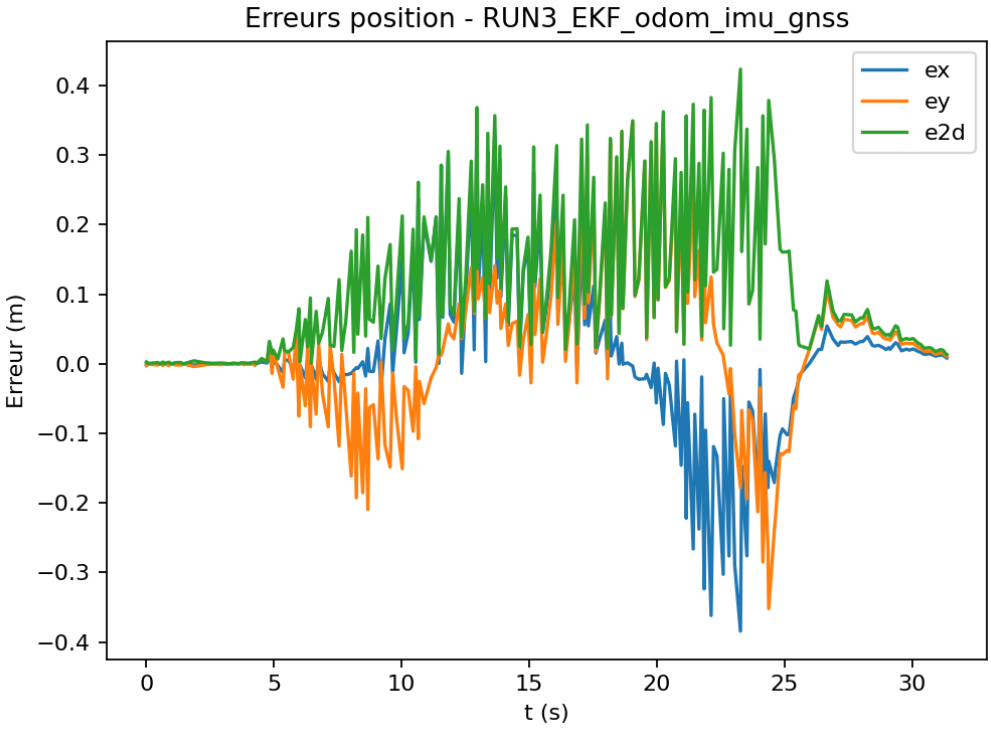


FIGURE 10. Erreurs temporelles de position : $e_x(t)$, $e_y(t)$ et $e_{2D}(t)$.

5.3.4 Conclusion RUN3

RUN3 montre que l'ajout GNSS ne modifie pas fortement les performances sur cette séquence simulée courte et cohérente. Son rôle principal est le recadrage global en position, qui devient

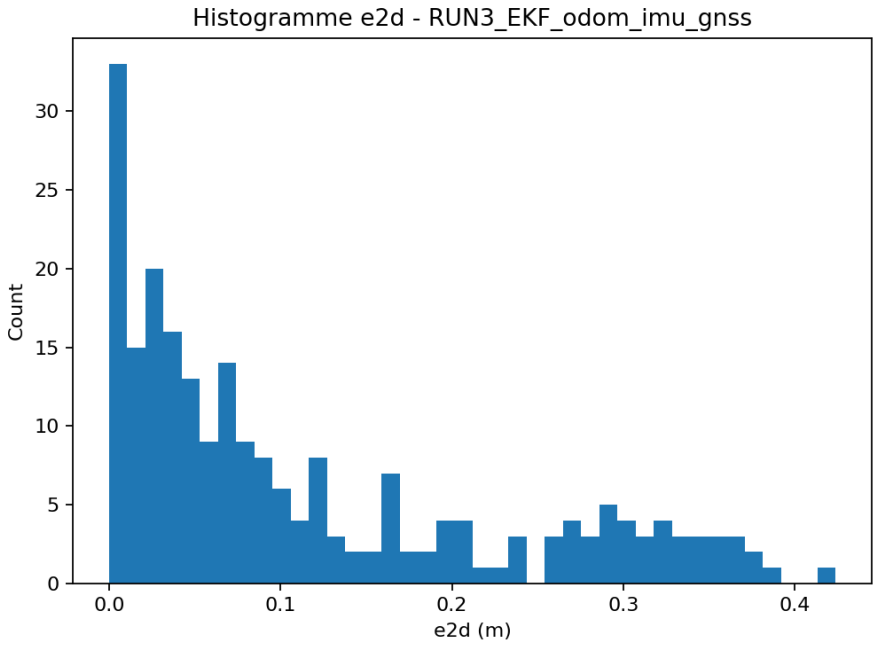


FIGURE 11. Histogramme de l'erreur 2D e_{2D}.

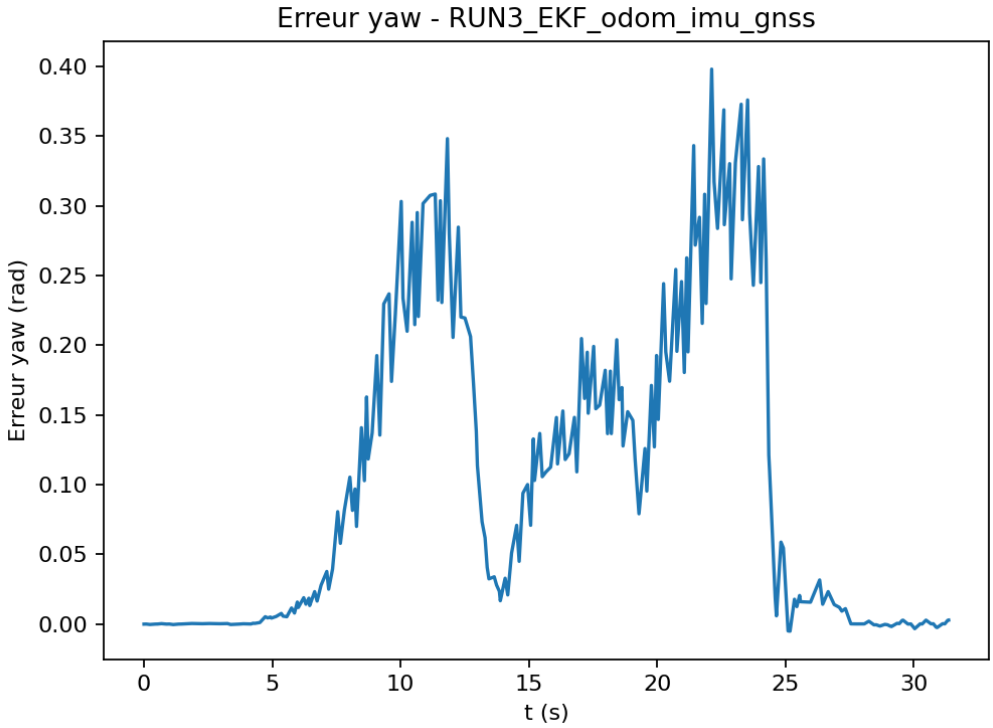


FIGURE 12. Erreur yaw $e_\psi(t)$. Le GNSS recadre la position mais n'agit pas directement sur l'orientation.

surtout pertinent lorsque la trajectoire est longue, lorsque l'odométrie dérive, ou lorsque l'on introduit du bruit réaliste. Par ailleurs, l'erreur yaw reste dominée par la manière dont l'IMU est intégrée, ce qui explique la proximité des comportements yaw entre RUN2 et RUN3.

5.4 Synthèse comparative (RUN1 à RUN3)

Afin de comparer les configurations de manière objective, nous synthétisons ici les tendances observées sur les métriques globales (RMSE) et sur les figures (stabilité, pics, comportement yaw).

5.4.1 Tableau comparatif (RMSE)

Le tableau 3 résume les performances des trois runs EKF. Les valeurs sont issues des fichiers de métriques générés automatiquement lors de l'analyse.

Run	**Capteurs**	RMSE$_x$ (m)	RMSE$_y$ (m)	RMSE$_{2D}$ (m)	RMSE$_{yaw}$ (rad)
RUN1	Odom	0.140	0.115	0.181	0.036
RUN2	Odom+IMU	0.124	0.120	0.173	0.151
RUN3	Odom+IMU+GNSS	0.124	0.119	0.172	0.153

Table 3. Comparaison RUN1–RUN3 (EKF) – métriques globales.

5.4.2 Lecture et interprétation

Deux constats se dégagent :

— **Position : différences faibles entre RUN1, RUN2 et RUN3.** Sur cette séquence CARLA courte et cohérente, l'odométrie est déjà très proche de la vérité terrain. L'ajout de l'IMU et du GNSS ne modifie donc pas fortement les erreurs de position : les RMSE$_{2D}$ restent proches.

— **Yaw : dégradation nette dès l'ajout de l'IMU (RUN2/RUN3).** L'erreur yaw augmente fortement dans RUN2, et reste comparable dans RUN3. Cela s'explique par le fait que le GNSS contraint principalement (x, y), et n'apporte pas directement d'information sur l'orientation. L'orientation dépend donc surtout de l'intégration IMU et de sa pondération (covariances, champs fusionnés, cohérence des repères).

Ces résultats motivent deux suites naturelles : (i) tester une variante de fusion IMU (par exemple limiter l'IMU au gyro uniquement), et (ii) comparer un modèle de fusion alternatif (UKF) sur la configuration odom+IMU.

5.5 Études complémentaires : influence des incertitudes (Q et P_0)

Ces expériences visent à relier le rôle théorique des incertitudes du filtre EKF à des résultats mesurables :

— Q (`process_noise_covariance`) : incertitude associée au modèle de prédiction ;

— P_0 (`initial_estimate_covariance`) : incertitude sur l'état initial.

Nous présentons d'abord une étude en conditions *non bruitées*, puis une étude en conditions *bruitées* (plus discriminante).

5.5.1 Étude paramétrique en conditions non bruitées

Objectif. Évaluer l'impact de Q et P_0 lorsque les mesures simulées sont très cohérentes avec la vérité terrain.

Protocole. Trois configurations sont comparées (rosbag identique) : **Qlow** (Q faible), **Qhigh** (Q élevé) et **P0high** (P_0 élevé). Les métriques RMSE sont calculées à partir des paires GT–estimation.

Résultats. Le tableau 4 montre des erreurs extrêmement faibles (ordre de 10^{-6} m), avec des différences limitées entre configurations.

Config	RMSE$_x$ (m)	RMSE$_y$ (m)	RMSE$_{2D}$ (m)	RMSE$_{yaw}$ (rad)
Qlow	3.63×10^{-6}	3.55×10^{-6}	5.08×10^{-6}	3.80×10^{-7}
Qhigh	1.31×10^{-6}	8.76×10^{-7}	1.58×10^{-6}	0
P0high	5.27×10^{-7}	5.17×10^{-7}	7.38×10^{-7}	1.02×10^{-7}

Table 4. Étude (non bruitée) : influence de Q et P_0 sur les RMSE.

Interprétation. Dans ce scénario CARLA propre, l'odométrie reste très cohérente avec la vérité terrain ; la correction domine et masque en grande partie l'effet de Q et P_0 sur la position. Cette étude valide toutefois la chaîne expérimentale et confirme que, sur données idéales, le réglage d'incertitudes peut avoir un impact faible sur les métriques globales.

5.5.2 Étude paramétrique en conditions bruitées (odométrie dégradée)

Objectif. Rendre l'analyse plus discriminante en dégradant artificiellement l'odométrie (bruit sur position/orientation), afin d'observer des différences plus nettes liées à Q et P_0.

Protocole. Une odométrie bruitée `/carla/hero/odometry_noisy` est générée à partir de `/carla/hero/odometry`. L'EKF est relancé en remplaçant l'entrée odométrie par cette version bruitée. Trois configurations sont comparées : **Qlow_noisy**, **Qhigh_noisy** et **P0high_noisy**.

Résultats. Le tableau 5 illustre que, sous bruit, les réglages deviennent nettement discriminants : les RMSE varient fortement selon Q et P_0.

Config	RMSE$_x$ (m)	RMSE$_y$ (m)	RMSE$_{2D}$ (m)
Qlow_noisy	1.681	2.303	2.851
Qhigh_noisy	5.108	3.310	6.087
P0high_noisy	0.028	0.033	0.043

Table 5. Étude (bruitée) : influence de Q et P_0 sur les RMSE (odométrie dégradée).

Interprétation. Ces résultats confirment l'effet attendu des incertitudes :

— un Q élevé rend le filtre plus réactif mais peut conduire à suivre davantage une mesure bruitée, ce qui dégrade la précision ;

— un Q faible rend le filtre plus "rigide" et peut limiter l'effet du bruit, au prix d'une moindre réactivité ;

— P_0 influence principalement le démarrage et la convergence initiale ; sur une fenêtre courte, il peut favoriser une convergence rapide, ce qui doit être interprété avec prudence.

Conclusion générale. La comparaison non bruitée vs bruitée montre que l'étude des paramètres Q et P_0 est peu discriminante sur un scénario simulé idéal, mais devient essentielle dès que l'on introduit du bruit réaliste. Ce résultat motive l'utilisation de scénarios plus complexes et prépare la comparaison avec un modèle de fusion alternatif (UKF).

Les expériences précédentes montrent que l'interprétation des performances dépend fortement du contexte (scénario propre vs bruité) et du réglage des incertitudes. Nous complétons donc l'étude en évaluant un modèle de fusion alternatif : l'Unscented Kalman Filter (UKF), afin de comparer sa stabilité à l'EKF dans les mêmes conditions.

5.6 RUN5 : UKF avec odométrie + IMU

5.6.1 Contexte et objectif

RUN5 introduit un modèle de fusion alternatif à l'EKF : l'**UKF** (`robot_localization ukf_node`). L'objectif est de comparer EKF et UKF sur les mêmes données et avec la même méthodologie d'évaluation. Afin d'obtenir une configuration stable et exploitable, l'IMU est intégrée de manière progressive : dans RUN5, elle est utilisée en *gyro-only* (vitesse angulaire yaw), ce qui permet d'apporter une information dynamique sur la rotation tout en limitant l'influence d'une orientation IMU potentiellement mal pondérée.

5.6.2 Configuration et point technique (stabilité UKF)

L'UKF s'est révélé plus sensible que l'EKF au conditionnement numérique. Les messages CARLA (`/carla/hero/imu` et `/carla/hero/odometry`) présentent des covariances nulles, ce qui peut conduire l'UKF à produire des états non valides . Pour stabiliser l'exécution, un prétraitement a été ajouté : un nœud republie `/carla/hero/imu_cov` et `/carla/hero/odometry_cov` en imposant (i) des covariances non nulles et (ii) des timestamps cohérents avec `/clock`. Le filtre UKF fusionne alors l'odométrie (position/vitesse) et l'IMU en *gyro-only*.

5.6.3 Métriques globales

Le run contient 245 paires GT–estimation sur une durée d'environ 9.7 s. Les métriques obtenues sont :

— RMSE$_x = 0.135$ m, RMSE$_y = 0.125$ m,

— RMSE$_{2D} = 0.184$ m,

— RMSE$_{yaw} = 0.097$ rad (soit 5.57°).

5.6.4 Analyse des figures

Trajectoire 2D (GT vs estimation). La Figure 13 montre une superposition globalement satisfaisante entre la trajectoire estimée et la vérité terrain. Comme dans les runs EKF, la contrainte odométrique reste dominante en position, ce qui explique la bonne reconstruction de la forme globale de la trajectoire.

Erreurs de position (e_x, e_y, e_{2D}). La Figure 14 indique une erreur 2D généralement comprise entre 0 et 0.3 m, avec quelques pics plus élevés. Comme pour RUN1–RUN3, ces pics peuvent être liés à la dynamique du mouvement et/ou à l'appariement temporel entre les flux GT et estimation (jitter lors de la mise en correspondance).

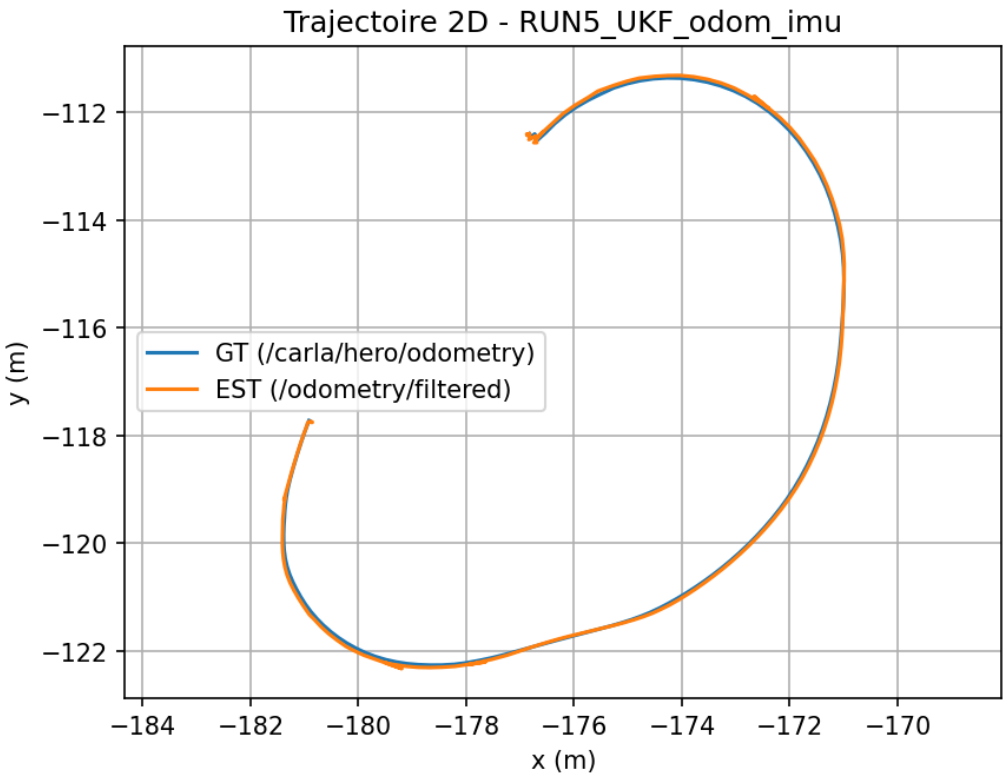


FIGURE 13. Trajectoire 2D : GT (`/odometry`) vs estimation UKF (`/odometry/filtered`).

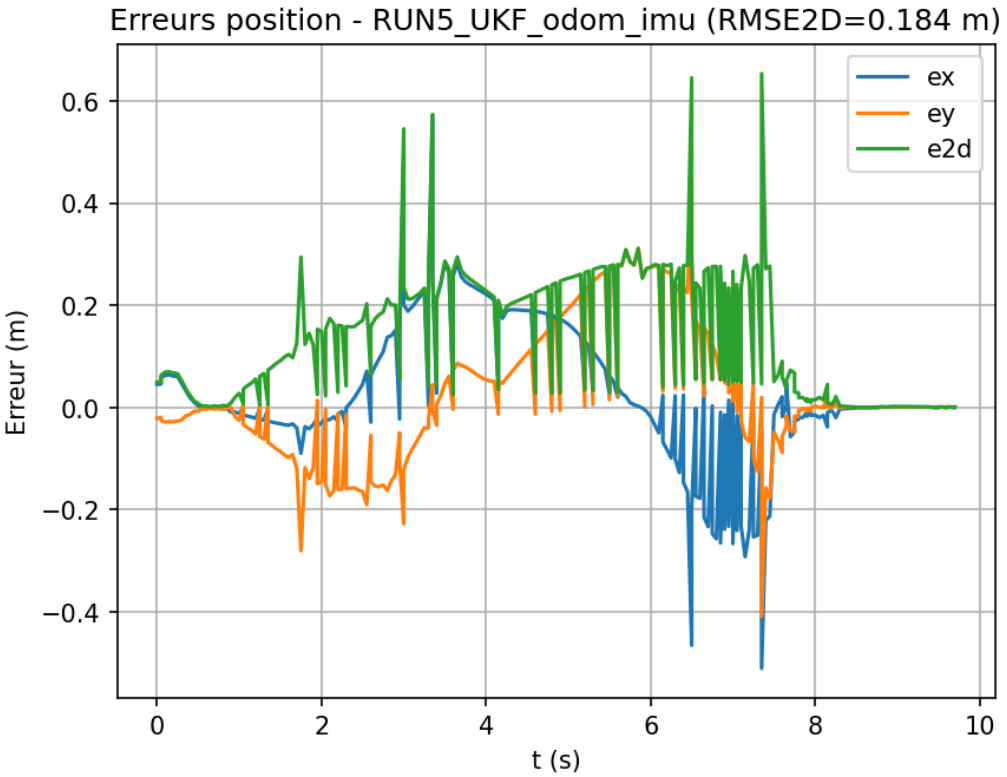


FIGURE 14. Erreurs temporelles de position : $e_x(t)$, $e_y(t)$ et $e_{2D}(t)$.

L'histogramme de la Figure 15 confirme que la majorité des erreurs e_{2D} est concentrée sous

$\sim$0.3 m, avec une queue correspondant à quelques instants où l'erreur augmente davantage.

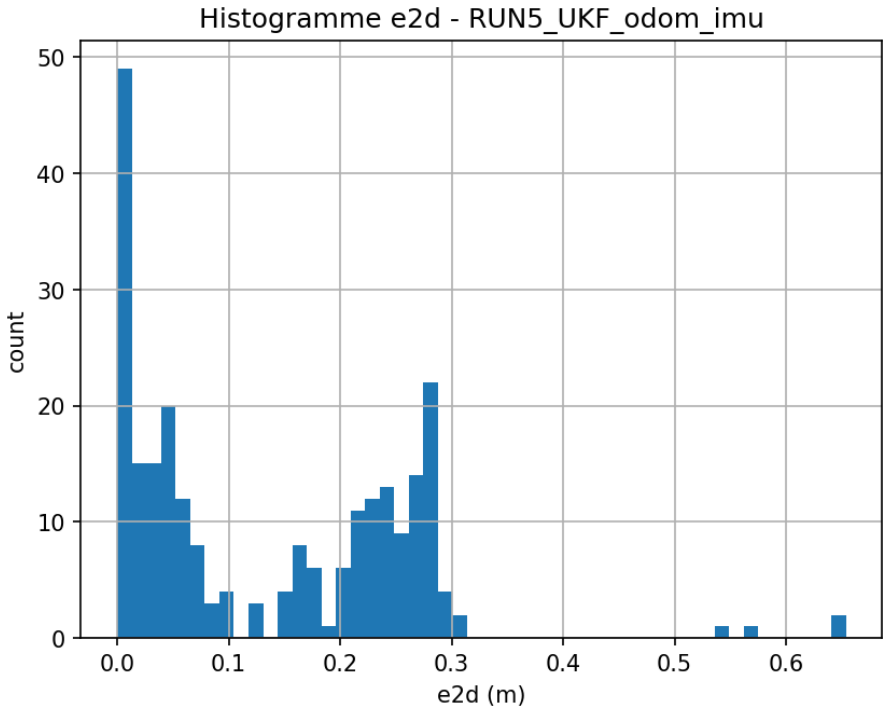


Figure 15. Histogramme de l'erreur 2D e_{2D}.

Erreur yaw. La Figure 16 montre une erreur yaw globalement contenue, avec des pics ponctuels. La RMSE yaw atteint 0.097 rad (5.6°). Dans cette configuration, l'IMU *gyro-only* fournit une information sur la dynamique de rotation (vyaw) sans contraindre directement l'orientation absolue, ce qui contribue à une meilleure stabilité yaw par rapport au run EKF odom+IMU.

5.6.5 Comparaison EKF vs UKF (odom + IMU)

Le tableau 6 compare RUN2 (EKF odom+IMU) et RUN5 (UKF odom+IMU en *gyro-only*). Les runs utilisent le même rosbag et la même méthode d'évaluation.

Run / Filtre	RMSE_x (m)	RMSE_y (m)	RMSE_{2D} (m)	RMSE_{yaw} (rad)
RUN2 : EKF (odom+IMU)	0.124	0.120	0.173	0.151
RUN5 : UKF (odom+IMU, gyro-only)	0.135	0.125	0.184	0.097

Table 6. Comparaison EKF vs UKF sur odométrie + IMU (RUN2 vs RUN5).

Analyse. Sur ce scénario simulé, les performances en position restent du même ordre de grandeur entre EKF et UKF, ce qui est cohérent avec la forte cohérence de l'odométrie avec la vérité terrain sur une séquence courte. En revanche, l'UKF améliore l'orientation : l'erreur yaw est réduite (0.151 $\rightarrow$ 0.097 rad). L'orientation apparaît ainsi comme la composante la plus sensible au modèle de fusion et au paramétrage associé. Enfin, RUN5 met en évidence un point pratique : l'UKF requiert des covariances non nulles et des timestamps cohérents pour être numériquement stable, ce qui justifie l'étape de prétraitement ajoutée dans notre pipeline.

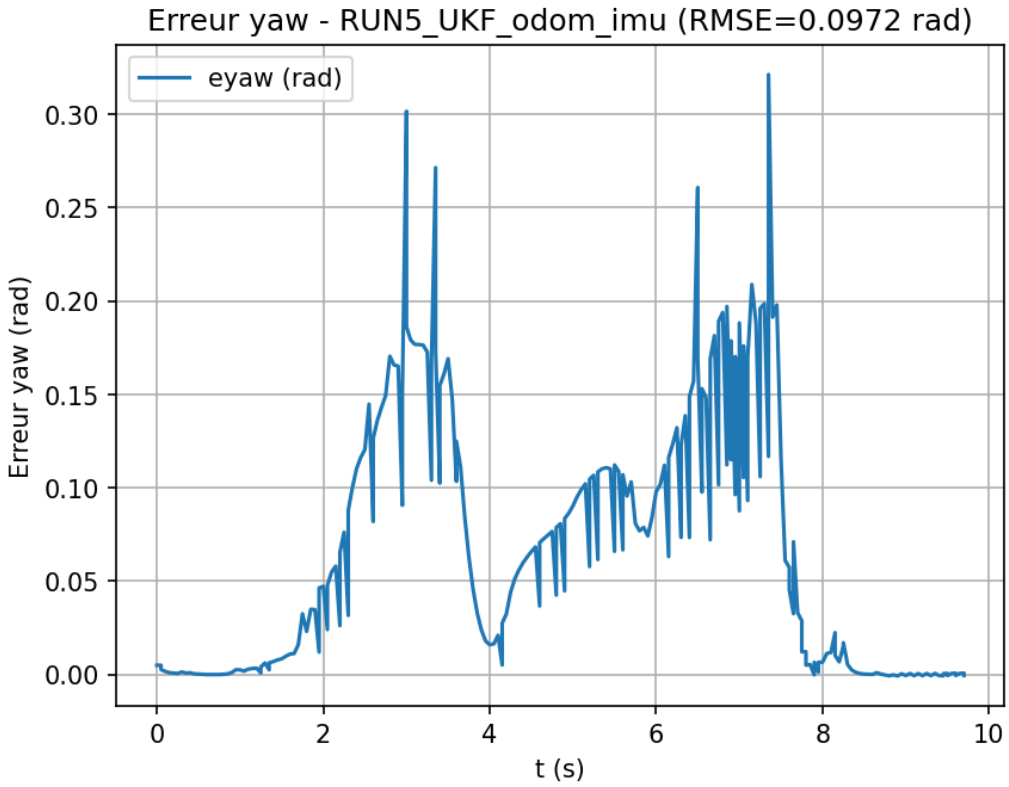


FIGURE 16. Erreur yaw $e_\psi(t)$ (RMSE$_{yaw} = 0.097$ rad).

5.6.6 Conclusion RUN5

RUN5 montre qu'une configuration UKF stable peut être obtenue en simulation à condition de fournir des covariances non nulles et un horodatage cohérent. Les performances en position restent comparables à celles obtenues avec l'EKF, tandis que l'erreur yaw est réduite, ce qui confirme l'intérêt de comparer plusieurs modèles de fusion lorsque l'orientation est un critère clé.

5.7 RUN6 : UKF avec odométrie + IMU

5.7.1 Contexte et objectif

RUN6 prolonge RUN5 (UKF stable) en activant une fusion IMU plus complète : l'IMU corrige désormais **le yaw** et **la vitesse angulaire yaw** (*yaw* + *vyaw*), alors que RUN5 n'utilisait que la vitesse angulaire (*gyro-only*). L'objectif est d'évaluer l'apport direct de la mesure d'orientation IMU sur la stabilité angulaire, tout en observant l'impact sur la précision en position.

5.7.2 Métriques globales

RUN6 contient 252 paires GT–estimation sur une durée d'environ 9.8 s. Les métriques obtenues sont :

— RMSE$_x = 0.144$ m, RMSE$_y = 0.138$ m,

— RMSE$_{2D} = 0.199$ m,

— RMSE$_{yaw} = 0.0275$ rad (soit 1.58°).

5.7.3 Analyse des figures

Trajectoire 2D (GT vs estimation). La Figure 17 montre une trajectoire estimée globalement cohérente avec la vérité terrain. La forme globale est correctement reproduite, avec un léger écart visible localement, cohérent avec les erreurs de position mesurées.

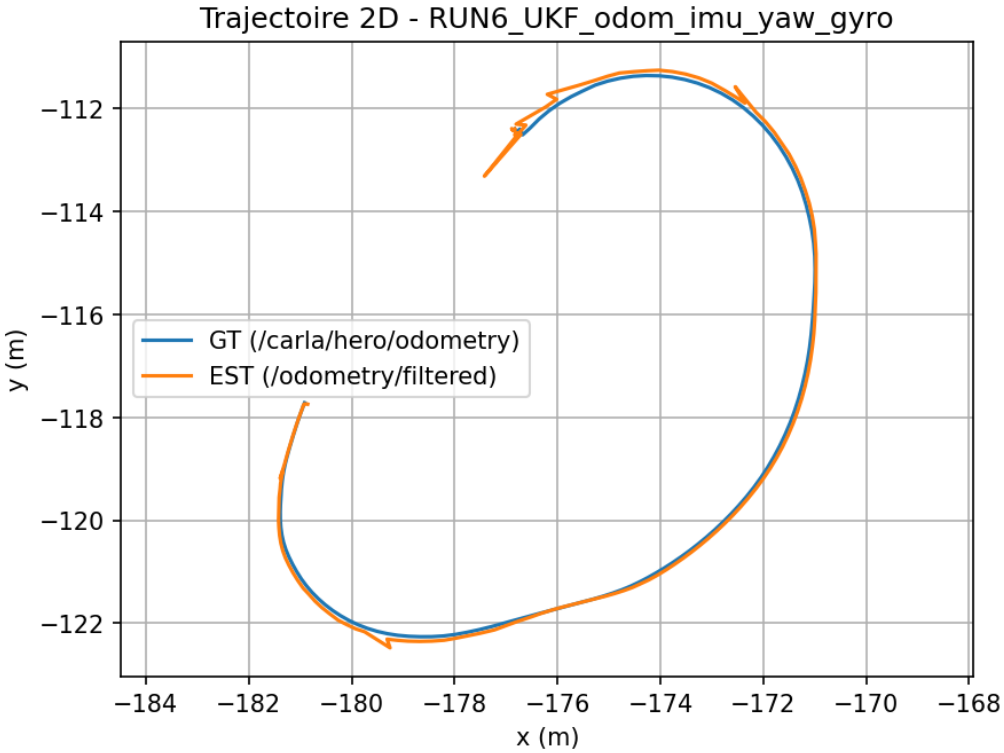


Figure 17. Trajectoire 2D : GT (`/odometry`) vs estimation UKF (`/odometry/filtered`).

Erreurs de position (e_x, e_y, e_{2D}). La Figure 18 indique une erreur 2D typiquement comprise entre 0 et 0.3 m, avec des pics plus importants vers la fin de la séquence (jusqu'à $\sim$1 m). Comme pour les runs précédents, ces pics peuvent résulter de la dynamique (phases de rotation) et/ou de l'appariement temporel (jitter) lors de la mise en correspondance GT–estimation.

L'histogramme de la Figure 19 confirme que la majorité des erreurs e_{2D} reste concentrée sous $\sim$0.3 m, avec une queue (peu fréquente) correspondant aux instants où l'erreur augmente fortement.

Erreur yaw. La Figure 20 met en évidence une erreur yaw très faible sur la quasi-totalité de la séquence. On observe des variations limitées autour de zéro, avec un pic isolé, mais la RMSE yaw reste très basse (0.0275 rad, soit 1.58°). Ce résultat illustre l'apport direct de la correction d'orientation IMU dans l'UKF.

5.7.4 Comparaison RUN5 vs RUN6 (UKF)

Le tableau 7 compare l'effet de l'activation yaw IMU (RUN6) par rapport à la version *gyro-only* (RUN5).

Conclusion RUN6 L'activation du yaw IMU dans l'UKF améliore fortement l'orientation (erreur yaw divisée par $\sim$3.5 par rapport à RUN5), ce qui confirme l'intérêt d'exploiter la mesure

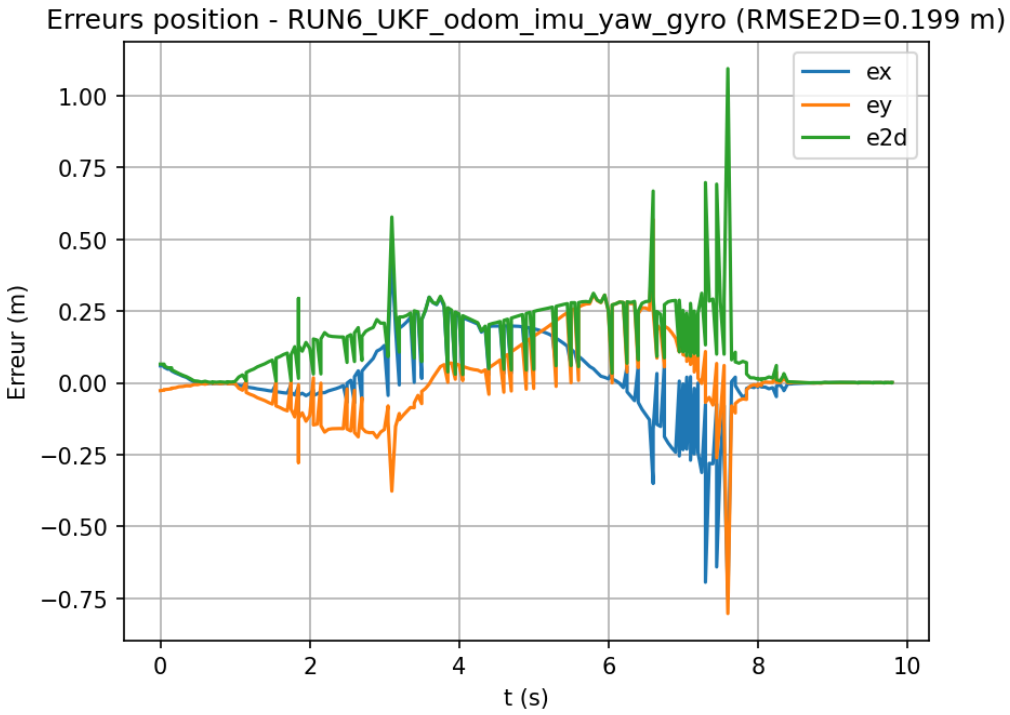


Figure 18. Erreurs temporelles de position : $e_x(t)$, $e_y(t)$ et $e_{2D}(t)$.

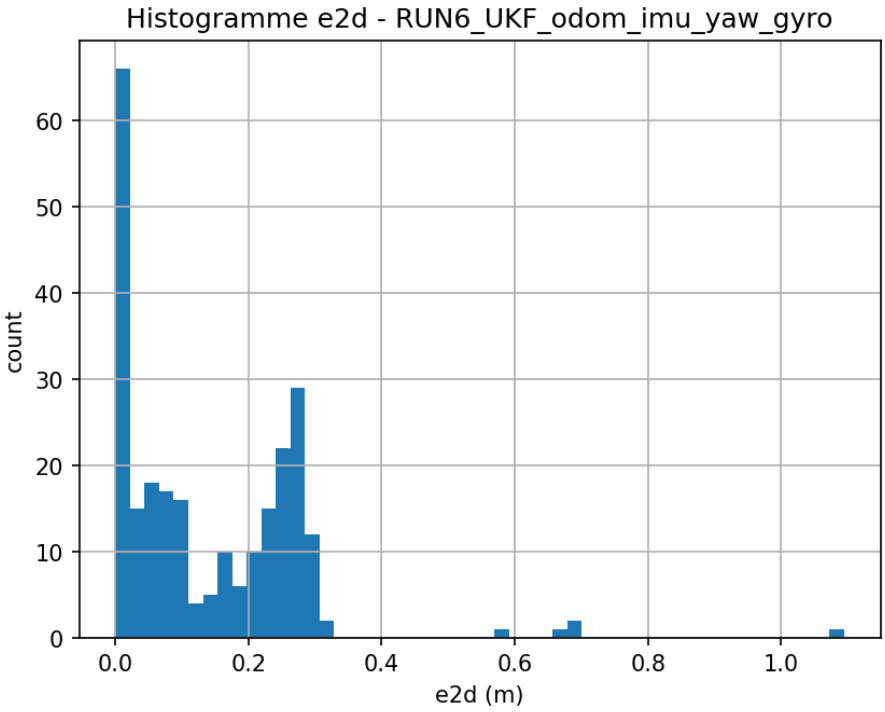


Figure 19. RUN6 : Histogramme de l'erreur 2D e_{2D}.

Run	RMSE$_x$ (m)	RMSE$_y$ (m)	RMSE$_{2D}$ (m)	RMSE$_{yaw}$ (rad)
RUN5 : UKF (gyro-only)	0.135	0.125	0.184	0.097
RUN6 : UKF (yaw + gyro)	0.144	0.138	0.199	0.0275

Table 7. Comparaison UKF : RUN5 (gyro-only) vs RUN6 (yaw + gyro).

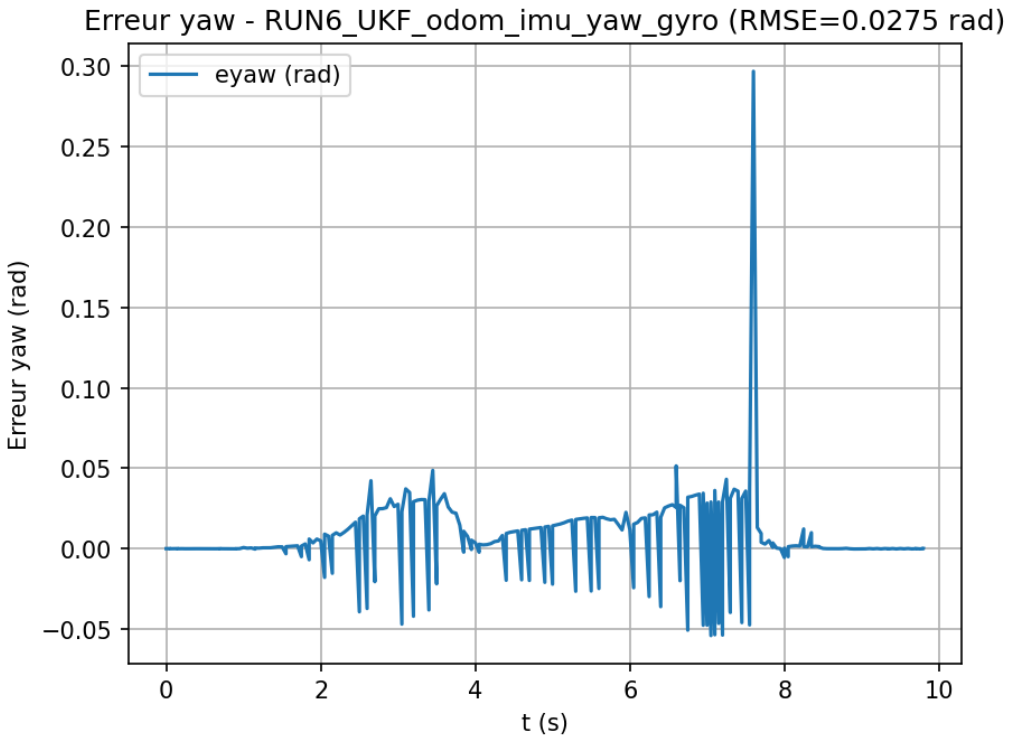


FIGURE 20. RUN6 : Erreur yaw $e_\psi(t)$ (RMSE$_{yaw} = 0.0275$ rad).

d'orientation lorsque celle-ci est cohérente et correctement pondérée. En contrepartie, l'erreur en position augmente légèrement (RMSE$_{2D}$), et quelques pics apparaissent. Ce résultat met en évidence un compromis classique en fusion multi-capteurs : renforcer une contrainte (orientation) peut modifier l'équilibre global du filtre et impacter la trajectoire planimétrique, ce qui justifie une analyse conjointe position/yaw et un réglage soigné des incertitudes.

5.8 Synthèse globale des résultats

Cette synthèse propose une lecture transversale des runs EKF (RUN1–RUN3), des runs UKF (RUN5–RUN6) et de l'étude paramétrique sur les incertitudes , en mettant l'accent sur les tendances robustes observées dans ce scénario CARLA.

5.8.1 (1) Position : métrique peu discriminante sur ce scénario court

Sur la séquence étudiée, les erreurs de position restent globalement du même ordre de grandeur d'une configuration à l'autre. En particulier, l'ajout de l'IMU et du GNSS ne modifie que marginalement les RMSE$_{2D}$ entre RUN1, RUN2 et RUN3 (cf. tableau 3). Cette observation est cohérente avec le caractère "propre" de la simulation : l'odométrie fournie par CARLA est déjà très cohérente avec la vérité terrain sur une durée relativement courte. Dans ces conditions, la correction odométrique domine l'estimation et masque partiellement l'apport des autres capteurs sur la position.

5.8.2 (2) Yaw : grandeur la plus sensible au modèle de fusion

À l'inverse, l'orientation (yaw) s'avère nettement plus discriminante. Dans la série EKF, l'intégration de l'IMU s'accompagne d'une augmentation de l'erreur yaw (RUN2/RUN3 vs RUN1, cf. tableau 3), ce qui suggère un fort impact du réglage des incertitudes IMU et des composantes effectivement

corrigées. Les runs UKF confirment que le yaw dépend fortement de la stratégie d'intégration inertielle : la configuration UKF "gyro-only" (RUN5) améliore le yaw par rapport à l'EKF odom+IMU (cf. tableau 6), et l'activation explicite de la correction yaw dans RUN6 réduit encore fortement l'erreur d'orientation (cf. tableau 7). En pratique, cela indique que l'orientation est la composante la plus sensible au choix du filtre (EKF vs UKF) et aux paramètres de fusion IMU.

5.8.3 (3) Compromis orientation / position en UKF

La comparaison RUN5 $\rightarrow$ RUN6 met en évidence un compromis classique en fusion multi-capteurs : renforcer la contrainte sur l'orientation améliore fortement le yaw, mais peut légèrement dégrader la précision planimétrique (RMSE_{2D}), comme observé entre RUN5 et RUN6 (cf. tableau 7). Ces résultats suggèrent que, même sur un scénario simulé, la meilleure configuration dépend du critère prioritaire : stabilité angulaire (yaw) ou précision planimétrique.

5.8.4 (4) Incertitudes Q et P_0 : effet faible en non bruité, effet fort en bruité

L'étude paramétrique confirme que l'influence des incertitudes du filtre dépend fortement du niveau de perturbation des mesures. En conditions non bruitées , les RMSE sont extrêmement faibles et peu discriminantes (cf. tableau 4), ce qui est cohérent avec des capteurs simulés idéaux. En revanche, dès que l'odométrie est dégradée artificiellement, les réglages deviennent déterminants et produisent des écarts importants en RMSE (cf. tableau 5). Cela montre que l'analyse de Q et P_0 est pertinente surtout dans des scénarios suffisamment difficiles (bruit, dérive, séquences plus longues), plus proches des conditions réelles.

5.8.5 Conclusion de la synthèse

Au global, les expériences montrent que (i) la position est peu discriminante sur une séquence CARLA courte et cohérente, (ii) le yaw constitue un indicateur plus sensible pour comparer les modèles et les stratégies de fusion IMU, et (iii) le réglage des incertitudes Q et P_0 prend tout son sens dès que l'on introduit du bruit ou une dégradation réaliste. Ces constats fournissent une base solide pour interpréter les résultats obtenus dans le cadre simulé, tout en soulignant que des scénarios plus longs et plus perturbés seraient nécessaires pour caractériser finement la robustesse des configurations.

6 Conclusion

Ce travail a permis de mettre en place une chaîne expérimentale reproductible sous **ROS 2/CARLA** pour évaluer des configurations de localisation par fusion multi-capteurs à l'aide du package `robot_localization`. À partir d'un rosbag rejoué en temps simulé, une méthodologie de comparaison *vérité terrain vs estimation* a été définie et automatisée : appariement temporel, calcul des erreurs e_x, e_y, e_{2D}, e_ψ et production de métriques globales (RMSE) et de figures standardisées.

Les résultats montrent que, sur une séquence CARLA courte et "propre", l'odométrie demeure très cohérente avec la vérité terrain, ce qui rend la position peu discriminante entre configurations de capteurs. En revanche, l'orientation (yaw) s'est révélée être un indicateur plus sensible : l'intégration de l'IMU peut dégrader ou améliorer l'erreur angulaire selon les variables effectivement corrigées et le réglage des incertitudes. L'étude des paramètres internes du filtre a également confirmé un point important : en conditions idéales, l'influence de Q et P_0 reste limitée sur les RMSE, mais dès qu'une dégradation réaliste est introduite (odométrie bruitée), ces paramètres deviennent déterminants et permettent de différencier la robustesse des configurations.

Enfin, la comparaison EKF/UKF a mis en évidence l'intérêt pratique de tester plusieurs modèles de fusion dans le même cadre expérimental : l'UKF peut améliorer la stabilité angulaire lorsque l'IMU est exploitée de manière cohérente, au prix d'un réglage plus exigeant (covariances non nulles et cohérence temporelle des messages). Au-delà des performances chiffrées, cette étape a surtout permis de structurer une démarche d'analyse reliant les courbes d'erreurs au fonctionnement interne des filtres (prédiction/correction et pondération via Q, R, P_0).

Perspectives. Les suites naturelles consistent à (i) étendre l'évaluation à des scénarios plus longs et plus variés afin de mieux faire apparaître les phénomènes de dérive et de latence, (ii) poursuivre l'étude systématique de l'influence de Q, R et P_0 sur des données plus discriminantes, et (iii) exploiter les capteurs de perception présents dans le rosbag (caméras, LiDAR, radar) via une brique intermédiaire (VO/VIO/SLAM) produisant des mesures de pose avec covariances, afin d'enrichir la fusion au-delà du triplet odométrie–IMU–GNSS.

A Reproductibilité : scripts et configurations

Cette annexe décrit la chaîne minimale permettant de rejouer les expériences et de reproduire les figures et métriques présentées dans le rapport. Le code complet (scripts Python, fichiers YAML, commandes et résultats) est disponible sur le dépôt GitHub du projet.

A.1 Pipeline reproductible (rosbag → filtre → log → analyse)

La reproductibilité repose sur un pipeline identique pour chaque configuration (EKF/UKF) :

1. **Rejeu du rosbag** en temps simulé (`/clock`).
2. **Exécution du filtre** (`robot_localization` : EKF ou UKF) avec un fichier YAML.
3. **Enregistrement** des paires GT–estimation dans un CSV (logger).
4. **Analyse** automatique du CSV : métriques (RMSE) et figures standardisées.

Les flux comparés sont : `/carla/hero/odometry` (vérité terrain, GT) et `/odometry/filtered` (sortie filtrée).

A.2 Commandes de référence (exécution d'un run)

1) Rejouer le rosbag (temps simulé).

```
ros2 bag play ~/Téléchargements/carla_data/carla_data_0.db3 --clock
```

2) Lancer un filtre EKF (exemple).

```
ros2 run robot_localization ekf_node --ros-args \
  -r __node:=ekf_runX \
  -p use_sim_time:=true \
  --params-file </home/etudiant/ros2_ws/src/my_py_pkg/config/ukf>
```

3) Lancer un filtre UKF (exemple).

```
ros2 run robot_localization ukf_node --ros-args \
```

```
  -r __node:=ukf_runX \
  -p use_sim_time:=true \
  --params-file </home/etudiant/ros2_ws/src/my_py_pkg/config/ukf>
```

4) Enregistrer les paires GT–estimation (logger).

```
python3 compare_logger_arrival.py --ros-args \
  -p use_sim_time:=true \
  -p gt_topic:=/carla/hero/odometry \
  -p est_topic:=/odometry/filtered \
  -p tolerance_s:=0.12
```

5) Générer métriques et figures.

```
python3 analyze_csv.py </home/etudiant/ros2_ws/src/my_py_pkg/scripts/csv>
```

Remarque (synchronisation). L'appariement GT–estimation est réalisé par **plus proche voisin en temps de réception** (arrival-time) sous une tolérance τ. Cette méthode est robuste aux légers décalages entre topics, mais peut produire des **pics isolés** lorsque le jitter augmente.

A.3 Fichiers produits (sorties attendues)

Chaque run génère systématiquement :

- **CSV** : `run_compare_arrival_<timestamp>.csv` (paires GT–EST et erreurs élémentaires).
- **Métriques** : `<RUN>_metrics.txt` (RMSE$_x$, RMSE$_y$, RMSE$_{2D}$, RMSE$_{yaw}$).
- **Figures** (PNG) : `<RUN>_traj2d.png`, `<RUN>_errors_pos.png`, `<RUN>_error_yaw.png`, `<RUN>_hist_e2d.pn`

Le CSV contient notamment les champs : timestamps, pose GT, pose estimée et erreurs (e_x, e_y, e_{2D}, e_ψ). Les unités utilisées sont le mètre (position) et le radian (yaw).

A.4 Paramètres YAML essentiels à documenter

Les fichiers YAML (`robot_localization`) rendent les tests strictement reproductibles. Les paramètres clés sont :

- **Mode 2D** : `two_d_mode` (réduction au plan).
- **Repères** : `map_frame`, `odom_frame`, `base_link_frame`.
- **Sources capteurs** : `odom0`, `imu0` (et `odom1` si GNSS converti).
- **Variables corrigées** : vecteurs `odom0_config`, `imu0_config` (etc.).
- **Incertitudes** : `process_noise_covariance` (Q) et `initial_estimate_covariance` (P_0).

Ces éléments suffisent pour expliquer : (i) quelles composantes de l'état sont réellement estimées/corrigées, et (ii) comment le réglage des incertitudes influence la stabilité et les erreurs observées.

A.5 Cas UKF : covariances nulles et stabilisation (`covariance_fixer.py`)

Lors des tests UKF, les messages CARLA (`/carla/hero/imu` et `/carla/hero/odometry`) contiennent des **covariances nulles**. L'UKF étant plus sensible au conditionnement numérique que l'EKF, cela peut provoquer une **divergence** et l'apparition de valeurs **NaN** dans l'état estimé.

Pour stabiliser l'UKF, un nœud de prétraitement (`covariance_fixer.py`) republie : `/carla/hero/imu_cov` et `/carla/hero/odometry_cov` en imposant :

— des **covariances diagonales non nulles** (valeurs simples, ajustables) ;

— des **timestamps cohérents** avec le temps simulé (`/clock`).

Cette étape ne modifie pas la mesure en elle-même (valeurs), mais fournit au filtre une incertitude exploitable, condition nécessaire pour obtenir une estimation UKF stable dans notre pipeline.

Références

[1] Open Robotics, *ROS 2 Documentation (Jazzy)*. https://docs.ros.org/en/jazzy/.

[2] T. Moore et al., *robot_localization (code source)*. https://github.com/cra-ros-pkg/robot_localization.

[3] T. Moore et al., *robot_localization – documentation (API docs)*. https://docs.ros.org/en/noetic/api/robot_localization/html/index.html.

[4] ROS Wiki, *robot_localization (page de présentation)*. https://wiki.ros.org/robot_localization.

[5] CARLA Team, *CARLA Simulator Documentation*. https://carla.readthedocs.io/.

[6] R. E. Kalman, *A New Approach to Linear Filtering and Prediction Problems*, Transactions of the ASME–Journal of Basic Engineering, 1960.

[7] S. J. Julier and J. K. Uhlmann, *Unscented Filtering and Nonlinear Estimation*, Proceedings of the IEEE, 2004.

[8] Open Robotics, *sensor_msgs/msg/Imu (Jazzy)*. https://docs.ros.org/en/jazzy/p/sensor_msgs/msg/Imu.html.

[9] Open Robotics, *nav_msgs/msg/Odometry (Jazzy)*. https://docs.ros.org/en/jazzy/p/nav_msgs/msg/Odometry.html.

[10] S. Thrun, W. Burgard, D. Fox, *Probabilistic Robotics*, MIT Press, 2005.

[11] T. D. Barfoot, *State Estimation for Robotics*, Cambridge University Press, 2017.

[12] sachinkum0009, *carla-multi-sensor-fusion (GitHub)*. https://github.com/sachinkum0009/carla-multi-sensor-fusion.

[13] chetangadidesi, *EKF_Localization (GitHub)*. https://github.com/chetangadidesi/EKF_Localization.

[14] Autoware Foundation, *autoware_core – localization (GitHub)*. https://github.com/autowarefoundation/autoware_core/tree/main/localization.

[15] soumya997, *carla-e2e-av-stack (GitHub)*. https://github.com/soumya997/carla-e2e-av-stack.

[16] jasleon, *Vehicle-State-Estimation (GitHub)*. https://github.com/jasleon/Vehicle-State-Estimation.