Étude des méthodes d'intelligence artificielle pour la navigation des projectiles

Alicia Roux

Année 2023 N° d'ordre : (attribué par le SCD)

UNIVERSITÉ DE HAUTE-ALSACE
UNIVERSITÉ DE STRASBOURG

THÈSE

Pour l'obtention du grade de

DOCTEUR DE L'UNIVERSITÉ DE HAUTE-ALSACE

ÉCOLE DOCTORALE : Mathématiques, Sciences de l'Information et de l'Ingénieur (ED 269)

Discipline : Automatique

Présentée et soutenue publiquement

par

Alicia ROUX

Le 14 septembre 2023

Étude des méthodes d'intelligence artificielle pour la navigation des projectiles

---

Sous la direction du Prof. Jean-Philippe LAUFFENBURGER, du Dr. HDR. Sébastien CHANGEY et du Prof. Jonathan WEBER.

Jury :	Prof. Mohamed BOUTAYEB, Université de Lorraine	(Président)
	Prof. Dalil ICHALAL, Université d'Evry	(Rapporteur)
	Prof. Marion GILSON, Université de Lorraine	(Rapporteur)
	Dr. Hassen FOURATI, Université de Grenoble	(Examinateur)
	Prof. Jean-Philippe LAUFFENBURGER, Université de Haute-Alsace	(Directeur de thèse)
	Dr. HDR. Sébastien CHANGEY, Institut de recherches de Saint-Louis	(co-Directeur de thèse)
	Prof. Jonathan WEBER, Université de Haute-Alsace	(co-Directeur de thèse)

Remerciements

Je tiens tout d'abord à remercier chaleureusement mon directeur de thèse, Jean-Philippe LAUFFENBURGER, pour son encadrement, son soutien et ses nombreux conseils au cours de cette thèse ainsi que lors de mes études d'ingénieurs. Je souhaite également remercier mes deux codirecteurs de thèse, Jonathan WEBER pour son expertise en intelligence artificielle et Sébastien CHANGEY pour ses connaissances des projectiles.

Je remercie également les rapporteurs de cette thèse, Dalil ICHALAL et Marion GILSON. Je suis également reconnaissant à Mohamed BOUTAYEB et Hassen FOURATI pour avoir respectivement accepté d'être président et examinateur de ce jury de thèse.

Ce travail n'aurait pas été possible sans le soutien financier et matériel de l'Institut franco-allemand de recherches de Saint-Louis *(ISL)*, qui m'a accueilli au sein du groupe Guidage, Navigation, Contrôle *(GNC)* et sans le laboratoire IRIMAS de l'Université de Haute-Alsace *(UHA)*.

Je souhaite également remercier l'ensemble des collègues du groupe GNC et du groupe STC *(Sensors, Télémétrie et Communications)* de l'ISL et plus particulièrement Nabil JARDAK, Valentin RISS, Guillaume STRUB, Gaëtan CHEVRIN et Loïc BERNARD pour leurs conseils et leurs aides tout au long de cette thèse. De plus, je tiens à remercier très chaleureusement l'ensemble de l'équipe du MIAM, particulièrement Anthony CHASSIGNET et Michel BASSET pour les nombreuses discussions lors de mes visites qui m'ont permis de prendre du recul sur mon travail de thèse.

Enfin, un grand merci à l'ensemble de mon entourage et plus particulièrement à Guillaume pour son soutien. Je remercie également mes amis pour leurs encouragements et leur soutien au cours de ces trois années de thèse. Une mention particulière à Teddy pour ses relectures et corrections de ce long manuscrit.

TABLE DES MATIÈRES

INTRODUCTION

Ce travail de thèse a débuté en octobre 2020 entre l'IRIMAS *(Institut de Recherche en Informatique, Mathématiques, Automatique et Signal)* et l'ISL *(Institut Franco-Allemand de recherche de Saint-Louis)*. Ces travaux se concentrent sur la problématique de l'intelligence artificielle appliquée à la navigation des projectiles.

Contexte

La navigation des projectiles se base principalement sur les mesures d'une Unité de Mesure Inertielle *(IMU - Inertial Measurement Unit)* et les mesures d'un récepteur GNSS *(Global Navigation Satellite Systems)* embarquées. D'une part, l'intégration des mesures IMU, composée généralement d'accéléromètres, de gyromètres et parfois de magnétomètres, fournit une estimation de la trajectoire précise à court terme, mais dérive à long terme en raison de l'accumulation des erreurs des capteurs. D'autre part, le récepteur GNSS, grâce à des constellations de satellites, fournit des informations de positionnement absolu précises à long terme à une fréquence nettement inférieure à celle de l'IMU. En raison de leur complémentarité évidente, les mesures de l'IMU et du GNSS sont classiquement fusionnées par différents types de filtres de Kalman pour l'estimation des trajectoires. Néanmoins, les signaux GNSS ne sont pas toujours disponibles du fait de la configuration du terrain et sont également vulnérables vis-à-vis du brouillage et du leurrage [1]. C'est pourquoi, il est nécessaire de développer des solutions permettant de s'affranchir de ces mesures.

De récents travaux ont montré que l'intelligence artificielle (IA) permet de résoudre des problèmes complexes. En effet, les réseaux de neurones peuvent fournir des résultats équivalents ou meilleurs que des algorithmes classiques, notamment en limitant l'impact de mesures erronées ou d'erreurs de modélisation sur la précision des estimations. Néanmoins, l'IA est principalement utilisée pour résoudre des problèmes de navigation terrestre et est très peu utilisée pour la navigation appliquée à des projectiles ou des drones aériens,

1. Le leurrage vise à transmettre de faux signaux GNSS au récepteur pour le détourner de sa véritable position.

soumis à de fortes contraintes dynamiques. Ainsi, ces travaux se concentrent sur l'utilisation des méthodes d'intelligence artificielle pour développer des solutions de navigation sans GNSS afin de pallier les limitations des modèles et des capteurs embarqués.

Contributions de la thèse

Les problématiques traitées dans ce mémoire de thèse concernent l'estimation de la trajectoire d'un projectile en combinant des méthodes classiques de navigation et des réseaux de neurones. Différents algorithmes classiques de navigation, principalement basés sur un filtre de Kalman, sont ajustés dynamiquement par un réseau de neurones afin d'optimiser l'estimation de la trajectoire du projectile. L'intelligence artificielle (IA) est donc utilisée en complément de modèles mathématiques connus, afin notamment de corriger des erreurs de modélisation, de compléter des mesures manquantes ou erronées, ou de modéliser des dynamiques complexes. En d'autres termes, ces travaux visent à développer et à tester des solutions de navigation basées en partie sur l'IA afin d'évaluer l'apport des réseaux de neurones à l'optimisation des algorithmes classiques de navigation. Pour cela, un jeu de données de simulation de trajectoires de différents projectiles est utilisé. Seules les mesures de la centrale inertielle embarquée sont utilisées et aucune mesure GNSS n'est considérée dans ces travaux pour les raisons mentionnées précédemment.

Trois méthodes hybrides d'estimation de la trajectoire d'un projectile ont été mises en œuvre. La première approche, illustrée dans la figure 1, consiste à ajuster un paramètre de bruit d'un filtre de Kalman par un réseau de neurones afin de tenir compte des erreurs et des corrélations des mesures, délicates à modéliser. Cette solution est appliquée à plu-

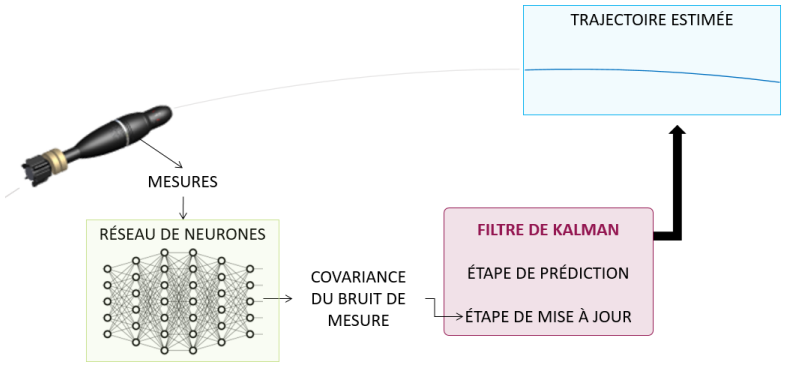

FIGURE 1 – Ajustement d'un paramètre de bruit d'un filtre de Kalman par un réseau de neurones.

sieurs filtres de Kalman, notamment à un filtre de Kalman Imparfait Invariant Étendu *(Imp. IEKF - Imperfect Invariant Extended Kalman Filter)*.
La deuxième méthode étudiée est l'estimation de la trajectoire d'un projectile à partir de réseaux de neurones récurrents, comme présentée en figure 2. Ce type de réseau de neurones est choisi pour ses facultés d'estimation de séries temporelles. Cette approche

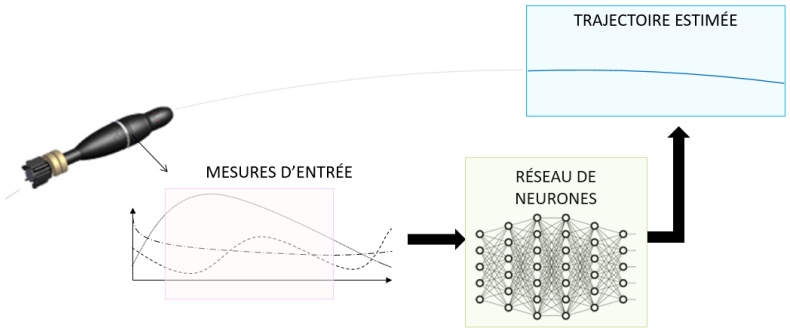

FIGURE 2 – Estimation de la trajectoire d'un projectile à partir de réseaux de neurones récurrents.

permet de modéliser des dynamiques complexes par des réseaux de neurones sans avoir recours à aucun modèle mathématique. Cette solution de navigation est appliquée à différentes trajectoires de projectiles.
La troisième méthode mise en œuvre, présentée dans la figure 3, vise à remplacer l'un des

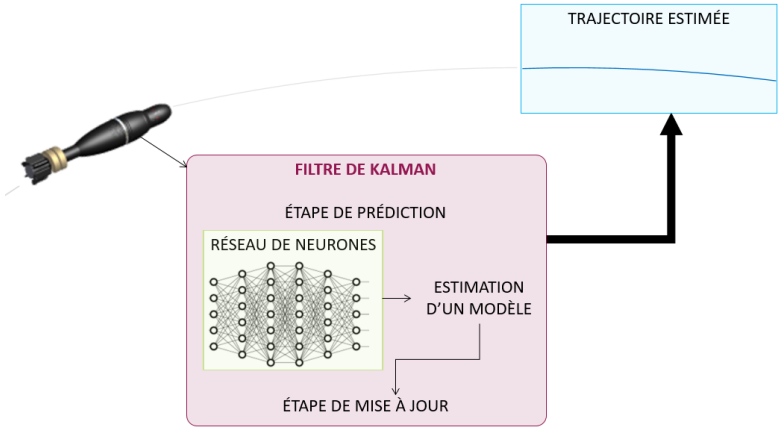

FIGURE 3 – Estimation de l'un des modèles d'un filtre de Kalman par un réseau de neurones.

modèles mathématiques d'un filtre de Kalman par un réseau de neurones. Cette solution permet principalement de limiter les erreurs liées à des approximations de modélisation.

L'ensemble de ces travaux est développé dans le présent document et illustré par des résultats évalués sur différentes simulations de trajectoires de projectiles.

Organisation du manuscrit

Afin de répondre aux objectifs cités précédemment, le manuscrit de thèse est structuré en cinq chapitres :

→ Le premier chapitre est dédié à l'état de l'art relatif aux projectiles, aux méthodes classiques de navigation par estimation d'états et aux approches basées sur l'IA. De plus, le jeu de données de trajectoires de projectiles, nécessaire au développement de ces travaux, est présenté.

→ Le second chapitre introduit les filtres de Kalman Invariant Étendus *(IEKF - Invariant Extended Kalman Filter)*. Il s'agit d'un filtre de Kalman non linéaire qui exploite les propriétés d'un système défini sur un groupe de Lie et associé à une erreur non linéaire. Pour cela, la théorie des IEKF ainsi qu'une étude comparative entre une variante de l'IEKF et un filtre de Kalman standard sont proposées.

→ Le troisième chapitre se concentre sur l'ajustement automatique d'un paramètre de bruit d'un filtre de Kalman par des réseaux de neurones. Un filtre de Kalman est sensible au réglage de la matrice de covariance du bruit de mesure, généralement constante et adaptée empiriquement par l'utilisateur. La solution proposée vise à ajuster dynamiquement cette matrice à l'aide d'un réseau de neurones convolutifs *(CNN - Convolutional Neural Network)*. Cela fournit ainsi une matrice de covariance variable dans le temps et adaptée aux différentes phases de vol du projectile. Pour cela, cette solution est appliquée à différents filtres de Kalman.

→ Le quatrième chapitre détaille l'estimation des trajectoires des projectiles par des réseaux de neurones récurrents. Cette solution vise à remplacer tous les modèles mathématiques par un réseau de neurones. Ainsi, à partir des mesures inertielles notamment, l'IA modélise les contraintes dynamiques du projectile, les erreurs des capteurs et les éventuelles perturbations afin d'estimer la trajectoire du projectile. Cette solution est appliquée à plusieurs projectiles caractérisés par des trajectoires différentes.

→ Le cinquième chapitre présente des filtres de Kalman où l'un des modèles mathématiques est remplacé par un réseau de neurones, également appelé *Deep Kalman Filter*. Cette solution est utilisée afin de suppléer des mesures manquantes dans un filtre de Kalman ou pour remplacer une étape de modélisation par un réseau de neurones. Pour cela, plusieurs cas sont étudiés, notamment l'adaptation d'un modèle linéaire à un modèle non linéaire par un réseau de neurones récurrents dans le cas où l'étape de modélisation standard est complexe.

Communications scientifiques

Cette thèse a conduit à la publication des communications scientifiques suivantes :

- **Revues internationales**

  Alicia Roux, Sébastien Changey, Jonathan Weber and Jean-Philippe Lauffenburger, "LSTM-Based Projectile Trajectory Estimation in a GNSS-Denied Environment", *Sensors 23, March 2023.*

- **Conférences internationales avec comité de lecture**

  Alicia Roux, Sébastien Changey, Jonathan Weber and Jean-Philippe Lauffenburger, "Mortar Trajectory Estimation by a Deep Error-State Kalman Filter in a GNSS-Denied Environment", *IEEE/ION Position Location and Navigation Symposium, ION PLANS 23, Monterey, California, April 2023.*

  Alicia Roux, Sébastien Changey, Jonathan Weber and Jean-Philippe Lauffenburger, "CNN-based Invariant Extended Kalman Filter for projectile trajectory estimation using IMU only", *International Conference on Control, Automation and Diagnosis, ICCAD 21, Grenoble, France, November 2021.*

  Alicia Roux, Sébastien Changey, Jonathan Weber and Jean-Philippe Lauffenburger, "Projectile trajectory estimation : performance analysis of an Extended Kalman Filter and an Imperfect Invariant Extended Kalman Filter", *9th International Conference on Systems and Control, ICSC 21, Caen, France, November 2021.*

- **Conférences nationales avec comité de lecture**

  Alicia Roux, Sébastien Changey, Jonathan Weber and Jean-Philippe Lauffenburger,"Estimation de la trajectoire d'un projectile ajustée dynamiquement par un réseau de neurones", *XXVIIIème Colloque Francophone de Traitement du Signal et des Images, GRETSI'22, Nancy, September 2022.*

  Alicia Roux, Sébastien Changey, Jonathan Weber and Jean-Philippe Lauffenburger,"Projectile trajectory estimation : an LSTM approach", *Conference on Artificial Intelligence for Defense, CAID, Rennes, France, November 2022.*

Chapitre 1

État de l'art

Sommaire

1.1 Introduction

Ce chapitre introduit les concepts fondamentaux nécessaires à la compréhension de ces travaux. Ainsi, une première partie *(1.2)* se focalise sur les projectiles et présente les capteurs embarqués, leurs utilisations et leurs limitations. Une seconde partie *(1.3)* est dédiée aux méthodes classiques de navigation des projectiles proposées dans la littérature et principalement basées sur des filtres de Kalman. Une troisième partie *(1.4)* se concentre sur l'utilisation de l'intelligence artificielle dans le domaine militaire et des solutions d'IA pour la navigation. Enfin, la dernière partie *(1.5)* présente le jeu de données de simulations de tirs de projectiles utilisé dans le cadre de cette thèse.

1.2 Capteurs embarqués dans les projectiles

Cette partie présente les différents capteurs embarqués dans un projectile ainsi que leurs limitations.

Un projectile est un corps lancé ou projeté par une arme pour atteindre une cible. Suivant l'application souhaitée et la distance de la cible, différents types de projectiles peuvent être envisagés. Toutefois, tous les projectiles sont constitués d'éléments semblables comme présenté dans la figure 1.1.

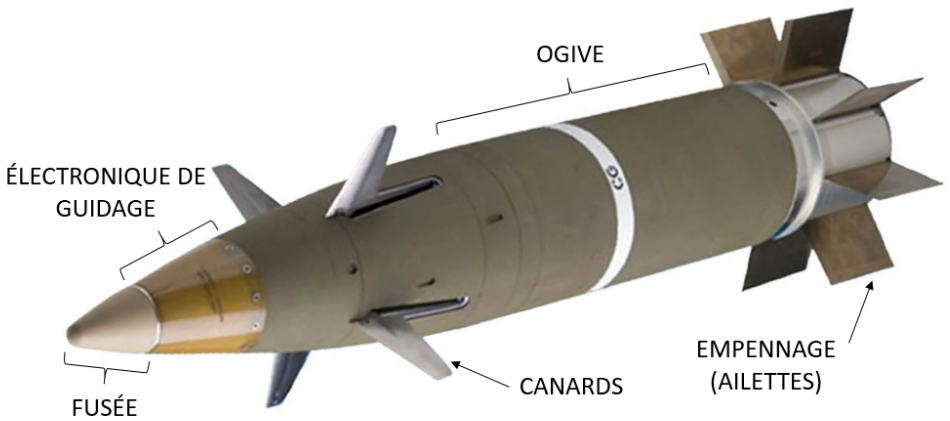

FIGURE 1.1 – Les différents éléments composant un projectile : illustration avec l'obus de 155 mm M982 Excalibur.

Comme illustré dans la figure 1.1, un projectile est constitué de différentes parties :

- **L'empennage**, disposé à la partie postérieure du projectile, peut être composé de plusieurs ailettes ajustables qui se déploient à l'issue de la phase de lancement, afin de stabiliser le projectile au cours de sa trajectoire.
- **L'ogive** constitue la charge destructive du projectile qui peut être de différents types (ogives explosives, à fragmentation, à charge creuse, éclairante, fumigène, chimique, biologique...).
- **Les canards** servent à piloter et à guider le projectile en corrigeant les écarts longitudinaux et latéraux.
- **L'unité électronique de guidage**, composée de capteurs et de batteries, permet le guidage et la localisation du projectile.
- **La fusée** déclenche la fonction explosive du projectile et peut également être composée de capteurs additionnels ou d'antennes.

Un projectile est soumis à de fortes contraintes dynamiques telles que des chocs importants à l'accélération [1] ou des vitesses de rotation élevées [2]. De plus, un projectile doit satisfaire des exigences de coût et d'espace, comme illustré dans la figure 1.2. De ce fait, l'unité électronique de guidage d'un projectile est généralement équipée de deux types de capteurs nécessaires à sa navigation et à son guidage ; une centrale inertielle *(IMU - Inertial Measurement Unit)* et un récepteur GNSS *(Global Navigation Satellite Systems)*.

FIGURE 1.2 – Caractéristiques dimensionnelles des projectiles de différents calibres [1].

1. Accélération de 8 000 g pour un mortier de 120 mm, 40 000 g pour un projectile de 40 mm, 25 000 g pour un obus de 155 mm.
2. Vitesse de rotation de 250 à 300 Hz pour un obus de 155 mm.

1.2.1 Centrale inertielle

Les centrales inertielles *(Inertial Measurement Unit - IMU)* embarquées dans les projectiles sont composées de trois gyromètres et de trois accéléromètres. Lorsque des magnétomètres y sont ajoutés, il s'agit alors d'une centrale magnéto-inertielle. Afin de satisfaire les contraintes de coût, d'espace, de poids et de faible consommation énergétique, les IMU embarquées dans les projectiles sont des capteurs miniaturisés basés sur la technologie microélectromécanique *MEMS (Microelectromechanical systems)* [2], [3].

L'accéléromètre

L'accéléromètre mesure la force spécifique du corps en mouvement qui est l'accélération résultante de toutes les forces à l'exception de la gravité [4]. Ces mesures informent sur les déplacements, les vitesses, les vibrations ou les chocs subis par le corps en mouvement.

Comme présenté dans la figure 1.3, les accéléromètres sont largement utilisés pour l'estimation des trajectoires et de l'attitude des projectiles du fait de leurs résistances aux phases de lancement [5], [6].

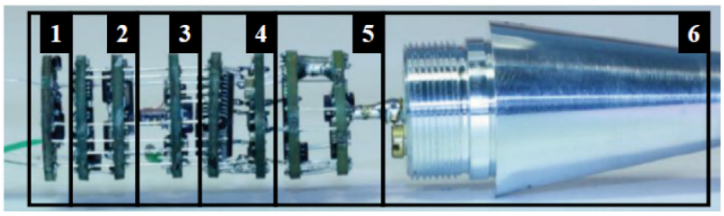

FIGURE 1.3 – Instrumentation d'un projectile de 30 mm : (1) bloc d'alimentation, (2) magnétomètre 3 axes, (3) accéléromètre 3 axes, (4) CPU, (5) émetteur radiofréquence (6) fusée comprenant l'antenne [6].

Le gyromètre

Le gyromètre mesure la vitesse angulaire d'un corps en mouvement par rapport à un référentiel fixe (le gyroscope mesure la position angulaire). Ces mesures informent sur les rotations subies par le corps en mouvement.

Les projectiles à forte rotation tels que les projectiles de 40 mm peuvent atteindre des vitesses de rotation de 1 kHz. Dans ce type d'application, beaucoup des gyromètres disponibles sur le marché ne peuvent satisfaire ces exigences techniques et saturent. Ainsi, sous

des contraintes de coûts raisonnables, l'utilisation des gyromètres est limitée, notamment pour les projectiles à forte rotation [7], [8].

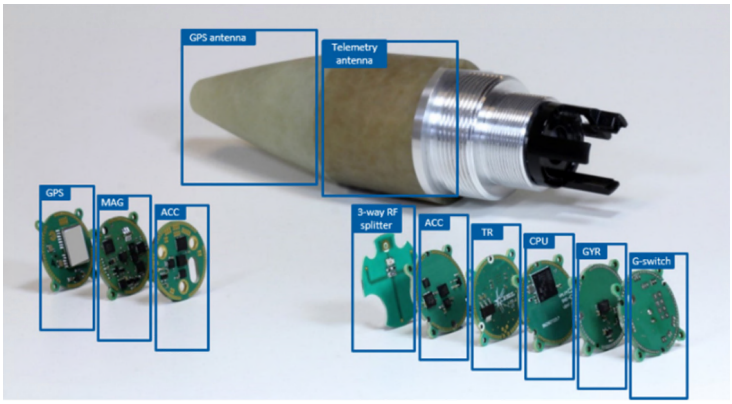

FIGURE 1.4 – Projectile équipé de magnétomètres, d'accéléromètres et de gyromètres MEMS, d'un récepteur GPS, d'un émetteur-récepteur de télémétrie et d'un G-switch utilisé pour allumer les composants électroniques après le lancement [8].

Le magnétomètre

Le magnétomètre mesure les projections du champ magnétique terrestre le long du corps en mouvement. Ces mesures informent sur l'orientation du corps en connaissant la direction du champ magnétique terrestre.

Les magnétomètres sont des capteurs à faible coût, avec des plages de fonctionnement importantes, et capables de résister à des vitesses de rotation importantes. Ainsi, ces capteurs équipent de plus en plus de projectiles à forte vitesse de rotation afin d'en estimer l'orientation comme présenté dans la figure 1.5. Ils permettent notamment de pallier les limitations des gyromètres [7]-[9].

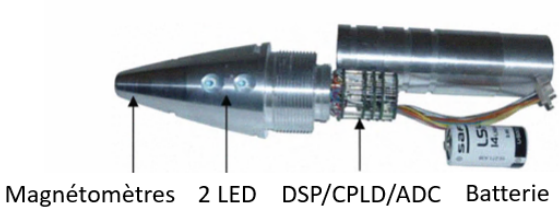

FIGURE 1.5 – Pointe de projectile, testée en vol, composée de deux magnétomètres radiaux pour estimer l'angle de roulis du projectile [9].

Limitations de l'IMU

Bien que très souvent utilisée pour des problèmes de navigation du fait de sa taille, de son coût et de sa faible consommation énergétique, la précision de la navigation par IMU se dégrade à long terme. En effet, la navigation par IMU vise à intégrer les mesures inertielles et donc les erreurs des capteurs, entraînant ainsi une dérive de cette solution de navigation à long terme. Les erreurs de l'IMU sont principalement dues aux imperfections des capteurs telles que des erreurs mécaniques ou électroniques (désalignement, biais, bruit, sensibilité, facteurs d'échelle, dérive linéaire ...) [10], [11].

1.2.2 Récepteur GNSS

Le système mondial de navigation par satellites *(GNSS - Global Navigation Satellite System)* permet de déterminer le positionnement précis (PVT - Position, Vitesse, Temps) d'un récepteur sur Terre via une constellation de satellites artificiels telle que le GPS [3], Galileo, GLONASS, COMPASS, IRNSS/NavIC, QZSS [12].

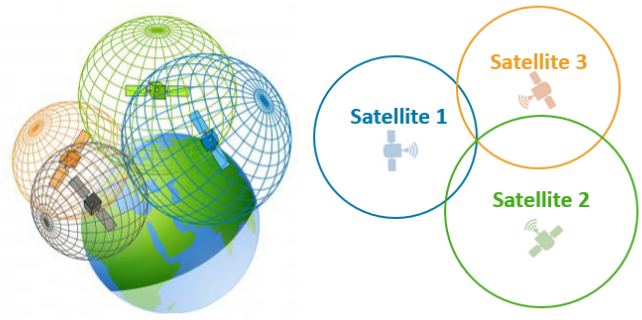

FIGURE 1.6 – Principe de fonctionnement du positionnement par GNSS.

Les satellites orbitent à une distance d'environ 25 000 km au-dessus de la Terre et transmettent en continu des signaux comprenant la position du satellite ainsi que l'heure exacte et la date d'émission du signal [4]. Le récepteur GNSS réceptionne les messages émis par au moins 4 satellites afin de déterminer sa localisation par la méthode de triangulation présentée dans la figure 1.6. Les messages de trois satellites déterminent la localisation du

3. GPS - *Global Positioning System*, GLONASS - *Global'naya Navigatsionnaya Sputnikowaya Sistema*, IRNSS - *Indian Regional Navigation Satellite System*, NavIC - *Navigation Indian Constellation*, QZSS - *Quasi-Zenith Satellite System*.

4. Les récepteurs GNSS fournissent des informations de position précises à long terme à une fréquence d'environ 10 Hz, nettement inférieure à celle de l'IMU (1kHz).

récepteur (latitude, longitude et altitude) et le message du dernier permet de synchroniser les satellites et le récepteur. Par ailleurs, le récepteur détermine aussi sa vitesse ainsi que la date et l'heure.

Limitations du GNSS

Malgré la précision du positionnement par GNSS, ces signaux sont sensibles aux menaces extérieures et à la configuration du terrain (interférence, disponibilité et fiabilité des signaux).

Disponibilité des signaux GNSS : Les signaux GNSS sont parfois indisponibles du fait de la configuration du terrain (terrains urbains, vallées profondes, tunnels...) ou du fait de la faible puissance des signaux (dissipation de l'énergie). En effet, des bruits élevés, la réflexion des signaux ou l'orientation de l'antenne peuvent dégrader les signaux GNSS et les rendre indisponibles [13], [14].

Brouillage : La seconde limitation du GNSS est sa vulnérabilité face au brouillage comme illustré dans la figure 1.7. En effet, les signaux transmis par les satellites sont de faible puissance et le brouillage vise à émettre un signal de plus forte amplitude sur les fréquences dédiées au GNSS afin que le récepteur GNSS ne décode pas le vrai signal satellite [14]-[16].

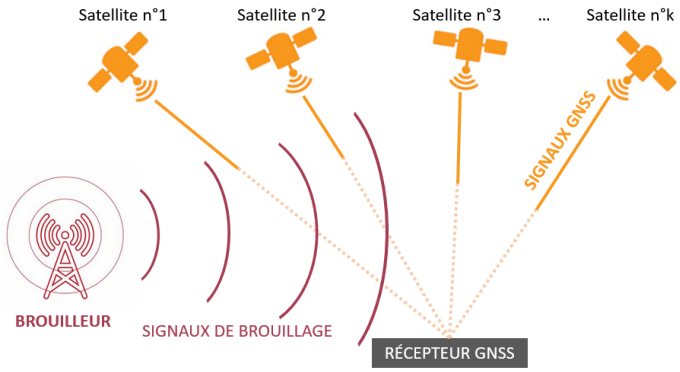

FIGURE 1.7 – Principe de fonctionnement du brouillage d'un récepteur GNSS.

Leurrage : La troisième limitation du GNSS est sa vulnérabilité face au leurrage. Comme présenté dans la figure 1.8, de faux signaux GNSS sont transmis aux récepteurs pour détourner le récepteur de sa véritable position [14], [16], [17]. Le leurrage est plus difficile à détecter que le brouillage.

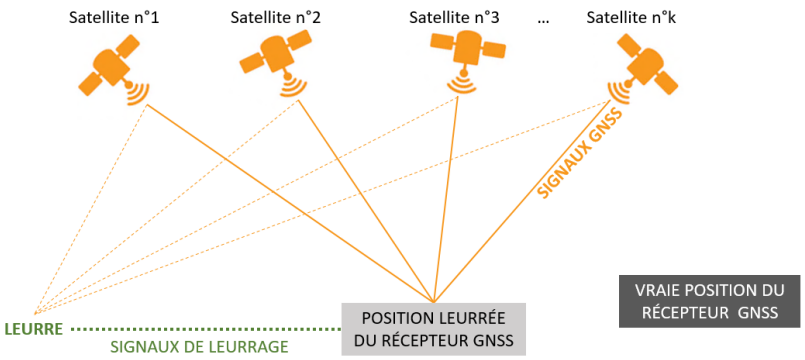

FIGURE 1.8 – Principe de fonctionnement du leurrage d'un récepteur GNSS.

1.2.3 Électronique embarquée dans un projectile

En plus des capteurs inertiels et GNSS embarqués, différents modules électroniques sont utilisés afin de communiquer et de traiter les données du projectile [9], [18] comme illustré sur la figure 1.9 :

- un convertisseur analogique-numérique *(Analogue-to-Digital Converter - ADC)* pour coder numériquement les grandeurs mesurées des capteurs.
- un processeur de signal numérique *(Digital Signal Processor - DSP)* pour traiter les données et communiquer avec les capteurs.
- un microcontrôleur pour formater les données.
- un émetteur/amplificateur afin de transmettre les données via l'antenne embarquée, généralement dans la fusée du projectile.

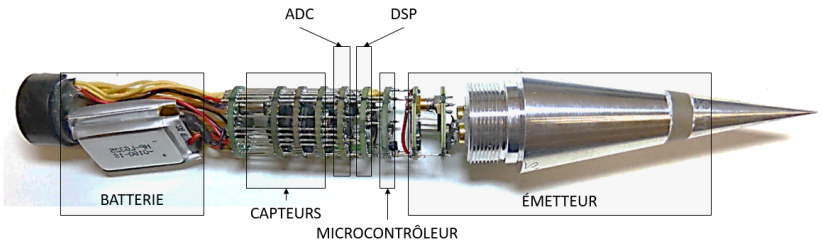

FIGURE 1.9 – Instrumentation d'un projectile de 40 mm [7].

1.2.4 Autres types de capteurs pour la navigation des projectiles

D'autres types de capteurs visent à être embarqués dans les projectiles tels que les caméras, bien que l'intégration de ces capteurs dans des projectiles soit délicate du fait de la sensibilité des optiques aux fortes accélérations.

Toutefois, la vision peut être utilisée à des fins de navigation. En effet, les caméras, couplées à une IMU, permettent de localiser le projectile dans l'espace en utilisant des algorithmes d'odométrie visuelle sans avoir recours à des récepteurs GNSS. De plus, la trajectoire d'un corps en mouvement peut également être déterminée en exploitant les propriétés des ondes lumineuses telles que la polarisation [19].

Par ailleurs, la vision peut être utilisée à des fins d'observation, telle que le projectile d'observation présenté dans la figure 1.10 qui vise à examiner une scène d'un tir à plusieurs kilomètres afin de vérifier que la cible soit atteinte.

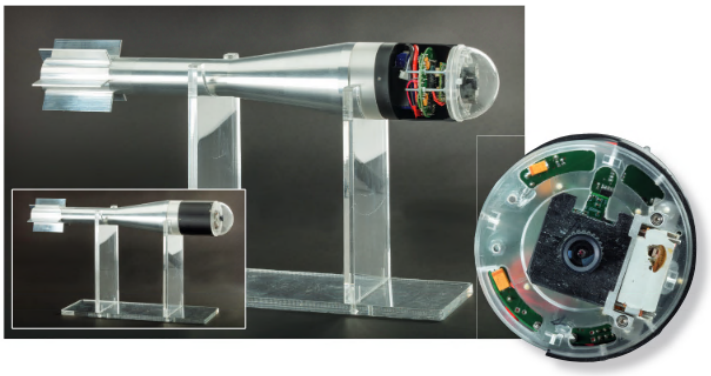

FIGURE 1.10 – Projectile d'observation développé par l'ISL.

1.3 Méthodes classiques de navigation des projectiles

La fusion des mesures de l'IMU et du récepteur GNSS embarqué via des filtres de Kalman permet d'estimer la trajectoire d'un projectile. L'intégration des mesures inertielles produit une estimation précise de la trajectoire du projectile à court terme mais dévie à long terme en raison de la dérive des capteurs. À l'inverse, les signaux GNSS fournissent des informations de localisation précises à long terme à une fréquence nettement inférieure à celle de l'IMU [5], mais ces signaux sont parfois indisponibles, leurrés ou brouillés. Cette partie détaille les méthodes de navigation inertielle ainsi que le principe de fonctionnement d'un filtre de Kalman dans le cas linéaire et non linéaire. En outre, différents filtres de Kalman implémentés pour la navigation des projectiles sont présentés.

1.3.1 Navigation inertielle

La navigation inertielle (*Dead Reckoning*), dont le principe de fonctionnement est illustré dans la figure 1.11, permet d'estimer la trajectoire d'un corps en mouvement en intégrant les mesures des accéléromètres et des gyromètres embarqués.

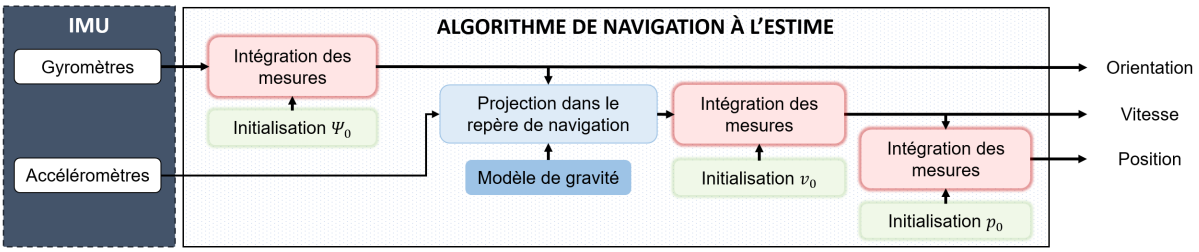

FIGURE 1.11 – Algorithme de navigation à l'estime (*Dead Reckoning*) [4].

Un algorithme de navigation à l'estime débute par l'initialisation de la position p_0, de la vitesse v_0 et de l'orientation Ψ_0 du corps en mouvement. L'orientation Ψ_k à instant discret k est déterminée par l'intégration des mesures des trois gyromètres ω_k et par l'orientation à l'instant précédent Ψ_{k-1}. La vitesse v_k est obtenue par l'intégration des trois accéléromètres a_k, projetée dans le repère de navigation grâce à l'orientation Ψ_k et à la vitesse à l'instant précédent v_{k-1}. La position p_k est déterminée par intégration de la vitesse v_k et de la position à l'instant précédent p_{k-1}.

5. Fréquence d'échantillonnage d'une IMU standard : $1kHz$. Fréquence d'échantillonnage d'un récepteur GNSS : de 1 à $10Hz$

La navigation inertielle permet de calculer la trajectoire d'un corps en mouvement à chaque instant en fonction de la trajectoire estimée à l'instant précédent. Ainsi, la précision de cet algorithme diverge à long terme car les erreurs des accéléromètres et des gyromètres sont intégrées et ajoutées aux prédictions successives [4].

1.3.2 Le filtre de Kalman Linéaire

Le filtre de Kalman et ses variantes constituent la base de beaucoup d'algorithmes de navigation. Un filtre de Kalman est une méthode d'estimation des états d'un système évoluant dans le temps à partir de mesures appelées observations [4], [20]-[23].

En d'autres termes, le filtre de Kalman linéaire estime les états x_k d'un système linéaire dont l'évolution temporelle est modélisée par une *équation d'état*, puis corrige ces estimations par des mesures y_k décrites par un *modèle d'observation* :

$$\textit{Équation d'état} \quad x_{k+1} = A_k x_k + B_k u_k + w_k \tag{1.1}$$

$$\textit{Modèle d'observation} \quad y_k = H_k x_k + v_k \tag{1.2}$$

avec

- $x_{k+1} \in \mathbb{R}^{n\times 1}$ les états décrivant le système à l'instant $k+1$,
- $A_k \in \mathbb{R}^{n\times n}$ la matrice d'évolution qui relie l'état au temps k à l'état au temps $k+1$,
- $u_k \in \mathbb{R}^{l\times 1}$ l'entrée de commande connue,
- $B_k \in \mathbb{R}^{n\times l}$ la matrice qui relie l'entrée de commande u_k à l'état x_k,
- $w_k \in \mathbb{R}^{n\times 1} \sim \mathcal{N}(0, Q_k)$ le bruit de modèle supposé blanc centré gaussien où Q_k est la matrice de covariance semi-définie positive, qui modélise l'incertitude sur le modèle,
- $y_k \in \mathbb{R}^{m\times 1}$ les observations mesurées,
- $H_k \in \mathbb{R}^{m\times n}$ la matrice d'observation qui relie les observations y_k aux états x_k,
- $v_k \in \mathbb{R}^{m\times 1} \sim \mathcal{N}(0, R_k)$ le bruit de mesure supposé blanc centré gaussien où R_k est la matrice de covariance définie positive, qui modélise l'incertitude sur les mesures. Les bruits w_k et v_k sont supposés indépendants, c'est-à-dire $\mathbb{E}[w_k v_{k+\tau}^T] = 0$.

Comme illustré dans la figure 1.12, le filtre de Kalman linéaire débute par une étape d'initialisation, puis alterne entre une étape de prédiction et une étape de mise à jour afin d'estimer l'état $\widehat{x}_k$ et la matrice de covariance d'erreur P_k qui représente les incertitudes dans les estimations.

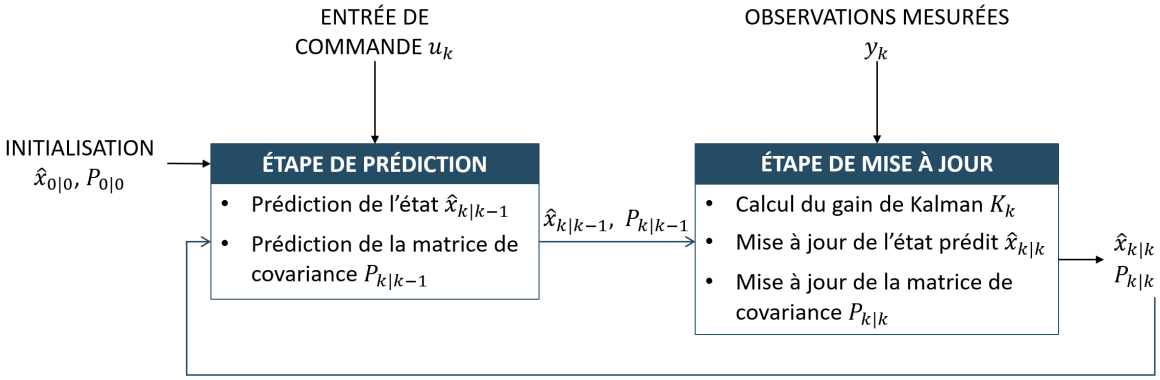

FIGURE 1.12 – Principe de fonctionnement d'un filtre de Kalman linéaire [4].

Étape d'initialisation : À l'instant initial $k = 0$ et en l'absence d'information, l'état et la covariance de l'erreur sont initialisés, tels que :

$$\widehat{x}_{0|0} = \mathbb{E}[x_0], \qquad P_{0|0} = \mathbb{E}[(x_0 - \widehat{x}_{0|0})(x_0 - \widehat{x}_{0|0})^T] \tag{1.3}$$

avec $\widehat{x}_{0|0}$ l'état estimé, $P_{0|0}$ la covariance estimée et x_0 l'état vrai à l'instant initial.

Étape de prédiction : L'étape de prédiction prédit l'état $\widehat{x}_{k|k-1}$ à l'instant k à partir de l'état estimé à l'instant précédent $\widehat{x}_{k-1|k-1}$ et de l'équation d'état dans le cas où il n'y a pas de perturbation :

$$\widehat{x}_{k|k-1} = A_k \widehat{x}_{k-1|k-1} + B_k u_k \tag{1.4}$$

La covariance prédite $P_{k|k-1}$ à l'instant k est déterminée à partir de l'erreur d'estimation $e_{k|k-1} = x_k - \widehat{x}_{k|k-1}$ tel que :

$$P_{k|k-1} = \mathbb{E}[(x_k - \widehat{x}_{k|k-1})(x_k - \widehat{x}_{k|k-1})^T] = A_k P_{k-1|k-1} A_k^T + Q_k \tag{1.5}$$

Étape de mise à jour : La prédiction de la covariance de l'erreur $P_{k|k-1}$, la matrice d'observation H et la matrice de covariance du bruit de mesure R_k permettent de déterminer le gain de Kalman K_k. Le gain K_k est calculé en fonction de la confiance accordée au modèle Q_k et aux mesures R_k tel que :

$$K_k = P_{k|k-1} H_k (H_k P_{k|k-1} H_k^T + R_k)^{-1} \tag{1.6}$$

Le gain de Kalman pondère l'innovation définie comme la différence entre les observations y_k et la prédiction de ces observations $H\widehat{x}_{k|k-1}$, afin de mettre à jour l'état estimé tel que :

$$\widehat{x}_{k|k} = \widehat{x}_{k|k-1} + K_k(y_k - H_k\widehat{x}_{k|k-1}) \tag{1.7}$$

La covariance de l'erreur d'estimation $P_{k|k}$ est mise à jour à partir du gain de Kalman K_k, de la covariance prédite $P_{k|k-1}$ et de la matrice d'observation H tel que :

$$P_{k|k} = (I_n - K_k H_k)P_{k|k-1} \tag{1.8}$$

Le filtre de Kalman linéaire est un estimateur optimal dans le cas où le système dynamique (1.1) - (1.2) est linéaire et que les incertitudes sont modélisées par des variables aléatoires gaussiennes. Sous ces conditions, le filtre de Kalman linéaire minimise l'erreur quadratique moyenne entre les états estimés $\widehat{x}$ et les états vrais x [23], [24]. De plus, sous des conditions classiques d'observabilité et de contrôlabilité du système dynamique (1.1) - (1.2), la stabilité asymptotique du filtre de Kalman associé est garantie [21].

1.3.3 Le filtre de Kalman Étendu

Le filtre de Kalman Étendu (EKF) est une extension du filtre de Kalman linéaire appliquée à un système non linéaire [20], [21], [23], [24]. L'EKF est donc une méthode d'estimation qui vise à linéariser un système autour de la trajectoire estimée et à construire un filtre de Kalman pour le modèle linéarisé.

Comme pour le cas linéaire, un EKF estime les états d'un système dynamique non linéaire et les corrige par des observations modélisées de la façon suivante :

$$\textit{Équation d'état} \quad x_{k+1} = f(x_k, u_k, w_k) \tag{1.9}$$

$$\textit{Modèle d'observation} \quad y_k = h(x_k, v_k) \tag{1.10}$$

avec $x_{k+1} \in \mathbb{R}^{n\times 1}$ les états du système, $y_k \in \mathbb{R}^{m\times 1}$ les observations mesurées, $f(.)$ et $h(.)$ des applications non linéaires, $w_k \sim \mathcal{N}(0, Q_k)$ et $v_k \sim \mathcal{N}(0, R_k)$ les bruits de modèle et de mesure supposés gaussiens et indépendants.

L'EKF produit une approximation de l'estimation optimale x_k en linéarisant le système dynamique non linéaire (1.9) - (1.10) autour de la dernière estimation d'état $\widehat{x}_{k-1|k-1}$ [23], [24]. Pour cela, les modèles non linéaires d'évolution $f(.)$ et d'observation $h(.)$ sont

linéarisés localement au premier ordre par des matrices jacobiennes évaluées telles que :

$$\textit{Matrice d'évolution} \quad A_k = \frac{\partial f}{\partial x}\Big|_{\widehat{x}_{k-1|k-1},u_k} \tag{1.11}$$

$$\textit{Matrice d'observation} \quad H_k = \frac{\partial h}{\partial x}\Big|_{\widehat{x}_{k|k-1}} \tag{1.12}$$

Comme dans le cas linéaire, l'EKF est tout d'abord initialisé d'après l'équation (1.3) puis alterne entre une étape de prédiction et une étape de mise à jour.

Étape de prédiction : L'état $\widehat{x}_{k|k-1}$ est prédit à partir du modèle d'évolution (1.9) dans le cas où le bruit de modèle est nul et la covariance $P_{k|k-1}$ est prédite à partir du modèle d'évolution linéarisé (1.11), de sorte que :

$$\widehat{x}_{k|k-1} = f(\widehat{x}_{k-1|k-1}, u_k, 0) \tag{1.13}$$

$$P_{k|k-1} = A_k P_{k-1|k-1} A_k^T + Q_k \tag{1.14}$$

Étape de mise à jour : L'état $\widehat{x}_{k|k-1}$ et la covariance $P_{k|k-1}$ prédite sont mis à jour à partir des observations mesurées y_k, du modèle d'observation non linéaire $h(.)$ (1.10) et de la matrice d'observation H_k (1.12) :

$$K_k = P_{k|k-1} H_k (H_k P_{k|k-1} H_k^T + R_k)^{-1} \tag{1.15}$$

$$\widehat{x}_{k|k} = \widehat{x}_{k|k-1} + K_k(y_k - h(\widehat{x}_{k|k-1}, 0)) \tag{1.16}$$

$$P_{k|k} = (I_n - K_k H_k) P_{k|k-1} \tag{1.17}$$

Ainsi, l'EKF est un estimateur d'état adapté à des systèmes non linéaires. La linéarisation du modèle d'évolution et/ou d'observation permet d'appliquer la méthodologie d'un filtre de Kalman linéaire à partir de ces modèles linéarisés. Néanmoins, la linéarisation des modèles influe sur plusieurs propriétés vérifiées par du filtre de Kalman linéaire :

- la convergence de l'EKF n'est pas garantie. En effet, les erreurs d'estimation peuvent être amplifiées par le gain de Kalman, calculé sous l'hypothèse que l'erreur d'estimation est petite afin que la linéarisation du premier ordre reste valide. La convergence de l'EKF nécessite de vérifier certaines propriétés restrictives (matrice de covariance de l'erreur bornée, observabilité, ...) pas toujours valables en pratique [4], [21], [25], [26]. Ainsi, comparé au filtre de Kalman linéaire, l'EKF n'est pas un filtre optimal à cause de la linéarisation du système dynamique (1.9) - (1.10).

- la linéarisation peut créer de fausses observabilités et donc un EKF peut être incohérent avec le système non linéaire. En d'autres termes, un état qui est non observable peut devenir observable avec la linéarisation des modèles [27]-[31].

1.3.4 Applications des filtres de Kalman pour la navigation des projectiles

Couplage IMU/GNSS

En raison de la complémentarité évidente de l'IMU et du récepteur GNSS (précision à court/long terme, fréquence élevée/faible), ces mesures sont classiquement fusionnées par des filtres de Kalman pour estimer la trajectoire d'un projectile, comme illustré dans la figure 1.13 [6]. La trajectoire estimée à partir des mesures inertielles est corrigée par les mesures du récepteur GNSS afin de limiter la dérive de l'IMU et de produire une solution de navigation en l'absence des mesures GNSS.

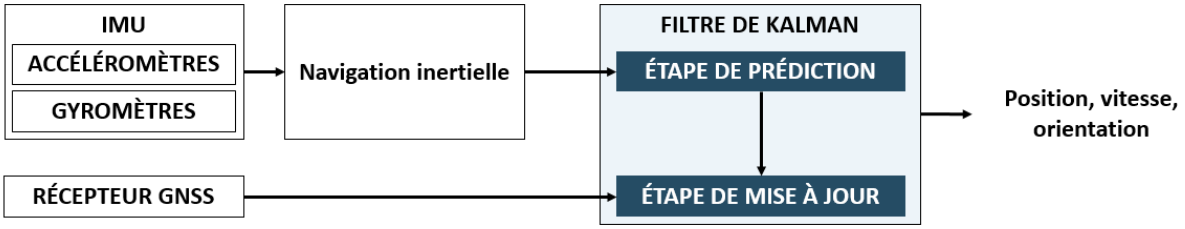

FIGURE 1.13 – Couplage des mesures IMU et GNSS dans un filtre de Kalman.

Cette solution est largement employée notamment pour la navigation des projectiles guidés tels que l'*Extended Range Guided Munition* (ERGM) et *Excalibur* [32]. Différents types de filtre de Kalman peuvent être utilisés pour le couplage [33]-[35].

Estimation de l'attitude d'un projectile

L'un des défis de la navigation des projectiles est d'estimer avec précision l'attitude. Pour cela, différentes approches sont employées suivant le type d'application visée.

Utilisations du GNSS : L'orientation des projectiles peut être estimée à partir des mesures GNSS comme dans [36], qui évalue l'angle de tangage et de lacet d'un projectile en

6. Il existe plusieurs architectures d'intégration pour coupler les mesures IMU et GNSS : le couplage lâche, le couplage serré et le couplage ultraserré comme détaillé dans [4].

utilisant exclusivement les vitesses du projectile mesurées par le récepteur GPS embarqué et sans considérer aucun autre capteur. Parfois, les mesures GNSS sont couplées à d'autres capteurs tels que [37] qui valide en vol une méthode d'estimation de la vitesse de roulis d'un projectile de 122 mm à partir de deux magnétomètres et d'un récepteur GNSS, ou [38] qui combine les mesures d'un magnétomètre triaxial, de deux gyromètres et d'un récepteur GPS via un filtre de Kalman pour estimer l'orientation d'un projectile.

Solutions basées sur les magnétomètres : L'orientation d'un projectile peut être déterminée sans mesure GNSS. Comme mentionné dans la partie *(1.2.1)*, les magnétomètres sont peu coûteux, capables de résister aux phases de lancement et fonctionnels sur de larges plages de variations. C'est pourquoi, ces capteurs sont de plus en plus employés pour l'estimation de l'orientation des projectiles comme résumé dans le tableau 1.1.

Article	Estimation	Méthode	Capteurs	Validation
[9]	Angle de roulis	Filtre de Kalman Étendu	Deux magnétomètres radiaux	Testé en vol sur un projectile supersonique à ailettes de 30 mm tournant à 35 Hz.
[39]	Angle de roulis	Filtre de Kalman Étendu Adaptatif	Deux magnétomètres radiaux	Testé en simulation sur un projectile tournant à 8 Hz et testé en vol sur un projectile de 120 mm tournant à 0.39 Hz.
[40]	Angle de tangage et de lacet	*Unscented Kalman Filter*	Deux magnétomètres radiaux	Validation en simulation et sur des données expérimentales.
[41]	Angle de roulis, tangage, lacet	Filtre de Kalman mixte	Trois magnétomètres	Testé en simulation sur un projectile tournant de 155 mm.
[42]	Angle de roulis, tangage, lacet	Filtre de Kalman Étendu	Trois magnétomètres	Testé en simulation sur un obus d'artillerie stabilisé en rotation.

TABLE 1.1 – Exemples d'utilisation des magnétomètres pour l'estimation de l'orientation d'un projectile.

Problématique des gyromètres : Comme mentionné dans la partie *(1.2.1)*, dans des limites de coût raisonnables, beaucoup de gyromètres saturent en raison des vitesses de rotation élevées et ne résistent pas aux phases de lancement des projectiles (chocs de 20 000 g à l'accélération). C'est pourquoi, plusieurs auteurs proposent des solutions d'estimation de l'orientation d'un projectile en excluant ce type de capteur.

L'article [43] présente une méthode, illustrée dans la figure 1.14, d'estimation de l'attitude basée sur un magnétomètre à trois axes, un accéléromètre à deux axes et des filtres de Kalman en cascade afin de discriminer les différentes dynamiques. Cette solution, testée en vol libre, montre que des capteurs à faible coût sont suffisants pour estimer l'orientation d'un projectile, sans que l'utilisation de gyromètres soit nécessaire.

De même, l'article [8] présente un algorithme d'estimation de l'attitude d'un projectile sans gyromètre en combinant un magnétomètre et des mesures GPS dans un filtre de Kalman.

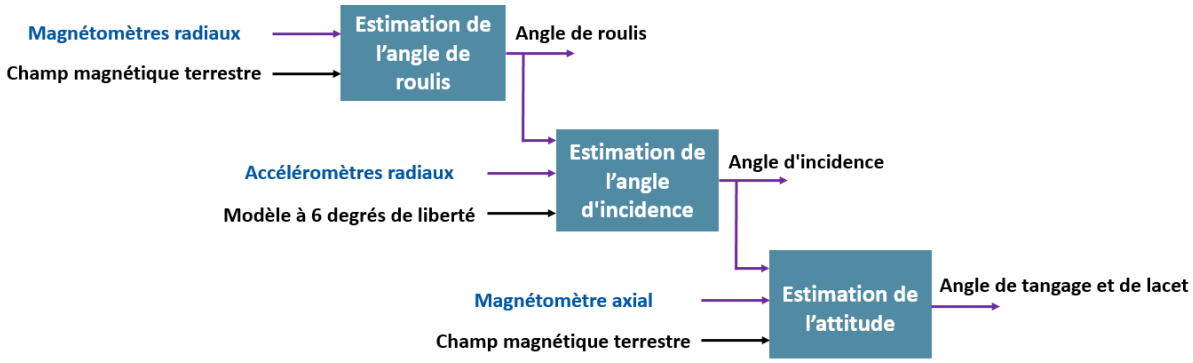

FIGURE 1.14 – Estimation de l'orientation d'un projectile à partir de magnétomètres et d'accéléromètres [43].

Solution de navigation sans GNSS

Les signaux GNSS sont de moins en moins utilisés pour la navigation des projectiles du fait de leurs indisponibilités et de la menace de brouillage et de leurrage [14]. Plusieurs approches sont proposées dans un environnement sans GNSS comme [44] qui propose une méthode d'estimation de la vitesse d'un projectile balistique à grande vitesse et à grande vitesse de rotation à partir d'accéléromètres. Une autre solution, présentée dans [45], propose un algorithme basé sur des filtres de Kalman Étendus pour déterminer la position, la vitesse et les angles de roulis, d'attaque et de dérapage à partir du champ magnétique terrestre et d'un magnétomètre trois axes embarqué dans le projectile.

De plus, l'intégration de caméras permet de pallier aux limitations du GNSS comme les articles [46], [47] qui présentent des méthodes d'estimation de trajectoires de projectiles basées sur la vision stéréo.

Pour conclure l'état de l'art relatif à la navigation des projectiles, plusieurs solutions de navigation ont été mises en œuvre, basées principalement sur un filtre de Kalman qui peut être de différentes natures suivant le modèle dynamique considéré. De plus, suivant le type de projectile considéré et la dynamique associée, certains capteurs ne peuvent être exploités. C'est le cas des mesures satellites qui sont de moins en moins utilisées du fait de leurs vulnérabilités importantes ou de certains gyromètres qui saturent au lancement pour des projectiles spécifiques. Au vu de ces diverses contraintes, l'intelligence artificielle vise à s'intégrer davantage dans les solutions de navigation afin de limiter l'influence de ces différentes limitations.

1.4 Intelligence artificielle appliquée au domaine militaire

Les récents résultats de l'intégration d'algorithmes d'intelligence artificielle (IA) dans des systèmes ont montré que cette approche permet de résoudre des problèmes complexes par ses facultés d'apprentissage. En effet, l'IA est employée dans de nombreux domaines tels que la robotique, l'éducation, la finance, la santé, le tourisme... En outre, l'IA est de plus en plus intégrée dans des programmes militaires à des fins de détection et d'identification de cibles, mais reste peu appliquée à la navigation d'objets soumis à de fortes contraintes dynamiques. Cette partie vise à présenter l'intelligence artificielle, ses enjeux et ses applications dans le domaine militaire et également son intégration dans des solutions de navigation.

1.4.1 Introduction à l'intelligence artificielle

L'intelligence artificielle est un ensemble de méthodes, basées sur une phase d'apprentissage, qui visent à imiter l'intelligence humaine afin d'accomplir des tâches complexes.

Comme illustré dans la figure 1.15, l'IA regroupe des approches de *Machine Learning* et de *Deep Learning* :

- Le *Machine Learning* est une branche de l'IA regroupant des techniques permettant d'apprendre automatiquement un ensemble de règles statistiques à partir de données structurées et catégorisées.
  Il existe plusieurs types d'apprentissage en *Machine Learning* : supervisé, semi-supervisé, non supervisé ou par renforcement [48].

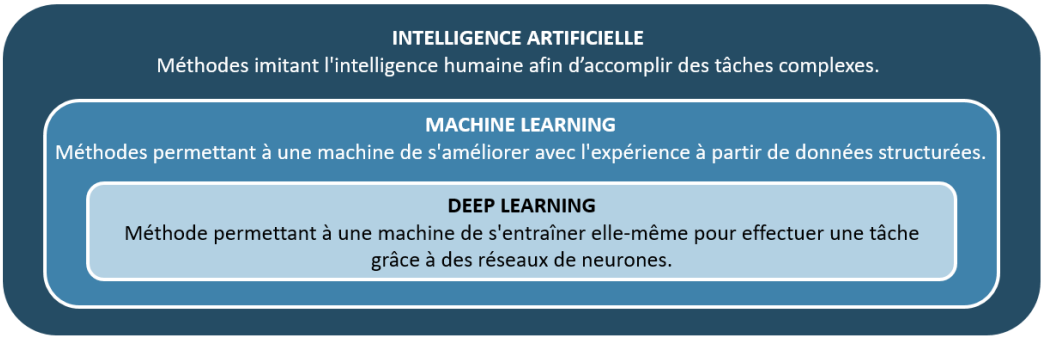

FIGURE 1.15 – Intelligence artificielle, *Machine Learning* et *Deep Learning*.

- Le *Deep Learning* est une branche de l'IA dérivée du *Machine Learning*. Le *Deep Learning* regroupe des algorithmes capables de mimer le fonctionnement du cerveau humain grâce à des réseaux de neurones artificiels afin de résoudre des tâches complexes, sans intervention humaine [49]. La phase d'apprentissage permet au réseau de corriger lui-même ses erreurs.

1.4.2 Les réseaux de neurones

Un réseau de neurones est une association de neurones artificiels interconnectés et organisés en couches permettant la résolution de problèmes complexes. Un neurone artificiel est une modélisation du fonctionnement d'un neurone biologique. Suivant le type d'application ciblée, différents types de réseaux de neurones artificiels sont utilisés.

Le neurone biologique

Le système nerveux humain est composé de nombreuses cellules formant un réseau de neurones biologiques. Chaque neurone interagit avec d'autres neurones pour former des réseaux plus ou moins complexes dans le but de transmettre et de traiter des informations.

Un neurone biologique est une cellule qui se caractérise par différents éléments comme illustré dans la figure suivante.

- Le *noyau* traite les informations reçues en activant les sorties en fonction des stimulations en entrée. Si la somme en entrée ne dépasse pas le seuil d'excitation, aucun message nerveux n'est transmis. À l'inverse, si la somme en entrée dépasse le seuil d'excitation, un message nerveux est émis via l'axone.

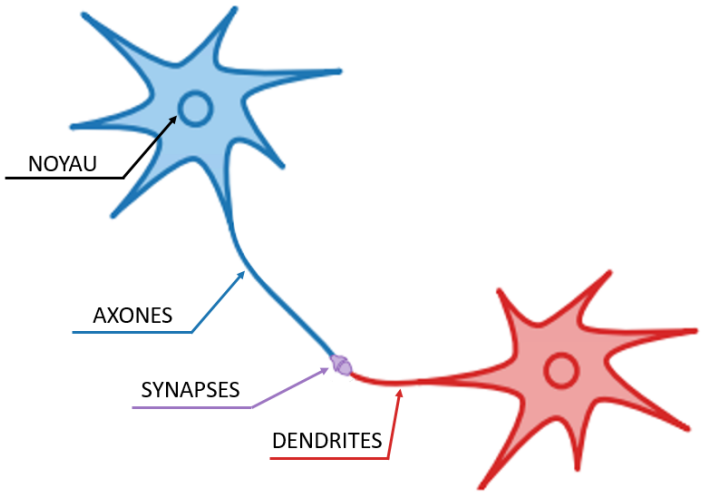

FIGURE 1.16 – Représentation d'un neurone biologique.

- Les *dendrites* acheminent vers le noyau les informations en provenance d'autres neurones.
- Les *axones* transmettent les informations vers d'autres neurones.
- Les *synapses*, points de contact entre l'axone d'un neurone et la dendrite d'un autre neurone, permettent aux neurones de communiquer entre eux.

C'est sur ce modèle de neurone biologique que se basent les éléments fondamentaux des réseaux de neurones.

Le neurone artificiel

Comme illustré dans la figure 1.17, le neurone artificiel est une représentation schématique d'un neurone biologique, de sorte que :

- les n *données d'entrée* d'un neurone $x = [x_1, ..., x_n]^T$ modélisent les dendrites,
- la *somme* Σ et la *fonction d'activation* $f(.)$ modélisent le fonctionnement du noyau d'un neurone biologique. La fonction d'activation $f(.)$ est une fonction mathématique qui convertit la somme des signaux d'entrée pondérés par les poids pour produire la sortie désirée. Suivant le problème traité, différentes fonctions d'activation peuvent être utilisées.
- les *poids* $w = [w_{1,j}, w_{2,j}, ..., w_{n,j}]^T$ pondèrent les données d'entrée x_i et modélisent les synapses d'un neurone biologique. La notation $w_{i,j}$ désigne le poids allant d'un neurone artificiel i au neurone j et est estimé lors de la phase d'apprentissage. Plus la valeur d'un poids $w_{i,j}$ est importante, plus l'entrée correspondante est influente.

- b le vecteur de biais, module la fonction d'activation afin que le modèle s'adapte aux données.
- la *sortie* y d'un neurone correspond aux axones.

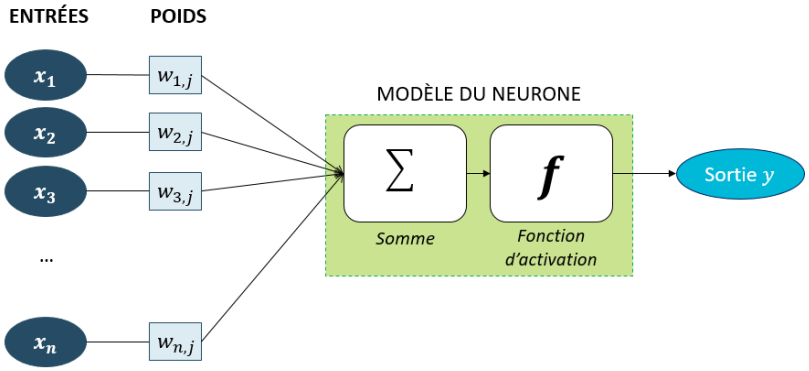

FIGURE 1.17 – Représentation d'un neurone artificiel.

Ainsi, un neurone artificiel effectue la somme des entrées pondérées par les poids afin d'être traitée par une fonction d'activation pour produire la sortie y du neurone. Le fonctionnement d'un neurone artificiel peut être résumé par l'équation suivante :

$$y = f(b + \omega^T x) \tag{1.18}$$

La fonction d'activation $f(.)$ permet de transformer de manière non linéaire les données d'entrée et peut être de différentes natures suivant le problème à traiter [50]. Les fonctions d'activation les plus communes sont la fonction *seuil*, la fonction *sigmoïde*, la fonction *Rectified Linear Unit (ReLU)*, la fonction *softmax*.

La structure d'un réseau

Un réseau de neurones est un empilement de couches composées de neurones aux propriétés spécifiques comme présenté dans la figure 1.18. Il est composé d'une couche d'entrée, d'une ou plusieurs couches cachées et d'une couche de sortie.

Suivant le problème à traiter, différentes architectures peuvent être envisagées dont les principales sont :

- Le **perceptron multicouches** *(Multilayers Perceptron - MLP)* est l'un des premiers réseaux de neurones et constitue le modèle de base des autres structures plus complexes. Le MLP est composé de plusieurs couches successives liées entre

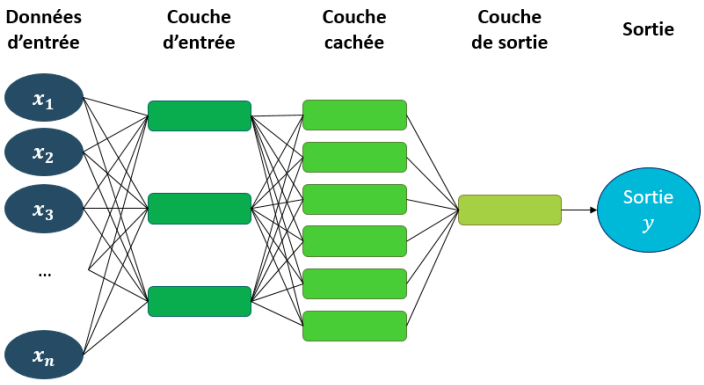

FIGURE 1.18 – Représentation d'un réseau de neurones avec une couche cachée.

elles et au sein desquelles l'information circule de la couche d'entrée vers la couche de sortie. Le MLP est utilisé pour résoudre des problèmes de classification et de régression non linéaire.

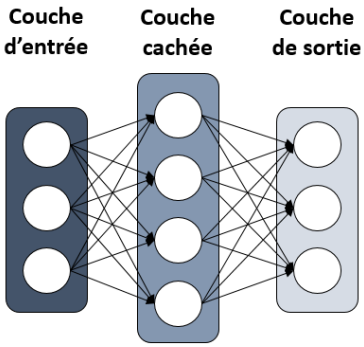

FIGURE 1.19 – Structure d'un perceptron multicouches *(Multilayers Perceptron - MLP)*.

- Le **réseau de neurones convolutifs** *(Convolutional Neural Network - CNN)* est une classe de réseau de neurones caractérisé par plusieurs couches dont les couches de convolution qui permettent de reconnaître et d'extraire des caractéristiques spatiales dans les données d'entrée.
  Un CNN est généralement composé de couches de *Pooling* pour condenser les informations détectées par la couche de convolution, de couche d'activation *ReLU* pour discriminer les informations importantes et de couche *Fully Connected* pour regrouper les informations apprises. Le CNN est principalement utilisé pour la classification d'image, la détection d'objet et la reconnaissance vocale.

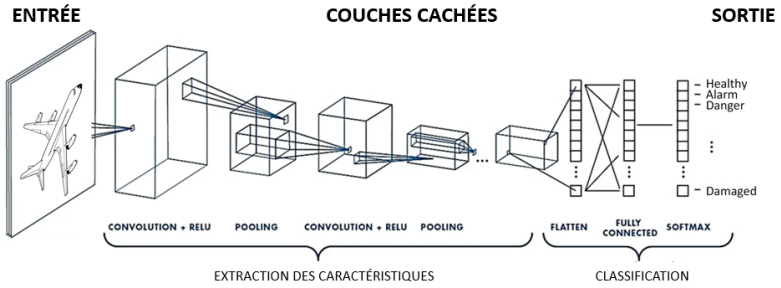

FIGURE 1.20 – Structure d'un réseau de neurones convolutifs *(Convolutional Neural Network - CNN)*.

- Le **réseau de neurones récurrents** *(Recurrent Neural Networks - RNN)* est une classe de réseau de neurones caractérisé par des connexions récurrentes ; les sorties précédentes sont utilisées comme entrées supplémentaires. Ces connexions récurrentes permettent de mémoriser des informations passées.
  Les deux variantes les plus populaires des RNN sont le LSTM *(Long Short-Term Memory)* et le GRU *(Gated Recurrent Unit)*. Les RNN sont essentiellement utilisés pour modéliser des données séquentielles comme la prédiction de séries temporelles ou la traduction.

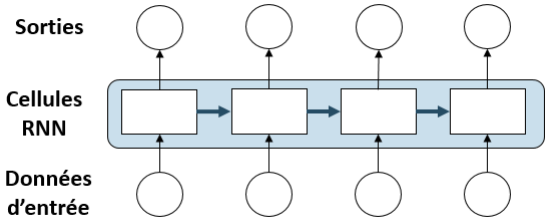

FIGURE 1.21 – Structure d'un réseau de neurones récurrents *(Recurrent Neural Networks - RNN)*.

1.4.3 Entraînement d'un réseau de neurones

L'apprentissage d'un réseau de neurones vise à modifier les poids $w_{i,j}$ de chaque neurone pour que le réseau réalise la fonction désirée. Ces paramètres sont modifiés par un algorithme d'optimisation afin de minimiser l'erreur d'estimation évaluée par une fonction de perte sur le jeu de données d'entraînement.

Jeu de données

Un réseau de neurones utilise trois jeux de données distincts :

- Le *jeu de données d'entraînement*, utilisé lors de la phase d'entraînement, est l'ensemble de données sur lequel les paramètres du réseau sont ajustés afin d'obtenir la fonction désirée.
- Le *jeu de données de validation*, utilisé lors de la phase d'entraînement, permet d'évaluer les performances du réseau sans mettre à jour les paramètres pour continuer ou non l'entraînement et détecter un éventuel sur apprentissage.
- Le *jeu de données de test* permet d'évaluer le réseau pour la fonction désirée.

Fonction de perte

La fonction de perte, notée $\mathcal{L}(y, \widehat{y})$, est une fonction mathématique qui vise à évaluer l'erreur entre la sortie désirée y et la sortie déterminée par le réseau $\widehat{y}$. La fonction d'erreur indique la précision des prévisions du réseau et donc, quel ajustement doit être apporté aux poids par l'algorithme d'optimisation à chaque itération. Il en existe plusieurs types pour des problèmes de régression dont les trois plus utilisés et présentés dans la figure 1.22 sont :

- l'erreur moyenne absolue *(MAE - Mean Absolute Error, L1 Loss)*,

$$\mathcal{L}(y, \widehat{y}) = |y - \widehat{y}| \tag{1.19}$$

  La MAE est facile à calculer, robuste face aux valeurs aberrantes et au bruit mais n'est pas dérivable en zéro.

- l'erreur quadratique moyenne *(MSE - Mean Squared Error, L2 Loss)*,

$$\mathcal{L}(y, \widehat{y}) = (y - \widehat{y})^2 \tag{1.20}$$

  La MSE est la fonction de perte par défaut pour des problèmes de régression. Elle pénalise de façon analogue les grandes et les petites erreurs et est donc peu robuste face aux valeurs aberrantes.

- la perte d'Huber *(Huber loss)*,

$$\mathcal{L}(y, \widehat{y}) = \begin{cases} \frac{1}{2}(y - \widehat{y})^2 & pour \quad |y - \widehat{y}| \leq \delta \\ \delta|y - \widehat{y}| - \frac{1}{2}\delta^2 & pour \quad |y - \widehat{y}| > \delta \end{cases} \tag{1.21}$$

La perte d'Huber atténue l'importance donnée aux valeurs aberrantes tout en donnant de l'importance aux petites erreurs. De plus, elle est dérivable en 0 et le paramètre δ peut être ajusté pour maximiser la précision du modèle. Néanmoins, elle est coûteuse.

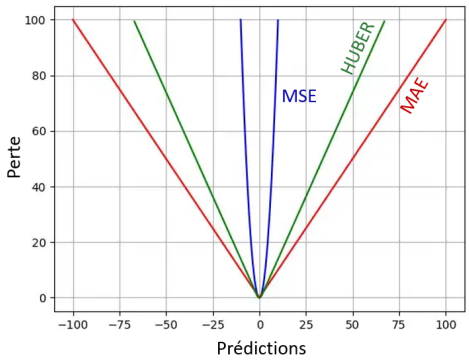

FIGURE 1.22 – Les fonctions de perte pour des problèmes de régression.

Algorithme d'optimisation

L'algorithme d'optimisation vise à déterminer de façon itérative les poids $w_{i,j}$ qui minimisent la fonction de perte $\mathcal{L}(y, \widehat{y})$. Les algorithmes d'optimisation utilisés pour l'entraînement des réseaux de neurones s'appuient majoritairement sur l'algorithme de descente du gradient qui permet de trouver le minimum de n'importe quelle fonction convexe en convergeant progressivement vers celui-ci. Cet algorithme est résumé par trois étapes :

1) Initialisation aléatoire des poids.
2) Après chaque nouvelle observation y, le gradient de la fonction de perte $\mathcal{L}(y, \widehat{y})$ est calculé par rapport aux poids du réseau, tel que :

$$\nabla \mathcal{L} = \frac{\partial \mathcal{L}(y, \widehat{y})}{\partial w_{i,j}} \tag{1.22}$$

   Le gradient ∇ représente la pente de la fonction de perte $\mathcal{L}(y, \widehat{y})$.
3) Les poids $w_{i,j}$ sont mis à jour dans la direction opposée au gradient afin de minimiser la fonction de perte $\mathcal{L}(.)$ de sorte que :

$$w_{i,j} \leftarrow w_{i,j} - \eta \nabla \mathcal{L} \tag{1.23}$$

   avec η le taux d'apprentissage.

Il existe différents algorithmes d'optimisation tels que la descente de gradient stochastique, *Nesterov Accelerated Gradient, Adagrad, AdaDelta* ou *Adam* [51].

Taux d'apprentissage

Le taux d'apprentissage η est un hyperparamètre qui impacte les performances du modèle comme présenté sur la figure 1.23 :

- si le taux d'apprentissage η est trop faible, la convergence vers un minimum local est lente et la mise à jour des poids prend du temps avant de trouver le minimum de la fonction de perte. Ainsi, le réseau converge lentement et le risque de trouver un minimum local est important.
- si le taux d'apprentissage η est trop grand, l'algorithme oscille autour d'un optimum sans stabilisation. Un taux d'apprentissage trop élevé peut faire diverger le réseau.

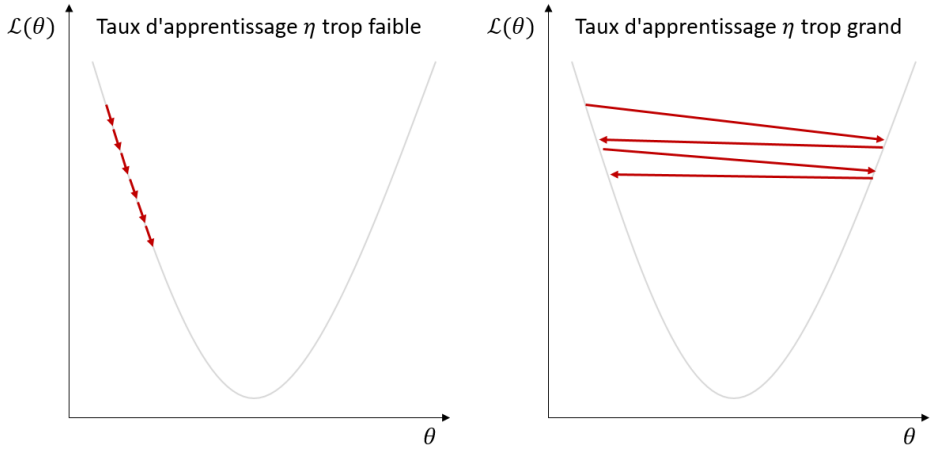

FIGURE 1.23 – Influence du taux d'apprentissage sur la recherche du minimum de la fonction de perte.

1.4.4 Applications de l'IA dans le domaine militaire

Les méthodes d'intelligence artificielle sont de plus en plus développées dans le domaine militaire [52]-[55], notamment par l'armée américaine pour :

- La **surveillance et la reconnaissance de cible** où l'IA est utilisée afin de détecter, d'identifier et suivre les objets d'intérêts présents dans une image provenant de divers systèmes.

Exemple : le projet MAVEN [56] présenté par le département américain de la Défense (DoD), exploite l'IA pour trier des données, dans un but de surveillance et de reconnaissance de zones d'intérêts à partir d'images provenant de drones. MAVEN a été déployé dès 2017 au Moyen-Orient pour aider les militaires à exploiter les vidéos provenant des drones ScanEagle afin d'identifier les objets d'intérêts.

- La **maintenance prédictive** où l'IA permet d'établir le moment optimal pour changer une pièce sur un système à partir de ses données enregistrées afin de réduire le temps et le coût de maintenance, et également, pour prévenir d'éventuelles défaillances.

  Exemple : le département américain de la Défense (DoD), en collaboration avec la compagnie C3IoT, développe un programme pour effectuer la maintenance prédictive du chasseur F-16 et de l'avion de surveillance AWACS E-3 Sentry. Ce programme collecte les données enregistrées par l'avion et l'IA analyse ces données afin de fournir des alertes et de déterminer le moment optimal pour changer des pièces [57].

- L'**entraînement militaire** où l'IA est utilisée par des simulateurs afin de créer une multitude de scénarios variés et réalistes, dans le but d'améliorer les formations des militaires.

  Exemple : l'entreprise Aptima travaille pour l'US Air Force Research Laboratory (AFRL) afin de développer un logiciel de formation des pilotes, pour les entraîner à piloter l'avion dans divers scénarios et également pour les former à réagir face à des attaques ennemies [58].

- L'**analyse et l'aide à la décision** où l'IA permet notamment d'extraire et de traiter les éléments pertinents d'un flux d'information, de les analyser ou de prédire des événements.

  Exemple : la Royal Navy a utilisé l'IA dans le cadre de deux applications nommées Startle et Sycoiea. Startle permet d'aider les marins dans les tâches de surveillance via des images aériennes en fournissant des recommandations et des alertes en temps réel. Sycoiea exploite les informations de Startle pour évaluer les menaces et recommander la meilleure façon d'agir pour répondre aux menaces. L'US Air Force utilise également l'IA pour aider les pilotes à travers un programme nommé Artuμ [59] qui commande différents capteurs d'un avion afin de trouver les lanceurs ennemis.

- La **cybersécurité** où l'IA vise à être utilisée pour détecter des intrusions malveillantes, anticiper des menaces, mesurer le niveau de résistance des systèmes et pour protéger et bloquer les réseaux, les communications, les programmes infor-

matiques et les données contre les cybermenaces.

Exemple : le programme CASTLE (Cyber Agents for Security Testing and Learning Environments) soutenu par la Defense Advanced Research Projects Agency (DARPA) vise à se défendre des cybermenaces en utilisant l'IA pour réduire la vulnérabilité d'un réseau [60]. De plus, en décembre 2015, les services de renseignement militaires russes ont mené une cyberattaque destructrice contre le réseau électrique ukrainien.

- Le **renseignement** où l'IA permet de prétraiter, d'identifier et d'extraire des informations pertinentes parmi un grand volume de données. L'IA peut également être utilisée pour filtrer de grandes quantités de contenu provenant d'actualités et de médias sociaux afin d'aider à identifier de nouvelles informations.

  Exemple : le projet SKYNET de l'agence nationale de la sécurité des États-Unis analyse des communications cellulaires grâce à l'IA pour extraire des informations sur d'éventuels suspects de terrorisme. Cette application est notamment employée au Pakistan pour identifier et détecter les communications entre cellules terroristes [61].

- Le **combat collaboratif** et la **fusion multicapteurs** où l'IA permet d'échanger, de partager, de fusionner et d'exploiter des informations provenant de différentes sources afin de coordonner des systèmes pour optimiser les décisions prises.

  Exemple : le projet Guardian Angel soutenu par l'ARL (U.S. Army Research Laborator), fusionne des informations de différents capteurs (radars infrarouges et à lumière visible, capteurs multispectraux, laser...) pour identifier des cibles en Irak et en Afghanistan [62].

La France a publié un rapport *"Task-Force IA"* [63] en 2019, pour proposer une stratégie d'utilisation de l'IA pour la défense française. Plusieurs axes sont identifiés tels que l'aide à la décision, le combat collaboratif, la cyberdéfense, la logistique, le soutien en condition opérationnelle, le renseignement, la robotique et l'administration.

1.4.5 Applications de l'IA en navigation

Bien que largement intégrée dans les programmes de développement militaire, l'IA est relativement peu appliquée à l'estimation de la trajectoire d'un projectile, quoique certaines solutions soient parfaitement adaptées à cette tâche. Les solutions de navigation et d'IA proposées dans la littérature peuvent être regroupées principalement en deux catégories :

- La *navigation hybride* regroupe les solutions de navigation basées d'une part sur un modèle mathématique et d'autre part sur un réseau de neurones. Les solutions

hybrides les plus employées sont les *Deep Kalman Filter* ; filtre de Kalman dont l'un ou plusieurs modèles du filtre sont remplacés par un réseau de neurones. Suivant le type d'application visée, différents réseaux peuvent être employés.

- La *navigation de bout-en-bout* vise à exclure tous modèles mathématiques et utiliser exclusivement des réseaux de neurones pour la navigation. Généralement, les réseaux employés sont des réseaux récurrents tels que les LSTM, les GRU ...

La navigation hybride

Plusieurs auteurs proposent des solutions de navigation hybride, principalement appliquées à l'estimation de trajectoires de véhicules terrestres.

IMU/GNSS : Comme mentionné dans la partie (*1.4*), les signaux GNSS ne sont pas toujours disponibles du fait de la configuration du terrain ou de la trop faible puissance des signaux. Ainsi, les filtres de Kalman, comme présentés dans la figure 1.13, basés sur

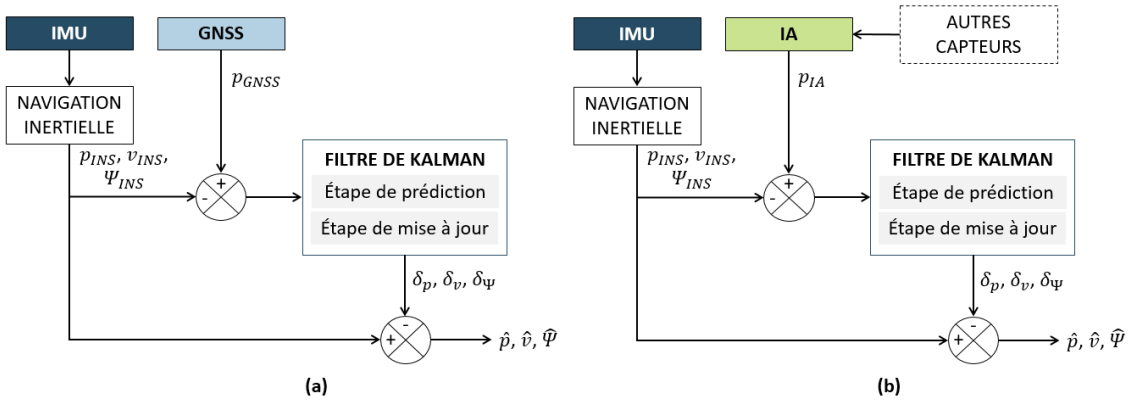

FIGURE 1.24 – (a) Filtre de Kalman IMU/GNSS pour l'estimation de la trajectoire d'un corps en mouvement dans le cas où les mesures GNSS sont disponibles. (b) *Deep Kalman Filter* proposé pour pallier aux problèmes d'indisponibilité du GNSS.

les mesures de l'IMU et du GNSS divergent lorsque ce dernier n'est plus disponible pour corriger les prédictions basées sur les mesures de l'IMU. Afin de pallier aux problèmes d'indisponibilité du signal GNSS, plusieurs *Deep Kalman Filter* sont proposés dans la littérature pour remplacer les signaux manquants par un réseau de neurones, comme illustré schématiquement dans la figure 1.24.

Les articles [64]-[70] proposent des *Deep Kalman Filters* pour résoudre les problèmes d'indisponibilité du GNSS à l'aide de réseaux de neurones pour générer des pseudos me-

sures dudit GNSS. Ces travaux sont principalement appliqués à la navigation des robots terrestres et utilisent différents types de réseaux suivant les données d'entrée considérées.

Observations d'un filtre de Kalman : La précision d'un filtre de Kalman est dépendante de la qualité des observations, nécessaires pour l'étape de mise à jour. Or, les observations ne sont pas toujours fidèles pour corriger les prédictions du filtre. En effet, pour satisfaire les contraintes financières et d'espace imposées au système, des capteurs à faible coût sont utilisés produisant des observations bruitées et biaisées. Une seconde contrainte est le traitement de ces mesures, nécessitant parfois des modélisations complexes et valables que sous certaines hypothèses. C'est pourquoi, l'IA peut être utilisée afin de générer des observations précises afin de corriger les prédictions du filtre comme illustré dans la figure 1.25.

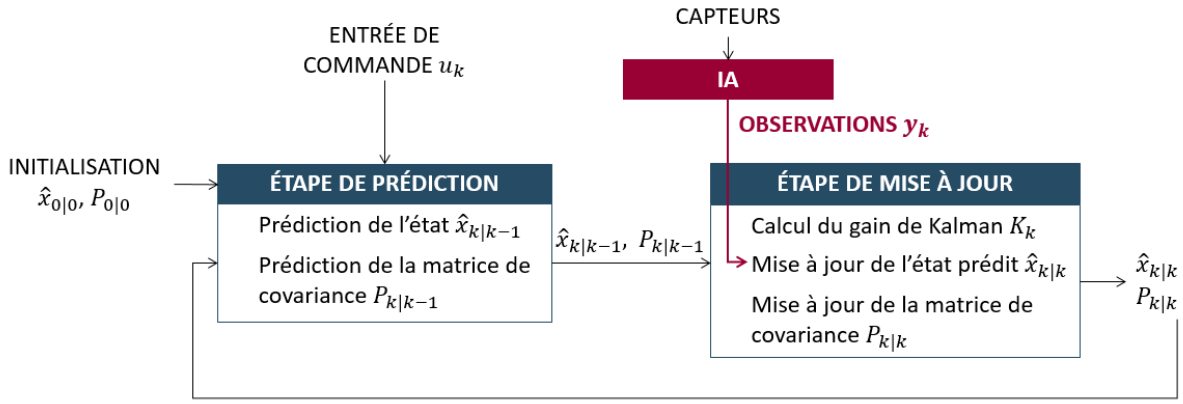

FIGURE 1.25 – *Deep Kalman Filter* où le réseau de neurones est utilisé pour prétraiter les mesures des capteurs afin de générer les observations d'un filtre de Kalman.

Les articles [71]-[76] présentent des filtres de Kalman pour la navigation où des réseaux de neurones sont entraînés pour générer des observations d'après les mesures de capteurs imparfaits ou complexes à traiter.
Par exemple, l'article [72] propose de déterminer la mesure et l'incertitude de l'angle de dérive d'un véhicule par des LSTM alimentés par des capteurs embarqués, car cet angle est complexe à mesurer. Des MLP sont présentés dans [73] afin de générer des pseudos mesures de l'angle de roulis d'un véhicule à l'aide de l'IMU pour satisfaire des contraintes de coût. Des CNN sont employés dans [75] pour extraire les caractéristiques des signaux électriques mesurés par des électrodes placées sur la peau. Sans avoir recours à un modèle mathématique de régression, le CNN produit ainsi des pseudos observations du mouvement afin de mettre à jour un filtre de Kalman pour estimer la position de la main.

Ce même article propose également plusieurs LSTM pour estimer le modèle d'évolution et d'observation, les covariances et le gain d'un filtre de Kalman.

Covariances d'un filtre de Kalman : Un filtre de Kalman est sensible au réglage de la matrice de covariance du bruit de modèle et du bruit de mesure, qui traduisent la confiance accordée aux modèles et aux mesures. Dans le cas d'un filtre de Kalman classique, ces matrices sont constantes et souvent réglées manuellement par l'utilisateur. Dans certaines applications, il est préférable d'utiliser des matrices de covariance variables et adaptées au problème de navigation, comme dans le cas où un capteur est dégradé dans une configuration donnée. Pour cela, des réseaux de neurones peuvent être entraînés pour régler dynamiquement les matrices de covariance du bruit de modèle et du bruit de mesure d'un filtre de Kalman comme présenté dans la figure 1.26. Pour ce faire, plusieurs solutions sont présentées dans [77]-[82] utilisant des réseaux de neurones comme le MLP, le CNN ou le RNN.

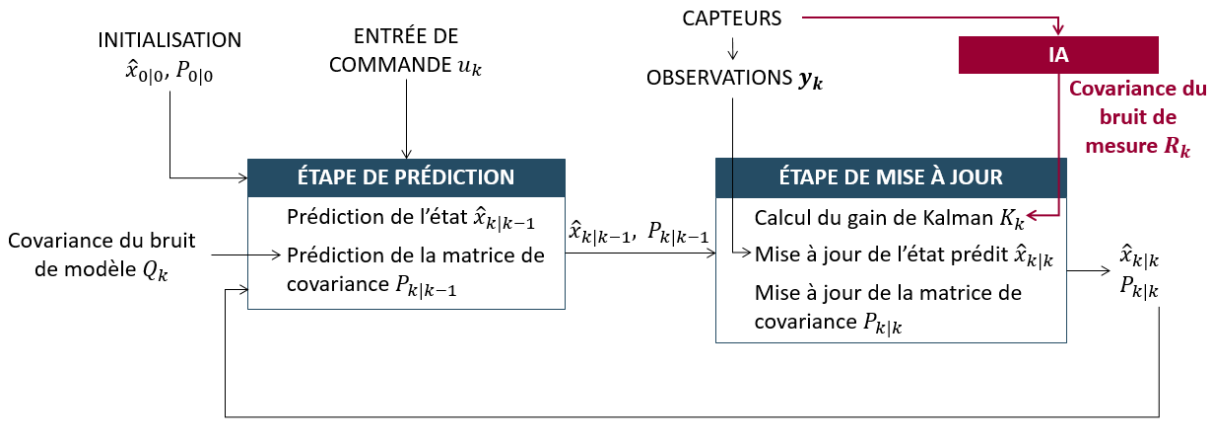

FIGURE 1.26 – *Deep Kalman Filter* dans le cas où la covariance du bruit de mesure est ajustée par un réseau de neurones.

Modèles d'un filtre de Kalman : Les différents modèles qui constituent un filtre de Kalman sont parfois complexes à identifier et nécessitent plusieurs approximations. C'est pourquoi, plusieurs articles proposent de modéliser certaines étapes d'un filtre de Kalman par un réseau de neurones comme illustré schématiquement dans la figure 1.27.
Ainsi, [83] propose de modéliser les erreurs de l'IMU d'un filtre de Kalman pour estimer la trajectoire d'un véhicule terrestre. L'article [84] présente des réseaux récurrents pour apprendre à calculer le gain de Kalman d'un filtre. Le modèle d'évolution du filtre peut également être déterminé par des réseaux de neurones comme dans [85]. Une méthode

d'estimation du modèle d'évolution, d'observation et des covariances du bruit de modèle et de mesure à partir de LSTM est présentée dans [86].

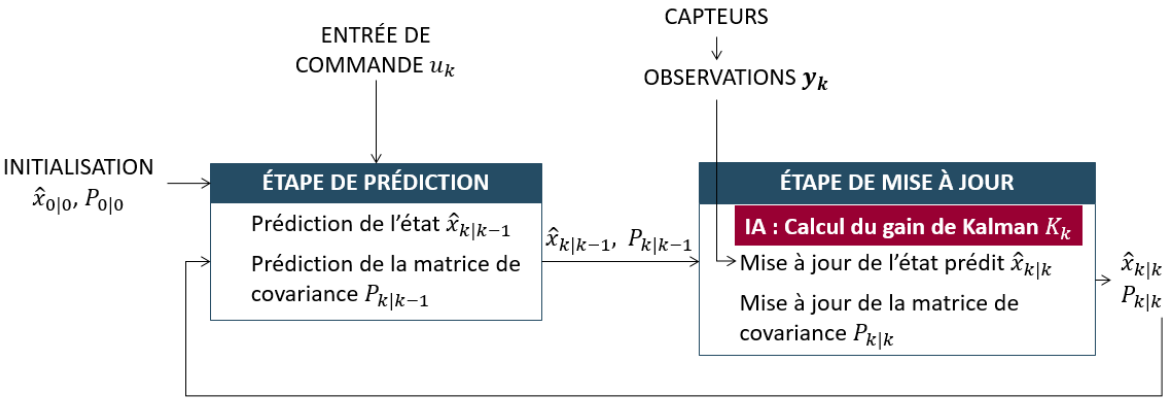

FIGURE 1.27 – *Deep Kalman Filter* pour estimer l'un des modèles d'un filtre de Kalman tel que le calcul du gain de Kalman.

Dans les chapitres 3 et 5 de ce document, plusieurs méthodes de navigation hybrides présentées ci-dessus sont appliquées à la navigation des projectiles, afin d'évaluer dans quelles mesures les réseaux de neurones permettent d'optimisation des filtres de Kalman.

La navigation de bout-en-bout

La prédiction de la trajectoire d'un objet en mouvement est essentielle pour son guidage. Pour cela, de nombreuses solutions, majoritairement basées sur des réseaux récurrents, sont utilisées. Ce type de réseau, parfaitement adapté au traitement de séries temporelles, permet d'apprendre un modèle à partir des données brutes des capteurs. Ainsi, le réseau apprend notamment les contraintes physiques du corps en mouvement ainsi qu'un modèle d'erreur des capteurs. Cette solution, bien que basée exclusivement sur l'IA, permet de capturer des relations complexes entre les données, ce qui est généralement difficile à modéliser mathématiquement. Des exemples d'utilisation de l'IA pour la prédiction de trajectoires sont présentés dans la table 1.2.

Le chapitre 4 de ce document présente une méthode de navigation de bout-en-bout pour estimer les trajectoires de plusieurs projectiles, afin d'évaluer la précision et les limites de cette approche.

Article	Application	Méthode	Données d'entrée
[87]	Prédiction de la trajectoire d'un avion	LSTM	Positionnements précédents enregistrés par des stations radars ADS-B au sol.
[88]	Prédiction et génération de trajectoires humaines	LSTM et GAN	Coordonnées et le temps associé.
[89]	Prédiction de la localisation d'un navire	LSTM bidirectionnel	Latitude, longitude, vitesse et cap du navire.
[90]	Prédiction du mouvement humain	GAN	Poses précédentes.
[91]	Prédiction du mouvement humain	RNN	Positions GPS précédentes.
[92]	Prédiction de la trajectoire d'un véhicule	LSTM encodeur décodeur	Vitesses du véhicule, vitesse de lacet, coordonnée et vitesse relative des autres véhicules.
[93]	Prédiction de la trajectoire d'un véhicule	MLP,LSTM, GRU, Transformer	Positions précédentes.
[94]	Prédiction de la trajectoire d'un projectile	LSTM	Données radar incomplètes et bruitées.
[95]	Prédiction de la trajectoire de piétons	CNN	Positions précédentes.

TABLE 1.2 – Utilisation des réseaux de neurones pour l'estimation des trajectoires de corps en mouvement.

1.5 Jeu de données de trajectoires de projectiles

Les résultats présentés dans ce document exploitent un ensemble de données de tirs de projectiles généré par BALCO *(BALlistic COde)* [96]-[98]. Cette partie présente le jeu de données de trajectoires de projectiles et définit également les repères employés dans ce document.

1.5.1 Aperçu général de BALCO *(BALlistic COde)*

BALCO est un logiciel de simulation de trajectoire de projectiles développé à l'ISL d'après la recommandation de normalisation de l'OTAN 4618 [96]. Un modèle mathématique de six ou sept degrés de liberté est développé pour décrire le mouvement du projectile et est discrétisé par une méthode de Runge-Kutta du septième ordre. BALCO est déve-

loppé pour simuler les trajectoires de projectiles conventionnels ainsi que des projectiles guidés dotés d'actionneurs (ailettes, canards ou propulseurs).

Le modèle de trajectoire de BALCO prend en compte plusieurs modèles tels que :

- le *modèle terrestre* : terre plate (gravité fixe définie par l'utilisateur), sphérique (modèle de masse ponctuelle STANAG 4355), ellipsoïdale (modèle mondial WGS84),
- le *modèle d'atmosphère* : atmosphère standard (ISO 2533-1975) ou définie par l'utilisateur,
- le *modèle aérodynamique* : profil aérodynamique axisymétrique ou non axisymétrique (coefficients aérodynamiques dépendant du temps), coefficients aérodynamiques décrits par des tables de correspondance ou des fonctions polynomiales,
- le *modèle de poussée* défini par l'utilisateur ou par le modèle STANAG 4355,
- le *modèle d'actionneur* : ailettes, canards, *spoilers*, propulseurs,
- le *modèle de guidage, navigation et contrôle (GNC)* implémenté sur une plate-forme indépendante du code à l'aide d'une interface de communication.

La précision de BALCO a été validée par rapport aux programmes de référence PRODAS *(Projectile Rocket Ordnance Design and Analysis System)* et HTRAJ *(US Army ARDEC)* [96]. Pour cela, 400 tests sont effectués en considérant quatre projectiles aux données aérodynamiques connues : le *50 cal. M8*, le *Basic Finner* 30 mm, le mortier 120 mm M934 et l'obus 155 mm M795. La validation a été effectuée en considérant différentes conditions initiales (vitesse initiale, élévation du canon, angle d'attaque, vitesse angulaire) et conditions météorologiques (standard, froid, chaud et vent).

1.5.2 Les systèmes de coordonnées pour la navigation

Avant de définir les données fournies par BALCO, il est nécessaire de présenter les différents repères de navigation utilisés au cours de ces travaux.

Le repère ECEF

Le repère ECEF *(Earth-Centered Earth-fixed)* est un système de coordonnées fixe et centré sur la Terre. L'origine est le centre de la Terre, l'axe z_{ECEF} pointe dans la direction du pôle Nord, l'axe x_{ECEF} vers l'intersection de l'équateur du méridien de Greenwich (longitude de 0 °) et l'axe y_{ECEF} vers l'intersection de l'équateur et du méridien de longitude 90°. Le repère ECEF est principalement utilisé pour le positionnement par satellite car il permet un positionnement par rapport au centre de la Terre. Cependant,

pour la plupart des problèmes de navigation pratiques, la connaissance du positionnement par rapport à la surface de la Terre est préférable.

Le repère local de navigation

Le repère local de navigation est un repère orthogonal centré sur un point fixe à la surface de la Terre et tangent à sa surface. Les axes x_n, y_n et z_n sont orientés respectivement selon le nord, l'est et le centre de la Terre *(NED - North East Down)*.

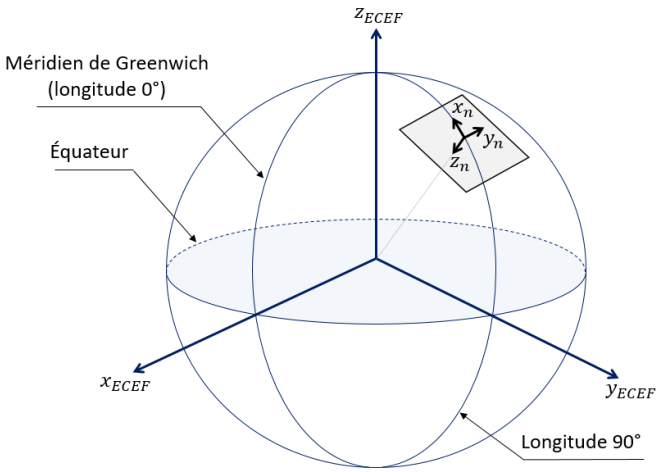

FIGURE 1.28 – Le repère ECEF *(Earth-Centered Earth-fixed)* et le repère local de navigation.

Le repère objet

Le repère objet (*body frame*) est un repère hypothétique idéal placé exactement au centre de gravité du projectile. Les axes x_b, y_b et z_b de ce repère sont orientés respectivement vers l'avant (direction de déplacement), la droite et le bas (direction de la gravité) du projectile.

Le repère capteur

Le repère capteur (*sensor frame*) est rigidement fixé au projectile et est désaxé par rapport au centre de gravité du projectile. Ce repère est considéré comme le référentiel où sont effectuées les mesures inertielles.

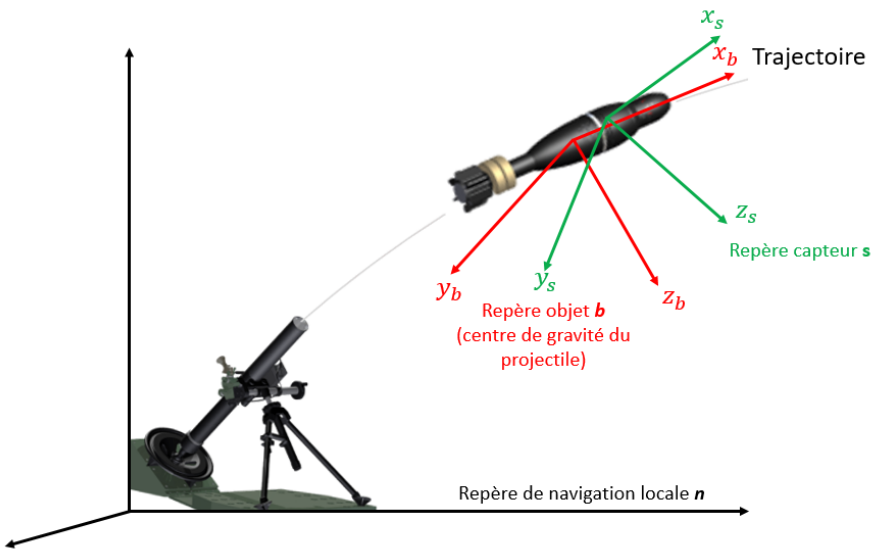

FIGURE 1.29 – Repère objet (*body frame*) et repère capteur (*sensor frame*).

1.5.3 Données de simulation BALCO

BALCO a permis de générer un jeu de données de trajectoires de projectiles nécessaire au développement des travaux de cette thèse. Plusieurs projectiles, caractérisés par des trajectoires et des paramètres spécifiques, ont été étudiés.

Données de BALCO

BALCO fournit des simulations de tirs de projectiles où chaque simulation comprend :

- les ***mesures IMU :*** mesures des gyromètres $\omega \in \mathbb{R}^{3\times1}$, des accéléromètres $a \in \mathbb{R}^{3\times1}$ et des magnétomètres $h \in \mathbb{R}^{3\times1}$.

  Comme le montre la figure 1.30, trois types de mesures inertielles sont disponibles :

  - les *mesures IMU parfaites* effectuées dans le repère objet ***b*** (repère rouge sur la figure 1.29), dans le cas idéal où les trois capteurs inertiels sont parfaitement placés au centre de gravité du projectile.

    Dans cette configuration, aucun modèle de capteur n'est pris en compte, fournissant ainsi des mesures inertielles idéales, c'est-à-dire sans bruit ni biais. Ces mesures ne sont pas exploitées dans ce travail mais sont nécessaires pour fournir des données inertielles réalistes.

- les *mesures IMU* effectuées dans le repère capteur $\boldsymbol{s}$ (repère vert sur la figure 1.29) issues des *mesures IMU parfaites* où un modèle d'erreur de capteur est ajouté.

  Ce modèle d'erreur de capteur comprend un modèle de désalignement, un facteur de sensibilité, un biais et un bruit (supposé gaussien de moyenne nulle) spécifique à chaque axe du capteur.

  Ainsi, ce modèle d'erreur modélise avec précision des mesures inertielles produites par une centrale inertielle embarquée dans un projectile.
- les *mesures IMU DYN* effectuées dans le repère capteur $\boldsymbol{s}$ (repère vert sur la figure 1.29) issues de *mesures IMU* auxquelles une fonction de transfert est ajoutée pour chacun des trois capteurs. Ainsi, les *mesures IMU DYN* sont modélisées par :

$$y_{capteur,\ IMU\ \ DYN}(s) = \frac{1}{1 + as + bs^2 + cs^3}\ y_{capteur,\ IMU}(s) \tag{1.24}$$

  avec $y_{capteur,IMU}$ les *mesures IMU* du capteur considéré, $y_{capteur,IMU\ \ DYN}$ les *mesures IMU DYN* correspondantes et avec a, b et c les coefficients de la fonction de transfert déterminés par BALCO. Ce modèle de capteur permet de modéliser la réponse des trois capteurs sur leurs plages de fonctionnement respectives.

Les paramètres des modèles d'erreur des capteurs inertiels sont déterminés d'après les fiches techniques de capteurs MEMS à bas coût disponibles dans le commerce et d'après des mesures d'étalonnage effectuées par l'ISL.

- le ***champ magnétique de référence*** $h_n \in \mathbb{R}^{3\times1}$ exprimé dans le repère de navigation $\boldsymbol{n}$ (repère noir sur la figure 1.29), supposé constant pendant le vol du projectile et définit d'après le modèle magnétique mondial.
- les ***paramètres de vol*** spécifiques à chaque type de projectile considéré. Ces paramètres sont détaillés dans la suite de ce document.
- le ***vecteur temps*** $k\Delta_t$ où Δ_t est la période d'échantillonnage de l'IMU.
- la ***trajectoire de référence*** comprenant la position $p \in \mathbb{R}^{3\times1}$, la vitesse $v \in \mathbb{R}^{3\times1}$ et les angles d'Euler $\Psi \in \mathbb{R}^{3\times1}$ du projectile dans le repère de navigation $\boldsymbol{n}$ à la fréquence de l'IMU.

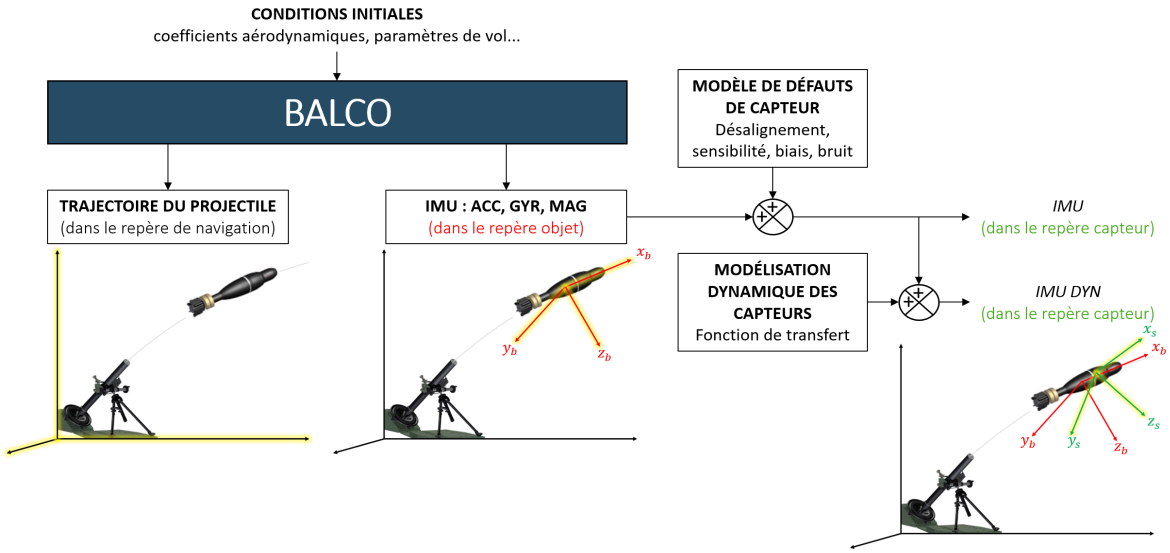

FIGURE 1.30 – Principe de fonctionnement de BALCO.

Projectiles étudiés

Le jeu de données produit par BALCO comprend les trajectoires simulées de quatre projectiles différents présentés dans la figure 1.31 : le *Basic Finner* de 30 mm, le projectile

	PROJECTILE STABILISÉ PAR AILETTES	PROJECTILE STABILISÉ EN ROTATION
TRAJECTOIRE BALISTIQUE DE PLUSIEURS DIZAINES DE KILOMÈTRES (de 4 à 30 km)	MORTIER 120 mm 	OBUS 155 mm 
PORTÉE	De 4 à 8 km	De 25 à 30 km
VITESSE DE ROTATION	Jusqu'à 6 Hz	300 Hz
TIR TENDU (JUSQU'À QUELQUES KILOMÈTRES)	BASIC FINNER 	**PROJECTILE de** 40 mm 
VITESSE DE ROTATION	Jusqu'à 6 Hz	600 Hz

FIGURE 1.31 – Caractéristiques des trajectoires d'un mortier de 120 mm, d'un obus de 155 mm, d'un *Basic Finner* de 30 mm et d'un projectile de 40 mm.

de 40 mm, le mortier de 120 mm et l'obus de 155 mm. L'ensemble des trajectoires de ces quatre projectiles est obtenu en considérant un modèle de Terre sphérique, un modèle d'atmosphère standard, des coefficients aérodynamiques fonctions du nombre de *Mach*[7] et sans aucun modèle de poussée.

Le mortier de 120 mm et l'obus de 155 mm sont caractérisés par des trajectoires balistiques contrairement au *Basic Finner* et au projectile de 40 mm qui sont caractérisés par des tirs tendus. Comme présenté dans la figure 1.32, une trajectoire balistique est une trajectoire parabolique qui permet au projectile d'atteindre un objectif placé derrière un obstacle. Les tirs tendus sont caractérisés par de faibles portées et les trajectoires se rapprochent de droites.

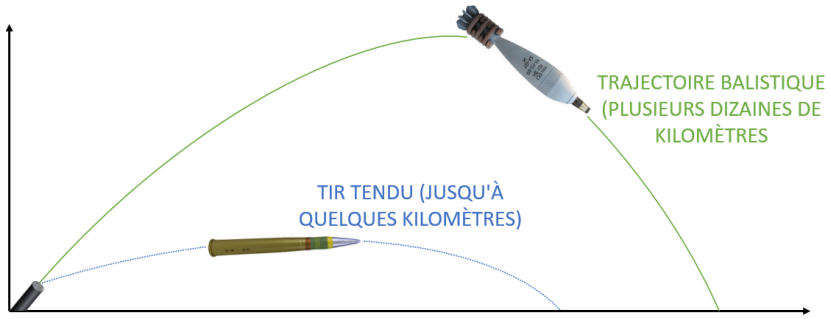

FIGURE 1.32 – Trajectoire balistique et tir tendu.

Le mortier de 120 mm et le *Basic Finner* disposent d'ailettes et ont donc de faibles vitesses de rotation contrairement à l'obus de 155 mm et au projectile de 40 mm. Les ailettes servent à stabiliser les projectiles pendant le vol.
Les paramètres de vol associés aux quatre projectiles sont présentés dans le tableau 1.3 suivant.

Mortier 120 mm et *Basic Finner*	Obus de 155 mm et projectile de 40 mm
Angle de braquage des ailettes	Roulis initial
Vitesse initiale	Vitesse initiale
Angle d'élévation du canon	Angle d'élévation du canon

TABLE 1.3 – Paramètres de vol spécifiques aux projectiles considérés.

La vitesse initiale v_0 représente la vitesse du projectile à la sortie du canon. L'angle d'élévation du canon α varie suivant les différentes trajectoires. Une trajectoire balistique

7. Vitesse du projectile en vol.

est caractérisée par un angle d'élévation du canon important alors qu'un tir tendu est caractérisé par un faible angle d'élévation du canon.

1.6 Conclusion

Ce chapitre a présenté l'état de l'art existant concernant les projectiles, la navigation des projectiles et l'intelligence artificielle.
Deux types de capteurs sont généralement embarqués dans les projectiles : une centrale inertielle composée d'accéléromètres, de gyromètres et possiblement de magnétomètres, et un récepteur GNSS. Les mesures inertielles fournissent une solution de navigation précise à court terme mais qui dévie à long terme à cause de la dérive des capteurs. À l'inverse, le récepteur GNSS fournit des informations de positionnement absolu mais les signaux ne sont pas toujours disponibles. D'après les articles présentés dans ce chapitre, les mesures de ces capteurs sont habituellement fusionnées par des filtres de Kalman. Toutefois, de plus en plus de travaux sont proposés dans un environnement sans GNSS.
De plus, les récents progrès de l'intelligence artificielle en font un outil intéressant pour le domaine militaire. Bien que largement intégrée dans des programmes militaires, l'IA est très peu employée pour des systèmes de navigation. En effet, la littérature présente majoritairement des applications de l'IA pour la navigation terrestre.
Enfin, au vu de la diversité du jeu de données de trajectoires utilisé dans le cadre de ces travaux, l'analyse de plusieurs approches de navigation basées en partie sur l'IA semble nécessaire, afin de définir leurs apports et leurs intérêts pour estimer les trajectoires des différents projectiles.

Chapitre 2

Les filtres de Kalman invariants pour la navigation des projectiles

Sommaire

2.1 Introduction

Les modèles de navigation sont non linéaires, c'est pourquoi, un filtre de Kalman Étendu *(Extended Kalman Filter - EKF)* est généralement employé pour l'estimation de la trajectoire d'un corps en mouvement. Les dynamiques non linéaires sont alors linéarisées au premier ordre autour des estimations précédentes afin d'appliquer la méthodologie

classique d'un filtre de Kalman. Cette étape suppose donc des erreurs d'estimation suffisamment petites, ce qui n'est pas toujours vérifié en pratique. C'est pourquoi, du fait de la linéarisation, l'EKF ne possède aucune garantie de convergence et peut créer des fausses observabilités. De plus, l'EKF est conçu pour les espaces euclidiens, ce qui rend complexe la représentation de l'orientation d'un corps en mouvement, plus facilement modélisée sur un groupe de Lie [99], [100].

Pour cela, une extension de l'EKF a été développée : le filtre de Kalman Étendu Invariant *(Invariant Extended Kalman Filter - IEKF)* [82], [99]-[106]. Un IEKF est un observateur non linéaire asymptotiquement stable défini sur un groupe de Lie matriciel. En effet, la dynamique de l'erreur invariante de l'IEKF est indépendante des états estimés, permettant ainsi de prouver les propriétés de stabilité et qu'un IEKF ne crée pas de fausses observabilités. De plus, l'utilisation des groupes de Lie matriciels permet à l'IEKF de prendre en compte les non linéarités de l'espace d'état.

L'IEKF est de plus en plus utilisé pour résoudre des problèmes de navigation du fait de ses propriétés de convergence et d'observabilité. Pour cela, le système doit satisfaire une propriété fondamentale qui n'est généralement vérifiée que par une partie de la dynamique du système. Dans ce cas, un filtre de Kalman Imparfait Invariant Étendu *(Imperfect Invariant Extended Kalman Filter - Imp.IEKF)* [99] peut être employé. Il s'agit d'une combinaison entre un EKF et un IEKF dans le sens où une partie des états estimés évolue dans un groupe de Lie et une seconde partie évolue dans un espace euclidien. L'Imp.IEKF ne bénéficie pas des propriétés de stabilité de l'IEKF mais présente tout de même des caractéristiques d'observabilité et de précision meilleures que celles d'un EKF.

Ce chapitre détaille la théorie des Filtres de Kalman Étendu Invariant et de ses drivées. Ces types de filtres sont employés dans le cadre de cette thèse pour résoudre plusieurs problèmes de navigation des projectiles. Les objectifs de ce chapitre sont :

- → de présenter la théorie des filtres de Kalman Étendu Invariant et des filtres de Kalman Imparfait Invariant Étendu, utilisés lorsque la dynamique n'évolue pas entièrement sur un groupe de Lie matriciel.
- → de vérifier, à travers un exemple d'application employé pour la navigation des projectiles, les propriétés d'un Imp.IEKF. Pour cela, la précision de l'Imp.IEKF est comparée à celle d'un EKF standard, dérivé à partir des mêmes modèles dynamiques. De plus, les propriétés d'observabilité des deux filtres sont analysées.

La première partie *(2.2)* présente la théorie associée aux filtres de Kalman Étendu Invariant. Une seconde partie *(2.3)* détaille le principe de fonctionnement des filtres de

Kalman Imparfait Invariant Étendu et ses spécificités. Enfin, une troisième partie *(2.4)* se focalise sur les performances d'un Imp.IEKF pour la navigation des projectiles afin de visualiser leurs pertinences. Ce filtre est comparé à un EKF standard et une étude d'observabilité est présentée.

2.2 Le filtre de Kalman Étendu Invariant

Un filtre de Kalman Étendu Invariant *(Invariant Extended Kalman Filter - IEKF)* est un observateur invariant non linéaire asymptotiquement stable pour des systèmes définis sur un groupe de Lie matriciel et possédant des symétries [99]-[105].

Les observateurs invariants, détaillés dans [107], sont une classe d'observateurs [1] appropriée aux systèmes dynamiques possédant des symétries, c'est-à-dire, aux systèmes ayant le même comportement après avoir subi une transformation. Pour cela, les groupes de Lie matriciels permettent de préserver les symétries d'un système.

Un IEKF, comme tous filtres de Kalman, vise à minimiser l'erreur d'estimation. Un IEKF est associé à une erreur invariante sur le groupe de Lie qui est linéarisée autour de l'élément neutre du groupe. La dynamique d'évolution de l'erreur invariante est totalement indépendante des estimations et est totalement définie par une équation différentielle linéaire sur l'algèbre de Lie associée. Les propriétés vérifiées par l'erreur invariante et la linéarisation autour de l'élément neutre du groupe permettent de démontrer qu'un IEKF est un observateur non linéaire asymptotiquement stable [2] [99]-[105], [108].

De plus, plusieurs exemples d'applications montrent que l'IEKF surpasse l'EKF pour résoudre des problèmes de navigation [82], [101]-[104], [106]. En conséquence, l'IEKF est un estimateur approprié pour les applications de navigation.

Cette partie présente tout d'abord les groupes de Lie matriciels, leurs algèbres et les applications associées. Puis, les théorèmes fondamentaux vérifiés par l'erreur invariante des IEKF sont énoncés et les équations des IEKF sont détaillées.

1. Un observateur estime l'état d'un système dynamique à partir de mesures incomplètes et indirectes de l'état [107].
2. L'erreur d'estimation d'un observateur invariant d'un système évoluant sur un groupe de Lie et possédant des symétries est indépendante des états estimés, donc les propriétés des groupes de Lie permettent de concevoir des observateurs convergents [99].

2.2.1 Groupes de Lie matriciels, algèbres et applications

Cette sous-partie détaille les groupes de Lie matriciels, les algèbres et les opérateurs associés. De plus, deux groupes de Lie généralement utilisés pour la navigation sont présentés : le groupe de rotation $SO(3)$ et le groupe des doubles isométries $SE_2(3)$.

Définition 2.2.1 (**Groupe [109]-[111]**). *Un groupe G est un ensemble muni d'une loi de composition interne définie telle que :*

$$(x, y) \mapsto x * y \tag{2.1}$$

La loi de composition interne (2.1) *possède les propriétés suivantes :*

- *elle est associative :* $\forall (a, b, c) \in G, \quad a * (b * c) = (a * b) * c$
- *elle admet un élément neutre, c'est-à-dire qu'il existe un élément unique* $e \in G$ *tel que* $\forall a \in G, \quad a * e = e * a = a$
- *tout élément* $a \in G$ *admet un symétrique, c'est-à-dire qu'il existe un élément de* G*, noté* a^{-1}*, tel que :* $\forall a \in G, \quad a^{-1} * a = a * a^{-1} = e$

Définition 2.2.2 (**Groupe de Lie matriciel [99]-[104], [112], [113]**). *Un groupe de Lie matriciel $G \subset \mathbb{R}^{N \times N}$ est un groupe muni d'une structure différentiable (les applications de multiplication matricielle et d'inversion matricielle sont dérivables). En d'autres termes, les propriétés suivantes sont vérifiées :*

$$I_N \in G, \qquad \forall (a, b) \in G, ab \in G, \qquad \forall a \in G, a^{-1} \in G \tag{2.2}$$

Ainsi, l'élément identité du groupe de Lie matriciel est la matrice identité. Le produit et l'inverse sont lisses (de classe C^∞) donc ces fonctions sont indéfiniment dérivables sur un intervalle donné [109], [110].

Les propriétés d'un groupe de Lie matriciel permettent de définir un espace tangent au groupe en tout point.

Définition 2.2.3 (**Algèbre de Lie [99]-[104]**). *L'espace tangent à l'élément identité I_N du groupe de Lie matriciel G est appelé l'algèbre de Lie $\mathfrak{g}$. L'algèbre de Lie $\mathfrak{g}$, nécessairement unique, est donc un sous-espace de $\mathbb{R}^{N \times N}$ de dimension d, tel que :*

$$\mathfrak{g} \in \mathbb{R}^{d \times d} \subset \mathbb{R}^{N \times N} \tag{2.3}$$

Localement, autour de la matrice d'identité I_N, le groupe de Lie matriciel G et l'algèbre sont identifiés à un espace vectoriel $\mathbb{R}^{d\times 1}$. Pour cela, comme présenté dans la figure 2.1, plusieurs opérateurs sur les groupes de Lie matriciels et leurs algèbres nécessitent d'être définis avant de présenter les IEKF.

Définition 2.2.4 (**Matrice exponentielle $exp_m(.)$ [99]-[104]**). *L'algèbre de Lie $\mathfrak{g}$ est reliée au groupe de Lie matriciel G par la matrice exponentielle $exp_m(.)$ classique :*

$$exp_m : \begin{pmatrix} \mathfrak{g} & \rightarrow & G & \\ A & \mapsto & exp_m(A) & = & \sum_{k=0}^{+\infty} \frac{A^k}{k!} \end{pmatrix} \tag{2.4}$$

Définition 2.2.5 (**Carte linéaire $\mathcal{L}_{\mathfrak{g}}(.)$ [99]-[104]**). *L'algèbre $\mathfrak{g}$ est identifiée à l'espace euclidien associé $\mathbb{R}^{d\times 1}$ par la carte linéaire $\mathcal{L}_{\mathfrak{g}}(.)$, tel que :*

$$\mathcal{L}_{\mathfrak{g}} : \begin{pmatrix} \mathbb{R}^{d\times 1} & \rightarrow & \mathfrak{g} \\ \xi & \mapsto & \mathcal{L}_{\mathfrak{g}}(\xi) \end{pmatrix} \tag{2.5}$$

Définition 2.2.6 (**Carte exponentielle $exp(.)$ [99]-[104]**). *L'espace euclidien $\mathbb{R}^{d\times 1}$ est relié au groupe de Lie matriciel G via l'application $exp(.)$ définie telle que :*

$$exp : \begin{pmatrix} \mathbb{R}^{d\times 1} & \rightarrow & G & \\ \xi & \mapsto & exp(\xi) & = & exp_m\left(\mathcal{L}_{\mathfrak{g}}(\xi)\right) \end{pmatrix} \tag{2.6}$$

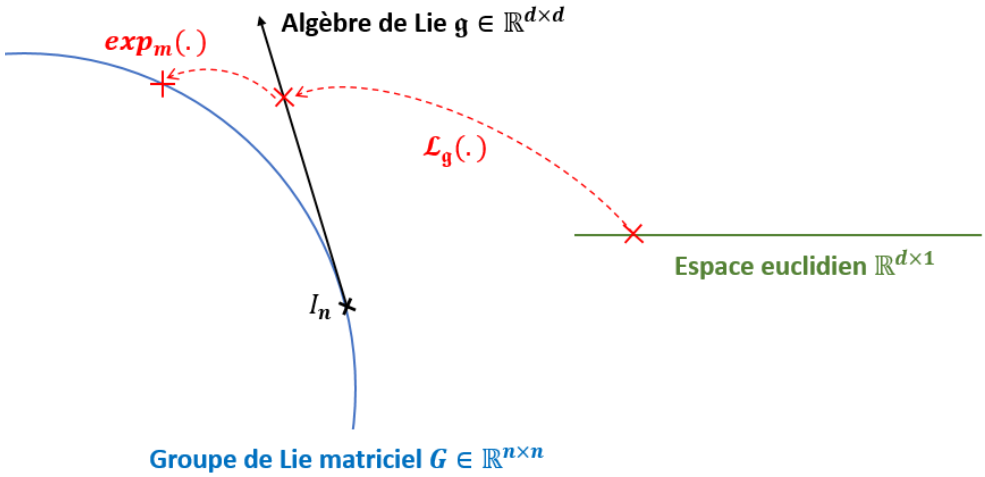

FIGURE 2.1 – Espace euclidien $\mathbb{R}^{d\times 1}$, algèbre de Lie $\mathfrak{g}$ et groupe de Lie G.

Il existe plusieurs groupes de Lie matriciels, mais seuls les groupes $SO(3)$ et $SE_2(3)$ [99]-[104] sont détaillés dans ce document du fait de leurs utilisations pour la navigation.

Le groupe de rotation $SO(3)$ est un groupe de Lie utilisé en navigation pour représenter l'orientation R d'un corps dans l'espace. Le groupe de rotation $SO(3)$ et son algèbre $\mathfrak{so}(3)$, représentant l'espace des matrices asymétriques, sont définis tels que :

$$SO(3) := \left\{ R \in \mathbb{R}^{3\times 3}, R^T R = I_3, det(R) = 1 \right\} \qquad \mathfrak{so}(3) = \left\{ A \in \mathbb{R}^{3\times 3}, A = -A^T \right\} \quad (2.7)$$

L'opérateur linéaire $\mathcal{L}_{\mathfrak{so}(3)} : \mathbb{R}^{3\times 1} \to \mathfrak{so}(3)$ et l'application exponentielle $\exp_{so(3)} : \mathbb{R}^{3\times 1} \to SO(3)$ sont définis tel que :

$$\mathcal{L}_{\mathfrak{so}(3)} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}_\times = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix} \quad (2.8)$$

$$exp_{SO(3)}(\xi_R) = I_3 + \frac{sin(\|\xi_R\|)}{\|\xi_R\|}[\xi_R]_\times + 2\frac{sin(\|\xi_R\|/2)^2}{\|\xi_R\|^2}[\xi_R]_\times^2 \quad (2.9)$$

L'opérateur $\mathcal{L}_{\mathfrak{so}(3)}(.)$, noté $[.]_\times$, vérifie plusieurs propriétés de sorte que $\forall\ (a,\ b) \in \mathbb{R}^{3\times 1}$, $\forall\ R \in SO(3)$,

$$[a]_\times = -[a]_\times^T, \qquad [a]_\times[b]_\times - [b]_\times[a]_\times = [-[b]_\times a]_\times, \qquad [Ra]_\times = R[a]_\times R^T. \quad (2.10)$$

Le groupe $SE_2(3)$ est un groupe de Lie matriciel permettant de modéliser l'orientation, la vitesse et la position d'un corps en mouvement de sorte que :

$$SE_2(3) := \left\{ \begin{bmatrix} R & v & p \\ 0_{1\times 3} & 1 & 0 \\ 0_{1\times 3} & 0 & 1 \end{bmatrix}, R \in SO(3), (v,p) \in \mathbb{R}^{3\times 1} \right\} \quad (2.11)$$

L'algèbre de Lie $\mathfrak{se}_2(3)$ associée à $SE_2(3)$ est :

$$\mathfrak{se}_2(3) = \left\{ \begin{bmatrix} [\xi_R]_\times & \xi_v & \xi_p \\ 0_{1\times 3} & 0 & 0 \\ 0_{1\times 3} & 0 & 0 \end{bmatrix}, (\xi_R, \xi_v, \xi_p) \in \mathbb{R}^{3\times 1} \right\} \quad (2.12)$$

L'opérateur linéaire $\mathcal{L}_{\mathfrak{se}_2(3)} : \mathbb{R}^{9\times 1} \to \mathfrak{g}_{\mathfrak{se}_2(3)}$ et l'application exponentielle $\exp_{SE_2(3)}$:

$\mathbb{R}^{9\times1} \rightarrow SE_2(3)$ sont définis comme suit :

$$\mathcal{L}_{\mathfrak{se}_2(3)} \begin{bmatrix} \xi_R \\ \xi_v \\ \xi_p \end{bmatrix} = \begin{bmatrix} [\xi_R]_\times & \xi_v & \xi_p \\ 0_{1\times3} & 0 & 0 \\ 0_{1\times3} & 0 & 0 \end{bmatrix} \tag{2.13}$$

$$\exp_{SE_2(3)} \left(\begin{bmatrix} \xi_R \\ \xi_v \\ \xi_p \end{bmatrix} \right) = I_5 + S + \frac{1-cos(\|\xi_R\|)}{\|\xi_R\|^2} S^2 + \frac{\|\xi_R\|-sin(\|\xi_R\|)}{\|\xi_R\|^3} S^3 \text{ avec } S = \mathcal{L}_{\mathfrak{se}_2(3)} \tag{2.14}$$

2.2.2 Théorèmes fondamentaux de la théorie des IEKF

Cette sous-partie présente les deux théorèmes fondamentaux de la théorie des IEKF vérifiés par l'erreur invariante ; le théorème de la dynamique d'erreur autonome et la propriété log-linéaire de l'erreur.

Soit $G \subset \mathbb{R}^{N\times N}$ un groupe de Lie matriciel et $\mathfrak{g}$ son algèbre de Lie. Soit un système dynamique défini par l'équation suivante :

$$\frac{d}{dt}\chi_t = f_{u_t}(\chi_t) \tag{2.15}$$

avec l'état $\chi_t \in G$ et u_t est une variable d'entrée.

Définition 2.2.7 (**Erreurs invariantes [99]-[104]**). *Les erreurs invariantes à gauche ${\eta_t}^L$ et à droite ${\eta_t}^R$ entre deux trajectoires distinctes χ_t et $\widehat{\chi}_t$ sont définies telles que :*

$$\textit{Erreur invariante à gauche } {\eta_t}^L = {\chi_t}^{-1}\widehat{\chi}_t \tag{2.16}$$

$$\textit{Erreur invariante à droite } {\eta_t}^R = \widehat{\chi}_t{\chi_t}^{-1} \tag{2.17}$$

L'erreur invariante est au cœur de la théorie des IEKF et permet de mesurer l'écart entre deux trajectoires sur le groupe de Lie matriciel. Ces erreurs sont invariantes aux multiplications à gauche ou à droite par un élément du groupe.

Définition 2.2.8 (**Propagation autonome de l'erreur [99]-[104]**). *L'erreur invariante η_t a une propagation indépendante des estimations s'il existe une fonction $g_{u_t}(.)$ indépendante de χ ou $\widehat{\chi}$ de sorte que :*

$$\frac{d}{dt}\eta_t = g_{u_t}(\eta_t) \tag{2.18}$$

Définition 2.2.9 (**Dynamique affine de groupe [99]-[104]**). *Une dynamique $f_{u_t}(.)$ évoluant sur le groupe de Lie matriciel G est dite affine de groupe si elle satisfait :*

$$\forall (a,b) \in G, \quad f_{u_t}(ab) = a f_{u_t}(b) + f_{u_t}(a) b - a f_{u_t}(I_N) b \tag{2.19}$$

où I_N est la matrice identité, c'est-à-dire l'élément neutre de G.

L'erreur invariante, associée à une dynamique affine de groupe permet de vérifier deux propriétés fondamentales qui constituent la base de la théorie des IEKF.

Théorème 2.2.1 (**Dynamique d'erreur autonome [99]-[104]**). *Les trois conditions suivantes sont équivalentes pour un système dont la dynamique est définie par* (2.15) *:*

1. *L'erreur invariante à gauche ${\eta_t}^L$ a une propagation autonome,*
2. *L'erreur invariante à droite ${\eta_t}^R$ a une propagation autonome,*
3. *La dynamique $f_{u_t}(.)$ est affine de groupe ; elle vérifie la proposition* (2.19).

Si la condition (2.19) *est satisfaite, les dynamiques des erreurs invariantes à droite et à gauche sont indépendantes de la trajectoire et satisfont les équations suivantes :*

$$\frac{d}{dt}{\eta_t}^L = g^L_{u_t}({\eta_t}^L) = f_{u_t}({\eta_t}^L) - f_{u_t}(I_N){\eta_t}^L \tag{2.20}$$

$$\frac{d}{dt}{\eta_t}^R = g^R_{u_t}({\eta_t}^R) = f_{u_t}({\eta_t}^R) - {\eta_t}^R f_{u_t}(I_N) \tag{2.21}$$

Le théorème 2.2.1 montre que si la dynamique du système (2.15) est *affine de groupe*, c'est-à-dire qu'elle vérifie la proposition (2.19), alors l'évolution de l'erreur invariante η_t est indépendante des états du système.

Théorème 2.2.2 (**Propriété log-linéaire de l'erreur [99]-[104]**). *Soit n_t l'erreur invariante telle que définie par* (2.17) *et* (2.16).
Soit une erreur initiale arbitraire $\xi_0 \in \mathbb{R}^{d\times 1}$ telle que $\eta_0 = exp(\xi_0)$. Soit A_t la matrice définie telle que : $g_{u_t}(exp(\xi_t)) = \mathcal{L}_{\mathfrak{g}}(A_t\xi_t) + o(\|\xi_t\|^2)$.
Si ξ_t est définit pour $t > 0$ par l'équation différentielle suivante dans $\mathbb{R}^{d\times 1}$,

$$\frac{d}{dt}\xi_t = A_t\xi_t \tag{2.22}$$

Alors,

$$\forall t \geqslant 0, \ \eta_t = exp(\xi_t) \tag{2.23}$$

Le théorème 2.2.2 [3] implique que l'erreur non linéaire peut être déterminée à partir d'une équation différentielle linéaire variant dans le temps. L'application exponentielle $exp(.)$ permet de linéariser η_t et d'approcher localement η_t par un vecteur $\xi_t \in \mathbb{R}^{d\times 1}$ [99]-[104] Ainsi, des problèmes non linéaires impliquent des équations d'erreur linéaires, à condition que la variable d'erreur soit judicieusement choisie.

2.2.3 Équations d'un filtre de Kalman Étendu Invariant

Cette sous-partie détaille les équations d'un L-IEKF *(Left-Invariant Extended Kalman Filter)* et d'un R-IEKF *(Right-Invariant Extended Kalman Filter).*

Soit le système dynamique non linéaire défini par l'équation suivante :

$$\frac{d}{dt}\chi_t = f_{u_t}(\chi_t) + \chi_t w_t \tag{2.24}$$

avec $u_t \in \mathbb{R}^{n_u}$ l'entrée de commande, χ_t l'état évoluant sur un groupe de Lie matriciel $G \subset \mathbb{R}^{N\times N}$ dont l'algèbre est notée $\mathfrak{g} \in \mathbb{R}^{d\times d}$, $w_t \in \mathfrak{g}$ un bruit blanc Gaussien, et avec $f_{u_t}(.)$ la fonction d'évolution *affine de groupe* vérifiant ainsi la proposition (2.19).

Filtre de Kalman Étendu Invariant à gauche : L-IEKF

Le système (2.24) est associé à des observations invariantes à gauche telles que [101] :

$$Y_t = \chi_t d + V \tag{2.25}$$

avec $d \in \mathbb{R}^{N\times 1}$ un vecteur connu constant et $V \in \mathbb{R}^{N\times 1}$ un bruit Gaussien.

L'erreur invariante à gauche $\eta_t \in G$ entre l'état vrai χ_t et l'état estimé $\widehat{\chi}_t$ est alors,

$$\eta_t = {\chi_t}^{-1}\widehat{\chi}_t. \tag{2.26}$$

D'après le théorème 2.2.1 *(Dynamique d'erreur autonome)*, comme la dynamique du système (2.24) est *affine de groupe* (2.19), la dynamique d'évolution de l'erreur invariante est indépendante de l'estimation $\widehat{\chi}_t$ et vérifie l'équation suivante :

$$\frac{d}{dt}\eta_t = g_{u_t}(\eta_t) - w_t\eta_t = f_{u_t}(\eta_t) - f_{u_t}(I_N)\eta_t - w_t\eta_t \tag{2.27}$$

3. La démonstration des théorèmes 2.2.1 et 2.2.2 sont disponibles dans [101].

Comme dans un EKF conventionnel, l'erreur invariante est supposée petite afin de linéariser au premier ordre le système dynamique pour appliquer la méthodologie standard d'un filtre de Kalman. Lorsque l'erreur invariante est petite, $\eta_t \approx I_N$, alors, localement autour de la matrice d'identité I_N, le groupe de Lie matriciel G et son algèbre peuvent être identifiés à un espace vectoriel $\mathbb{R}^{d\times 1}$. Donc, l'erreur invariante η_t est identifiée par l'application exponentielle $exp(.)$ à un élément de $\mathbb{R}^{d\times 1}$, nommée erreur linéarisée. L'erreur linéarisée $\xi_t \in \mathbb{R}^{d\times 1}$ est alors définie telle que :

$$\eta_t = exp(\,\xi_t) \,= exp_m(\mathcal{L}_{\mathfrak{g}}(\xi_t)) \approx I_N + \mathcal{L}_{\mathfrak{g}}(\xi_t) \tag{2.28}$$

Étape de prédiction d'un LIEKF : La prédiction de l'état estimé $\widehat{\chi_t}$ est obtenue en considérant la dynamique du système (2.24) avec des bruits nuls et la prédiction de la covariance P_t est donnée par l'équation de Riccati :

$$\frac{d}{dt}\widehat{\chi_t} = f_{u_t}(\widehat{\chi_t}), \qquad \frac{d}{dt}P_t = A_t P_t + P_t {A_t}^T + \widehat{Q}_t \tag{2.29}$$

avec $\widehat{\chi_t} \in G$ l'état estimé, $f_{u_t}(.)$ la fonction d'évolution *affine de groupe* vérifiant l'équation (2.19), $P_t \in \mathbb{R}^{d\times d}$ la matrice de covariance de l'erreur linéarisée, $A_t \in \mathbb{R}^{d\times d}$ la matrice d'évolution et $\widehat{Q}_t \in \mathbb{R}^{d\times d}$ la matrice de covariance du bruit de modèle modifié $\widehat{w}_t \in \mathbb{R}^{d\times 1}$ définie telle que $\widehat{Q}_t = cov(\widehat{w}_t)$.
La dynamique d'évolution de l'erreur linéarisée $\frac{d}{dt}\xi_t = A_t\xi_t + \widehat{w}_t$ d'après le théorème 2.2.2 permet d'identifier A_t tel que $g_{u_t}(\,exp(\,\xi_t)\,) \,= \mathcal{L}_{\mathfrak{g}}(\,A_t\xi_t) + o(\,\|\xi_t\|^2)$, et $\widehat{w}_t \in \mathbb{R}^{d\times 1}$ le bruit de modèle modifié tel que $\mathcal{L}_{\mathfrak{g}}(\widehat{w}_t) = -w_t$ avec $\widehat{w}_t \in \mathbb{R}^{d\times 1}$ et $w_t \in \mathfrak{g}$.

La discrétisation de (2.29) permet de déterminer l'état $\widehat{\chi}_{k|k-1}$ et la covariance $P_{k|k-1}$ prédite à temps discret.

Étape de mise à jour d'un LIEKF : Les prédictions sont mises à jour via les observations invariantes à gauche (2.25) de sorte que :

$$\widehat{\chi}_{k|k} = \widehat{\chi}_{k|k-1} exp\left(K_k\left((\widehat{\chi}_{k|k-1})^{-1}Y - d\right)\right), \qquad P_{k|k} = (I_N - K_k H)\,P_{k|k-1} \tag{2.30}$$

avec K_k la matrice de gain de Kalman telle que $K_k = P_{k|k-1}H^T(HP_{k|k-1}H^T + \widehat{N}_k)^{-1}$, H la matrice d'observation telle que $H\xi_t = -\mathcal{L}_{\mathfrak{g}}(\xi_t)d$ et $\widehat{N}_k$ la matrice de covariance du bruit de mesure modifié $\widehat{\chi}_k^{-1}V_k$.

Filtre de Kalman Étendu Invariant à droite : R-IEKF

Soit le système dynamique (2.24) associé à des observations invariantes à droite :

$$Y_t = {\chi_t}^{-1} d + V \tag{2.31}$$

avec $d \in \mathbb{R}^{N\times 1}$ un vecteur connu constant et $V \in \mathbb{R}^{N\times 1}$ un bruit Gaussien.

L'erreur invariante à droite est par définition $\eta_t = \widehat{\chi}_t {\chi_t}^{-1} \in G$. D'après le théorème 2.2.1, la dynamique du système (2.24) est *affine de groupe* (2.19) donc la dynamique d'évolution de l'erreur invariante est indépendante de l'estimation $\widehat{\chi}_t$:

$$\frac{d}{dt}\eta_t = g_{u_t}(\,\eta_t) \,-\left(\widehat{\chi}_t w_t \widehat{\chi}_t^{-1}\right)\eta_t = f_{u_t}(\,\eta_t) \,-\eta_t f_{u_t}(I_N) - \left(\widehat{\chi}_t w_t \widehat{\chi}_t^{-1}\right)\eta_t \tag{2.32}$$

Dans le cas de petites erreurs, l'erreur invariante η_t peut être linéarisée. Soit $\xi_t \in \mathbb{R}^{d\times 1}$ l'erreur linéarisée définie telle que :

$$\eta_t = exp(\,\xi_t) \,= exp_m(\mathcal{L}_{\mathfrak{g}}(\xi_t)) \approx I_N + \mathcal{L}_{\mathfrak{g}}(\xi_t) \tag{2.33}$$

Étape de prédiction d'un RIEKF : La prédiction de l'état $\widehat{\chi_t}$ et de la covariance P_t sont données par les équations suivantes :

$$\frac{d}{dt}\widehat{\chi_t} = f_{u_t}(\widehat{\chi_t}), \qquad \frac{d}{dt}P_t = A_t P_t + P_t {A_t}^T + \widehat{Q}_t \tag{2.34}$$

avec $\widehat{\chi_t} \in G$ l'état estimé, $f_{u_t}(.)$ la fonction d'évolution *affine de groupe* (2.19), $P_t \in \mathbb{R}^{d\times d}$ la matrice de covariance de l'erreur linéarisée, $A_t \in \mathbb{R}^{d\times d}$ la matrice d'évolution du système et $\widehat{Q}_t \in \mathbb{R}^{d\times d}$ la matrice de covariance du bruit de modèle modifié $\widehat{w}_t$.
D'après le théorème 2.2.2, la dynamique d'évolution de l'erreur linéarisée

$$\frac{d}{dt}\xi_t = A_t \xi_t + \widehat{w}_t \tag{2.35}$$

permet d'identifier A_t définie telle que $g_{u_t}(\,exp(\,\xi_t)\,) \,= \mathcal{L}_{\mathfrak{g}}(\,A_t\xi_t) + o(\,\|\xi_t\|^2)$ et $\widehat{w}_t$ le bruit de modèle modifié tel que :

$$\mathcal{L}_{\mathfrak{g}}(\widehat{w}_t) = -\widehat{\chi}_t w_t \widehat{\chi}_t^{-1}, \qquad \widehat{w}_t = -A_{d_{\chi_t}} {\mathcal{L}_{\mathfrak{g}}}^{-1}(w_t) \qquad \text{avec } \widehat{w}_t \in \mathbb{R}^{d\times 1} \text{ et } w_t \in \mathfrak{g} \tag{2.36}$$

$\widehat{Q}_t = A_{d_{\chi_t}} cov(w_t) {A_{d_{\chi_t}}}^{-1}$ est la matrice de covariance du bruit de modèle modifié $\widehat{w}_t \in \mathbb{R}^{d\times 1}$

et $A_{d_{\chi_t}}$ est l'opérateur adjoint défini sur l'algèbre de Lie $\mathfrak{g}$ tel que :

$$A_{d_{\chi_t}} : \begin{pmatrix} \mathfrak{g} & \rightarrow & \mathfrak{g} \\ \mathcal{L}_{\mathfrak{g}}(\xi) & \mapsto & A_{d_{\chi_t}}\left(\mathcal{L}_{\mathfrak{g}}(\xi)\right) = \chi_t \mathcal{L}_{\mathfrak{g}}(\xi) {\chi_t}^{-1} \end{pmatrix} \tag{2.37}$$

Étape de mise à jour d'un RIEKF : Les prédictions sont mises à jour via les observations invariantes à gauches (2.31) de sorte que :

$$\widehat{\chi}_{k|k} = exp\left(K_k\left(\widehat{\chi}_{k|k-1}Y - d\right)\right)\widehat{\chi}_{k|k-1}, \qquad P_{k|k} = \left(I_N - K_k H\right) P_{k|k-1} \tag{2.38}$$

avec K_k la matrice de gain de Kalman telle que $K_k = P_{k|k-1}H^T(HP_{k|k-1}H^T + \widehat{N}_k)^{-1}$, H la matrice d'observation telle que $H\xi_t = \mathcal{L}_{\mathfrak{g}}(\xi_t)d$ et $\widehat{N}_k$ la matrice de covariance du bruit de mesure modifié $\widehat{\chi}_k V_k$.

Comme l'IEKF satisfait le théorème 2.2.1 *(Dynamique d'erreur autonome)*, alors l'IEKF est un observateur localement asymptotiquement stable lorsque aucun bruit n'est considéré. La démonstration de stabilité est disponible dans [101] et des études de stabilités d'IEKF sont proposées dans [99]. De plus, la propriété *log-linéaire* de l'erreur (théorème 2.2.2) permet à la covariance estimée d'être propagée de manière cohérente. Cette propriété explique pourquoi un IEKF ne crée pas de fausses observabilités [100].

2.3 Filtre de Kalman Imparfait Invariant Étendu

2.3.1 Introduction à l'Imp.IEKF

Tous les modèles de navigation ne sont pas modélisés par une dynamique *affine de groupe*, nécessaire au développement d'un IEKF. En effet, la dynamique d'un corps en mouvement estimée à partir de mesures IMU imparfaites, c'est-à-dire lorsque les mesures sont par exemple altérées par des biais, ne satisfait pas cette propriété. Pour cause, aucun groupe de Lie matriciel ne permet de modéliser à la fois la dynamique du corps en mouvement et les imperfections des capteurs, tout en vérifiant la propriété de (2.19). Dans ce cas, il est préférable de concevoir un IEKF imparfait *(Imp.IEKF - Imperfect Invariant Extended Kalman Filter)*.

Comme présenté dans la figure 2.2, un Imp.IEKF est un filtre de Kalman non linéaire basé à la fois sur la théorie de l'IEKF et sur celle de l'EKF. Une partie des états à estimer est embarquée dans un groupe de Lie matriciel et associée à une erreur invariante.

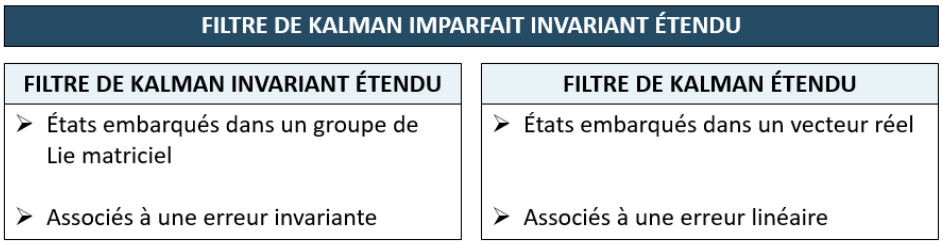

FIGURE 2.2 – Présentation générale d'un Imp.IEKF

L'autre partie des états est embarquée dans un vecteur réel auquel est associée une erreur linéaire. La concaténation et la linéarisation de ces erreurs permettent d'obtenir l'erreur d'estimation associée au système non linéaire et d'identifier les matrices jacobiennes de linéarisation. Ainsi, la théorie classique du filtre de Kalman est appliquée mais la mise à jour exponentielle est utilisée pour les états embarqués dans les groupes de Lie matriciels.

De nombreuses propriétés de l'IEKF ne sont pas vérifiées par l'Imp.IEKF car la propagation de l'erreur dépend désormais des états estimés, mais l'Imp.IEKF présente toutefois plusieurs avantages comparés à un EKF.

2.3.2 Équations d'un Imp.IEKF

Un Imp.IEKF estime les états d'un système dynamique non linéaire et les corrige par des observations modélisées de la façon suivante [99], [102], [104] :

$$\textit{Équation d'état} \quad \frac{d}{dt}\begin{bmatrix}\chi_t\\ \theta_t\end{bmatrix} = \begin{bmatrix}f_{(\theta,u_t)}(\chi_t) + w_{\chi_t}\\ g(\theta_t) + w_\theta\end{bmatrix} \tag{2.39}$$

$$\textit{Modèle d'observation} \quad y_n = h(\chi_t, \theta_t, w_y) \tag{2.40}$$

avec $\chi_t \in G$ la partie des états à estimer embarquée dans un groupe de Lie matriciel, $\theta \in \mathbb{R}^{p\times 1}$ l'autre partie des états à estimer embarquée dans un vecteur réel, et avec $f(.)$, $g(.)$ et $h(.)$ les modèles d'évolution et de mesures non linéaires. La dynamique $f(.)$ est *affine de groupe* uniquement pour des valeurs fixes de θ. Le bruit de modèle $w_t \sim \mathcal{N}(0, Q_t)$ est défini comme une concaténation de w_{χ_t} et de w_θ et $w_y \sim \mathcal{N}(0, R_t)$ représente le bruit de mesure.

L'erreur non linéaire associée au système (2.39) - (2.40) est définie telle que :

$$e_t = \begin{bmatrix} \eta_t \\ \widehat{\theta}_t - \theta_t \end{bmatrix} \tag{2.41}$$

avec η_t l'erreur invariante associée aux états χ_t conformément à la théorie des IEKF et $\epsilon_\theta = \widehat{\theta}_t - \theta_t$ l'erreur linéaire associée aux états θ_t conformément à la théorie des EKF.

Afin de développer la méthodologie d'un filtre de Kalman classique, l'erreur (2.41) doit être linéarisée. De bonnes estimations de l'état $\widehat{\chi}_t$ impliquent que η_t est proche de $I_{n\times n}$. Ainsi, localement autour de l'élément identité du groupe de Lie G et par un développement en série de Taylor de l'application exponentielle $exp_m(.)$, l'erreur invariante η_t est linéarisée de sorte que :

$$\eta_t = exp(\xi_t) = exp_m(\mathcal{L}_{\mathfrak{g}}(\xi_t)) \approx I_{n\times n} + \mathcal{L}_{\mathfrak{g}}(\xi_t) + o(\|{\xi_t}^2\|) \tag{2.42}$$

L'erreur linéarisée associée au système d'équations (2.39) - (2.40) est alors :

$$e_\xi = \begin{bmatrix} \xi_t \\ \widehat{\theta}_t - \theta_t \end{bmatrix} \tag{2.43}$$

La dynamique de l'erreur linéarisée permet d'identifier les matrices jacobiennes d'évolution comme des approximations au premier ordre de l'évolution de l'erreur telles que :

$$\frac{d}{dt}e_\xi = A_t(\widehat{\chi}_t, \widehat{\theta}_t)e_\xi + A_{d_\chi}(\widehat{\chi}_t, \widehat{\theta}_t)w_t + o(e_\xi) + o(w_t) \tag{2.44}$$

La matrice d'évolution linéarisée $A_t(\widehat{\chi}_t, \widehat{\theta}_t)$ est alors fonction des estimations comme la dynamique d'évolution n'est pas *affine de groupe*.

Étape de prédiction de l'Imp.IEKF

La prédiction des états est donnée, comme tous filtres de Kalman, par la dynamique d'évolution dans le cas où aucune perturbation n'est présente :

$$\frac{d}{dt}\begin{bmatrix} \chi_t \\ \theta_t \end{bmatrix} = \begin{bmatrix} f_{(\theta,u_t)}(\chi_t) \\ g(\theta_t) \end{bmatrix} \tag{2.45}$$

La prédiction de la matrice de covariance P_t de l'erreur linéarisée est donnée par

l'équation de Riccati suivante :

$$\frac{d}{dt}P_t = A_t(\widehat{\chi}_t, \widehat{\theta}_t)P_t + P_t A_t(\widehat{\chi}_t, \widehat{\theta}_t)^T + A_{d_\chi}(\widehat{\chi}_t, \widehat{\theta}_t)Q_t A_{d_\chi}(\widehat{\chi}_t, \widehat{\theta}_t)^T \tag{2.46}$$

avec les matrices $A_t(\widehat{\chi}_t, \widehat{\theta}_t)$ et $A_{d_\chi}(\widehat{\chi}_t, \widehat{\theta}_t)$ identifiées d'après la dynamique d'évolution de l'erreur linéarisée et Q_t la matrice de covariance du bruit de modèle w_t.

Étape de mise à jour de l'Imp.IEKF

Les prédictions obtenues à l'étape précédente sont mises à jour par des observations y_t modélisées par (2.40) :

$$y_t = h(\chi_t, \theta_t, w_y) \tag{2.47}$$

Le modèle d'observation linéarisé avec le respect de l'erreur linéarisée e_ξ est déterminé par une approximation de premier ordre [104] de sorte que :

$$h(x, w_y) - h(\widehat{x}, 0) = H(\widehat{\chi}_t, \widehat{\theta}_t)e_\xi + N(\widehat{\chi}_t, \widehat{\theta}_t)w_h + o(e_\xi) + o(w_y) \tag{2.48}$$

Conformément à la théorie classique des filtres de Kalman, la matrice de gain est définie par les équations suivantes :

$$\begin{gathered} S_k = H(\widehat{\chi}_t, \widehat{\theta}_t)P_{k|k-1}H(\widehat{\chi}_t, \widehat{\theta}_t)^T + N(\widehat{\chi}_t, \widehat{\theta}_t)RN(\widehat{\chi}_t, \widehat{\theta}_t)^T \\ K_k = P_{k|k-1}H(\widehat{\chi}_t, \widehat{\theta}_t)^T(S_k)^{-1} \end{gathered} \tag{2.49}$$

Suivant le type d'erreur invariante, les états et la matrice de covariance sont mis à jour d'après les équations suivantes :

$$\begin{bmatrix} \xi_{k|k} \\ \epsilon_{\theta_{k|k}} \end{bmatrix} = K_k(y_k - \widehat{y}_k) \tag{2.50}$$

$$\begin{aligned} \textit{Imp L-IEKF} \begin{bmatrix} \widehat{\chi}_{k|k} \\ \widehat{\theta}_{k|k} \end{bmatrix} &= \begin{bmatrix} \widehat{\chi}_{k|k-1}exp(\xi_k) \\ \widehat{\theta}_{k|k-1} + \epsilon_{\theta_{k|k}} \end{bmatrix} \\ \textit{Imp R-IEKF} \begin{bmatrix} \widehat{\chi}_{k|k} \\ \widehat{\theta}_{k|k} \end{bmatrix} &= \begin{bmatrix} exp(\xi_k)\widehat{\chi}_{k|k-1} \\ \widehat{\theta}_{k|k-1} + \epsilon_{\theta_{k|k}} \end{bmatrix} \end{aligned} \tag{2.51}$$

$$P_{k|k} = (I_{n\times n} - K_k H(\widehat{\chi}_t, \widehat{\theta}_t))P_{k|k-1} \tag{2.52}$$

2.3.3 Propriétés de l'Imp.IEKF

Comme l'Imp.IEKF ne satisfait pas la propriété *affine de groupe*, la dynamique d'évolution de l'erreur est dépendante des états estimés et donc l'Imp.IEKF ne partage pas exactement les mêmes propriétés que l'IEKF. Toutefois, du fait de la forme de l'erreur et de la mise à jour exponentielle, l'Imp.IEKF possède des caractéristiques importantes détaillées dans [99], [108].

Influence de l'erreur non linéaire : Lorsqu'un état x_1 d'un système est non observable, ce même état doit rester non observable pour le filtre de Kalman construit autour de la dynamique associée à x_1. Si aucune information n'est disponible pour évaluer x_1 alors la covariance d'erreur associée à cet état est grande.
Dans le cas d'un système non linéaire, les dynamiques sont linéarisées afin d'appliquer la méthodologie d'un filtre de Kalman. Si, à cause de la linéarisation, l'état x_1 devient observable, alors le filtre crée une fausse observabilité. Le filtre acquiert alors des informations inexistantes concernant l'état non observable du système et la matrice de covariance P_{x_1} associée à cet état est sous-évaluée et est donc extrêmement optimiste.
La linéarisation des dynamiques permet de construire le filtre de Kalman correspondant. La dynamique d'évolution de l'erreur permet d'identifier les matrices jacobiennes qui interviennent ensuite dans l'équation de Riccati qui propage la covariance de l'erreur. La covariance propage donc les propriétés d'observabilité du filtre linéarisé, donc la linéarisation peut créer une fausse observabilité. D'après les travaux présentés dans [99], une erreur non linéaire permet de corriger les problèmes de fausse observabilité d'un filtre. En effet, le choix d'une variable d'erreur influe sur les matrices jacobiennes et donc sur la covariance d'erreur. Une erreur non linéaire permet de propager la covariance afin de correspondre aux états non observables par le système. Ainsi, une erreur non linéaire permet de conserver les propriétés d'observabilité du système.

Remarque 2.3.1. *Dans le cas d'un IEKF, les matrices jacobiennes sont indépendantes des états estimés comme la dynamique des erreurs ne dépends pas des états. Ainsi, les propriétés d'observabilité du système linéarisé sont exactement les mêmes que celles du système dynamique. C'est pourquoi, un IEKF ne crée pas de fausse observabilité.*

Influence de la mise à jour exponentielle : Dans le cas d'un EKF, l'état et la covariance sont mis à jour dans l'espace tangent à l'état estimé précédent du fait de la mise à jour linéaire. Donc, comme présenté dans la figure 2.3, les estimations mises à jour évoluent dans un sous-espace qui ne correspond pas nécessairement à celui de l'état vrai.

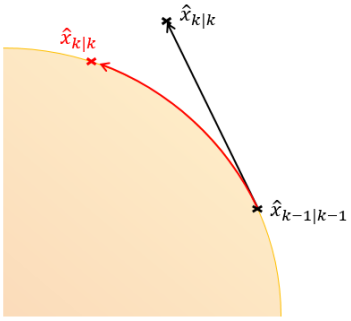

FIGURE 2.3 – État mis à jour par une application linéaire comme l'EKF (en noir). État mis à jour par une application exponentielle (en rouge).

D'après les résultats proposés dans [99], [100], les estimations d'un EKF basé sur une erreur non linéaire et une mise à jour exponentielle évoluent dans le même sous-espace que l'état vrai. En effet, la mise à jour exponentielle permet aux états estimés de correspondre à l'espace non euclidien dans lequel évolue l'état du système. Donc, un Imp.IEKF est adapté à des problèmes non linéaires évoluant dans des espaces non euclidiens.

2.4 Application de l'Imp.IEKF à la navigation de projectiles

Afin de vérifier les propriétés de l'Imp.IEKF énoncées précédemment, cette partie compare un Imp.RIEKF *(Imperfect Right-Invariant Extended Kalman Filter)* initialement proposé dans [82], [104] et un EKF pour la navigation des projectiles. Pour cela, les deux filtres sont dérivés à partir du même modèle d'évolution et d'observation. Les propriétés d'observabilités et la précision des estimations des deux filtres sont comparées.

2.4.1 Formulation du problème de navigation

Un Imp.RIEKF et un EKF sont implémentés afin d'estimer la trajectoire d'un projectile à partir de mesures inertielles. Pour cela, la trajectoire est prédite par les mesures bruitées et biaisées de l'accéléromètre et du gyromètre embarqué. Ces prédictions sont ensuite corrigées par la mesure du désalignement entre l'IMU et le centre de gravité du projectile.

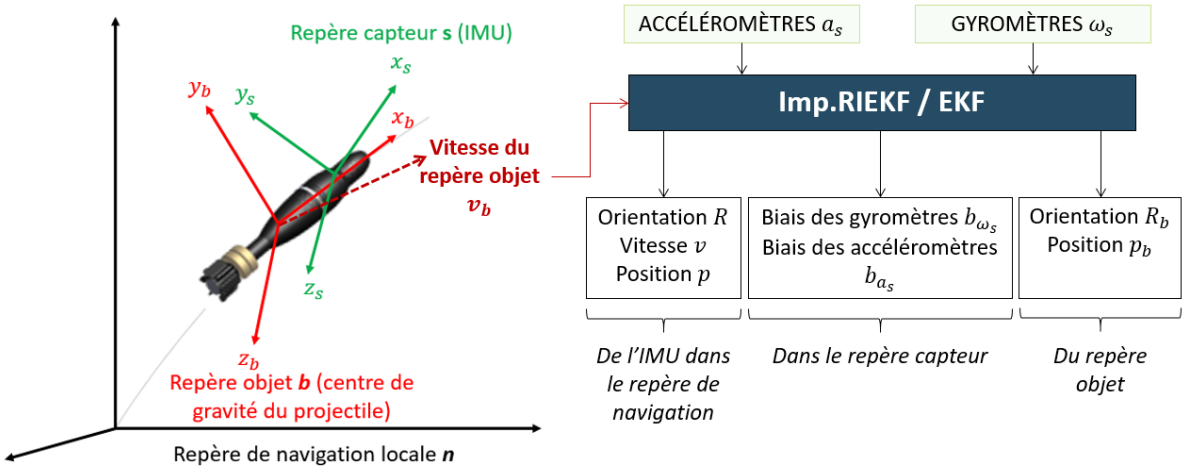

FIGURE 2.4 – Principe de fonctionnement de l'Imp.RIEKF et de l'EKF.

Comme présenté dans la figure 2.4, l'Imp.RIEKF et l'EKF visent à estimer, à partir des mesures bruitées et biaisées de l'accéléromètre et du gyromètre :

- l'orientation $R \in SO(3)$ du repère capteur $\boldsymbol{s}$ par rapport au repère local de navigation $\boldsymbol{n}$, la vitesse $v \in \mathbb{R}^{3\times1}$ et la position $p \in \mathbb{R}^{3\times1}$ du repère capteur $\boldsymbol{s}$ dans le repère local de navigation $\boldsymbol{n}$,
- le biais des gyromètres $b_{\omega_s} \in \mathbb{R}^{3\times1}$ et des accéléromètres $b_{a_s} \in \mathbb{R}^{3\times1}$ dans le repère capteur $\boldsymbol{s}$,
- l'orientation $R_b \in SO(3)$ et la position $p_b \in \mathbb{R}^{3\times1}$ du repère objet $\boldsymbol{b}$ par rapport au repère capteur $\boldsymbol{s}$. Ces deux grandeurs modélisent le désalignement entre l'IMU et le centre de gravité du projectile bien qu'une phase de calibration puisse être envisagée [114], [115]. Par conséquent, les états R_b et p_b sont associés à de faibles dynamiques.

Les mesures des accéléromètres et des gyromètres, effectuées dans le repère capteur $\boldsymbol{s}$ sont biaisées et bruitées. Le modèle de mesure de ces deux capteurs est alors :

$$\tilde{\omega}_s = \omega_s + b_{\omega_s} + W_{\omega_s}, \qquad \tilde{a}_s = a_s + b_{a_s} + W_{a_s}, \tag{2.53}$$

avec $\tilde{\omega}_s$ et $\tilde{a}_s \in \mathbb{R}^{3\times1}$ les modèles de mesure des gyromètres et des accéléromètres dans $\boldsymbol{s}$, ω_s et $a_s \in \mathbb{R}^{3\times1}$ les mesures des gyromètres et des accéléromètres dans $\boldsymbol{s}$, b_{ω_s} et $b_{a_s} \in \mathbb{R}^{3\times1}$ les biais des gyromètres et des accéléromètres dans $\boldsymbol{s}$, et W_{ω_s} et $W_{a_s} \in \mathbb{R}^{3\times1}$ les bruits des mesures supposés blancs Gaussien.

Le modèle d'évolution des biais des accéléromètres et des gyromètres est un mouvement brownien modélisé par les équations suivantes :

$$\dot{b}_{\omega_s} = W_{b_{\omega_s}}, \qquad \dot{b}_{a_s} = W_{b_{a_s}}, \tag{2.54}$$

avec $W_{b_{\omega_s}}$ et $W_{b_{a_s}} \in \mathbb{R}^{3\times 1}$ des bruits Gaussien de moyenne nulle.

La dynamique d'évolution de l'orientation R, de la vitesse v et de la position p du repère objet $\boldsymbol{b}$ dans le repère local de navigation $\boldsymbol{n}$ est :

$$\dot{R} = R[\tilde{\omega}_s - b_{\omega_s} - W_{\omega_s}]_\times, \tag{2.55}$$

$$\dot{v} = R(\tilde{a}_s - b_{a_s} - W_{a_s}) + g, \tag{2.56}$$

$$\dot{p} = v, \tag{2.57}$$

avec $[.]_\times$ l'opérateur linéaire du groupe de Lie $SO(3)$ et g le vecteur gravité supposé constant pour la durée de vol du projectile.
La dynamique de l'orientation R_b et de la position p_b du repère objet $\boldsymbol{b}$ par rapport au repère capteur $\boldsymbol{s}$ est approximativement constante et définie par les équations suivantes :

$$\dot{R}_b = R_b[W_{R_b}]_\times, \qquad \dot{p_b} = W_{p_b}, \tag{2.58}$$

avec W_{R_b} et W_{p_b} des bruits blancs supposés gaussiens.

L'observation considérée pour mettre à jour l'Imp.RIEKF et l'EKF est la vitesse relative du repère objet $\boldsymbol{b}$ par rapport au repère capteur $\boldsymbol{s}$, qui est nulle et modélisée par l'équation suivante :

$$v_b = {R_b}^T R^T v + [\tilde{\omega}_s - b_{\omega_s}]_\times p_b + w_y \in \mathbb{R}^{3\times 1}. \tag{2.59}$$

avec w_y le bruit de mesures.

2.4.2 Imp.RIEKF pour la navigation de projectiles

D'après les modèles présentés dans la partie *(2.4.1)*, l'Imp.RIEKF vise à estimer, à partir d'une configuration initiale donnée, les états suivants :

$$x \triangleq \Big(R, \quad v, \quad p, \quad b_{\omega_s}, \quad b_{a_s}, \quad R_b, \quad p_b\Big) \tag{2.60}$$

L'entrée de commande représente les mesures des capteurs inertiels telles que $u_t = \begin{bmatrix}\tilde{\omega}_s^T & \tilde{a}_s^T\end{bmatrix}^T \in \mathbb{R}^{6\times 1}$ et les observations considérées sont la vitesse du repère objet $\boldsymbol{b}$ par rapport au repère capteur $\boldsymbol{s}$.

Le bruit de modèle w_t associé à la dynamique d'évolution (2.54) - (2.58) est alors :

$$w_t = \begin{bmatrix}W_{\omega_s}^T & W_{a_s}^T & 0_{3\times 1}^T & W_{b_{\omega_s}}^T & W_{b_{a_s}}^T & W_{R_b}^T & W_{p_b}^T\end{bmatrix}^T \in \mathbb{R}^{21\times 1}. \tag{2.61}$$

Selon la méthodologie des Imp.IEKF [102], [104], une partie des états évolue dans un groupe de Lie matriciel, associée à une erreur invariante (à droite). Ainsi, l'orientation, la vitesse et la position du projectile sont représentées par une matrice du groupe de Lie matriciel $SE_2(3)$ car $R \in SO(3)$, $v \in \mathbb{R}^{3\times 1}$ et $p \in \mathbb{R}^{3\times 1}$. De même, l'orientation R_b du repère objet $\boldsymbol{b}$ est modélisée par un élément du groupe de Lie matriciel $SO(3)$ telle que :

$$\chi_t = \begin{bmatrix} R & v & p \\ 0_{1\times 3} & 1 & 0 \\ 0_{1\times 3} & 0 & 1 \end{bmatrix} \in SE_2(3), \qquad R_c \in SO(3). \tag{2.62}$$

Les biais b_{ω_s} et b_{a_s} des capteurs inertiels et la position p_b du repère objet $\boldsymbol{b}$ sont des éléments de $\mathbb{R}^{3\times 1}$.

Erreur d'estimation de l'Imp.RIEKF

Conformément à la méthodologie des Imp.IEKF, les erreurs associées aux états évoluant dans des groupes de Lie sont invariantes à droite alors que des erreurs linéaires sont associées aux états évoluant dans un espace euclidien. Ainsi, l'erreur non linéaire d'estimation de l'Imp.RIEKF est :

$$e = (\eta_{\chi t},\ \ \xi_{b_{\omega_s}},\ \ \xi_{b_{a_s}},\ \ \eta_{R_b},\ \ \xi_{p_b}) \tag{2.63}$$

avec $\xi_{b_{\omega_s}} = \widehat{b}_{\omega_s} - b_{\omega_s}$, $\xi_{b_{a_s}} = \widehat{b}_{a_s} - b_{a_s}$, $\xi_{p_b} = \widehat{p}_b - p_b \in \mathbb{R}^{3\times 1}$ respectivement les erreurs linéaires du biais des gyromètres, des accéléromètres et de la position p_b.

L'erreur invariante à droite $\eta_{\chi_t} \in SE_2(3)$ associée aux états (R, v, p) est par définition :

$$\eta_{\chi t} = \widehat{\chi}_t \chi_t^{-1} = \begin{bmatrix} \widehat{R} & \widehat{v} & \widehat{p} \\ 0_{1\times 3} & 1 & 0 \\ 0_{1\times 3} & 0 & 1 \end{bmatrix} \begin{bmatrix} R & v & p \\ 0_{1\times 3} & 1 & 0 \\ 0_{1\times 3} & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} \widehat{R}R^T & \widehat{v} - \widehat{R}R^T v & \widehat{p} - \widehat{R}R^T p \\ 0_{1\times 3} & 1 & 0 \\ 0_{1\times 3} & 0 & 1 \end{bmatrix} \tag{2.64}$$

L'erreur invariante à droite de l'orientation du repère R_b est :

$$\eta_{R_b} = \widehat{R}_b R_b^T \in SO(3) \tag{2.65}$$

Une bonne estimation de $\widehat{\chi}_t$ signifie que η_t est proche de $I_{5\times5}$. Ainsi, localement autour de l'élément neutre du groupe et en utilisant un développement en série de Taylor de premier ordre de l'application exponentielle $exp_m(.)$, l'erreur invariante $\widehat{\chi}_t$ est linéarisée telle que :

$$\begin{array}{rcccc} \eta_t & = & exp(\xi_t) = exp_m(\mathcal{L}_{\mathfrak{g}}(\xi_t)) & \approx & I_{5\times5} + \mathcal{L}_{\mathfrak{g}}(\xi_t) + o(\|{\xi_t}^2\|) \\ & = & \begin{bmatrix} \widehat{R}R^T & \widehat{v} - \widehat{R}R^T v & \widehat{p} - \widehat{R}R^T p \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix} & \approx & \begin{bmatrix} I_{3\times3} + [\xi_R]_\times & \xi_v & \xi_p \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix} \end{array} \tag{2.66}$$

Les erreurs linéarisées associées aux états (R, v, p) sont alors :

$$\eta_R = \widehat{R}R^T \approx I_{3\times3} + [\xi_R]_\times, \quad \eta_v = \widehat{v} - \widehat{R}R^T v \approx \xi_v, \quad \eta_p = \widehat{p} - \widehat{R}R^T p \approx \xi_p. \tag{2.67}$$

De même, l'erreur invariante d'orientation η_{R_b} est linéarisée telle que :

$$\eta_{R_b} = \widehat{R}_b R_b^T \approx I_{3\times3} + [\xi_{R_b}]_\times \tag{2.68}$$

L'erreur linéarisée de l'Imp.RIEKF est donc :

$$\xi = \begin{bmatrix} \xi_R^T & \xi_v^T & \xi_p^T & \xi_{b_{\omega_s}}^T & \xi_{b_{a_s}}^T & \xi_{R_b}^T & \xi_{p_b}^T \end{bmatrix}^T \in \mathbb{R}^{21\times1} \tag{2.69}$$

Étape de prédiction de l'Imp.RIEKF

La prédiction de l'état x (2.60) est donnée par le système d'évolution (2.54) - (2.58) dans le cas où aucune perturbation n'est présente :

$$\frac{d}{dt}\begin{bmatrix} \widehat{\chi}_t \\ \widehat{b}_{\omega_s} \\ \widehat{b}_{a_s} \\ \widehat{R}_b \\ \widehat{p}_b \end{bmatrix} = \begin{bmatrix} f_{u_t}(\widehat{\chi}_t) \\ 0_{3\times1} \\ 0_{3\times1} \\ 0_{3\times3} \\ 0_{3\times1} \end{bmatrix}, \text{ avec } f_{u_t}(\widehat{\chi}_t) = \begin{bmatrix} \widehat{R}[\tilde{\omega}_s - \widehat{b}_{\omega_s}]_\times & \widehat{R}(\tilde{a}_s - \widehat{b}_{a_s}) + g & \widehat{v} \\ 0_{1\times3} & 0 & 0 \\ 0_{1\times3} & 0 & 0 \end{bmatrix} \tag{2.70}$$

Remarque 2.4.1. *Contrairement à un IEKF classique, l'évolution $f_{u_t}(.)$ est fonction des états estimés.*

La prédiction de la matrice de covariance P_t est donnée par l'équation de Riccati suivante [82], [104] :

$$\frac{d}{dt}P_t = A_t P_t + P_t A_t^T + A_{d_\chi} Q_t A_{d_\chi}^T \tag{2.71}$$

avec Q_t la covariance du bruit de modèle w_t (2.61) et avec $A_t \in \mathbb{R}^{21\times 21}$ et $A_{d_\chi} \in \mathbb{R}^{21\times 21}$ les matrices jacobiennes déterminées par la dynamique d'erreur linéarisée telle que :

$$A_t = \begin{bmatrix} 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & -R & 0_{3\times3} & 0_{6\times3} \\ [g]_\times & 0_{3\times3} & 0_{3\times3} & -[v]_\times R & -R & 0_{6\times3} \\ 0_{3\times3} & I_{3\times3} & 0_{3\times3} & -[p]_\times R & 0_{3\times3} & 0_{6\times3} \\ & 0_{12\times9} & & & 0_{12\times12} & \end{bmatrix}, \; A_{d_\chi} = \begin{bmatrix} R & 0_{3\times3} & 0_{3\times3} & 0_{12\times3} \\ [v]_\times R & R & 0_{3\times3} & 0_{12\times3} \\ [p]_\times R & 0_{3\times3} & R & 0_{12\times3} \\ & 0_{12\times9} & & I_{12\times12} \end{bmatrix}. \tag{2.72}$$

Remarque 2.4.2. *La démonstration de l'identification des matrices A_t et A_{d_χ} est disponible dans l'Annexe A.*

Étape de prédiction de l'Imp.RIEKF à temps discret

La prédiction de l'état x à la période d'échantillonage des capteurs Δt est :

$$\begin{aligned} \widehat{R}_{k|k-1} &= \widehat{R}_{k-1|k-1} exp_{SO(3)}([\tilde{\omega}_{s_k} - \widehat{b}_{\omega_{s,k-1|k-1}}]_\times \Delta t) \\ \widehat{v}_{k|k-1} &= \widehat{v}_{k-1|k-1} + (\widehat{R}_{k-1|k-1}(\tilde{a}_{s_k} - \widehat{b}_{a_{s,k-1|k-1}}) + g)\Delta t, \\ \widehat{p}_{k|k-1} &= \widehat{p}_{k-1|k-1} + \widehat{v}_{k-1|k-1}\Delta t + \tfrac{1}{2}(\widehat{R}_{k-1|k-1}(\widehat{R}_{k-1|k-1}(\tilde{a}_{s_k} - \widehat{b}_{a_{s,k-1|k-1}}) + g)\Delta t^2 \end{aligned} \tag{2.73}$$

$$\widehat{b}_{\omega_{s,k|k-1}} = \widehat{b}_{\omega_{s,k-1|k-1}}, \quad \widehat{b}_{a_{s,k|k-1}} = \widehat{b}_{a_{s,k-1|k-1}}, \quad \widehat{R}_{b_{k|k-1}} = \widehat{R}_{b_{k-1|k-1}}, \quad \widehat{p}_{b_{k|k-1}} = \widehat{p}_{b_{k-1|k-1}} \tag{2.74}$$

La prédiction de la matrice de covariance de l'erreur à temps discret est donnée par :

$$P_{k|k-1} = \Phi_k P_{k-1|k-1} \Phi_k^T + Q_k. \tag{2.75}$$

avec $\Phi_k = exp_m(A_t \Delta t)$ et $Q_k = \Phi_k \left(A_{d_\chi} Q_t A_{d_\chi}^T \right) \Phi_k^T \Delta t$ avec Q_t la matrice de covariance du bruit de modèle w_t (2.61).

Étape de mise à jour de l'Imp.RIEKF

Les prédictions obtenues à l'étape précédente sont mises à jour par la mesure de la vitesse du repère objet $\boldsymbol{b}$ par rapport au repère capteur $\boldsymbol{s}$ modélisée telle que :

$$v_b = {R_b}^T R^T v + [\tilde{\omega}_s - b_{\omega_s}]_\times p_b + w_y \in \mathbb{R}^{3\times 1} \tag{2.76}$$

La mesure de v_b ne peut pas être formulée comme une observation invariante à droite donc, contrairement à un IEKF, le terme de correction de l'Imp.RIEKF est calculé par addition vectorielle standard telle que $K(v_b - \widehat{v}_b)$.

La linéarisation du modèle d'observation (2.76) avec le respect de l'erreur linéarisée ξ_t (2.69) permet d'identifier la matrice d'observation H et la covariance du bruit de mesures N. Le gain de Kalman est alors évalué tel que :

$$K_k = P_{k|k-1}H_k^T(H_k P_{k|k-1} H_k^T + N_k cov(w_y) N_k^T)^{-1} \tag{2.77}$$

avec $N_k = I_{3\times 3}$ et H la matrice d'observation définie telle que :

$$H = \begin{bmatrix} 0_{3\times 3} & R_b^T R & 0_{3\times 3} & [p_b]_\times & 0_{3\times 3} & R_b^T[R^T v]_\times & -[\tilde{\omega}_s - b_{\omega_s}]_\times \end{bmatrix} \in \mathbb{R}^{3\times 21}. \tag{2.78}$$

Le calcul de la matrice de gain de Kalman et des observations permettent de mettre à jour les prédictions telles que :

$$\begin{bmatrix} \xi_{Rk} & \xi_{vk} & \xi_{pk} & \xi_{b_{\omega_s}k} & \xi_{b_{a_s}k} & \xi_{R_bk} & \xi_{p_bk} \end{bmatrix}^T = K_k(0_{3\times 1} - \widehat{v}_{b_k}) \tag{2.79}$$

De plus, d'après la méthodologie des Imp.IEKF, $\widehat{\chi}_{k|k} = exp_{SE_2(3)}\left(\begin{bmatrix} {\xi_{R}}_k^T & {\xi_{v}}_k^T & {\xi_{p}}_k^T \end{bmatrix}^T\right)\widehat{\chi}_{k|k-1}$, donc les états estimés et la covariance estimée par l'Imp.RIEKF sont :

$$\widehat{R}_{k|k} = \xi_{R_{k|k}}\widehat{R}_{k|k-1}, \quad \widehat{v}_{k|k} = \xi_{v_{k|k}} + \xi_{R_{k|k}}\widehat{v}_{k|k-1}, \quad \widehat{p}_{k|k} = \xi_{p_{k|k}} + \xi_{R_{k|k}}\widehat{p}_{k|k-1} \tag{2.80}$$

$$\widehat{b}_{\omega_{s\,k|k}} = \widehat{b}_{\omega_{s\,k|k-1}} + \xi_{b_{\omega_s k}}, \qquad \widehat{b}_{a_{s\,k|k}} = \widehat{b}_{a_{s\,k|k}} + \xi_{b_{a_s k}}, \tag{2.81}$$

$$\widehat{R}_{b_{k|k}} = exp_{SO(3)}(\xi_{R_{b_{k|k}}})\widehat{R}_{b_{k|k-1}}, \qquad \widehat{p}_{b_{k|k}} = \xi_{p_{b_k}} + \widehat{p}_{b_{k|k-1}} \tag{2.82}$$

$$P_{k|k} = (I_{21\times 21} - K_k H)P_{k|k-1} \tag{2.83}$$

2.4.3 EKF pour la navigation de projectiles

Afin de visualiser l'intérêt d'un Imp.RIEKF pour la navigation des projectiles, un EKF est dérivé en utilisant le même modèle que celui employé pour l'Imp.RIEKF.

D'après les modèles présentés dans la partie *(2.4.1)*, l'EKF vise à estimer, à partir d'une configuration initiale donnée, les états suivants :

$$x \triangleq \Big(q, \quad v, \quad p, \quad b_{\omega_s}, \quad b_{a_s}, \quad q_b, \quad p_b\Big) \tag{2.84}$$

avec q et q_b les quaternions associés aux matrices d'orientation R et R_b qui représentent respectivement l'orientation du repère capteur $\boldsymbol{s}$ et l'orientation du repère objet $\boldsymbol{b}$.
Comme pour l'Imp.RIEKF, l'entrée de commande représente les mesures des capteurs inertiels définies par l'équation (2.53) et les observations considérées sont la vitesse du repère objet $\boldsymbol{b}$ par rapport au repère capteur $\boldsymbol{s}$ (2.59).

La discrétisation des équations (2.54) - (2.58) et la dynamique associée aux quaternions q et q_c par la méthode d'Euler à la période d'échantillonnage $\Delta_t = t_{k+1} - t_k$ permet d'obtenir le modèle discret d'évolution associé aux états x (2.84) :

$$x_{k+1} = f(x_k, u_k) + w_k \tag{2.85}$$

$$\begin{bmatrix} q_{k+1} \\ v_{k+1} \\ p_{k+1} \\ b_{\omega_{s\,k+1}} \\ b_{a_{s\,k+1}} \\ q_{b_{k+1}} \\ p_{b_{k+1}} \end{bmatrix} = \begin{bmatrix} \Big(I_{4\times4} + \frac{\Delta_t}{2}\Omega(\tilde{\omega}_{s_k} - b_{\omega_{s_k}})\Big)\, q_k \\ v_k + (R_k(\tilde{a}_{s_k} - b_{a_{s_k}}) + g)\Delta_t \\ p_k + v_k\Delta_t \\ b_{\omega_{s\,k}} \\ b_{a_{s\,k}} \\ q_{b_k} \\ p_{b_k} \end{bmatrix} + \begin{bmatrix} -\frac{\Delta_t}{2}\mathbb{E}(q_k)W_{\omega_{s\,k}} \\ -R_k W_{a_{s\,k}}\Delta_t \\ 0_3 \\ W_{b_{\omega_s}}\Delta_t \\ W_{b_{a_s}}\Delta_t \\ \frac{\Delta_t}{2}\mathbb{E}(q_{b_k})W_{R_b} \\ W_{p_b}\Delta_t \end{bmatrix} \tag{2.86}$$

avec $x_k \in \mathbb{R}^{23\times1}$ les états à estimer par l'EKF, $f(.)$ le modèle d'évolution et w_k les bruits de modèle.

Étape de prédiction de l'EKF

L'étape de prédiction de l'EKF est donnée par les équations suivantes :

$$x_{k|k-1} = f(x_{k-1|k-1}, u_k) \tag{2.87}$$

$$P_{k|k-1} = \phi_k P_{k-1|k-1}\phi_k^T + Q_k \tag{2.88}$$

avec ϕ_k la matrice jacobienne du modèle d'évolution $f(.)$ en temps discret définie telle que :

$$\phi_k = I_{23\times 23} + \begin{bmatrix} \frac{1}{2}\Omega(\tilde{\omega_s}_k - b_{\omega_s k}) & 0_{4\times 3} & 0_{4\times 3} & -\frac{1}{2}\mathbb{E}(q_k) & 0_{4\times 3} & 0_{4\times 7} \\ \frac{\partial}{\partial q}(q \otimes (\tilde{a}_s - b_{a_s}) \otimes q^*) & 0_3 & 0_3 & 0_3 & -R_k & 0_{3\times 7} \\ 0_{3\times 4} & I_3 & 0_3 & 0_3 & 0_3 & 0_{3\times 7} \\ & & 0_{13\times 23} & & & \end{bmatrix} \Delta_t \quad (2.89)$$

et $Q_k = w_k w_k^T$ la matrice de covariance du bruit de modèle w_k (2.86) et $\frac{\partial}{\partial q}(q \otimes a \otimes q^*)$ défini dans l'annexe *B*.

Remarque 2.4.3. *La dynamique d'évolution du quaternion et la matrice jacobienne ϕ_k sont présentées respectivement dans les annexes C.1. et C.2. de ce document.*

Étape de mise à jour de l'EKF

Comme pour l'Imp.RIEKF, les observations considérées pour mettre à jour les prédictions de l'EKF sont :

$$v_b = R_b^T R^T v + [\tilde{\omega}_s - b_{\omega_s}]_\times p_b + w_y \quad (2.90)$$

Les états et la covariance sont alors mis à jour d'après la méthodologie standard d'un EKF tel que :

$$K_k = P_{k|k-1} H_k^T \left(H_k P_{k|k-1} H_k^T + R_k \right)^{-1} \quad (2.91)$$

$$\widehat{x}_{k|k} = \widehat{x}_{k|k-1} + K_k(0_{3\times 1} - \widehat{v}_{b_k}) \quad (2.92)$$

$$P_{k|k} = (I - K_k H_k) P_{k|k-1} \quad (2.93)$$

avec R_k la matrice de covariance du bruit de mesure et H_k la matrice jacobienne du modèle d'observation de sorte que :

$$H = \frac{\partial h}{\partial x}|_x = \begin{bmatrix} R_b^T \frac{\partial}{\partial q}(q^* \otimes v \otimes q) & R_b^T R^T & 0_{3\times 3} & [p_b]_\times & 0_{3\times 3} & \frac{\partial}{\partial q_b}(q_b^* \otimes R^T v \otimes q_b) & [\tilde{\omega}_s - b_{\omega_s}]_\times \end{bmatrix} \quad (2.94)$$

avec les opérations de quaternions définies dans l'annexe *B*.

2.4.4 Étude d'observabilité

Les propriétés d'observabilité de l'EKF et de l'Imp. RIEKF sont étudiées afin de vérifier la cohérence des deux filtres.
Pour cela, les états observables du système non linéaire (2.54) - (2.58) sont d'abord identifiés puis comparés aux états observables par l'Imp.RIEKF et l'EKF afin de conclure sur la cohérence des deux filtres [28], [31].

Analyse d'observabilité du système non linéaire à temps continu

La première étape de l'analyse de l'observabilité est l'identification des états non observables du système non linéaire en temps continu (2.54) - (2.58) utilisés pour dériver l'EKF et l'Imp.IEKF.
Si le système non linéaire a un ou plusieurs états non observables, alors ces mêmes états devraient être non observables pour les filtres dérivés à partir de ce système. Sinon, cela signifie que les filtres créent de fausses observabilités.

Soit le système non linéaire en temps continu :

$$\begin{aligned} \dot{x} &= f(x,u) \\ z &= h(x) \end{aligned} \tag{2.95}$$

avec $f(.)$ et $h(.)$ les modèles d'évolution et de mesure non linéaires, et u les entrées.

Définition 2.4.1 (**Observabilité d'un système non linéaire [27]**). *L'observabilité du système non linéaire* (2.95) *est déterminée par la dimension de l'espace couvert par les gradients des dérivées de Lie* $\nabla \mathcal{L}_f^i h(x)$.
La matrice d'observabilité $O(x,u)$ *du système* (2.95) *est définie comme la matrice composée des gradients des dérivées de Lie tels que :*

$$O(x,u) = \begin{bmatrix} \nabla \mathcal{L}_f^0 h(x) \\ \nabla \mathcal{L}_f^1 h(x) \\ \dots \\ \nabla \mathcal{L}_f^k h(x) \end{bmatrix} \quad avec \quad \begin{array}{c} \mathcal{L}_f^0 h(x) = h(x), \\ \mathcal{L}_f^k h(x) = (\nabla \mathcal{L}_f^{k-1} h(x)) f(x), \\ \nabla \mathcal{L}_f^k h(x) = \frac{\partial \mathcal{L}_f^k h(x)}{\partial x} \end{array} \tag{2.96}$$

Le système (2.95) *est observable lorsque* $O(x,u)$ *satisfait la condition de rang, c'est-à-dire lorsque* $O(x,u)$ *est de rang plein. Dans le cas d'un système inobservable, l'analyse de l'espace nul de la matrice* $O(x,u)$ *permet de déterminer les états inobservables.*

Définition 2.4.2 (**Identification des états non observables**). *L'espace nul de la matrice d'observabilité $O(x,u)$ permet de déterminer les états non observables du système [116]. L'espace nul $U_c(x,u)$ de la matrice d'observabilité du système non-linéaire $O(x,u)$ est donné par :*

$$O(x,u)U_c(x,u) = 0 \tag{2.97}$$

Le modèle d'évolution non linéaire en temps continu utilisé pour dériver l'EKF et l'Imp.IEKF est défini par les équations (2.54) - (2.58). Les termes de bruit sont omis car ils n'influencent pas l'analyse d'observabilité.
Afin d'alléger les calculs, ces équations sont définies dans le repère capteur $\boldsymbol{s}$ et les matrices de rotation R et R_b sont représentées à partir de leurs vecteurs de rotation $\phi \in \mathbb{R}^{3\times1}$ et $\phi_b \in \mathbb{R}^{3\times1}$ d'après la méthodologie présentée dans [117], [118] :

$$\begin{aligned} R &= exp(\phi), \\ R_b &= exp(\phi_b) \end{aligned} \tag{2.98}$$

Donc, le modèle d'évolution (2.54) - (2.58) devient :

$$\begin{gathered} \dot{\phi} = R(\tilde{\omega}_s - b_{\omega_s}), \\ (\dot{R^T v}) = [\tilde{\omega}_s - b_{\omega_s}]_\times R^T v + (\tilde{a}_s - b_{a_s}) + R^T g, \\ (\dot{R^T p}) = [\tilde{\omega}_s - b_{\omega_s}]_\times R^T p + R^T v \\ \dot{b}_{\omega_s} = 0_{3\times1}, \qquad \dot{b}_{a_s} = 0_{3\times1}, \qquad \dot{\phi}_b = 0_{3\times1}, \qquad \dot{p}_b = 0_{3\times1} \end{gathered} \tag{2.99}$$

Les notations suivantes sont introduites :

$$\begin{aligned} \underline{v} &= R^T v, & \underline{p} &= R^T p, \\ \underline{\omega} &= \tilde{\omega}_s - b_{\omega_s}, & \underline{a} &= \tilde{a}_s - b_{a_s}. \end{aligned} \tag{2.100}$$

Le modèle d'évolution et de mesure non linéaire considéré est le suivant :

$$\begin{aligned} \dot{\underline{x}} = \begin{bmatrix} \phi^T & \underline{v}^T & \underline{p}^T & b_{\omega_s}^T & b_{a_s}^T & \phi_b^T & p_b^T \end{bmatrix}^T &= f(\underline{x},\underline{u}) \\ &= \begin{bmatrix} R\underline{\omega} \\ [\underline{\omega}]_\times \underline{v} + \underline{a} + R^T g \\ [\underline{\omega}]_\times \underline{p} + \underline{v} \\ 0_{12\times1} \end{bmatrix} \end{aligned} \tag{2.101}$$

$$z = h(\underline{x}) = R_b^T \underline{v} + [\underline{\omega}]_\times p_b \tag{2.102}$$

D'après la définition *(2.4.1)*, la matrice d'observabilité du système non linéaire (2.101) - (2.102) est :

$$O(\underline{x},\underline{u}) = R_b^T \begin{bmatrix} 0 & I_3 & 0 & R_b[p_b]_\times & 0 & [\underline{v}]_\times & R_b[\underline{\omega}]_\times \\ R^T[g]_\times & [\underline{\omega}]_\times & 0 & [\underline{v}]_\times & -I_3 & [[\underline{\omega}]_\times \underline{v} + \underline{a} + R^T g]_\times & 0 \\ 0 & [\underline{\omega}]_\times^2 & 0 & \delta_{1,2} & -[\underline{\omega}]_\times & [\underline{\omega}]_\times [[\underline{\omega}]_\times \underline{v} + \underline{a})]_\times & 0 \\ [\underline{\omega}]_\times^2 R^T[g]_\times & [\underline{\omega}]_\times^3 & 0 & \delta_{2,3} & -[\underline{\omega}]_\times^2 & [\underline{\omega}]_\times^2 [[\underline{\omega}]_\times \underline{v} + \underline{a} + R^T g]_\times & 0 \\ 0 & [\underline{\omega}]_\times^4 & 0 & \delta_{1,4} & -[\underline{\omega}]_\times^3 & [\underline{\omega}]_\times^3 ([\underline{\omega}]_\times \underline{v} + \underline{a}) & 0 \\ [\underline{\omega}]_\times^4 R^T[g]_\times & [\underline{\omega}]_\times^5 & 0 & \delta_{2,5} & -[\underline{\omega}]_\times^4 & [\underline{\omega}]_\times^4 [[\underline{\omega}]_\times \underline{v} + \underline{a} + R^T g]_\times & 0 \\ 0 & [\underline{\omega}]_\times^6 & 0 & \delta_{1,6} & -[\underline{\omega}]_\times^5 & [\underline{\omega}]_\times^5 ([\underline{\omega}]_\times \underline{v} + \underline{a}) & 0 \end{bmatrix} \tag{2.103}$$

avec :

$$\delta_{1,i} = \frac{\partial}{\partial b_{\underline{\omega}}} [\underline{\omega}]_\times^{i-1} ([\underline{\omega}]_\times \underline{v} + \underline{a}) \qquad \delta_{2,i} = \frac{\partial}{\partial b_{\underline{\omega}}} [\underline{\omega}]_\times^{i-1} ([\underline{\omega}]_\times \underline{v} + \underline{a} + R^T g) \tag{2.104}$$

La matrice d'observabilité $O(\underline{x},\underline{u})$ (2.103) n'est pas une matrice de rang complet, ainsi la condition de rang n'est pas satisfaite et donc le système non linéaire (2.101) - (2.102) n'est pas complètement observable.

D'après la définition *(2.4.1)*, l'espace nul de la matrice d'observabilité $O(\underline{x},\underline{u})$ permet d'identifier les états inobservables du système non linéaire (2.101) - (2.102). L'espace nul de la matrice d'observabilité $O(\underline{x},\underline{u})$, noté $U_c(\underline{x},\underline{u})$ est donné par :

$$U_c(\underline{x},\underline{u}) = \begin{bmatrix} g & 0_{3\times1} & 0_{3\times1} \\ 0_{3\times1} & 0_{3\times1} & 0_{3\times1} \\ 0_{3\times1} & I_{3\times1} & 0_{3\times1} \\ 0_{3\times1} & 0_{3\times1} & 0_{3\times1} \\ 0_{3\times1} & 0_{3\times1} & 0_{3\times1} \\ 0_{3\times1} & 0_{3\times1} & 0_{3\times1} \\ 0_{3\times1} & 0_{3\times1} & R_c\underline{\omega} \end{bmatrix} \tag{2.105}$$

D'après la première colonne de $U_c(\underline{x},\underline{u})$, la rotation autour de l'axe de gravité (angle de lacet ψ) n'est pas observable. La seconde colonne de $U_c(\underline{x},\underline{u})$ indique que la position p n'est pas observable. Conformément à la troisième colonne de $U_c(\underline{x},\underline{u})$, la position p_b n'est pas observable avec une seule mesure $\underline{\omega}$. La position p_b devient observable lorsque plusieurs mesures différentes sont obtenues.

Selon l'analyse d'observabilité, le système non linéaire (2.54) - (2.58) a trois degrés de liberté non observables ; ψ, p et p_b (lorsque ω est constant). Ainsi, lorsque le filtre de Kalman est utilisé pour l'estimation, les mêmes propriétés d'observabilité doivent être constatées. Si, lors de l'étape de linéarisation, la matrice d'observabilité du filtre gagne en rang à cause d'erreurs dans les estimations des états, alors le filtre crée une fausse observabilité.

Analyse d'observabilité de l'Imp.RIEKF et l'EKF

Le modèle d'évolution de l'erreur linéarisée d'un filtre de Kalman non linéaire varie dans le temps donc la matrice d'observabilité locale est utilisée pour effectuer l'analyse d'observabilité [28].
L'analyse de la matrice d'observabilité locale permet de conclure sur les états observables par un filtre de Kalman non linéaire. Il est ainsi facile de vérifier que les états observables par le filtre soient les mêmes que les états observables du système non linéaire.

Définition 2.4.3 (**Observabilité locale [119]**). *Soit le modèle discret d'un filtre de Kalman non linéaire :*

$$x_{k+1} = f(x_k, u_k) + w_x \tag{2.106}$$
$$y_k = h(x_x) + w_y \tag{2.107}$$

avec w_x et w_y les bruits de modèle et de mesure, et avec Φ_k et H_k les matrices jacobiennes à temps discret des modèles linéarisés d'évolution $f(.)$ et de mesure $h(.)$.
Le système (2.106) - (2.107) *est localement observable si l'état $x(k)$ peut être déterminé à partir de la connaissance de $y(j)$ pour $j \in [k, k+n-1]$, où n est l'ordre du système.*

Définition 2.4.4 (**Matrice d'observabilité locale**). *La matrice d'observabilité locale M est définie comme une fonction du modèle de mesure linéarisé H_k et du modèle d'évolution linéarisé Φ_k, fonction du point de linéarisation tel que :*

$$M = \begin{bmatrix} H_k \\ H_{k+1}\Phi_k \\ H_{k+2}\Phi_{k+1}\Phi_k \\ ... \\ H_{k+n}\Phi_{k+n-1}...\Phi_k \end{bmatrix} \tag{2.108}$$

Le système (2.106) - (2.107) *est observable localement sur la période de temps* $[k, k+n-1]$*, si et seulement si la matrice d'observabilité locale* M *est de rang* n *pour la période* $[k, k+n-1]$*. Dans le cas contraire, l'étude de l'espace nul de* M *permet de déterminer les directions non observables par le filtre.*

Remarque 2.4.4. *La matrice d'observabilité locale* M *dépend des points de linéarisation utilisés pour calculer les jacobiennes. Ainsi, le choix des points de linéarisation affecte les propriétés d'observabilité du filtre [28].*

Il a été démontré que le système physique a trois degrés de liberté non observables qui sont l'angle de lacet ϕ, la position p et la position p_b lorsque ω est constant. Ainsi, l'Imp.RIEKF doit partager les mêmes propriétés d'observabilité.

D'après la définition *2.4.3*, la matrice d'observabilité locale de l'Imp.RIEKF est :

$$M_{Imp.RIEKF} = \begin{bmatrix} 0 & R_k^T & 0 & -R_{b_k}[p_{b_k}]_\times & 0 & [R_k^T v_k]_\times & \delta_{0.7} \\ \delta_{1.1} & \delta_{1.2} & 0 & \delta_{1.4} & \delta_{1.5} & \delta_{1.6} & \delta_{1.7} \\ \delta_{2.1} & \delta_{2.2} & 0 & \delta_{2.4} & \delta_{2.5} & \delta_{2.6} & \delta_{2.7} \\ \delta_{3.1} & \delta_{3.2} & 0 & \delta_{3.4} & \delta_{3.5} & \delta_{3.6} & \delta_{3.7} \\ \delta_{4.1} & \delta_{4.2} & 0 & \delta_{4.4} & \delta_{4.5} & \delta_{4.6} & \delta_{4.7} \\ \delta_{5.1} & \delta_{5.2} & 0 & \delta_{5.4} & \delta_{5.5} & \delta_{5.6} & \delta_{5.7} \\ \delta_{6.1} & \delta_{6.2} & 0 & \delta_{6.4} & \delta_{6.5} & \delta_{6.6} & \delta_{6.7} \end{bmatrix} \tag{2.109}$$

avec $\delta_{i.1} = iR_{k+i}^T[g]_\times\Delta_t$, $\delta_{i.2} = R_{k+i}^T$, $\delta_{i.5} = -\sum_{j=0}^{i-1} R_{k+j}$, $\delta_{i.6} = [R_{k+i}^T v_{k+i}]_\times$,
$\delta_{i.4} = -R_{k+i}^T\left(\sum_{j=0}^{i-1}[v_{k+j}]_\times R_{k+j}\Delta_t + \sum_{j=0}^{i-1}\frac{2(i-j)-1}{2}[g]_\times R_{k+j}\Delta_t^2\right)$, $\delta_{i.7} = -R_{b_{k+i}}[\tilde{\omega}_{s_{k+i}} - b_{\omega_{sk+i}}]_\times$.

La matrice d'observabilité locale de l'Imp.RIEKF $M_{Imp.RIEKF}$ n'est pas de rang complet, donc plusieurs états ne sont pas observables par le filtre.
La première colonne de $M_{Imp.RIEKF}$ (2.109) est $\delta_{i.1} = iR_{k+i}^T[g]_\times\Delta_t$, et la troisième colonne de $[g]_\times$ est toujours nulle donc l'angle de lacet ψ est non observable. La troisième colonne de $M_{Imp.RIEKF}$ est nulle donc la position p est non observable. D'après la dernière colonne de $M_{Imp.RIEKF}$, lorsque $\tilde{\omega}_s$ varie dans le temps, la position p_b est entièrement observable et donc la dernière colonne est de rang complet. Lorsque $\tilde{\omega}_s$ est constant pendant au moins 7 pas de temps consécutifs, la position p_b devient non observable (cas rare).
Par conséquent, l'Imp.RIEKF est cohérent avec le système non linéaire car il partage les mêmes propriétés d'observabilité que le système non linéaire donc l'Imp.RIEKF ne crée pas de fausse observabilité.

D'après la définition *2.4.3*, la matrice d'observabilité locale de l'EKF est :

$$M_{EKF} = \begin{bmatrix} \delta_{1.1} & R_{b_k}^T R_k^T & 0 & [p_{b_k}]_\times & 0 & \delta_{1.6} & [\tilde{\omega}_{s_k} - b_{\omega_{s_k}}]_\times \\ \delta_{2.1} & R_{b_{k+1}}^T R_{k+1}^T & 0 & \delta_{2,4} & \delta_{2,5} & \delta_{2.6} & [\tilde{\omega}_{s_{k+1}} - b_{\omega_{s_{k+1}}}]_\times \\ \delta_{3.1} & R_{b_{k+2}}^T R_{k+2}^T & 0 & \delta_{3,4} & \delta_{3,5} & \delta_{3.6} & [\tilde{\omega}_{s_{k+2}} - b_{\omega_{s_{k+2}}}]_\times \\ \delta_{4.1} & R_{b_{k+3}}^T R_{k+3}^T & 0 & \delta_{4,4} & \delta_{4,5} & \delta_{4.6} & [\tilde{\omega}_{s_{k+3}} - b_{\omega_{s_{k+3}}}]_\times \\ \delta_{5.1} & R_{b_{k+4}}^T R_{k+4}^T & 0 & \Delta_{5,4} & \delta_{5,5} & \delta_{5.6} & [\tilde{\omega}_{s_{k+4}} - b_{\omega_{s_{k+4}}}]_\times \\ \delta_{6.1} & R_{b_{k+5}}^T R_{k+5}^T & 0 & \Delta_{6,4} & \delta_{6,5} & \delta_{6.6} & [\tilde{\omega}_{s_{k+5}} - b_{\omega_{s_{k+5}}}]_\times \\ \delta_{7.1} & R_{b_{k+6}}^T R_{k+6}^T & 0 & \Delta_{7,4} & \delta_{7,5} & \delta_{7.6} & [\tilde{\omega}_{s_{k+6}} - b_{\omega_{s_{k+6}}}]_\times \end{bmatrix} \tag{2.110}$$

avec $\delta_{i,j}$ des fonctions de :

$$R_{b_{k+i}}^T \frac{\partial}{\partial q_{k+i}} \left(q_{k+i}^* \otimes v_{k+i} \otimes q_{k+i} \right), \quad \frac{\partial}{\partial q_{b_{k+i}}} \left(q_{b_{k+i}}^* \otimes R_{k+i}^T v_{k+i} \otimes q_{b_{k+i}} \right), \tag{2.111}$$

$$\frac{\partial}{\partial q_{k+i}} \left(q_{k+i} \otimes (\tilde{a}_{s_{k+i}} - b_{a_{s_{k+i}}}) \otimes q_{k+i}^* \right) \Delta t, \quad R_{b_{k+i}} R_{k+i}^T, \quad \mathbb{E}(q_{k+i}), \quad \Omega(\omega_{s_{k+i}} - b_{\omega_{s_{k+i}}}) \tag{2.112}$$

La matrice M_{EKF} (2.110) n'est pas de rang complet, donc plusieurs états ne sont pas observables par le filtre. La troisième colonne de M_{EKF} est nulle donc la position p n'est pas observable par l'EKF. De plus, d'après la dernière colonne de M_{EKF}, lorsque $\tilde{\omega}_s$ varie dans le temps, la position p_b est observable et lorsque $tilde\omega_s$ est une constante non nulle pendant au moins 7 pas de temps consécutifs, p_b devient non observable (cas rare). En outre, il est important de souligner que l'angle de lacet ψ est observable par l'EKF alors que cet angle est non observable par le système en temps continu.

Par conséquent, l'EKF n'est pas cohérent avec le système non linéaire car il ne partage pas les mêmes états observables. En effet, l'angle de lacet ψ est observable pour l'EKF. L'incohérence de l'EKF est due à la dépendance de la matrice d'observabilité locale $\mathcal{M}$ aux points de linéarisation pour évaluer les matrices jacobiennes Φ_k et H_k. En effet, le choix des points de linéarisation affecte les propriétés d'observabilité de l'EKF. Différents auteurs [31], [104] ont déjà discuté et proposé des solutions pour résoudre l'incohérence de l'EKF, principalement appliquées au problème de l'EKF-SLAM *(Simultaneous Localization and Mapping)*.

Pour conclure l'analyse d'observabilité, l'EKF est incohérent avec le système non linéaire car il crée une fausse observabilité contrairement à l'Imp.RIEKF. Ces observations confirment les propriétés de l'Imp.IEKF énoncées dans la partie *(2.3)*, à savoir qu'une erreur non linéaire résout le problème d'incohérence dans ce cas.

2.4.5 Étude des performances

Afin de quantifier la précision de l'EKF et de l'Imp.RIEKF, ces deux filtres sont évalués sur des simulations de trajectoires de mortier de 120 mm, générées par BALCO présenté dans la partie *(1.5)*.
Pour éviter toute divergence, chaque filtre est initialisé avec la position p_0, la vitesse v_0 et les angles d'Euler Ψ_0 de référence. L'initialisation des biais des capteurs ainsi que la position p_b et l'orientation R_b sont identiques pour les deux filtres, de même pour la covariance initiale, ainsi que la covariance du bruit de modèle et de mesure.

La précision des filtres est évaluée avec l'erreur quadratique moyenne *(RMSE - Root Mean Square Error)* définie comme suit :

$$RMSE_i = \sqrt{\frac{\sum_{k=1}^{N} \left(x_{ref_k} - \widehat{x}_k\right)^2}{N}} \tag{2.113}$$

avec N le nombre d'échantillons de la simulation i, $\widehat{x}_k$ la grandeur estimée par le filtre considéré et x_{ref_k} la grandeur de référence fournie par BALCO.

Les performances des deux filtres sont évaluées sur 100 simulations de trajectoires de mortier de 120 mm, d'après les configurations initiales mentionnées ci-dessus. Les portées des trajectoires testées varient de 3 000 m à 4 000 m avec des apogées allant de 800 m à 1500 m. Les figures 2.5 - 2.7 présentent les RMSE de l'EKF en fonction des RMSE de l'Imp.RIEKF pour les 100 trajectoires considérées.

D'après les figures 2.5 et 2.6, la plupart des marqueurs sont situés dans la partie supérieure des figures, donc l'Imp.RIEKF surpasse l'EKF pour estimer les positions et les vitesses de mortier de 120 mm. En effet, pour la plupart des trajectoires évaluées, les erreurs de l'Imp.RIEKF sont inférieures à celles de l'EKF. La figure 2.7 montre l'avantage de l'Imp.RIEKF pour estimer les angles d'Euler car presque tous les marqueurs sont situés dans la partie supérieure. En effet, l'Imp.RIEKF estime l'orientation du projectile comme un élément d'un groupe de Lie matriciel contrairement à l'EKF qui estime l'orientation comme un élément d'un espace euclidien. La représentation par un groupe de Lie de l'orientation est mieux adaptée à un tel problème de navigation.

D'après les résultats présentés, l'Imp.RIEKF est plus précis que l'EKF pour estimer la trajectoire d'un projectile et donc, l'Imp.IEKF est particulièrement adapté aux problèmes de navigation. Cette conclusion s'explique par la structure de ces filtres. En effet, un Imp.IEKF est basé sur une erreur non linéaire, linéarisée en partie sur l'espace euclidien identifié autour de l'élément neutre du groupe de Lie et associée à une mise à

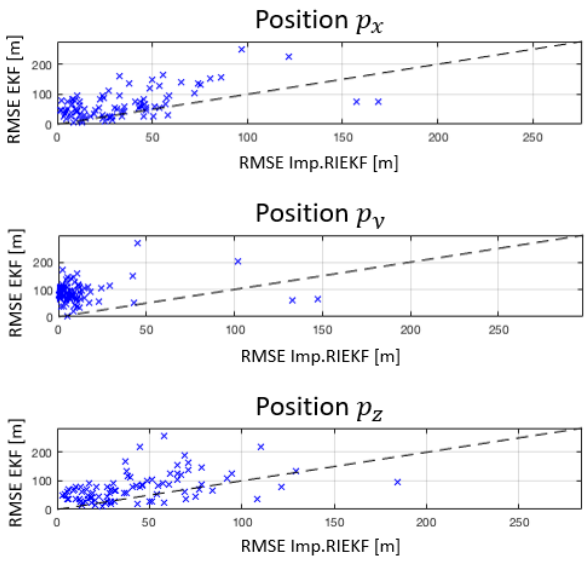

FIGURE 2.5 – Estimation de la position du projectile sur l'ensemble du jeu de données : $(RMSE_{EKF}, RMSE_{IEKF})$.

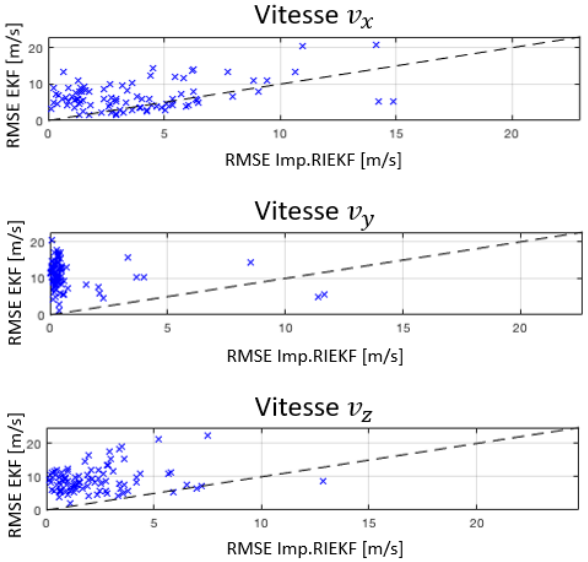

FIGURE 2.6 – Estimation de la vitesse du projectile sur l'ensemble du jeu de données : $(RMSE_{EKF}, RMSE_{IEKF})$.

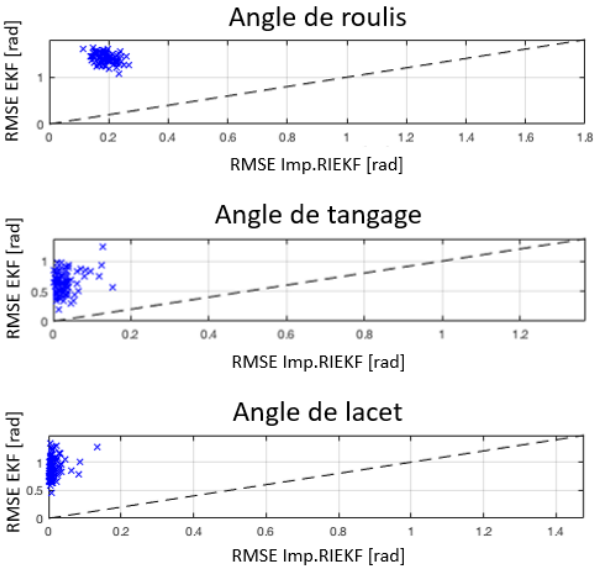

FIGURE 2.7 – Estimation des angles d'Euler du projectile sur l'ensemble du jeu de données : $(RMSE_{EKF}, RMSE_{IEKF})$.

jour exponentielle permettant d'augmenter la précision des estimations et d'éviter des problèmes d'incohérence.

2.5 Conclusion

Ce chapitre présente la méthodologie du Filtre de Kalman Étendu Invariant et de ses drivées. L'IEKF est un filtre de Kalman non linéaire, définit sur un groupe de Lie matriciel, qui possède des propriétés remarquables d'observabilité et de convergence, similaires à celles d'un filtre de Kalman linéaire. Pour cela, la dynamique d'évolution d'un IEKF doit satisfaire une propriété du groupe.

Dans le cas d'un problème de navigation, cette propriété n'est pas toujours satisfaite par l'ensemble du système. Il s'agit alors d'un Filtre de Kalman Imparfait Invariant Étendu. Ce filtre, basé à la fois sur la théorie des IEKF et à la fois sur la théorie des EKF, présente des propriétés intéressantes qui ont été vérifiées sur un exemple d'application. En se basant sur les mêmes modèles d'évolution et de mesure, un Imp.IEKF et un EKF ont été présentés. L'analyse d'observabilité montre que l'EKF crée de fausses observabilités contrairement à l'Imp.IEKF. De plus, ces deux filtres sont évalués sur des simulations de trajectoires de mortier de 120 mm et les résultats montrent l'Imp.IEKF est adapté à la navigation des projectiles contrairement à l'EKF.

Au vu des résultats de ce chapitre, un Imp.IEKF semble être adéquate pour la navigation des projectiles. Afin d'optimiser les estimations, une approche vise à employer des réseaux de neurones pour ajuster dynamiquement des paramètres de bruit d'un tel filtre.

Chapitre 3

Réglage du bruit de mesure d'un filtre de Kalman par un réseau de neurones

Sommaire

3.1 Introduction

Dans le domaine militaire, la connaissance de la position, de la vitesse et de l'orientation d'un projectile à chaque instant est essentielle pour son guidage. Pour cela, les solutions de navigation proposées dans la littérature exploitent soit l'intégralité, soit une partie des mesures du récepteur GNSS [1] et de l'IMU [2] embarquée, en les fusionnant par différents types de filtres de Kalman [9], [32]-[42].

Le filtre de Kalman est une solution populaire d'estimation d'état basée sur les mesures de différents capteurs [4]. En effet, comme présenté dans la partie *(1.3)* de ce document, le Filtre de Kalman Linéaire *(Linear Kalman Filter - LKF)* est un estimateur optimal basé sur des modèles d'évolution et de mesure linéaires et où tous les bruits sont blancs et gaussiens [21], [23], [24].
Cependant, pour les dynamiques non linéaires comme celles d'évolution de la trajectoire d'un projectile, un filtre de Kalman non linéaire comme le Filtre de Kalman Étendu *(Extended Kalman Filter - EKF)* est préférable. Dans ce cas, le modèle d'évolution et/ou le modèle de mesure est non linéaire. Ces modèles sont alors linéarisés autour de l'estimation précédente à l'aide d'un développement de premier ordre, impliquant des erreurs d'estimation faibles pour être valides. La théorie standard du filtre de Kalman est ensuite appliquée aux modèles linéarisés. Bien que l'EKF soit largement utilisé pour résoudre différents problèmes d'estimation, en particulier des problèmes de navigation, ce filtre présente plusieurs limitations causées par la linéarisation [4], [21], [25]-[31]. En effet, contrairement au filtre de Kalman linéaire, la stabilité d'un EKF n'est pas garantie et une fausse initialisation peut conduire à sa divergence. De plus, un EKF peut être incohérent et créer une fausse observabilité.
Comme présenté au chapitre *2*, une extension de l'EKF exploitant les symétries des modèles grâce aux groupes de Lie permet de surmonter plusieurs limitations de l'EKF. En effet, le filtre de Kalman Étendu Invariant *(Invariant Extended Kalman Filter - IEKF)* partage des propriétés similaires à celles du filtre de Kalman Linéaire pour des systèmes non linéaires. Toutefois, l'IEKF est adapté exclusivement aux problèmes de navigation à partir de capteurs idéaux, ne nécessitant pas d'évaluer les dérives de ces mesures. Dans

1. GNSS : *Global navigation satellite system*
2. IMU : *Inertial Measurement Unit*

le cas de capteurs imparfaits, le filtre de Kalman Imparfait Invariant Étendu *(Imperfect Invariant Extended Kalman Filter - Imp.IEKF)* est plus adéquat. En effet, l'Imp.IEKF permet de prendre en compte les dérives des capteurs tout en présentant des performances d'estimation supérieures à celles d'un EKF standard du fait de l'utilisation de groupes de Lie matriciels, d'erreurs non linéaires et de mises à jour en partie exponentielles.

La précision de tous filtres de Kalman dépend fortement du réglage des matrices de covariance des bruits. Généralement, ces matrices sont constantes et identifiées empiriquement par l'utilisateur. Toutefois, des matrices de bruit constantes ne sont pas adaptées à tous les problèmes d'estimation et peuvent largement influer sur les précisions obtenues. Ainsi, pour des problèmes de navigation, des matrices variables et adaptées aux différentes trajectoires sont souvent préférables. D'après les résultats rapportés dans [77]-[82], une solution vise à utiliser des réseaux de neurones afin d'adapter dynamiquement les valeurs de ces matrices aux différentes trajectoires.

Ce chapitre étudie une méthode d'estimation de la trajectoire d'un projectile basée sur un filtre de Kalman pour lequel la matrice de covariance du bruit de mesure est déterminée par un réseau de neurones. Cette matrice, pertinente lors de l'étape de mise à jour du filtre, est délicate à régler afin d'optimiser toutes les trajectoires estimées. Le modèle d'IA sélectionné est un réseau de neurones convolutifs *(Convolutional Neural Network - CNN)* pour ajuster dynamiquement la matrice de covariance du bruit de mesure du filtre afin de produire une matrice variable dans le temps, adaptée aux dynamiques et aux différentes phases du vol du projectile. En d'autres termes, les objectifs de ce chapitre sont :

- → d'évaluer l'apport des réseaux de neurones pour régler dynamiquement les paramètres de bruit de mesure d'un filtre de Kalman. Pour cela, les performances d'estimation de plusieurs filtres de Kalman sont étudiées dans le cas où des réseaux de neurones convolutifs sont entraînés à ajuster dynamiquement la matrice de covariance du bruit de mesure.
- → de tester cette solution d'ajustement de la matrice de covariance du bruit de mesure sur un filtre de Kalman Imparfait Invariant Étendu *(Imperfect Invariant Extended Kalman Filter - Imp.IEKF)* pour estimer la trajectoire d'un mortier de 120 mm à partir des mesures de l'IMU et du champ magnétique de référence.
- → d'évaluer l'apport de cette méthode à un filtre de Kalman Étendu *(Extended Kalman Filter - EKF)* qui estime l'angle et la vitesse de roulis d'un projectile caractérisé par un tir tendu à partir des deux magnétomètres radiaux embarqués dans la munition. L'EKF considéré a été testé en vol dans des travaux antérieurs.

→ de régler la covariance du bruit de mesure d'un filtre de Kalman Invariant Étendu *(Invariant Extended Kalman Filter - IEKF)*, partageant les mêmes propriétés qu'un filtre de Kalman Linéaire *(Linear Kalman Filter - LKF)*, afin de déterminer dans quelles mesures cette méthode d'optimisation est adaptée.

La première partie *(3.2)* présente le principe de fonctionnement d'un réseau de neurones convolutifs *(Convolutional Neural Network - CNN)* ainsi que la méthode d'ajustement de la matrice de covariance mise en place. La partie *(3.3)* présente les résultats d'estimation de la trajectoire d'un mortier de 120 mm par un Imp.IEKF dont la matrice de covariance du bruit de mesure est ajustée dynamiquement par un CNN. La partie *(3.4)* propose les résultats du réglage dynamique de la matrice de covariance du bruit de mesure par un CNN appliqué à un filtre de Kalman Étendu et un filtre de Kalman Invariant Étendu.

3.2 Réseau de neurones convolutifs

Cette partie introduit la démarche mise en place pour ajuster dynamiquement la matrice de covariance du bruit de mesure d'un filtre de Kalman. Elle sera complétée par le choix du type d'architecture et le principe de fonctionnement d'un réseau de neurones convolutifs *(Convolutional Neural Network - CNN)*.

3.2.1 Principe d'estimation de la covariance d'un filtre par un CNN

Un filtre de Kalman non linéaire tel que le filtre de Kalman Étendu *(Extended Kalman Filter - EKF)* peut diverger dans le cas où les matrices de covariance du bruit de mesure et du bruit de modèle sont incorrectement réglées. Généralement, ces deux matrices sont constantes, fixes et déterminées empiriquement à l'aide de connaissances sur le modèle et les capteurs afin d'obtenir les meilleures estimations possibles.

Toutefois, la matrice de covariance du bruit de mesure, qui reflète la confiance accordée aux observations, joue un rôle déterminant lors de l'étape de mise à jour car elle intervient dans le calcul du gain de Kalman, et donc, dans la mise à jour des états et de la covariance de l'erreur d'estimation. Ainsi, une matrice de covariance du bruit de mesure constante et fixe ne peut pas refléter des dynamiques et des conditions changeantes, lorsque par exemple, les mesures ne sont plus viables sous l'influence de certains facteurs extérieurs.

D'après ces observations, une matrice de covariance du bruit de mesure variable semble être appropriée à des filtres où les mesures peuvent être plus ou moins fiables suivant les conditions extérieures. Comme présenté dans la figure 3.1, pour produire une matrice de covariance du bruit de mesure variable et adaptée aux conditions de mesures extérieures, une solution vise à utiliser des réseaux de neurones tels que présentés dans [77]-[82], afin d'ajuster dynamiquement cette matrice aux mesures d'un filtre de Kalman.

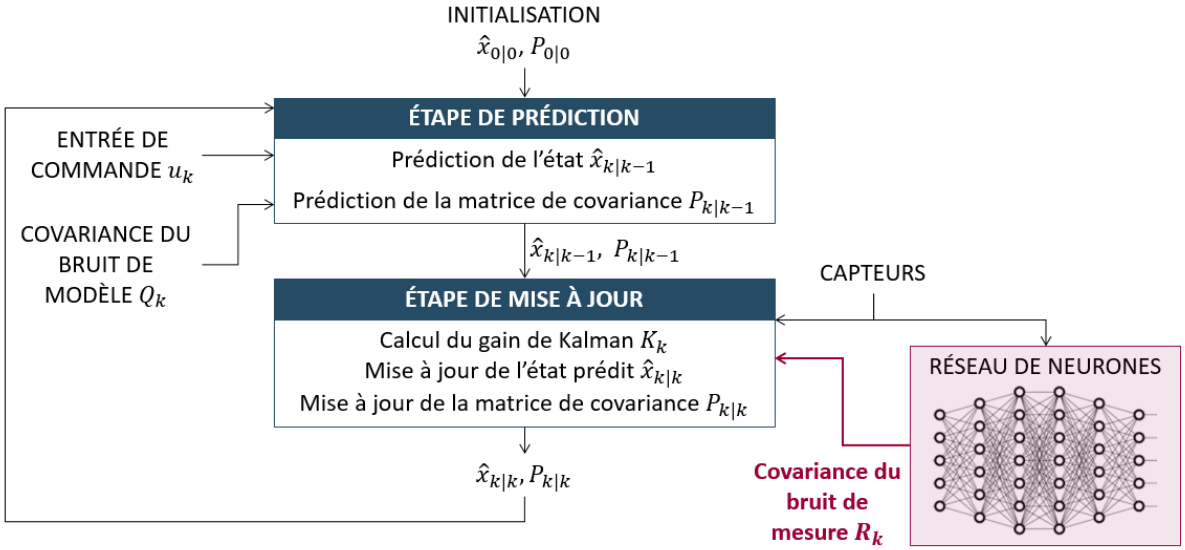

FIGURE 3.1 – Principe de fonctionnement du réglage de la matrice de covariance du bruit de mesure d'un filtre de Kalman par un réseau de neurones.

Afin de produire une matrice de covariance du bruit de mesure variable qui reflète la confiance accordée aux mesures en fonction des différents facteurs influents sur l'obtention de ces mesures, plusieurs réseaux de neurones peuvent être sélectionnés. Dans le cadre des ces travaux, un réseau de neurones convolutifs *(Convolutional Neural Network - CNN)* est préféré car cette architecture présente plusieurs avantages [82]. En effet, pour ce type d'application, le CNN nécessite un nombre limité de paramètres contrairement à une autre architecture de réseau telle que les réseaux récurrents. De plus, l'estimation de la covariance du bruit de mesure est déterminée uniquement à partir des observations et non pas à partir des estimations précédentes, contrairement aux réseaux récurrents qui nécessitent une connaissance préalable du contexte d'estimation.

3.2.2 La structure d'un réseau de neurones convolutifs

Un réseau de neurones convolutifs *(Convolutional Neural Network - CNN)* est une architecture de réseau de neurones dont certaines couches sont basées sur des opérations de convolution. Pour cela, les CNN sont le type de réseau de neurones le plus performant pour résoudre des problèmes de vision par ordinateur, principalement la reconnaissance et l'identification d'objets dans des images.

La structure classique d'un CNN est présentée dans la figure 3.2.

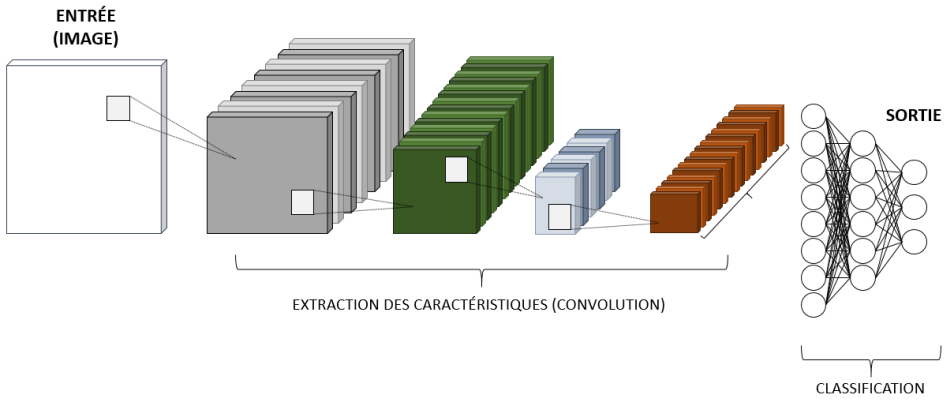

FIGURE 3.2 – Principe de fonctionnement d'un réseau de neurones convolutifs *(Convolutional Neural Network - CNN)*.

Comme présenté dans la figure 3.2, un CNN est composé de deux parties :

- la première partie d'un CNN vise à extraire les caractéristiques spatiales dans les données d'entrée. Pour cela, l'extracteur de caractéristiques est généralement composé d'une succession de couches de convolution et de fonctions d'activation non linéaires.
- la seconde partie d'un CNN comprend des couches entièrement connectées (*fully connected*), comme un MLP *(Multilayer perceptron)*, afin de combiner les informations apprises par les opérations de convolution pour réaliser la fonction souhaitée.

Comme mentionné précédemment, pour résoudre un problème de régression non linéaire, un CNN est principalement composé de trois types de couches : les couches de convolution, les couches d'activation non linéaires et les couches *fully connected*.

La couche de convolution

La couche de convolution constitue toujours la première couche des CNN. Cette couche, basée sur l'opération mathématique de convolution, vise à analyser les données d'entrée et à détecter la présence de caractéristiques spatiales plus ou moins complexes.

Pour cela, comme illustré dans la figure 3.3, un filtre (également appelé noyau) glisse sur la donnée d'entrée et effectue l'opération de convolution, notée $*$, entre la matrice d'entrée et le filtre de convolution. Les différents filtres, de taille définie, sont évalués lors de l'étape d'entraînement et permettent de détecter des caractéristiques spécifiques.

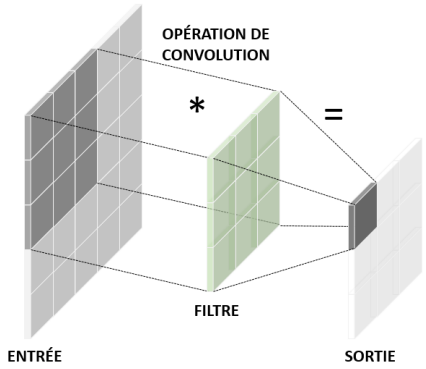

FIGURE 3.3 – Opération de convolution d'un CNN.

Les couches d'activation non linéaires

Les couches de convolution sont généralement suivies par des couches d'activation non linéaires afin d'ajouter des non linéarités au modèle pour améliorer l'apprentissage. Les non linéarités sont nécessaires pour inclure de la complexité afin que le réseau de neurones apprenne à traiter des problèmes complexes, sinon, les couches du réseau pourraient se résumer par une fonction linéaire. Les fonctions d'activation les plus couramment utilisées sont :

- la fonction d'activation *ReLU (Rectified Linear Unit)* définie telle que :

$$\forall x \in \mathbb{R}, \quad ReLU(x) = \begin{cases} x, & x > 0 \\ 0, & x \leq 0 \end{cases} \in [0; +\infty[\tag{3.1}$$

  Cette fonction est très employée dans les réseaux de neurones car elle supprime toutes les valeurs négatives et sa forme permet au réseau d'apprendre rapidement.

Toutefois, cette fonction n'est pas adaptée lorsque les données recherchées sont à valeur négative.

- la fonction d'activation *Sigmoid* définie telle que :

$$\forall x \in \mathbb{R}, \quad Sigmoid(x) = \frac{1}{1+e^{-x}} \quad \in [0;1] \tag{3.2}$$

Cette fonction est généralement employée pour la classification binaire comme les sorties sont bornées et donc normalise la sortie des neurones. Cependant, les données ne sont pas centrées sur zéro et pour des valeurs importantes ou très faibles, les sorties restent similaires.

- la fonction d'activation *TanH (Hyperbolic Tangent)* définie telle que :

$$\forall x \in \mathbb{R}, \quad TanH(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \quad \in]-1;1[\tag{3.3}$$

Cette fonction est majoritairement utilisée pour traiter des données en continu et est centrée sur zéro donc elle permet également de traiter les données à valeurs négatives. Toutefois, pour des valeurs très importantes ou très faibles, les sorties resteront semblables.

Les couches d'opérations de mise en commun

Les couches *Fully Connected* constituent la seconde partie d'un CNN et permettent de regrouper les informations apprises lors des précédentes couches de convolution pour effectuer la tâche souhaitée. Pour cela, les neurones d'une couche de *Fully Connected* sont complètement connectés à tous les neurones de la couche suivante, comme pour un MLP *(Multilayer perceptron)*. Les couches de *Fully Connected* effectuent des combinaisons linéaires entrecoupées de fonctions d'activation non linéaires.

D'après la littérature [120], [121], un CNN est parfaitement adapté pour détecter des caractéristiques spatiales dans des images. Une image est une matrice avec plus ou moins de canaux. Il est alors aisé d'effectuer l'analogie avec le problème d'estimation traité dans ce chapitre. Les mesures utilisées pour corriger les prédictions d'un filtre de Kalman sont des matrices, assimilables aux matrices décrivant des images et un CNN permet de détecter des caractéristiques spatiales dans ces données. Les couches de *fully connected* sont ensuite utilisées pour effectuer la tâche souhaitée, à savoir, un problème de régression.

3.3 Réglage de la covariance d'un Imp.RIEKF par un CNN

Cette partie présente un filtre de Kalman Imparfait Invariant Étendu à droite *(Imperfect Right Invariant Extended Kalman Filter - Imp.RIEKF)* pour estimer la trajectoire d'un mortier de 120 mm à partir des capteurs inertiels embarqués dans la munition et de la connaissance du champ magnétique de référence. La matrice de covariance du bruit de mesure de ce filtre est ajustée dynamiquement par un CNN. Plusieurs méthodes de prétraitement des données d'entrée du réseau sont étudiées afin d'évaluer la pertinence de la solution proposée.

3.3.1 Imp.RIEKF pour estimer la trajectoire d'un projectile à partir de l'IMU

La figure 3.4 présente le filtre de Kalman Imparfait Invariant Étendu à droite employé pour estimer la trajectoire du projectile et les biais des capteurs en utilisant exclusivement les mesures de l'accéléromètre, du gyromètre et du magnétomètre embarqué ainsi que la connaissance du champ magnétique de référence. La méthodologie des Imp.IEKF est détaillée au chapitre 2 de ce document.

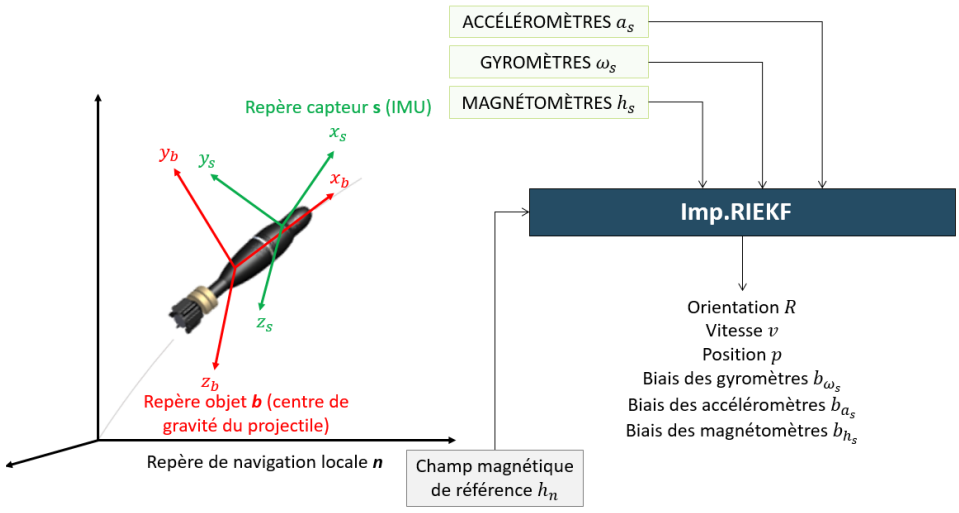

FIGURE 3.4 – Filtre de Kalman Imparfait Invariant Étendu à droite *(Imp.RIEKF)* pour estimer la trajectoire d'un projectile à partir de l'IMU embarquée et du champ magnétique de référence.

Comme présenté dans la figure 3.4, l'Imp.RIEKF vise à estimer, à partir des mesures bruitées et biaisées de l'accéléromètre a_s, du gyromètre ω_s et du magnétomètre h_s ainsi que de la connaissance du champ magnétique de référence h_n :

- l'orientation $R \in SO(3)$, la vitesse $v \in \mathbb{R}^{3\times1}$ et la position $p \in \mathbb{R}^{3\times1}$ du projectile dans le repère local de navigation $\boldsymbol{n}$,
- les biais des gyromètres $b_{\omega_s} \in \mathbb{R}^{3\times1}$, des accéléromètres $b_{a_s} \in \mathbb{R}^{3\times1}$ et des magnétomètres $b_{h_s} \in \mathbb{R}^{3\times1}$ dans le repère capteur $\boldsymbol{s}$.

Modèles et notations

Les mesures du gyromètre, de l'accéléromètre et du magnétomètre sont modélisées telles que :

$$\tilde{\omega}_s = \omega_s + b_{\omega_s} + W_{\omega_s}, \quad \tilde{a}_s = a_s + b_{a_s} + W_{a_s}, \quad \tilde{h}_s = a_s + b_{h_s} + W_{h_s} \tag{3.4}$$

avec ω_s, a_s et $h_s \in \mathbb{R}^{3\times1}$ les mesures du gyromètre, de l'accéléromètre et du magnétomètre dans le repère capteur $\boldsymbol{s}$, b_{ω_s}, b_{a_s} et $b_{h_s} \in \mathbb{R}^{3\times1}$ les biais des capteurs inertiels dans le repère capteur $\boldsymbol{s}$, et $W_{\omega_s} \sim \mathcal{N}(0_{3\times1}, \Sigma_\omega)$, $W_{a_s} \sim \mathcal{N}(0_{3\times1}, \Sigma_a)$ et $W_{h_s} \sim \mathcal{N}(0_{3\times1}, R)$ des bruits blancs gaussiens. Le modèle d'évolution des biais des capteurs inertiels est un mouvement brownien défini par les équations suivantes :

$$\frac{d}{dt}b_{\omega_s} = W_{b_{\omega_s}}, \qquad \frac{d}{dt}b_{a_s} = W_{b_{a_s}}, \qquad \frac{d}{dt}b_{h_s} = W_{b_{h_s}}, \tag{3.5}$$

avec $W_{b_{\omega_s}} \sim \mathcal{N}(0_{3\times1}, \Sigma_{b_\omega})$, $W_{b_{a_s}} \sim \mathcal{N}(0_{3\times1}, \Sigma_{b_a})$ et $W_{b_{h_s}} \sim \mathcal{N}(0_{3\times1}, \Sigma_{b_h})$ des bruits blancs gaussiens.

La Terre est supposée plate pendant la durée de vol du projectile et la rotation de la Terre sans influence sur l'estimation de la trajectoire du projectile. Ainsi, la dynamique d'évolution de la trajectoire du projectile en temps continu est donnée par :

$$\tfrac{d}{dt}R = R[\tilde{\omega}_s - b_{\omega_s} - W_{\omega_s}]_\times, \tag{3.6}$$

$$\tfrac{d}{dt}v = R(\tilde{a}_s - b_{a_s} - W_{a_s}) + g, \tag{3.7}$$

$$\tfrac{d}{dt}p = v, \tag{3.8}$$

avec $[.]_\times$ l'opérateur du groupe de Lie $SO(3)$ présenté dans l'équation (2.10) et g le vecteur gravité constant.

À partir d'une condition initiale donnée, des mesures inertielles et du champ magnétique de référence, l'Imp.RIEKF vise à estimer les états suivants :

$$x = \begin{pmatrix} R, & v, & p, & b_{\omega_s}, & b_{a_s}, & b_{h_s} \end{pmatrix}. \tag{3.9}$$

Les observations considérées pour mettre à jour les prédictions du filtre sont l'estimation du champ magnétique de référence $h_n \in \mathbb{R}^{3\times1}$ dans le repère local de navigation $\boldsymbol{n}$ à partir des mesures des magnétomètres h_s. Le modèle de mesure est alors :

$$h_n = R(\tilde{h}_s - b_{h_s} - W_{h_s}) \tag{3.10}$$

L'entrée de commande u_t comprend les mesures du gyromètre et de l'accéléromètre telles que :

$$u_t = \begin{bmatrix} \omega_s \\ a_s \end{bmatrix} \in \mathbb{R}^{6\times1} \tag{3.11}$$

D'après le modèle dynamique (3.5) - (3.8) associé aux états x (3.9), le bruit de modèle $W \sim \mathcal{N}(0_{18\times1}, Q_k)$ est défini tel que :

$$W = \begin{bmatrix} W_{\omega_s}^T & W_{a_s} & 0_{3\times1}^T & W_{b_{\omega_s}}^T & W_{b_{a_s}}^T & W_{b_{h_s}}^T \end{bmatrix}^T \tag{3.12}$$

La matrice de covariance du bruit de modèle $Q_k \in \mathbb{R}^{18\times18}$ est alors :

$$Q_k = cov\left(\Sigma_\omega, \Sigma_a, 0_{3\times1}, \Sigma_{b_\omega}, \Sigma_{b_a}, \Sigma_{b_h}\right). \tag{3.13}$$

Erreur d'estimation de l'Imp.RIEKF

Comme mentionné dans le chapitre 2, une partie des états d'un Imp.IEKF est embarquée dans un groupe de Lie matriciel associée à une erreur non linéaire et une seconde partie des états est embarquée dans un espace euclidien associée à une erreur linéaire. Les états du projectile (R, v, p) sont embarqués dans le groupe de Lie matriciel $SE_2(3) \subset \mathbb{R}^{5\times5}$ tel que :

$$\chi_t \triangleq \begin{bmatrix} R & v & p \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix} \in SE_2(3), \quad R \in SO(3),\ v \in \mathbb{R}^{3\times1},\ p \in \mathbb{R}^{3\times1}. \tag{3.14}$$

Les dynamiques des biais des capteurs inertiels b_{ω_s}, b_{a_s} et b_{h_s} évoluent dans $\mathbb{R}^{3\times1}$ et sont associées à une erreur linéaire entre l'état estimé, noté $\widehat{.}$, et l'état réel tel que :

$$\xi_{b_\omega} = \widehat{b}_{\omega_s} - b_{\omega_s}, \quad \xi_{b_a} = \widehat{b}_{a_s} - b_{a_s}, \quad \xi_{b_h} = \widehat{b}_{h_s} - b_{\omega_s}. \tag{3.15}$$

L'erreur non linéaire d'estimation de l'Imp.RIEKF relative aux états x (3.9) est composée de l'erreur invariante à droite associée à (R, v, p) et de l'erreur linéaire associée aux biais telle que :

$$e_t = \begin{pmatrix} \eta_{\chi t}, & \xi_{b_\omega}, & \xi_{b_a}, & \xi_{b_h} \end{pmatrix} \tag{3.16}$$

avec $\eta_{\chi t} = \widehat{\chi}_t \chi_t^{-1} \in SE_2(3)$ l'erreur invariante à droite des états R, v et p tels que :

$$\eta_{\chi t} = \widehat{\chi}_t \chi_t^{-1} = \begin{bmatrix} \widehat{R} & \widehat{v} & \widehat{p} \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix} \begin{bmatrix} R^T & -R^T v & -R^T p \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix} = \begin{bmatrix} \widehat{R}R^T & \widehat{v} - \widehat{R}R^T v & \widehat{p} - \widehat{R}R^T p \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix} \tag{3.17}$$

Une estimation valide de $\widehat{\chi}_t$ signifie que $\eta_{\chi t}$ est proche de l'élément identité du groupe de Lie, et donc de l'algèbre $\mathfrak{g} \in \mathbb{R}^{d\times d}$. Ainsi, localement autour de l'identité, l'algèbre est identifiée à l'espace euclidien $\mathbb{R}^{d\times1}$ par un développement au premier ordre de la fonction exponentielle $exp_m(.)$. L'erreur invariante est linéarisée d'après l'équation suivante :

$$\eta_{\chi t} = exp(\xi_t) = exp_m(\mathcal{L}_{\mathfrak{se}_2(3)}(\xi_t)) \approx \quad I_5 + \mathcal{L}_{\mathfrak{se}_2(3)}(\xi_t) \tag{3.18}$$

Ainsi, l'erreur invariante définie par (3.17) est linéarisée telle que :

$$\eta_{\chi t} = \begin{bmatrix} \widehat{R}R^T & \widehat{v} - \widehat{R}R^T v & \widehat{p} - \widehat{R}R^T p \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix} \approx \begin{bmatrix} I_3 + [\xi_R]_\times & \xi_v & \xi_p \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix} \tag{3.19}$$

avec $\eta_R = \widehat{R}R^T \approx I_3 + [\xi_R]_\times$ l'erreur invariante d'orientation, $\eta_v = \widehat{v} - \widehat{R}R^T \approx \xi_v$ l'erreur invariante de vitesse et $\eta_p = \widehat{p} - \widehat{R}R^T p \approx \xi_p$ l'erreur invariante de position.

L'erreur linéarisée $\xi_t \sim \mathcal{N}(0, P_t)$ de l'Imp.RIEKF associée aux états x (3.9) est alors :

$$\xi_t = \begin{pmatrix} \xi_R, & \xi_v, & \xi_p, & \xi_{b_\omega}, & \xi_{b_a}, & \xi_{b_h} \end{pmatrix} \in \mathbb{R}^{18\times1}. \tag{3.20}$$

Étape de prédiction de l'Imp.RIEKF

La prédiction des états x (3.9) en temps continu est donnée par les modèles dynamiques (3.5) - (3.8) dans le cas où aucune perturbation W (3.13) n'est présente :

$$\frac{d}{dt}\begin{bmatrix}\widehat{\chi}_t\\ \widehat{b}_{\omega_s}\\ \widehat{b}_{a_s}\\ \widehat{b}_{h_s}\end{bmatrix} = \begin{bmatrix}f_{u_t}(\widehat{\chi}_t, 0_{9\times1})\\ 0_{3\times1}\\ 0_{3\times1}\\ 0_{3\times1}\end{bmatrix} f_{u_t}(\chi_t, W) = \begin{bmatrix}R[\tilde{\omega}_s - b_{\omega_s} - W_{\omega_s}]_\times & R(\tilde{a}_s - b_{a_s} - W_{a_s}) + g & v\\ 0_{1\times3} & 0 & 0\\ 0_{1\times3} & 0 & 0\end{bmatrix} \tag{3.21}$$

La discrétisation de la prédiction des états (3.21) à la période d'échantillonnage des capteurs inertiels définie telle que $\Delta t = t_k - t_{k-1}$ est alors :

$$\widehat{R}_{k|k-1} = \widehat{R}_{k-1|k-1} exp_{SO(3)}\left((\tilde{\omega}_{s_k} - \widehat{b}_{\omega_{s_{k-1|k-1}}})\Delta t\right) \tag{3.22}$$

$$\widehat{v}_{k|k-1} = \widehat{v}_{k-1|k-1} + \left(\widehat{R}_{k-1|k-1}(\tilde{a}_{s_k} - \widehat{b}_{a_{s_{k-1|k-1}}}) + g\right)\Delta t \tag{3.23}$$

$$\widehat{p}_{k|k-1} = \widehat{p}_{k-1|k-1} + \widehat{v}_{k-1|k-1}\Delta t \tag{3.24}$$

$$\widehat{b}_{\omega_{s_{k|k-1}}} = \widehat{b}_{\omega_{s_{k|k-1}}},\ \widehat{b}_{a_{s_{k|k-1}}} = \widehat{b}_{a_{s_{k|k-1}}},\ \widehat{b}_{h_{s_{k|k-1}}} = \widehat{b}_{h_{s_{k|k-1}}} \tag{3.25}$$

avec $exp_{SO(3)}(.)$ l'application exponentielle du groupe de Lie $SO(3)$ définie au chapitre 2.

La matrice de covariance de l'erreur linéarisée ξ_t (3.20), notée P_t, est calculée par l'équation de Riccati :

$$\frac{d}{dt}P_t = A_t P_t + P_t A_t^T + G_w Q_t G_w^T \tag{3.26}$$

avec Q_t, la matrice de covariance du bruit de modèle définie par l'équation (3.13), et A_t et G_w les matrices identifiées d'après la dynamique d'évolution de l'erreur linéarisée :

$$\frac{d}{dt}\xi_t = A_t\xi_t + G_w W + o(||\xi_t||) \tag{3.27}$$

D'après la dynamique d'évolution de l'erreur linéarisée ξ_t (3.20), les matrices A_t et G_w sont identifiées telles que :

$$A_t = \begin{bmatrix}0_{3\times3} & 0_{3\times3} & 0_{3\times3} & -R & 0_{3\times3} & 0_{3\times3}\\ [g]_\times & 0_{3\times3} & 0_{3\times3} & -[v]_\times R & -R & 0_{3\times3}\\ 0_{3\times3} & I_{3\times3} & 0_{3\times3} & -[p]_\times R & 0_{3\times3} & 0_{3\times3}\\ & & & 0_{9\times18} & & \end{bmatrix}, \quad G_w = \begin{bmatrix}R & 0_{3\times3} & 0_{3\times3} & 0_{3\times3}\\ [v]_\times R & R & 0_{3\times3} & 0_{3\times3}\\ [p]_\times R & 0_{3\times3} & R & 0_{3\times3}\\ & 0_{9\times9} & & I_{9\times9}\end{bmatrix} \tag{3.28}$$

Dans le cas où les biais sont fixes, la dynamique de l'erreur linéarisée est indépendante de la trajectoire estimée et est *affine de groupe*, donc les deux théorèmes fondamentaux de la théorie IEKF sont satisfaits. Dans cette partie, les biais des capteurs sont évalués, alors la dynamique d'évolution des erreurs (3.20) dépend de la trajectoire estimée. La dynamique associée n'est pas *affine de groupe*.

Remarque 3.3.1. *L'identification des matrices A_t et G_W est présentée dans l'annexe D.1. de ce document.*

La prédiction de la matrice de covariance en temps discret est [102] :

$$P_{k|k-1} = \Phi_k P_{k-1|k-1} {\Phi_k}^T + Q_k \tag{3.29}$$

avec Φ_k la matrice de transition d'état [99], [100], [102] évaluée telle que :

$$\frac{d}{dt}\Phi(t,t_k) = A_t(t)\Phi(t,t_k), \qquad \Phi(t_0,t_0) = I \tag{3.30}$$

de sorte que la matrice de transition d'état en temps discret est :

$$\Phi_k = exp\left(\int_{t_k}^{t_{k+1}} A(\tau)d\tau\right) = exp_m\left(A_t\Delta t\right) \tag{3.31}$$

et avec Q_k la matrice de covariance du bruit de modèle en temps discret donnée par l'équation suivante :

$$Q_k = \int_{t_k}^{t_{k+1}} \Phi(t_{k+1},\tau)G_w Q_t(\tau) G_w^T \Phi(t_{k+1},\tau)^T d\tau \approx \Phi_k G_w Q_t G_w \Phi_k^T \Delta t \tag{3.32}$$

Étape de mise à jour de l'Imp.RIEKF

Les prédictions obtenues à l'étape précédente (3.22) - (3.25) et (3.29) sont mises à jour avec les observations qui sont l'estimation du champ magnétique de référence dans le repère local de navigation $\boldsymbol{n}$ à partir des mesures du magnétomètre. Le modèle de mesure est alors :

$$h_n = h(x, W_{h_n}) = R(\tilde{h}_s - b_{h_s} - W_{h_s}) \in \mathbb{R}^{3\times 1} \tag{3.33}$$

avec $h_n \in \mathbb{R}^{3\times 1}$ le champ magnétique dans le repère local de navigation $\boldsymbol{n}$, $\tilde{h}_s$ le modèle des mesures du magnétomètre, b_{h_s} les biais, et $W_{h_s} \in \mathbb{R}^{3\times 1}$ les bruits des mesures.

La linéarisation du modèle d'observation par rapport à ξ_t (3.20) permet d'identifier la matrice d'observation $H \in \mathbb{R}^{18\times 3}$ et la matrice de covariance du bruit de mesure $N \in \mathbb{R}^{3\times 3}$

telle que :

$$H = \begin{bmatrix} [R\tilde{h}_s]_\times & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & R \end{bmatrix} \tag{3.34}$$

$$N = cov(RW_{h_s}) = Rcov(W_{h_s})R^T \tag{3.35}$$

Remarque 3.3.2. *L'identification des matrices H et N est présentée dans l'annexe D.2. de ce document.*

L'identification de la matrice d'observation (3.34) et de la matrice de covariance du bruit de mesure (3.35) permettent d'évaluer le gain de Kalman tel que :

$$K_k = P_{k|k-1}H_k^T(H_kP_{k|k-1}H_k^T + N_k)^{-1} \tag{3.36}$$

Le vecteur d'erreur linéarisée mis à jour est alors :

$$\begin{bmatrix} \xi_{R_{k|k}}^T, & \xi_{v_{k|k}}^T, & \xi_{p_{k|k}}^T, & \xi_{b_{\omega_{k|k}}}^T, & \xi_{b_{a_{k|k}}}^T, & \xi_{b_{h_{k|k}}}^T \end{bmatrix}^T = K_k(h_n - \widehat{h}_n) \tag{3.37}$$

L'état prédit $\widehat{x}_{k|k-1}$ et la matrice de covariance d'erreur prédite $P_{k|k-1}$ sont mis à jour conformément à la méthodologie des Imp.IEKF rappelée au chapitre 2 :

$$\begin{gathered} \widehat{\chi}_{k|k} = exp_{SE_2(3)}\left(\begin{bmatrix} \xi_{R_{k|k}}^T, & \xi_{v_{k|k}}^T, & \xi_{p_{k|k}}^T \end{bmatrix}^T\right)\widehat{\chi}_{k|k-1} \\ \widehat{b}_{\omega_{s_{k|k}}} = \widehat{b}_{\omega_{s_{k|k-1}}} + \xi_{b_{\omega_{k|k}}}, \quad \widehat{b}_{a_{s_{k|k}}} = \widehat{b}_{a_{s_{k|k-1}}} + \xi_{b_{a_{k|k}}}, \quad \widehat{b}_{h_{s_{k|k}}} = \widehat{b}_{h_{s_{k|k-1}}} + \xi_{b_{h_{k|k}}} \\ P_{k|k} = (I_{18} - K_kH_k)P_{k|k-1} \end{gathered} \tag{3.38}$$

avec $exp_{SE_2(3)}(.)$ l'application exponentielle de $SE_2(3)$ définie dans le chapitre 2.

3.3.2 Détails d'entraînement du CNN pour le réglage de la covariance du bruit de mesure de l'Imp.RIEKF

L'Imp.RIEKF présenté dans la partie *(3.3.1)* est, comme tout filtre de Kalman, sensible au réglage de la matrice de covariance du bruit de mesure défini par l'équation (3.35) telle que :

$$N = Rcov\left(W_{h_s}\right)R^T \tag{3.39}$$

avec $R \in SO(3)$ la matrice d'orientation du projectile, et $cov\,(W_{h_s}) \in \mathbb{R}^{3\times1}$ la covariance du bruit des mesures des magnétomètres.

Habituellement, la covariance du bruit des mesures $cov\,(W_{h_s})$ est ajustée manuellement, de façon empirique en fonction des indications des erreurs des capteurs et est constante pendant la durée de vol du projectile. Afin d'améliorer les estimations de l'Imp.RIEKF et d'évaluer l'apport de l'intelligence artificielle pour optimiser un filtre de Kalman, un réseau de neurones convolutifs *(Convolutional Neural Network - CNN)* est entraîné à partir des mesures des magnétomètres h_s pour estimer $cov\,(W_{h_s})$, variable dans le temps et adaptée aux différentes phases du vol du projectile. Il est supposé qu'il n'y ait pas de corrélations entre les trois axes du magnétomètre, donc que $cov\,(W_{h_s})$ est une matrice diagonale. Cette hypothèse est couramment employée pour des filtres de Kalman destinés à la navigation.

Présentation du CNN

Comme illustré dans la figure 3.5, à chaque instant discret k, le réseau de neurones convolutifs estime trois paramètres α_k, β_k et γ_k à partir des mesures $h_{s_k} \in \mathbb{R}^{3\times1}$ du magnétomètre dans le repère capteur $\boldsymbol{s}$ pour déterminer $cov\,(W_{h_s})$ et où les paramètres σ_x, σ_y et σ_z sont fixés à partir des spécifications du capteur.

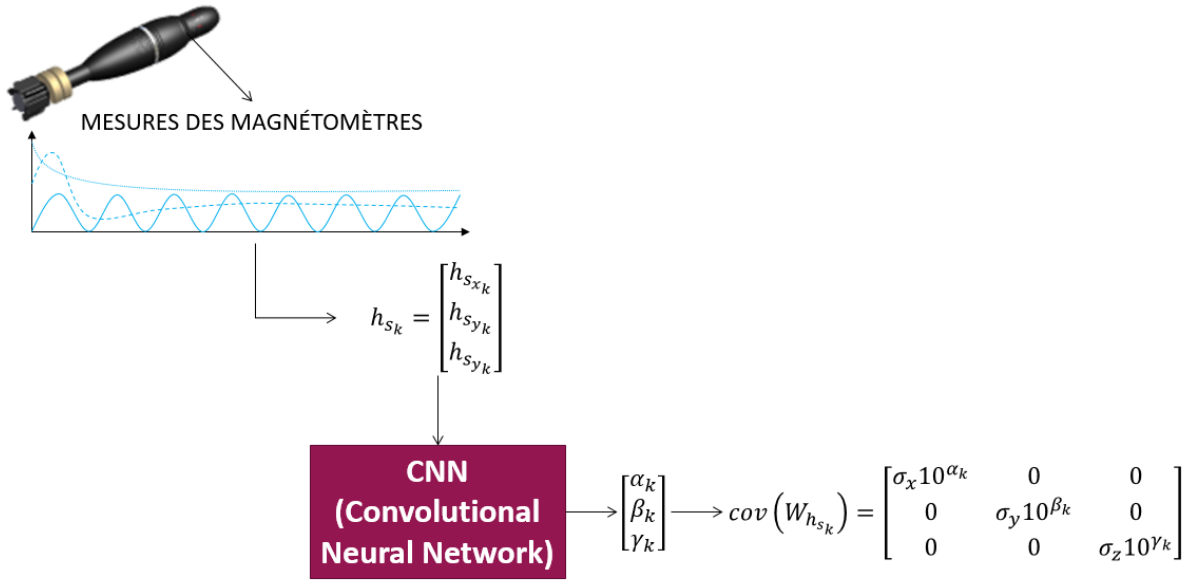

FIGURE 3.5 – Réglage de la covariance du bruit de mesures de l'Imp.RIEKF par un CNN.

En d'autres termes, les mesures de l'accéléromètre et du gyromètre sont utilisées pour l'étape de prédiction du filtre (3.22) - (3.29). Ensuite, le CNN estime $cov\left(W_{h_{s_k}}\right)$ à partir des mesures du magnétomètre. Enfin, les mesures du magnétomètre et les prédictions $cov\left(W_{h_{s_k}}\right)$ déterminées par le CNN sont utilisées dans l'étape de mise à jour du filtre pour estimer la position, la vitesse et l'orientation du projectile ainsi que les biais des capteurs inertiels. Le CNN permet ainsi de moduler les valeurs de la matrice de covariance du bruit de mesure en fonction de la phase de vol du projectile.

Détails d'entraînement du CNN

Le CNN est entraîné et testé sur des simulations de trajectoires de mortier de 120 mm générées par BALCO présenté dans la partie *(1.5)* de ce document. Le jeu de données d'entraînement se compose de 1000 simulations de trajectoires de mortier, le jeu de données de validation comprend 100 trajectoires et le jeu de données de test est composé de 100 trajectoires de mortier. La structure des couches du CNN implémenté pour ajuster la covariance du bruit des mesures des magnétomètres est présentée dans la figure 3.6.

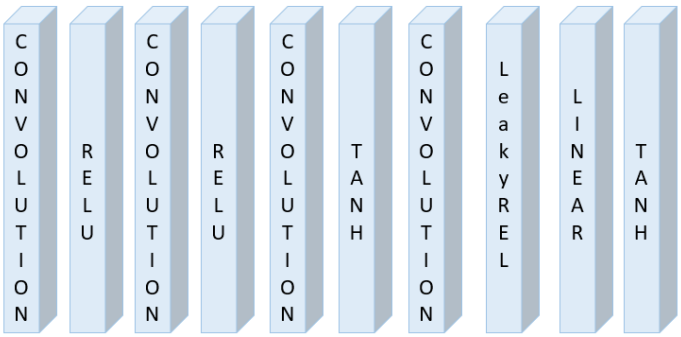

FIGURE 3.6 – Couches du CNN entraîné pour ajuster la covariance du bruit de mesure de l'Imp.RIEKF.

Des couches de convolution unidimensionnelles sont utilisées pour extraire les caractéristiques spatiales dans les données d'entrée. Comme pour tous les réseaux, des fonctions d'activation non linéaires sont appliquées : la fonction *ReLU (Rectified Linear Unit)*, la fonction *TanH (Hyperbolic Tangent)* et la fonction *LeakyReLU*, extension de la fonction *ReLU* permettant de prendre en compte des valeurs négatives. Les résultats des couches de convolution et des fonctions d'activation sont transmises aux couches entièrement connectées pour estimer les trois paramètres α, β et γ permettant de moduler la covariance du bruit de mesure des magnétomètres.

L'entraînement vise à ajuster les paramètres du CNN pour minimiser l'erreur entre les estimations obtenues avec l'Imp.RIEKF où la matrice de covariance du bruit de mesure est évaluée par le CNN et les données de référence fournies par BALCO. La fonction de perte employée lors de l'entraînement est l'erreur quadratique moyenne *(Mean Squared Error - MSE)*, couramment utilisée pour des problèmes de régression et définie telle que :

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(x_i - \widehat{x}_i)^2 \tag{3.40}$$

avec N le nombre de données, $x_i = \begin{bmatrix} p & v & \Psi \end{bmatrix}^T$ où Ψ est le vecteur comprenant respectivement les angles de roulis, de tangage et de lacet.
L'algorithme d'optimisation d'Adam [51] est utilisé pour apprendre les paramètres du CNN afin de minimiser la perte.

3.3.3 Analyse des résultats d'estimation

La contribution des réseaux de neurones pour optimiser des algorithmes de navigation est à présent évaluée. Dans un premier temps, l'utilisation conjointe du CNN pour ajuster l'Imp.RIEKF est analysée sur une trajectoire de mortier de 120 mm, puis, les observations sont généralisées sur les 100 simulations de trajectoires de mortier de 120 mm du jeu de données de test.

Analyse des performances pour une trajectoire de mortier de 120 mm

Afin de visualiser les performances d'estimation de l'algorithme proposé, trois méthodes d'estimation sont comparées :

1) l'algorithme de *Dead Reckoning*, c'est-à-dire, l'étape de prédiction de l'Imp.RIEKF décrit par les équations (3.22) - (3.25).
2) l'Imp.RIEKF où la matrice de covariance du bruit de mesure est constante pour les simulations du jeu de données de test, comme pour un filtre de Kalman classique.
3) l'Imp.RIEKF où la matrice de covariance du bruit de mesure est ajustée dynamiquement par le CNN présenté dans la partie *(3.3.2)*.

Les figures 3.7, 3.8 et 3.9 présentent les erreurs de position, de vitesse et d'orientation de l'algorithme de *Dead Reckoning* (méthode 1 en vert), de l'Imp.RIEKF (méthode 2 en noir) et de l'Imp.RIEKF ajusté par le CNN (méthode 3 en bleu) pour une trajectoire de mortier de 120 mm.

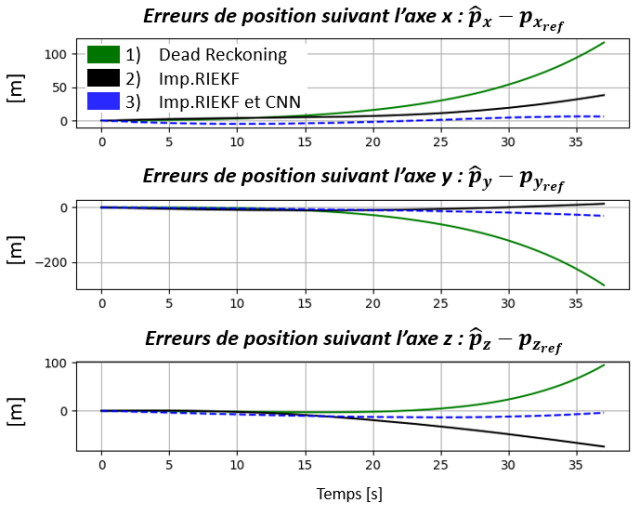

FIGURE 3.7 – Erreurs d'estimation de la position d'un mortier de 120 mm [m] obtenues par le *Dead Reckoning* (en vert), l'Imp.RIEKF avec une matrice de covariance constante (en noir) et l'Imp.RIEKF dont la matrice est ajustée par un CNN (en bleu).

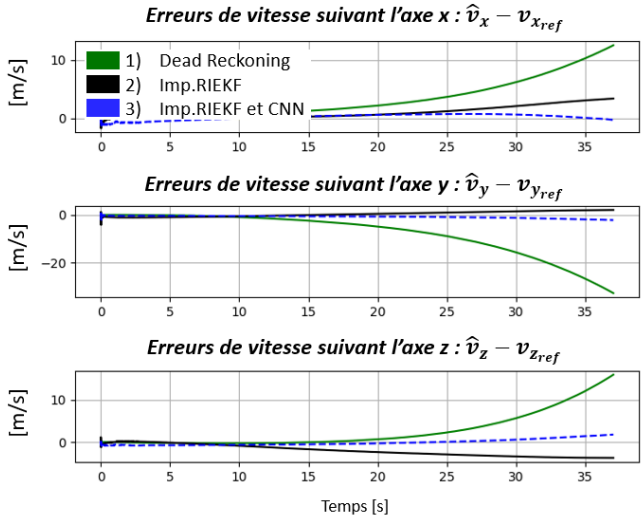

FIGURE 3.8 – Erreurs d'estimation de la vitesse d'un mortier de 120 mm [m/s] obtenues par le *Dead Reckoning* (en vert), l'Imp.RIEKF avec une matrice de covariance constante (en noir) et l'Imp.RIEKF dont la matrice est ajustée par un CNN (en bleu).

Les figures 3.7 - 3.9 montrent que les estimations basées sur l'Imp.RIEKF (méthodes 2 et 3) présentent des erreurs moins importantes que celles du *Dead Reckoning* (méthode 1). Ainsi, un filtre de Kalman basé sur une erreur d'estimation en partie non linéaire, un groupe de Lie matriciel et une mise à jour exponentielle permet d'optimiser les estimations

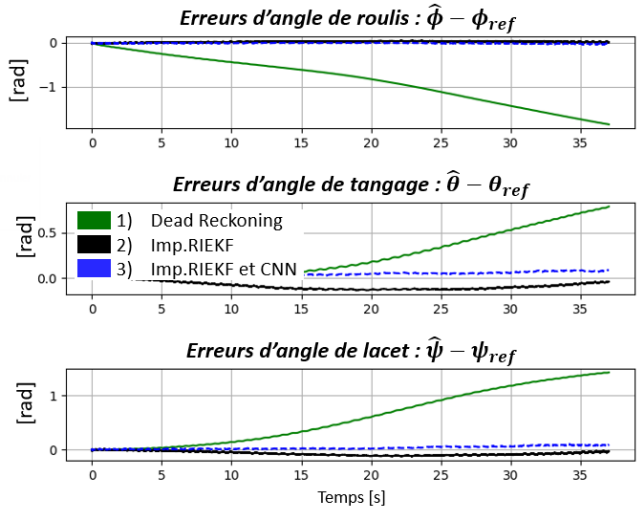

FIGURE 3.9 – Erreurs d'estimation des angles d'Euler d'un mortier de 120 mm [rad] obtenues par le *Dead Reckoning* (en vert), l'Imp.RIEKF avec une matrice de covariance constante (en noir) et l'Imp.RIEKF dont la matrice est ajustée par un CNN (en bleu).

de position, de vitesse et d'orientation. De plus, l'utilisation d'un CNN pour ajuster la covariance du bruit de mesure du filtre permet d'optimiser ces résultats. Toutefois, d'après les figures 3.7 et 3.8, les erreurs d'estimation de la position et de la vitesse suivant l'axe y sont significatives par rapport à la plage de variation de ces quantités le long de cet axe. En effet, les plages de variation de la position d'un mortier de 120 mm selon l'axe x et l'axe z sont de l'ordre de plusieurs kilomètres alors que la plage de variation de la position suivant l'axe y est de quelques mètres. Afin d'optimiser les estimations de position et de vitesse, une piste d'amélioration est d'utiliser une méthode de prétraitement des données d'entrée du réseau de neurones.

Résultats sur l'ensemble du jeu de données de test

Afin de valider ces observations, à savoir, que le réglage de la covariance du bruit de mesure d'un filtre par un CNN améliore les estimations, les performances globales des trois méthodes d'estimation sont évaluées sur l'ensemble du jeu de données de test. Pour cela, le critère d'évaluation considéré est l'erreur quadratique moyenne globale *(Root Mean Square Error - RMSE)*. Ce critère est évalué pour l'ensemble du jeu de données de test tel que :

$$\mathcal{C}_{RMSE} = \frac{1}{N_{simu}} \sum_{i=1}^{N_{simu}} \sqrt{\frac{1}{N} \sum_{k=1}^{N} \left(x_{k,ref} - \widehat{x}_k\right)^2} \tag{3.41}$$

avec N_{simu} le nombre de simulations du jeu de données de test, N le nombre d'échantillons de la simulation considérée, $x_{k,ref}$ la valeur de référence donnée par BALCO et $\widehat{x}_k$ la valeur estimée par l'algorithme considéré.

La figure 3.10 présente le critère d'erreur $\mathcal{C}_{RMSE}$ (3.41) évalué pour l'estimation des positions, des vitesses et des angles d'Euler par les trois méthodes considérées, à savoir l'étape de prédiction de l'Imp.RIEKF (méthode 1, en vert), l'Imp.RIEKF (méthode 2, en noir) et l'Imp.RIEKF réglé par le CNN (méthode 3, en bleu).

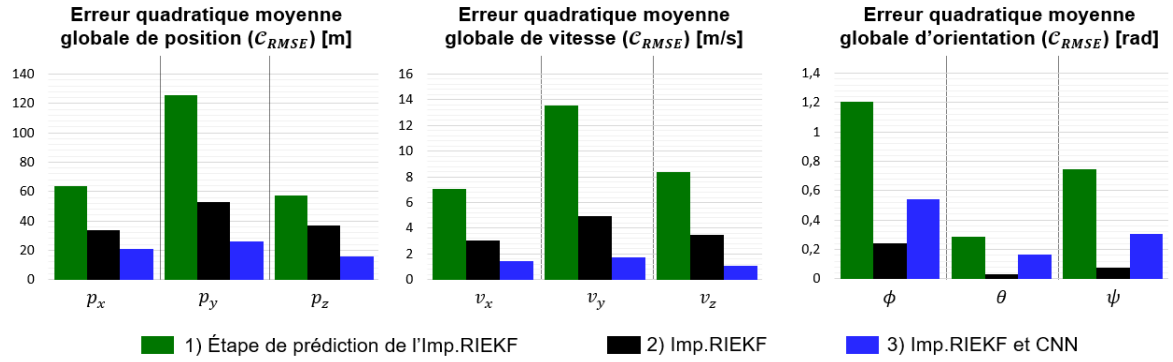

FIGURE 3.10 – Critère d'erreur $\mathcal{C}_{RMSE}$ (3.41) évalué pour l'estimation des positions, des vitesses et des angles d'Euler de mortier de 120 mm par le *Dead Reckoning* (méthode 1, en vert), l'Imp.RIEKF (méthode 2, en noir) et l'Imp.RIEKF réglé par le CNN (méthode 3, en bleu).

La figure 3.10 montre clairement que les méthodes basées sur l'Imp.RIEKF surpasse le *Dead Reckoning* pour l'estimation des positions, des vitesses et des angles d'Euler d'un mortier de 120 mm. En effet, l'Imp.RIEKF (méthode 2) améliore l'estimation des positions par rapport à l'étape de prédiction du filtre (méthode 1) selon les trois axes. De même pour l'estimation des vitesses du projectile.

De plus, la figure 3.10 confirme que l'utilisation d'un CNN pour régler la matrice de covariance du bruit de mesure de l'Imp.RIEKF (méthode 3) permet de réduire significativement les erreurs d'estimation de position et de vitesse par rapport à l'Imp.RIEKF dont la matrice de covariance est constante et réglée classiquement. En effet, les erreurs d'estimation des positions par l'Imp.RIEKF ajusté par le CNN sont de l'ordre de 20 m alors que celles de l'Imp.RIEKF avec une matrice de covariance constante sont d'environ 40 m. L'analyse des angles d'Euler est plus ambivalente comme le montre la figure 3.10. En effet, le CNN (méthode 3) détériore l'estimation des angles d'Euler par rapport au filtre avec une matrice constante (méthode 2). Plusieurs solutions peuvent être envisagées

pour améliorer l'estimation des angles d'Euler notamment la normalisation des données d'entrée du CNN afin que chaque mesure influe sur le réseau de façon équivalente.

Afin de visualiser l'apport d'une matrice de covariance variable dans le temps et ajustée aux différentes phases de vol du projectile, les figures 3.11 et 3.12 présentent l'erreur quadratique moyenne (RMSE) des vitesses et des angles estimés par la méthode 3 (Imp.RIEKF et CNN) en fonction de la RMSE obtenue par la méthode 2 (Imp.RIEKF) pour les N_{sim} simulations de l'ensemble de données de test.

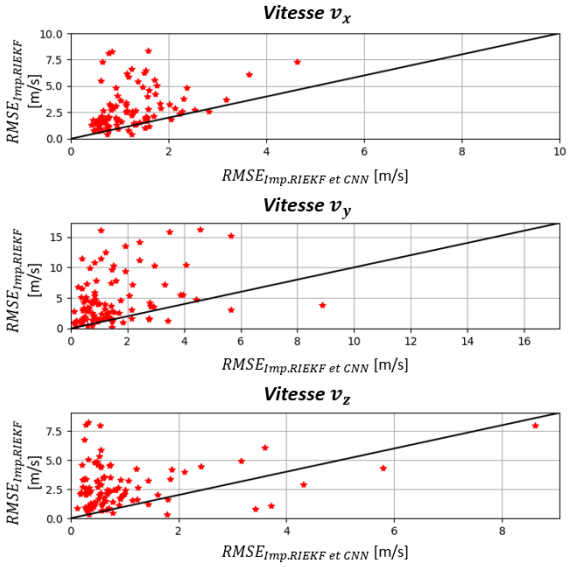

FIGURE 3.11 – Représentation des RMSE de la méthode 3 (Imp.RIEKF et CNN) en fonction des RMSE de la méthode 2 (Imp.RIEKF) pour l'estimation des vitesses des trajectoires de test.

La figure 3.11 met en évidence la contribution d'un réseau de neurones pour régler la matrice de covariance d'un filtre de Kalman. En effet, la plupart des marqueurs sont situés dans la partie supérieure de la figure 3.11, donc les erreurs d'estimation des vitesses obtenues par la méthode 3 (Imp.RIEKF et CNN) sont nettement inférieures aux erreurs obtenues par la méthode 2 (Imp.RIEKF). Cela implique que pour la majorité des trajectoires du jeu de données de test, le réglage de la matrice de covariance du bruit de mesure par un CNN améliore les estimations des vitesses du projectile. Des conclusions similaires sont observées pour l'estimation des positions.
La figure 3.12 confirme les premières observations. En effet, la plupart des marqueurs sont situés dans la partie inférieure de la figure donc les erreurs d'estimation des angles d'Euler par la méthode 3 (Imp.RIEKF et CNN) sont plus importantes que celles obtenues

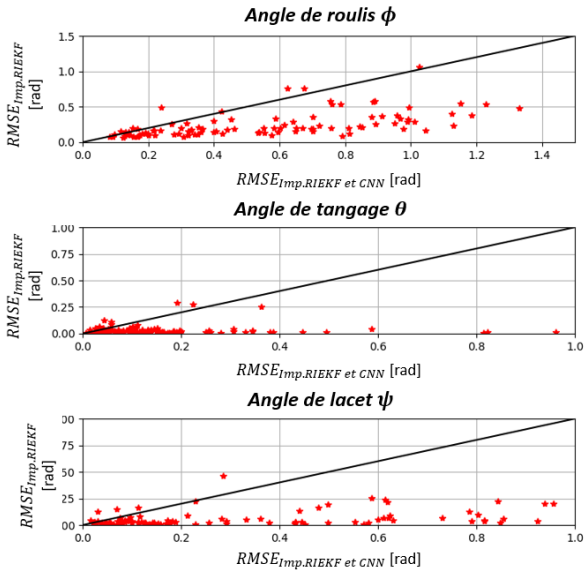

FIGURE 3.12 – Représentation des RMSE de la méthode 3 (Imp.RIEKF et CNN) en fonction des RMSE de la méthode 2 (Imp.RIEKF) pour l'estimation des angles d'Euler des trajectoires de test.

par méthode 2 (Imp.RIEKF). En d'autres termes, le réglage de la matrice de covariance de l'Imp.RIEKF par un CNN dégrade les estimations d'angle de roulis, de tangage et de lacet par rapport à un filtre sans réseau de neurones.

Ces constatations confirment les premières observations des figures 3.7 - 3.10. Le réglage de la covariance du bruit de mesure d'un Imp.RIEKF par un CNN est parfaitement adapté pour optimiser l'estimation des positions et des vitesses d'un mortier de 120 mm mais n'est pas approprié pour l'estimation des angles d'Euler de cette munition notamment à cause des variations de l'angle de roulis.

3.3.4 Méthode de prétraitement des données d'entrée

D'après l'analyse des résultats précédents, le CNN permet d'optimiser l'estimation des positions et des vitesses d'un projectile mais dégrade l'estimation des angles d'Euler par rapport à une solution classique de navigation. Afin d'améliorer ces résultats, deux méthodes de prétraitement des données d'entrée du CNN sont employées, à savoir : la normalisation des données d'entrée du réseau et la rotation du repère local de navigation. Ces deux méthodes visent à redéfinir les données sur des plages de variation similaires.

Normalisation des données d'entrée

La normalisation des données d'entrée d'un réseau de neurones est une approche de prétraitement des données qui vise à redimensionner les données d'entrée sur des plages de variation similaires tout en préservant la même distribution et les mêmes ratios que les données d'origine. Toutefois, la normalisation conduit à une perte d'information.
La normalisation des données d'entrée est utilisée pour éviter que certaines valeurs d'entrée influent davantage que d'autres lors de la phase d'apprentissage. En effet, des données d'entrée avec des plages de variation différentes peuvent entraîner une baisse des performances d'estimation du réseau. En d'autres termes, les petites valeurs ont une faible influence donc les poids du réseau sont mis à jour en fonction des valeurs d'entrée plus élevées. Cela peut conduire à une mise à jour importante des poids et donc à une convergence du réseau plus lente ou à une convergence vers un minimum local.

Deux types de normalisation sont étudiées : la normalisation Min/Max $MM(.)$ et la normalisation $STD(.)$.

- *La normalisation Min/Max*, notée $MM(.)$, est définie comme suit :

$$x_{MM} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{3.42}$$

  avec x_{max} et x_{min} respectivement le maximum et le minimum de la grandeur x à normaliser.
  Cette normalisation redéfinit les valeurs dans l'intervalle $[0, 1]$ et conserve les ratios entre les valeurs d'une grandeur. La normalisation Min/Max est dépendante des valeurs aberrantes comme elle dépend des extremums de la grandeur. Toutefois, dans le cas de l'estimation des trajectoires des projectiles, aucune valeur aberrante n'est permise comme les mesures simulées des capteurs sont bornées.
- *La normalisation STD*, notée $STD(.)$, est définie comme suit :

$$x_{STD} = \frac{x - \mu}{\sigma} \tag{3.43}$$

  avec x la quantité à normaliser, μ sa moyenne et σ son écart type. Ainsi x_{STD} est une quantité avec une moyenne nulle et un écart type de un.
  Cette normalisation est particulièrement utilisée pour les données d'entrée avec des unités différentes et est moins dépendante aux valeurs aberrantes que la normalisation Min/Max.

Rotation du repère de navigation

La seconde méthode de prétraitement des données mise en place est la rotation du repère local de navigation $\boldsymbol{n}$.
Cette méthode, illustrée dans la figure 3.13, vise à faire pivoter le repère local de navigation $\boldsymbol{n}$ d'un angle fixe $\boldsymbol{\gamma}$. En d'autres termes, une grandeur dans le repère local de navigation $\boldsymbol{n}$ s'exprime dans le repère local de navigation tourné $\boldsymbol{n_\gamma}$ de la façon suivante :

$$x_\gamma = R_\gamma x \tag{3.44}$$

avec $x \in \mathbb{R}^{3\times1}$ le vecteur initial dans le repère local de navigation $\boldsymbol{n}$, $x_\gamma \in \mathbb{R}^{3\times1}$ ce même vecteur exprimé dans le repère de navigation tourné $\boldsymbol{n_\gamma}$ et $R_\gamma \in SO(3)$ la matrice de rotation du repère local de navigation $\boldsymbol{n}$ au repère local de navigation tourné $\boldsymbol{n_\gamma}$ définie telle que :

$$R_\gamma = \begin{bmatrix} cos(\gamma) & -sin(\gamma) & 0 \\ sin(\gamma) & cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos(\gamma) & 0 & sin(\gamma) \\ 0 & 1 & 0 \\ -sin(\gamma) & 0 & cos(\gamma) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\gamma) & -sin(\gamma) \\ 0 & sin(\gamma) & cos(\gamma) \end{bmatrix} \tag{3.45}$$

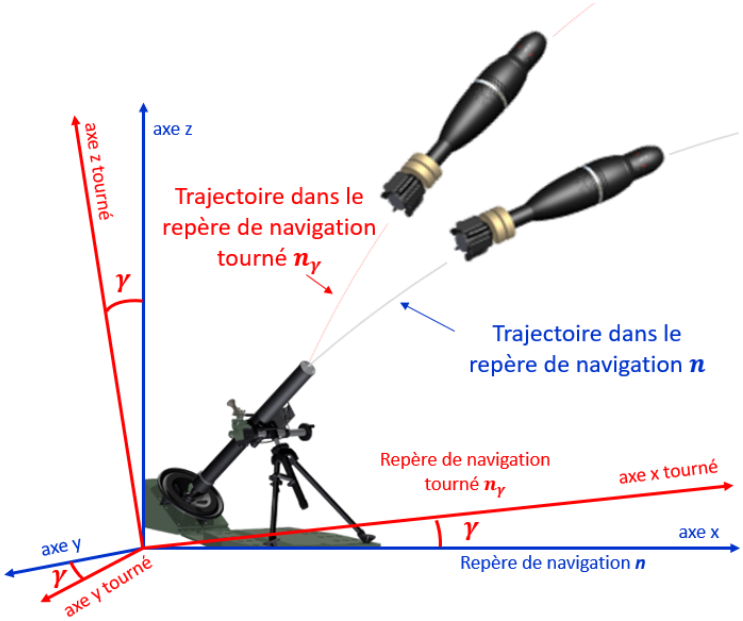

FIGURE 3.13 – Repère local de navigation $\boldsymbol{n}$ et repère local de navigation tourné $\boldsymbol{n_\gamma}$.

La rotation du repère de navigation permet à un vecteur $x_\gamma \in \mathbb{R}^{3\times1}$ exprimé dans le repère local de navigation tourné $\boldsymbol{n}_\gamma$ d'avoir ses trois composantes définies sur des plages de variation similaires. En d'autres termes, la rotation du repère de navigation permet d'obtenir des plages de variation semblables d'une grandeur selon les trois axes. Cette approche est utilisée afin de s'assurer que le CNN estime de façon équivalente la matrice de covariance dans le cas où une grandeur estimée est de très faible amplitude par rapport aux deux autres.

Cette méthode est mise en place pour améliorer les estimations des positions et des vitesses suivant l'axe longitudinal. En effet, les plages de variation de la position d'un mortier de 120 mm suivant les axes x et z sont de plusieurs kilomètres, tandis que suivant l'axe y, sa position varie de quelques mètres. Comme illustré sur la figure 3.14, la position d'un projectile exprimée dans le repère local de navigation tourné $\boldsymbol{n}_\gamma$ a des amplitudes similaires le long des trois axes contrairement à la position exprimée dans le repère de navigation local $\boldsymbol{n}$.

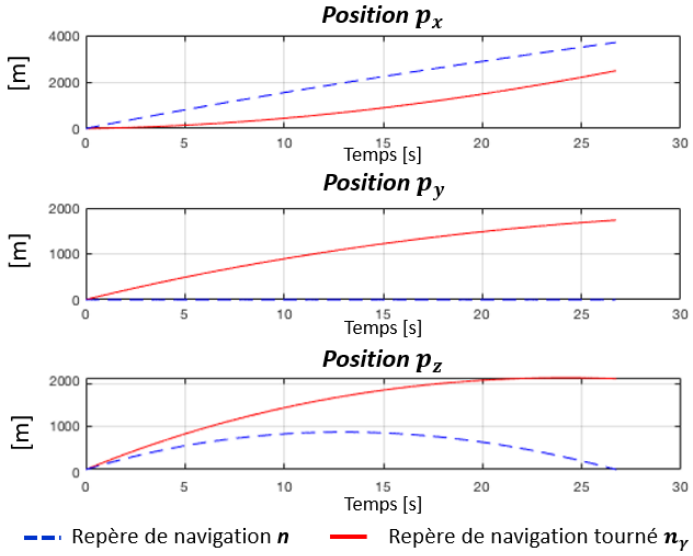

FIGURE 3.14 – Influence de la rotation du repère local de navigation sur la position d'un mortier de 120 mm : position exprimée dans le repère local de navigation $\boldsymbol{n}$ (ligne bleue), position exprimée dans le repère local de navigation tourné $\boldsymbol{n}_\gamma$ (ligne rouge).

Toutes les grandeurs définies initialement dans $\boldsymbol{n}$ sont exprimées dans le repère local de navigation tourné $\boldsymbol{n}_\gamma$, c'est-à-dire la position p, la vitesse v et les angles d'Euler Ψ du projectile. En d'autres termes, lors de la phase d'apprentissage, les trajectoires sont prédites dans le repère local de navigation tourné $\boldsymbol{n}_\gamma$. Ces estimations sont ensuite comparées

aux données de référence également exprimées dans le repère local de navigation tourné $\boldsymbol{n}_\gamma$ afin de rétropropager la perte et mettre à jour les paramètres du réseau. Cela permet ainsi d'évaluer la perte entre les positions, les vitesses et les angles estimés et les données de référence dans le cas où les plages de variations sont équivalentes suivant les trois axes. Lors de la phase de test, les trajectoires sont prédites dans le repère local de navigation tourné $\boldsymbol{n}_\gamma$, puis ces estimations sont exprimées dans le repère local de navigation initial $\boldsymbol{n}$ par la rotation inverse. Cette approche a donc les mêmes objectifs que la normalisation.

L'angle de rotation du repère de navigation $\boldsymbol{\gamma}$ est fixe pour toutes les trajectoires du jeu de données et est le même pour exprimer la position, la vitesse et les angles d'Euler dans le repère local de navigation tourné $\boldsymbol{n}_\gamma$. Cet angle de rotation $\boldsymbol{\gamma}$ est déterminé en fonction des données utilisées dans ces travaux, notamment afin d'obtenir des plages de variation similaires pour les trois positions et par conséquent, pour les trois vitesses.

Influence de la normalisation et de la rotation du repère de navigation sur la précision des estimations

L'influence de la normalisation des données d'entrée du CNN et l'impact de la rotation du repère de navigation sur la précision des estimations sont analysés par plusieurs CNN entraînés avec les spécifications décrites dans la partie *(3.3.2)*. À cette fin, six CNN sont considérés pour estimer la matrice de covariance du bruit de mesure de l'Imp.RIEKF et dont les caractéristiques sont présentées dans le tableau 3.1.

TABLE 3.1 – Spécifications des CNN considérés pour estimer la matrice de covariance du bruit de mesure de l'Imp.RIEKF.

NOM	**NORMALISATION**	**ROTATION**
CNN	NON	NON
CNN_{MM}	$MM(h_s)$	NON
CNN_{STD}	$STD(h_s)$	NON
CNN_R	NON	OUI
$CNN_{R_{MM}}$	$MM(h_s)$	OUI
$CNN_{R_{STD}}$	$STD(h_s)$	OUI

Afin d'évaluer l'influence de la normalisation et de la rotation du repère de navigation sur la précision des estimations, les performances des Imp.RIEKF ajustés par les réseaux mentionnés ci-dessus sont évaluées par deux critères d'erreur qui sont :

- le score, noté $\mathcal{C}_{score}$ qui représente le nombre de trajectoires où l'erreur quadratique moyenne (RMSE) de l'Imp.RIEKF ajusté par le CNN considéré est strictement inférieure à l'erreur quadratique moyenne de l'Imp.RIEKF avec une matrice de covariance constante. En d'autres termes, $\mathcal{C}_{score}$ est évalué tel que :

$$\mathcal{C}_{score} = \sum_{k=1}^{N_{sim}} \left(RMSE\left(\hat{x}_{CNN}\right) < RMSE\left(\hat{x}_{CST}\right) \right) \tag{3.46}$$

avec N_{sim} est le nombre de simulations du jeu de données de test, $\hat{x}_{CNN}$ l'estimation de l'Imp.RIEKF ajusté par le CNN considéré, et $\hat{x}_{CST}$ l'estimation de l'Imp.RIEKF dont la matrice de covariance est constante.

- le taux d'erreur, noté $\mathcal{C}_e$ qui représente les erreurs de l'Imp.RIEKF ajusté par le CNN considéré par rapport aux erreurs de l'Imp.RIEKF sans réseaux de neurones. Le taux d'erreur $\mathcal{C}_e$ est défini tel que :

$$\mathcal{C}_e = 100 \times \frac{C_{RMSE_{CNN}}}{C_{RMSE_{CST}} + C_{RMSE_{CNN}}} \tag{3.47}$$

avec $C_{RMSE_{CST}}$ définit par l'équation (3.41), l'erreur quadratique moyenne globale de l'Imp.RIEKF avec une matrice de covariance constante, et $C_{RMSE_{CST}}$ l'erreur quadratique moyenne globale de l'Imp.RIEKF réglé par le CNN considéré.

Les figures 3.15 et 3.16 représentent le $\mathcal{C}_{score}$ (3.46) et le $\mathcal{C}_e$ (3.47) évalués pour les Imp.RIEKF ajustés par le CNN, CNN_{MM}, CNN_{STD}, CNN_R, $CNN_{R_{MM}}$ et $CNN_{R_{STD}}$.

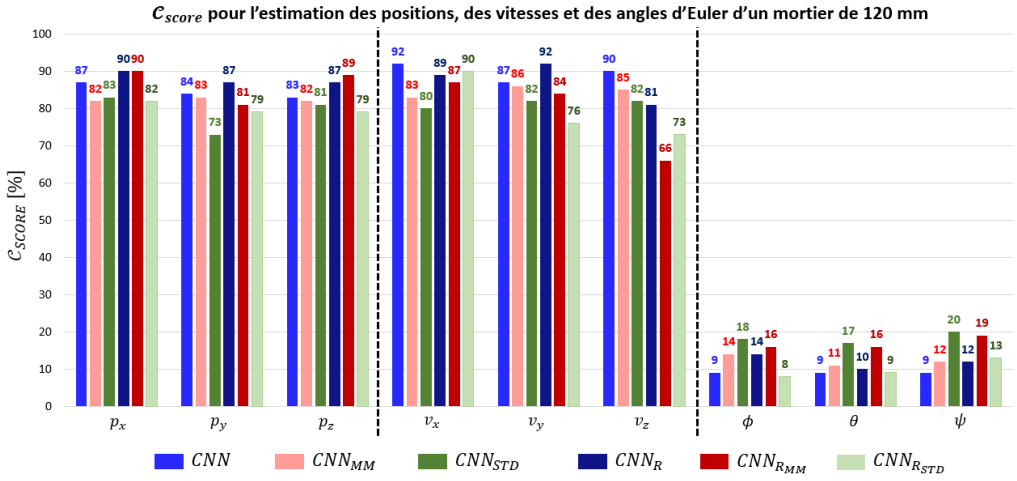

FIGURE 3.15 – Critère $\mathcal{C}_{score}$ (3.46) évalué pour les réseaux CNN, CNN_{MM}, CNN_{STD}, CNN_R, $CNN_{R_{MM}}$ et $CNN_{R_{STD}}$.

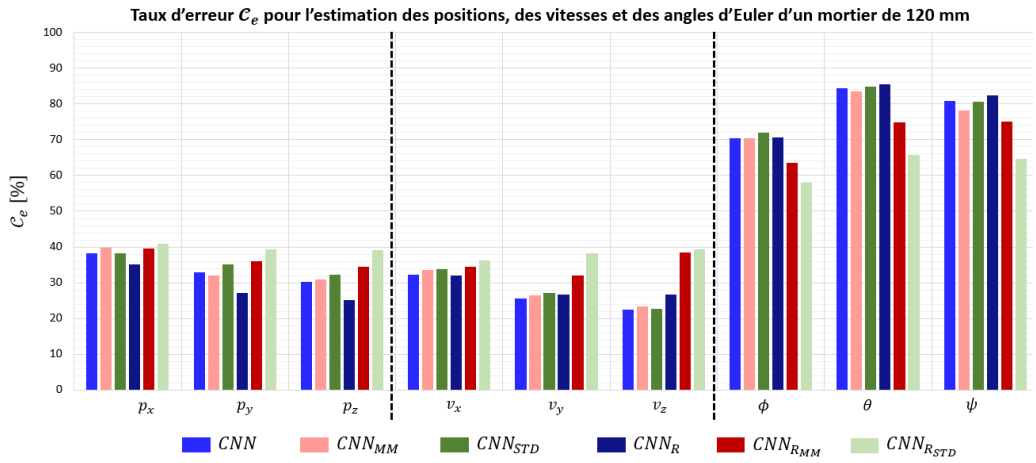

FIGURE 3.16 – Critère $\mathcal{C}_e$ (3.47) évalué pour les réseaux CNN, CNN_{MM}, CNN_{STD}, CNN_R, $CNN_{R_{MM}}$ et $CNN_{R_{STD}}$.

D'après les figures 3.15 - 3.16, l'ensemble des Imp.RIEKF ajustés par des CNN optimisent l'estimation des positions et des vitesses du projectile contrairement à l'estimation des angles d'Euler. De plus, la rotation du repère de navigation améliore significativement l'estimation des positions et des vitesses du projectile par rapport aux estimations sans rotation du repère de navigation. En effet, les scores obtenus par l'Imp.RIEKF ajusté par le CNN_R sont supérieurs à ceux obtenus par l'Imp.RIEKF ajusté par le CNN. Cette observation est validée par l'analyse du taux d'erreur $\mathcal{C}_e$ qui est moins importante pour le CNN_R et le CNN. Ainsi, la rotation du repère de navigation permet de réduire principalement les erreurs d'estimation des positions et des vitesses du projectile, bien que cette solution nécessite un temps de calcul plus important. La normalisation des données d'entrée du CNN affecte les estimations de position, de vitesse et d'orientation par rapport aux résultats sans normalisation. En effet, les normalisations Min/Max et STD dégradent l'estimation des positions et des vitesses du projectile par rapport au réseau sans normalisation. Ces observations sont confirmées par les scores $\mathcal{C}_{score}$ et les taux d'erreur $\mathcal{C}_e$ des Imp.RIEKF ajustés par les CNN_{MM} et CNN_{STD} qui sont plus importants que ceux obtenus pour l'Imp.RIEKF ajusté par le CNN. Les mêmes observations peuvent être formulées concernant les réseaux $CNN_{R_{MM}}$ et $CNN_{R_{STD}}$. Toutefois, la normalisation des données d'entrée du CNN permet de réduire les erreurs d'estimation des angles d'Euler.

Les résultats rapportés dans cette partie montrent que le réglage de la matrice de covariance du bruit de mesure d'un Imp.RIEKF par un CNN permet d'optimiser principalement les estimations des positions et des vitesses d'un mortier de 120 mm par rapport

aux estimations obtenues par l'Imp.RIEKF où la matrice de covariance est constante. Toutefois, cette solution n'est pas optimale pour réduire les erreurs d'estimation des angles d'Euler de cette munition. Afin de réduire les erreurs d'estimation, deux méthodes de prétraitement des données sont étudiées : la normalisation des données d'entrée et la rotation du repère de navigation. Ces deux méthodes de prétraitement redéfinissent des plages de variation similaires pour une grandeur suivant les trois axes. Les résultats présentés dans cette partie montrent que la rotation du repère de navigation est une solution appropriée pour optimiser les estimations d'un mortier de 120 mm. À l'inverse, la normalisation des données d'entrée détériore l'estimation des positions et des vitesses mais améliore légèrement l'estimation des angles d'Euler par rapport à une solution sans réseau de neurones.

3.4 Réglage de la covariance du bruit de mesures par un CNN : application à un EKF et à un IEKF

D'après les résultats obtenus dans la partie précédente, le réglage de la matrice de covariance du bruit de mesure d'un filtre de Kalman par un CNN semble adapté à l'estimation de la trajectoire d'un projectile. Toutefois, cette approche n'est pas optimale pour l'estimation des angles d'Euler. Afin de valider ces affirmations et d'évaluer dans quelles mesures cette approche est applicable à la navigation des projectiles, cette méthode d'optimisation est appliquée à deux autres filtres de Kalman aux propriétés différentes.

3.4.1 Application à un filtre de Kalman Étendu

Le réglage de la matrice de covariance du bruit de mesure d'un filtre de Kalman par un CNN est appliquée à un filtre de Kalman Étendu *(Extended Kalman Filter - EKF)*. L'EKF considéré, testé en vol et présenté dans [9] ainsi que dans le chapitre *5* de ce document, vise à estimer l'angle et la vitesse de roulis d'un projectile caractérisé par un tir tendu à partir des deux magnétomètres radiaux embarqués dans la munition. Plusieurs formes de la matrice de covariance du bruit de mesure sont considérées afin d'évaluer leurs pertinences sur la précision des estimations obtenues.

Présentation générale de l'EKF

Comme présenté dans la figure 3.17, à partir des mesures des deux magnétomètres radiaux embarqués notés respectivement H_a et H_b et de la connaissance du champ ma-

gnétique de référence H_n dans le repère local de navigation $\boldsymbol{n}$, l'EKF vise à estimer :

- l'angle de roulis ϕ et la vitesse de roulis ω_c d'un projectile caractérisé par un tir tendu dans le repère local de navigation $\boldsymbol{n}$. Ce type de trajectoire signifie que les variations des angles de tangage et de lacet de la munition sont négligeables.
- les amplitudes A_{H_a} et A_{H_b} des modèles des mesures des deux magnétomètres radiaux embarqués,
- les biais b_{H_a} et b_{H_b} des modèles des mesures des magnétomètres.

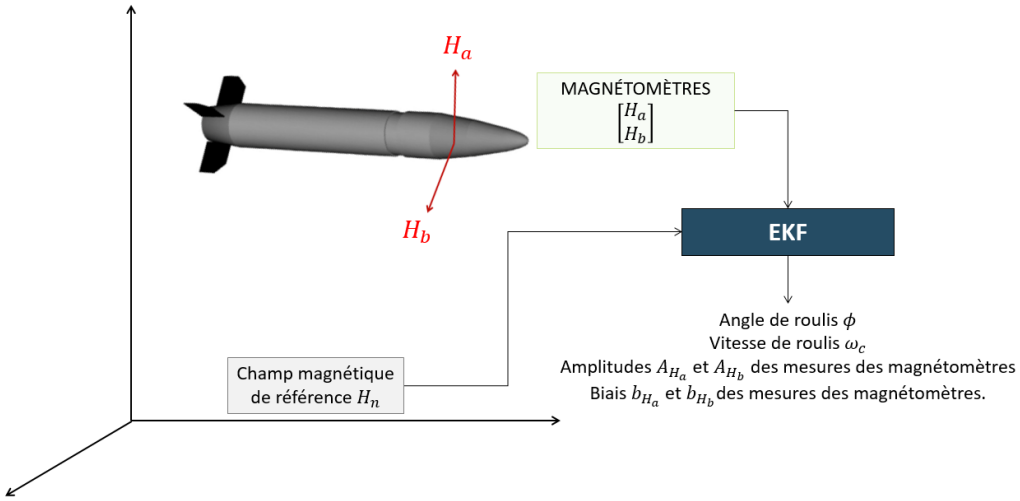

FIGURE 3.17 – Principe de fonctionnement de l'EKF pour estimer l'angle de roulis et la vitesse de roulis d'une munition caractérisée par un tir tendu à partir des deux magnétomètres embarqués et de la connaissance du champ magnétique de référence dans le repère de navigation.

Les mesures des deux magnétomètres radiaux embarqués, notées respectivement H_a et H_b, sont modélisées par des amplitudes A_{H_a} et A_{H_b}, des biais b_{H_a} et b_{H_b} et des bruits blancs gaussiens W_{H_a} et W_{H_b} tels que :

$$\tilde{H}_a = A_{H_a} H_a + b_{H_a} + W_{H_a}, \quad \tilde{H}_b = A_{H_b} H_b + b_{H_b} + W_{H_b}, \tag{3.48}$$

avec $\tilde{H}_a$ et $\tilde{H}_b$ les modèles des deux magnétomètres, et H_a et H_b les mesures des capteurs.

L'EKF vise à estimer les états suivants :

$$x = \begin{bmatrix} \phi^T & \omega_c^T & b_{H_a}^T & A_{H_a}^T & b_{H_b}^T & A_{H_b}^T \end{bmatrix}^T \in \mathbb{R}^{6\times 1}. \tag{3.49}$$

Les prédictions sont corrigées par des observations qui sont l'estimation des mesures

des magnétomètres à partir du champ magnétique de référence. Le modèle de mesure est alors :

$$\begin{bmatrix} H_a \\ H_b \end{bmatrix} = \begin{bmatrix} A_{H_a} H_{env} sin(\phi + \lambda) + b_{H_a} + W_{H_a} \\ A_{H_b} H_{env} cos(\phi + \lambda) + b_{H_b} + W_{H_b} \end{bmatrix} \tag{3.50}$$

avec W_{H_a} et W_{H_b} les bruits des mesures, et H_{env} et λ des constantes déterminées à partir des valeurs du champ magnétique de référence H_n.

Le projectile considéré dans cette partie est un projectile de 40 mm caractérisé par un tir tendu. Dans ce cas, les angles de tangage et de lacet ont une dynamique négligeable par rapport à celle de l'angle de roulis, de sorte que $\theta \approx \psi \approx 0$. De plus, les dynamiques des biais et des amplitudes des magnétomètres sont caractérisées par des mouvements browniens. Ainsi, la dynamique d'évolution des états (3.49) est donnée par :

$$\dot{\phi} = \omega_c + W_\phi, \quad \dot{\omega}_c = W_{\omega_c} \tag{3.51}$$

$$\dot{A}_{H_a} = W_{A_{H_a}}, \quad \dot{b}_{H_a} = W_{b_{H_a}}, \quad \dot{A}_{H_b} = W_{A_{H_b}}, \quad \dot{b}_{H_b} = W_{b_{H_b}} \tag{3.52}$$

avec W_ϕ, W_{ω_c}, $W_{A_{H_a}}$, $W_{b_{H_a}}$, $W_{A_{H_b}}$ et $W_{b_{H_b}}$ des bruits blancs supposés gaussiens.

Étape de prédiction de l'EKF : d'après les modèles d'évolution (3.51) - (3.52) associés aux états (3.49), l'étape de prédiction à temps discret de l'EKF est donnée par :

$$\begin{aligned} \widehat{x}_{k|k-1} &= A_d \widehat{x}_{k-1|k-1}, \\ P_{k|k-1} &= A_d P_{k-1|k-1} A_d^T + Q_k \end{aligned} \tag{3.53}$$

avec $A_d = (2I_{6\times6} - A_t\Delta t)^{-1}(2I_{6\times6} - A_t\Delta t)$, avec $A_t = \begin{bmatrix} 0 & 1 & 0 & & & \\ & & & 0 & 0 & 0 \\ & & & 0_{6\times6} & & \end{bmatrix}$, avec Δt la période d'échantillonnage des magnétomètres, et avec Q_k la matrice de covariance du bruit de modèle définie telle que $Q_k = cov\left(W_\phi, W_{\omega_c}, W_{b_{H_a}}, W_{A_{H_a}}, W_{b_{H_b}}, W_{A_{H_b}}\right)$.

Étape de mise à jour de l'EKF : d'après le modèle d'observation (3.50) et les mesures des magnétomètres H_a et H_b, les états prédits $\widehat{x}_{k|k-1}$ et la matrice de covariance $P_{k|k-1}$

prédite sont mis à jour par les équations suivantes :

$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1} \tag{3.54}$$

$$\widehat{x}_{k|k} = \widehat{x}_{k|k-1} + K_k \left(\begin{bmatrix} H_a \\ H_b \end{bmatrix} - \begin{bmatrix} \widehat{A}_{H_{a_{k|k-1}}} H_{env} sin(\widehat{\phi}_{k|k-1} + \lambda) + \widehat{b}_{H_{a_{k|k-1}}} \\ \widehat{A}_{H_{b_{k|k-1}}} H_{env} cos(\widehat{\phi}_{k|k-1} + \lambda) + \widehat{b}_{H_{b_{k|k-1}}} \end{bmatrix} \right) \tag{3.55}$$

$$P_{k|k} = (I - K_k H_k) P_{k|k-1} \tag{3.56}$$

avec $R_k = cov(W_{H_a}, W_{H_b})$ la matrice de covariance du bruit des mesures des magnétomètres et avec H_k la matrice d'observation déterminée par la linéarisation du modèle d'observation de sorte que :

$$H_k = \begin{bmatrix} \widehat{A}_{H_{a_k}} H_{env} cos(\widehat{\phi}_k + \lambda) & 0 & 1 & H_{env} sin(\widehat{\phi}_k + \lambda) & 0 & 0 \\ -\widehat{A}_{H_{b_k}} H_{env} sin(\widehat{\phi}_k + \lambda) & 0 & 0 & 0 & 1 & H_{env} cos(\widehat{\phi}_k + \lambda) \end{bmatrix} \tag{3.57}$$

Suivant le même principe que celui présenté dans la partie *(3.3.2)* de ce document, un CNN est entraîné pour estimer la matrice de covariance du bruit de mesure de l'EKF. Ainsi, à partir des mesures des deux magnétomètres radiaux H_a et H_b, le CNN estime la matrice de covariance du bruit de mesure de la forme :

$$R = \begin{bmatrix} \sigma_{H_a} \times 10^{\alpha} & 0 \\ 0 & \sigma_{H_b} \times 10^{\beta} \end{bmatrix} \tag{3.58}$$

avec σ_{H_a} et σ_{H_b} les paramètres de la covariance des mesures des magnétomètres déterminés d'après des tests en vol et avec α et β les paramètres estimés par le CNN.

Résultats préliminaires d'estimation de l'angle et de la vitesse de roulis

Les figures 3.18 et 3.19 présentent l'estimation et les erreurs obtenues pour l'estimation de l'angle de roulis et de la vitesse de roulis d'un projectile de 40 mm par l'EKF présenté précédemment sans réseau de neurones (en vert), et l'EKF dont la matrice de covariance du bruit de mesure est déterminée par un CNN (en bleu).

Les figures 3.18 et 3.19 montrent que l'EKF ajusté par un CNN (en bleu) et l'EKF avec une matrice de covariance constante (en vert) présentent des performances similaires pour l'estimation de l'angle de roulis et de la vitesse de roulis d'un projectile de 40 mm. En effet, les erreurs obtenues pour l'estimation de cette trajectoire sont du même ordre de grandeur avec les deux méthodes considérées. Ainsi, le CNN ne semble pas optimiser

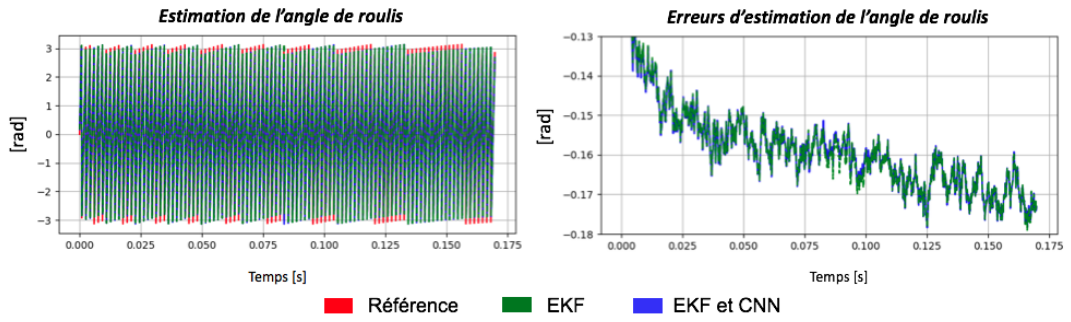

FIGURE 3.18 – Estimation de l'angle de roulis d'un projectile de 40 mm par l'EKF (en vert) et l'EKF ajusté par un CNN (en bleu).

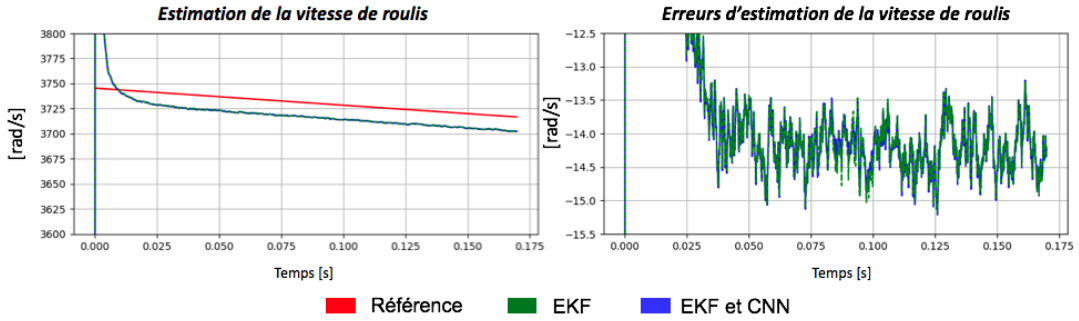

FIGURE 3.19 – Estimation de la vitesse de roulis d'un projectile de 40 mm par l'EKF (en vert) et l'EKF ajusté par un CNN (en bleu).

l'estimation des angles et des vitesses de roulis d'un projectile caractérisé par un tir tendu.

Performances globales d'estimation

Afin de visualiser si l'ajustement dynamique de la matrice de covariance du bruit de mesure de l'EKF par un CNN permet d'optimiser l'estimation de l'angle et de la vitesse de roulis d'un projectile de 40 mm, la figure 3.20 présente les erreurs quadratiques moyennes (RMSE) de l'EKF ajusté par le CNN en fonction des RMSE obtenues par l'EKF sans réseau de neurones pour les N_{sim} simulations de l'ensemble de données de test.

La figure 3.20 montre que les marqueurs sont situés aux abords de la ligne centrale pour l'estimation de l'angle et de la vitesse de roulis du projectile. Donc, les erreurs quadratiques moyennes de l'EKF ajusté par le CNN sont similaires à celles de l'EKF avec une matrice de covariance constante. En d'autres termes, le CNN ne permet pas d'optimiser l'estimation de l'angle et de la vitesse de roulis d'un projectile de 40 mm.

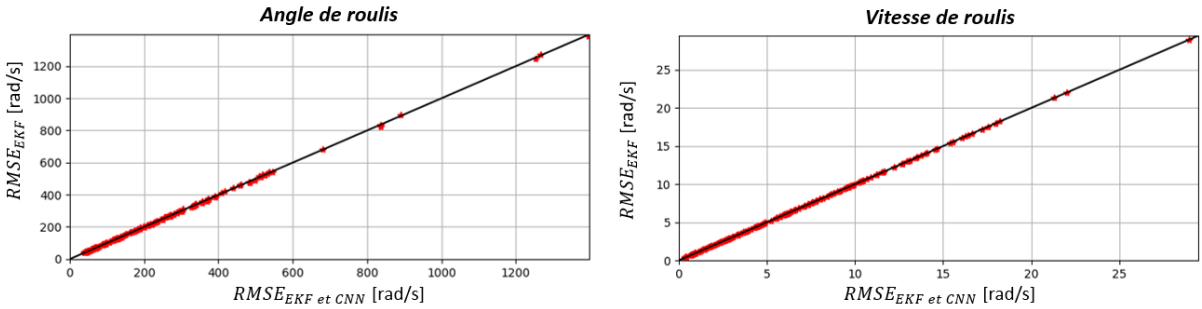

FIGURE 3.20 – Représentation des RMSE de l'EKF ajusté par le CNN en fonction des RMSE de l'EKF avec une matrice de covariance constante pour l'estimation de l'angle et de la vitesse de roulis d'un projectile de 40 mm.

Afin de confirmer ces observations, le score $\mathcal{C}_{score}$ (3.46) et le taux d'erreur $\mathcal{C}_e$ (3.47) sont évalués pour l'EKF ajusté par le CNN (en bleu) et pour l'EKF sans réseau de neurones (en vert) et sont présentés dans la figure 3.21.

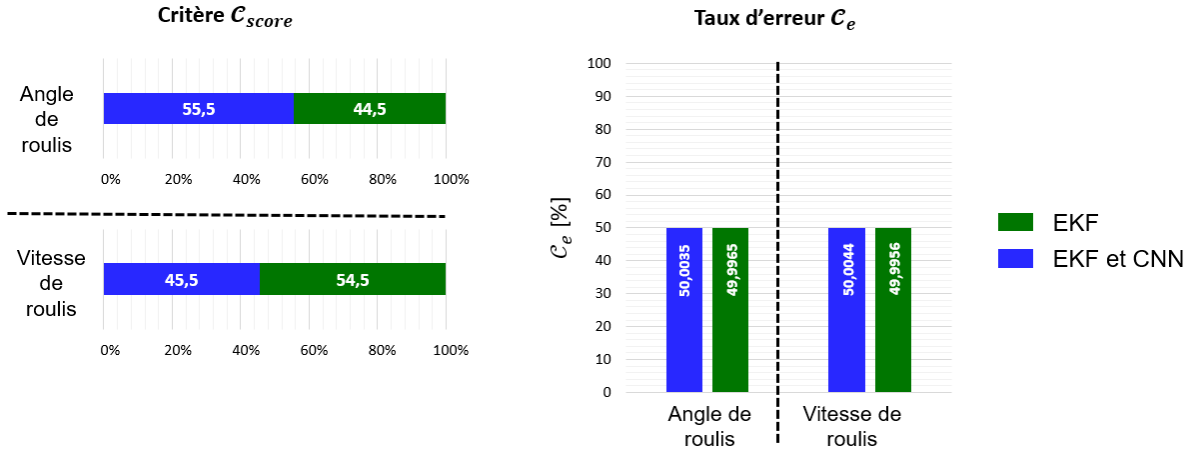

FIGURE 3.21 – Représentation du critère d'évaluation $\mathcal{C}_{score}$ (3.46) et du taux d'erreur $\mathcal{C}_e$ (3.47) pour l'estimation de l'angle et de la vitesse de roulis par l'EKF ajusté par le CNN (en bleu) et l'EKF avec une matrice de covariance constante (en vert).

La figure 3.21 confirme les observations précédentes. Le CNN ne permet pas de réduire les erreurs d'estimation de l'angle et de la vitesse de roulis d'un projectile de 40 mm par rapport à une méthode standard de réglage de la matrice de covariance. En effet, les scores $\mathcal{C}_{score}$ et les taux d'erreur $\mathcal{C}_e$ sont semblables pour l'EKF ajusté par un CNN et pour l'EKF avec une matrice de covariance constante. Cela signifie que l'ajustement dynamique de la covariance du bruit de mesure de l'EKF influe peu sur les estimations obtenues.

Ces performances d'estimation mitigées peuvent s'expliquer par le fait que l'EKF est déjà réglé de façon optimale avec les paramètres de covariance σ_{H_a} et σ_{H_b}, et donc, l'usage d'un CNN ne peut réduire les erreurs d'estimation. Une seconde explication est que le CNN ne parvient pas à optimiser l'estimation de l'angle de roulis du fait de la forte dynamique associée à cet angle.

Solutions d'optimisation proposées

Afin d'expliquer les résultats obtenus, différentes formes de la matrice de covariance du bruit de mesure sont étudiées. Pour cela, trois CNN sont considérés :

- le CNN présenté précédemment qui estime les paramètres α et β pour moduler la covariance du bruit de mesure de sorte que :

$$R = \begin{bmatrix} \sigma_{H_a} \times 10^{\alpha} & 0 \\ 0 & \sigma_{H_b} \times 10^{\beta} \end{bmatrix} \tag{3.59}$$

- le CNN_{square} qui estime la covariance du bruit de mesure de la forme :

$$R_{square} = \begin{bmatrix} \alpha^2 & 0 \\ 0 & \beta^2 \end{bmatrix} \tag{3.60}$$

- le CNN_{abs} qui estime la covariance du bruit de mesure de la forme :

$$R_{abs} = \begin{bmatrix} |\alpha| & 0 \\ 0 & |\beta| \end{bmatrix} \tag{3.61}$$

Les valeurs au carré et les valeurs absolues sont utilisées afin que le CNN estime des valeurs strictement positives pour s'assurer que l'EKF ne diverge pas.

La figure 3.22 présente les critères de score $\mathcal{C}_{score}$ (3.46) et de taux d'erreur $\mathcal{C}_e$ (3.47) pour les EKF ajustés par les réseaux CNN (3.59), CNN_{square} (3.60), CNN_{abs} (3.61).

La figure 3.22 montre que la forme de la matrice de covariance du bruit de mesure estimée par le CNN influe sur la précision des estimations. En effet, les CNN_{square} (3.60) et CNN_{abs} (3.61) présentent de moins bons résultats que le CNN (3.59) pour l'estimation de l'angle et de la vitesse de roulis d'un projectile de 40 mm. Toutefois, d'après le taux d'erreur $\mathcal{C}_e$, les réseaux CNN et CNN_{abs} présentent des performances semblables pour l'estimation de la vitesse de roulis. Les résultats présentés dans la figure 3.22 indiquent que le CNN ne converge pas autour des valeurs constantes de covariance σ_{H_a} et σ_{H_b} qui

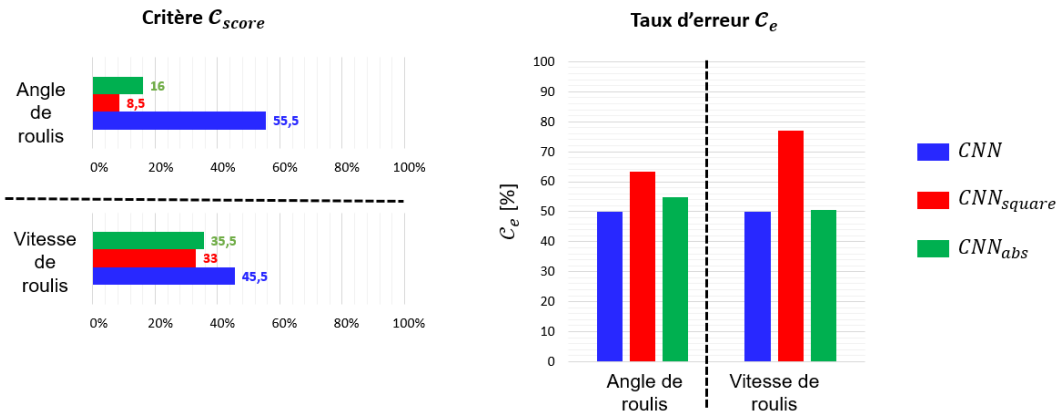

FIGURE 3.22 – Représentation du critère d'évaluation $\mathcal{C}_{score}$ (3.46) et du taux d'erreur $\mathcal{C}_e$ (3.47) pour les EKF ajustés par les réseaux CNN (3.59) (en bleu), CNN_{square} (3.60) (en rouge), CNN_{abs} (3.61) (en vert) pour l'estimation de l'angle et de la vitesse de roulis.

présentent les les meilleures performances d'estimation. Donc, la méthode proposée ne permet pas, dans le cas de l'estimation de l'angle et de la vitesse de roulis, de converger vers des valeurs optimales de covariance du bruit de mesures pour ces deux grandeurs.

Au vu des résultats obtenus, la forme de la matrice de covariance du bruit de mesure présentant les meilleures performances est celle décrite par l'équation (3.59). Afin d'optimiser ces résultats, et d'après les variations importantes de l'angle de roulis d'un projectile de 40 mm, le réseau CNN présenté dans l'équation (3.59) est entraîné avec différentes fonctions de perte. En effet, la fonction de perte évalue les erreurs entre les prédictions du réseau de neurones et les données de référence afin de minimiser ces erreurs lors de la phase d'apprentissage. La fonction de perte permet donc de distinguer une bonne prédiction d'une moins bonne. Pour cela, trois réseaux CNN sont considérés pour visualiser l'influence du choix de la fonction de perte sur la précision des estimations :

- le CNN qui estime la matrice de covariance présentée par l'équation (3.59) et dont l'Erreur Quadratique Moyenne *(Mean Squared Error - MSE)* est employée lors de la phase d'apprentissage telle que :

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(x_i - \widehat{x}_i)^2 \tag{3.62}$$

- le CNN_{MAE} qui estime une matrice de covariance de la même forme que précédemment et où la fonction de perte employée est l'Erreur Moyenne Absolue *(Mean*

Absolute Error - MAE) définie telle que :

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|x_i - \widehat{x}_i|^2 \tag{3.63}$$

- le CNN_{Huber} qui estime une matrice de covariance décrite par l'équation (3.59) et où la fonction de perte employée est la perte d'Huber *(Huber loss)* définie telle que :

$$Huber_{Loss} = \begin{cases} \frac{1}{2}(x-\widehat{x})^2 & pour \quad |x-\widehat{x}| \leq \delta \\ \delta|x-\widehat{x}| - \frac{1}{2}\delta^2 & pour \quad |x-\widehat{x}| > \delta \end{cases} \tag{3.64}$$

La figure 3.23 présente les critères de score $\mathcal{C}_{score}$ (3.46) et de taux d'erreur $\mathcal{C}_e$ (3.47) pour les EKF ajustés par les réseaux CNN (3.59) (en bleu), CNN_{MAE} (en rouge) et CNN_{Huber} (en vert) pour l'estimation de l'angle et de la vitesse de roulis d'un projectile de 40 mm.

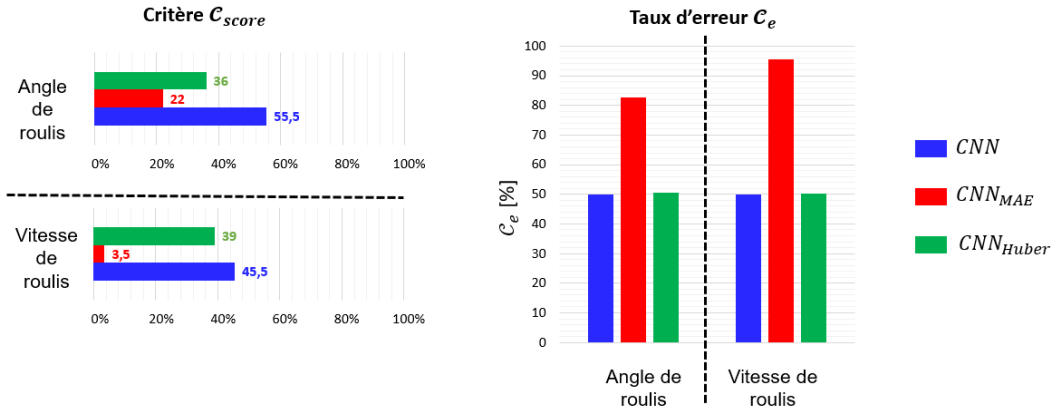

FIGURE 3.23 – Représentation du critère d'évaluation $\mathcal{C}_{score}$ (3.46) et du taux d'erreur $\mathcal{C}_e$ (3.47) pour les EKF ajustés par les réseaux CNN (3.59) (en bleu), CNN_{MAE} (en rouge) et CNN_{Huber} (en vert) pour l'estimation de l'angle et de la vitesse de roulis d'un projectile de 40 mm.

La figure 3.23 indique la fonction de perte influe sur la précision des estimations obtenues. En effet, l'EKF ajusté par le CNN_{MAE} (en rouge) présente les scores les plus faibles et les taux d'erreurs les plus importants pour l'estimation de l'angle et de la vitesse de roulis. Ainsi, cette fonction de perte n'est pas adaptée à l'estimation de l'angle de roulis et de la vitesse de roulis d'un projectile de 40 mm comme cette fonction est préférée lorsque les données comprennent des valeurs aberrantes. À l'inverse, les EKF ajustés par les réseaux CNN (en bleu) et CNN_{Huber} (en vert) présentent des résultats semblables,

bien que la perte d'Huber semble légèrement dégrader les estimations de l'angle et de la vitesse de roulis par rapport à l'EKF entraîné avec le CNN.

Les résultats présentés dans cette partie révèlent que le réglage dynamique de la matrice de covariance du bruit de mesure d'un EKF ne permet pas d'optimiser l'estimation de l'angle et de la vitesse de roulis d'un projectile de 40 mm par rapport à un EKF avec une matrice de covariance constante. Toutefois, l'EKF ajusté par le CNN ne dégrade pas les estimations. De plus, malgré différentes formes de la matrice de covariance du bruit et différentes fonctions de perte, aucune méthodes n'a permis de surpasser l'EKF sans réseau de neurones pour l'estimation de l'angle et de la vitesse de roulis. Une piste d'explication est que ce filtre de Kalman est peu optimisable du fait que les valeurs de covariance sont suffisamment ajustées et qu'il est difficile d'améliorer ces résultats déjà existants.

3.4.2 Application à un IEKF corrigé par les vitesses GPS

D'après les résultats rapportés dans ce chapitre, le réglage de la covariance du bruit de mesure d'un Imp.RIEKF permet d'optimiser principalement les estimations des positions et des vitesses d'un projectile contrairement à un EKF ajusté par un CNN qui ne permet pas de réduire les erreurs d'estimation de l'angle et de la vitesse de roulis d'un projectile. Afin d'expliquer les performances obtenues sur l'EKF et pour vérifier si des tirs tendus ou l'estimation d'un angle avec de fortes variations ne sont pas les causes de ces faibles performances, un LIEKF *(Left-Invariant Extended Kalman Filter)* dont la matrice de covariance du bruit de mesure est ajustée dynamiquement par un CNN est implémenté. Le LIEKF utilisé vise à estimer les positions, les vitesses et les angles d'Euler d'un mortier de 120 mm à partir des mesures des accéléromètres et des gyromètres embarqués dans la munition ainsi que de la vitesse mesurée par un GPS. Les données BALCO ne produisant pas de mesures GPS, les vitesses sont générées d'après les vitesses de référence auxquelles un bruit est ajouté.

Présentation générale du LIEKF

Comme présenté dans la figure 3.24, le LIEKF *(Left-Invariant Extended Kalman Filter)* vise à estimer l'orientation $R \in SO(3)$, la vitesse $v \in \mathbb{R}^{3\times1}$ et la position $p \in \mathbb{R}^{3\times1}$ d'un projectile dans le repère de navigation $\boldsymbol{n}$ à partir des mesures des accéléromètres et des gyromètres effectuées dans le repère capteur $\boldsymbol{s}$ ainsi que de la vitesse du projectile mesurée par un GPS dans le repère de navigation $\boldsymbol{n}$.

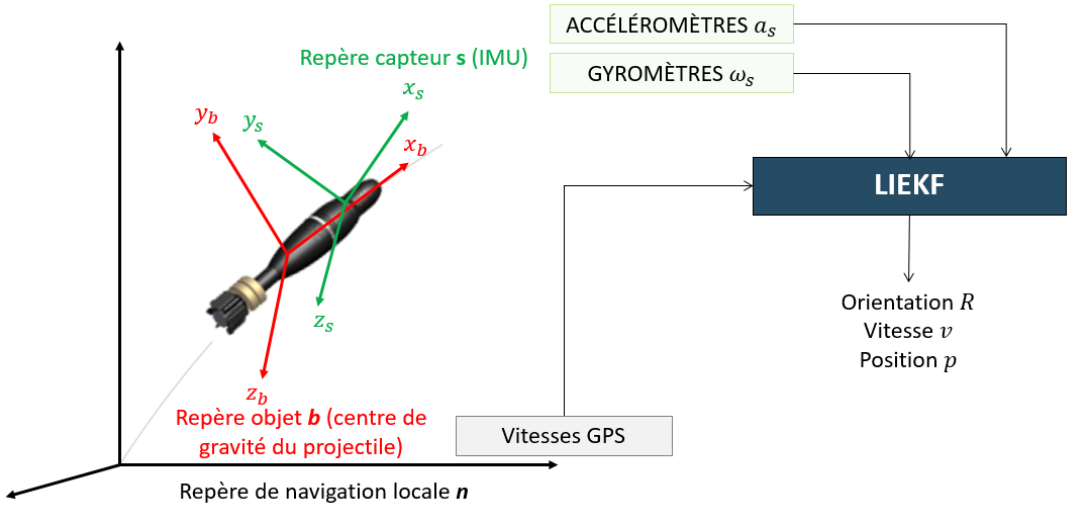

FIGURE 3.24 – Principe de fonctionnement du LIEKF *(Left-Invariant Extended Kalman Filter)* pour estimer l'orientation, la vitesse et la position d'un mortier de 120 mm à partir des mesures des accéléromètres et les gyromètres embarqués ainsi que de la vitesse du projectile mesurée par un GPS.

Dans le cas d'un IEKF, les biais des capteurs inertiels ne sont pas pris en compte comme aucun groupe de Lie matriciel ne permet de modéliser ces dynamiques tout en vérifiant la propriété *affine de groupe*. Ainsi, les modèles des mesures des capteurs inertiels sont :

$$\tilde{\omega}_s = \omega_s + W_\omega \qquad \tilde{a}_s = a_s + W_a \tag{3.65}$$

avec $\tilde{\omega}_s$ et $\tilde{a}_s \in \mathbb{R}^{3\times1}$ le modèle de mesure des gyromètres et des accéléromètres dans le repère capteur $\boldsymbol{s}$, ω_s et $a_s \in \mathbb{R}^{3\times1}$ les mesures des gyromètres et des accéléromètres dans le repère capteur $\boldsymbol{s}$, et W_ω et $W_a \in \mathbb{R}^{3\times1}$ des bruits blancs gaussiens.

La dynamique d'évolution du projectile en temps continu est donnée par :

$$\dot{R} = R[\tilde{\omega}_s - W_\omega]_\times, \qquad \dot{v} = R(\tilde{a}_s - W_a) + g, \qquad \dot{p} = v \tag{3.66}$$

avec $[.]_\times$ l'opérateur linéaire du groupe de Lie $SO(3)$, et g le vecteur gravité.

Le LIEKF vise à estimer, à partir d'une configuration initiale donnée, l'orientation, la vitesse et la position du projectile représenté par une matrice du groupe de Lie matriciel

$SE_2(3)$ car $R \in SO(3)$, $v \in \mathbb{R}^{3\times1}$ et $p \in \mathbb{R}^{3\times1}$:

$$\widehat{\chi}_t = \begin{bmatrix} \widehat{R} & \widehat{v} & \widehat{p} \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix} \in SE_2(3) \tag{3.67}$$

Le système d'évolution associé aux états $\widehat{\chi}_t$ s'écrit sous forme matricielle telle que :

$$\begin{aligned} \tfrac{d}{dt}\widehat{\chi}_t &= \begin{bmatrix} \widehat{R}[\tilde{\omega}_s]_\times & \widehat{R}\tilde{a}_s + g & \widehat{v} \\ 0_{1\times3} & 0 & 0 \\ 0_{1\times3} & 0 & 0 \end{bmatrix} - \begin{bmatrix} \widehat{R} & \widehat{v} & \widehat{p} \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix} \begin{bmatrix} [W_\omega]_\times & W_a & 0_{3\times1} \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix} \\ &= f_{u_t}(\widehat{\chi}_t) - \widehat{\chi}_t \mathcal{L}_{\mathfrak{g}}(\widehat{w}_t) \end{aligned} \tag{3.68}$$

L'application $f_{u_t}(.)$ (3.68) est *affine de groupe*. Par conséquent, d'après le théorème 2.2.1 *(Dynamique d'erreur autonome)*, la dynamique d'erreur invariante est indépendante des états du système.

L'erreur invariante à gauche associée aux états $\widehat{\chi}_t$ (3.67) est évaluée telle que :

$$\begin{aligned} \eta_{\chi t} = \chi_t^{-1}\widehat{\chi}_t &= \begin{bmatrix} R & v & p \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} \widehat{R} & \widehat{v} & \widehat{p} \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} R^T\widehat{R} & R^T\widehat{v} - R^Tv & R^T\widehat{p} - R^Tp \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix} \end{aligned} \tag{3.69}$$

L'erreur invariante à gauche $\eta_{\chi t}$(3.69) est linéarisée sur l'espace euclidien associé tel que l'erreur linéarisée $\xi_t = \begin{bmatrix} \xi_R^T & \xi_v^T & \xi_p^T \end{bmatrix}^T \in \mathbb{R}^{9\times1}$ soit définie par :

$$\eta_{\chi t} = exp(\,\xi_t) = exp_m(\mathcal{L}_{\mathfrak{g}}(\xi_t)) \approx I_{5\times5} + \mathcal{L}_{\mathfrak{g}}(\xi_t) \approx I_{5\times5} + \begin{bmatrix} [\xi_R]_\times & \xi_v & \xi_p \\ 0_{1\times3} & 0 & 0 \\ 0_{1\times3} & 0 & 0 \end{bmatrix} \tag{3.70}$$

L'erreur invariante d'orientation η_R, de vitesse η_v et de position η_p sont alors identifiées et linéarisées de la façon suivante :

$$\eta_R = R^T\widehat{R} \approx I_{3\times3} + [\xi_R]_\times \qquad \eta_v = R^T\widehat{v} - R^Tv \approx \xi_v \qquad \eta_p = R^T\widehat{p} - R^Tp \approx \xi_p \tag{3.71}$$

Étape de prédiction du LIEKF : la prédiction de l'état et de la matrice de covariance de l'erreur linéarisée en temps continu est donnée par les équations suivantes :

$$\begin{aligned} \tfrac{d}{dt}\widehat{\chi_t} &= f_{u_t}(\widehat{\chi_t}) \\ \tfrac{d}{dt}P_t &= A_t P_t + P_t {A_t}^T + \widehat{Q}_t \end{aligned} \tag{3.72}$$

avec $f_{u_t}(.)$ la dynamique d'évolution définie par l'équation (3.68), et avec $A_t \in \mathbb{R}^{9\times 9}$ la matrice d'évolution et $\widehat{Q}_t \in \mathbb{R}^{9\times 9}$ la matrice de covariance du bruit de modèle tel que :

$$A_t = \begin{bmatrix} -[\tilde{\omega}_s]_\times & 0_{3\times 3} & 0_{3\times 3} \\ -[\tilde{a}_s]_\times & -[\tilde{\omega}_s]_\times & 0_{3\times 3} \\ 0_{3\times 3} & I_{3\times 3} & -[\tilde{\omega}_s]_\times \end{bmatrix}, \qquad \widehat{Q}_t = cov(\widehat{w}_t) = cov\left(\begin{bmatrix} W_\omega \\ W_a \\ 0_{3\times 1} \end{bmatrix}\right) \tag{3.73}$$

D'après le théorème 2.2.1 *(Dynamique d'erreur autonome)*, la dynamique $f_{u_t}(.)$ est *affine de groupe*(2.19) donc la dynamique d'évolution de l'erreur invariante $\frac{d}{dt}\eta_t$ est indépendante de l'estimation $\widehat{\chi}_t$. Cette propriété est vérifiée par la matrice A_t (3.73) qui est indépendantes des états χ_t estimés.

L'étape de prédiction du LIEKF à temps discret avec Δt la période d'échantillonnage des capteurs est alors :

$$\begin{aligned} \widehat{R}_{k|k-1} &= \widehat{R}_{k-1|k-1} exp_{SO(3)}([\tilde{\omega}_{s_k}]\Delta t) \\ \widehat{v}_{k|k-1} &= \widehat{v}_{k-1|k-1} + (\widehat{R}_{k-1|k-1}\tilde{a}_{s_k} + g)\Delta t \\ \widehat{p}_{k|k-1} &= \widehat{p}_{k-1|k-1} + \widehat{v}_{k-1|k-1}\Delta t + \tfrac{1}{2}(\widehat{R}_{k-1|k-1}\tilde{a}_{s_k} + g)\Delta t^2 \\ P_{k|k-1} &= \Phi_k P_{k-1|k-1} {\Phi_k}^T + Q_k \end{aligned} \tag{3.74}$$

avec $\Phi_k = exp_m\left(A_t \Delta t\right) = \sum_{k=0}^{+\infty} \frac{(A_t\Delta t)^k}{k!}$ et $Q_k \approx \Phi_k Q_t \Phi_k^T \Delta t$.

Étape de mise à jour du LIEKF : les observations considérées pour mettre à jour les prédictions du LIEKF sont la vitesse du projectile mesurée par le GPS qui sont invariantes à gauche et qui s'écrivent donc sous la forme :

$$Y = \begin{bmatrix} y_{tk} \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} R & v & p \\ 0_{1\times 3} & 1 & 0 \\ 0_{1\times 3} & 0 & 1 \end{bmatrix} \begin{bmatrix} 0_{3\times 1} \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} W_y \\ 0 \\ 0 \end{bmatrix} \tag{3.75}$$

avec y_{tk} la vitesse du projectile, $d = \begin{bmatrix} 0_{3\times 1} & 1 & 0 \end{bmatrix}^T \in \mathbb{R}^{5\times 1}$, et $W_y \in \mathbb{R}^{3\times 1}$ le bruit de mesure.

La matrice de gain de Kalman est définie par l'équation suivante :

$$K_k = P_{k|k-1}H^T(HP_{k|k-1}H^T + \widehat{N}_k)^{-1} \tag{3.76}$$

avec $\widehat{N}_k$ la matrice de covariance du bruit de mesure modifié tel que $\widehat{N}_k = \widehat{R}^{-1}cov(W_y) \in \mathbb{R}^{3\times3}$ et H la matrice d'observation telle que $H = \begin{bmatrix}0_{3\times3} & I_{3\times3} & 0_{3\times3}\end{bmatrix} \in \mathbb{R}^{3\times9}$.

D'après la théorie des IEKF, la mise à jour de l'état estimé par un filtre LIEKF est la suivante :

$$\widehat{\chi}_{k|k} = \widehat{\chi}_{k|k-1}exp\left(K_k(\widehat{\chi}_{k|k-1}^{-1}Y - b)\right) = \begin{bmatrix}\widehat{R}_{k|k-1} & \widehat{v}_{k|k-1} & \widehat{p}_{k|k-1}\\ 0_{1\times3} & 1 & 0\\ 0_{1\times3} & 0 & 1\end{bmatrix}\begin{bmatrix}\xi_{Rk|k} & \xi_{vk|k} & \xi_{pk|k}\\ 0_{1\times3} & 1 & 0\\ 0_{1\times3} & 0 & 1\end{bmatrix} \tag{3.77}$$

Donc, la mise à jour des états estimés à partir des observations à temps discret est :

$$\begin{aligned}\widehat{R}_{k|k} = \widehat{R}_{k|k-1}\xi_{Rk|k}, \quad \widehat{v}_{k|k} = \widehat{R}_{k|k-1}\xi_{vk|k} + \widehat{v}_{k|k-1},\\ \widehat{p}_{k|k} = \widehat{R}_{k|k-1}\xi_{pk|k} + \widehat{p}_{k|k-1}\\ P_{k|k} = (I_{9\times9} - K_kH)P_{k|k-1}\end{aligned} \tag{3.78}$$

Le jeu de données de trajectoires de projectiles BALCO présenté dans la partie *(1.5)* de ce document ne dispose pas de mesures GPS. Afin de générer de telles mesures, les vitesses de référence sont bruitées dans le but de produire des mesures de la forme :

$$\begin{bmatrix}v_{x_{GPS}}\\ v_{y_{GPS}}\\ v_{z_{GPS}}\end{bmatrix} = \begin{bmatrix}v_{x_{ref}}\\ v_{y_{ref}}\\ v_{z_{ref}}\end{bmatrix} + \begin{bmatrix}\Sigma_{v_x}\\ \Sigma_{v_y}\\ \Sigma v_z\end{bmatrix} \tag{3.79}$$

avec v_{ref} les vitesses de référence fournies par BALCO et Σ_{v_x}, Σ_{v_y} et Σ_{v_z} des distributions aléatoires normales.

Suivant le même principe que celui présenté dans la partie *(3.3.2)*, un CNN est entraîné pour estimer trois paramètres α, β et γ à partir des vitesses GPS pour moduler la covariance du bruit de mesure du LIEKF définie telle que :

$$R = \begin{bmatrix}\sigma_{v_x} \times 10^{\alpha} & 0 & 0\\ 0 & \sigma_{v_y} \times 10^{\beta} & 0\\ 0 & 0 & \sigma_{v_z} \times 10^{\gamma}\end{bmatrix} \tag{3.80}$$

Résultats préliminaires d'estimation

Les figures 3.25 et 3.26 présentent l'estimation des positions et des angles d'Euler d'un mortier de 120 mm ainsi que les erreurs obtenues par le LIEKF présenté précédemment sans réseau de neurones (en noir), et le LIEKF dont la matrice de covariance du bruit de mesure est déterminée par un CNN (en bleu).

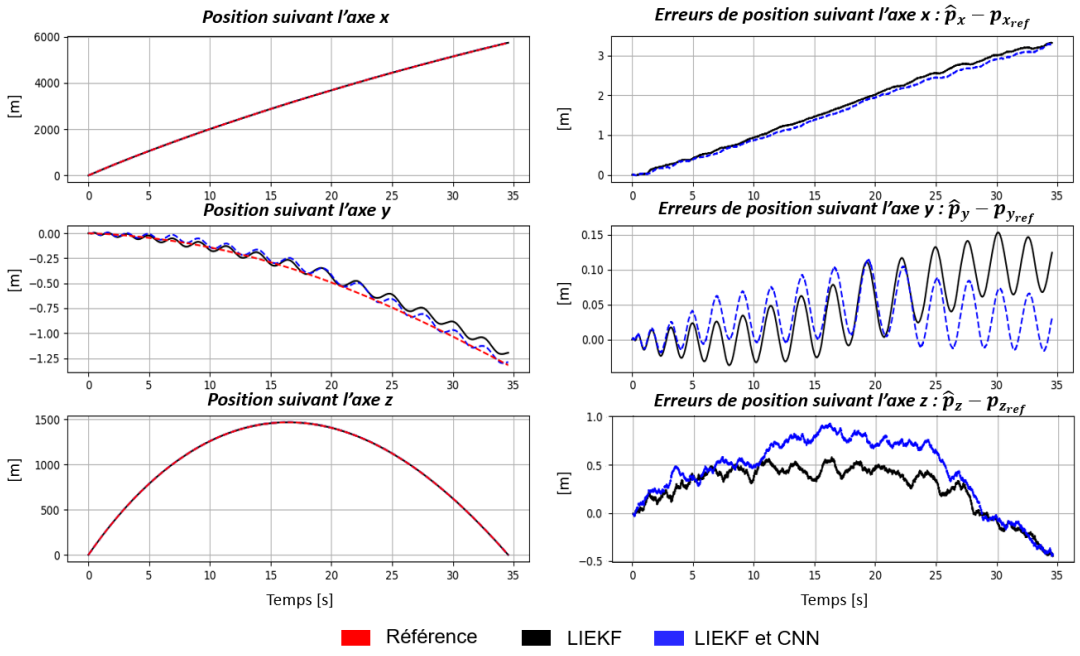

FIGURE 3.25 – Estimation de la position d'un mortier de 120 mm et erreurs associées obtenues par le LIEKF (en noir) et le LIEKF dont la matrice de covariance du bruit de mesure est déterminée par un CNN (en bleu).

Les figures 3.25 et 3.26 indiquent que le LIEKF (en noir) et le LIEKF ajusté par un CNN (en bleu) présentent des performances semblables pour l'estimation des positions et des angles d'Euler du mortier de 120 mm. Les mêmes observations peuvent être formulées pour la vitesse. Toutefois, ces deux méthodes d'estimation présentent plusieurs différences.

Performances globales

Afin de visualiser si l'ajustement de la matrice de covariance du bruit de mesure du LIEKF par un CNN optimise l'estimation des positions, des vitesses et des angles d'Euler, les figures 3.27 et 3.28 présentent le score $\mathcal{C}_{score}$ (3.46) et le taux d'erreur $\mathcal{C}_e$ (3.47) de l'EKF ajusté par le CNN (en bleu) et de l'EKF sans réseau de neurones (en noir).

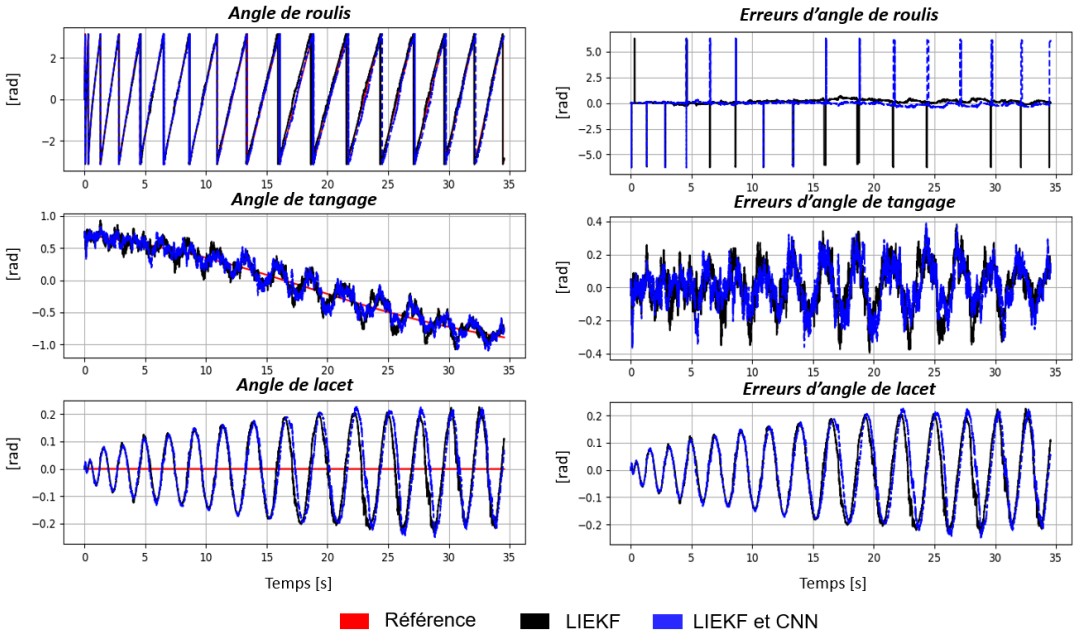

FIGURE 3.26 – Estimation des angles d'Euler d'un mortier de 120 mm et erreurs associées obtenues par le LIEKF (en noir) et le LIEKF dont la matrice de covariance du bruit de mesure est déterminée par un CNN (en bleu).

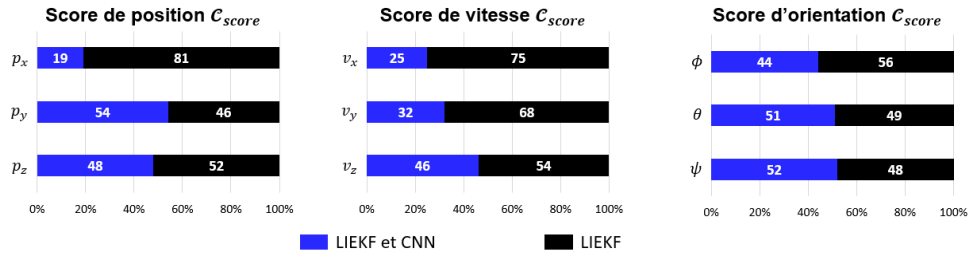

FIGURE 3.27 – Critère $\mathcal{C}_{score}$ (3.46) du LIEKF ajusté par le CNN (en bleu) et du LIEKF (en noir) pour l'estimation de 100 trajectoires de mortiers de 120 mm.

Les figures 3.27 et 3.28 montrent que les performances d'estimation du LIEKF et du LIEKF ajusté par un CNN sont similaires comme les taux d'erreur $\mathcal{C}_e$ sont du même ordre de grandeur pour chacune des grandeurs. Toutefois, il est intéressant de remarquer que le réglage dynamique de la covariance du bruit de mesure du LIEKF dégrade l'estimation des positions p_x, p_z, des vitesses v_x et v_z et des angles de roulis ϕ et de lacet ψ contrairement aux positions, aux vitesses et aux angles suivant l'axe longitudinal. Ainsi, le réglage de la covariance du bruit de mesure d'un LIEKF présente les mêmes constatations que précédemment. Dans le cas où les observations sont fidèles, le CNN ne permet pas d'optimiser

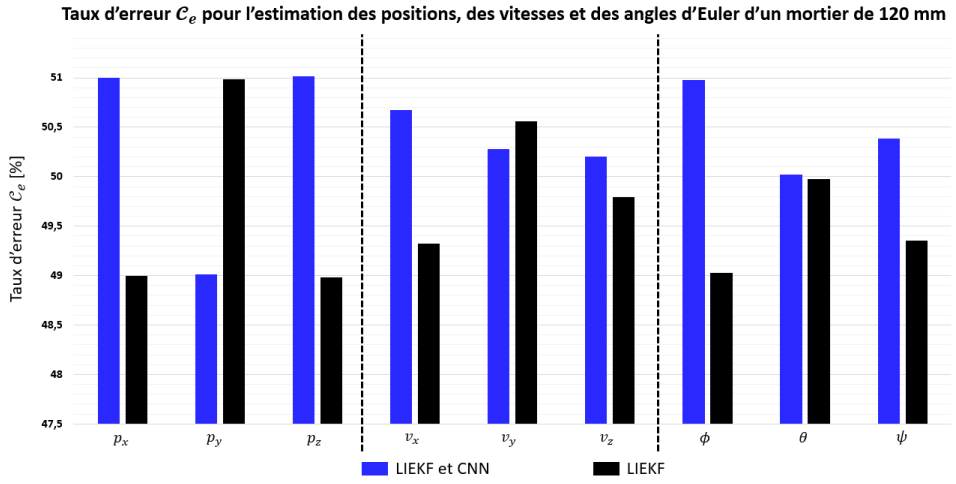

FIGURE 3.28 – Taux d'erreur $\mathcal{C}_e$ (3.47) du LIEKF ajusté par le CNN (en bleu) et du LIEKF (en noir) pour l'estimation de 100 trajectoires de mortiers de 120 mm.

toutes les grandeurs estimées. Toutefois, cette solution permet d'optimiser les estimations suivant l'axe longitudinal dont les dynamiques associées sont très faibles.

3.4.3 Application à un cas simple

Afin de vérifier si la méthode de réglage de la covariance du bruit de mesures par un CNN permet avec certitude de converger vers des valeurs optimales, un filtre de Kalman linéaire est implémenté dans le cas où l'ensemble des paramètres est connu. Pour cela, sur un cas d'application simple, un filtre de Kalman linéaire est développé et l'ensemble des mesures et des paramètres de bruit sont connus avec certitude.

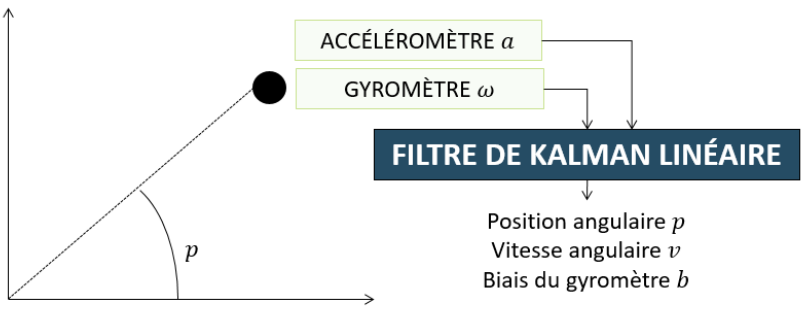

FIGURE 3.29 – Principe de fonctionnement du filtre de Kalman linéaire implémenté pour estimer la position et la vitesse angulaire d'un mobile en mouvement suivant un axe à partir de la mesure de l'accéléromètre et du gyromètre.

Comme illustré sur la figure 3.29, le filtre de Kalman linéaire vise à estimer la position angulaire p et la vitesse angulaire v d'un mobile en déplacement suivant un axe ainsi que le biais du gyromètre b à partir des mesures d'un accéléromètre et d'un gyromètre.

La dynamique d'évolution de la position angulaire p et de la vitesse angulaire v ainsi que du biais du gyromètre b est alors :

$$\frac{d}{dt}v = W_v, \qquad \frac{d}{dt}p = v + W_p, \qquad \frac{d}{dt}b = W_b \tag{3.81}$$

avec W_v, W_p et W_b des bruits blancs supposés gaussiens.

L'étape de prédiction en temps discret avec Δt la période d'échantillonnage est alors :

$$\begin{matrix}\widehat{x}_{k|k-1} = A_k\widehat{x}_{k-1|k-1}, \\ P_{k|k-1} = A_kP_{k-1|k-1}A_k^T + Q\end{matrix}, \quad A_k = \begin{bmatrix}1 & 0 & 0\\ \Delta t & 1 & 0\\ 0 & 0 & 1\end{bmatrix},\ Q = cov\left(\begin{bmatrix}W_v\\ W_p\\ W_b\end{bmatrix}\right) \tag{3.82}$$

Les observations considérées sont les mesures du gyromètre ω et de l'accéléromètre a qui sont modélisées telles que $\widehat{y}_k = H\widehat{x}_{k|k-1} + W$:

$$\begin{bmatrix}\widehat{\omega}_k\\ \widehat{a}_k\end{bmatrix} = \begin{bmatrix}1 & 0 & 1\\ 0 & 1 & 0\end{bmatrix}\begin{bmatrix}\widehat{v}_{k|k-1}\\ \widehat{p}_{k|k-1}\\ \widehat{b}_{k|k-1}\end{bmatrix} + \begin{bmatrix}W_\omega\\ W_a\end{bmatrix} \tag{3.83}$$

avec W_ω et W_a les bruits des mesures du gyromètre et de l'accéléromètre dont la matrice de covariance est notée R.

L'étape de mise à jour du filtre de Kalman linéaire est alors :

$$\begin{matrix}K = P_{k|k-1}H(HP_{k|k-1}H^T + R) \\ \widehat{x}_{k|k} = \widehat{x}_{k|k-1} + K\left(\begin{bmatrix}\omega_k\\ a_k\end{bmatrix} - \begin{bmatrix}\widehat{\omega}_k\\ \widehat{a}_k\end{bmatrix}\right), \qquad P_{k|k} = (I - KH)P_{k|k-1}\end{matrix} \tag{3.84}$$

Un jeu de données est généré avec des paramètres connus de bruit de capteur. Les valeurs de la position et de la vitesse angulaire ainsi que le biais du gyromètre sont identiques pour l'ensemble des trajectoires du jeu de données, de sorte que :

$$\begin{matrix}v_{ref}(t) = \pi(\omega t)sin(\omega t), \qquad p_{ref}(t) = -\pi cos(\omega t) + \pi, \\ b_{ref}(t) = 10t + 20sin(0.1\omega t).\end{matrix} \tag{3.85}$$

Les mesures du gyromètre et de l'accéléromètre sont simulées telles que :

$$\omega(t) = v_{ref} + b_{ref} + W_\omega \times \mathcal{N}(0,1), \qquad a(t) = p_{ref} + W_a \times \mathcal{N}(0,1) \tag{3.86}$$

avec W_ω et W_a les bruits des mesures de chacune des simulations de sorte que :

$$W_\omega = 0.3k, \qquad W_a = 0.1k \tag{3.87}$$

avec k l'index de la simulation considérée variant de 1 à 1000.

Quarte méthodes sont comparées pour l'estimation de la position angulaire, de la vitesse angulaire du mobile ainsi que du biais estimé du gyromètre :

- le filtre de Kalman linéaire dont les valeurs de la matrice de covariance sont constantes pour chacune des trajectoires et représentent les valeurs exactes, c'est-à-dire les écarts types des bruits des capteurs tels que :

$$R_{True} = \begin{bmatrix} W_\omega^2 & 0 \\ 0 & W_a^2 \end{bmatrix} \tag{3.88}$$

- le filtre de Kalman linéaire dont les valeurs de la matrice de covariance Σ_ω et Σ_a sont fixes pour l'ensemble des trajectoires et déterminées classiquement telles que,

$$R_{CST} = \begin{bmatrix} \Sigma_\omega & 0 \\ 0 & \Sigma_a \end{bmatrix} \tag{3.89}$$

- le filtre de Kalman ajusté par un CNN entraîné pour déterminer les paramètres α et β de la matrice de covariance du bruit de mesure du filtre tel que :

$$R_{CNN} = \begin{bmatrix} \Sigma_\omega \times 10^\alpha & 0 \\ 0 & \Sigma_a \times 10^\beta \end{bmatrix} \tag{3.90}$$

- le filtre de Kalman ajusté par un CNN entraîné pour déterminer les paramètres α et β de la matrice de covariance du bruit de mesure du filtre tel que :

$$R_{CNN_{abs}} = \begin{bmatrix} |\alpha| & 0 \\ 0 & |\beta| \end{bmatrix} \tag{3.91}$$

La figure 3.30 présente le taux d'erreur $\mathcal{C}_e$ (3.47) du filtre avec les valeurs exactes (en

vert), du filtre avec des valeurs constantes (en noir), du filtre ajusté par le CNN d'après l'équation (3.90) (en bleu) et du filtre ajusté par le CNN d'après l'équation (3.91) (en jaune).

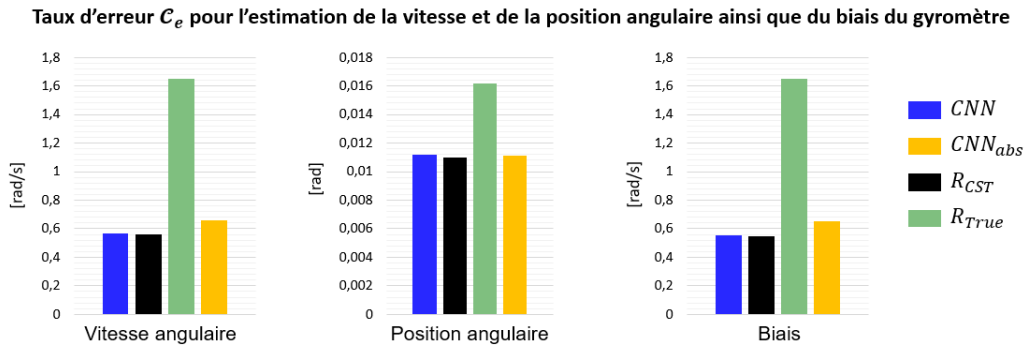

FIGURE 3.30 – Taux d'erreur $\mathcal{C}_e$ (3.47) du filtre avec les valeurs exactes (en vert), du filtre avec des valeurs constantes (en noir), du filtre ajusté par le CNN d'après l'équation (3.90) (en bleu) et du filtre ajusté par le CNN d'après l'équation (3.91) (en jaune).

La figure 3.30 montre tout d'abord que le filtre avec les valeurs exactes de covariance (en vert) présente les moins bonnes performances d'estimation. De plus, les filtres ajustés par les CNN (en bleu et jaune) et le filtre avec des valeurs fixes de covariance (en noir) présentent des performances similaires pour l'estimation de la position et de la vitesse angulaire du mobile ainsi que du biais du gyromètre. Toutefois, la forme de la covariance définie par l'équation (3.90) semble plus appropriée que celle présentée par l'équation (3.91). Ainsi, dans le cas du filtre de Kalman linéaire présenté ci-dessus, un CNN permet de déterminer un ordre de grandeur des valeurs de la covariance du bruit de mesure afin d'obtenir les meilleures estimations. En effet, le CNN détermine les valeurs diagonales de la matrice de covariance du bruit de mesure du filtre sous la forme de deux paramètres strictement positifs comme présenté dans l'équation (3.91). Ainsi, dans ce cas d'application, le CNN permet d'identifier des valeurs optimales de covariance.

3.5 Conclusion

Ce chapitre a présenté une méthode de navigation basée sur un filtre de Kalman où la matrice de covariance du bruit de mesure est ajustée par un réseau de neurones convolutifs. Le CNN permet ainsi de déterminer une matrice de covariance variable dans le temps et adaptée aux phases de vol du projectile.

Cette méthode d'optimisation est implémentée pour un filtre de Kalman Imparfait Invariant Étendu à droite qui estime la position, la vitesse et l'orientation d'un mortier de 120 mm à partir des capteurs inertiels embarqués et de la connaissance du champ magnétique de référence. Les résultats indiquent que cette approche optimise significativement l'estimation des positions et des vitesses de la munition mais n'est pas optimale pour estimer les angles d'Euler. Par conséquent, deux méthodes de prétraitement des données qui visent à redimensionner une grandeur sur des plages de variation similaires sont employées, à savoir, la normalisation des données d'entrée du CNN et la rotation du repère de navigation. Les résultats obtenus indiquent que la rotation du repère de navigation permet d'optimiser l'estimation des trajectoires du projectile contrairement à la normalisation.

Afin de valider cette méthode d'ajustement de la matrice de covariance du bruit de mesure d'un filtre, cette approche est employée avec un filtre de Kalman Étendu qui vise à estimer l'angle et la vitesse de roulis d'un projectile à partir des deux magnétomètres radiaux embarqués. Ce filtre, testé en vol lors de travaux antérieurs, est appliqué à un projectile de 40 mm caractérisé par un tir tendu. Le réglage de la covariance du bruit de mesure de ce filtre par un CNN n'améliore pas l'estimation des angles et des vitesses de roulis. En effet, les performances d'estimation de l'EKF sans réseau de neurones et de l'EKF ajusté par un CNN sont similaires. Afin d'optimiser ces résultats, plusieurs formes de la matrice de covariance sont étudiées, notamment l'estimation directe des paramètres diagonaux. Comme précédemment, cette forme de matrice n'a pas permis d'accroître les résultats d'estimation des angles et des vitesses de roulis. Les mêmes observations sont formulées concernant l'étude de l'influence du choix de la fonction de perte lors de la phase d'entraînement du CNN.

Au vu des résultats obtenus pour le projectile de 40 mm, le réglage de la matrice de covariance du bruit de mesure par un CNN est implémenté sur un LIEKF qui estime l'orientation, la vitesse et la position d'un mortier de 120 mm à partir des mesures des accéléromètres et des gyromètres ainsi que de la vitesse du projectile mesurée par un GPS. De même que pour l'EKF, l'ajustement de la covariance du bruit de mesure par un CNN n'a pas permis d'optimiser l'intégralité des grandeurs estimées. Les propriétés d'observabilité des filtres peuvent également induire ces résultats.

Les résultats obtenus par l'EKF et le LIEKF permettent d'affirmer que le type de projectile étudié n'est pas l'origine des faibles performances d'estimation obtenues par la solution proposée. De plus, l'Imp.RIEKF, l'EKF et le LIEKF sont trois filtres dont au

moins l'un des modèles est non linéaire. Afin de vérifier si la nature des modèles influe sur la précision de la solution proposée, cette approche a été testée sur un filtre de Kalman linéaire. Les résultats obtenus montrent que dans le cas d'application de ce filtre, le CNN converge globalement vers des valeurs optimales de covariance.

La solution proposée dans ce chapitre est basée sur des réseaux de neurones convolutifs. Ce type de réseau exploite les propriétés spatiales des données en entrée. Il s'agit à présent d'exploiter les caractéristiques temporelles des données inertielles afin de déterminer les trajectoires des projectiles.

Chapitre 4

Estimation de la trajectoire d'un projectile par un LSTM

Sommaire

4.1 Introduction

Comme présenté dans le chapitre 1, la navigation des projectiles exploite principalement les mesures de l'IMU *(Inertial Measurement Unit)* et du récepteur GNSS *(Global*

Navigation Satellite System) en raison des diverses contraintes imposées au système. Ces mesures sont ensuite combinées par des filtres de Kalman [32]-[35] pour estimer la trajectoire du projectile. Néanmoins, les signaux GNSS sont de plus en plus exclus en raison de leurs indisponibilités et de leurs vulnérabilités face à des conditions hostiles [13]-[17].

Des réseaux de neurones sont en mesure de pallier aux limitations de ces signaux. Plusieurs approches peuvent être envisagées notamment la navigation de bout-en-bout [87]-[94], qui vise à remplacer tous les modèles mathématiques de navigation par des réseaux de neurones. Par conséquent, ce chapitre étudie une méthode d'estimation de la trajectoire des projectiles basée exclusivement sur l'IA, dans un environnement sans GNSS en utilisant uniquement les mesures de l'IMU et des paramètres connus de la munition.

En considérant une trajectoire comme une série temporelle, l'IA fournit des approches intéressantes pour son estimation. La prédiction de séries temporelles est généralement basée sur les réseaux de neurones récurrents *(Recurrent Neural Networks - RNN)* [122]-[124], qui sont une classe de réseau de neurones capable de mémoriser les données passées pour prédire les données futures. C'est pourquoi, les RNN sont particulièrement bien adaptés à la prédiction de séries temporelles, particulièrement le *Long Short-Term Memory (LSTM)* [122]-[124], pour ses facultés de mémorisation des informations passées.

Ce chapitre se focalise sur une méthode de navigation basée exclusivement sur l'IA. Des LSTM sont entraînés pour estimer les trajectoires des projectiles à partir des mesures IMU et des paramètres connus de la munition. Les objectifs de ce chapitre sont :

- → d'étudier dans quelles mesures des réseaux de neurones peuvent remplacer tous les modèles mathématiques habituellement employés pour la navigation des projectiles et de visualiser comment l'IA modélise les différentes dynamiques et contraintes liées à ces systèmes. Pour cela, les trajectoires des projectiles sont estimées par des LSTM (*Long Short-Term Memory*) à partir des mesures de l'IMU embarquée, de la connaissance du champ magnétique de référence, des paramètres de lancement de la munition et d'un vecteur temporel.
- → d'appliquer cette solution de navigation à plusieurs projectiles afin de déterminer les limites de cette approche. Pour cela, les trajectoires balistiques d'un mortier de 120 mm et d'un obus de 155 mm et les tirs tendus d'un projectile de 40 mm et d'un *Basic Finner* sont estimés par plusieurs LSTMs.
- → d'évaluer l'influence de plusieurs méthodes de prétraitement des données d'entrée et l'impact des différents modèles d'erreurs des capteurs inertiels sur la précision des estimations.

Tout d'abord, la partie (*4.2*) présente le principe de fonctionnement des réseaux de neurones récurrents, particulièrement celui des LSTMs (*Long Short-Term Memory*). Puis, la partie (*4.3*) détaille les caractéristiques des LSTMs entraînés pour estimer les trajectoires des projectiles. Enfin, les parties (*4.4*) et (*4.5*) analysent les résultats d'estimation des trajectoires d'un mortier de 120 mm, d'un obus de 155 mm, d'un projectile de 40 mm et d'un *Basic Finner* par un LSTM.

4.2 Les réseaux de neurones récurrents

Cette partie présente le principe de fonctionnement général des réseaux de neurones récurrents *(Recurrent Neural Network - RNN)*, du Vanilla RNN, la forme la plus simple des RNN, ainsi que celui des LSTM *(Long Short-Term Memory)*.

4.2.1 Les RNN pour la prédiction de séries temporelles

Les réseaux de neurones récurrents (RNN) sont une classe de réseaux de neurones composés de plusieurs cellules RNN dont les connexions sont caractérisées par des boucles de rétroaction. En effet, la sortie d'une cellule RNN est interconnectée à la cellule suivante pour produire une estimation. Ces connexions permettent ainsi aux RNN de mémoriser des informations passées. C'est pourquoi, les RNN sont parfaitement adaptés au traite-

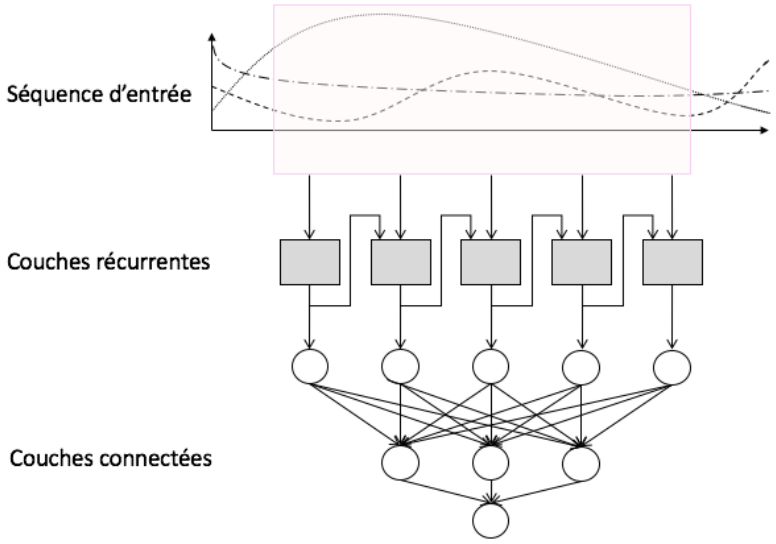

FIGURE 4.1 – Principe de fonctionnement d'un réseau de neurones récurrents.

ment de séquences temporelles et sont principalement employés pour la reconnaissance automatique de la parole, la traduction automatique et pour la prédiction de séries temporelles [122]-[124].

Le principe de fonctionnement d'un réseau de neurones récurrents est schématiquement illustré dans la figure 4.1. Ainsi, une séquence d'une série temporelle est transmise en entrée d'une couche RNN, composée de plusieurs cellules interconnectées entre elles, qui vont prédire un vecteur. Ces données sont ensuite transmises à des couches entièrement connectées, comme dans tous réseaux de neurones, afin de réaliser la fonction souhaitée.

Données d'entrée d'un RNN

Les données d'entrée d'un réseau de neurones récurrents sont une série temporelle. Pour cela, une fenêtre glissante se déplace sur les données dans le sens de parcours des signaux. Cette fenêtre est ensuite transmise en entrée du RNN. Ainsi, à chaque pas de temps, un réseau de neurones récurrents considère comme donnée d'entrée une séquence $x = [x_0, x_1, ..., x_\tau] \in \mathbb{R}^{\tau \times F}$, de longueur τ, et où chaque donnée d'entrée x_i comprend F caractéristiques nécessaires à décrire ces données.

Fonctionnement d'un RNN

La figure 4.2 représente un RNN afin de mettre en évidence les connexions récurrentes.

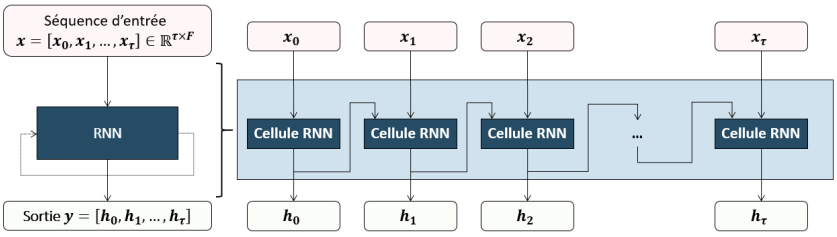

FIGURE 4.2 – Réseau de neurones récurrents : représentation *many-to-many*.

Un RNN est composé de plusieurs cellules interconnectées entre elles. Au pas de temps $i \in [0, \tau]$, la i-ème cellule estime une sortie y à partir de l'entrée courante x_i et de la sortie de la cellule RNN précédente h_{i-1}, appelée l'état caché. Ainsi, à travers les informations contenues dans l'état caché h_{i-1}, le RNN mémorise les informations pertinentes passées à court terme.

Suivant le type d'application souhaitée, la taille de la séquence des prédictions y_i diffère. Les deux architectures les plus utilisées sont l'architecture *many-to-many* et l'architecture *many-to-one*.
Dans le cas d'un RNN *many-to-many*, illustré dans la figure 4.2, le réseau prédit une séquence de $y = [h_0, h_1, ..., h_\tau]$ à partir des données d'entrée $x = [x_0, x_1, ..., x_\tau]$. Cette représentation est généralement utilisée pour la traduction.
Dans le cas d'un RNN *many-to-one*, illustré dans la figure 4.3, le réseau prédit y à partir d'une séquence d'entrée $x = [x_0, x_1, ..., x_\tau]$. Cette représentation est fréquemment utilisée pour la classification et les problèmes de prédiction.

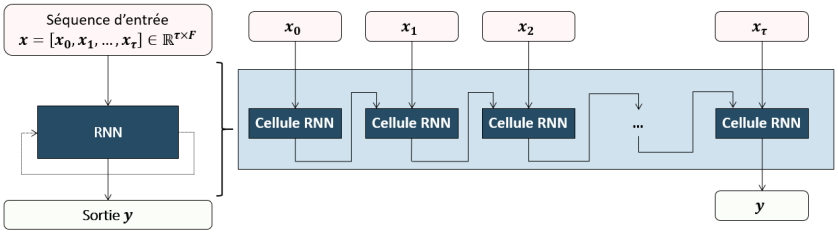

FIGURE 4.3 – Réseau de neurones récurrents : représentation *many-to-one*.

Pour la prédiction de séries temporelles, il existe essentiellement trois types d'architectures : le *Vanilla RNN*, la forme la plus simple des réseaux récurrents mais qui présente plusieurs limitations, le *LSTM (Long Short-Term Memory)* et le *GRU (Gated Recurrent Unit)*, extensions du *Vanilla RNN* permettant à la fois de gérer la mémoire à court et à long terme [122]-[124].

Le Vanilla RNN

Le *Vanilla RNN*, illustré dans la figure 4.2, est la structure la plus simple des réseaux récurrents. Pour prédire y_i au pas de temps i, le *Vanilla RNN* utilise l'entrée courante x_i et l'état caché h_{i-1} à l'instant précédent, contenant les caractéristiques passées au pas de temps précédemment. De ce fait, le *Vanilla RNN* mémorise uniquement les informations passées à court terme. De plus, lors de la phase d'apprentissage, le *Vanilla RNN* souffre de problèmes de disparition et d'explosion de gradient [122], [125], [126] :

- dans le cas de la disparition du gradient, la rétropropagation de la dernière couche à la première couche conduit à une réduction du gradient. Ainsi, les poids de la première couche ne sont plus mis à jour pendant l'apprentissage et le *Vanilla RNN* n'apprend aucune fonctionnalité.

- dans le cas de l'explosion du gradient, ces derniers deviennent de plus en plus importants, entraînant ainsi des mises à jour des poids importantes et causant potentiellement la divergence du réseau.

Afin de surmonter le problème de mémorisation à long terme et le problème de rétropropagation du gradient lors de la phase d'apprentissage, des cellules mémoire gérées par des portes sont ajoutées au *Vanilla RNN*, formant ainsi les LSTM et les GRU [122].

4.2.2 Extension du RNN simple : le *Long Short-Term Memory*

Un *Long Short-Term Memory (LSTM)* est une extension du *Vanilla RNN* qui permet de résoudre les problèmes de disparition et d'explosion du gradient lors de la phase d'apprentissage, ainsi que de mémoriser des informations passées à court terme et à long terme. Pour cela, comme présenté dans la figure 4.4, une cellule mémoire est ajoutée en plus de l'état caché et les informations sont gérées par des portes afin de mémoriser les informations passées à long terme [122]-[124].

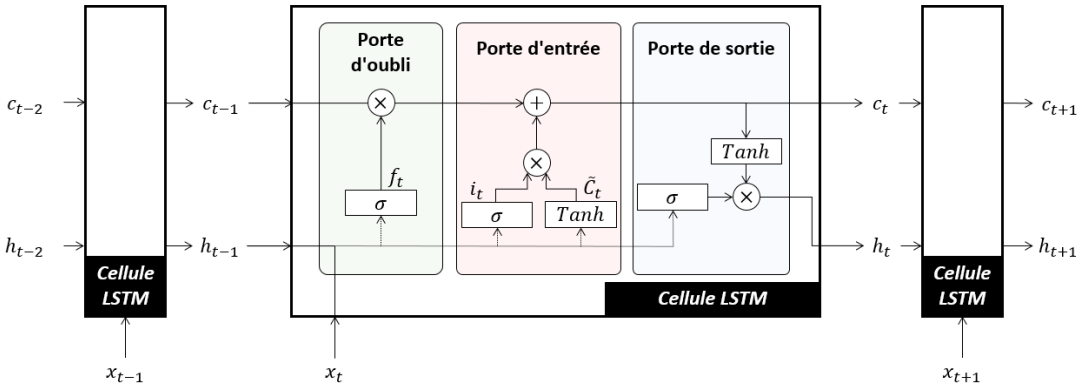

FIGURE 4.4 – Principe de fonctionnement de la cellule LSTM composée de trois portes avec x_t l'entrée à l'instant courant, h_{t-1} l'état caché à l'instant précédent, et c_{t-1} l'état de la cellule mémoire à l'instant précédent.

Cellule mémoire et état caché du LSTM

Comme pour le *Vanilla RNN*, une séquence $x = [x_0, x_1, ..., x_\tau] \in \mathbb{R}^{\tau \times F}$ est transmise en entrée du LSTM composé d'une succession de cellules LSTM interconnectées, pour prédire une séquence de sortie $y = [h_0, h_1, ..., h_\tau]$.

Ainsi, une cellule LSTM utilise trois données d'entrée pour prédire une sortie h_t : la

donnée d'entrée x_t à l'instant courant, l'état caché h_{t-1} à l'instant précédent et la cellule mémoire c_{t-1} à l'instant précédent. L'état caché h_{t-1} mémorise les informations passées à court terme et la cellule mémoire c_{t-1} mémorise les informations passées à long terme.

Principe de fonctionnement d'une cellule LSTM

Comme le montre la figure 4.4, une cellule LSTM est composée de trois portes afin de gérer efficacement les informations à mémoriser dans l'état caché et la cellule mémoire :

- la porte d'oubli *(forget gate)* filtre, à travers une fonction sigmoïde $\sigma(.)$, les données contenues dans la concaténation de x_t et h_{t-1}. Les données sont oubliées pour les valeurs proches de 0 et sont mémorisées pour les valeurs proches de 1. Le modèle de la porte d'oubli est :

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \tag{4.1}$$

- la porte d'entrée *(input gate)* extrait les informations pertinentes de $[h_{t-1}, x_t]$ en appliquant une fonction sigmoïde $\sigma(.)$ et une fonction $Tanh(.)$. La porte d'entrée est représentée par :

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_f), \qquad \tilde{C}_t = tanh(W_c.[h_{t-1}, x_t] + b_c) \tag{4.2}$$

  La cellule mémoire c_t est mise à jour avec la porte d'oubli et la porte d'entrée pour mémoriser les données pertinentes :

$$c_t = f_t \times c_{t-1} + i_t \times \tilde{C}_t \tag{4.3}$$

- la porte de sortie *(output gate)* définit le prochain état caché h_t contenant des informations sur les entrées précédentes. L'état caché h_t est mis à jour avec la cellule mémoire c_t, l'état caché h_{t-1} et la donnée d'entrée x_t. Le modèle de la porte de sortie est :

$$h_t = \sigma(W_h.[h_{t-1}, x_t] + b_h) \times tanh(c_t) \tag{4.4}$$

  avec $W_{(.)}$ et $b_{(.)}$, les différentes matrices de poids de porte et biais.

Il existe plusieurs variantes au principe de fonctionnement du LSTM présenté précédemment, tel que [92], [127] :

- le *LSTM bidirectionnel*, formé par un empilement de deux couches LSTM classiques

et présenté dans la figure 4.5. La première couche LSTM traite la séquence d'entrée dans le sens de propagation du signal, alors que la seconde couche LSTM traite la séquence d'entrée dans le sens inverse du sens de propagation du signal. Les sorties des deux couches LSTM sont ensuite combinées pour produire la sortie désirée. Cette architecture présente l'avantage de traiter simultanément des informations passées et futures et est principalement utilisée pour l'analyse et le traitement du langage. Toutefois, cette architecture, plus complexe qu'un LSTM, est difficilement applicable en temps réel.

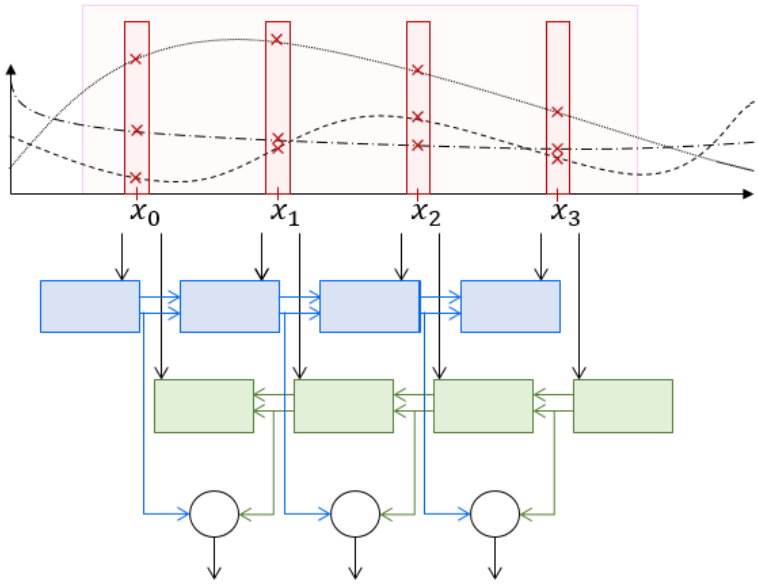

FIGURE 4.5 – Principe de fonctionnement d'un LSTM bidirectionnel.

- le *LSTM encodeur-décodeur*, formé de deux LSTM, et présenté dans la figure 4.6. Le premier LSTM, l'encodeur, parcourt la séquence d'entrée et prédit un vecteur de sortie. Le second LSTM, le décodeur, est alimenté par le vecteur de sortie de l'encodeur pour prédire une sortie. Ce type d'architecture est principalement utilisé pour la traduction automatique.

Applications des LSTM pour la prédiction de séries temporelles

Les LSTM sont couramment utilisés pour la navigation, comme la prédiction de la trajectoire d'avions [87], [93], de véhicules terrestres [92], de routes maritimes [89] ou de mouvements humains [90], [91]. Actuellement, peu de travaux sont réalisés pour l'estima-

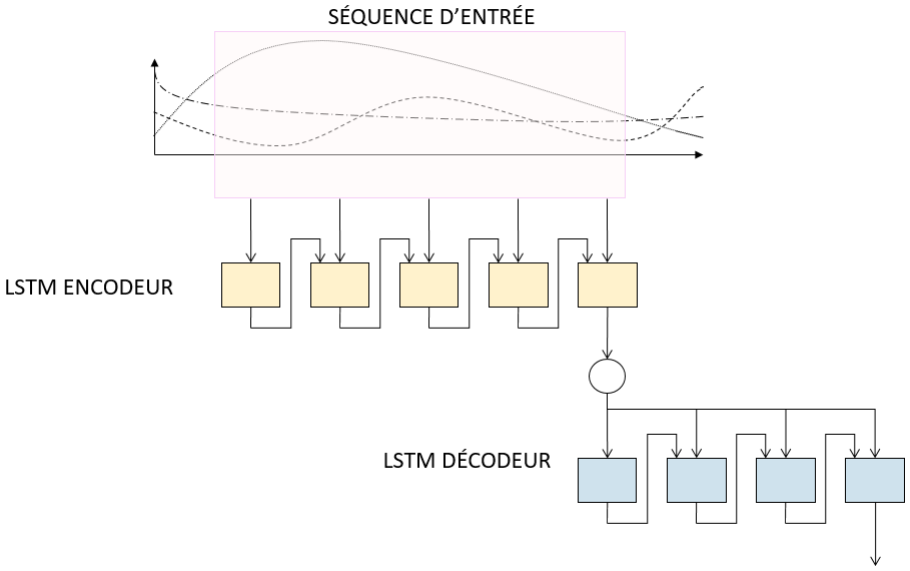

FIGURE 4.6 – Principe de fonctionnement d'un LSTM encodeur/décodeur.

tion de la trajectoire d'un projectile. Il est cependant intéressant de mentionner [94] qui estime la trajectoire d'un projectile par un LSTM entraîné à partir de mesures radars incomplètes et bruitées.

4.3 Caractéristiques d'estimation de la trajectoire d'un projectile par un LSTM

Cette partie présente les LSTMs entraînés pour estimer les trajectoires des différents projectiles étudiés dans ce chapitre, les données d'entrée et de sortie considérées, ainsi que les méthodes de prétraitement des données d'entrée.

4.3.1 Formulation du problème

Le LSTM estime la trajectoire d'un projectile à partir des capteurs inertiels embarqués et des paramètres connus de la munition. La finalité est que le LSTM fournisse, tout comme un récepteur GNSS, des informations de positionnement absolu, mais sans partager les limitations de ces mesures, et tout en produisant une solution de navigation à la même fréquence que les capteurs inertiels.

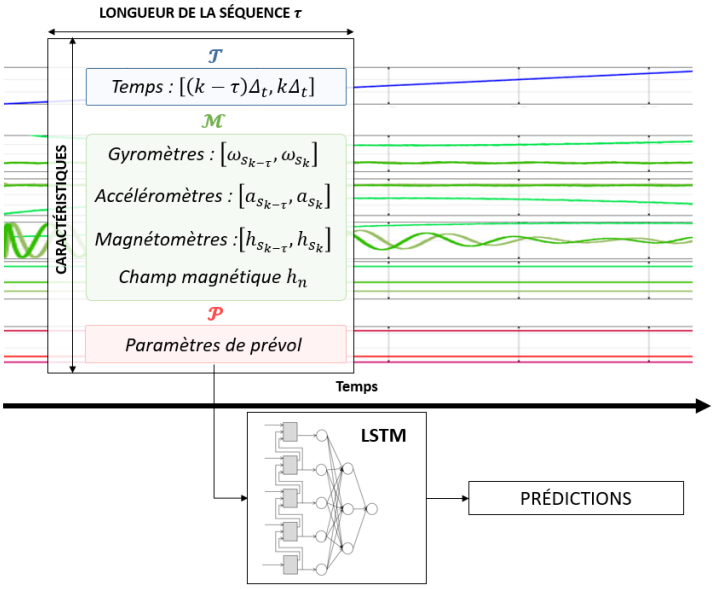

FIGURE 4.7 – LSTM pour l'estimation de la trajectoire d'un projectile.

Comme le montre la figure 4.7, les prédictions du LSTM au temps k sont obtenues à partir de données d'entrée tridimensionnelles de taille $(B_{atch\ size}, S_{eq\ len}, I_{n\ Features})$, avec $B_{atch\ size}$ le nombre de séquences considérées, $S_{eq\ len}$ le nombre de pas de temps dans la séquence et $I_{n\ Features}$ le nombre de caractéristiques décrivant chaque pas de temps.

Données d'entrée

En se basant sur le jeu de données de trajectoires de projectiles BALCO présenté dans la partie (*1.5*), le nombre de caractéristiques d'entrée décrivant chaque pas de temps est de 16, de sorte que $I_{n\ Features} = (\mathcal{M}, \mathcal{P}, \mathcal{T}) \in \mathbb{R}^{16}$ avec :

- $\mathcal{M} \in \mathbb{R}^{12}$ **les données inertielles** comprenant :
  - les mesures *IMU* ou *IMU DYN* dans le repère capteur $\boldsymbol{s}$, c'est-à-dire les mesures des accéléromètres $a_s \in \mathbb{R}^{3\times1}$, des gyromètres $\omega_b \in \mathbb{R}^{3\times1}$ et des magnétomètres $h_b \in \mathbb{R}^{3\times1}$. Suivant l'application souhaitée, les deux modèles d'erreur des capteurs inertiels produisant les mesures *IMU* ou *IMU DYN* sont considérés.
  - le champ magnétique de référence $h_n \in \mathbb{R}^{3\times1}$ dans le repère local de navigation $\boldsymbol{n}$, constant pendant la durée de vol du projectile.
- $\mathcal{P} \in \mathbb{R}^3$ **les paramètres de vol** spécifiques à la munition considérée.
  Dans le cas d'un mortier de 120 mm ou d'un *Basic Finner*, les trois paramètres de vol sont l'angle de braquage des ailettes, la vitesse initiale en sortie de canon et

l'angle d'élévation du canon. Dans le cas de l'obus de 155 mm ou du projectile de 40 mm, les trois paramètres de vol sont le roulis initial, la vitesse initiale en sortie de canon et l'angle d'élévation du canon.

- $\mathcal{T} \in \mathbb{R}^1$ **le vecteur temps** évalué tel que $\mathcal{T} = k\Delta_t$ avec k le pas de temps considéré, et Δ_t la période d'échantillonnage des capteurs inertiels.

Données de sortie

Plusieurs LSTM sont entraînés et diffèrent en fonction des caractéristiques de sortie apprises. En effet, à partir des données d'entrée de taille $(B_{atch\ size}, S_{eq\ len}, I_{n\ Features})$, les LSTM estiment un vecteur de sortie de taille $(B_{atch\ size}, O_{ut\ Features})$ où $O_{ut\ Features}$ représente le nombre de caractéristiques de sortie. Selon le LSTM considéré, les caractéristiques de sortie $O_{ut\ Features}$ sont 9 ou 3 de sorte que :

- $LSTM_{ALL}$ estime 9 caractéristiques de sortie : la position $p \in \mathbb{R}^3$, la vitesse $v \in \mathbb{R}^3$ et les angles d'Euler $\Psi \in \mathbb{R}^3$ dans le repère de navigation $\boldsymbol{n}$.
- $LSTM_{POS}$ estime 3 caractéristiques de sortie qui sont les positions du projectile $p \in \mathbb{R}^3$ exprimées dans le repère de navigation $\boldsymbol{n}$.
- $LSTM_{VEL}$ estime 3 caractéristiques de sortie qui sont les vitesses du projectile $v \in \mathbb{R}^3$ exprimées dans le repère de navigation $\boldsymbol{n}$.
- $LSTM_{ANG}$ estime 3 caractéristiques de sortie qui sont les angles d'Euler du projectile $\Psi \in \mathbb{R}^3$ dans le repère de navigation $\boldsymbol{n}$.

Ces quatre types de LSTM permettent d'identifier si une grandeur telle que l'orientation peut être apprise indépendamment des autres ou si les positions, les vitesses et les orientations sont nécessairement corrélées pour le LSTM.

Détails des entraînements

La perte entre la prédiction du LSTM et les données de référence est évaluée avec l'erreur quadratique moyenne *(Mean Squared Srror - MSE)* définie telle que :

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(X_{ref} - X_{LSTM})^2 \tag{4.5}$$

avec X_{ref} les positions, les vitesses et/ou les angles d'Euler de référence fournis par BALCO, et X_{LSTM} les grandeurs estimées par le LSTM. L'algorithme d'optimisation employé est l'algorithme d'optimisation d'Adam [51].

4.3.2 Prétraitement des données d'entrée

Deux méthodes de prétraitement des données d'entrée du LSTM ont été mises en place ; la normalisation des données d'entrée et la rotation du repère de navigation. Ces deux méthodes, présentées au chapitre 3, redimensionnent chaque composante d'un vecteur sur des plages de variation similaires.

Normalisation des données d'entrée

La normalisation des données d'entrée d'un réseau de neurones redimensionne les données sur des plages de variation similaires en préservant la même distribution et les mêmes ratios que les données d'origine mais conduit à une perte d'information.

La normalisation semble nécessaire comme les plages de variation des données d'entrée $I_{n\ Features} = (\mathcal{M}, \mathcal{P}, \mathcal{T})$ sont disparates. Par exemple, la plage de variation du vecteur temps d'une trajectoire de mortier de 120 mm est d'environ $[0; 27]$ s contrairement aux données inertielles qui sont bornées par les limites des capteurs tels que ± 5 m/s^2 pour l'accéléromètre triaxial. Ainsi, la normalisation redimensionne ces valeurs sur des plages de variation semblables.

Deux types de normalisation sont étudiés :

- *La normalisation Min/Max*, notée $MM(.)$, définie telle que :

$$x_{MM} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{4.6}$$

  avec x_{max} et x_{min} le maximum et le minimum de la grandeur x à normaliser.
- *La normalisation STD*, notée $STD(.)$, définie telle que :

$$x_{STD} = \frac{x - \mu}{\sigma} \tag{4.7}$$

  avec x la quantité à normaliser, μ sa moyenne et σ son écart type. Ainsi x_{STD} est une quantité avec une moyenne nulle et un écart type de un.

L'impact de la normalisation des données d'entrée du LSTM sur la précision des estimations est étudié en normalisant l'ensemble des caractéristiques d'entrée ou en normalisant chacune des caractéristiques d'entrée. Ainsi, les facteurs de normalisation x_{max}, x_{min}, μ_x et σ_x sont calculés sur le jeu de données d'entraînement et sont fixes pour l'en-

traînement et le test du réseau tel que :

$$x_{max} = \frac{1}{N_{sim}} \sum_{i=1}^{N_{sim}} max\left\{\chi_i\right\}, \qquad x_{min} = \frac{1}{N_{sim}} \sum_{i=1}^{N_{sim}} min\left\{\chi_i\right\} \tag{4.8}$$

$$\mu_x = \frac{1}{N_{sim}} \sum_{i=1}^{N_{sim}} \mu\left(\chi_i\right), \qquad \sigma_x = \frac{1}{N_{sim}} \sum_{i=1}^{N_{sim}} \sigma\left(\chi_i\right) \tag{4.9}$$

avec N_{sim} le nombre de simulations dans le jeu de données d'apprentissage et avec χ_i les quantités considérées de la simulation n° i, qui sont $\chi_i = \begin{bmatrix}\mathcal{M} & \mathcal{P} & \mathcal{T}\end{bmatrix}$ dans le cas de la normalisation pour l'ensemble des caractéristiques d'entrée, et $\chi_i = \mathcal{M}$ ou $\chi_i = \mathcal{P}$ ou $\chi_i = \mathcal{T}$ dans le cas de la normalisation pour chacune des caractéristiques d'entrée.

Rotation du repère de navigation

La seconde méthode de prétraitement est la rotation du repère local de navigation $\boldsymbol{n}$, présentée au chapitre précédent. Pour cela, le repère local de navigation $\boldsymbol{n}$ est pivoté d'un angle fixe $\boldsymbol{\gamma}$, afin que les trois composantes d'un vecteur $x_\gamma \in \mathbb{R}^{3\times1}$ exprimé dans le repère local de navigation tourné $\boldsymbol{n_\gamma}$ aient des plages de variation similaires. La rotation du repère de navigation permet au LSTM d'estimer de façon équivalente les trois composantes d'un vecteur dont au moins une des composantes est de très faible amplitude par rapport aux deux autres. Cette méthode est mise en place notamment pour améliorer l'estimation de la position p_y d'un projectile qui est en moyenne 1000 fois moins importante que les positions suivant les deux autres axes.

Comme présenté dans la figure 4.8, toutes les grandeurs définies initialement dans le repère local de navigation $\boldsymbol{n}$ sont exprimées dans le repère local de navigation tourné $\boldsymbol{n_\gamma}$. Lors de la phase d'apprentissage, le LSTM prédit les trajectoires dans le repère local de navigation tourné $\boldsymbol{n_\gamma}$. Ces estimations sont ensuite comparées aux données de référence également exprimées dans le repère local de navigation tourné $\boldsymbol{n_\gamma}$ afin de rétropropager la perte et mettre à jour les paramètres du réseau. Lors de la phase de test, les LSTM estiment les trajectoires dans le repère local de navigation tourné $\boldsymbol{n_\gamma}$, puis, par la rotation inverse, ces estimations exprimées dans le repère local de navigation initial $\boldsymbol{n}$.

L'angle $\boldsymbol{\gamma}$ est fixe pour toutes les trajectoires du jeu de données et est le même pour exprimer les positions, les vitesses et les angles. Il est déterminé afin d'obtenir des plages de variation similaires pour les trois positions et par conséquent, pour les trois vitesses.

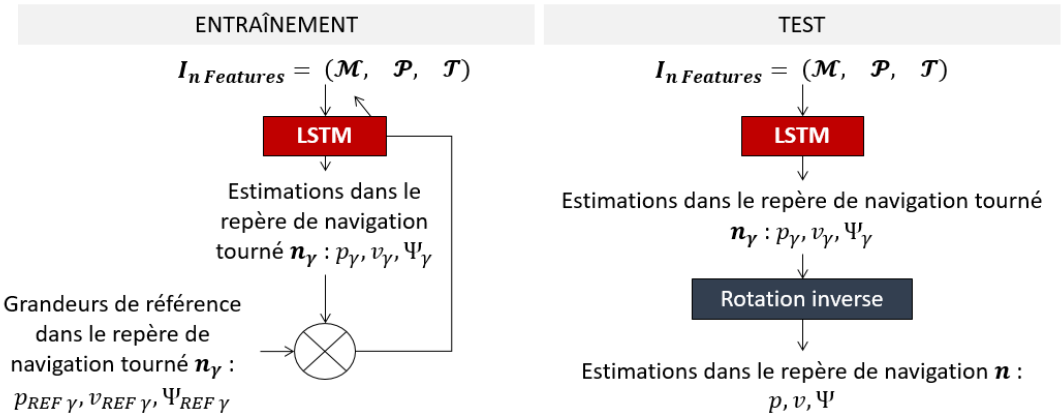

FIGURE 4.8 – Entraînement du LSTM avec la rotation du repère de navigation.

4.4 Estimation de la trajectoire d'un mortier de 120 mm par un LSTM : résultats et analyse

Cette partie présente les résultats d'estimation de la trajectoire d'un mortier de 120 mm par un LSTM. À cette fin, une première section est consacrée à l'étude de l'influence de la normalisation et de la rotation du repère de navigation sur la précision des estimations pour un petit entraînement. La deuxième section se concentre sur la validation des performances d'estimation du LSTM sur un grand ensemble de données, en analysant particulièrement l'impact du modèle d'erreur des capteurs inertiels sur les estimations.

Les performances des différents LSTM sont évaluées par rapport à un algorithme de navigation classique : un *Dead Reckoning* [4]. Un algorithme de *Dead Reckoning* vise à intégrer les mesures du gyromètre ω_s et de l'accéléromètre a_s effectuées dans le repère capteur $\boldsymbol{s}$ pour estimer à chaque instant discret k :

$$R_k = R_{k-1}[\omega_{b_k}\Delta_t]_\times \tag{4.10}$$

$$v_k = v_{k-1} + (R_{k-1}a_{b_k} + g)\,\Delta_t \tag{4.11}$$

$$p_k = p_{k-1} + v_{k-1}\Delta_t + \tfrac{1}{2}(R_{k-1}a_{b_k} + g)\,\Delta_t^2 \tag{4.12}$$

avec $R_k \in SO(3)$ la matrice de rotation du capteur $\boldsymbol{s}$ au repère local de navigation $\boldsymbol{n}$ déterminée par les angles d'Euler du projectile, $g \in \mathbb{R}^{3\times1}$ le vecteur de gravité constant pour la durée de vol du projectile, $p_k \in \mathbb{R}^{3\times1}$ et $v_k \in \mathbb{R}^{3\times1}$ respectivement la position et la vitesse du projectile, et $[.]_\times$ un opérateur $SO(3)$ défini dans la partie (*2.2.1*).

4.4.1 Impact de la normalisation et de la rotation du repère de navigation sur la précision des estimations

Pour étudier l'influence de la normalisation Min/Max $MM(.)$ et $STD(.)$ ainsi que l'influence de la rotation du repère local de navigation sur la précision des estimations, les réseaux $LSTM_{ALL}$, $LSTM_{POS}$, $LSTM_{VEL}$ et $LSTM_{ANG}$ présentés dans la partie (*4.3*) sont déclinés en 8 versions détaillées dans le tableau 4.1.

TABLE 4.1 – Spécifications des versions : Influence de la normalisation des données d'entrée du LSTM et de la rotation du repère de navigation.

NOM	**NORMALISATION**	**ROTATION**
V_1	Non	Non
V_2	$MM(\mathcal{T}), MM(\mathcal{M}), MM(\mathcal{P})$	Non
V_3	$MM(\mathcal{T}, \mathcal{M}, \mathcal{P})$	Non
V_4	$STD(\mathcal{T}), STD(\mathcal{M}), STD(\mathcal{P})$	Non
V_5	$STD(\mathcal{T}, \mathcal{M}, \mathcal{P})$	Non
V_6	Non	Oui
V_7	$MM(\mathcal{T}, \mathcal{M}, \mathcal{P})$	Oui
V_8	$STD(\mathcal{T}, \mathcal{M}, \mathcal{P})$	Oui

Les réseaux $LSTM_{ALL,\ POS,\ VEL,\ ANG,\ V_{1-8}}$ sont entraînés sur un jeu de données d'entraînement composé de 100 simulations de trajectoires de mortier de 120 mm, un jeu de données de validation composé de 10 simulations et un jeu de données de test composé de 20 simulations, générées par BALCO présenté dans la partie (*1.5*). Les réseaux sont entraînés d'après les données d'entrée $(\mathcal{M}, \mathcal{P}, \mathcal{T})$ décrites dans la partie (*4.3*) et avec le modèle de capteurs inertiels produisant les données *IMU*. La taille du *batch* est de 64 et la taille de la séquence d'entrée est de 20 pas de temps pour capturer suffisamment de dépendances à long terme sans dépendre du bruit des mesures. Les réseaux sont mis à jour par l'algorithme d'optimisation d'Adam [51] et la perte est évaluée avec l'erreur quadratique moyenne définie par l'équation (4.5).
Les hyperparamètres d'entraînement des réseaux $LSTM_{ALL,\ POS,\ VEL,\ ANG,\ V_{1-8}}$ sont résumés dans le tableau 4.2.

TABLE 4.2 – Caractéristiques d'entraînement de $LSTM_{ALL,\ POS,\ VEL,\ ANG,\ V_{1-8}}$ sur des trajectoires de mortier de 120 mm.

Jeu de données	Jeu de données d'entraînement : Jeu de données de validation : Jeu de données de test :	100 simulations 10 simulations 20 simulations
Caractéristiques des données d'entrée	$B_{atch\ size}$: $S_{eq\ len}$: Données d'entrée :	64 20 pas de temps $I_{n\ Features} = (\mathcal{M}, \mathcal{P}, \mathcal{T}) \in \mathbb{R}^{16}$ avec les données IMU.
Caractéristiques d'entraînement	Fonction de coût : Algorithme d'optimisation : Taux d'apprentissage :	*Mean Squared Error (MSE)* Adam $1e^{-4}$

Résultats préliminaires d'estimation d'une trajectoire de mortier de 120 mm

Les figures 4.9 - 4.11 présentent la position, la vitesse et les angles d'Euler estimés ainsi que les erreurs associées pour une trajectoire de mortier de 120 mm de l'ensemble de test. Pour des raisons de lisibilité, trois méthodes d'estimation sont d'abord comparées :

- l'algorithme de *Dead Reckoning* (4.10) - (4.12),
- $LSTM_{ALL,V_1}$ (sans normalisation et sans rotation du repère local de navigation),
- $LSTM_{ALL,V_6}$ (sans normalisation et avec la rotation du repère local de navigation).

Comme le montrent les figures 4.9 et 4.10, les positions et les vitesses estimées par les LSTM sont nettement plus précises que celles du *Dead Reckoning*. Concernant l'estimation de l'orientation du projectile (figure 4.11), les LSTM ne sont précis que pour estimer l'angle de tangage θ et l'angle de lacet ψ. Les erreurs dans l'estimation de l'angle de roulis ϕ par les LSTM sont dues à la vitesse de rotation du mortier. Les LSTM ne parviennent pas à capturer pleinement toutes les variations d'angle de roulis. De plus, selon les figures 4.9 et 4.11, la rotation du repère local de navigation améliore l'estimation de la position et de la vitesse du projectile mais dégrade légèrement l'estimation de l'angle de tangage θ (figure 4.11).

Analyse sur l'ensemble des données de test

Afin de valider les observations précédentes, $LSTM_{ALL,\ POS,\ VEL,\ ANG,\ V_{1-8}}$ sont évalués sur l'ensemble du jeu de données de test selon deux critères basés sur l'erreur qua-

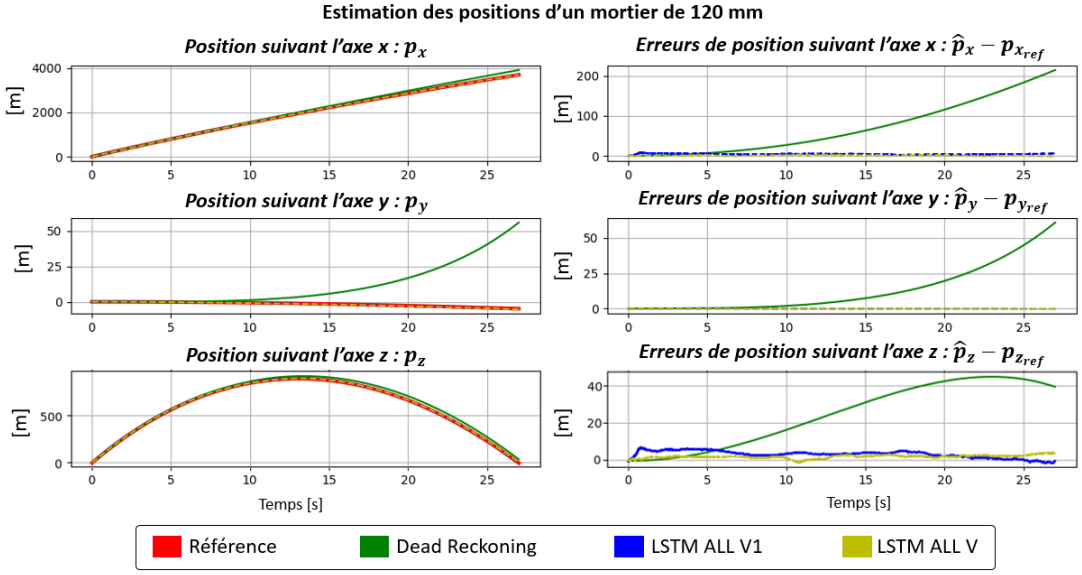

FIGURE 4.9 – Position estimée et erreurs associées [m] obtenues par le *Dead Reckoning* (vert), $LSTM_{ALL,\ V_1}$ (bleu), $LSTM_{ALL,\ V_6}$ (jaune) et la position de référence (rouge).

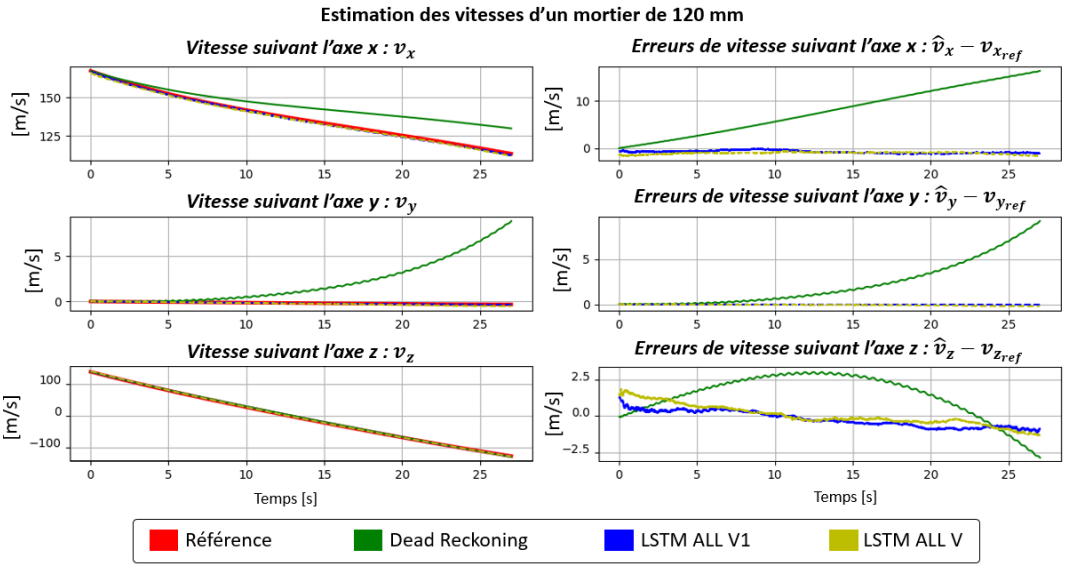

FIGURE 4.10 – Vitesse estimée et erreurs associées [m/s] obtenues par le *Dead Reckoning* (vert), $LSTM_{ALL,\ V_1}$ (bleu), $LSTM_{ALL,\ V_6}$ (jaune) et la vitesse de référence (rouge).

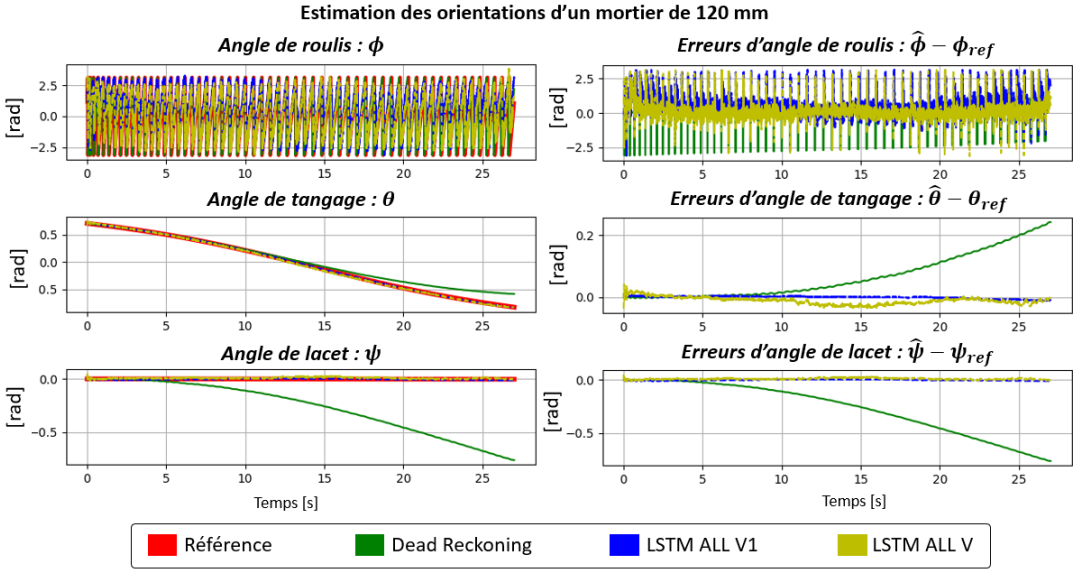

FIGURE 4.11 – Orientation estimée et erreurs associées [rad] obtenues par le *Dead Reckoning* (vert), $LSTM_{ALL,\ V_1}$ (bleu), $LSTM_{ALL,\ V_6}$ (jaune) et l'orientation de référence (rouge).

dratique moyenne *(Root Mean Square Error - RMSE)* définie comme suit :

$$RMSE_x = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\left(x_{k,ref} - \widehat{x}_k\right)^2} \tag{4.13}$$

avec $\widehat{x}$ la quantité estimée, x_{ref} la référence et N le nombre d'échantillons de la simulation.

Les deux critères d'évaluation sont :

- le *taux de réussite* $\mathcal{C}_1$ qui représente le nombre de trajectoires où la RMSE du LSTM considéré est strictement inférieure à celle du *Dead Reckoning* :

$$\mathcal{C}_1 = \sum_{k=1}^{N_{sim}} RMSE_{LSTM} < RMSE_{DR} \tag{4.14}$$

- le *taux d'erreurs* $\mathcal{C}_2$ qui représente le pourcentage d'erreur d'estimation du LSTM considéré par rapport aux erreurs du *Dead Reckoning*, évalué tel que :

$$\mathcal{C}_2 = \frac{100}{N_{sim}} \sum_{k=1}^{N_{sim}} \frac{RMSE_{LSTM}}{RMSE_{LSTM} + RMSE_{DR}} \tag{4.15}$$

avec N_{sim} le nombre de simulations dans le jeu de données de test.

Les figures 4.12 - 4.14 présentent le taux de réussite $\mathcal{C}_1$ (4.14) et le taux d'erreurs $\mathcal{C}_2$ (4.15) pour l'estimation des positions, des vitesses et des angles d'Euler avec les réseaux $LSTM_{ALL,\ POS,\ VEL,\ ANG,\ V_{1-8}}$ (voir tableau 4.2).

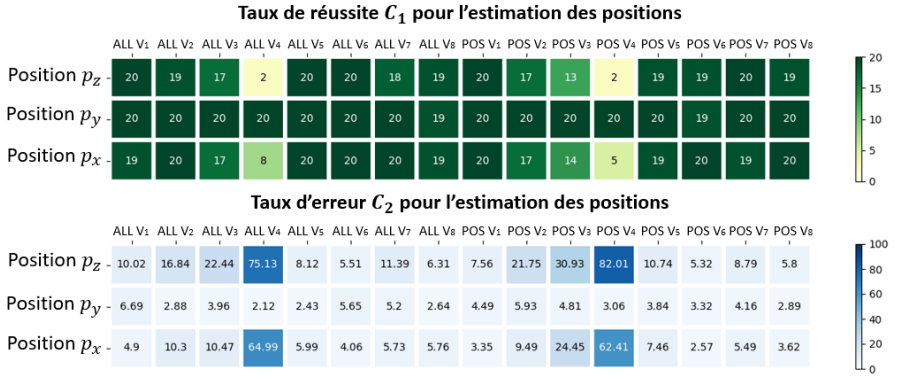

FIGURE 4.12 – Analyse de l'estimation des positions : taux de réussite $\mathcal{C}_1$ (4.14) et taux d'erreurs $\mathcal{C}_2$ (4.15) de $LSTM_{ALL,\ V_1-V_8}$ et $LSTM_{POS,\ V_1-V_8}$.

Résultats d'estimation des positions : D'après la figure 4.12, les LSTM surpassent largement l'algorithme de *Dead Reckoning* pour l'estimation de la position, en particulier le long de l'axe y.

De plus, $LSTM_{POS,\ V_1,\ V_{6-8}}$ spécialisé dans l'estimation des positions, surpasse légèrement $LSTM_{ALL,\ V_1,\ V_{6-8}}$ d'après $\mathcal{C}_1$ et $\mathcal{C}_2$, pour l'estimation des positions suivant les trois axes lorsqu'il n'y a aucune normalisation des données d'entrée.

Les normalisations affectent différemment les estimations de position. Premièrement, d'après les taux de réussite et d'erreur, les versions V_3 et V_4 correspondant aux normalisations $MM(\mathcal{T}, \mathcal{M}, \mathcal{P})$ et $STD(\mathcal{T}), STD(\mathcal{M}), STD(\mathcal{P})$ ne conviennent pas à cette application. Deuxièmement, la précision des réseaux avec normalisation (Min/Max $V_{2,3,7}$ et STD $V_{4,5,8}$) est moins bonne que celle des réseaux sans normalisation ($V_{1,6}$) comme la normalisation implique une perte d'information. Enfin, la normalisation Min/Max par caractéristique (V_2) est meilleure qu'une normalisation Min/Max pour toutes les caractéristiques d'entrée (V_3), contrairement à la normalisation STD où une normalisation pour l'ensemble des caractéristiques (V_5) est plus adaptée qu'une normalisation par caractéristique (V_4).

La rotation du repère de navigation améliore l'estimation des positions. En effet, en comparant $LSTM_{ALL,\ V_1}$ avec $LSTM_{ALL,\ V_6}$ et $LSTM_{POS,\ V_1}$ avec $LSTM_{POS,\ V_6}$, les versions sans rotation présentent des taux d'erreurs plus importants qu'avec la rotation du repère de navigation. Donc, cette méthode permet d'améliorer les estimations des positions.

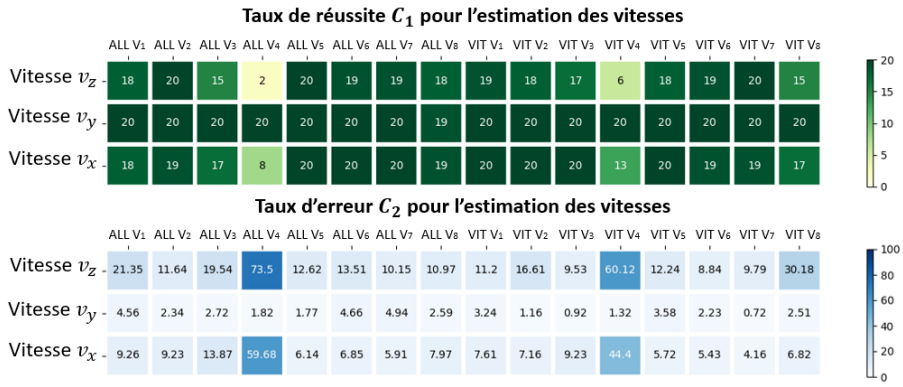

FIGURE 4.13 – Analyse de l'estimation des vitesses : taux de réussite $\mathcal{C}_1$ (4.14) et taux d'erreurs $\mathcal{C}_2$ (4.15) de $LSTM_{ALL,\ V_1-V_8}$ et $LSTM_{POS,\ V_1-V_8}$.

Résultats d'estimation des vitesses : D'après la figure 4.13, des observations similaires aux résultats de position peuvent être formulées. Les LSTM surpassent clairement le *Dead Reckoning* pour l'estimation des vitesses. Les réseaux spécialisés $LSTM_{VEL,\ V_1,\ V_{6-8}}$ sans normalisation surpassent très légèrement $LSTM_{ALL,\ V_1,\ V_{6-8}}$.
De même que pour la position, les versions V_3 ($MM(\mathcal{T}, \mathcal{M}, \mathcal{P})$) et V_4 ($STD(\mathcal{T})$, $STD(\mathcal{M})$, $STD(\mathcal{P})$) présentent les erreurs les plus importantes. Ainsi, ces normalisations ne sont pas adaptées à l'estimation des vitesses. La normalisation STD pour toutes les caractéristiques d'entrée (V_5) présente les meilleurs résultats parmi les différentes options de normalisation étudiées, en particulier pour la vitesse le long de l'axe z.
De plus, la rotation du repère de navigation (V_6) réduit les erreurs d'estimation des vitesses.

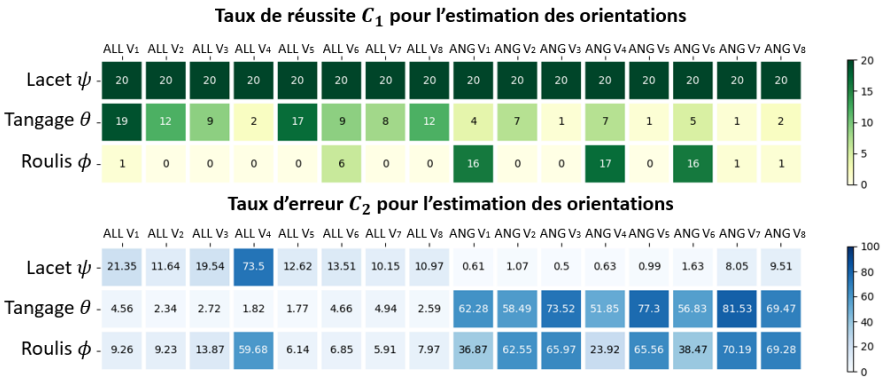

FIGURE 4.14 – Analyse de l'estimation des angles d'Euler : taux de réussite $\mathcal{C}_1$ (4.14) et taux d'erreurs $\mathcal{C}_2$ (4.15) de $LSTM_{ALL,\ V_1-V_8}$ et $LSTM_{POS,\ V_1-V_8}$.

Résultats d'estimation des angles d'Euler : L'estimation des angles d'Euler est plus mitigée d'après la figure 4.14. D'après le taux de réussite $\mathcal{C}_1$, certains LSTM ne parviennent pas à estimer les angles de roulis et de tangage par rapport au *Dead Reckoning*, mais estiment avec précision l'angle de lacet. De plus, les réseaux spécialisés $LSTM_{ANG}$ sont plus adaptés que les réseaux généralistes $LSTM_{ALL}$ pour l'estimation des angles de roulis et de lacet contrairement à l'angle de tangage.
La normalisation détériore considérablement la précision des estimations. Comme précédemment, la normalisation $STD(\mathcal{T}, \mathcal{M}, \mathcal{P})$ de $LSTM_{ALL}$ semble présenter les meilleures performances pour l'estimation du roulis et du tangage parmi les formes de normalisation étudiées.
De même que pour la position et la vitesse, la rotation du repère de navigation améliore les estimations.

En résumé, un LSTM est particulièrement approprié à l'estimation des positions et des vitesses d'un mortier de 120 mm. Selon les figures 4.12 - 4.14, les réseaux spécialisés, excepté pour l'orientation, n'améliorent pas significativement la précision des estimations et nécessitent d'entraîner trois réseaux spécialisés pour estimer la trajectoire complète du projectile. De plus, ces résultats montrent que la normalisation $STD(\mathcal{T}, \mathcal{M}, \mathcal{P})$ est plus appropriée pour estimer la trajectoire d'un projectile bien que les réseaux sans normalisation soient plus performants. Enfin, la rotation du repère de navigation est une méthode efficace pour optimiser l'estimation de la position et de la vitesse du projectile.

4.4.2 Impact du modèle de capteurs inertiels et de la rotation du repère de navigation sur la précision des estimations

D'après les résultats précédents, la normalisation n'est pas adaptée au problème d'estimation de la trajectoire d'un projectile contrairement à la rotation du repère de navigation. De plus, le jeu de données présenté dans la partie (*1.5*) contient deux types de modèle d'erreur des capteurs inertiels produisant ainsi les *mesures IMU*, utilisées jusqu'à présent, et les *mesures IMU DYN*, où les capteurs sont caractérisés par un modèle dynamique de second ordre. Cette section se concentre sur l'impact du modèle d'erreur de l'IMU ainsi que sur la rotation du repère de navigation sur la précision des estimations des LSTM.

À cet effet, quatre LSTM sont entraînés avec les spécifications données dans le tableau 4.3 pour estimer les positions, les vitesses et l'orientation d'un mortier de 120 mm comme $LSTM_{ALL}$. Les désignations suivantes sont utilisées dans cette section :

- $LSTM_{IMU,\ V_1}$: LSTM entraîné selon les spécifications données par la table 4.3. Les données d'entrée $(\mathcal{M}, \mathcal{P}, \mathcal{T})$ ne sont pas normalisées et aucune rotation du repère de navigation n'est effectuée.
- $LSTM_{IMU\ DYN,\ V_1}$, entraîné avec les *mesures IMU DYN* sans normalisation et sans rotation du repère de navigation.
- $LSTM_{IMU,\ V_6}$: réseau avec les mêmes spécifications que $LSTM_{IMU,V_1}$ (pas de normalisation, *mesures IMU*) mais avec la rotation du repère de navigation.
- $LSTM_{IMU\ DYN,\ V_6}$: réseau avec les mêmes spécifications que $LSTM_{IMU\ DYN,V_1}$ (pas de normalisation, *mesures IMU DYN*) et la rotation du repère de navigation est utilisée.

TABLE 4.3 – Caractéristiques d'entraînement de $LSTM_{IMU,\ V_1}$, $LSTM_{IMU\ DYN,\ V_1}$, $LSTM_{IMU,\ V_6}$ et de $LSTM_{IMU\ DYN,\ V_6}$ sur des trajectoires de mortier de 120 mm.

Jeu de données	Jeu de données d'entraînement :	4 000 simulations
	Jeu de données de validation :	400 simulations
	Jeu de données de test :	400 simulations
Caractéristiques des données d'entrée	$S_{eq\ len}$:	20 pas de temps
	Données d'entrée :	$I_{n\ Features} = (\mathcal{M}, \mathcal{P}, \mathcal{T}) \in \mathbb{R}^{16}$
Caractéristiques d'entraînement	Fonction de coût :	*Mean Squared Error (MSE)*
	Algorithme d'optimisation :	Adam
	Taux d'apprentissage :	$1e^{-4}$

Impact de la rotation du repère de navigation avec les mesures IMU

Cette section analyse $LSTM_{IMU,\ V_1}$ (pas de rotation), $LSTM_{IMU,\ V_6}$ (rotation) et le *Dead Reckoning* (4.10) - (4.12) pour l'estimation des trajectoires d'un mortier de 120 mm. Ces réseaux, présentés dans le tableau 4.3, sont entraînés avec les *mesures IMU*.

Performances d'estimation de $LSTM_{IMU,\ V_1}$: Les figures 4.15 - 4.17 présentent la RMSE (4.13) du $LSTM_{IMU,\ V_1}$ en fonction de la RMSE du *Dead Reckoning* pour l'estimation des 400 trajectoires du jeu de données de test.

Selon les figures 4.15 et 4.16, $LSTM_{IMU,\ V_1}$ surpasse considérablement le *Dead Reckoning* pour l'estimation de la position et de la vitesse le long des axes x et y comme la plupart des marqueurs sont situés dans la partie supérieure. Cependant, les performances sont légèrement moins importantes pour l'estimation de la position et de la vitesse le long

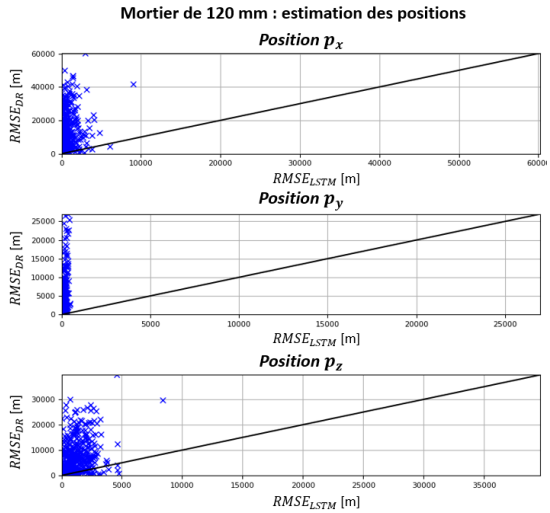

FIGURE 4.15 – Positions estimées : $RMSE_{LSTM}$ en fonction de $RMSE_{DR}$.

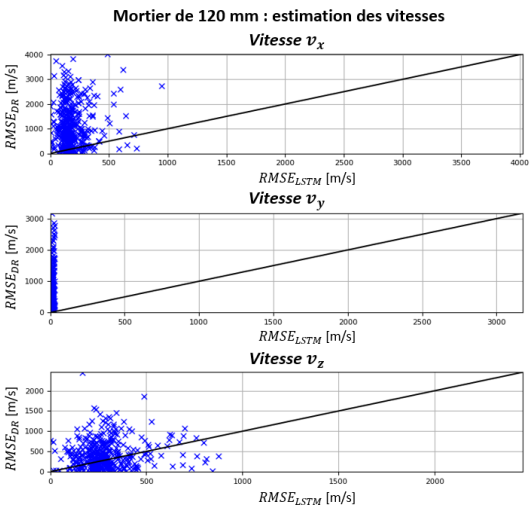

FIGURE 4.16 – Vitesses estimées : $RMSE_{LSTM}$ en fonction de $RMSE_{DR}$.

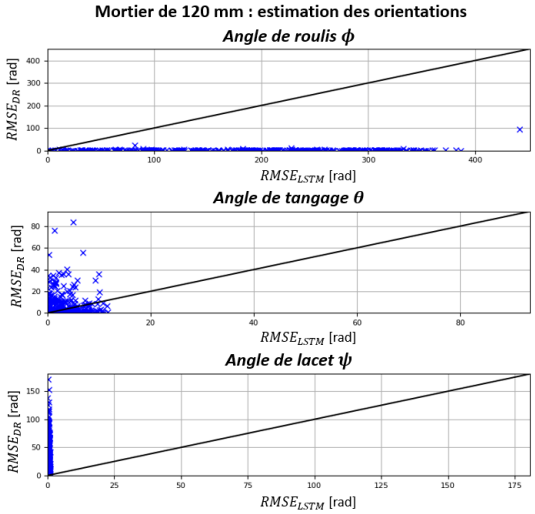

FIGURE 4.17 – Angles d'Euler estimés : $RMSE_{LSTM}$ en fonction de $RMSE_{DR}$.

de l'axe z car de nombreux marqueurs sont situés dans la partie inférieure. D'après la figure 4.17, $LSTM_{IMU,\ V_1}$ ne parvient pas à estimer correctement l'angle de roulis comme la majorité des marqueurs sont situés dans la partie inférieure, mais estime adéquatement les angles de tangage et de lacet du projectile.

Ces premières observations confirment que le LSTM généralise avec succès les caractéristiques apprises sur un grand ensemble de données, en particulier pour l'estimation de la position et de la vitesse.

Performances d'estimation de $LSTM_{IMU,\ V_1}$, $LSTM_{IMU,\ V_6}$ et du Dead Reckoning :

Les figures 4.18 - 4.20 présentent les distributions des erreurs moyennes $\bar{e}$ et les écarts types correspondants σ obtenus par le $LSTM_{IMU,\ V_1}$, le $LSTM_{IMU,\ V_6}$ et l'algorithme de *Dead Reckoning*.

L'*erreur moyenne* $\bar{e}$ est utilisée pour valider les trois modèles en visualisant la fiabilité du modèle considéré. Elle est évaluée telle que :

$$\bar{e} = \frac{1}{N} \sum_{k=1}^{N} (x_{ref} - \hat{x}) \tag{4.16}$$

L'*écart type* σ est employé pour évaluer la dispersion des erreurs en fonction du modèle d'erreur considéré. Il est évalué tel que :

$$\sigma = \sqrt{\frac{1}{N} \sum_{k=1}^{N} ([x_{ref} - \hat{x}] - \bar{e})^2} \tag{4.17}$$

avec x_{ref} la quantité de référence, $\hat{x}$ la quantité estimée et N le nombre d'échantillons dans la simulation considérée.

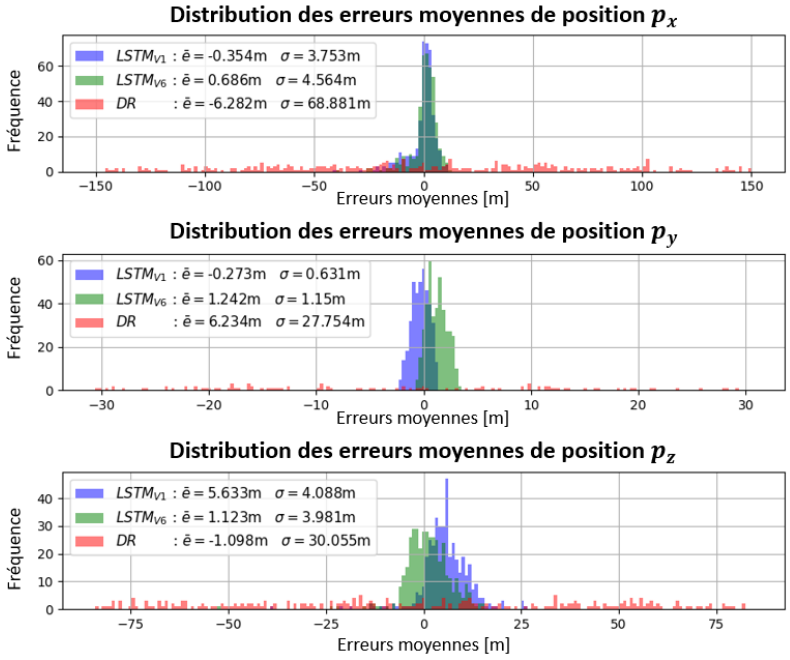

FIGURE 4.18 – Distribution des erreurs moyennes de position de $LSTM_{IMU,\ V_1}$ (bleu), $LSTM_{IMU,\ V_6}$ (vert) et du *Dead Reckoning* (rouge).

Analyse de la distribution des erreurs d'estimation de la position (figure 4.18) : La dispersion des erreurs du *Dead Reckoning* est très importante contrairement à celle des LSTM (vert et bleu), regroupées autour de quelques mètres le long des trois axes.
Les erreurs de $LSTM_{IMU,\ V_1}$ (pas de rotation) sont centrées sur zéro suivant les axes x et y contrairement aux erreurs de $LSTM_{IMU,\ V_6}$ (rotation) centrées sur zéro suivant les trois axes. Donc, la position p_z estimée par $LSTM_{IMU,\ V_1}$ (pas de rotation) est légèrement biaisée et la rotation du repère de navigation corrige ce biais en permettant au LSTM d'être plus sensible à la gravité terrestre. De plus, aucune valeur aberrante n'apparaît sur les distributions des erreurs de position le long des trois axes pour les deux LSTM.
Les distributions des erreurs moyennes de position montrent qu'un LSTM est largement plus précis qu'un algorithme de *Dead Reckoning*. De plus, les performances des deux LSTM sont similaires pour l'estimation de la position, à l'exception de l'axe z où la rotation du repère de navigation améliore l'estimation p_z.

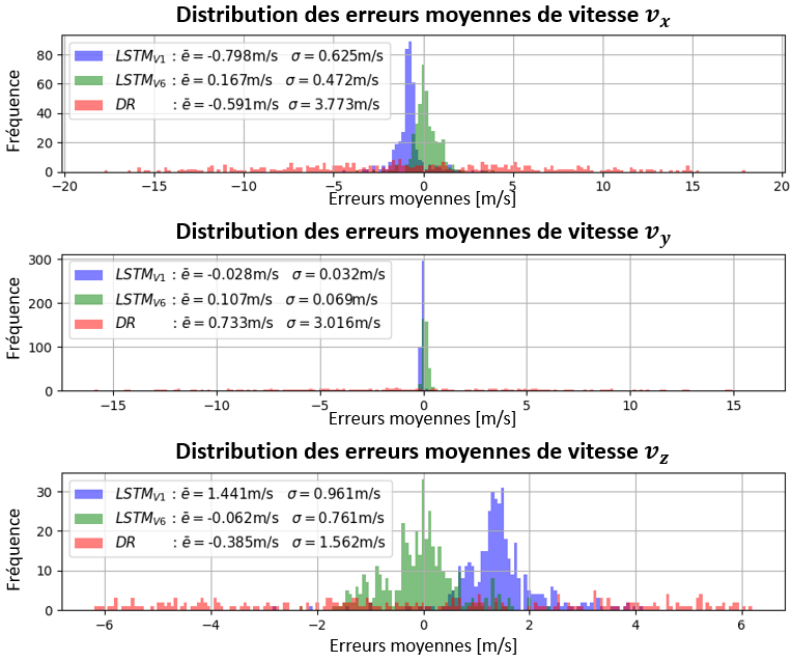

FIGURE 4.19 – Distribution des erreurs moyennes de vitesse de $LSTM_{IMU,\ V_1}$ (bleu), $LSTM_{IMU,\ V_6}$ (vert) et du *Dead Reckoning* (rouge).

Analyse de la distribution des erreurs d'estimation de la vitesse (figure 4.19) : Comme pour les positions, les LSTM sont nettement plus précis que le *Dead Reckoning* pour l'estimation des vitesses d'après les dispersions des erreurs. De plus, seules les erreurs de $LSTM_{IMU,\ V_6}$ (rotation) sont centrées sur zéro le long des axes x et z. Ainsi, les

estimations de $LSTM_{IMU,\ V_1}$ (pas de rotation) sont légèrement biaisées et peuvent être corrigées par la rotation du repère de navigation. La figure 4.19 confirme l'apport des LSTM pour l'estimation des vitesses par rapport à un *Dead Reckoning*, ainsi que l'intérêt de la rotation du repère de navigation pour améliorer l'estimation de v_x et v_z.

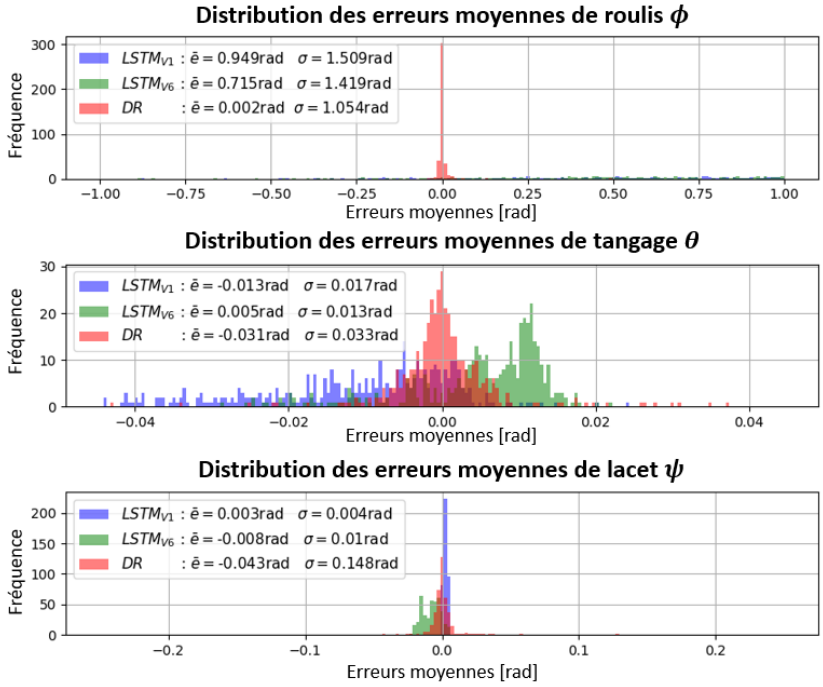

FIGURE 4.20 – Distribution des erreurs moyennes d'orientation de $LSTM_{IMU,\ V_1}$ (bleu), $LSTM_{IMU,\ V_6}$ (vert) et du *Dead Reckoning* (rouge).

Analyse de la distribution des erreurs d'estimation de l'orientation (figure 4.20) : La distribution des erreurs indique que les LSTM ne parviennent pas à estimer l'angle de roulis du projectile. En effet, les LSTM ne captent pas les variations d'angle de roulis, même si la vitesse de rotation du mortier de 120 mm est faible comparée à d'autres projectiles. Concernant l'angle de tangage, la dispersion des erreurs de $LSTM_{IMU,\ V_1}$ (pas de rotation) est importante et celle de $LSTM_{IMU,\ V_6}$ (rotation) n'est pas centrée sur zéro. Ainsi, les LSTM ne sont pas la méthode la plus appropriée pour estimer cet angle. Toutefois, $LSTM_{IMU,\ V_1}$ (pas de rotation) surpasse significativement le *Dead Reckoning* et le $LSTM_{IMU,\ V_6}$ (rotation) pour l'estimation de l'angle de lacet.
D'après la figure 4.20, les LSTM ne sont pas adaptés à l'estimation de l'orientation du projectile. En effet, les LSTM estiment correctement l'angle de lacet, qui est de très faible variation. Une étude spécifique aux angles d'Euler est proposée dans la suite de ce chapitre.

Pour conclure cette première analyse, les figures 4.18 - 4.20 indiquent que le LSTM entraîné à partir des *mesures IMU* est parfaitement adapté pour estimer les positions et les vitesses d'un mortier de 120 mm, et de plus, surpasse significativement un algorithme de *Dead Reckoning*. Néanmoins, l'estimation de l'orientation du projectile par un LSTM est mitigée et nécessite une attention particulière. Enfin, la rotation du repère de navigation permet d'optimiser les positions et les vitesses estimées, notamment le long de l'axe z.

Impact de la rotation du repère de navigation avec les mesures *IMU DYN*

Le jeu de données comprend des *mesures IMU DYN* pour lesquelles la dynamique des capteurs est prise en compte. L'influence de ce modèle d'erreur est examinée avec les réseaux entraînés avec les *mesures IMU DYN*, à savoir, $LSTM_{IMUDYN,\ V_1}$ (pas de rotation), $LSTM_{IMUDYN,\ V_6}$ (rotation) et l'algorithme *Dead Reckoning* (4.10) - (4.12). Les caractéristiques d'entraînement sont résumées dans le tableau 4.3.

Les figures 4.21 - 4.23 présentent les distributions des erreurs moyennes (4.16) d'estimation des positions, des vitesses et des angles d'Euler obtenues par $LSTM_{IMU\ DYN,\ V_1}$, $LSTM_{IMU\ DYN,\ V_6}$, et l'algorithme de *Dead Reckoning* (4.10) - (4.12).

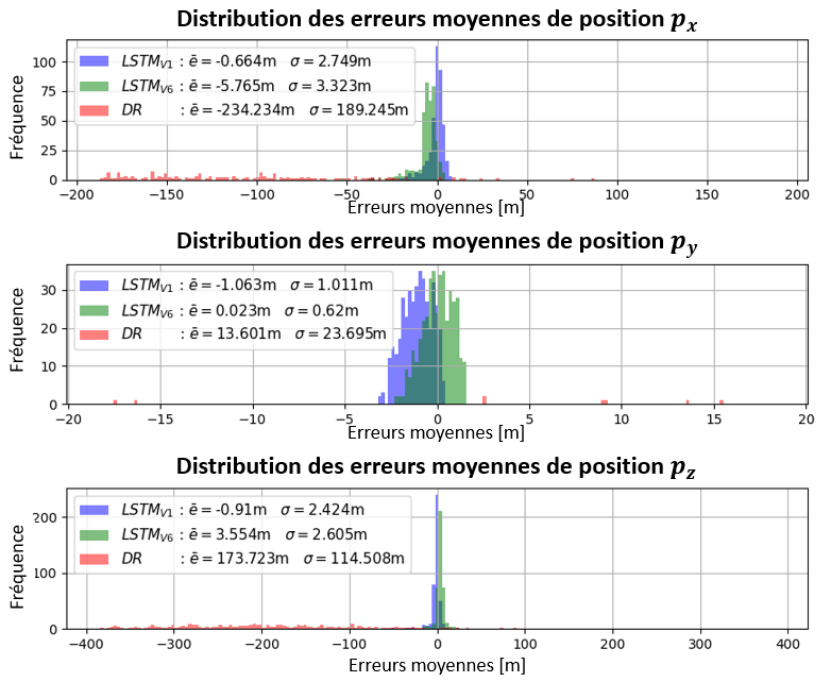

FIGURE 4.21 – Distribution des erreurs moyennes de position de $LSTM_{IMU\ DYN,\ V_1}$ (bleu), $LSTM_{IMU\ DYN,\ V_6}$ (vert) et du *Dead Reckoning* (rouge).

Analyse de la distribution des erreurs d'estimation de la position (figure 4.21) : Le *Dead Reckoning* dévie pour l'estimation des positions avec les *mesures IMU DYN* comme la plupart des erreurs sont situées sur le côté gauche. Les *mesures IMU DYN* conduisent à des biais systématiques dans les estimations du *Dead Reckoning*. Les erreurs moyennes de position de $LSTM_{IMU\ DYN,\ V_1}$ le long des axes x et z sont centrées sur zéro, contrairement à l'axe y. À l'inverse, les distributions de $LSTM_{IMU\ DYN,\ V_6}$ (rotation) sont centrées sur zéro pour l'axe y. Ainsi, la rotation du repère de navigation optimise principalement l'estimation de la position le long de l'axe y. La figure 4.21 met en évidence l'apport d'un LSTM pour l'estimation des positions d'un projectile, même avec des *mesures IMU DYN*, contrairement au *Dead Reckoning* qui dévie complètement. De plus, la rotation du repère de navigation optimise l'estimation de la position suivant l'axe y.

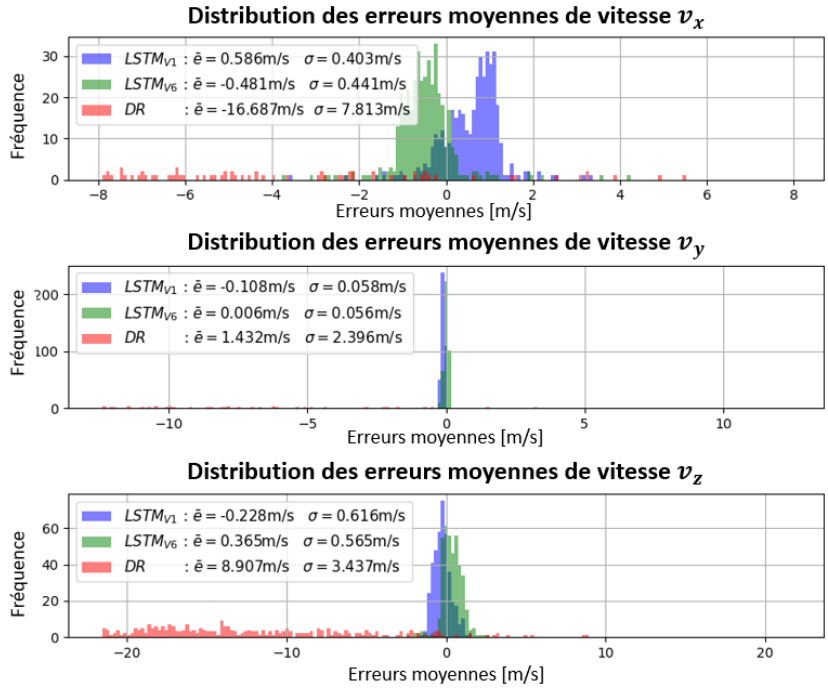

FIGURE 4.22 – Distribution des erreurs moyennes de vitesse de $LSTM_{IMU\ DYN,\ V_1}$ (bleu), $LSTM_{IMU\ DYN,\ V_6}$ (vert) et du *Dead Reckoning* (rouge).

Analyse de la distribution des erreurs d'estimation de la vitesse (figure 4.22) : Les *mesures IMU DYN* impliquent des erreurs importantes pour l'estimation des vitesses par le *Dead Reckoning* (rouge) contrairement aux LSTM (vert et bleu). De plus, le centrage des erreurs moyennes de $LSTM_{IMU\ DYN,\ V_6}$ (rotation) montre que la rotation du repère de navigation améliore l'estimation de v_x et v_y comparée à $LSTM_{IMU\ DYN,\ V_1}$ (sans rotation).

Comme pour l'estimation de la position, la distribution des erreurs moyennes de vitesse met en évidence la contribution du LSTM pour l'estimation des vitesses du projectile par rapport à un algorithme *Dead Reckoning*. En effet, un LSTM est précis pour estimer la vitesse du projectile avec les *mesures IMU DYN*.

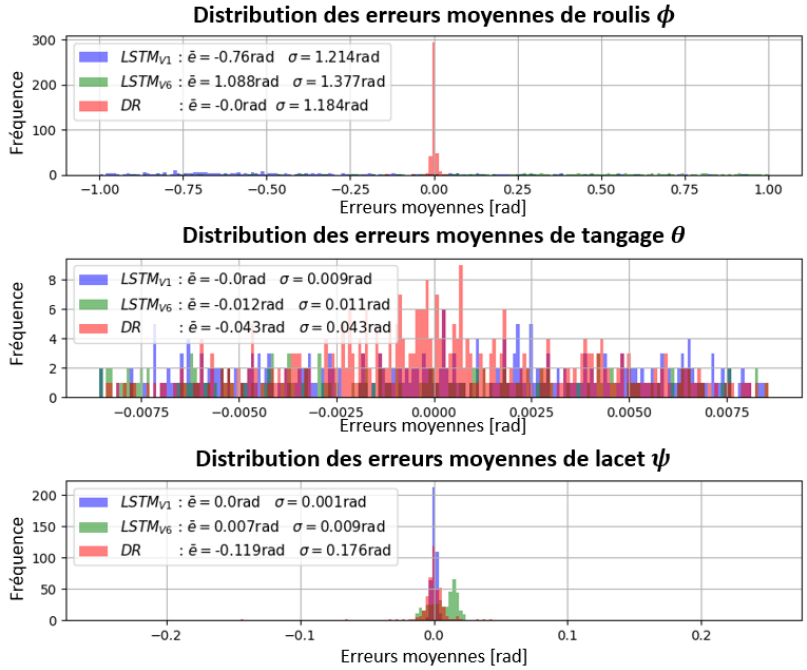

FIGURE 4.23 – Distribution des erreurs moyennes d'orientation de $LSTM_{IMU\ DYN,\ V_1}$ (bleu), $LSTM_{IMU\ DYN,\ V_6}$ (vert) et du *Dead Reckoning* (rouge).

Analyse de la distribution des erreurs d'estimation de l'orientation (figure 4.23) : De même que pour les orientations précédentes, les deux LSTM ne parviennent pas à estimer l'angle de roulis du projectile d'après les dispersions des erreurs. Néanmoins, les LSTM surpassent légèrement le *Dead Reckoning* pour l'estimation de l'angle de tangage mais l'analyse est encore mitigée. Toutefois, le $LSTM_{IMU,\ V_1}$ surpasse significativement le *Dead Reckoning* et le $LSTM_{IMU,\ V_6}$ (rotation) pour l'estimation des angles de lacet.

Pour conclure cette seconde analyse, un LSTM estime avec précision les positions et les vitesses d'un projectile sur un grand ensemble de données et malgré des données inertielles dynamiques, provoquant à l'inverse la divergence du *Dead Reckoning*. Néanmoins, comme pour les *mesures IMU*, un LSTM n'est pas une approche efficace pour estimer l'orientation d'un projectile. Enfin, la rotation du repère de navigation avec les *mesure IMU DYN* permet d'optimiser la position le long de l'axe y et la vitesse le long des axes x et y.

Métrique d'évaluation

Les performances globales de $LSTM_{IMU\ V_1,\ V_6}$, $LSTM_{IMU\ DYN\ V_1,\ V_6}$ et de l'algorithme *Dead Reckoning* sont évaluées à l'aide de deux critères d'évaluation, calculés pour chaque simulation du jeu de données de test. Les deux critères d'évaluation sont :

- l'*erreur moyenne absolue (Mean Absolute Error - MAE)* définie telle que :

$$MAE = \frac{1}{N}\sum_{k=1}^{N} |x_{ref} - \widehat{x}| \tag{4.18}$$

  avec x_{ref} la grandeur de référence donnée par BALCO, $\widehat{x}$ la quantité estimée et N le nombre d'échantillons dans la simulation.
- le *score* noté $\mathcal{C}_{score}$, représentant le nombre de simulations du jeu de données où la méthode considérée obtient la plus petite RMSE (4.13) parmi les autres méthodes d'estimation. À titre d'exemple, le critère $\mathcal{C}_{score}$ d'une méthode $\mathcal{X}$ comparée à des méthodes $\mathcal{Y}$ et $\mathcal{Z}$ est évalué tel que :

$$\mathcal{C}_{score} = \sum_{k=1}^{N_{sim}} \left(min\left[MSE_{\mathcal{X}}, RMSE_{\mathcal{Y}}, RMSE_{\mathcal{Z}}\right] = RMSE_{\mathcal{X}}\right) \tag{4.19}$$

La MAE est évaluée sur l'ensemble du jeu de données de test tel que :

$$\mathcal{C}_{MAE} = \frac{1}{N_{sim}}\sum_{k=1}^{N_{sim}} MAE_{sim_k} \tag{4.20}$$

avec N_{sim} le nombre de simulations dans le jeu de données de test.

Analyse du critère $\mathcal{C}_{MAE}$: La figure 4.24 présente le critère $\mathcal{C}_{MAE}$ (4.20) évalué sur l'ensemble du jeu de données de test pour les méthodes d'estimation $LSTM_{IMU,\ V_1,\ V_6}$, $LSTM_{IMU\ DYN,\ V_1,\ V_6}$ et l'algorithme de *Dead Reckoning*.

D'après la figure 4.24, les LSTM surpassent largement le *Dead Reckoning* pour l'estimation des positions et des vitesses le long des trois axes, à la fois avec les *mesures IMU* et les *mesures IMU DYN*. D'une part, en se concentrant sur les résultats avec les *mesures IMU*, les erreurs moyennes de position des LSTM sont d'environ 4,03 m tandis que les erreurs moyennes de position du *Dead Reckoning* sont de 45,39 m. D'autre part, selon les résultats avec les *mesures IMU DYN*, les erreurs moyennes de position des LSTM sont d'environ 3,31 m. De plus, ce critère d'erreur confirme les observations précédentes, la rotation du repère de navigation améliore l'estimation de p_z, v_z et v_z pour les LSTM

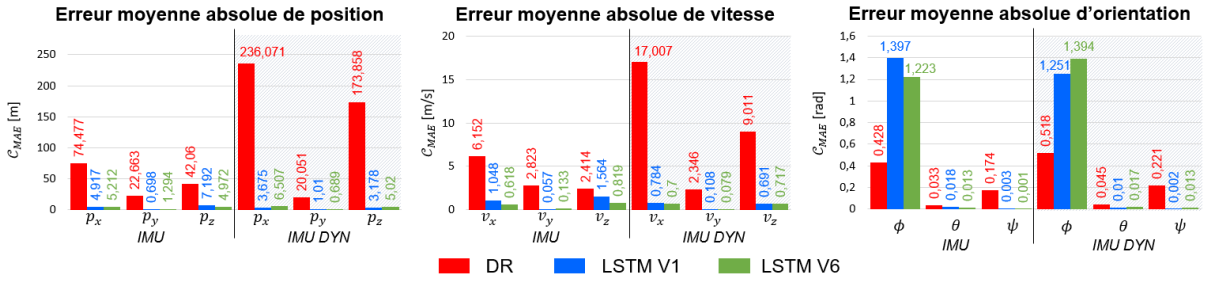

FIGURE 4.24 – Critère d'erreur $\mathcal{C}_{MAE}$ (4.20) de $LSTM_{IMU,\ V_1}$, $LSTM_{IMU,\ V_6}$, $LSTM_{IMU\ DYN,\ V_1}$, $LSTM_{IMU\ DYN,\ V_6}$ et du *Dead Reckoning*.

entraînés avec les *mesures IMU*, et l'estimation de p_y, v_x et v_y pour les LSTM entraînés avec les *mesures IMU DYN*. La figure 4.24 confirme qu'un LSTM n'est pas adapté à l'estimation de l'angle de roulis contrairement à l'angle de tangage ou l'angle de lacet. De plus, la rotation du repère de navigation améliore légèrement l'estimation de l'angle de tangage avec les *mesures IMU DYN*.

Analyse du critère $\mathcal{C}_{score}$: La figure 4.25 présente le score $\mathcal{C}_{score}$ (4.19), évalué sur l'ensemble du jeu de données de test pour $LSTM_{IMU,\ V_1}$, $LSTM_{IMU\ DYN,\ V_1}$, $LSTM_{IMU\ DYN,\ V_6}$ et le *Dead Reckoning*.

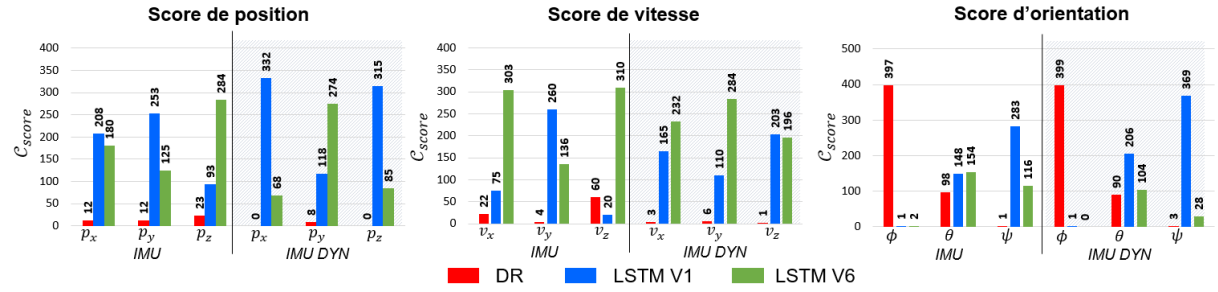

FIGURE 4.25 – Critère d'erreur $\mathcal{C}_{score}$ (4.19) de $LSTM_{IMU,\ V_1}$, $LSTM_{IMU,\ V_6}$, $LSTM_{IMU\ DYN,\ V_1}$, $LSTM_{IMU\ DYN,\ V_6}$ et du *Dead Reckoning*.

D'après la figure 4.25, $LSTM_{IMU,\ V_{1,6}}$ et $LSTM_{IMU\ DYN,\ V_{1,6}}$ surpassent de manière significative le *Dead Reckoning* pour l'estimation des positions et des vitesses. De plus, la rotation du repère de navigation améliore p_z, v_x et v_z en cas de *mesures IMU* et améliore p_y, v_x et v_y en cas de *mesures IMU DYN*. Selon les scores pour l'estimation des angles d'Euler, les LSTM sont efficaces pour estimer les angles de tangage et de lacet, pour les deux types de données inertielles. Cependant, la rotation du repère de navigation détériore l'estimation de l'orientation du projectile, particulièrement l'estimation de l'angle de lacet.

Pour conclure sur l'estimation de la trajectoire d'un mortier de 120 mm et d'après les figures 4.15 - 4.25, un LSTM est une approche précise pour estimer les positions et les vitesses d'un mortier dans un environnement sans GNSS. Cependant, un LSTM n'est pas la méthode optimale pour l'estimation de l'orientation. De plus, un LSTM est capable de généraliser les caractéristiques apprises sur un grand ensemble de trajectoires ainsi que d'apprendre la trajectoire à partir de différents modèles de capteurs. Enfin, les analyses $\mathcal{C}_{MAE}$ et $\mathcal{C}_{score}$ indiquent que la rotation du repère de navigation est une méthode appropriée pour optimiser les estimations de position et de vitesse.

Erreurs au point d'impact

À présent, les erreurs au point d'impact de $LSTM_{IMU,\ V_1,\ V_6}$, $LSTM_{IMU\ DYN,\ V_1,\ V_6}$, et du *Dead Reckoning* (4.10) - (4.12) sont évaluées. Les erreurs au point d'impact représentent les erreurs de position (p_x, p_y) à l'instant final d'un tir.

Erreurs au point d'impact avec les mesures IMU : La figure 4.26 montre les erreurs au point d'impact de $LSTM_{IMU,V_1}$, $LSTM_{IMU,V_6}$ et du *Dead Reckoning*, et la figure 4.27 présente les localisations de ces erreurs.

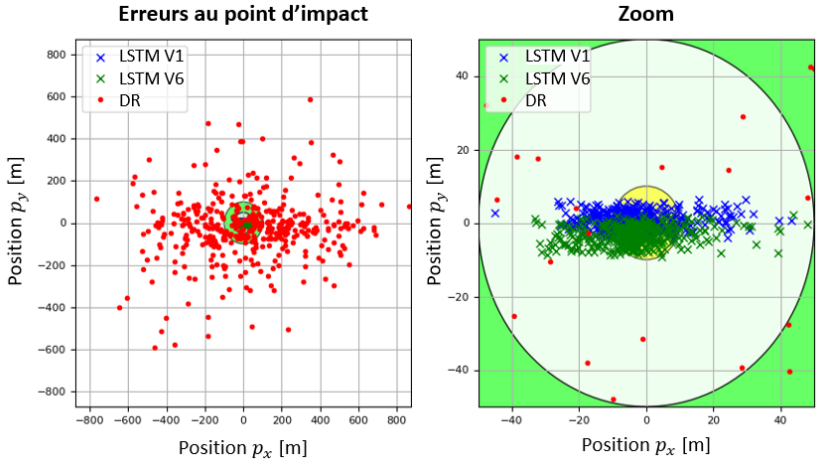

FIGURE 4.26 – Erreurs au point d'impact obtenues par $LSTM_{IMU,\ V_1}$ (croix bleues), $LSTM_{IMU,\ V_6}$ (croix vertes) et le *Dead Reckoning* (points rouge).

D'après les figures 4.26 et 4.27, la majorité des erreurs au point d'impact du *Dead Reckoning* sont supérieures à 100 m contrairement aux LSTM où les erreurs sont strictement inférieures à 100 m. La plupart des erreurs au point d'impact de $LSTM_{IMU,\ V_6}$ (rotation) sont inférieures à 5 m, contrairement à $LSTM_{IMU,\ V_1}$ où la majorité des erreurs

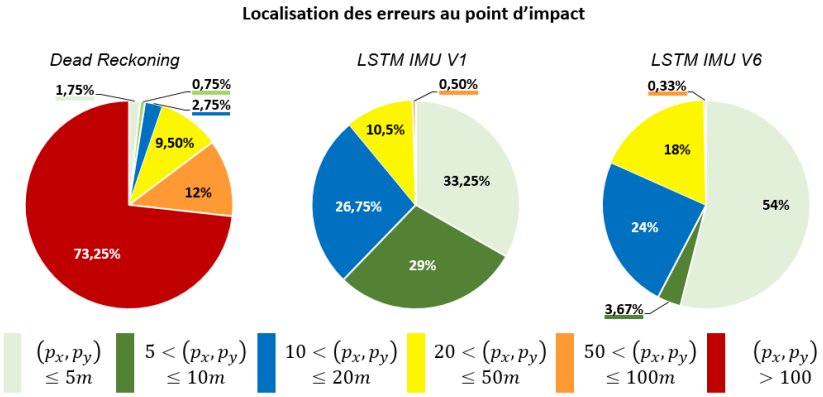

FIGURE 4.27 – Localisation des erreurs au point d'impact du $LSTM_{IMU,\ V_1}$, du $LSTM_{IMU,\ V_6}$ et du *Dead Reckoning* : $e_f \leq 5\ m$ (vert clair), $5\ m < e_f \leq 10\ m$ (vert foncé), $10\ m < e_f \leq 20\ m$ (bleu), $20\ m < e_f \leq 50\ m$ (jaune), $50\ m < e_f \leq 100\ m$ (orange), $e_f > 100\ m$ (rouge).

sont inférieures à 20 m. Ainsi, la rotation du repère de navigation permet de réduire significativement les erreurs au point d'impact. De plus, les erreurs au point d'impact des LSTM et des mortiers guidés par GNSS du commerce sont du même ordre de grandeur.

Erreurs au point d'impact avec les mesures IMU DYN : La figure 4.28 montre les erreurs au point d'impact de $LSTM_{IMU\ DYN,V_1}$, $LSTM_{IMU\ DYN,V_6}$ et du *Dead Reckoning*, et la figure 4.29 la localisation de ces erreurs.

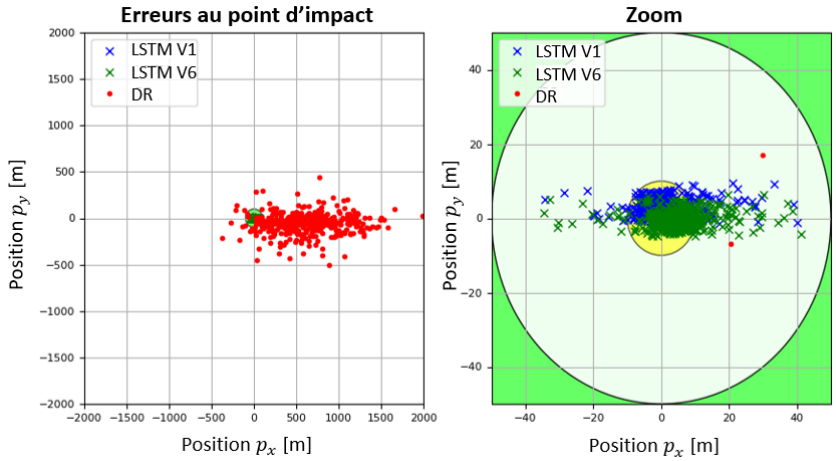

FIGURE 4.28 – Erreurs au point d'impact obtenues par $LSTM_{IMU\ DYN,\ V_1}$ (croix bleues), $LSTM_{IMU\ DYN,\ V_6}$ (croix vertes) et le *Dead Reckoning* (points rouge).

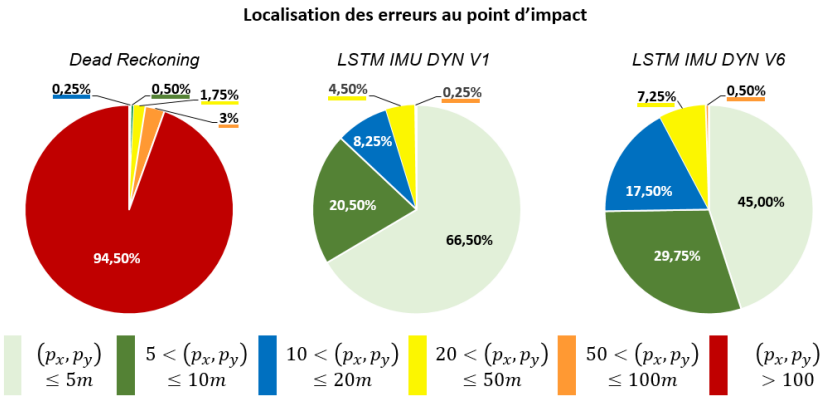

FIGURE 4.29 – Localisation des erreurs au point d'impact du $LSTM_{IMU\ DYN,\ V_1}$, du $LSTM_{IMU\ DYN,\ V_6}$ et du *Dead Reckoning* : $e_f \leq 5\ m$ (vert clair), $5\ m < e_f \leq 10\ m$ (vert foncé), $10\ m < e_f \leq 20\ m$ (bleu), $20\ m < e_f \leq 50\ m$ (jaune), $50\ m < e_f \leq 100\ m$ (orange), $e_f > 100\ m$ (rouge).

Les figures 4.28 et 4.29 confirment les analyses précédentes. Les *mesures IMU DYN* provoquent la divergence du *Dead Reckoning*. De plus, la rotation du repère de navigation détériore la précision des estimations. En effet, dans le cas du $LSTM_{IMU\ DYN,\ V_1}$, 66.5% des simulations ont des erreurs au point d'impact inférieures à 5 m alors que 45% des simulations ont des erreurs au point d'impact inférieures à 5 m avec le $LSTM_{IMU\ DYN,\ V_6}$.

Les figures 4.15 - 4.29 montrent qu'un LSTM entraîné à partir de mesures inertielles, des paramètres de vol et d'un vecteur de temps, est une approche précise d'estimation des positions et des vitesses avec des erreurs inférieures à dix mètres dans un environnement sans GNSS. Néanmoins, les figures 4.15 - 4.29 confirment qu'un LSTM n'est pas approprié pour estimer l'orientation d'un projectile. Une méthode classique de navigation sera privilégiée. De plus, selon l'analyse de la rotation du repère de navigation, cette méthode est intéressante dans le cas des *mesures IMU* et permet d'optimiser fortement l'estimation des positions et des vitesses du mortier de 120 mm.

4.5 Généralisation à d'autres types de projectiles

Cette partie vise à généraliser les résultats précédents à l'obus de 155 mm, caractérisé par une trajectoire balistique de plusieurs dizaines de kilomètres mais avec un fort taux de rotation comparé au mortier de 120 mm, ainsi qu'à appliquer cette méthode d'estimation au projectile de 40 mm et au *Basic Finner*, tous deux caractérisés par des tirs tendus.

4.5.1 Estimation de la trajectoire d'un obus 155 mm par un LSTM

Un obus de 155 mm est caractérisé par une trajectoire balistique d'une portée bien plus importante que celle d'un mortier pouvant aller jusqu'à 20 000 m. De plus, comme l'obus de 155 mm ne dispose pas d'ailettes, sa vitesse de rotation est bien plus importante que celle d'un mortier de 120 mm. Comme pour l'estimation de la trajectoire d'un mortier de 120 mm, plusieurs LSTMs sont entraînés avec des normalisations et la rotation du repère de navigation. Seuls les résultats de $LSTM_{obus_{155}}$, sans normalisation et sans rotation sont proposés comme des conclusions similaires à celles du mortier de 120 mm peuvent être formulées concernant ces méthodes de prétraitement des données. $LSTM_{obus_{155}}$ estime la

TABLE 4.4 – Caractéristiques d'entraînement de $LSTM_{obus_{155}}$ sur des trajectoires d'obus de 155 mm.

Jeu de données	Jeu de données d'entraînement :	1 000 simulations
	Jeu de données de validation :	100 simulations
	Jeu de données de test :	100 simulations
Caractéristiques des données d'entrée	$B_{atch\ size}$:	64 ($S_{eq\ len}$:10 pas de temps)
	Données d'entrée :	$I_{n\ Features} = (\mathcal{M}, \mathcal{P}, \mathcal{T}) \in \mathbb{R}^{16}$
Caractéristiques d'entraînement	Fonction de coût :	*Mean Squared Error (MSE)*
	Algorithme d'optimisation :	Adam

position, la vitesse et les angles d'Euler d'un obus de 155 mm à partir des données d'entrée $I_{n\ Features} = (\mathcal{M}, \mathcal{P}, \mathcal{T}) \in \mathbb{R}^{16}$. Les paramètres de vol de cette munition sont l'angle de roulis initial, la vitesse initiale et l'angle d'élévation du canon. Les caractéristiques de l'entraînement de $LSTM_{obus_{155}}$ sont présentées dans le tableau 4.4.

Résultats préliminaires sur une trajectoire d'obus de 155 mm

Les figures 4.30 - 4.31 présentent la position et les angles d'Euler estimés ainsi que les erreurs obtenues par le $LSTM_{obus_{155}}$ et le *Dead Reckoning* pour une trajectoire d'obus de 155 mm.

Les figures 4.30 - 4.31 montrent qu'un LSTM parvient à capturer globalement les dynamiques des positions de cette munition, bien que les résultats soient moins performants que ceux obtenus précédemment. Des observations similaires peuvent être formulées pour

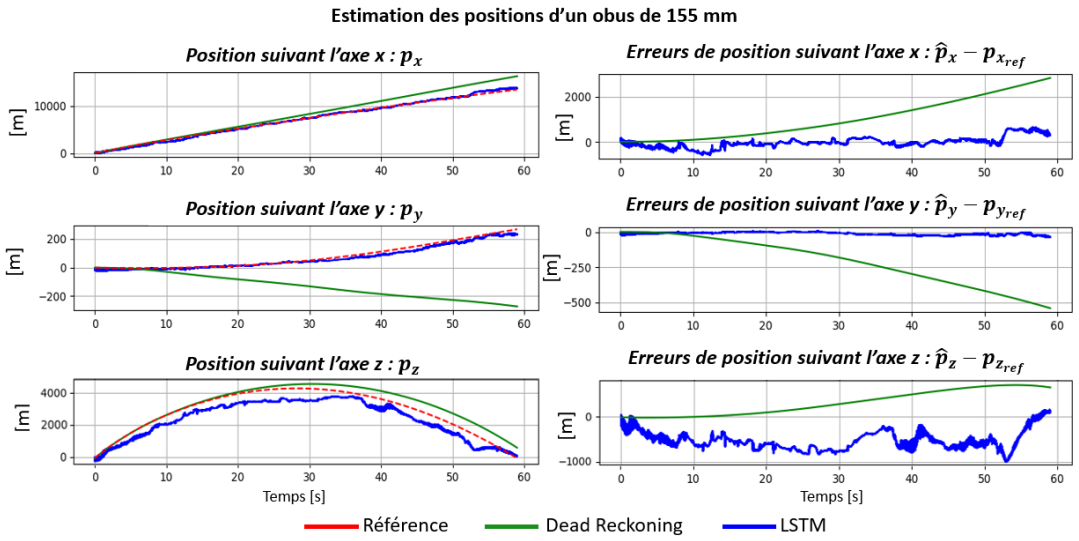

FIGURE 4.30 – Position estimée et erreurs associées [m] obtenues par l'algorithme de *Dead Reckoning* (vert), le $LSTM_{obus_{155}}$ (bleu) et la position de référence (rouge).

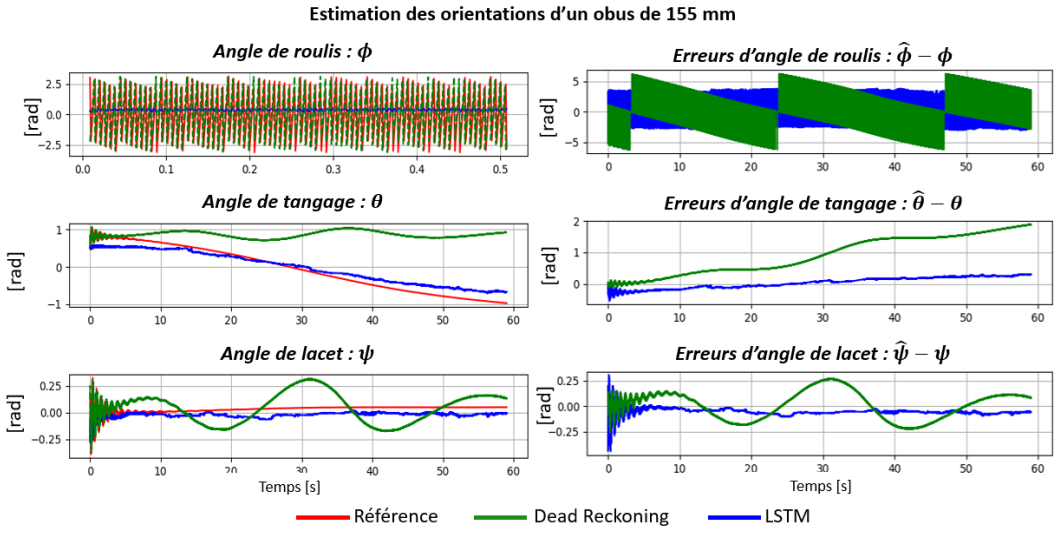

FIGURE 4.31 – Orientation estimée et erreurs associées [rad] obtenues par l'algorithme de *Dead Reckoning* (vert), le $LSTM_{obus_{155}}$ (bleu) et l'orientation de référence (rouge).

la vitesse. Toutefois, comme pour le mortier de 120 mm, le LSTM ne parvient pas à estimer l'orientation d'un obus de 155 mm. En effet, le LSTM capture globalement les dynamiques des angles de tangage et de lacet contrairement à l'angle de roulis, où le LSTM n'estime aucune dynamique.

Généralisation à l'ensemble du jeu de données de test

Afin de valider ces premières observations, les performances de $LSTM_{obus_{155}}$ sont évaluées sur l'ensemble du jeu de données de test avec les critères $\mathcal{C}_{MAE}$ (4.20) et $\mathcal{C}_{score}$ (4.19). Les figures 4.32 et 4.33 présentent ces deux critères de performance obtenus par le $LSTM_{obus_{155}}$ et le *Dead Reckoning* pour l'estimation des positions, des vitesses et des angles d'Euler d'un obus de 155 mm.

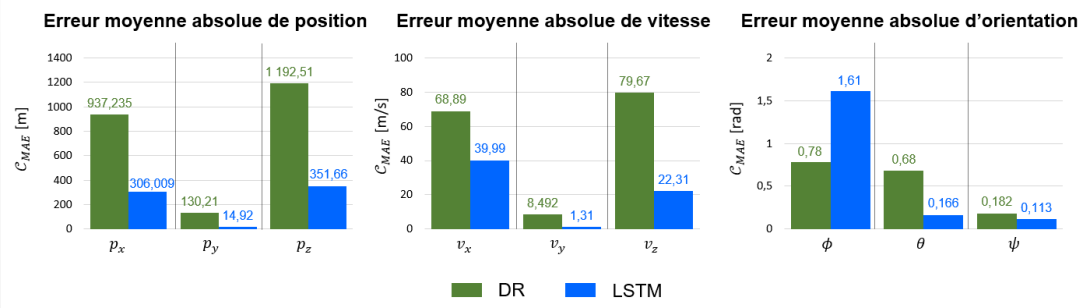

FIGURE 4.32 – Critère d'erreur $\mathcal{C}_{MAE}$ (4.20) de $LSTM_{obus_{155}}$ et du *Dead Reckoning*.

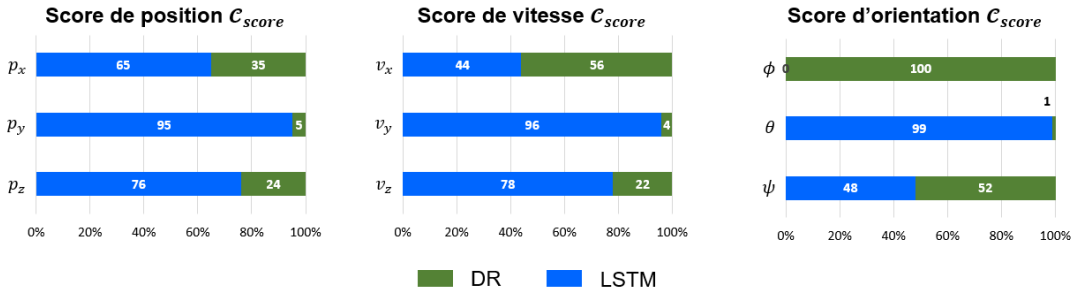

FIGURE 4.33 – Critère d'erreur $\mathcal{C}_{score}$ (4.19) de $LSTM_{obus_{155}}$ et du *Dead Reckoning*.

Les figures 4.32 et 4.33 montrent que $LSTM_{obus_{155}}$ surpasse le *Dead Reckoning* pour l'estimation des positions et des vitesses d'un obus de 155 mm. Toutefois, les erreurs moyennes de position et de vitesse $\mathcal{C}_{MAE}$ du $LSTM_{obus_{155}}$ sont relativement importantes malgré les fortes dynamiques de cette munition. En effet, les erreurs moyennes de positions de $LSTM_{obus_{155}}$ sont d'environ 300 m suivant les axes x et z. Concernant l'estimation des angles d'Euler, le $LSTM_{obus_{155}}$ ne parvient pas à capturer la dynamique de l'angle de roulis contrairement aux angles de tangage et de lacet.

D'après les résultats présentés dans les figures 4.30 - 4.33, un LSTM est partiellement adapté à l'estimation des positions et des vitesses d'un obus de 155 mm comparé aux

résultats obtenus pour le mortier de 120 mm. Ces observations s'expliquent par les importantes vitesses de rotation de l'obus de 155 mm qui ne disposent pas d'ailettes. Cela implique de fortes dynamiques angulaires qui influent sur la précision des estimations du LSTM. Ainsi, dans le cas d'un projectile à forte vitesse de rotation, des réseaux spécialisés dans l'estimation des positions et des vitesses exclusivement sont préférables.
De plus, les figures 4.30 - 4.33 confirment les constatations formulées précédemment sur le mortier de 120 mm ; un LSTM qui estime la trajectoire complète d'un projectile ne parvient pas à estimer convenablement l'angle de roulis de la munition. Ces observations sont d'autant plus vraies lorsque la vitesse de rotation du projectile est importante. Dans le cas de l'obus de 155 mm, le LSTM ne capture aucune dynamique de l'angle de roulis. Toutefois, cette solution est adaptée aux angles de tangage et de lacet de la munition.

Pour conclure sur l'estimation de la trajectoire balistique d'un projectile, un LSTM est parfaitement adapté à l'estimation des positions des vitesses. En effet, au vu des résultats obtenus pour le mortier de 120 mm ainsi que ceux obtenus pour une autre munition non présentée dans ce document, un LSTM est parfaitement adapté à l'estimation des positions et des vitesses d'une munition caractérisée par une trajectoire balistique avec un faible taux de roulis. Toutefois, d'après les résultats obtenus pour l'obus de 155 mm, les performances d'estimation d'un LSTM sont dégradées dans le cas d'un projectile caractérisé par une trajectoire balistique avec une forte vitesse de rotation. Néanmoins, pour l'ensemble des projectiles étudiés jusqu'à présent, un LSTM ne parvient pas à capturer les dynamiques angulaires, même dans le cas de projectiles à faibles vitesses de rotation.

Problématique d'estimation des angles d'Euler

D'après les résultats d'estimation des trajectoires du mortier de 120 mm et de l'obus de 155 mm, les LSTM ne sont pas adaptés à l'estimation des angles d'Euler des munitions, particulièrement à l'angle de roulis. Afin de fournir des éléments de réponse à ces observations, cette section se focalise exclusivement sur l'estimation des angles d'Euler.

D'après la littérature, les LSTMs ne sont pas adaptés à des fonctions périodiques comme présentées dans [128], [129]. Afin de résoudre cette problématique, d'autres structures ont été étudiées telles qu'un LSTM encodeur/décodeur ou un BiLSTM. Mais ces deux solutions n'ont pas présenté de résultats convaincants. De plus, un LSTM a été entraîné pour estimer le quaternion orientation plutôt que les angles d'Euler directement. Mais, comme précédemment, cette approche n'a présenté aucun résultat convaincant.

Une autre approche, inspirée de l'article [43] et illustrée dans la figure 1.14, est d'es-

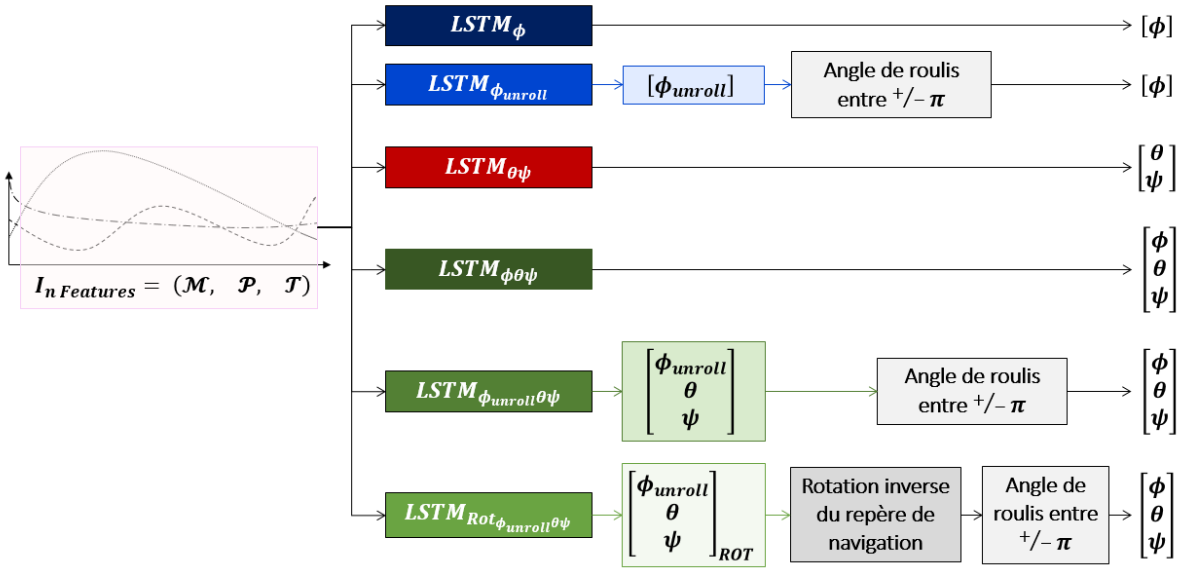

FIGURE 4.34 – Présentation de $LSTM_{\phi}$, $LSTM_{\phi_{unroll}}$, $LSTM_{\theta,\psi}$, $LSTM_{\phi,\theta,\psi}$, $LSTM_{\phi_{unroll},\theta,\psi}$, $LSTM_{ROT_{\phi_{unroll},\theta,\psi}}$.

timer indépendamment les grandeurs de l'orientation dans le but de discriminer les différentes dynamiques. Ainsi, sur le principe des réseaux spécialisés, 6 LSTM, présentés dans la figure 4.34, ont été entraînés spécifiquement dans les mêmes conditions d'après les données d'entrée $I_{n\ Features} = (\mathcal{M}, \mathcal{P}, \mathcal{T}) \in \mathbb{R}^{16}$ présentée à la partie *(4.3)* de sorte que :

- $LSTM_{\phi}$ estime l'angle de roulis ϕ de la munition.
- $LSTM_{\phi_{unroll}}$ estime l'angle de roulis déplié ϕ_{unroll}, puis cet angle est exprimé entre $\pm\pi$ comme la munition tourne sur elle-même.
- $LSTM_{\theta,\psi}$ estime l'angle de tangage θ et de lacet ψ de la munition.
- $LSTM_{\phi,\theta,\psi}$ estime les angles de roulis ϕ, tangage θ et lacet ψ de la munition.
- $LSTM_{\phi_{unroll},\theta,\psi}$ estime l'angle de roulis déplié ϕ_{unroll}, l'angle de tangage θ et de lacet ψ de la munition, puis l'angle de roulis est exprimé entre $\pm\pi$.
- $LSTM_{ROT_{\phi_{unroll},\theta,\psi}}$ estime l'angle de roulis déplié ϕ_{unroll}, l'angle de tangage θ et de lacet ψ de la munition dans le repère de navigation tourné $\boldsymbol{n}_{\gamma}$, puis l'angle de roulis est exprimé entre $\pm\pi$ et la rotation inverse est appliquée afin d'exprimer les trois angles dans le repère de navigation local $\boldsymbol{n}$.

Les figures 4.35 - 4.37 présentent les résultats d'estimation de $LSTM_{\phi}$, $LSTM_{\phi_{unroll}}$, $LSTM_{\theta,\psi}$, $LSTM_{\phi,\theta,\psi}$, $LSTM_{\phi_{unroll},\theta,\psi}$, $LSTM_{ROT_{\phi_{unroll},\theta,\psi}}$ pour l'estimation des trois angles d'Euler d'un mortier de 120 mm.

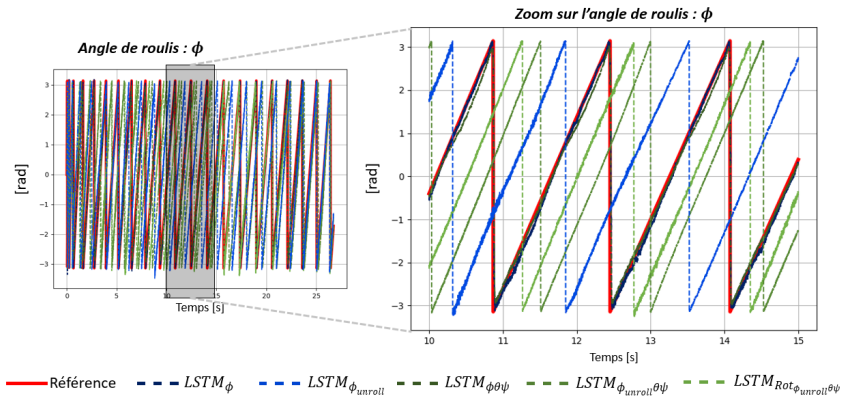

FIGURE 4.35 – Estimation de l'angle de roulis par $LSTM_{\phi}$, $LSTM_{\phi_{unroll}}$, $LSTM_{\phi,\theta,\psi}$, $LSTM_{\phi_{unroll},\theta,\psi}$ et $LSTM_{ROT_{\phi_{unroll},\theta,\psi}}$.

La figure 4.35 montre l'estimation de l'angle de roulis d'un mortier de 120 mm par $LSTM_{\phi}$, $LSTM_{\phi_{unroll}}$, $LSTM_{\phi,\theta,\psi}$, $LSTM_{\phi_{unroll},\theta,\psi}$ et par $LSTM_{ROT_{\phi_{unroll},\theta,\psi}}$. L'angle de roulis est estimé correctement par $LSTM_{\phi}$ et par $LSTM_{\phi,\theta,\psi}$. Ces réseaux parviennent à estimer la dynamique de roulis moyennant un nombre important de couches. À l'inverse, et contrairement aux résultats attendus, le dépliage de l'angle de roulis et la rotation du repère de navigation détériorent l'estimation de cet angle. En effet, $LSTM_{\phi_{unroll}}$, $LSTM_{\phi_{unroll},\theta,\psi}$ et $LSTM_{ROT_{\phi_{unroll},\theta,\psi}}$ présentent un retard.

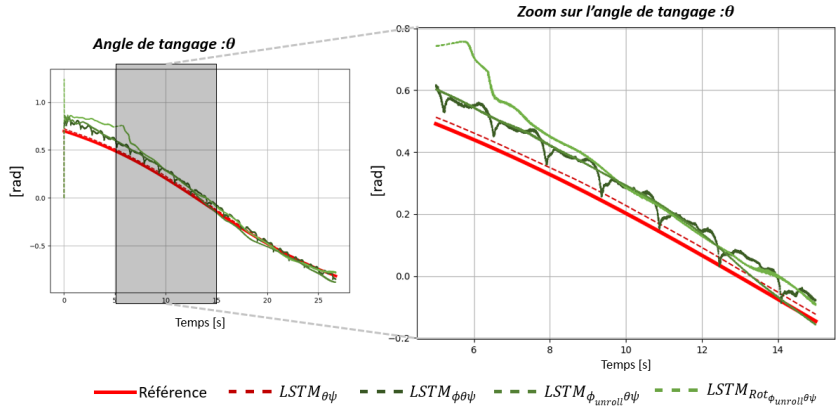

FIGURE 4.36 – Estimation de l'angle de tangage par $LSTM_{\theta,\psi}$, $LSTM_{\phi,\theta,\psi}$, $LSTM_{\phi_{unroll},\theta,\psi}$ et $LSTM_{ROT_{\phi_{unroll},\theta,\psi}}$.

Les figures 4.36 et 4.37 présentent respectivement l'estimation de l'angle de tangage et de lacet d'un mortier de 120 mm par $LSTM_{\theta,\psi}$, $LSTM_{\phi,\theta,\psi}$, $LSTM_{\phi_{unroll},\theta,\psi}$, $LSTM_{ROT_{\phi_{unroll},\theta,\psi}}$. Comme pour l'angle de roulis, la solution qui semble la plus adaptée

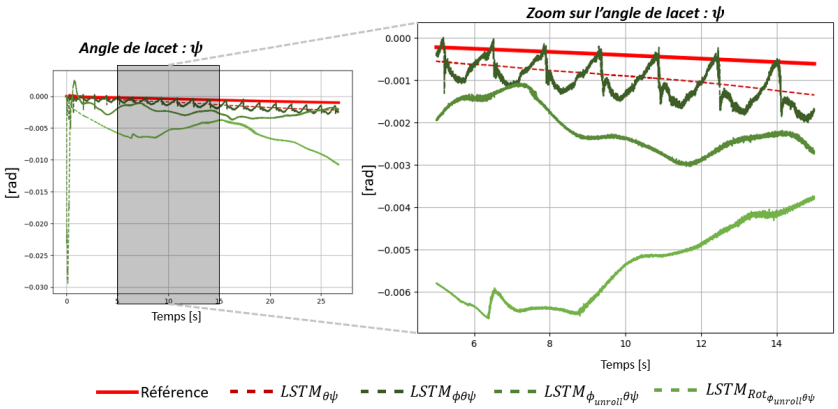

FIGURE 4.37 – Estimation de l'angle de lacet par $LSTM_{\theta,\psi}$, $LSTM_{\phi,\theta,\psi}$, $LSTM_{\phi_{unroll},\theta,\psi}$ et $LSTM_{ROT_{\phi_{unroll}},\theta,\psi}$.

est l'estimation de $LSTM_{\theta,\psi}$, sans dépliage de l'angle de roulis et sans rotation du repère de navigation. De plus, $LSTM_{\phi,\theta,\psi}$ présente de moins bonnes performances que $LSTM_{\theta,\psi}$ donc un réseau spécialisé dans l'estimation de tangage et de lacet semble préférable.

Afin d'analyser les performances globales des réseaux, la figure 4.38 présente le critère d'erreur $\mathcal{C}_{score}$ (4.19) évalué pour 50 trajectoires de mortier de 120 mm.

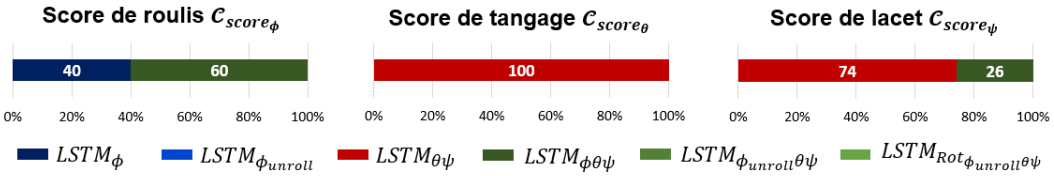

FIGURE 4.38 – Critère d'erreur $\mathcal{C}_{score}$ (4.19) de $LSTM_{\phi}$, de $LSTM_{\phi_{unroll}}$, de $LSTM_{\theta,\psi}$, de $LSTM_{\phi,\theta,\psi}$, de $LSTM_{\phi_{unroll},\theta,\psi}$ et de $LSTM_{ROT_{\phi_{unroll}},\theta,\psi}$.

La figure 4.38 confirme les observations préliminaires. Des réseaux spécialisés sont nécessaires pour estimer convenablement les angles d'Euler d'un mortier de 120 mm. En effet, $LSTM_{\phi,\theta,\psi}$ présente les meilleurs résultats d'estimation de l'angle de roulis, et $LSTM_{\theta,\psi}$ présente les meilleures performances d'estimation des angles de tangage et de lacet. Ainsi, un réseau spécialisé parvient à capturer les trois dynamiques angulaires.

D'après les analyses formulées précédemment, la solution la plus appropriée à l'estimation de l'angle de roulis du mortier de 120 mm est le réseau $LSTM_{\phi,\theta,\psi}$. Pour cela, ce même réseau est entraîné pour estimer les trois angles d'Euler d'un obus de 155 mm. Les résultats d'estimation pour une trajectoire sont présentés dans la figure 4.39.

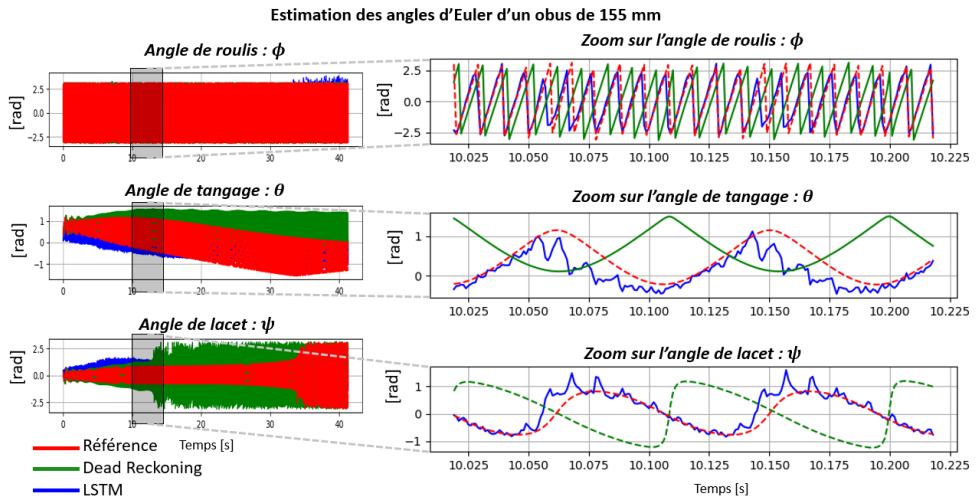

FIGURE 4.39 – Estimation des angles d'Euler d'un obus de 155 mm par le *Dead Reckoning* (en vert), le $LSTM_{\phi,\theta,\psi}$ (en bleu) et les angles de référence (en rouge).

La figure 4.39 montre que le LSTM parvient à apprendre la dynamique d'orientation de l'obus de 155 mm. En effet, contrairement aux résultats présentés dans la figure 4.31, le $LSTM_{\phi,\theta,\psi}$ parvient à estimer convenablement les trois angles d'Euler de l'obus. L'analyse sur l'ensemble du jeu de données de test montre que le $LSTM_{\phi,\theta,\psi}$ surpasse le *Dead Reckoning* pour l'intégralité des simulations testées et pour les trois angles d'Euler.

Pour conclure la problématique de l'estimation des angles d'Euler, pour des projectiles stabilisés ou non par ailettes, un LSTM spécialisé dans l'estimation des angles d'Euler est nécessaire. Malgré des résultats cohérents, cette solution nécessite donc l'emploie de plusieurs réseaux de neurones pour estimer la trajectoire complète des projectiles.

Influence de la taille de la séquence d'entrée

Les figures 4.30 - 4.33 montrent que le LSTM parvient à capturer globalement la dynamique d'évolution de l'obus de 155 mm. Un paramètre influant ces résultats est la taille de la séquence d'entrée du LSTM. Pour cela, afin de visualiser l'influence de la taille de la séquence d'entrée, plusieurs LSTM dont les caractéristiques sont présentées dans la table 4.4 ont été entraînés. Ainsi, $LSTM_5$ est entraîné avec une séquence d'entrée de 5 pas de temps, $LSTM_{10}$ avec 10 pas de temps, et $LSTM_{50}$ est entraîné avec 50 pas de temps afin d'estimer les positions, les vitesses et les angles d'Euler d'un obus de 155 mm.

Les figures 4.40 - 4.42 présentent les positions, les vitesses et les angles d'Euler d'un obus de 155 mm estimés par le $LSTM_5$, le $LSTM_{10}$ et le $LSTM_{50}$.

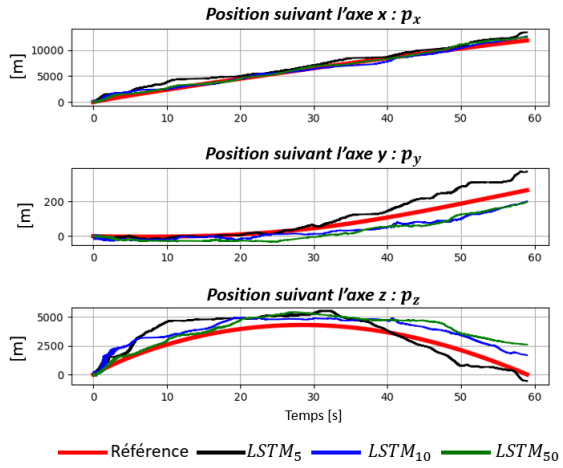

FIGURE 4.40 – Estimation de la position d'un obus de 155 mm par $LSTM_5$ (noir), $LSTM_{10}$ (bleu) et $LSTM_{50}$ (vert).

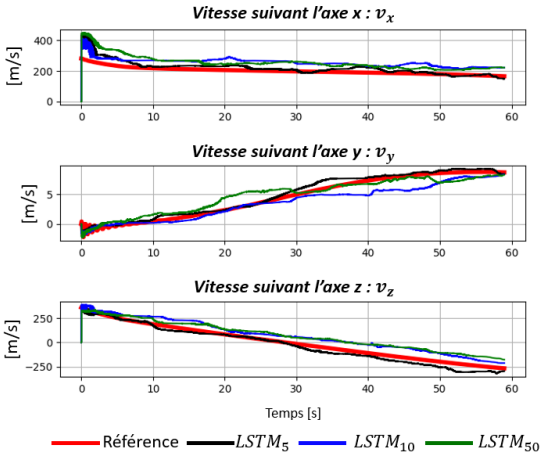

FIGURE 4.41 – Estimation de la vitesse d'un obus de 155 mm par $LSTM_5$ (noir), $LSTM_{10}$ (bleu) et $LSTM_{50}$ (vert).

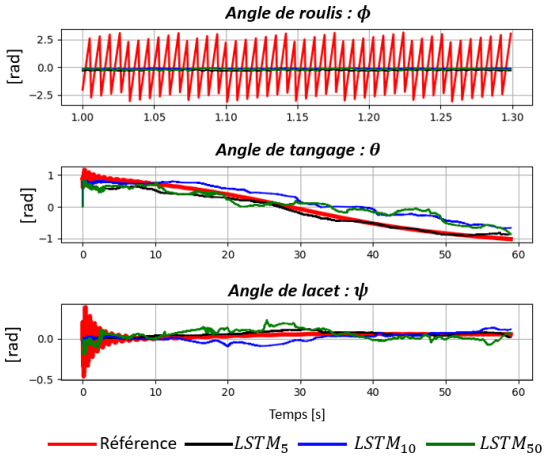

FIGURE 4.42 – Estimation des angles d'Euler d'un obus de 155 mm par $LSTM_5$ (noir), $LSTM_{10}$ (bleu) et $LSTM_{50}$ (vert).

Les figures 4.40 - 4.42 montrent que les réseaux $LSTM_5$, $LSTM_{10}$ et $LSTM_{50}$ parviennent à capturer globalement les dynamiques des positions et des vitesses. Comme observé précédemment, les réseaux ne capturent pas la dynamique de l'angle de roulis mais parviennent à estimer de façon cohérente les angles de tangage et de lacet.

La figure 4.43 présente la somme des erreurs quadratiques moyennes, notée $\mathcal{C}_{RMSE}$, évaluée pour $LSTM_5$ (noir), $LSTM_{10}$ (bleu) et $LSTM_{50}$ (vert) sur l'ensemble de test.

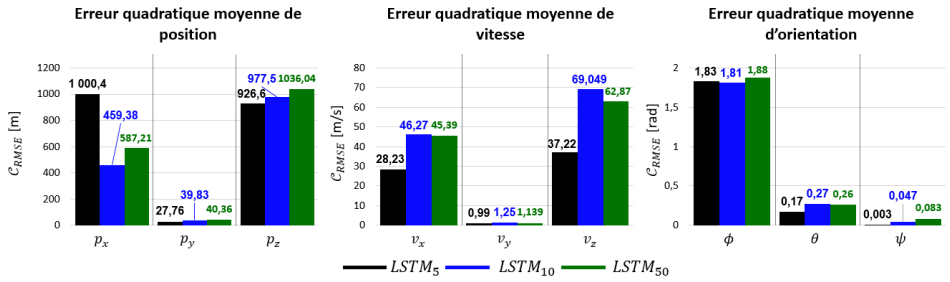

FIGURE 4.43 – Erreur quadratique moyenne $\mathcal{C}_{RMSE}$ évaluée pour $LSTM_5$ (noir), $LSTM_{10}$ (bleu) et $LSTM_{50}$(vert).

La figure 4.43 montre que la longueur de la séquence d'entrée influe sur la précision des estimations. En effet, les séquences les plus courtes présentent en moyenne les plus faibles erreurs d'estimation. Ainsi, pour l'estimation des positions d'un obus de 155 mm, une séquence de 10 pas de temps est préférable contrairement aux vitesses où une séquence de 5 pas de temps semble plus appropriée. La longueur de la séquence est un paramètre à définir lors de l'entraînement d'un LSTM et est réglée de façon empirique suivant la nature et la forme des données d'entrée. Concernant l'obus de 155 mm, malgré plusieurs tests préliminaires, la taille de la séquence, comme le nombre de couches, n'ont pas permis d'estimer précisément les positions et les vitesses de cette munition, du fait de la forte vitesse de rotation de l'obus.

4.5.2 Estimation de la trajectoire d'un *Basic Finner* et d'un projectile de 40 mm

Cette partie présente les résultats d'estimation de la trajectoire d'un projectile de 40 mm et du *Basic Finner*, deux projectiles caractérisés par des tirs tendus.

Estimation de la trajectoire d'un *Basic Finner* par un LSTM

Le $LSTM_{BF}$ estime la position, la vitesse et les angles d'Euler d'un *Basic Finner*, caractérisé par un tir tendu et une vitesse de rotation faible, similaire à celui du mortier de 120 mm. Les données d'entrée de $LSTM_{BF}$ sont $I_{n\ Features} = (\mathcal{M}, \mathcal{P}, \mathcal{T}) \in \mathbb{R}^{16}$ avec $\mathcal{M} \in \mathbb{R}^{12}$ les données inertielles comprenant les mesures IMU et le champ magnétique de référence, avec $\mathcal{P} \in \mathbb{R}^3$ les paramètres de vol qui sont l'angle de braquage des ailettes, la vitesse initiale et l'angle d'élévation du canon et avec $\mathcal{T} \in \mathbb{R}^1$ le vecteur temps. Aucune

normalisation et aucune rotation du repère de navigation n'est prise en compte pour l'entraînement de $LSTM_{BF}$, dont les caractéristiques sont présentées dans le tableau 4.5.

TABLE 4.5 – Caractéristiques d'entraînement de $LSTM_{BF}$ sur des trajectoires de *Basic Finner*.

Jeu de données	Jeu de données d'entraînement : Jeu de données de validation : Jeu de données de test :	1 000 simulations 100 simulations 100 simulations
Caractéristiques des données d'entrée	$B_{atch\ size}$: $S_{eq\ len}$:	64 30 pas de temps
Caractéristiques d'entraînement	Fonction de coût : Algorithme d'optimisation : Taux d'apprentissage :	*Mean Squared Error (MSE)* Adam $1e^{-4}$

Les figures 4.44 - 4.46 présentent les résultats d'estimation de la position, de la vitesse et des angles d'Euler d'une trajectoire d'un *Basic Finner* ainsi que les erreurs associées.

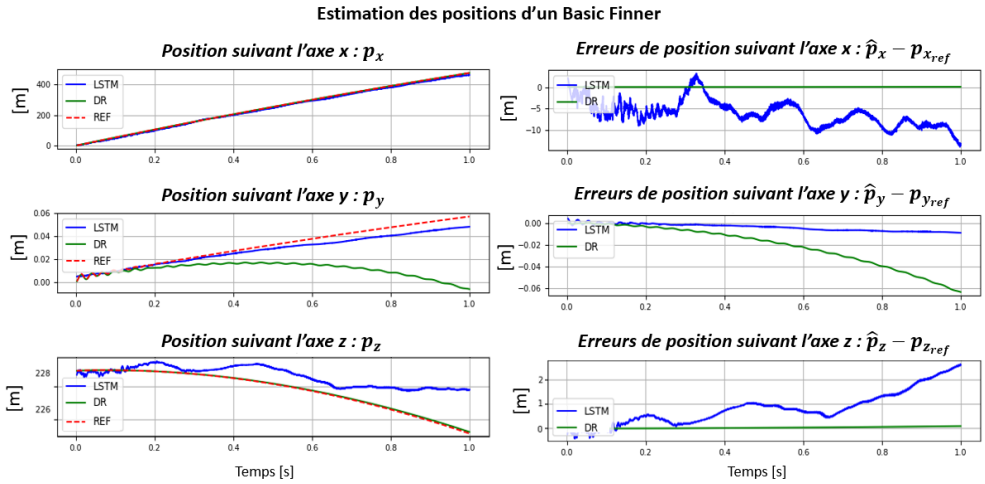

FIGURE 4.44 – *Basic Finner* : Position estimée et erreurs associées [m] obtenues par l'algorithme de *Dead Reckoning* (vert), $LSTM_{BF}$ (bleu) et la position de référence (rouge).

Les figures 4.44 - 4.46 montrent que, malgré les faibles dynamiques, le $LSTM_{BF}$ est largement moins performant qu'un *Dead Reckoning* pour l'estimation des positions, des vitesses et des angles d'Euler d'un *Basic Finner*, à l'exception de la position suivant l'axe y. Toutefois, le $LSTM_{BF}$ parvient à capturer les dynamiques des positions, des vitesses et des angles de tangage et de lacet.

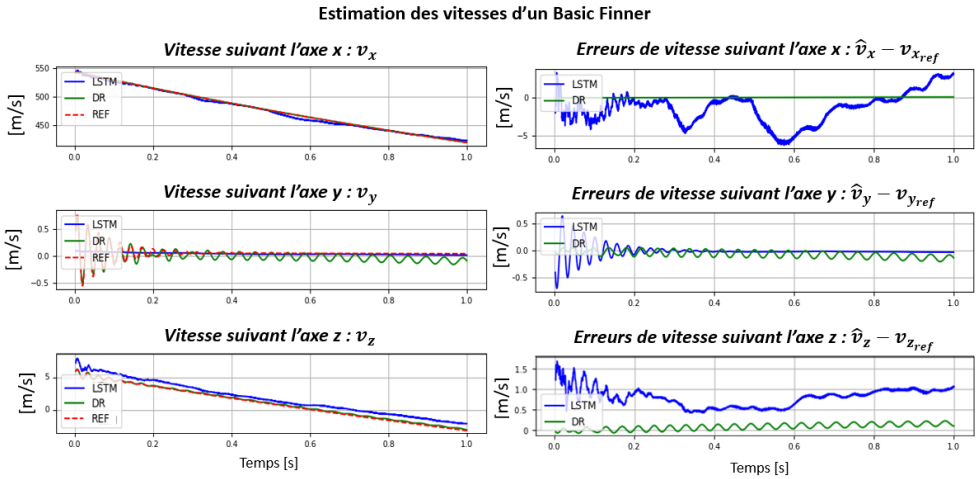

FIGURE 4.45 – *Basic Finner* : Vitesse estimée et erreurs associées [m/s] obtenues par l'algorithme de *Dead Reckoning* (vert), $LSTM_{BF}$ (bleu) et la vitesse de référence (rouge).

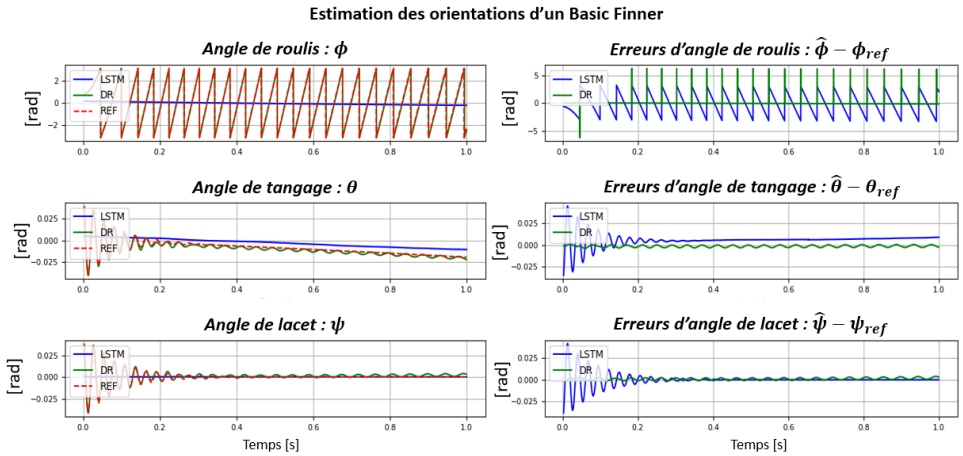

FIGURE 4.46 – *Basic Finner* : Orientation estimée et erreurs associées [rad] obtenues par l'algorithme de *Dead Reckoning* (vert), $LSTM_{BF}$ (bleu) et orientation de référence (rouge).

Les performances du $LSTM_{BF}$ et du *Dead Reckoning* sont évaluées sur l'ensemble du jeu de données de test avec les critères $\mathcal{C}_{score}$ (4.19) et $\mathcal{C}_{RMSE}$ définis tels que :

$$\mathcal{C}_{RMSE} = \frac{1}{N_{sim}} \sum_{k=1}^{N_{sim}} RMSE(sim_k) \tag{4.21}$$

avec N_{sim} le nombre de simulations du jeu de données de test et avec $RMSE(sim_k)$ l'erreur quadratique moyenne de la grandeur considérée, évaluée pour la simulation k.

Les figures 4.47 - 4.48 présentent les critères $\mathcal{C}_{RMSE}$ (4.21) et $\mathcal{C}_{score}$ (4.19) du $LSTM_{BF}$ et du *Dead Reckoning* pour l'estimation de la trajectoire d'un *Basic Finner*.

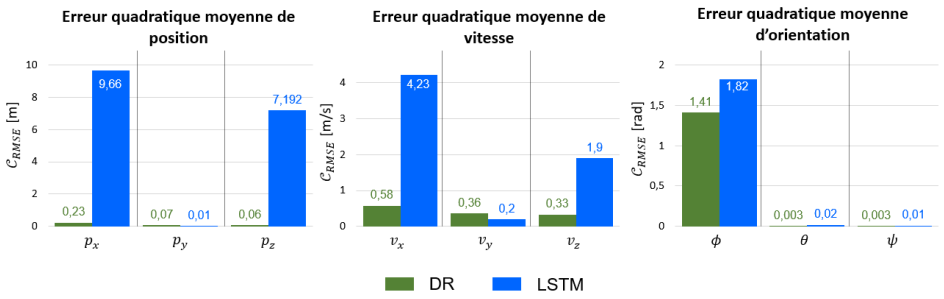

FIGURE 4.47 – *Basic Finner* : $\mathcal{C}_{RMSE}$ (4.21) du $LSTM_{BF}$ et du *Dead Reckoning*.

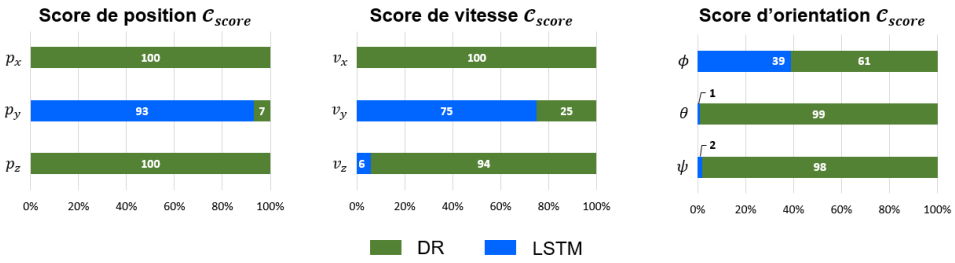

FIGURE 4.48 – *Basic Finner* : $\mathcal{C}_{score}$ (4.19) du $LSTM_{BF}$ et du *Dead Reckoning*.

Les figures 4.47 - 4.48 montrent que le $LSTM_{BF}$ n'est pas du tout performant pour estimer la trajectoire d'un *Basic Finner*. En effet, les performances d'estimation du LSTM sont moins bonnes que celles d'un *Dead Reckoning*. Malgré plusieurs tests préliminaires tels que la modification de plusieurs hyperparamètres ou la modification des couches, aucun réseau n'a surpassé le *Dead Reckoning* pour l'estimation de la trajectoire d'un *Basic Finner*.

Estimation de la trajectoire d'un projectile de 40 mm par un LSTM

Le $LSTM_{40\ mm}$ estime la position, la vitesse et les angles d'Euler d'un projectile de 40 mm, caractérisé par un tir tendu et une vitesse de rotation élevée, similaire à celle de l'obus de 155 mm. Les données d'entrée du $LSTM_{40\ mm}$ sont $I_{n\ Features} = (\mathcal{M}, \mathcal{P}, \mathcal{T}) \in \mathbb{R}^{16}$ avec $\mathcal{M} \in \mathbb{R}^{12}$ les données inertielles, $\mathcal{P} \in \mathbb{R}^3$ les paramètres de vol qui sont l'angle de roulis initial, la vitesse initiale et l'angle d'élévation du canon et avec $\mathcal{T} \in \mathbb{R}^1$ le vecteur temps. Les caractéristiques de $LSTM_{40\ mm}$ sont présentées dans le tableau 4.6.

TABLE 4.6 – Caractéristiques d'entraînement de $LSTM_{40\ mm}$ sur des trajectoires projectiles de 40 mm.

Jeu de données	Jeu de données d'entraînement : Jeu de données de validation : Jeu de données de test :	1 000 simulations 100 simulations 100 simulations
Caractéristiques des données d'entrée	$B_{atch\ size}$: $S_{eq\ len}$:	64 10 pas de temps
Caractéristiques d'entraînement	Fonction de coût : Algorithme d'optimisation : Taux d'apprentissage :	*Mean Squared Error (MSE)* Adam $1e^{-4}$

Les figures 4.49 - 4.50 présentent les critères $\mathcal{C}_{RMSE}$ (4.21) et $\mathcal{C}_{score}$ (4.19) du $LSTM_{40\ mm}$ et du *Dead Reckoning* pour l'estimation de la trajectoire d'un projectile de 40 mm.

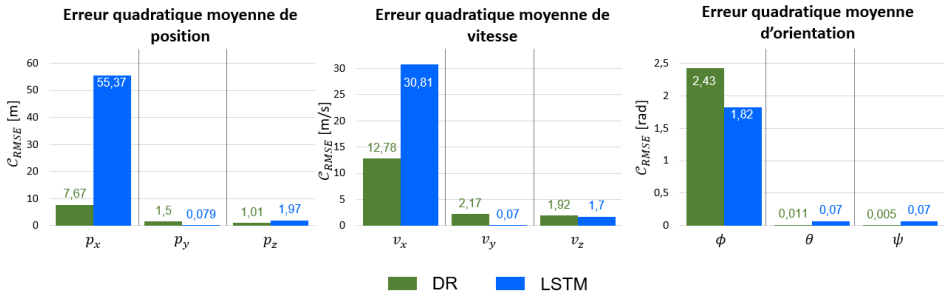

FIGURE 4.49 – Projectile de 40 mm : $\mathcal{C}_{RMSE}$ (4.21) du $LSTM_{40\ mm}$ et du *Dead Reckoning*.

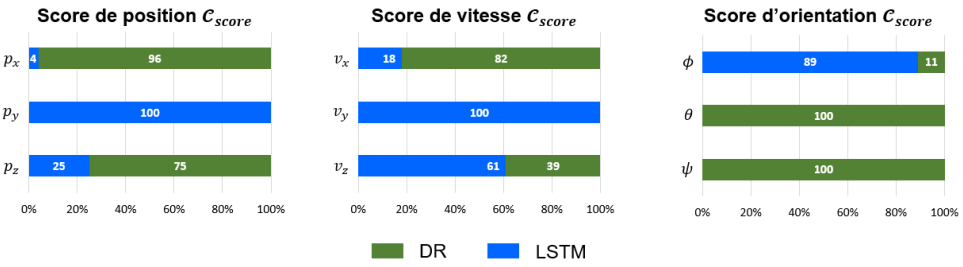

FIGURE 4.50 – Projectile de 40 mm : $\mathcal{C}_{score}$ (4.19) du $LSTM_{40\ mm}$ et du *Dead Reckoning*.

Comme pour le *Basic Finner*, les figures 4.49 - 4.50 montrent que le $LSTM_{40\ mm}$ ne surpasse pas l'algorithme de *Dead Reckoning* pour l'estimation de la trajectoire d'un projectile de 40 mm. Comme précédemment, malgré différents entraînements, le $LSTM_{40\ mm}$ ne surpasse pas les estimations d'un *Dead Reckoning*.

Explication du problème d'estimation de la trajectoire d'un *Basic Finner* et d'un projectile de 40mm par un LSTM

Les figures 4.44 - 4.49 montrent qu'un LSTM ne parvient pas à estimer convenablement les trajectoires d'un *Basic Finner* et d'un projectile de 40 mm caractérisés par des tirs tendus. L'hypothèse formulée est qu'un LSTM ne parvient pas à estimer la trajectoire d'un projectile caractérisé par un tir tendu, du fait de la faible durée de vol et des faibles amplitudes de variation associées. Afin de vérifier cette hypothèse, $LSTM_{120\ mm}$ est entraîné pour estimer la trajectoire tronquée d'un mortier de 120 mm sachant que ce réseau performe pour l'estimation de la trajectoire complète de ce projectile. $LSTM_{120\ mm}$ est entraîné d'après les spécifications de $LSTM_{ALL}$ présenté dans la partie (*4.3.1*) afin d'estimer la trajectoire tronquée d'un mortier de 120 mm, correspondant à un tir tendu.

Les figures 4.51 - 4.53 présentent l'estimation de la position, de la vitesse et des angles d'Euler d'une trajectoire d'un mortier de 120 mm en tir tendu ainsi que les erreurs associées.

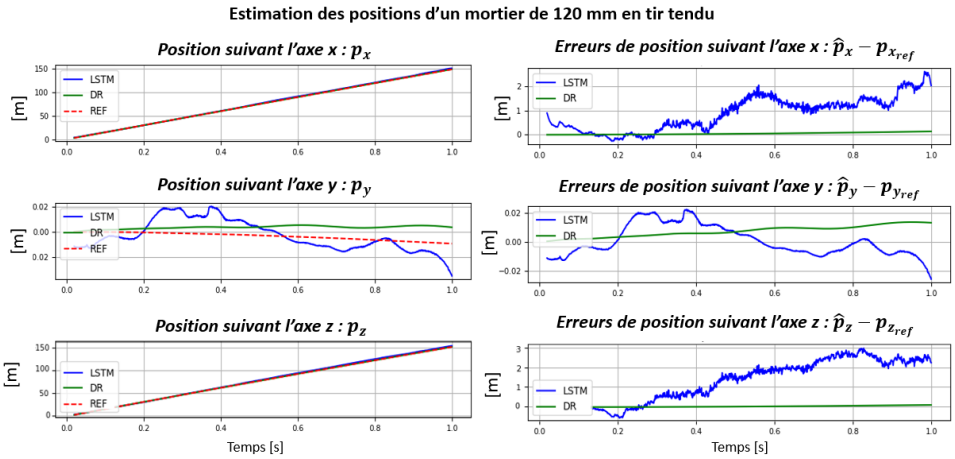

FIGURE 4.51 – Projectile de 120 mm en tir tendu : Position estimée et erreurs associées [m] obtenues par l'algorithme de *Dead Reckoning* (vert), $LSTM_{120\ mm}$ (bleu) et la position de référence (rouge).

Comme le montre les figures 4.51 - 4.53, le $LSTM_{120\ mm}$ ne parvient pas à estimer la trajectoire d'un mortier de 120 mm en tir tendu, alors que ce même réseau estimait avec précision la trajectoire complète du mortier de 120 mm. Cette constatation est particulièrement vérifiée pour l'estimation des vitesses et des angles de tangage et de lacet du projectile. En particulier, une erreur systématique, observée également pour le *Basic Finner* et le projectile de 40 mm, est présente pour l'estimation des vitesses.

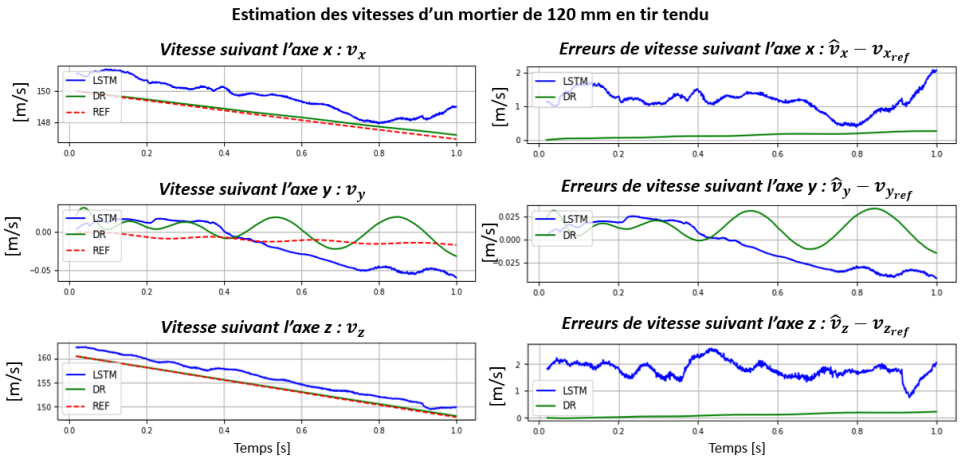

FIGURE 4.52 – Projectile de 120 mm en tir tendu : Vitesse estimée et erreurs associées [m/s] obtenues par l'algorithme de *Dead Reckoning* (vert), $LSTM_{120\ mm}$ (bleu) et la vitesse de référence (rouge).

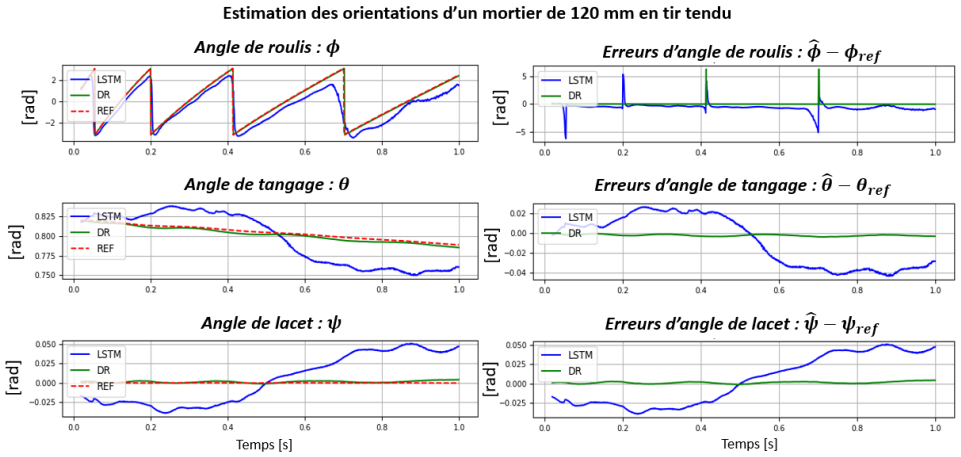

FIGURE 4.53 – Projectile de 120 mm en tir tendu : Orientation estimée et erreurs associées [rad] obtenues par l'algorithme de *Dead Reckoning* (vert), $LSTM_{120\ mm}$ (bleu) et l'orientation de référence (rouge).

Afin de valider l'hypothèse qu'un LSTM n'est pas adapté à l'estimation de la trajectoire d'un projectile caractérisé par un tir tendu, les figures 4.54 - 4.55 présentent respectivement les critères d'évaluation $\mathcal{C}_{RMSE}$ (4.21) et $\mathcal{C}_{score}$ (4.19) obtenus par le $LSTM_{120\ mm}$ et le *Dead Reckoning* pour l'estimation des positions, des vitesses et des angles d'Euler d'un mortier de 120 mm dont la trajectoire est tronquée pour correspondre à un tir tendu.

Les figures 4.54 et 4.55 montrent que le *Dead Reckoning* est largement plus précis que $LSTM_{120\ mm}$ pour estimer les positions, les vitesses et les angles d'Euler d'un mortier de

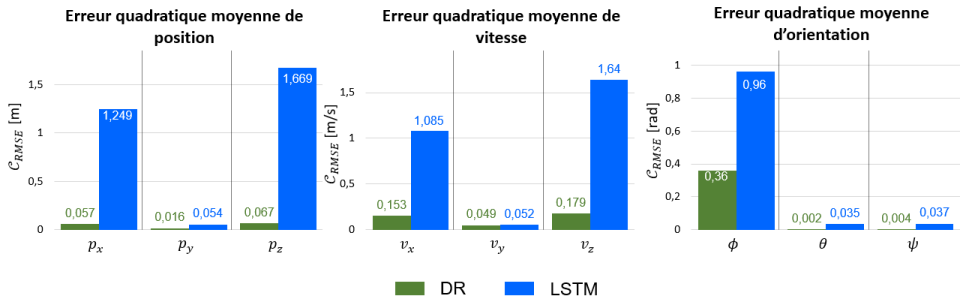

FIGURE 4.54 – Mortier de 120 mm en tir tendu : erreur quadratique moyenne globale $\mathcal{C}_{RMSE}$ (4.21).

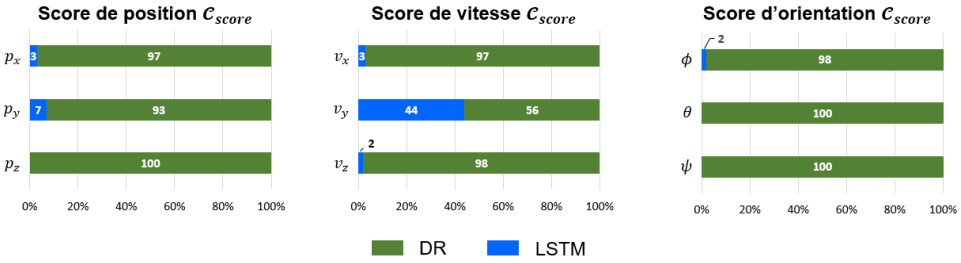

FIGURE 4.55 – Mortier de 120 mm en tir tendu : score $\mathcal{C}_{score}$ (4.19).

120 mm dont la trajectoire est tronquée pour correspondre à un tir tendu. Ces résultats sont en contradiction avec ceux obtenus pour l'estimation de la trajectoire complète d'un mortier de 120 mm où le LSTM présentait de très faibles erreurs d'estimation comparées aux erreurs d'un *Dead Reckoning*. Ainsi, ces observations confirment l'hypothèse formulée. Le $LSTM_{120\ mm}$ ne parvient pas à estimer la trajectoire d'un mortier de 120 mm en tir tendu, alors que ce même réseau est performant pour estimer la trajectoire complète du mortier de 120 mm. Ces observations expliquent donc les mauvaises performances d'estimation obtenues sur les projectiles de 40 mm et le *Basic Finner*. Ainsi, un LSTM n'est pas adapté à l'estimation de la trajectoire d'un projectile caractérisé par un tir tendu.

4.6 Conclusion

Ce chapitre présente une méthode de navigation des projectiles basée exclusivement sur l'IA. Des LSTM *(Long Short-Term Memory)* estiment les trajectoires de plusieurs projectiles à partir des mesures IMU embarquées, des paramètres de lancement de la munition, et d'un vecteur temporel. Des réseaux spécialisés dans l'estimation d'une grandeur,

des méthodes de prétraitement des données d'entrée et l'influence du modèle d'erreur des capteurs inertiels sont analysés.

Cette solution est parfaitement adaptée à l'estimation des positions et des vitesses d'un mortier de 120 mm mais n'est pas optimale pour estimer l'orientation de cette munition. Pour cela, un réseau spécialisé dans l'estimation des angles d'Euler est préférable. D'une part, la normalisation des données d'entrée du LSTM ne permet pas d'optimiser les estimations contrairement à la rotation du repère de navigation. D'autre part, les résultats montrent qu'un LSTM estime précisément la trajectoire d'un mortier de 120 mm sans être impacté par le modèle d'erreur des capteurs inertiels. Ainsi, un LSTM est parfaitement adapté à l'estimation de la trajectoire d'un mortier de 120 mm à partir des mesures inertielles embarquées, des paramètres de lancement et d'un vecteur temporel.

L'estimation de la trajectoire d'un obus de 155 mm par un LSTM présente de moins bonnes performances que celles obtenues pour le mortier de 120 mm. Toutefois, cette solution reste adaptée à l'estimation des positions et des vitesses de cette munition. De plus, comme pour le mortier, les angles d'Euler de l'obus de 155 mm nécessitent d'être estimés indépendamment des autres grandeurs. Les résultats obtenus pour le mortier de 120 mm, l'obus de 155 mm et une autre munition qui ne tourne pas avec une trajectoire similaire à celle du mortier, permettent de conclure qu'un LSTM est parfaitement adapté à l'estimation des positions et des vitesses des projectiles caractérisés par une trajectoire balistique. De plus, cette solution est optimale lorsque la munition ne tourne que faiblement.

Les LSTM entraînés pour estimer les trajectoires d'un *Basic Finner* et d'un projectile de 40 mm prouvent que cette solution n'est pas applicable pour des projectiles caractérisés par des tirs tendus. Cette conclusion est validée en entraînant un LSTM sur des trajectoires tronquées d'un mortier de 120 mm. Ce même réseau est parfaitement adapté pour estimer la trajectoire complète de ce projectile alors qu'il ne permet pas d'estimer la trajectoire du mortier en tir tendu.

Au vu des résultats obtenus dans ce chapitre, les LSTM sont parfaitement adaptés à l'estimation des positions et des vitesses de projectiles caractérisés par des trajectoires balistiques et avec de faibles vitesses de rotation comme le mortier de 120 mm. Cette méthode peut ainsi être optimisée en l'intégrant dans des filtres de Kalman pour produire des solutions de navigation hybrides à faible coût, sans nécessiter de mesures GNSS tout en étant précises à long terme.

Chapitre 5

Optimisation d'un filtre de Kalman par des réseaux de neurones

Sommaire

5.1 Introduction

L'intelligence artificielle (IA) est de plus en plus intégrée dans les programmes militaires [52]-[55], [63], comme présenté dans la partie *(1.4.4)* de ce document. Toutefois, l'IA présente plusieurs limitations restreignant son intégration dans les programmes de navigation militaire [59], telle que la transparence et l'interprétabilité des modèles, qui limitent l'utilisation des réseaux de neurones pour des applications critiques, la vulnérabilité des modèles d'IA face à de nouvelles données inconnues, notamment dans le cas de la reconnaissance de formes où la modification d'un pixel peut conduire à une réponse contradictoire, ou encore la conception d'un jeu de données représentatif du problème à traiter, qui nécessite l'acquisition et le traitement de données fidèles et réalistes.

Dans le cadre de cette thèse, le jeu de données généré par BALCO [96] est considéré suffisamment représentatif des différentes trajectoires possibles des projectiles étudiés. De plus, malgré les précisions obtenues, la solution d'estimation présentée au chapitre précédent se base exclusivement sur l'intelligence artificielle et dépend complètement du modèle d'IA entraîné. Afin de limiter la dépendance des solutions proposées aux modèles d'IA, il est possible d'intégrer partiellement des réseaux de neurones dans des filtres de Kalman. Il s'agit alors d'un *Deep Kalman Filter*, bénéficiant ainsi des avantages des réseaux de neurones en termes de modélisation, tout en améliorant l'interprétabilité de la solution proposée.

Un *Deep Kalman Filter* est un filtre de Kalman classique dont l'un des modèles est déterminé par un réseau de neurones. Comme relaté dans [130], l'intégration d'IA dans un filtre de Kalman permet notamment de régler des paramètres du filtre comme dans [77]-[82], d'évaluer un modèle comme dans [83]-[86], ou de générer des pseudo-mesures comme dans [71]-[76]. Ainsi, ce chapitre se focalise sur plusieurs *Deep Kalman Filter* pour la navigation des projectiles. En d'autres termes, les objectifs de ce chapitre sont :

- → de valider la notion de *Deep Kalman Filter* pour la navigation des projectiles à travers un filtre de Kalman Imparfait Invariant Étendu (Imp.LIEKF) corrigé par des observations générées par un réseau de neurones.
- → d'étudier les performances d'estimation d'un *Deep Error State Kalman Filter* dans le cas où le modèle d'évolution est déterminé par un réseau de neurones.
- → d'évaluer comment l'IA permet d'adapter un modèle mathématique. Pour cela, un filtre de Kalman, valide dans le cas d'un tir tendu, est adapté à une trajectoire balistique en employant l'IA pour déterminer les modèles inconnus.

La première partie *(5.2)* présente un filtre de Kalman Imparfait Invariant Étendu corrigé par des observations générées par un réseau de neurones. La seconde partie *(5.3)* introduit la notion d'apprentissage par transfert nécessaire à l'entraînement d'un *Deep Kalman Filter*. La troisième partie *(5.4)* se concentre sur un *Deep Error State Kalman Filter* où le modèle d'évolution du filtre est déterminé par un réseau de neurones. Enfin, la quatrième partie *(5.5)* détaille un filtre de Kalman testé en vol, valide dans le cas d'un tir tendu et adapté à l'estimation d'une trajectoire balistique grâce aux réseaux de neurones.

5.2 Correction d'un filtre de Kalman par des pseudo-mesures générées par un LSTM

Cette partie présente un *Deep Kalman Filter* pour estimer la trajectoire d'un mortier de 120 mm où les observations du filtre sont générées par un LSTM. Ainsi, comme dans [71]-[76], un réseau de neurones est utilisé afin de générer des observations pour corriger les prédictions d'un filtre de Kalman. Cette approche permet ainsi de traiter et d'extraire les mesures des capteurs extérieurs afin de produire des observations fidèles lorsque ces dernières sont difficilement exploitables ; quand les mesures sont bruitées, biaisées, indisponibles ou nécessitant des prétraitements complexes tels que l'extraction de caractéristiques spatiales dans des images.

Dans l'exemple d'application présenté dans cette partie, un Filtre de Kalman Imparfait Invariant Étendu *(Imperfect Left Invariant Extended Kalman Filter - Imp.LIEKF)* prédit la trajectoire du projectile à partir des mesures des accéléromètres et des gyromètres embarqués, puis, la position et la vitesse estimée par un LSTM ainsi que le champ magnétique estimé à partir des magnétomètres corrigent ces prédications. En effet, d'après les résultats obtenus au chapitre précédent, un LSTM est parfaitement adapté à l'estimation des positions et des vitesses d'un mortier de 120 mm. De plus, comme aucune mesure GNSS n'est considérée dans ces travaux, cette méthode est adéquate pour générer des observations de positionnement absolu afin de corriger les prédictions du filtre de Kalman.

5.2.1 Imp.LIEKF pour l'estimation de la trajectoire d'un projectile

La figure 5.1 présente le *Deep Imp.LIEKF* implémenté pour estimer la trajectoire d'un mortier de 120 mm où les observations sont générées en partie par un modèle d'IA. L'objectif est donc d'utiliser un *Long Short-Term Memory (LSTM)* afin de générer des mesures de position et de vitesse de la munition, pouvant être considérées comme des mesures GNSS sans les inconvénients de ces signaux. De plus, les mesures des magnétomètres et du champ magnétique de référence sont exploitées afin de corriger les prédictions de ce filtre.

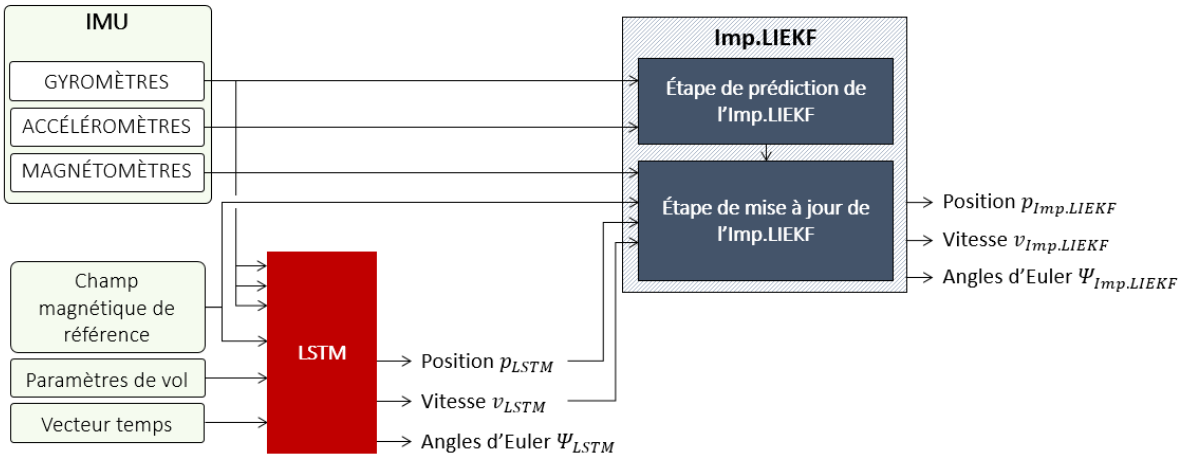

FIGURE 5.1 – Principe de fonctionnement du *Deep Imp.LIEKF* implémenté pour estimer la trajectoire d'un mortier de 120 mm à partir des mesures de l'IMU embarquée, du champ magnétique de référence, et des positions et des vitesses estimées par un LSTM.

Comme illustré dans la figure 5.1, lors de l'étape de prédiction de l'Imp.LIEKF, la trajectoire du projectile et la matrice de covariance de l'erreur sont prédites d'après des modèles mathématiques et les mesures des gyromètres et des accéléromètres embarqués dans la munition. Puis, d'une part, un LSTM pré-entraîné détermine la position et la vitesse du projectile à partir des mesures inertielles, du champ magnétique de référence et des paramètres de prévol. D'autre part, les mesures des magnétomètres sont utilisées afin d'estimer le champ magnétique de référence. Ensuite, ces trois observations sont utilisées lors de l'étape de mise à jour du filtre pour corriger la prédiction de la trajectoire et de la matrice de covariance de l'erreur.

États et mesures de l'Imp.LIEKF

Comme présenté dans la figure 5.1, à partir d'une configuration initiale donnée, des mesures des gyromètres ω_s, des accéléromètres a_s, des magnétomètres h_s et du champ magnétique de référence, le *Deep Imp.LIEKF* vise à estimer les états suivants :

$$x \triangleq \begin{pmatrix} R, & v, & p, & b_{\omega_s}, & b_{a_s}, & b_{h_s} \end{pmatrix} \tag{5.1}$$

avec $R \in SO(3)$, v et $p \in \mathbb{R}^{3\times1}$, l'orientation, la vitesse et la position du projectile dans le repère local de navigation $\boldsymbol{n}$, et avec b_{ω_s}, b_{a_s} et $b_{h_s} \in \mathbb{R}^{3\times1}$ les biais des capteurs inertiels.

Les mesures considérées pour mettre à jour le *Deep Imp.LIEKF* sont la position p_{LSTM} et la vitesse v_{LSTM} du projectile déterminées par un LSTM, ainsi que le champ magnétique de référence $h_n \in \mathbb{R}^{3\times1}$ dans le repère local de navigation $\boldsymbol{n}$. En d'autres termes, les mesures utilisées pour corriger l'Imp.LIEKF sont :

$$y_k = \begin{bmatrix} p_{LSTM}^T & v_{LSTM}^T & h_n^T \end{bmatrix}^T \in \mathbb{R}^{9\times1}. \tag{5.2}$$

Modèles et dynamiques

Les modèles de mesure des gyromètres, des accéléromètres et des magnétomètres sont :

$$\tilde{\omega}_s = \omega_s + b_{\omega_s} + W_{\omega_s}, \qquad \tilde{a}_s = a_s + b_{a_s} + W_{a_s}, \qquad \tilde{h}_s = h_s + b_{h_s} + W_{h_s}, \tag{5.3}$$

avec $\tilde{\omega}_s$, $\tilde{a}_s$ et $\tilde{h}_s \in \mathbb{R}^{3\times1}$ les modèles de mesure des gyromètres, des accéléromètres et des magnétomètres dans le repère capteur $\boldsymbol{s}$, avec ω_s, a_s et $h_s \in \mathbb{R}^{3\times1}$ les mesures vraies dans le repère capteur, b_{ω_s}, b_{a_s} et $b_{h_s} \in \mathbb{R}^{3\times1}$ les biais, et W_{ω_s}, W_{a_s} et $W_{h_s} \in \mathbb{R}^{3\times1}$ les bruits des mesures supposés blancs et gaussiens.
Le modèle d'évolution des biais des capteurs inertiels est un mouvement brownien modélisé par les équations suivantes,

$$\dot{b}_{\omega_s} = W_{b_{\omega_s}}, \qquad \dot{b}_{a_s} = W_{b_{a_s}}, \qquad \dot{b}_{h_s} = W_{b_{h_s}}, \tag{5.4}$$

avec $W_{b_{\omega_s}}$, $W_{b_{a_s}}$ et $W_{b_{h_s}} \in \mathbb{R}^{3\times1}$ des bruits gaussiens de moyenne nulle.

La dynamique d'évolution de l'orientation R, de la vitesse v et de la position p du projectile dans le repère local de navigation $\boldsymbol{n}$ est alors :

$$\dot{R} = R[\tilde{\omega}_s - b_{\omega_s} - W_{\omega_s}]_\times, \qquad \dot{v} = R(\tilde{a}_s - b_{a_s} - W_{a_s}) + g, \qquad \dot{p} = v, \tag{5.5}$$

avec $[.]_\times$ l'opérateur linéaire du groupe de Lie $SO(3)$ et g le vecteur gravité supposé constant pour la durée de vol du projectile.

D'après les dynamiques d'évolution (5.4) - (5.5), le bruit de modèle w_t associé aux états (5.1) est alors :

$$w_t = \begin{bmatrix} W_{\omega_s}^T & W_{a_s}^T & 0_{3\times1}^T & W_{b_{\omega_s}}^T & W_{b_{a_s}}^T & W_{b_{h_s}}^T \end{bmatrix}^T \in \mathbb{R}^{18\times1}. \tag{5.6}$$

5.2.2 Équations de l'Imp.LIEKF

Selon la méthodologie des Imp.IEKF [102], [104], l'orientation, la vitesse et la position du projectile sont représentées par une matrice du groupe de Lie matriciel $SE_2(3)$ telle que :

$$\chi_t = \begin{bmatrix} R & v & p \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix} \in SE_2(3). \tag{5.7}$$

Les biais b_{ω_s}, b_{a_s} et b_{h_s} des capteurs inertiels sont des éléments de $\mathbb{R}^{3\times1}$.

L'erreur non linéaire d'estimation de l'Imp.LIEKF est alors $e = (\eta_{\chi_t},\ \xi_{b_{\omega_s}},\ \xi_{b_{a_s}},\ \xi_{b_{h_s}})$ avec $\xi_{b_{\omega_s}} = \widehat{b}_{\omega_s} - b_{\omega_s}$, $\xi_{b_{a_s}} = \widehat{b}_{a_s} - b_{a_s}$, et $\xi_{b_{h_s}} = \widehat{b}_{h_s} - b_{h_s}$, les erreurs linéaires des biais des gyromètres, des accéléromètres et des magnétomètres, et avec $\eta_{\chi_t} \in SE_2(3)$ l'erreur invariante à gauche associée à χ_t (5.7) définie telle que :

$$\eta_{\chi_t} = \chi_t^{-1}\widehat{\chi}_t = \begin{bmatrix} R & v & p \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} \widehat{R} & \widehat{v} & \widehat{p} \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix} = \begin{bmatrix} R^T\widehat{R} & R^T\widehat{v} - R^Tv & R^T\widehat{p} - R^Tp \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix} \tag{5.8}$$

L'erreur invariante η_{χ_t} (5.8) est linéarisée telle que :

$$\eta_t \ = \ exp(\xi_t) = exp_m(\mathcal{L}_{\mathfrak{g}}(\xi_t)) \ \approx \ I_{5\times5} + \mathcal{L}_{\mathfrak{g}}(\xi_t) \ \approx \ \begin{bmatrix} I_{3\times3} + [\xi_R]_\times & \xi_v & \xi_p \\ 0_{1\times3} & 1 & 0 \\ 0_{1\times3} & 0 & 1 \end{bmatrix} \tag{5.9}$$

avec $\eta_R = R^T\widehat{R} \approx I_{3\times3} + [\xi_R]_\times$ l'erreur invariante d'orientation, $\eta_v = R^T\widehat{v} - R^Tv \approx \xi_v$ l'erreur invariante de vitesse et $\eta_p = R^T\widehat{p} - R^Tp \approx \xi_p$ l'erreur invariante de position.

Étape de prédiction de l'Imp.LIEKF

La prédiction de l'état x (5.1) est donnée par le système d'évolution (5.4) - (5.5) dans le cas où aucune perturbation n'est présente :

$$\frac{d}{dt}\begin{bmatrix}\widehat{\chi}_t\\ \widehat{b}_{\omega_s}\\ \widehat{b}_{a_s}\\ \widehat{b}_{h_s}\end{bmatrix} = \begin{bmatrix}f_{u_t}(\widehat{\chi}_t)\\ 0_{3\times1}\\ 0_{3\times1}\\ 0_{3\times1}\end{bmatrix}, \text{ avec } f_{u_t}(\widehat{\chi}_t) = \begin{bmatrix}\widehat{R}[\tilde{\omega}_s - \widehat{b}_{\omega_s}]_\times & \widehat{R}(\tilde{a}_s - \widehat{b}_{a_s}) + g & \widehat{v}\\ 0_{1\times3} & 0 & 0\\ 0_{1\times3} & 0 & 0\end{bmatrix} \tag{5.10}$$

La prédiction de la matrice de covariance P_t est donnée par l'équation suivante :

$$\frac{d}{dt}P_t = A_t P_t + P_t A_t^T + A_{d_\chi} Q_t A_{d_\chi}^T \tag{5.11}$$

avec Q_t la covariance du bruit de modèle w_t (5.6), et $A_t \in \mathbb{R}^{18\times18}$ et $A_{d_\chi} \in \mathbb{R}^{18\times18}$ les matrices jacobiennes déterminées par la dynamique d'erreur linéarisée telle que :

$$A_t = \begin{bmatrix}-[\tilde{\omega}_s]_\times & 0_{3\times3} & 0_{3\times3} & -I_{3\times3} & 0_{3\times3} & 0_{3\times3}\\ -[\tilde{a}_s]_\times & -[\tilde{\omega}_s]_\times & 0_{3\times3} & 0_{3\times3} & -I_{3\times3} & 0_{3\times3}\\ 0_{3\times3} & I_{3\times3} & -[\tilde{\omega}_s]_\times & 0_{3\times3} & 0_{3\times3} & 0_{3\times3}\\ & & 0_{9\times18} & & & \end{bmatrix}, A_{d_\chi} = \begin{bmatrix}I_{3\times3} & 0_{3\times3} & \\ 0_{3\times3} & I_{3\times3} & 0_{9\times12}\\ 0_{3\times3} & 0_{3\times3} & \\ & 0_{9\times9} & -I_{9\times9}\end{bmatrix}. \tag{5.12}$$

Étape de mise à jour de l'Imp.LIEKF

Les prédictions sont corrigées par des mesures y_k (5.2) qui sont la position p_{LSTM} et la vitesse v_{LSTM} du projectile déterminées par un LSTM et le champ magnétique de référence $h_n \in \mathbb{R}^{3\times1}$ dans le repère local de navigation $\boldsymbol{n}$. Le modèle de mesures est alors :

$$\widehat{y}_k = \begin{bmatrix}\widehat{p}\\ \widehat{v}\\ \widehat{h}_n\end{bmatrix} = \begin{bmatrix}\widehat{p}\\ \widehat{v}\\ \widehat{R}h_s\end{bmatrix} \in \mathbb{R}^{9\times1}. \tag{5.13}$$

Le gain de Kalman est alors évalué tel que :

$$K_k = P_{k|k-1}H_k^T(H_k P_{k|k-1} H_k^T + R_k)^{-1} \tag{5.14}$$

avec H_k la matrice d'observation et R_k la matrice de covariance du bruit de mesures,

identifiées d'après la linéarisation du modèle d'observation (5.13). La matrice de covariance du bruit de mesure et la matrice d'observation sont définies telles que :

$$R_k = cov\left(W_{p_{LSTM}}, W_{v_{LSTM}} W_{W_{h_s}}\right) \in \mathbb{R}^{9\times 9} \tag{5.15}$$

$$H_k = \begin{bmatrix} 0_{3\times3} & 0_{3\times3} & I_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & I_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ -[\tilde{h}_s]_\times & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & I_{3\times3} \end{bmatrix} \in \mathbb{R}^{9\times18} \quad . \tag{5.16}$$

avec $W_{p_{LSTM}}$, $W_{v_{LSTM}}$ et $W_{h_s} \in \mathbb{R}^{3\times1}$ des bruits blancs supposés gaussiens de l'estimation des positions et des vitesses par le LSTM et des magnétomètres.

D'après la méthodologie des Imp.LIEKF présentée dans le chapitre *2* de ce document, les prédictions sont mises à jour telles que :

$$\begin{bmatrix} \xi_{R_k}^T & \xi_{v_k}^T & \xi_{p_k}^T & \xi_{b_{\omega_s k}}^T & \xi_{b_{a_s k}}^T & \xi_{b_{h_s k}}^T \end{bmatrix}^T = K_k(y_k - \widehat{y}_k) \tag{5.17}$$

$$\widehat{R}_{k|k} = \widehat{R}_{k|k-1}\xi_{R_{k|k}}, \quad \widehat{v}_{k|k} = \widehat{R}_{k|k-1}\xi_{v_{k|k}} + \widehat{v}_{k|k-1}, \quad \widehat{p}_{k|k} = \widehat{R}_{k|k-1}\xi_{p_{k|k}} + \widehat{p}_{k|k-1}, \tag{5.18}$$

$$\widehat{b}_{\omega_{s_{k|k}}} = \widehat{b}_{\omega_{s_{k|k-1}}} + \xi_{b_{\omega_{s_k}}}, \quad \widehat{b}_{a_{s_{k|k}}} = \widehat{b}_{a_{s_{k|k}}} + \xi_{b_{a_{s_k}}}, \quad \widehat{b}_{h_{s_{k|k}}} = \widehat{b}_{h_{s_{k|k}}} + \xi_{b_{h_{s_k}}}, \tag{5.19}$$

$$P_{k|k} = (I_{18\times18} - K_k H_k) P_{k|k-1} \tag{5.20}$$

5.2.3 *Deep Imp.LIEKF*

L'Imp.LIEKF présenté dans la partie *(5.2.2)* est corrigé en partie par la position et la vitesse du projectile estimées par un LSTM. Cette solution est choisie d'après les résultats obtenus au chapitre précédent, qui montrent qu'un LSTM est parfaitement adapté pour estimer les positions et les vitesses d'un mortier de 120 mm. De plus, cette méthode permet de disposer des mesures absolues de position et de vitesse du projectile sans impliquer de mesures GNSS, en s'affranchissant des limitations de ces signaux et en utilisant que des capteurs inertiels à faible coût et facilement embarquables.
Pour cela, le $LSTM_{ALL}$, sans normalisation et sans rotation du repère de navigation, présenté dans la partie *(4.4)* est réemployé. Pour rappel, les caractéristiques des données d'entrée du LSTM sont $I_{n\ Features} = (\mathcal{M}, \mathcal{P}, \mathcal{T}) \in \mathbb{R}^{16}$ avec :

- $\mathcal{M} \in \mathbb{R}^{12}$ **les données inertielles** les mesures IMU dans le repère capteur $\boldsymbol{s}$ et le champ magnétique de référence dans le repère local de navigation $\boldsymbol{n}$.
- $\mathcal{P} \in \mathbb{R}^3$ **les paramètres de vol :** l'angle de braquage des ailettes, la vitesse initiale en sortie de canon et l'angle d'élévation du canon.

- $\mathcal{T} \in \mathbb{R}^1$ **le vecteur temps** évalué tel que $\mathcal{T} = k\Delta_t$ avec k le pas de temps considéré et $\Delta_t = 1e^{-3}$ la période d'échantillonnage des capteurs.

Conformément aux caractéristiques d'entraînement présentées dans le tableau *(4.2)*, le $LSTM$ est initialement entraîné pour estimer la position, la vitesse et l'orientation du projectile. Lorsque la convergence du $LSTM$ est établie, les résultats d'estimation de ce réseau sont introduites dans le *Deep Imp.LIEKF*. Afin de valider la notion de *Deep Kalman Filter*, le $LSTM$ est entraîné puis intégré au filtre de Kalman sans effectuer d'entraînement spécifique pour produire les pseudo-observations pour le filtre.

Le fonctionnement du *Deep Imp.LIEKF* est résumé dans l'algorithme *1*.

Algorithme 1 *Deep Imp.LIEKF* pour estimer la trajectoire d'un projectile.

Initialisation : $\widehat{\chi}_0$, $\widehat{b}_{\omega_{s_0}}$, $\widehat{b}_{a_{s_0}}$, $\widehat{b}_{h_{s_0}}$, P_0

Étape de prédiction du Deep Imp.LIEKF (5.10) - (5.11) :

$$\begin{bmatrix} \widehat{\chi}_t \\ \widehat{b}_{\omega_s} \\ \widehat{b}_{a_s} \\ \widehat{b}_{h_s} \end{bmatrix} = \begin{bmatrix} f_{u_t}(\widehat{\chi}_t) \\ 0_{3\times1} \\ 0_{3\times1} \\ 0_{3\times1} \end{bmatrix}, \qquad \frac{d}{dt}P_t = A_t P_t + P_t A_t^T + A_{d_\chi} Q_t A_{d_\chi}^T$$

Étape de mise à jour du Deep Imp.LIEKF :

Lecture des mesures : $y_k = \begin{bmatrix} p_{LSTM}^T & v_{LSTM}^T & h_n^T \end{bmatrix}^T$

Gain de Kalman (5.14) - (5.16) : $K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1}$

Mise à jour des prédictions et de la covariance : (5.17) - (5.20)

5.2.4 Résultats d'estimation du *Deep Imp.LIEKF*

Afin d'évaluer l'apport de l'IA pour générer une partie des observations d'un filtre de Kalman nécessaires lors de l'étape de mise à jour, le *Deep Imp.LIEKF* présenté dans l'algorithme *1* est évalué sur les simulations de trajectoires de mortier de 120 mm, générées par BALCO présenté dans la partie *(1.5)* de ce document.

Les figures 5.2 - 5.4 présentent les résultats d'estimation de la position, de la vitesse et des angles d'Euler d'une trajectoire de mortier de 120 mm en comparant le $LSTM$ (en bleu) et le *Deep Imp.LIEKF* (en vert).

Les figures 5.2 - 5.4 montrent que le $LSTM$ (en bleu) et le *Deep Imp.LIEKF* (en vert) ont des performances semblables pour l'estimation des positions et des vitesses du projectile. En effet, le *Deep Imp.LIEKF* lisse les estimations du $LSTM$ grâce à l'étape de prédiction du filtre basée sur des modèles mathématiques. De plus, le *Deep Imp.LIEKF* (en

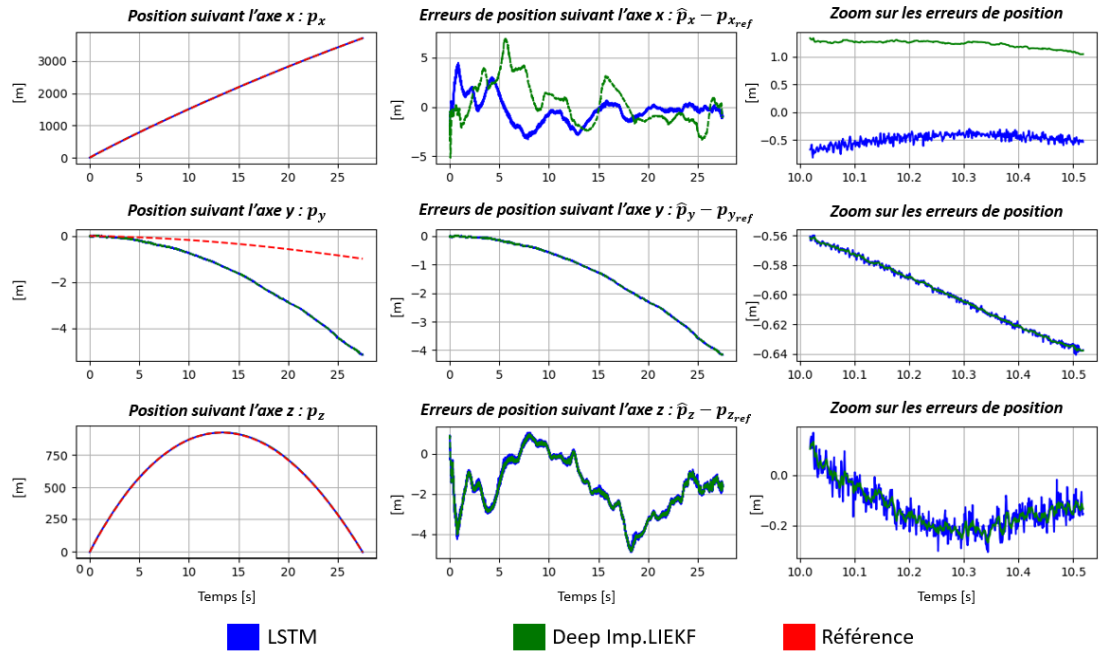

FIGURE 5.2 – Positions estimées et erreurs associées [m] obtenues par le *Deep Imp.LIEKF* (en vert), le *LSTM* (en bleu) et la position de référence (en rouge).

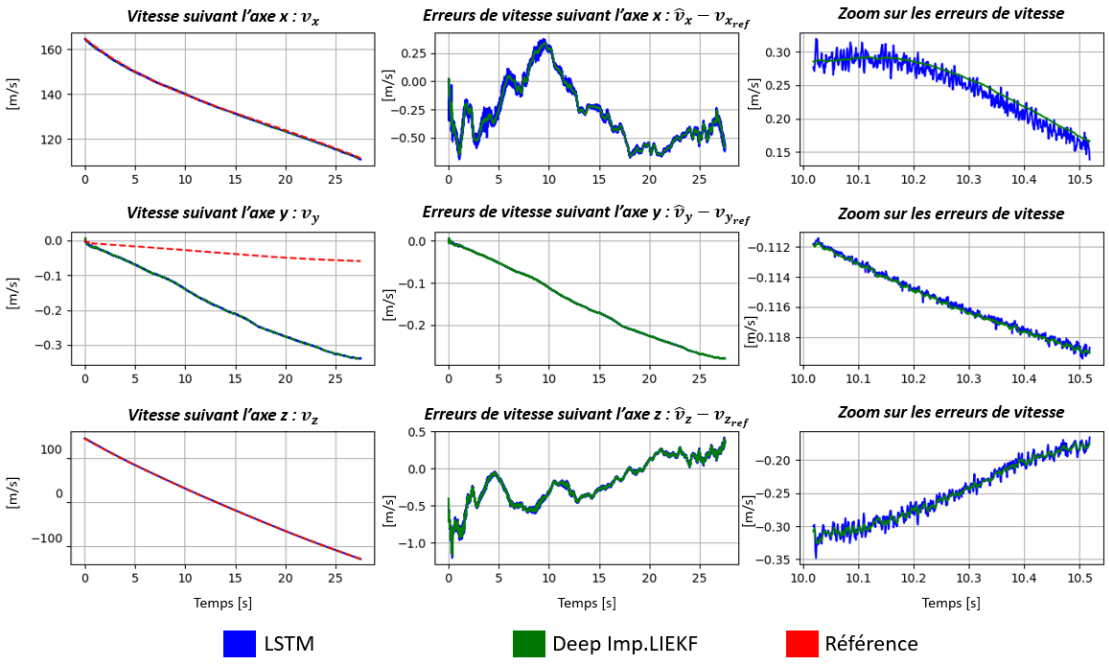

FIGURE 5.3 – Vitesses estimées et erreurs associées [m/s] obtenues par le *Deep Imp.LIEKF* (en vert), le *LSTM* (en bleu) et la vitesse de référence (en rouge).

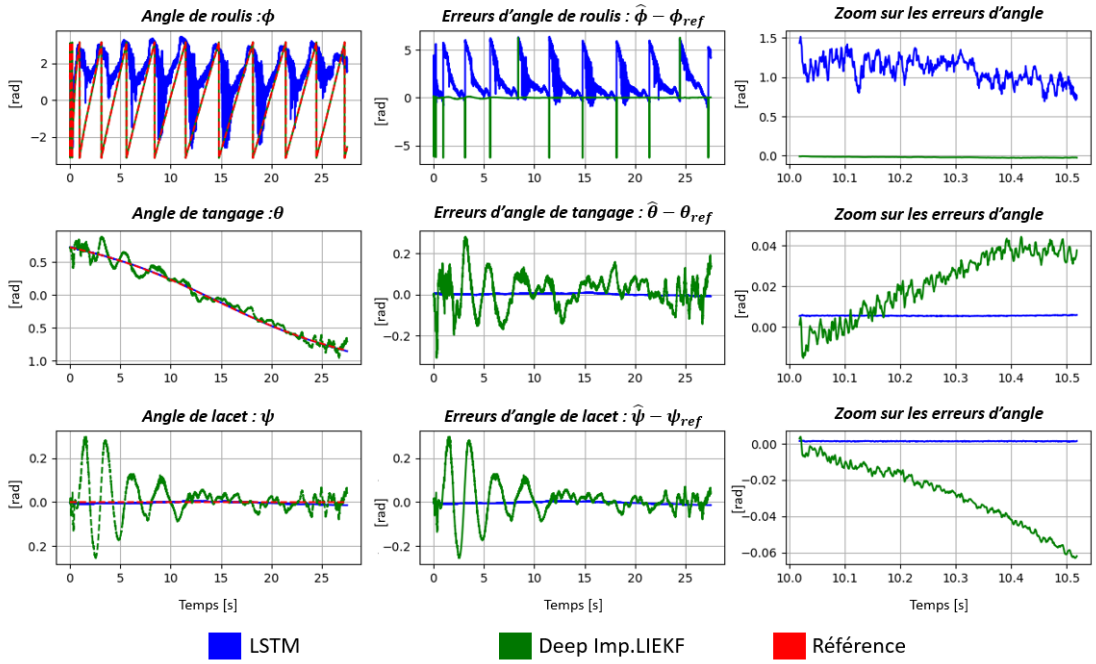

FIGURE 5.4 – Orientations estimées et erreurs associées [rad] obtenues par le *Deep Imp.LIEKF* (en vert), le *LSTM* (en bleu) et l'orientation de référence (en rouge).

vert) exhibe de meilleures performances d'estimation de l'angle de roulis, contrairement aux angles de tangage et de lacet.
Ces premières observations valident l'apport de l'IA pour optimiser un filtre de Kalman. En effet, le *Deep Imp.LIEKF* conserve les bonnes estimations de position et de vitesse du *LSTM*, tout en optimisant l'estimation de l'angle de roulis par l'utilisation d'un modèle mathématique basé sur la connaissance du champ magnétique de référence et des mesures des magnétomètres.

Afin de vérifier ces premières observations, les performances du *Deep Imp.LIEKF* et du *LSTM* sont évaluées via l'erreur quadratique moyenne globale, notée $\mathcal{C}_{RMSE}$, et le score noté $\mathcal{C}_{score}$, obtenu sur l'ensemble du jeu de données de test. Ces deux critères d'erreur sont définis par les équations suivantes :

$$\mathcal{C}_{RMSE} = \frac{1}{N_{sim}} \sum_{k=1}^{N_{sim}} RMSE(sim_k), \tag{5.21}$$

$$\mathcal{C}_{score} = \sum_{k=1}^{N_{sim}} RMSE(Deep\ _{Imp.LIEKF})_{sim_k} < RMSE(LSTM)_{sim_k} \tag{5.22}$$

avec N_{sim} le nombre de simulations du jeu de données de test et avec $RMSE(sim_k)$ l'erreur quadratique moyenne de la grandeur considérée, évaluée pour la simulation k.

La figure 5.5 présente les critères $\mathcal{C}_{RMSE}$ (5.21) et le score $\mathcal{C}_{score}$ (5.22) évalués sur l'ensemble du jeu de données de test en comparant le *Deep Imp.LIEKF* (en vert) et *LSTM* (en bleu) pour l'estimation de trajectoires de mortier de 120 mm.

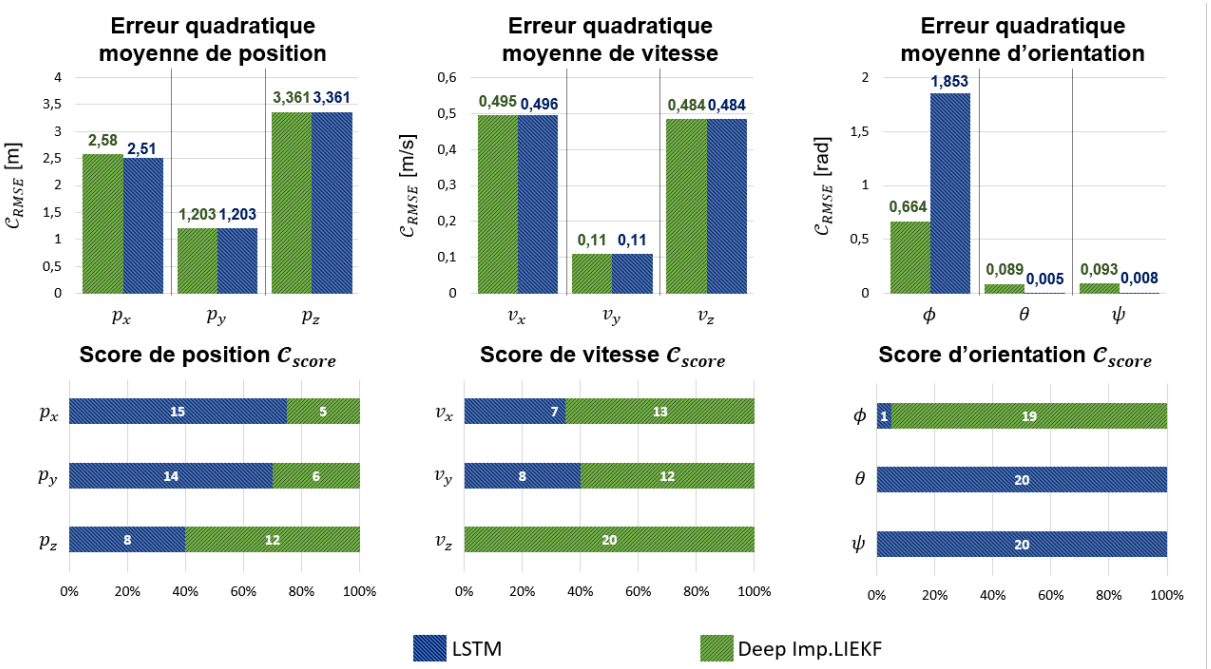

FIGURE 5.5 – Erreur quadratique moyenne globale $\mathcal{C}_{RMSE}$ (5.21) et score $\mathcal{C}_{score}$ (5.22) du *Deep Imp.LIEKF* (en vert) et du *LSTM* (en bleu).

La figure 5.5 montre que les performances des deux algorithmes sont équivalentes pour l'estimation des positions et des vitesses d'un mortier de 120 mm, malgré que le $\mathcal{C}_{score}$ du *LSTM* soit supérieur à celui du *Deep Imp.LIEKF*. De plus, les erreurs d'estimation sont inférieures à 5 m pour les positions et de l'ordre d'un mètre par seconde pour les vitesses. Toutefois, l'intérêt du *Deep Imp.LIEKF* est clairement illustré pour l'estimation de l'angle de roulis. En effet, les prédictions des angles d'Euler par le *Deep Imp.LIEKF* sont corrigées par l'estimation du champ magnétique de référence à partir des magnétomètres. Donc, les angles d'Euler, et tout particulièrement l'angle de roulis, sont estimés par un modèle mathématique basé sur des prédictions précises obtenus aux instants précédents. Cette méthode permet donc de surpasser les limitations du *LSTM* pour l'estimation de cet angle. Toutefois, la formulation du *Deep Imp.LIEKF* détériore l'estimation des angles de tangage et de lacet du fait des biais estimés des magnétomètres.

D'après les figures 5.2 - 5.5, le *Deep Imp.LIEKF* est une méthode précise d'estimation des positions, des vitesses et des angles d'Euler d'un mortier de 120 mm. Cette méthode

présente donc plusieurs avantages. Le premier est que la solution proposée est basée à la fois sur un modèle d'IA et à la fois sur un modèle mathématique. Ainsi, cette méthode gagne en interprétabilité par rapport à une solution basée exclusivement sur l'IA. Le second avantage est que le *Deep Imp.LIEKF* permet de ne considérer aucune mesure GNSS tout en disposant des mesures absolues de position et de vitesse du projectile. En effet, les signaux GNSS ne sont pas toujours disponibles et sont facilement brouillables et leurrables. L'utilisation d'un LSTM permet d'obtenir des mesures similaires à celles d'un GNSS sans partager ces limitations dans le cas où le réseau est entraîné à partir des mesures des capteurs inertiels, de la connaissance du champ magnétique de référence et des paramètres de prévol de la munition. Le dernier avantage du *Deep Imp.LIEKF* est que cette méthode permet de résoudre les mauvaises estimations de l'angle de roulis d'un LSTM, tout en bénéficiant des bonnes estimations de position et de vitesse.

Ainsi, d'après les résultats présentés dans cette partie, la combinaison d'un modèle d'IA et d'un filtre de Kalman est une solution appropriée à l'estimation de la trajectoire d'un mortier de 120 mm. Un LSTM est parfaitement adapté pour produire les pseudo-observations afin de corriger les prédictions d'un filtre de Kalman.

5.3 Estimation du modèle d'erreur d'un algorithme de navigation à l'estime

D'après les résultats présentés dans la partie *(5.2)*, un *Deep Kalman Filter* est parfaitement adapté à l'estimation de la trajectoire d'un mortier de 120 mm dans le cas où un modèle d'IA est utilisé pour générer les observations d'un filtre. Toutefois, un filtre de Kalman est fortement dépendant des observations mesurées lors de l'étape de mise à jour et une mesure erronée peut faire diverger le filtre. Il semble alors préférable, dans le cas où des observations sont disponibles, d'employer l'IA pour remplacer des modèles mathématiques. Il s'agit à présent d'évaluer comment l'IA peut permettre de déterminer l'un des modèles d'un filtre de Kalman afin d'optimiser les estimations.

La première méthode investiguée et proposée dans cette partie, vise à utiliser un réseau de neurones afin de corriger les estimations d'un algorithme de *Dead Reckoning*. Ainsi, le réseau de neurones a pour objectif d'estimer le modèle d'erreur de l'algorithme. Cette méthode a été mise en place dans un cas simple pour vérifier si un réseau de neurones est capable d'estimer un modèle à partir des prédictions précédentes. L'objectif est d'étendre, dans un second temps, cette solution à l'un des modèles d'un filtre de Kalman.

5.3.1 Estimation d'un modèle à partir des prédictions précédentes

La première solution d'utilisation de l'IA pour estimer un modèle a été d'entraîner un réseau de neurones capable de corriger les erreurs d'estimation d'un algorithme de *Dead Reckoning* comme présenté dans la figure 5.6.

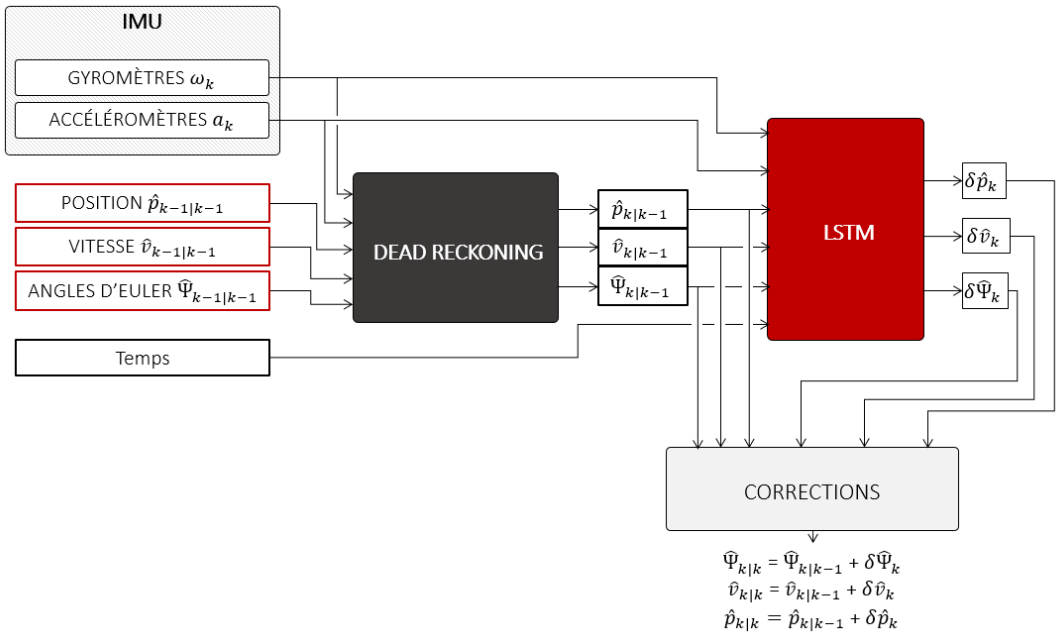

FIGURE 5.6 – Estimation du modèle d'erreur d'un *Dead Reckoning* par un réseau LSTM.

Comme présenté dans la figure 5.6, l'estimation du modèle d'erreur d'un algorithme de *Dead Reckoning* se déroule en plusieurs étapes :

- *Étape n°1 :* un algorithme de *Dead Reckoning* estime la position $\widehat{p}_{k|k-1}$, la vitesse $\widehat{v}_{k|k-1}$ et les angles d'Euler $\widehat{\Psi}_{k|k-1}$ d'un projectile à partir des mesures de l'accéléromètre a_{s_k} et du gyromètre ω_{s_k} effectuées dans le repère capteur et des prédictions aux instants précédents.
- *Étape n°2 :* Un LSTM est entraîné pour estimer les erreurs de position $\delta\widehat{p}_k$, de vitesse $\delta\widehat{v}_k$ et d'orientation $\delta\widehat{\Psi}_k$ de l'algorithme de *Dead Reckoning*, à partir de la trajectoire estimée $\widehat{p}_{k|k-1}$, $\widehat{v}_{k|k-1}$ et $\widehat{\Psi}_{k|k-1}$, des mesures de l'accéléromètre a_{s_k} et du gyromètre ω_{s_k} et d'un vecteur temps.
- *Étape n°3 :* Les prédictions du LSTM $\delta\widehat{p}_k$, $\delta\widehat{v}_k$ et $\delta\widehat{\Psi}_k$ sont utilisées pour corriger les estimations du *Dead Reckoning* $p_{k|k-1}$, $\widehat{v}_{k|k-1}$ et $\widehat{\Psi}_{k|k-1}$.

Cette solution a été testée par différents entraînements de LSTM en modifiant les hyperparamètres ainsi que les couches, mais, aucun résultat cohérent n'a été obtenu. En effet, les erreurs $\delta\widehat{p}_k$, $\delta\widehat{v}_k$ et $\delta\widehat{\Psi}_k$, estimées par le LSTM étaient très importantes et donc les prédictions du *Dead Reckoning* corrigé devenaient erronées. Comme ces prédictions erronées sont utilisées pour la prédiction suivante, cette solution n'a jamais présenté de résultats cohérents.

Ainsi, cette méthode a permis de conclure que sans un pré-entraînement du LSTM, les *Deep Kalman Filter* où l'un des modèles est estimé par un réseau de neurones ne pouvaient présenter de bons résultats. Le pré-entraînement du LSTM est également connu comme une méthode d'apprentissage par transfert. De plus, d'après la littérature [83]-[86], l'apprentissage par transfert est couramment employé pour les *Deep Kalman Filter*.

5.3.2 L'apprentissage par transfert

L'apprentissage par transfert consiste à utiliser un réseau de neurones déjà entraîné à résoudre un problème à exploiter les connaissances apprises afin d'entraîner de nouveau ce même réseau à résoudre un problème similaire mais différent [131]-[133]. Le principe de fonctionnement de l'apprentissage par transfert est présenté sur la figure 5.7.

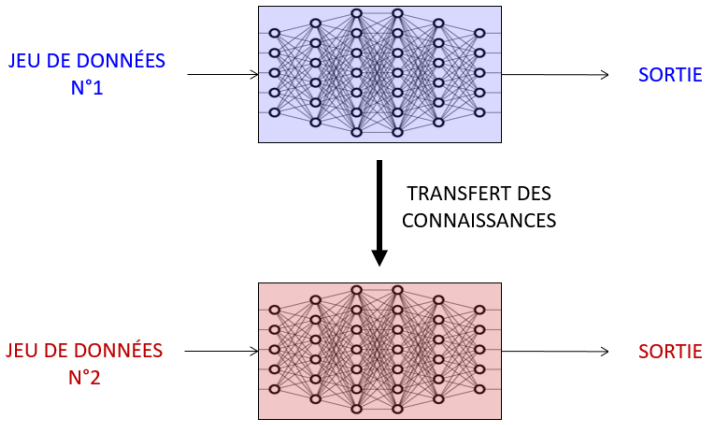

FIGURE 5.7 – L'apprentissage par transfert.

L'apprentissage par transfert permet donc d'utiliser les connaissances déjà apprises sur un premier jeu de données et de les transférer à un second afin d'apprendre des caractéristiques plus spécifiques. Ainsi, le réseau pré-entraîné est adapté afin de résoudre une autre tâche. Cette méthode est utilisée principalement pour optimiser la phase d'apprentissage d'un réseau de neurones, dans le but de réduire le coût d'apprentissage et

d'obtenir de meilleures performances. En effet, le réseau pré-entraîné contient les connaissances globales du problème à traiter et l'apprentissage par transfert permet de réexploiter ces connaissances en se focalisant sur des caractéristiques plus complexes pour résoudre une tâche spécifique.

Il existe différentes catégories d'apprentissage par transfert [134], tel que :

- l'apprentissage par transfert inductif (*inductive transfer learning*), utilisé lorsque les données des deux entraînements sont similaires mais pour résoudre des problèmes différents. Les connaissances apprises lors du premier entraînement sont transférées pour les adapter à une autre tâche. Cet apprentissage est notamment utilisé pour la détection d'objet dans des images. Par exemple, un réseau est pré-entraîné pour identifier des animaux dans des images, puis, l'apprentissage par transfert inductif permet de détecter uniquement les différentes races de chiens.
- l'apprentissage par transfert non supervisé (*unsupervised transfer learning*), semblable à l'apprentissage par transfert inductif, est utilisé lorsque les données des deux entraînements sont similaires mais pas nécessairement étiquetées afin de résoudre des problèmes différents.
- l'apprentissage par transfert transductif (*transductive transfer learning*), utilisé lorsque les tâches sont identiques mais lorsque les données sont de différentes natures .

L'apprentissage par transfert est principalement utilisé pour la classification d'images, la prédiction de séries temporelles et le traitement du langage. Il existe plusieurs méthodes d'apprentissage par transfert suivant le problème à traiter.
Par exemple, l'apprentissage par transfert est utilisé en tant qu'extracteur de caractéristiques, essentiellement appliqué à des problèmes de vision. Il s'agit d'utiliser un réseau de neurones pré-entraîné et de l'entraîner de nouveau afin d'utiliser ses facultés d'extraction de caractéristiques. L'idée est donc de conserver les premières couches du réseau pré-entraîné et d'entraîner uniquement les dernières couches comme les caractéristiques globales sont apprises dans les premières couches. Une seconde utilisation de l'apprentissage par transfert est le *Fine-tuning*. Il s'agit d'utiliser un réseau pré-entraîné et de l'entraîner de nouveau pour l'adapter à une nouvelle tâche, généralement avec un taux d'apprentissage plus faible. Cette méthode permet ainsi d'ajuster le modèle à la nouvelle tâche, pour optimiser les résultats obtenus.

5.4 Estimation du modèle d'évolution d'un filtre de Kalman

Les résultats des parties *(5.2)* et *(5.3)* montrent qu'un *Deep Kalman Filter* est une solution appropriée à l'estimation des trajectoires d'un projectile. Toutefois, dans le cas de l'estimation d'un modèle d'un filtre de Kalman par l'IA, une méthode d'apprentissage par transfert est nécessaire pour obtenir des prédictions cohérentes. Ainsi, d'après ces premières observations, des réseaux de neurones peuvent être entraînés afin de remplacer l'un des modèles d'un filtre de Kalman comme dans [83]-[86]. Cette solution permet notamment de résoudre des problèmes d'approximation afin de modéliser correctement des dynamiques inconnues et d'éventuelles corrélations entre les capteurs.

De plus, comme mentionné dans la partie *(1.2)*, un projectile est soumis à de fortes contraintes dynamiques, ainsi qu'à des contraintes de coût, d'espace et de consommation énergétique. C'est pourquoi, seules une unité de mesure inertielle (IMU) et un récepteur GNSS sont embarqués dans les projectiles. Cependant, du fait de la vulnérabilité des signaux GNSS face aux menaces extérieures, ces signaux ne sont pas exploités dans cette thèse. De surcroît, l'intégration des mesures IMU fournit une solution de navigation précise à court terme mais dévie à long terme en raison de la dérive des capteurs. Ainsi, une modélisation précise des erreurs des capteurs inertiels permet d'obtenir un modèle précis de navigation inertielle. C'est ce problème de modélisation qui est traité dans cette partie.

Cette partie présente un *Deep Kalman Filter* pour estimer avec précision la trajectoire d'un projectile dans un environnement sans GNSS en utilisant uniquement l'IMU intégrée et le champ magnétique de référence. Pour cela, un *Deep Error State Kalman Filter* est proposé où le modèle d'évolution du filtre est remplacé par un LSTM, permettant ainsi de modéliser les erreurs des capteurs inertiels afin de produire une solution de navigation précise à long terme. De plus, d'après les observations formulées dans la partie *(5.3)*, une méthode d'apprentissage par transfert est employée pour minimiser les erreurs.

5.4.1 Présentation du problème d'estimation

Afin de présenter le *Deep Error State Kalman Filter (Deep ES-KF)* où le modèle d'évolution est remplacé par un LSTM, un *Error State Kalman Filter (ES-KF)* traditionnel est d'abord formulé pour estimer la trajectoire d'un projectile à partir de l'IMU et du champ magnétique de référence. Cette section détaille le fonctionnement général du *Deep*

ES-KF, introduit la méthodologie de l'ES-KF et présente les notations employées pour dériver le *Deep ES-KF*.

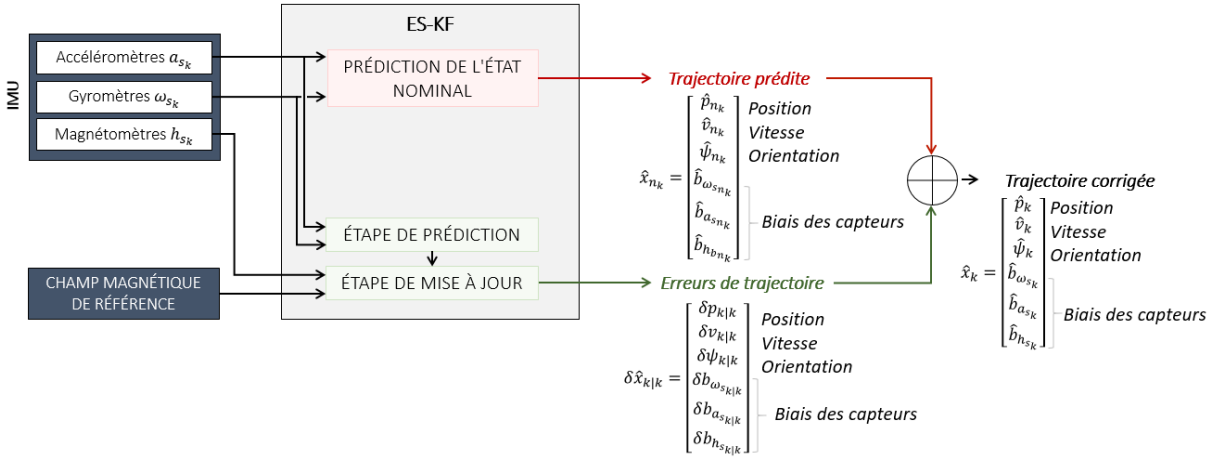

FIGURE 5.8 – ES-KF traditionnel pour estimer une trajectoire de projectile (position, vitesse, angles d'Euler) à partir de l'IMU et du champ magnétique de référence.

Dans le cas d'un *Error State Kalman Filter* traditionnel comme présenté dans la figure 5.8, la trajectoire nominale est tout d'abord prédite à partir d'un modèle mathématique d'évolution et des mesures des accéléromètres et des gyromètres embarqués. Dans un second temps, l'étape de prédiction de l'ES-KF détermine les erreurs de trajectoire et la matrice de covariance de l'erreur à partir des mesures des accéléromètres et des gyromètres et de la linéarisation du modèle mathématique d'évolution. Dans un troisième temps, l'étape de mise à jour de l'ES-KF corrige ces erreurs de trajectoire à partir d'observations qui sont l'estimation du champ magnétique de référence à partir des mesures des magnétomètres. Enfin, la trajectoire nominale initialement estimée par un modèle mathématique est corrigée par les erreurs de trajectoire évaluées lors de l'étape de mise à jour de l'ES-KF.

A partir de ce filtre, un *Deep Error State Kalman Filter* est développé. L'étape de prédiction de l'état nominal de l'ES-KF présenté sur la figure 5.8 est remplacée par un *Long Short-Term Memory (LSTM)* pour prédire la trajectoire nominale comme illustré sur la figure 5.9. Ensuite, l'ES-KF détermine classiquement les erreurs de trajectoire, afin de corriger la trajectoire nominale prédite par le LSTM.

Les données d'entrée du LSTM sont les mesures de l'IMU embarquée (accéléromètre, gyromètre et magnétomètre), le champ magnétique de référence, des paramètres de prévol spécifiques à la munition considérée et un vecteur temps. Le LSTM permet ainsi de modé-

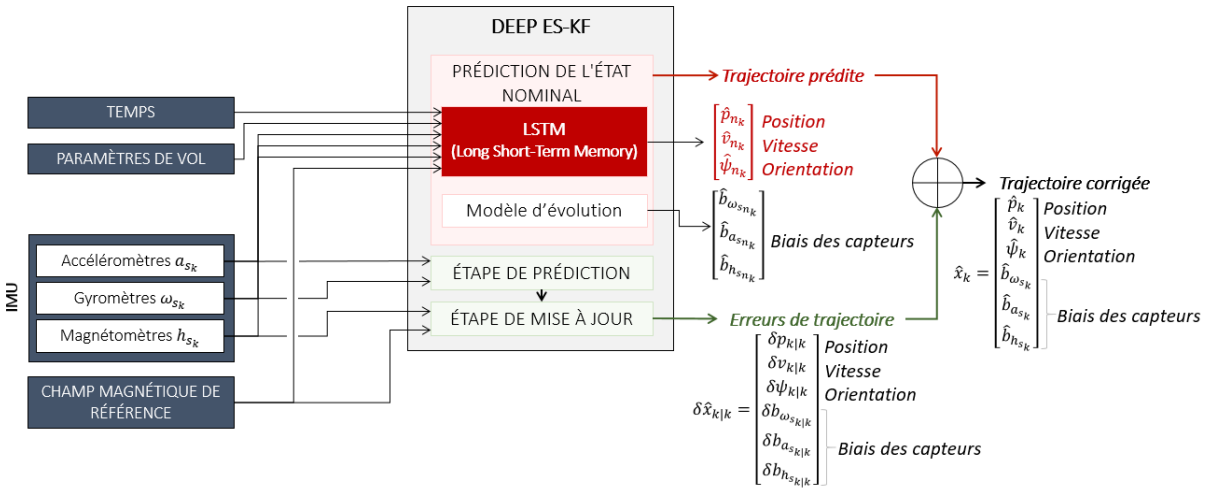

FIGURE 5.9 – *Deep ES-KF* pour estimer la trajectoire d'un projectile. Une partie de l'état nominal est prédit par un LSTM dont les données d'entrée sont un vecteur temporel, des paramètres de vol de la munition, l'IMU embarquée et le champ magnétique de référence. L'état nominal prédit est ensuite corrigé par les erreurs estimées par l'ES-KF lors de l'étape de prédiction et de mise à jour.

liser les erreurs des capteurs inertiels afin de produire une solution de navigation précise à long terme. De plus, afin d'optimiser la solution proposée et d'après les observations formulées dans la partie *(5.3)*, une méthode d'apprentissage par transfert est proposée pour minimiser les erreurs d'estimation *Deep ES-KF*.

Méthodologie d'un *Error State Kalman Filter (ES-KF)*

La méthodologie d'un *Error State Kalman Filter* est détaillée dans [117], [135], [136] et est brièvement rappelée dans cette partie.
Soit un système non linéaire à temps discret défini comme suit :

$$x_k = f(x_{k-1}, u_k, w_k) \tag{5.23}$$

$$y_k = h(x_k, v_k) \tag{5.24}$$

avec $x_k \in \mathbb{R}^{n\times 1}$ les états, $u_k \in \mathbb{R}^{l\times 1}$ l'entrée de commande, $y_k \in \mathbb{R}^{m\times 1}$ les mesures, $f(.)$ et $h(.)$ les modèles d'évolution et d'observation non linéaires, et $w_k \sim \mathcal{N}(0, Q_k)$ et $v_k \sim \mathcal{N}(0, R_k)$ les bruits de modèle et de mesure supposés blancs gaussiens et indépendants.

Un *Error State Kalman Filter (ES-KF)* estime les états x_k du système non linéaire

(5.23) comme une combinaison de l'état nominal, noté x_{n_k}, et de l'état d'erreur noté δx_k :

$$x_k = x_{n_k} \oplus \delta x_k \tag{5.25}$$

avec $\oplus$ la somme de deux variables définies dans $\mathbb{R}$ sauf pour les quaternions où cet opérateur désigne le produit de quaternions :

$$q_k = q_{n_k} \otimes \delta q. \tag{5.26}$$

Tout d'abord, l'ES-KF évalue l'état nominal $\widehat{x}_{n_k}$ avec le modèle d'évolution $f(.)$. Deuxièmement, l'état d'erreur $\widehat{\delta}x_{k|k-1}$ et la covariance d'état d'erreur $P_{k|k-1}$ sont prédits par linéarisation du modèle d'évolution $f(.)$. Troisièmement, les observations mesurées y_k mettent à jour l'état d'erreur $\widehat{\delta}x_{k|k}$ et la covariance $P_{k|k}$ comme tous les filtres de Kalman standards. Enfin, l'état nominal est corrigé par l'état d'erreur tel que $\widehat{x}_{k|k} = \widehat{x}_{n_k} \oplus \widehat{\delta}x_{k|k}$.

L'état d'erreur δx_k, erreur entre l'état vrai x_k et l'état nominal x_{n_k}, est un signal de petite amplitude et associé à une faible dynamique. Ainsi, sous l'hypothèse de faibles bruits, la linéarisation au premier ordre de cette dynamique reste valable, contrairement à un EKF traditionnel qui suppose de petites erreurs d'estimation pas toujours satisfaites. Cela explique pourquoi un ES-KF présente de meilleures performances qu'un EKF sur des problèmes similaires. De plus, l'état d'erreur δx_k est petit et proche de l'origine donc l'ES-KF évite d'éventuels problèmes de singularité [117].

Modèles et notations

Les modèles de mesure du gyromètre $\tilde{\omega}_s$, de l'accéléromètre $\tilde{a}_s$ et du magnétomètre $\tilde{h}_s$ sont définis tels que :

$$\tilde{\omega}_s = \omega_s + b_{\omega_s} + W_{\omega_s}, \quad \tilde{a}_s = a_s + b_{a_s} + W_{a_s}, \quad \tilde{h}_s = h_s + b_{h_s} + W_{h_s}, \tag{5.27}$$

avec ω_s, a_s et h_s les mesures des capteurs inertiels, W_{ω_s}, W_{a_s} et W_{h_s} des bruits blancs gaussiens, et b_{ω_s}, b_{a_s}, b_{a_s} les biais des trois capteurs dont la dynamique est modélisée par :

$$\dot{b}_{\omega_s} = W_{b_{\omega_s}}, \qquad \dot{b}_{a_s} = W_{b_{a_s}}, \qquad \dot{b}_{h_s} = W_{b_{h_s}}, \tag{5.28}$$

avec $W_{b_{\omega_s}}$, $W_{b_{a_s}}$ et $W_{b_{h_s}}$ des bruits blancs gaussiens.

L'ES-KF estime la trajectoire d'un projectile dans le repère de navigation local $\boldsymbol{n}$ à

partir de l'IMU et du champ magnétique de référence. Les 20 états à estimer par l'ES-KF sont alors :

$$x \triangleq \begin{bmatrix} q^T & v^T & p^T & b_{\omega_s}^T & b_{a_s}^T & b_{h_s}^T \end{bmatrix}^T \tag{5.29}$$

avec $q \in \mathbb{H}$ le quaternion d'orientation qui modélise l'orientation du projectile dans le repère de navigation local $\boldsymbol{n}$, v et $p \in \mathbb{R}^{3\times 1}$ la vitesse et la position du projectile et b_{ω_s}, b_{a_s}, $b_{h_s} \in \mathbb{R}^{3\times 1}$ les biais des capteurs inertiels.

La dynamique associée aux états x décrivant la trajectoire d'un projectile est :

$$\begin{array}{ccc} \dot{x} & = & f(x,u,w) \\ \begin{bmatrix} \dot{q} \\ \dot{v} \\ \dot{p} \\ \dot{b}_{\omega_s} \\ \dot{b}_{a_s} \\ \dot{b}_{h_s} \end{bmatrix} & = & \begin{bmatrix} \frac{1}{2} q \otimes q\{\omega_s\} \\ R\{q\} a_s + g \\ v \\ W_{b_{\omega_s}} \\ W_{b_{a_s}} \\ W_{b_{h_s}} \end{bmatrix} \end{array} \tag{5.30}$$

avec $R\{q\} \in SO(3)$ la matrice de cosinus directeur représentant l'orientation du projectile dans le repère de navigation et définie dans l'annexe B, avec $g \in \mathbb{R}^{3\times 1}$ le vecteur gravité supposé constant pendant le vol du projectile, $W_{b_{\omega_s}}$, $W_{b_{a_s}}$ et $W_{b_{h_s}} \in \mathbb{R}^{3\times 1}$ des bruits blancs gaussiens de l'évolution des biais.

Le bruit de modèle $w \sim \mathcal{N}(0, Q)$ et l'entrée de commande u sont :

$$w = \begin{bmatrix} W_{\omega_s}^T & W_{a_s}^T & W_{b_{\omega_s}}^T & W_{b_{a_s}}^T & W_{b_{h_s}}^T \end{bmatrix}^T \in \mathbb{R}^{15\times 1}, \quad u = \begin{bmatrix} \omega_s^T & a_s^T \end{bmatrix}^T \in \mathbb{R}^{6\times 1}. \tag{5.31}$$

Les observations considérées sont l'estimation du champ magnétique dans le repère de navigation à partir de mesures des magnétomètres. Le modèle d'observation non linéaire (5.24) en temps continu est alors :

$$\begin{array}{lll} y = & h(x, W_h) & \\ h_n = & R\{q\} h_s & = R\{q\} \left(\tilde{h}_s - b_{h_s} - W_{h_s} \right) \end{array} \tag{5.32}$$

avec $h_n \in \mathbb{R}^{3\times 1}$ le champ magnétique de référence exprimé dans le repère de navigation local $\boldsymbol{n}$ et $W_{h_s} \in \mathbb{R}^{3\times 1} \sim \mathcal{N}(0, R)$ le bruit de mesure des magnétomètres.

L'état nominal de l'ES-KF x_n et l'état d'erreur δx, modélisant les erreurs entre l'état

vrai x et l'état nominal x_n, sont définis par les équations suivantes :

$$x_n \triangleq \begin{bmatrix} q_n^T & v_n^T & p_n^T & b_{\omega_{s_n}}^T & b_{a_{s_n}}^T & b_{h_{s_n}}^T \end{bmatrix}^T \in \mathbb{R}^{19\times 1}, \tag{5.33}$$

$$\delta x \triangleq \begin{bmatrix} \delta\theta^T & \delta v^T & \delta p^T & \delta b_{\omega_s}^T & \delta b_{a_s}^T & \delta b_{h_s}^T \end{bmatrix}^T \in \mathbb{R}^{18\times 1}, \tag{5.34}$$

avec $\delta\theta \in \mathbb{R}^{3\times 1}$ l'erreur d'angle du quaternion de sorte que le quaternion orientation soit évalué par l'équation suivante :

$$q = q_n \otimes \delta q = q_n \otimes \begin{bmatrix} 1 \\ \frac{1}{2}\delta\theta \end{bmatrix}. \tag{5.35}$$

5.4.2 ES-KF pour estimer la trajectoire d'un projectile

Le fonctionnement de l'*Error State Kalman Filter (ES-KF)* implémenté pour estimer la trajectoire d'un projectile à partir des mesures des accéléromètres et des gyromètres, puis corrigé par les mesures des magnétomètres et le champ magnétique de référence, est illustré dans la figure 5.8. Cette section fournit les équations composant ce filtre, à savoir, la prédiction de l'état nominal, les étapes de prédiction et de mise à jour et l'étape de correction de l'état nominal.

Prédiction de l'état nominal $\widehat{x}_{n_k}$

L'état nominal $\widehat{x}_{n_k}$ est prédit tel que,

$$\widehat{x}_{n_k} = f(\widehat{x}_{k-1|k-1}, u_k, 0) \tag{5.36}$$

Ainsi, selon la dynamique du projectile (5.30), l'état nominal prédit $\widehat{x}_{n_k}$ en temps discret est donné par les équations suivantes :

$$\widehat{q}_{n_k} = \left(\sum_{k=0}^{\infty} \frac{\left(\frac{1}{2}\Omega(\tilde{\omega}_{s_k} - \widehat{b}_{\omega_{s_{k-1|k-1}}})\Delta t\right)^k}{k!} \right) \widehat{q}_{k-1|k-1}, \tag{5.37}$$

$$\widehat{v}_{n_k} = \widehat{v}_{k-1|k-1} + \left(\widehat{R}_{k-1|k-1}(\tilde{a}_{s_k} - \widehat{b}_{a_{s_{k-1|k-1}}}) + g \right) \Delta t, \tag{5.38}$$

$$\widehat{p}_{n_k} = \widehat{p}_{k-1|k-1} + \widehat{v}_{k-1|k-1}\Delta t + \tfrac{1}{2}\left(\widehat{R}_{k-1|k-1}(\tilde{a}_{s_k} - \widehat{b}_{a_{s_{k-1|k-1}}}) + g \right) \Delta t^2, \tag{5.39}$$

$$\widehat{b}_{\omega_{s_{n_k}}} = \widehat{b}_{\omega_{s_{k-1|k-1}}},\ \widehat{b}_{a_{s_{n_k}}} = \widehat{b}_{a_{s_{k-1|k-1}}},\ \widehat{b}_{h_{s_{n_k}}} = \widehat{b}_{h_{s_{k-1|k-1}}}, \tag{5.40}$$

avec Δt la période d'échantillonnage des capteurs inertiels et l'opérateur des quaternions

$\Omega(.)$ présenté dans l'annexe B de ce document.

Étape de prédiction de l'ES-KF

L'état d'erreur $\widehat{\delta}x_{k|k-1}$ et la prédiction de la matrice de covariance d'état d'erreur $P_{k|k-1}$ sont donnés par les équations suivantes :

$$\widehat{\delta}x_{k|k-1} = F_k\widehat{\delta}x_{k-1|k-1} + F_w w_k, \qquad P_{k|k-1} = F_k P_{k-1|k-1}F_k^T + F_w Q_k F_w^T, \tag{5.41}$$

avec F_k la matrice jacobienne en temps discret du modèle d'évolution $f(.)$ avec le respect de l'état d'erreur δx_k et F_w la matrice jacobienne en temps discret du modèle d'évolution $f(.)$ avec le respect du bruit du modèle w_k tel que :

$$F_k = \textstyle\sum_{k=0}^{\infty} \frac{(A\Delta t)^k}{k!} \approx I + A\Delta t \;\text{ avec } A = \left.\frac{\partial f}{\partial \delta x}\right|_{(x,u)}, \text{ et } F_w = \left.\frac{\partial f}{\partial w}\right|_{(x_k,\delta x_k,u_k,w)}. \tag{5.42}$$

La dynamique de l'état d'erreur δx en temps continu est alors :

$$\delta\dot{\theta} = -[\tilde{\omega}_s - b_{\omega_s}]_\times \delta\theta - \delta b_{\omega_s} - W_{\omega_s}, \tag{5.43}$$

$$\delta\dot{v} = -R[\tilde{a}_s - b_{a_s}]_\times \delta\theta - R\delta b_{a_s} - RW_{a_s}, \qquad \delta\dot{p} = \delta v, \tag{5.44}$$

$$\delta\dot{b}_{\omega_s} = W_{b_{\omega_s}}, \qquad \delta\dot{b}_{a_s} = W_{b_{a_s}}, \qquad \delta\dot{b}_{h_s} = W_{b_{h_s}}. \tag{5.45}$$

La dérivation complète de l'état d'erreur δx est donnée dans [117].

La discrétisation de la dynamique de l'état d'erreur (5.43) - (5.45) à la période d'échantillonnage Δt permet d'identifier la matrice jacobienne $F_k \in \mathbb{R}^{18\times18}$ et $F_w \in \mathbb{R}^{18\times15}$ définie par (5.42) telle que :

$$F_k = I_{18\times18} + \begin{bmatrix} -[\tilde{\omega}_{s_k} - \widehat{b}_{\omega_{s_{n_k}}}]_\times & 0 & 0 & -I & 0 & 0 \\ -\widehat{R}_{n_k}[\tilde{a}_{s_k} - \widehat{b}_{a_{s_{n_k}}}]_\times & 0 & 0 & 0 & -\widehat{R}_{n_k} & 0 \\ 0 & I & 0 & 0 & 0 & 0 \\ & & & 0_{9\times18} & & \end{bmatrix}\Delta t, \quad F_w = \begin{bmatrix} -I\Delta t & 0 & \\ 0 & -\widehat{R}_{n_k}\Delta t & 0_{9\times9} \\ 0 & 0 & \\ 0_{9\times6} & & I_{9\times9} \end{bmatrix} \tag{5.46}$$

La prédiction de la covariance est donnée par (5.41) avec Q_k la matrice de covariance du bruit du modèle (5.31).

Étape de mise à jour de l'ES-KF

L'état d'erreur $\widehat{\delta}x_{k|k}$ et la covariance $P_{k|k}$ sont mises à jour avec les observations y_k, le modèle de mesure $h(.)$ (5.32) et le gain de Kalman K_k tel que :

$$K_k = P_{k|k-1}H_k^T(H_kP_{k|k-1}H_k^T + R_k)^{-1}, \tag{5.47}$$

$$\delta\widehat{x}_{k|k} = K_k(y_k - h(\widehat{x}_{k|k-1}, 0)), \tag{5.48}$$

$$P_{k|k} = (I - K_kH_k)P_{k|k-1}, \tag{5.49}$$

avec H_k la matrice jacobienne en temps discret du modèle d'observation $h(.)$ par rapport à l'état d'erreur δx_k tel que :

$$H_k = \frac{\partial h}{\partial \delta x}\Big|_{x_k} = \frac{\partial h}{\partial x}\Big|_{x_k}\frac{\partial x}{\partial \delta x}\Big|_{x_k} = H_xX_{\delta X}. \tag{5.50}$$

D'après le modèle de mesure (5.32), les matrices jacobiennes en temps discret $H_x \in \mathbb{R}^{3\times 20}$ et $X_{\delta X} \in \mathbb{R}^{20\times 18}$ sont :

$$H_x = \begin{bmatrix} \frac{\partial R\{q\}h_b}{\partial q} & 0 & 0 & 0 & 0 & R\{q\} \end{bmatrix}, \quad X_{\delta X} = \begin{bmatrix} \frac{\partial(q\otimes\delta q)}{\partial\delta\theta} & 0_{3\times 15} \\ 0_{15\times 3} & I_{15\times 15} \end{bmatrix} \tag{5.51}$$

avec les matrices $\frac{\partial R\{q\}h_b}{\partial q}$ et $\frac{\partial(q\otimes\delta q)}{\partial\delta\theta}$ définies dans l'annexe *B*.

Correction de l'état nominal

L'état nominal x_{n_k} (5.37) - (5.40) est mis à jour avec l'état d'erreur $\delta\widehat{x}_{k|k}$ tel que :

$$\widehat{q}_{k|k} = \widehat{q}_{n_k} \otimes q\{\widehat{\delta}q_{k|k}\} = \widehat{q}_{n_k} \otimes \begin{bmatrix} 1 \\ \frac{1}{2}\widehat{\delta}\theta_{k|k} \end{bmatrix}, \tag{5.52}$$

$$\widehat{v}_{k|k} = \widehat{v}_{n_k} + \widehat{\delta}v_{k|k}, \tag{5.53}$$

$$\widehat{p}_{k|k} = \widehat{p}_{n_k} + \widehat{\delta}p_{k|k}, \tag{5.54}$$

$$\widehat{b}_{\omega_{s_{k|k}}} = \widehat{b}_{\omega_{s_{n_k}}} + \widehat{\delta}b_{\omega_{s_{k|k}}}, \quad \widehat{b}_{a_{s_{k|k}}} = \widehat{b}_{a_{s_{n_k}}} + \widehat{\delta}b_{a_{s_{k|k}}}, \quad \widehat{b}_{h_{s_{k|k}}} = \widehat{b}_{h_{s_{n_k}}} + \widehat{\delta}b_{h_{s_{k|k}}}. \tag{5.55}$$

Le quaternion orientation $\widehat{q}_{k|k}$ (5.52) est normalisé tel que :

$$\widehat{q}_{k|k} = \frac{\widehat{q}_{k|k}}{\|\widehat{q}_{k|k}\|}. \tag{5.56}$$

Les équations de l'ES-KF traditionnel implémenté pour estimer une trajectoire de projectile sont résumées dans l'algorithme *2* ci-dessous.

Algorithme 2 ES-KF pour estimer la trajectoire d'un projectile.

Initialisation : $\widehat{x}_{n_0}$, $\widehat{\delta}x_{0|0}$, $P_{0|0}$
Prédiction de l'état nominal (5.37)-(5.40) :
$\widehat{x}_{n_k} = f(\widehat{x}_{k-1|k-1}, u_k, 0)$
Étape de prédiction de l'ES-KF (5.41) - (5.46) :
$\widehat{\delta}x_{k|k-1} = F_k\widehat{\delta}x_{k-1|k-1} + F_w w_k$
$P_{k|k-1} = F_k P_{k-1|k-1}F_k^T + F_w Q_k F_w^T$
Étape de mise à jour de l'ES-KF (5.47)-(5.50) :
$K_k = P_{k|k-1}H_k^T(H_k P_{k|k-1}H_k^T + R_k)^{-1}$
$\delta\widehat{x}_{k|k} = K_k(y_k - h(\widehat{x}_{k|k-1}, 0))$
$P_{k|k} = (I - K_k H_k)P_{k|k-1}$
Correction de l'état nominal (5.52)-(5.56) :
$\widehat{x}_{k|k} = \widehat{x}_{n_k} \oplus \delta\widehat{x}_{k|k}$

5.4.3 *Deep ES-KF*

A partir de l'*Error State Kalman Filter (ES-KF)* présenté dans la partie précédente *(5.4.2)*, un *Deep Kalman Filter*, illustré dans la figure 5.9, est développé. Un *Long Short-Term Memory (LSTM)* est entraîné pour estimer une partie des états nominaux de l'ES-KF x_{n_k}, c'est-à-dire les angles d'Euler nominaux ψ_{n_k}, la vitesse nominale v_{n_k} et la position nominale p_{n_k}. Les biais nominaux $b_{\omega_{s_{n_k}}}$, $b_{a_{s_{n_k}}}$ et $b_{h_{s_{n_k}}}$ sont déterminés à partir du modèle défini par l'équation (5.40). Cette section détaille les données d'entrée ainsi que les caractéristiques d'entraînement du LSTM.

Données d'entrée du LSTM

Les caractéristiques des données d'entrée sont $I_{n\ Features} = (\mathcal{M}, \mathcal{P}, \mathcal{T}) \in \mathbb{R}^{16}$ avec :

- $\mathcal{M} \in \mathbb{R}^{12}$ *les données inertielles* comprenant les mesures de l'IMU et le champ magnétique de référence,
- $\mathcal{P} \in \mathbb{R}^3$ *les paramètres de prévol*, c'est-à-dire l'angle de braquage des ailettes, la vitesse initiale et l'angle d'élévation du canon dans le cas d'un mortier de 120 mm.
- $\mathcal{T} \in \mathbb{R}^1$ *le vecteur temps* évalué tel que $\mathcal{T} = k\Delta_t$ avec k le pas de temps considéré et Δ_t la période d'échantillonnage des capteurs inertiels.

Le LSTM estime 9 caractéristiques de sortie qui sont :

$$O_{ut\ Features} = \begin{bmatrix} \Psi & v & p \end{bmatrix} \in \mathbb{R}^9, \tag{5.57}$$

avec Ψ, v et p les angles d'Euler, la vitesse et la position du projectile dans le repère de navigation.

Estimation de l'état nominal par un LSTM

Une méthode d'apprentissage par transfert est utilisée pour entraîner le LSTM. Dans le cas *Deep ES-KF*, comme le montre la figure 5.10, le LSTM est initialement entraîné pour estimer la trajectoire du projectile $O_{ut\ Features}$ (5.57) à partir des données d'entrée $I_{n\ Features}$.
Une fois la convergence LSTM établie, ce réseau est à nouveau entraîné avec la méthode de *fine tuning* pour prédire les états $\begin{bmatrix} \Psi_{n_k} & v_{n_k} & p_{n_k} \end{bmatrix}$ tandis que les biais $b_{\omega_{s_{n_k}}}$, $b_{a_{s_{n_k}}}$ et $b_{h_{s_{n_k}}}$ sont donnés par l'équation (5.40). Avec l'état nominal complet x_{n_k}, déterminé à la fois par le LSTM et par un modèle mathématique d'évolution des biais, l'état d'erreur de l'ES-KF $\widehat{\delta}x_{k|k}$ est prédit puis mis à jour pour corriger l'état nominal x_{n_k} conformément aux équations (5.52) - (5.56). Les deux parties du processus d'apprentissage du LSTM sont illustrées dans la figure 5.10.

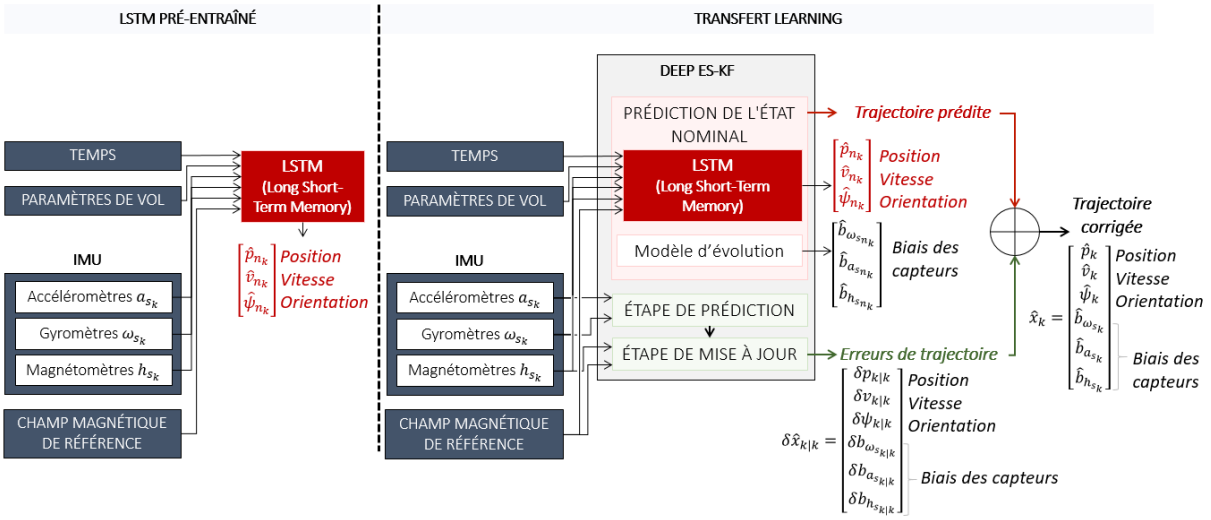

FIGURE 5.10 – 1) LSTM entraîné pour estimer la trajectoire du projectile (Ψ, v, p). 2) Apprentissage par transfert : ajustement des paramètres LSTM pour prédire les états nominaux ES-KF Ψ_{n_k}, v_{n_k} et p_{n_k}.

Les données d'entrée du LSTM $I_{n\ Features}$ n'incluent pas les trajectoires prédites aux instants précédents $\widehat{x}_{k-1|k-1}$. Cette option permet ainsi au *Deep ES-KF* d'être indépendant entre chaque estimation du LSTM et évite ainsi tout problème de dérive de l'ES-KF.

Détails d'entraînement LSTM

Le jeu de données d'entraînement est composé de 1 000 simulations, le jeu de données de validation est composé de 100 simulations et le jeu de données de test est composé de 200 simulations générées par BALCO [96]. La taille du *batch* est de 64 et la longueur de la séquence est de 20 pas de temps pour capturer suffisamment de dépendances à long terme sans dépendre du bruit de mesure.
Les réseaux sont mis à jour par l'algorithme d'optimisation d'Adam [51] et la perte entre les prédictions $\widehat{x}_k$ et la trajectoire de référence x_{ref_k} est évaluée avec l'erreur quadratique moyenne *(Mean Squared Error - MSE)* :

$$MSE = \frac{1}{N}\sum_{k=1}^{N}(\widehat{x}_k - x_{ref_k})^2 \tag{5.58}$$

Le *Deep ES-KF* mis en œuvre pour estimer la trajectoire d'un projectile est résumé dans l'algorithme *3* ci-dessous.

Algorithme 3 *Deep ES-KF* pour estimer la trajectoire d'un projectile.

Initialisation : $\widehat{\delta x}_{0|0}$, $P_{0|0}$
Prédiction de l'état nominal $\widehat{x}_{n_k}$:

$$\begin{bmatrix}\widehat{\Psi}_{n_k} & \widehat{v}_{n_k} & \widehat{p}_{n_k}\end{bmatrix} = LSTM(\mathcal{M},\mathcal{P},\mathcal{T})$$
$$\begin{bmatrix}\widehat{b}_{\omega_{s_{n_k}}} & \widehat{b}_{a_{s_{n_k}}} & \widehat{b}_{h_{s_{n_k}}}\end{bmatrix} = \begin{bmatrix}\widehat{b}_{\omega_{s_{k-1|k-1}}} & \widehat{b}_{a_{s_{k-1|k-1}}} & \widehat{b}_{h_{s_{k-1|k-1}}}\end{bmatrix}$$

Étape de prédiction de l'ES-KF (5.46) :

$$\widehat{\delta x}_{k|k-1} = F_k\widehat{\delta x}_{k-1|k-1} + F_w w_k$$
$$P_{k|k-1} = F_k P_{k-1|k-1}F_k^T + F_w Q_k F_w^T$$

Étape de mise à jour de l'ES-KF (5.47)-(5.50) :

$$K_k = P_{k|k-1}H_k^T(H_k P_{k|k-1}H_k^T + R_k)^{-1}$$
$$\delta\widehat{x}_{k|k} = K_k(y_k - h(\widehat{x}_{k|k-1}, 0))$$
$$P_{k|k} = (I - K_k H_k)P_{k|k-1}$$

Correction de l'état nominal (5.52)-(5.56) :

$$\widehat{x}_{k|k} = \widehat{x}_{n_k} \oplus \delta\widehat{x}_{k|k}$$

5.4.4 Résultats d'estimation du *Deep ES-KF*

Afin de valider l'apport de l'IA pour évaluer l'un des modèles mathématiques d'un filtre de Kalman, la précision du *Deep ES-KF* est évaluée sur les trajectoires de mortier de 120 mm générées par BALCO présenté dans la partie *(1.5)* de ce document. Pour cela, quatre méthodes d'estimation sont comparées :

- le *Deep ES-KF* présenté dans la partie *(5.4.3)* - *Algorithme 3*,
- le LSTM pré-entraîné pour estimer $O_{ut\ Features}$ (5.57),
- l'ES-KF sans réseaux de neurones présenté dans la partie *(5.4.2)* - *Algorithme 2*,
- un algorithme de *Dead Reckoning* correspondant à la prédiction d'état nominal de l'ES-KF (5.37) - (5.40).

Validation qualitative : analyse d'une trajectoire de mortier de 120 mm

Les figures 5.11 - 5.13 présentent les erreurs de position, de vitesse et d'orientation obtenues par les quatre méthodes mentionnées ci-dessus, pour une trajectoire de mortier de 120 mm d'une portée de 4k m et d'une apogée de 1.5 km.

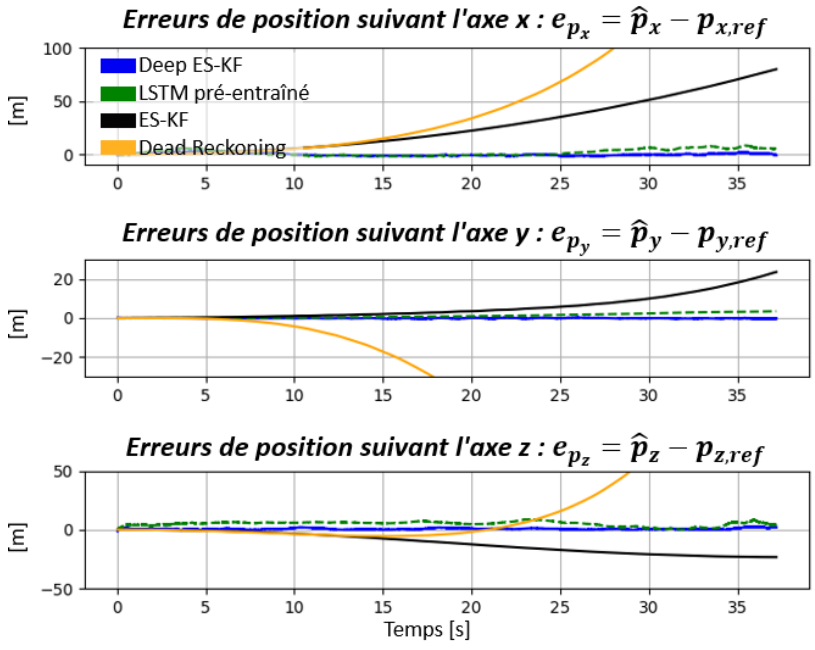

FIGURE 5.11 – Erreurs d'estimation de position pour une trajectoire de mortier : *Deep ES-KF* (bleu), LSTM pré-entraîné (vert), ES-KF (noir), *Dead Reckoning* (jaune).

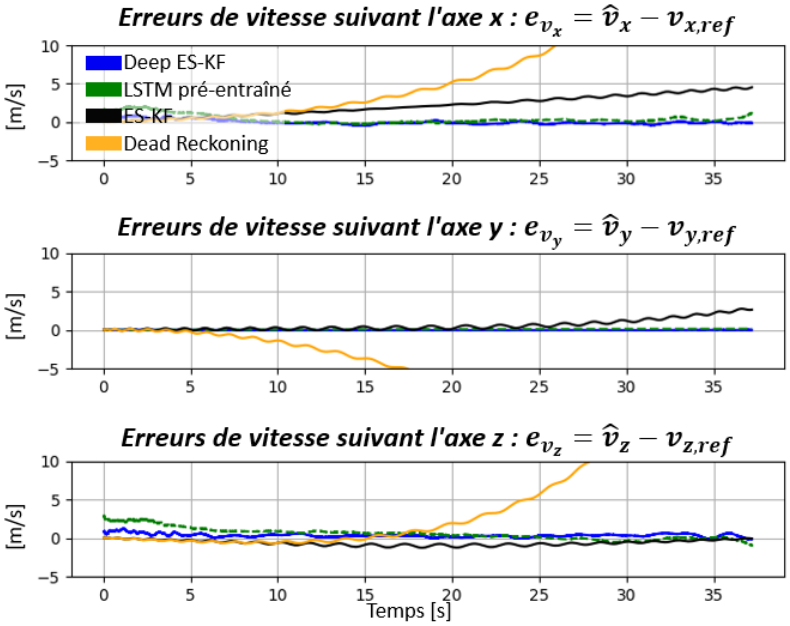

FIGURE 5.12 – Erreurs d'estimation de vitesse pour une trajectoire de mortier : *Deep ES-KF* (bleu), LSTM pré-entraîné (vert), ES-KF (noir), *Dead Reckoning* (jaune).

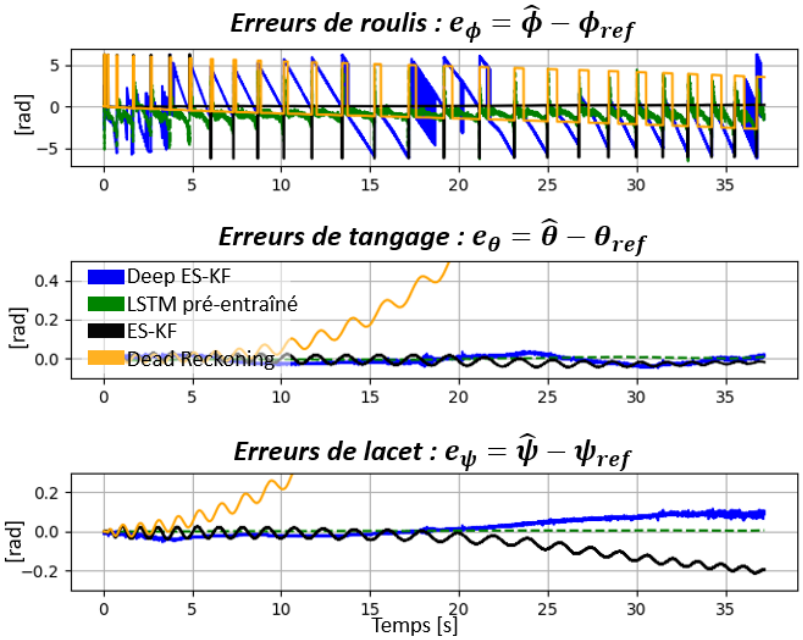

FIGURE 5.13 – Erreurs d'estimation d'orientation pour une trajectoire de mortier : *Deep ES-KF* (bleu), LSTM pré-entraîné (vert), ES-KF (noir), *Dead Reckoning* (jaune).

Les figures 5.11 et 5.12 indiquent que le *Deep ES-KF* (en bleu) et le LSTM pré-entraîné (en vert) sont des solutions précises d'estimation des positions et des vitesses du projectile. L'ES-KF (en noir), moins précis que les solutions basées sur l'IA, présente également de faibles erreurs, contrairement au *Dead Rekoning* (en jaune) qui diverge. Ces erreurs illustrent la précision de l'IMU modélisée par BALCO.
La figure 5.13 révèle que les solutions basées sur l'IA ne parviennent pas à estimer l'angle de roulis du projectile contrairement à l'ES-KF (en noir). Ces mauvaises prédictions, déjà constatées au chapitre précédent, ne peuvent être corrigées adéquatement lors de l'étape de mise à jour afin d'estimer avec précision les angles d'Euler de la munition. De plus, le *Deep ES-KF* (en bleu) dégrade l'estimation de l'angle de roulis par rapport au LSTM pré-entraîné (en vert). Néanmoins, le *Deep ES-KF* et le LSTM pré-entraîné semblent appropriés pour estimer les angles de tangage et de lacet du projectile. Enfin, malgré des résultats cohérents pour l'angle de roulis, le *Dead Reckoning* (en jaune) dévie pour l'estimation des angles de tangage et de lacet.

Évaluation quantitative : analyse sur l'ensemble des données de test

Afin de valider les premières observations formulées dans la partie précédente, le *Deep ES-KF*, le LSTM pré-entraîné, l'ES-KF et le *Dead Reckoning* sont évalués sur les 200 trajectoires de mortier du jeu de données de test selon trois critères d'erreur :

- l'erreur quadratique moyenne *(Root Mean Square Error - RMSE)* évaluée telle que :

$$RMSE = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\left(x_{k,ref} - \widehat{x}_k\right)^2}, \tag{5.59}$$

- l'erreur moyenne absolue *(Mean Absolute Error - MAE)* évaluée telle que :

$$MAE = \frac{1}{N}\sum_{k=1}^{N}\left|x_{k,ref} - \widehat{x}_k\right|, \tag{5.60}$$

- le taux de réussite, évalué comme le nombre de simulations dans le jeu de données où la méthode considérée présente la plus petite RMSE :

$$\mathcal{S}_R = \sum_{k=1}^{N_{sim}} min\left(RMSE_{Deep\ ES-KF}, RMSE_{LSTM}, RMSE_{ES-KF}, RMSE_{DR}\right), \tag{5.61}$$

avec $N_{sim} = 200$ le nombre de simulations du jeu de données de test, $\widehat{x}$ la quantité estimée, x_{ref} la quantité de référence et N le nombre d'échantillons pour une simulation.

La $RMSE$ et la MAE sont évaluées sur l'ensemble du jeu de données de test tel que :

$$\mathcal{C}_{RMSE} = \frac{1}{N_{sim}} \sum_{k=1}^{N_{sim}} RMSE_k, \qquad \mathcal{C}_{MAE} = \frac{1}{N_{sim}} \sum_{k=1}^{N_{sim}} MAE_k. \tag{5.62}$$

Les figures 5.14 - 5.16 présentent $\mathcal{C}_{RMSE}$, $\mathcal{C}_{MAE}$ et $\mathcal{S}_R$ évalués sur 200 trajectoires pour les quatre méthodes d'estimation considérées.

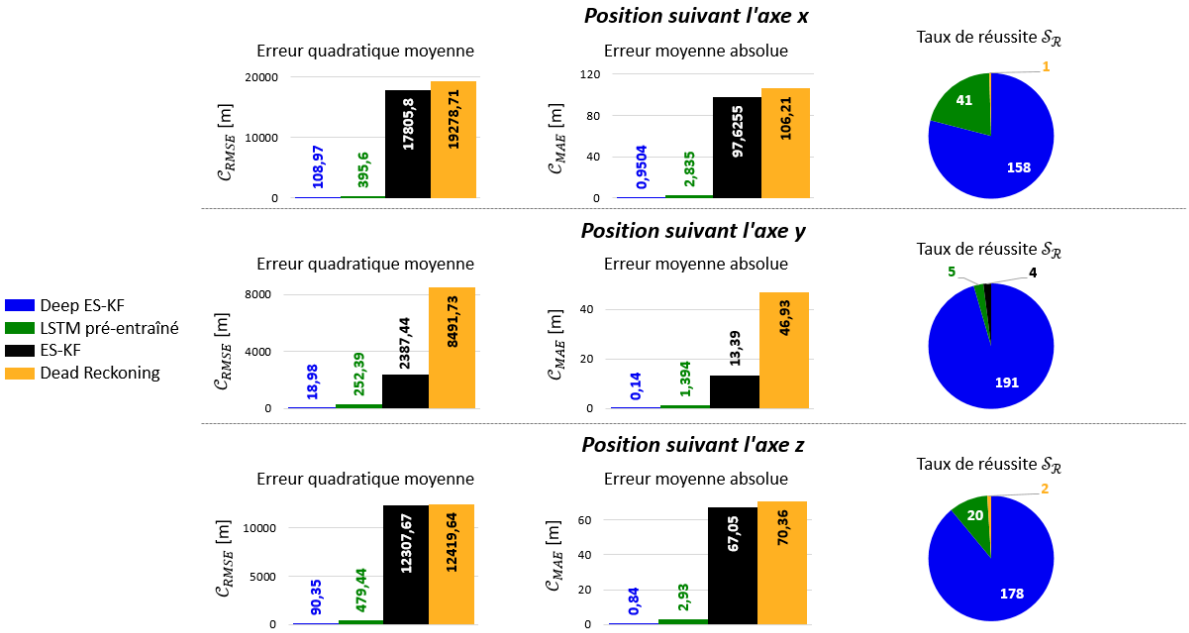

FIGURE 5.14 – Analyse de la précision des estimations des positions : évaluation des critères $\mathcal{C}_{RMSE}$, $\mathcal{C}_{MAE}$ (5.62), $\mathcal{S}_R$ (5.61) obtenus par le *Deep ES-KF* (en bleu), le LSTM pré-entraîné (en vert), l'ES-KF (en noir) et le *Dead Reckoning* (en jaune).

La figure 5.14 illustre l'apport de l'intelligence artificielle pour estimer les positions d'un projectile. En effet, les critères $\mathcal{C}_{RMSE}$, $\mathcal{C}_{MAE}$ et $\mathcal{S}_R$ montrent que le *Deep ES-KF* (en bleu) et le LSTM pré-entraîné (en vert) surpassent l'ES-KF (noir) et le *Dead Reckoning* (en jaune). Plus précisément, les erreurs moyennes de position $\mathcal{C}_{MAE}$ sont inférieures au mètre pour le *Deep ES-KF* (en bleu), autour de quelques mètres pour le LSTM pré-entraîné (en vert) et supérieures à plusieurs dizaines de mètres pour le ES-KF (en noir) et le *Dead Rekoning* (en jaune). Enfin, parmi les deux approches basées sur l'IA et selon les trois critères d'erreur, le *Deep ES-KF* surpasse le LSTM pré-entraîné pour l'estimation des positions.
L'analyse des trois critères d'erreur $\mathcal{C}_{RMSE}$, $\mathcal{C}_{MAE}$ (5.62) et $\mathcal{S}_R$ (5.61) confirme les observa-

tions précédentes rapportées dans la figure 5.11, le *Deep ES-KF* est une approche précise pour estimer les positions d'un mortier de 120 mm. De plus, cette méthode hybride est préférable à une solution basée exclusivement sur l'IA pour l'estimation des positions.

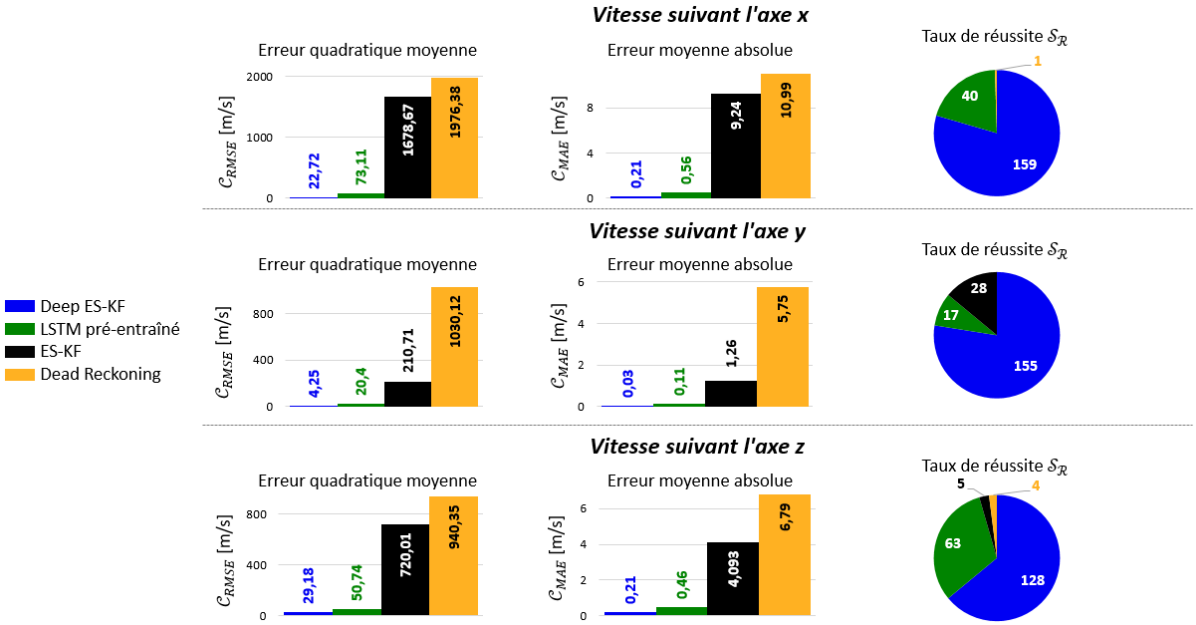

FIGURE 5.15 – Analyse de la précision des estimations des vitesses : évaluation des critères $\mathcal{C}_{RMSE}$, $\mathcal{C}_{MAE}$ (5.62), $\mathcal{S}_R$ (5.61) obtenus par le *Deep ES-KF* (en bleu), le LSTM pré-entraîné (en vert), l'ES-KF (en noir) et le *Dead Reckoning* (en jaune).

D'après la figure 5.15, des observations similaires aux résultats de position peuvent être formulées pour l'estimation de la vitesse du projectile. En effet, le *Deep ES-KF* (en bleu) et le LSTM pré-entraîné (en vert) surpassent l'ES-KF (en noir) et le *Dead Reckoning* (en jaune) suivant les trois axes. De plus, le *Deep ES-KF* présente les erreurs d'estimation de vitesse les plus faibles comparées au trois autres méthodes.

La figure 5.16 révèle que les approches basées sur l'intelligence artificielle sont inadéquates pour estimer l'angle de roulis du projectile comme l'ES-KF (en noir) et le *Dead Reckoning* (en jaune) présentent de plus faibles erreurs. Selon les trois critères d'erreur $\mathcal{C}_{RMSE}$, $\mathcal{C}_{MAE}$ et $\mathcal{S}_R$, l'ES-KF (en noir) reste la solution optimale pour estimer l'angle de roulis du projectile. Le *Dead Reckoning* (en jaune) diverge pour estimer les angles de tangage et de lacet du projectile contrairement au *Deep ES-KF* (en bleu), au LSTM pré-entraîné (en vert) et à l'ES-KF (en noir). Cependant, d'après le taux de réussite $\mathcal{S}_R$, le LSTM pré-entraîné est la solution la plus appropriée pour estimer les angles de tangage et

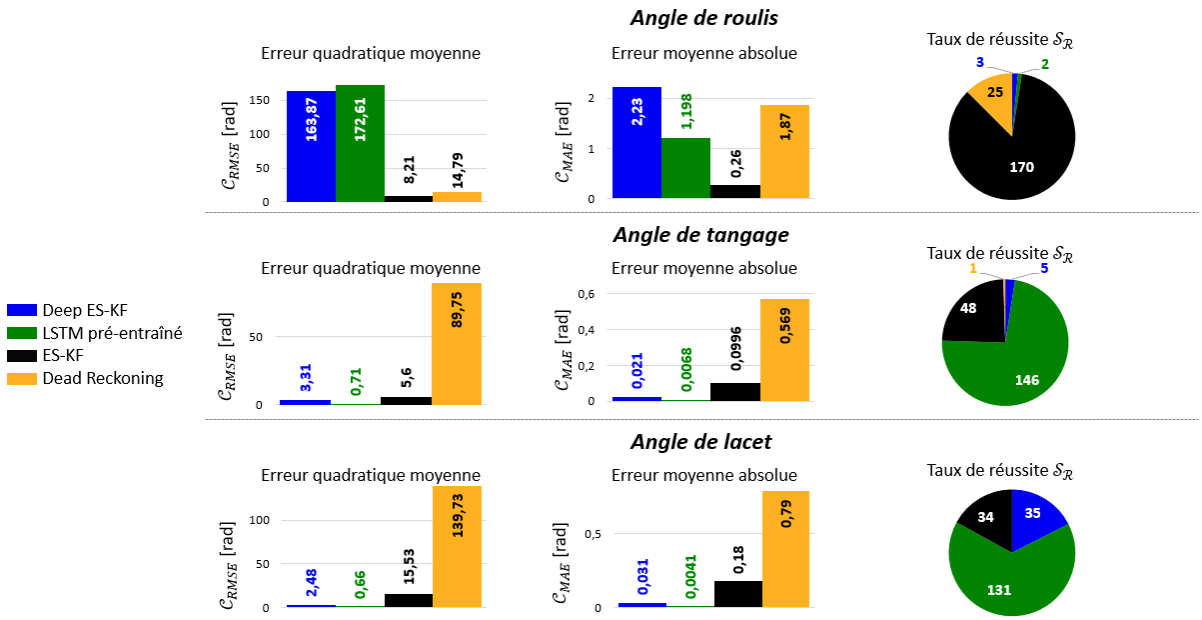

FIGURE 5.16 – Analyse de la précision des estimations des angles d'Euler : évaluation des critères $\mathcal{C}_{RMSE}$, $\mathcal{C}_{MAE}$ (5.62), $\mathcal{S}_R$ (5.61) obtenus par le *Deep ES-KF* (en bleu), le LSTM pré-entraîné (en vert), l'ES-KF (en noir) et le *Dead Reckoning* (en jaune).

de lacet du projectile. L'analyse de l'estimation de l'orientation confirme les observations formulées dans la figure 5.13, le *Deep ES-KF* n'est pas adapté pour estimer l'orientation du projectile. Toutefois, ces conclusions étaient attendues au vu des résultats analysés au chapitre précédent et dans la partie *(5.2)* de ce document.

En résumé, selon les figures 5.11 - 5.16, le *Deep ES-KF* est une solution précise pour estimer les positions et les vitesses d'un mortier de 120 mm, mais ne convient pas à l'estimation de l'orientation. De plus, par rapport aux algorithmes de navigation conventionnels tels que le *Dead Reckoning* ou l'ES-KF, les orientations incorrectes estimées par les méthodes basées sur l'IA n'ont aucune influence sur la précision de l'estimation de la position et de la vitesse.
Ces résultats confirment donc qu'un *Deep Kalman Filter* où l'un des modèles du filtre est remplacé par un réseau de neurones est une solution adéquate pour l'estimation des positions et des vitesses d'un projectile. Cette méthode présente plusieurs avantages. Le premier est que le *Deep Kalman Filter* présenté dans cette partie permet d'estimer la trajectoire d'un projectile dans un environnement sans GNSS en utilisant uniquement le champ magnétique de référence ainsi que l'unité de mesure inertielle (IMU) embarquée et

des connaissances sur les conditions de lancement de la munition. Cette méthode nécessite donc des capteurs à faible coût, adéquate pour la navigation de projectiles. Le second avantage de la solution proposée est que le réseau de neurones permet de modéliser précisément les erreurs de l'IMU et les corrélations entre les capteurs, sans devoir formuler des hypothèses complexes et restreignantes. Le troisième avantage de cette solution est que le LSTM est entraîné à partir de l'IMU, du champ magnétique de référence, des paramètres prévol propres à la munition et d'un vecteur temps. Les données d'entrée du réseau ne dépendent pas des prédictions précédentes. Ainsi, une mauvaise estimation à un instant n'influe pas sur les estimations aux instants suivants. Toutefois, le *Deep Kalman Filter* présenté dans cette partie ne permet pas d'estimer convenablement l'orientation du projectile. Ces observations confirment donc qu'un modèle mathématique est préférable pour l'estimation de l'orientation d'un projectile, comme proposé dans la partie *(5.2)*.
Un *Deep Kalman Filter* où l'un des modèles du filtre est remplacé par un réseau de neurones est une approche valide pour estimer avec précision les positions et les vitesses des projectiles.

5.5 Adaptation d'un filtre de Kalman

Les analyses des *Deep Kalman Filter* précédents montrent que cette méthode est adaptée à l'estimation des positions et des vitesses d'un projectile, mais n'est pas optimale pour évaluer les angles d'Euler. De plus, il s'est avéré qu'une méthode d'apprentissage par transfert est nécessaire afin d'obtenir des résultats optimaux. Afin de répondre au problème d'estimation de l'orientation d'un projectile, et plus précisément à l'estimation de l'angle de roulis, cette partie se concentre sur un *Deep Kalman Filter* pour estimer cet angle. Pour cela, un filtre de Kalman testé en vol est considéré dans le but d'estimer l'angle de roulis d'un projectile de 30 mm caractérisé par un tir tendu. Ce même filtre est ensuite adapté, à l'aide de réseaux de neurones, à un projectile caractérisé par une trajectoire balistique. Le filtre de Kalman considéré est celui employé dans la partie *(3.4.1)* de ce manuscrit à des fins d'optimisation des paramètres de bruit. A l'inverse, cette partie se concentre sur les facultés de l'intelligence artificielle à adapter un modèle mathématique à une dynamique différente.

5.5.1 Filtre de Kalman testé en vol pour l'estimation du roulis

Les travaux présentés dans cette partie se basent sur le filtre de Kalman Étendu présenté dans [9] dans le cadre du projet GSP *(Guided Supersonic Projectile)*. L'objectif de ce projet est de modifier, en vol, la trajectoire d'un projectile supersonique à ailettes de 30 mm à l'aide d'actionneurs pyrotechniques. Ce projectile est caractérisé par un tir tendu et une vitesse de rotation faible. En d'autres termes, il s'agit d'éjecter un actionneur à impulsion latérale de la munition à un moment spécifique afin de modifier la trajectoire du projectile. Pour cela, la connaissance de l'angle de roulis est nécessaire afin de modifier la trajectoire dans la direction désirée. A cette fin, un filtre de Kalman Étendu *(Extended Kalman Filter - EKF)* est développé pour estimer avec précision l'angle et la vitesse de roulis du projectile à partir des deux magnétomètres radiaux embarqués dans la munition.

Présentation générale du filtre

Le fonctionnement du filtre de Kalman Étendu mis en place pour estimer l'angle et la vitesse de roulis du projectile à partir des deux magnétomètres radiaux embarqués est présenté dans la figure 5.17.

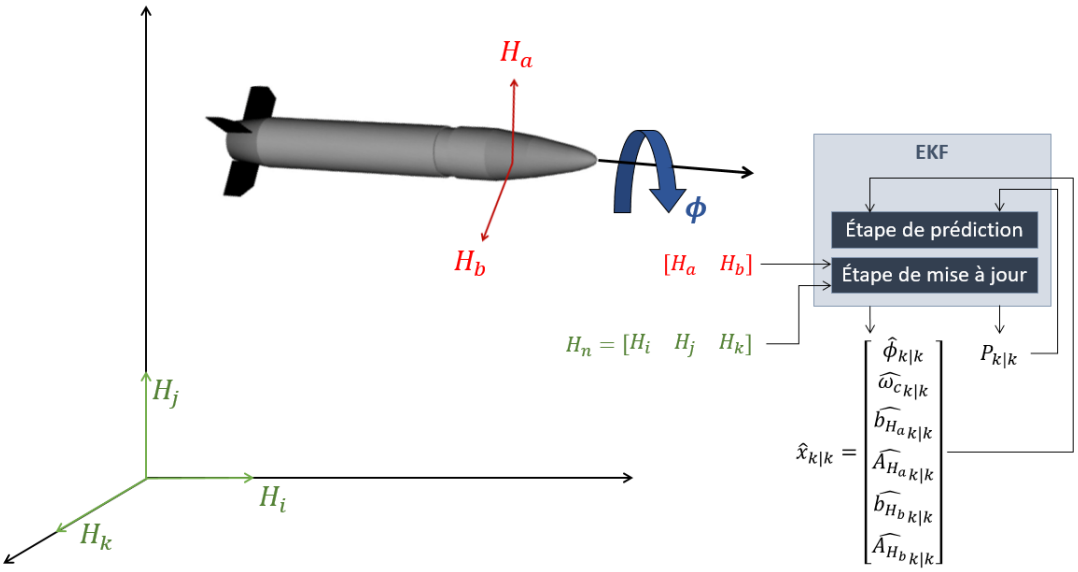

FIGURE 5.17 – Principe de fonctionnement du filtre de Kalman Étendu pour estimer l'angle et la vitesse de roulis d'un projectile de 30 mm à partir de deux magnétomètres radiaux.

Les mesures des deux magnétomètres radiaux embarqués dans le projectile de 30 mm sont modélisées comme suit :

$$\tilde{H}_a = A_{H_a} H_a + b_{H_a} + W_{H_a}, \quad \tilde{H}_b = A_{H_b} H_b + b_{H_b} + W_{H_b} \tag{5.63}$$

avec $\tilde{H}_a$ et $\tilde{H}_b$ les modèles des mesures, H_a et H_b les mesures des capteurs, A_{H_a} et A_{H_b} les amplitudes modélisées des deux mesures, b_{H_a} et b_{H_b} les biais respectifs, et W_{H_a} et W_{H_b} les bruits des mesures supposés gaussiens.

A partir des mesures des deux magnétomètres radiaux embarqués H_a et H_b, et la connaissance du champ magnétique de référence $H_n = \begin{bmatrix} H_i & H_j & H_k \end{bmatrix}^T$ dans le repère local de navigation $\boldsymbol{n}$, le filtre de Kalman Étendu vise à estimer :

- l'angle de roulis ϕ et la vitesse de roulis ω_c du projectile dans le repère local de navigation $\boldsymbol{n}$,
- les amplitudes A_{H_a} et A_{H_b} des mesures des deux magnétomètres H_a et H_b embarqués dans la munition,
- les biais b_{H_a} et b_{H_b} des mesures des deux magnétomètres H_a et H_b.

L'EKF estime donc, à partir des mesures H_a et H_b, et de la connaissance du champ magnétique de référence H_n, les états suivants :

$$x = \begin{bmatrix} \phi^T & \omega_c^T & b_{H_a}^T & A_{H_a}^T & b_{H_b}^T & A_{H_b}^T \end{bmatrix}^T \in \mathbb{R}^{6\times 1}. \tag{5.64}$$

Le projectile de 30 mm dans lequel ce filtre est implémenté est caractérisé par un tir tendu. En d'autres termes, les variations des angles de tangage et de lacet de cette munition sont négligeables ($\theta \approx \psi \approx 0$). Donc, les dynamiques de l'angle de roulis et de la vitesse de roulis de ce projectile sont :

$$\dot{\phi} = \omega_c + W_\phi, \qquad \dot{\omega}_c = W_{\omega_c} \tag{5.65}$$

avec W_ϕ et W_{ω_c} des bruit blancs supposés gaussiens.

Les dynamiques des biais et des amplitudes des magnétomètres sont caractérisées par des mouvements browniens tels que :

$$\dot{A}_{H_a} = W_{A_{H_a}}, \quad \dot{b}_{H_a} = W_{b_{H_a}}, \quad \dot{A}_{H_b} = W_{A_{H_b}}, \quad \dot{b}_{H_b} = W_{b_{H_b}} \tag{5.66}$$

avec $W_{A_{H_a}}$, $W_{b_{H_a}}$, $W_{A_{H_b}}$ et $W_{b_{H_b}}$ des bruit blancs supposés gaussiens.

Les observations considérées pour mettre à jour le filtre sont l'estimation des mesures des magnétomètres à partir du champ magnétique de référence et de l'angle de roulis estimé tel que le modèle de mesure soit :

$$\begin{bmatrix} \widehat{H_a} \\ \widehat{H_b} \end{bmatrix} = \begin{bmatrix} A_{H_a} H_{env} sin(\phi + \lambda) + b_{H_a} + W_{H_a} \\ A_{H_b} H_{env} cos(\phi + \lambda) + b_{H_b} + W_{H_b} \end{bmatrix} \tag{5.67}$$

avec W_{H_a} et W_{H_b} les bruits des mesures, et H_{env} et λ des variables fonctions du champ magnétique de référence $H_n = \begin{bmatrix} H_i & H_j & H_k \end{bmatrix}^T$ et des angles d'Euler de la munition. Pour rappel, le projectile de 30 mm pour lequel ce filtre est développé est caractérisé par un tir tendu, donc $\theta \approx \psi \approx 0$. Ainsi, les variables H_{env} et λ sont définies telles que :

$$H_{env} = \sqrt{H_1^2 + H_2^2} \quad \text{avec} \quad \begin{cases} H_1 = -sin(\theta) H_i + cos(\theta) H_j + \psi sin(\theta) H_k \approx H_j \\ H_2 = \psi H_i + H_k \approx H_k \end{cases} \tag{5.68}$$

$$\lambda \quad \text{tel que} \quad \begin{cases} cos(\lambda) = \frac{H_2}{H_{env}} \\ sin(\lambda) = \frac{H_1}{H_{env}} \end{cases} \tag{5.69}$$

Étape de prédiction

La prédiction de l'état x est donnée par l'équation suivante :

$$\widehat{x}_{k|k-1} = A_d \widehat{x}_{k-1|k-1} \tag{5.70}$$

avec $A_d = (2I_{6\times6} - A_t \Delta t)^{-1}(2I_{6\times6} - A_t \Delta t)$, avec $A_t = \begin{bmatrix} 0 & 1 & 0 & & & \\ & & & 0 & 0 & 0 \\ & & & 0_{6\times6} & & \end{bmatrix}$ et avec Δt la période d'échantillonnage des magnétomètres.

La prédiction de la covariance de l'erreur est donnée par :

$$P_{k|k-1} = A_d P_{k-1|k-1} A_d^T + Q_k \tag{5.71}$$

avec $Q_k = cov\left(W_\phi, W_{\omega_c}, W_{b_{H_a}}, W_{A_{H_a}}, W_{b_{H_b}}, W_{A_{H_b}}\right)$ la matrice de covariance du bruit de modèle.

Étape de mise à jour

L'état $\widehat{x}_{k|k-1}$ et la covariance $P_{k|k-1}$ prédite sont mis à jour par l'estimation des mesures des magnétomètres H_a et H_b et à partir de la connaissance du champ magnétique de

référence (5.67). Ainsi, l'étape de mise à jour est donnée par les équations suivantes :

$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1} \tag{5.72}$$

$$\widehat{x}_{k|k} = \widehat{x}_{k|k-1} + K_k \left(\begin{bmatrix} H_a \\ H_b \end{bmatrix} - \begin{bmatrix} \widehat{A}_{H_{a_{k|k-1}}} H_{env} sin(\widehat{\phi}_{k|k-1} + \lambda) + \widehat{b}_{H_{a_{k|k-1}}} \\ \widehat{A}_{H_{b_{k|k-1}}} H_{env} cos(\widehat{\phi}_{k|k-1} + \lambda) + \widehat{b}_{H_{b_{k|k-1}}} \end{bmatrix} \right) \tag{5.73}$$

$$P_{k|k} = (I - K_k H_k) P_{k|k-1} \tag{5.74}$$

avec $R_k = cov(W_{H_a}, W_{H_b})$ la matrice de covariance du bruit de mesure et avec H_k la matrice d'observation déterminée par la linéarisation du modèle d'observation (5.67) par rapport aux états prédits $\widehat{x}_{k|k-1}$ tels que :

$$H_k = \begin{bmatrix} \widehat{A}_{H_{a_{k|k-1}}} H_{env} cos(\widehat{\phi}_{k|k-1} + \lambda) & 0 & 1 & H_{env} sin(\widehat{\phi}_{k|k-1} + \lambda) & 0 & 0 \\ -\widehat{A}_{H_{b_{k|k-1}}} H_{env} sin(\widehat{\phi}_{k|k-1} + \lambda) & 0 & 0 & 0 & 1 & H_{env} cos(\widehat{\phi}_{k|k-1} + \lambda) \end{bmatrix} \tag{5.75}$$

5.5.2 *Deep EKF*

Le filtre de Kalman Étendu présenté dans la partie *(5.5.1)* est valide dans le cas d'un tir tendu qui suppose des variations infinitésimales de l'angle de tangage et de lacet du projectile de 30 mm. A partir de ce modèle de filtre, l'objectif est d'utiliser des réseaux de neurones afin de l'adapter à une trajectoire balistique de mortier de 120 mm.

Dans le cas d'une trajectoire balistique, les angles de tangage et de lacet ont des variations non négligeables donc le modèle de mesure (5.67) et la matrice d'observation (5.75) ne sont plus valides. L'idée initiale est donc de remplacer le modèle de mesure et la matrice d'observation par des réseaux de neurones. Comme présenté dans la figure 5.18, les mesures estimées $\widehat{H_a}$ et $\widehat{H_b}$ sont déterminées par un premier LSTM et la matrice H_k est déterminée par un second LSTM pour estimer l'angle et la vitesse de roulis d'un mortier de 120 mm. Comme dans le cas de l'EKF traditionnel, seules les mesures des deux magnétomètres radiaux H_a et H_b ainsi que le champ magnétique $H_n = \begin{bmatrix} H_i & H_j & H_k \end{bmatrix}^T$ dans le repère local de navigation sont considérés.

Afin d'estimer les deux modèles par des réseaux de neurones, les deux LSTM, notés respectivement $LSTM_Y$ pour l'estimation du modèle de mesure et $LSTM_H$ pour l'estimation de la matrice d'observation, sont tous deux entraînés séparément par des méthodes d'apprentissage par transfert.

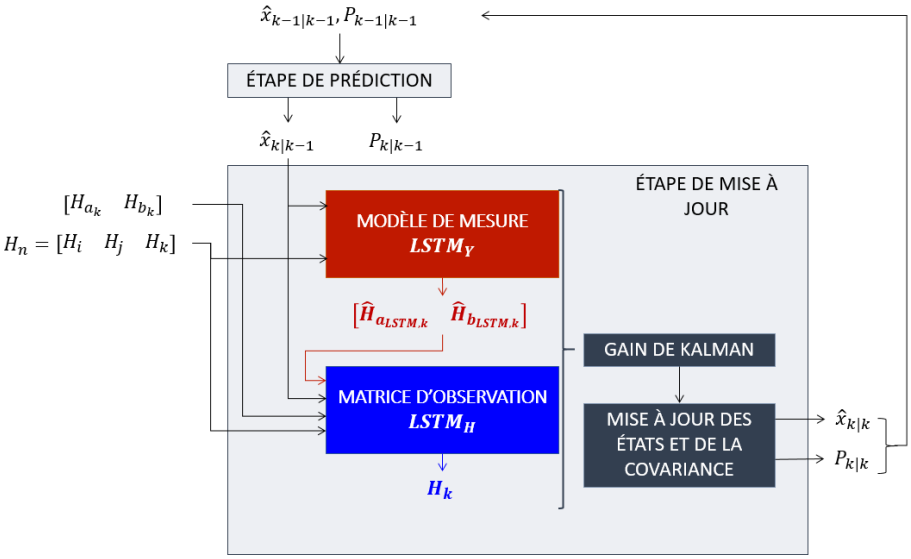

FIGURE 5.18 – Principe de fonctionnement du filtre de Kalman Étendu adapté à une trajectoire balistique pour estimer l'angle et la vitesse de roulis d'un projectile à partir de deux magnétomètres radiaux dans le cas où le modèle de mesure et la matrice d'observation sont déterminés par des LSTM.

5.5.3 Estimation du modèle de mesure

D'après le filtre de Kalman Étendu traditionnel présenté dans la partie *(5.5.1)*, les mesures estimées $\widehat{H}_a$ et $\widehat{H}_b$ sont déterminées à partir des amplitudes $\widehat{A}_{H_{a_{k|k-1}}}$ et $\widehat{A}_{H_{b_{k|k-1}}}$, des biais $\widehat{b}_{H_{a_{k|k-1}}}$ et $\widehat{b}_{H_{b_{k|k-1}}}$, et de l'angle de roulis $\widehat{\phi}_{k|k-1}$ prédit, ainsi qu'à partir des deux paramètres λ et H_{env}. Ce modèle de mesure est déterminé en supposant de faibles variations des angles de tangage et de lacet. Or, dans le cas d'une trajectoire balistique, cette hypothèse n'est plus valide. Pour cela, un *Deep EKF* est proposé afin de déterminer les mesures estimées $\widehat{H}_a$ et $\widehat{H}_b$ par un LSTM.

Pré-entraînement de $LSTM_Y$

Comme illustré dans la figure 5.19, le $LSTM_Y$ est pré-entraîné pour estimer les modèles des mesures, notés respectivement $\widehat{H}_{a_{LSTM}}$ et $\widehat{H}_{b_{LSTM}}$, à partir de 10 caractéristiques d'entrée qui sont :

- $\widehat{x}_{k|k-1} = \left[\widehat{\phi}^T_{k|k-1} \quad \widehat{\omega}^T_{c_{k|k-1}} \quad \widehat{b}^T_{H_{a_{k|k-1}}} \quad \widehat{A}^T_{H_{a_{k|k-1}}} \quad \widehat{b}^T_{H_{a_{k|k-1}}} \quad \widehat{A}^T_{H_{b_{k|k-1}}}\right]^T \in \mathbb{R}^6$ les états prédits par l'EKF traditionnel. Pour cela, les états $\widehat{x}_{k|k-1}$ sont déterminés en amont du pré-entraînement du $LSTM_Y$, par l'EKF traditionnel présenté dans la partie

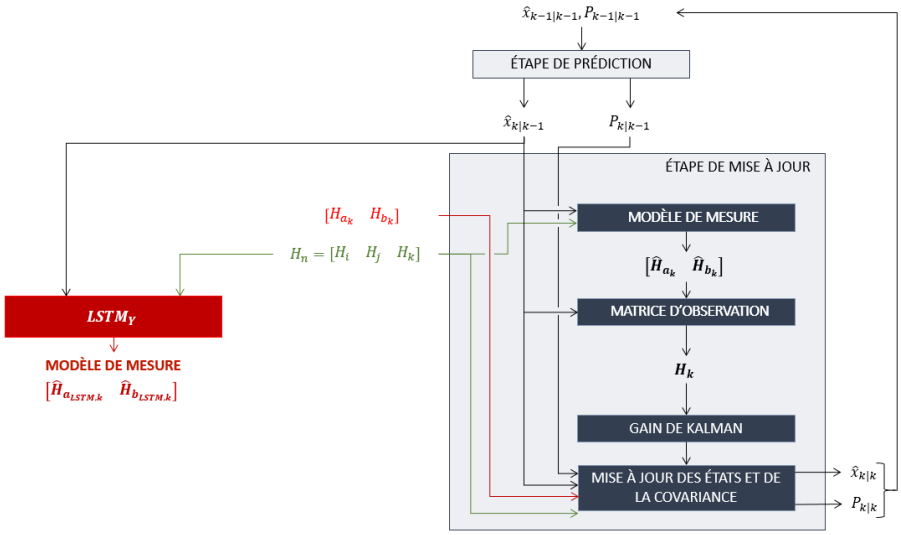

FIGURE 5.19 – Pré-entraînement de $LSTM_Y$ pour déterminer les mesures estimées $\widehat{H}_{a_{LSTM}}$ et $\widehat{H}_{b_{LSTM}}$ à partir de l'état prédit par l'EKF traditionnel et du champ magnétique de référence.

(5.5.1). Chaque état prédit $\widehat{x}_{k|k-1}$ est obtenu notamment à partir de l'état estimé à l'instant précédent $\widehat{x}_{k-1|k-1}$ et mis à jour traditionnellement d'après les équations (5.72) - (5.74).

- $H_n = \begin{bmatrix} H_i & H_j & H_k \end{bmatrix}^T \in \mathbb{R}^3$ le champ magnétique de référence dans le repère local de navigation $\boldsymbol{n}$, constant pendant la durée de vol du projectile.
- $\mathcal{T} = k\Delta_t \in \mathbb{R}^1$ le vecteur temps avec k le pas de temps considéré et Δ_t la période d'échantillonnage des magnétomètres.

Le pré-entraînement du LSTM est nécessaire comme les données d'entrée dépendent des prédictions précédentes. L'objectif est donc d'exploiter dans un premier temps les prédictions de l'EKF afin que le LSTM égale les estimations $\widehat{H}_a$ et $\widehat{H}_b$ obtenues avec des modèles mathématiques. Dans un second temps, il s'agit d'affiner les paramètres appris pour estimer précisément l'angle et la vitesse de roulis du mortier de 120 mm.

Le $LSTM_Y$ est entraîné sur 100 trajectoires de mortier de 120 mm, validé sur 50 trajectoires et testé sur 30 trajectoires. La perte entre les estimations $\widehat{H}_{a_{LSTM}}$ et $\widehat{H}_{b_{LSTM}}$ et les mesures vraies des magnétomètres H_a et H_b est évaluée avec l'erreur quadratique moyenne *(Mean Squared Error - MSE)* :

$$MSE = \frac{1}{N}\sum_{k=1}^{N}(x_{ref_k} - \widehat{x}_k)^2. \tag{5.76}$$

Les figures 5.20 et 5.21 présentent les estimations $\widehat{H}_{a_{LSTM}}$ et $\widehat{H}_{b_{LSTM}}$ du *Deep EKF* ajusté par le $LSTM_Y$ pré-entraîné (en bleu), ainsi que les estimations $\widehat{H}_a$ et $\widehat{H}_b$ du filtre de Kalman traditionnel (en vert) présenté dans la partie *(5.5.1)* pour une trajectoire de mortier de 120 mm.

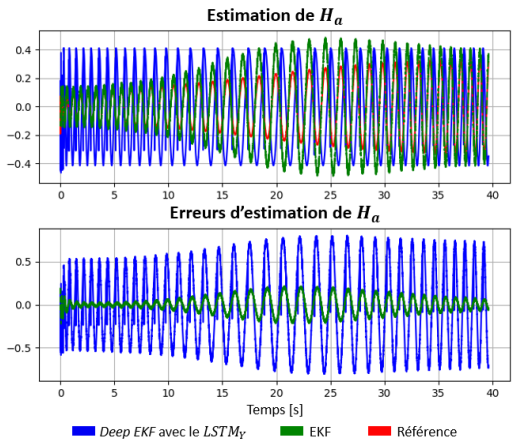

FIGURE 5.20 – Estimation et erreurs d'estimation de la mesure H_a pour une trajectoire de mortier de 120 mm : le *Deep EKF* adapté par le $LSTM_Y$ pré-entraîné (en bleu), l'EKF traditionnel (en vert) et la mesure H_a du magnétomètre (en rouge).

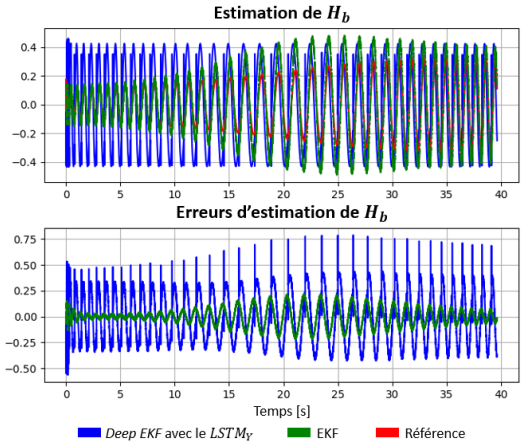

FIGURE 5.21 – Estimation et erreurs d'estimation de la mesure H_b pour une trajectoire de mortier de 120 mm : le *Deep EKF* adapté par le $LSTM_Y$ pré-entraîné (en bleu), l'EKF traditionnel (en vert) et la mesure H_b du magnétomètre (en rouge)

D'après les figures 5.20 et 5.21, le $LSTM_Y$ ne parvient pas à égaler les estimations $\widehat{H}_a$ et $\widehat{H}_b$ de l'EKF, bien que ce filtre ne soit pas adapté à une trajectoire balistique. Une explication de ces mauvais résultats est que les mesures des magnétomètres radiaux sont des signaux périodiques. Or, d'après la littérature [128], [129], les réseaux de neurones récurrents ne sont pas adaptés à l'estimation de fonction périodique comme pour une même séquence en entrée du $LSTM$, le réseau envisage plusieurs solutions.
Afin de résoudre ce problème tout en bénéficiant des propriétés de prédiction d'un LSTM, une première solution est de supposer les fonctions $cos(.)$ et $sin(.)$ connues. Ces fonctions interviennent dans le modèle de mesure définit par l'équation (5.67). Toutefois, les résultats préliminaires obtenus ont indiqué que la connaissance des fonctions $cos(.)$ et $sin(.)$ n'est pas suffisante pour que le $LSTM_Y$ produise des résultats convaincants et correspondants à la périodicité des mesures des magnétomètres. Ainsi, l'estimation des mesures $\widehat{H}_a$

et $\widehat{H}_b$ est modifiée de sorte que d'après les données d'entrée $I_{n\ Features} = (\widehat{x}_{k|k-1}, H_n, \mathcal{T}) \in \mathbb{R}^{10}$ présentées précédemment, le $LSTM_Y$ estime le vecteur suivant :

$$O_{ut\ Features} = \begin{bmatrix} X_1 & X_2 & X_3 & X_4 & X_5 & X_6 \end{bmatrix} \tag{5.77}$$

Les mesures des magnétomètres sont ensuite déterminées par les équations suivantes :

$$\begin{bmatrix} \widehat{H}_{a_{LSTM}} \\ \widehat{H}_{b_{LSTM}} \end{bmatrix} = \begin{bmatrix} (X_1 + \widehat{A}_{H_a}) \times H_{env} \times sin(\lambda + \widehat{\phi} + X_2) + (X_3 + \widehat{b}_{H_a}) \\ (X_4 + \widehat{A}_{H_b}) \times H_{env} \times cos(\lambda + \widehat{\phi} + X_5) + (X_6 + \widehat{b}_{H_b}) \end{bmatrix} \tag{5.78}$$

Les figures 5.22 et 5.23 présentent les estimations $\widehat{H}_{a_{LSTM}}$ et $\widehat{H}_{b_{LSTM}}$ du *Deep EKF* ajusté par le $LSTM_Y$ pré-entraîné conformément à l'équation (5.78) (en bleu) ainsi que les estimations $\widehat{H}_a$ et $\widehat{H}_b$ du filtre de Kalman traditionnel (en vert) présenté dans la partie *(5.5.1)* pour une trajectoire de mortier de 120 mm.

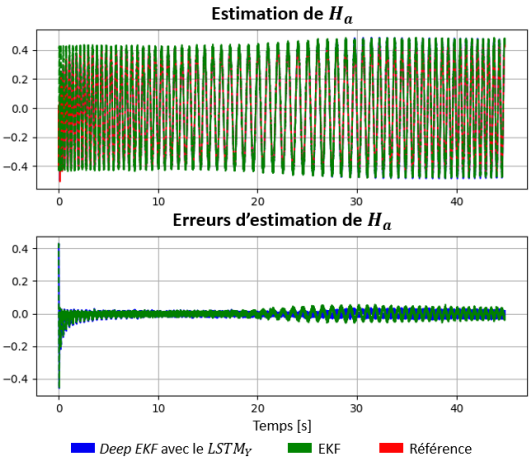

FIGURE 5.22 – Estimation et erreurs d'estimation de la mesure H_a pour une trajectoire de mortier de 120 mm : le *Deep EKF* adapté par le $LSTM_Y$ pré-entraîné conformément à l'équation (5.78) (en bleu), l'EKF traditionnel (en vert) et la mesure H_a du magnétomètre (en rouge).

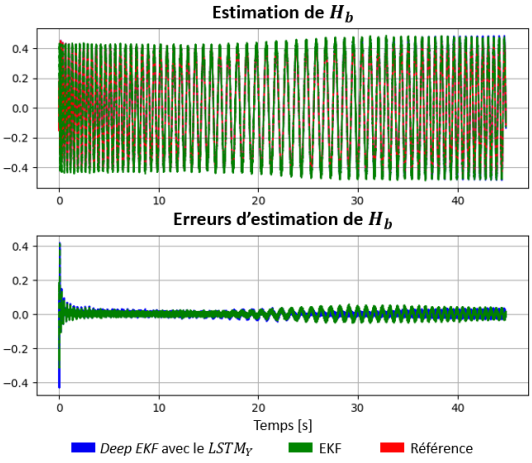

FIGURE 5.23 – Estimation et erreurs d'estimation de la mesure H_b pour une trajectoire de mortier de 120 mm : le *Deep EKF* adapté par le $LSTM_Y$ pré-entraîné conformément à l'équation (5.78) (en bleu), l'EKF traditionnel (en vert) et la mesure H_b du magnétomètre (en rouge).

Les figures 5.22 et 5.23 indiquent que le LSTM parvient à estimer de façon cohérente les mesures H_a et H_b des deux magnétomètres radiaux à partir des prédictions de l'EKF.

Ainsi, la connaissance de l'état prédit, des paramètres H_{env} et λ et des fonctions $cos(.)$ et $sin(.)$ est nécessaire pour évaluer le modèle de mesure.

Deep EKF : modèle de mesure estimé par $LSTM_Y$

D'après les résultats présentés dans les figures 5.22 et 5.23, le $LSTM_Y$ pré-entraîné à estimer $O_{ut\ Features}$ (5.77) permet de déterminer les mesures des deux magnétomètres conformément à l'équation (5.78). Il s'agit à présent d'exploiter les connaissances apprises lors du pré-entraînement de $LSTM_Y$ afin d'évaluer $\widehat{H}_{a_{LSTM}}$ et $\widehat{H}_{b_{LSTM}}$ à partir des estimations précédentes obtenues lors de l'étape de mise à jour du filtre.

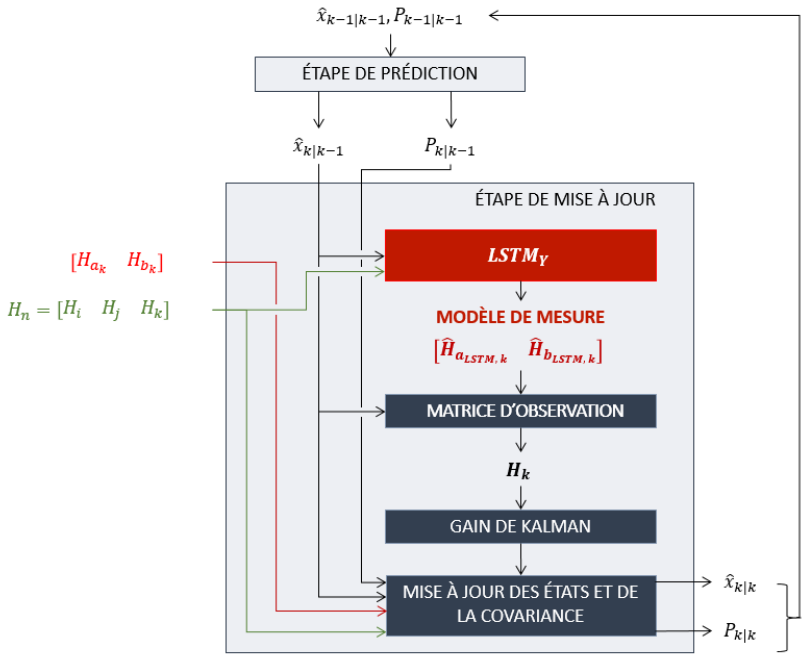

FIGURE 5.24 – *Deep EKF* : estimation du modèle de mesure de l'EKF par le $LSTM_Y$.

Comme présenté dans la figure 5.24, le $LSTM_Y$ est intégré au filtre de Kalman afin de déterminer les mesures H_a et H_b des deux magnétomètres. Pour cela, le filtre de Kalman est d'abord initialisé traditionnellement. Puis, l'état $\hat{x}_{k|k-1}$ et la matrice de covariance de l'erreur $P_{k|k-1}$ sont prédits conformément aux équations (5.70) et (5.71). Ensuite, ces prédictions sont employées comme données d'entrée du $LSTM_Y$, de sorte que les caractéristiques d'entrée soient $I_{n\ Features} = (\hat{x}_{k|k-1}, H_n, \mathcal{T}) \in \mathbb{R}^{10}$. A partir de ces prédictions, le $LSTM_Y$ estime le vecteur $O_{ut\ Features}$ (5.77) afin de déterminer les mesures des magnétomètres $\widehat{H}_{a_{LSTM}}$ et $\widehat{H}_{b_{LSTM}}$ conformément à l'équation (5.78). La connaissance des

mesures estimées par le LSTM et les mesures vraies H_a et H_b permet de mettre à jour l'état $\widehat{x}_{k|k}$ et la matrice de covariance de l'erreur $P_{k|k}$ conformément aux équations de l'EKF traditionnel (5.74). Le *Deep EKF* présenté dans la figure 5.24 est entraîné suivant les mêmes caractéristiques que le $LSTM_Y$ pré-entraîné présenté dans la partie *(5.5.3)* de ce document.

Les figures 5.25 et 5.26 présentent les résultats d'estimation des mesures des deux magnétomètres par le *Deep EKF* illustré dans la figure 5.24 (en bleu) et par l'EKF traditionnel (en vert) pour une trajectoire de mortier de 120 mm.

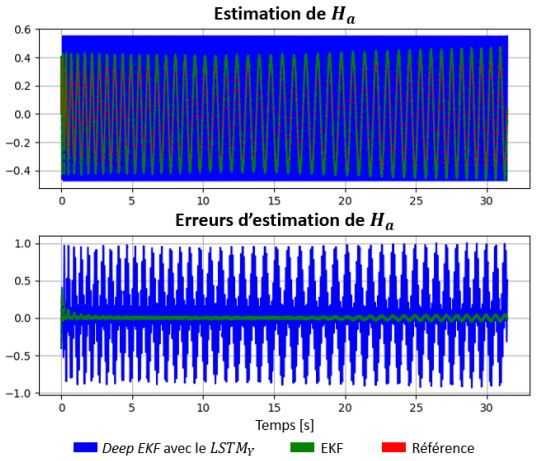

FIGURE 5.25 – Estimation et erreurs d'estimation de la mesure H_a pour une trajectoire de mortier de 120 mm : le *Deep EKF* adapté par le $LSTM_Y$ (en bleu), l'EKF traditionnel (en vert) et la mesure H_a du magnétomètre (en rouge).

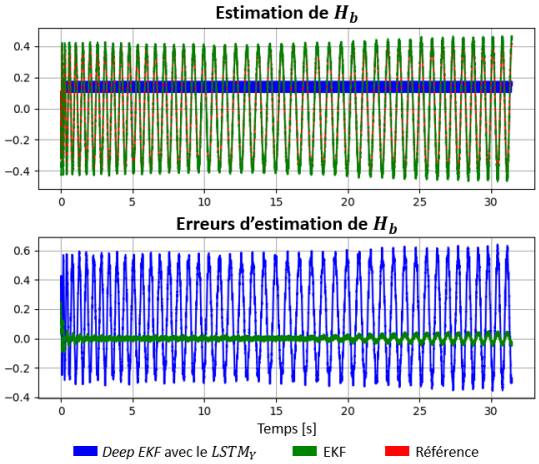

FIGURE 5.26 – Estimation et erreurs d'estimation de la mesure H_b pour une trajectoire de mortier de 120 mm : le *Deep EKF* adapté par le $LSTM_Y$ (en bleu), l'EKF traditionnel (en vert) et la mesure H_b du magnétomètre (en rouge).

Les figures 5.25 et 5.26 montrent que contrairement aux résultats du pré-entraînement, le $LSTM_Y$ ne parvient pas à capturer la dynamique des magnétomètres. Ainsi, les mauvaises prédictions de $\widehat{H}_{a_{LSTM}}$ et $\widehat{H}_{b_{LSTM}}$ par le $LSTM_Y$ impactent de manière significative les estimations obtenues à l'étape de mise à jour de l'EKF. Ces mauvaises estimations influent ensuite sur les prédictions successives comme ces données alimentent le $LSTM_Y$. Ces observations concordent avec celles formulées dans la partie *(5.3)* de ce document. L'utilisation des prédictions antérieures du LSTM comme données d'entrée ne permet pas de résoudre le problème d'estimation, bien que le $LSTM_Y$ ait été préalablement pré-entraîné.

5.5.4 Estimation du modèle d'observation

D'après le filtre de Kalman Étendu traditionnel présenté dans la partie *(5.5.1)* de ce document, la matrice d'observation H_k (5.75) est déterminée à partir des amplitudes $\widehat{A}_{H_{a_{k|k-1}}}$ et $\widehat{A}_{H_{b_{k|k-1}}}$, des biais $\widehat{b}_{H_{a_{k|k-1}}}$ et $\widehat{b}_{H_{b_{k|k-1}}}$, et de l'angle de roulis $\widehat{\phi}_{k|k-1}$ prédit, ainsi qu'à partir des deux paramètres λ et H_{env}. La dérivation du modèle de mesure (5.67) par rapport aux états prédits $\widehat{x}_{k|k-1}$ permet d'identifier la matrice d'observation H_k. Dans le cas d'une trajectoire balistique, ce modèle n'est plus valide du fait des variations non négligeables des angles de tangage et de lacet.
Comme présenté dans la figure 5.27, la matrice d'observation H_k est déterminée par un réseau $LSTM_H$ afin de tenir compte des variations des angles de tangage et de lacet.

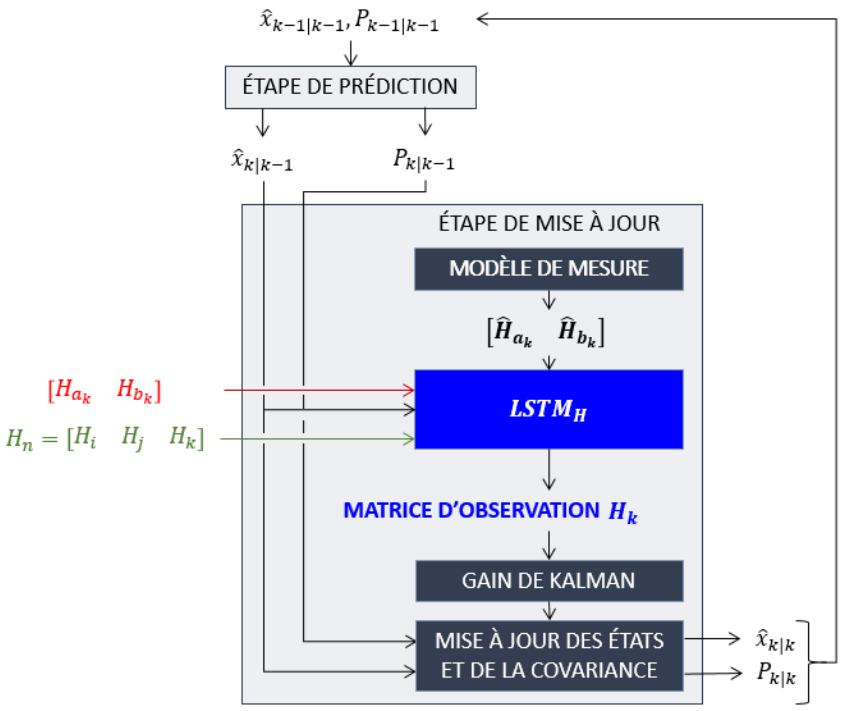

FIGURE 5.27 – *Deep EKF* : estimation du modèle d'observation de l'EKF par le $LSTM_H$.

Le $LSTM_H$ est entraîné à partir de 14 caractéristiques d'entrée qui sont :

- $\widehat{x}_{k|k-1} = \left[\widehat{\phi}^T_{k|k-1} \quad \widehat{\omega}^T_{c_{k|k-1}} \quad \widehat{b}^T_{H_{a_{k|k-1}}} \quad \widehat{A}^T_{H_{a_{k|k-1}}} \quad \widehat{b}^T_{H_{a_{k|k-1}}} \quad \widehat{A}^T_{H_{b_{k|k-1}}}\right]^T \in \mathbb{R}^6$ les états prédits par l'EKF lors de l'étape de prédiction.
- $H_n = \left[H_i \quad H_j \quad H_k\right]^T \in \mathbb{R}^3$ le champ magnétique de référence dans le repère local de navigation $\boldsymbol{n}$, constant pendant la durée de vol du projectile.

- $y_k = \begin{bmatrix} H_{a_k} & H_{b_k} \end{bmatrix} \in \mathbb{R}^2$ les mesures vraies des deux magnétomètres embarqués.
- $\widehat{y}_k = \begin{bmatrix} \widehat{H}_{a_k} & \widehat{H}_{b_k} \end{bmatrix} \in \mathbb{R}^2$ les mesures des deux magnétomètres, estimées à partir des états prédits de l'EKF et définies par l'équation (5.67).
- $\mathcal{T} = k\Delta_t \in \mathbb{R}^1$ le vecteur temps avec k le pas de temps considéré et Δ_t la période d'échantillonnage des magnétomètres.

Comme pour le $LSTM_Y$, le $LSTM_H$ est pré-entraîné à partir des prédictions de l'EKF traditionnel. Pour cela, les états $\widehat{x}_{k|k-1}$ et les modèles de mesures $\widehat{y}_k$ sont déterminés en amont du pré-entraînement du réseau, par les modèles mathématiques de l'EKF traditionnel présenté dans la partie *(5.5.1)*.

Le $LSTM_H$ est entraîné sur 100 trajectoires de mortier de 120 mm, validé sur 10 trajectoires et testé sur 10 trajectoires. La perte entre les estimations $\widehat{x}_{k|k}$ évaluée à partir de la matrice d'observation déterminée par le $LSTM_H$ et les données de référence x_k est évaluée avec l'erreur quadratique moyenne *(Mean Squared Error - MSE)* définie par l'équation (5.76).

Les figures 5.28 et 5.29 présentent les résultats d'estimation de l'angle et de la vitesse de roulis par le *Deep EKF* illustré dans la figure 5.27 (en bleu) et par l'EKF traditionnel (en vert) pour une trajectoire de mortier de 120 mm.

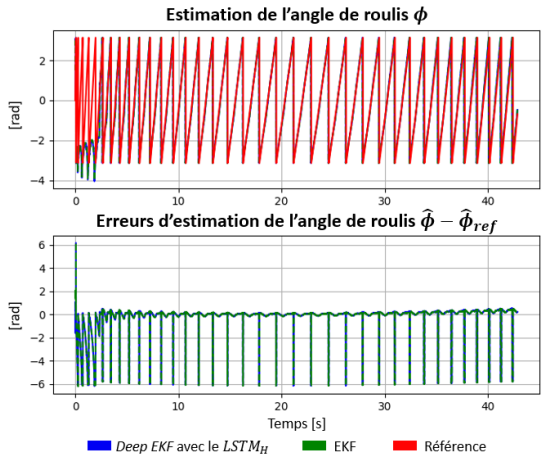

FIGURE 5.28 – Estimation et erreurs d'estimation de l'angle de roulis pour une trajectoire de mortier de 120 mm : le *Deep EKF* adapté par le $LSTM_H$ (en bleu), l'EKF traditionnel (en vert) et l'angle de référence (en rouge).

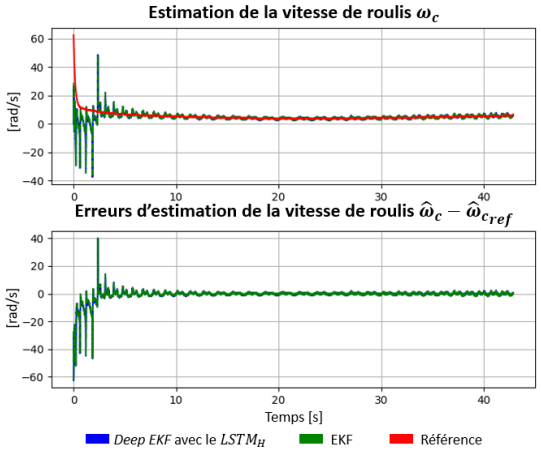

FIGURE 5.29 – Estimation et erreurs d'estimation de la vitesse de roulis pour une trajectoire de mortier de 120 mm : le *Deep EKF* adapté par le $LSTM_H$ (en bleu), l'EKF traditionnel (en vert) et la vitesse de référence (en rouge).

Les figures 5.28 et 5.29 montrent que le *Deep EKF* où la matrice d'observation est évaluée par le $LSTM_H$ et l'EKF traditionnel présenté dans la partie *(5.5)* ont des performances semblables pour l'estimation de l'angle de roulis et de la vitesse de roulis d'un mortier de 120 mm. Afin de valider ces observations et de vérifier l'apport de cette méthode hybride pour estimer l'angle et la vitesse de roulis d'un mortier de 120 mm, deux critères d'erreur sont évalués sur l'ensemble du jeu de données de test :

- l'erreur quadratique moyenne globale, notée $\mathcal{C}_{RMSE}$ et définie telle que :

$$\mathcal{C}_{RMSE} = \frac{1}{N_{sim}} \sum_{k=1}^{N_{sim}} RMSE(sim_k) \tag{5.79}$$

- le score noté $\mathcal{C}_{score}$, tel que :

$$\mathcal{C}_{score} = \sum_{k=1}^{N_{sim}} RMSE(Deep\ EKF)_{sim_k} < RMSE(EKF)_{sim_k} \tag{5.80}$$

avec N_{sim} le nombre de simulations du jeu de données de test et $RMSE(.)$ définie comme la racine carrée de l'erreur quadratique moyenne (5.58).

La figure 5.30 présente les critères $\mathcal{C}_{RMSE}$ (5.79) et $\mathcal{C}_{score}$ (5.80) évalués sur l'ensemble du jeu de données de test pour l'estimation des angles et des vitesses de roulis d'un mortier de 120 mm en comparant le *Deep EKF* avec la matrice d'observation déterminée par le $LSTM_H$ (en bleu) et l'EKF traditionnel présenté dans la partie *(5.5.1)* de ce document (en vert).

La figure 5.30 indique que les erreurs quadratiques moyennes globales $\mathcal{C}_{RMSE}$ du *Deep EKF* et de l'EKF traditionnel sont similaires pour l'estimation de l'angle de roulis. Des observations semblables peuvent être formulées pour l'estimation des vitesses de roulis. Toutefois, les scores $\mathcal{C}_{score}$ indiquent que l'EKF traditionnel surpasse légèrement le *Deep EKF*. Ainsi, malgré un pré-entraînement, le *Deep EKF* où le modèle d'observation est estimé par $LSTM_H$ ne surpasse pas une méthode classique d'estimation basée sur des modèles mathématiques traditionnels. Ces constatations concordent avec celles observées pour le *Deep EKF* où le modèle de mesure est estimé par $LSTM_Y$. La forme périodique des signaux à estimer influe sur les performances des deux LSTM.

D'après les figures 5.20 - 5.30, la solution de *Deep Kalman Filter* proposée dans cette partie n'est pas optimale. En effet, l'ajustement du modèle de mesure et du modèle d'observation d'un EKF par des LSTM ne réduit pas les erreurs d'estimation comparé à un

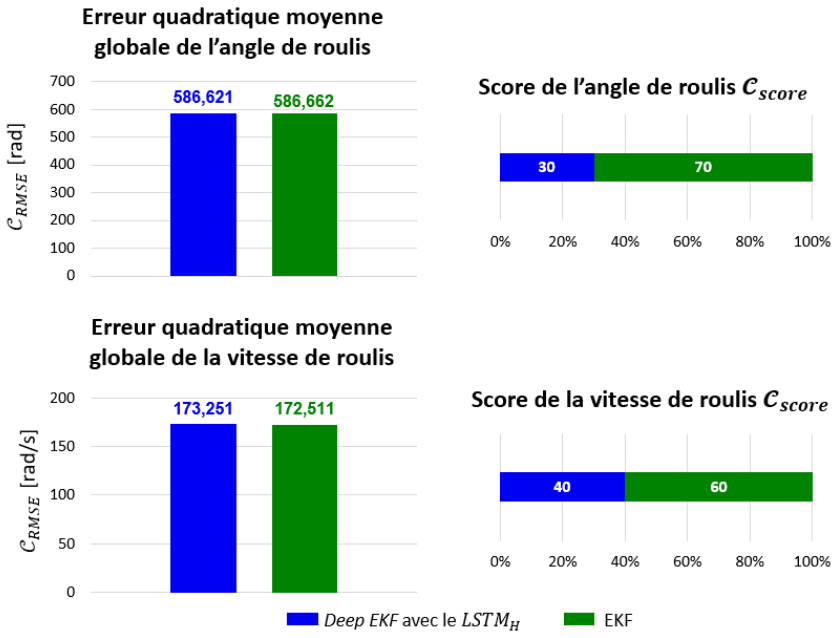

FIGURE 5.30 – Critères $\mathcal{C}_{RMSE}$ (5.79) et $\mathcal{C}_{score}$ (5.80) obtenus par le *Deep EKF* avec la matrice d'observation déterminée par le $LSTM_H$ (en bleu) et l'EKF traditionnel (en vert) pour l'estimation de l'angle et de la vitesse de roulis d'un mortier de 120 mm.

filtre de Kalman traditionnel sans réseau de neurones. Ces conclusions s'expliquent en partie par la forme des signaux à estimer qui sont périodiques et pour lesquels le LSTM n'est pas optimal. Une piste d'amélioration vise à considérer d'autres formes de réseaux de neurones récurrents pour résoudre ce problème d'estimation.

5.6 Conclusion

Ce chapitre étudie plusieurs *Deep Kalman Filter* afin d'évaluer leurs apports et leurs intérêts pour estimer la trajectoire d'un projectile. Les réseaux de neurones permettent ainsi de pallier les limites d'un modèle mathématique ou de compléter des mesures erronées d'un filtre de Kalman.

Trois formes de *Deep Kalman Filter* sont évaluées dans ce chapitre.
La première solution implémentée vise à remplacer les observations, nécessaires lors de l'étape de mise à jour d'un filtre, par un réseau de neurones. Pour cela, et en se basant sur le principe de fusion des données IMU et GNSS, un filtre de Kalman est corrigé par les positions et les vitesses déterminées par un LSTM. Les résultats obtenus permettent

de valider la notion de *Deep Kalman Filter*. De plus, la combinaison d'un modèle d'IA et d'un filtre de Kalman est une solution appropriée à l'estimation de la trajectoire d'un projectile.
La seconde méthode se focalise sur l'estimation de l'un des modèles mathématiques d'un filtre de Kalman par un réseau de neurones. Pour cela, un *Deep Error State Kalman Filter* est proposé dans le cas où le modèle d'évolution du filtre est remplacé par un LSTM, afin de modéliser les erreurs des capteurs inertiels et de produire une solution de navigation précise à long terme. Les résultats rapportés confirment que cette configuration hybride est adéquate pour estimer les positions et les vitesses d'un projectile.
La dernière approche hybride mise en place vise à évaluer comment l'IA permet d'adapter un modèle mathématique. Pour cela, un filtre de Kalman, valide dans le cas d'un tir tendu, est adapté à une trajectoire balistique en employant l'IA pour déterminer les modèles inconnus. Deux réseaux de neurones récurrents sont employés afin d'évaluer les deux modèles mathématiques inconnus. Malgré divers tests, les résultats d'estimation sont mitigés. En effet, cette approche se concentre exclusivement sur l'estimation de l'angle de roulis de la munition alors qu'un LSTM n'est pas optimal pour estimer de tels signaux.

Au vu des résultats obtenus dans ce chapitre, les *Deep Kalman Filter* sont des solutions prometteuses pour l'estimation des positions et des vitesses d'un projectile caractérisé par une trajectoire balistique. Plusieurs formes de *Deep Kalman Filter* nécessitent d'être formulées comme l'estimation du gain de Kalman ou l'évaluation de plusieurs modèles d'un même filtre par des réseaux de neurones. De plus, l'influence de différents types de réseaux de neurones pour de telles prédictions nécessite d'être étudiée.

CONCLUSION

Conclusion générale

Les travaux présentés dans ce document évaluent l'apport et l'intérêt des algorithmes d'intelligence artificielle (IA) pour estimer les trajectoires de différents projectiles. Pour cela, plusieurs approches combinant des méthodes classiques de navigation et des réseaux de neurones sont étudiées.
Les trajectoires des projectiles sont habituellement estimées par des filtres de Kalman fusionnant les mesures inertielles et GNSS *(Global Navigation Satellite Systems)* embarquées. Cependant, comme ces mesures présentent plusieurs limitations, l'IA vise à être intégrée dans les solutions de navigation afin de corriger les erreurs induites par ces mesures. Ces travaux évaluent ainsi l'apport de l'IA pour la navigation des projectiles.

Trois approches d'estimation de la trajectoire d'un projectile sont analysées :

→ La première approche vise à utiliser des réseaux de neurones convolutifs pour ajuster dynamiquement la matrice de covariance du bruit de mesure d'un filtre de Kalman. Cette solution est notamment appliquée à un filtre de Kalman Imparfait Invariant Étendu, un filtre de Kalman Étendu et un filtre de Kalman linéaire. Les résultats obtenus montrent que cette solution n'est pas adaptée à tous les filtres. En effet, l'apport de cette approche est principalement bénéfique dans le cas du filtre de Kalman Imparfait Invariant Étendu qui estime la trajectoire d'un mortier de 120 mm à partir des mesures de l'accéléromètre, du gyromètre et du magnétomètre embarqué ainsi que de la connaissance du champ magnétique de référence.

→ La deuxième méthode implémentée vise à remplacer tous les modèles mathématiques décrivant l'évolution de la trajectoire d'un projectile par un *Long Short-Term Memory*. D'après les résultats obtenus, cette approche est parfaitement adaptée à l'estimation des positions et des vitesses des projectiles caractérisés par des trajectoires balistiques. Toutefois, cette méthode n'est pas optimale pour estimer l'orientation des projectiles avec de faibles vitesses de rotation. De plus, cette solution de navigation n'est pas appropriée aux projectiles caractérisés par des tirs tendus du fait des courtes durées de vols et des faibles variations des grandeurs.

→ La troisième méthode mise en œuvre se concentre sur les *Deep Kalman Filter* qui vise à remplacer l'un des modèles mathématiques d'un filtre de Kalman par un réseau de neurones. Plusieurs approches sont étudiées notamment la génération d'observations par un réseau de neurones pour corriger un filtre de Kalman, l'estimation d'un modèle de prédiction d'un filtre de Kalman, et l'adaptation de modèles à une dynamique différente. Les résultats proposés indiquent que l'apprentissage par transfert est nécessaire dans le cas d'un *Deep Kalman Filter*. De plus, cette solution reste prometteuse comme elle se base à la fois sur des méthodes classiques de navigation et sur l'intelligence artificielle. Toutefois, l'entraînement d'un *Deep Kalman Filter* n'est pas aisé et nécessite de choisir soigneusement les données d'entrée.

Les résultats proposés dans ce document montrent que l'IA présente un intérêt pour la navigation des projectiles, notamment lorsque les capteurs classiquement utilisés comme le récepteur GNSS ne sont pas disponibles. Toutefois, les résultats obtenus indiquent que l'IA n'est pas la solution privilégiée pour estimer l'orientation d'un projectile.
Ces travaux visent à être approfondis afin de résoudre les problèmes d'estimation des angles d'Euler ainsi que les estimations inexactes des trajectoires des projectiles caractérisés par des tirs tendus.

Perspectives

L'IA présente des performances intéressantes pour le traitement d'image. C'est pourquoi, en collaboration avec Guillaume COURTIER, doctorant à l'ISL sur le développement d'un système de navigation basé sur la polarisation de la lumière solaire [19], une méthode de navigation exploitant l'IA et des images a été évaluée.

Comme présenté dans la figure suivante, cette approche vise estimer le cap d'un véhicule terrestre à partir d'images polarisées du ciel et d'un réseau de neurones convolutifs *(Convolutional Neural Network - CNN)*. En effet, l'information de polarisation contenue dans les images du ciel capturées par une caméra spécialisée permet de déterminer le cap d'un véhicule à partir de deux paramètres :

- l'angle de polarisation linéaire *(Angle of Linear Polarization - AoLP)* qui correspond à l'angle de polarisation de la lumière du ciel par rapport à l'axe du capteur de la caméra.
- le degré de polarisation linéaire *(Degree of Linear Polarization - DoLP)* qui re-

présente la proportion de la lumière polarisée dans la lumière totale reçue par la caméra.

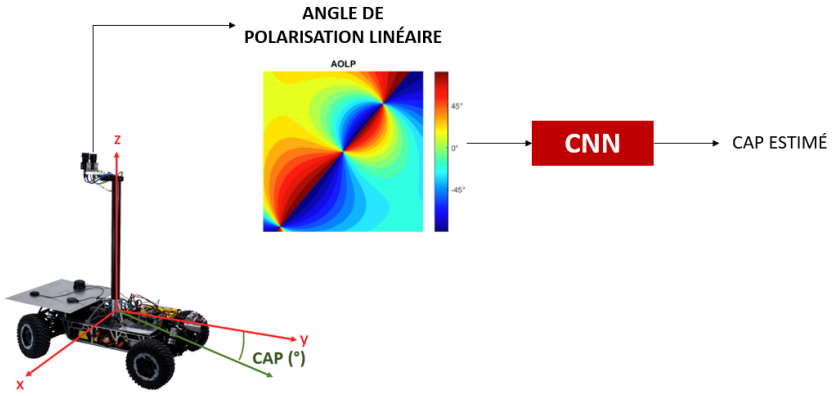

FIGURE 5.31 – Estimation du cap d'un véhicule terrestre à partir d'images polarisées du ciel et d'un réseau de neurones convolutifs *(Convolutional Neural Network - CNN)*.

Afin d'estimer le cap d'un véhicule terrestre par un CNN et à partir d'images polarisées du ciel, un simulateur est employé. Le simulateur génère les AoLP, les DoLP et les caps associés à partir de la connaissance de la position du soleil dans le ciel. Un CNN est alors entraîné pour estimer le cap du véhicule à partir de l'AoLP et du DoLP.

Les caps estimés par des traitements d'images et des modèles mathématiques développés par Guillaume COURTIER présentent des erreurs moyennes de l'ordre du degré. Les caps estimés par le CNN alimenté par l'AoLP et le DoLP conduisent à des erreurs similaires, à l'exception de certaines positions spécifiques du soleil. Ainsi, l'apport de l'IA pour estimer le cap d'un véhicule à partir des informations de polarisation contenues dans la lumière du ciel est validé.
L'intérêt de cette approche est de pallier certaines limitations du traitement d'image classique, comme l'estimation des trois angles d'orientation du robot qui génère de grandes erreurs. De ce fait, l'IA permet de remédier à cette problématique.

ANNEXES

Annexe A. Imp.RIEKF pour la navigation des projectiles

L'annexe A est dédiée à l'identification des matrices jacobiennes A_t et A_{d_χ} de l'Imp.RIEKF présenté dans la partie *(2.4.2)* de ce document. Ces matrices sont déterminées d'après la dynamique de l'erreur linéarisée :

$$\frac{d}{dt}\xi_t = A_t\xi_t + A_{d_\chi}w_t \tag{A.1}$$

D'après les propriétés de l'opérateur $[.]_\times$ rappelées par l'équation (2.10), la dynamique de l'erreur d'orientation η_R est :

$$\begin{array}{rcl} \frac{d}{dt}\eta_R & = & \frac{d}{dt}(\widehat{R}R^T) = \widehat{R}^{-1}R^T + \widehat{R}(R^{-1})^T \\ & = & \widehat{R}[\tilde{\omega}_s - \widehat{b}_{\omega_s}]_\times R^T + \widehat{R}(R[\tilde{\omega}_s - b_{\omega_s} - W_{\omega_s}]_\times)^T \\ & = & \widehat{R}[W_{\omega_s} - \xi_{b_{\omega_s}}]_\times \widehat{R}^T\widehat{R}R^T = [\widehat{R}(W_{\omega_s} - \xi_{b_{\omega_s}})]_\times\eta_R \\ & \approx & [\widehat{R}(W_{\omega_s} - \xi_{b_{\omega_s}})]_\times[\xi_R]_\times + o(\|\xi_R\|) \approx \frac{d}{dt}[\xi_R]_\times \end{array} \tag{A.2}$$

Donc, la dynamique d'évolution de l'erreur linéarisée d'orientation est :

$$\frac{d}{dt}\xi_R = \widehat{R}(W_{\omega_s} - \xi_{b_{\omega_s}})\xi_R \tag{A.3}$$

D'après la définition des erreurs invariantes, la dynamique de l'erreur de vitesse η_v est :

$$\begin{array}{rcl} \frac{d}{dt}\eta_v & = & \frac{d}{dt}(\widehat{v} - \widehat{R}R^T v) = (\widehat{v})^{-1} - (\widehat{R}R^T)^{-1}v - \widehat{R}R^T v^{-1} \\ & = & \widehat{R}(W_{a_s} - \xi_{b_{a_s}}) + g - (\eta_R)^{-1}v - \eta_R g \\ & \approx & \widehat{R}(W_{a_s} - \xi_{b_{a_s}}) - [\widehat{R}(W_{\omega_s} - \xi_{b_{\omega s}})]_\times v - [\xi_R]_\times g \\ & \approx & \widehat{R}(W_{a_s} - \xi_{b_{a_s}}) + [v]_\times\widehat{R}(W_{\omega_s} - \xi_{b_{\omega_s}}) + [g]_\times\xi_R \approx \frac{d}{dt}\xi_v \end{array} \tag{A.4}$$

Donc, la dynamique d'évolution de l'erreur linéarisée de vitesse est :

$$\frac{d}{dt}\xi_v = \widehat{R}(W_{a_s} - \xi_{b_{a_s}}) + [v]_\times \widehat{R}(W_{\omega_s} - \xi_{b_{\omega_s}}) + [g]_\times \xi_R \tag{A.5}$$

D'après les définitions des erreurs invariantes d'orientation et de vitesse, la dynamique de l'erreur de position η_p est :

$$\begin{array}{rcc} \frac{d}{dt}\eta_p & = & \frac{d}{dt}(\widehat{p} - \widehat{R}R^T p) = (\widehat{p})^{-1} - (\widehat{R}R^T)^{-1}p - \widehat{R}R^T p^{-1} \\ & = & (\widehat{v} - \widehat{R}R^T v) - (\widehat{R}R^T)^{-1}p = \eta_v - (\eta_R)^{-1}p \\ & \approx & \xi_v - [\widehat{R}(W_{\omega_s} - \xi_{b_{\omega_s}})]_\times p \\ & \approx & \xi_v + [p]_\times \widehat{R}(W_{\omega_s} - \xi_{b_{\omega_s}}) \\ & \approx & \frac{d}{dt}\xi_p \end{array} \tag{A.6}$$

Donc, la dynamique d'évolution de l'erreur linéarisée de position est :

$$\frac{d}{dt}\xi_p = \xi_v + [p]_\times \widehat{R}(W_{\omega_s} - \xi_{b_{\omega_s}}) \tag{A.7}$$

Pour conclure, sachant (A.3), (A.5) et (A.7) et d'après la dynamique d'évolution des erreurs de biais et de l'erreur invariante η_{R_b},

$$\begin{array}{l} \frac{d}{dt}\xi_{b_{\omega_s}} = W_{b_{\omega_s}}, \\ \frac{d}{dt}\xi_{b_{a_s}} = W_{b_{a_s}}, \\ \frac{d}{dt}\xi_{R_b} = W_{R_b}, \\ \frac{d}{dt}\xi_{p_b} = W_{p_b}, \end{array} \tag{A.8}$$

la dynamique de l'erreur linéarisée ξ_t permet d'identifier les matrices jacobiennes A_t et A_{d_χ} de sorte que :

$$\frac{d}{dt}\xi_t = \begin{bmatrix} 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & -R & 0_{3\times3} & 0_{6\times3} \\ [g]_\times & 0_{3\times3} & 0_{3\times3} & -[v]_\times R & -R & 0_{6\times3} \\ 0_{3\times3} & I_{3\times3} & 0_{3\times3} & -[p]_\times R & 0_{3\times3} & 0_{6\times3} \\ & 0_{12\times9} & & & 0_{12\times12} & \end{bmatrix} \xi_t + \begin{bmatrix} R & 0_{3\times3} & 0_{3\times3} & 0_{12\times3} \\ [v]_\times R & R & 0_{3\times3} & 0_{12\times3} \\ [p]_\times R & 0_{3\times3} & R & 0_{12\times3} \\ & 0_{12\times9} & & I_{12\times12} \end{bmatrix} w_t \tag{A.9}$$

Annexe B. Quaternions

Cette annexe vise à rappeler brièvement le formalisme des quaternions et des opérations associées [117], [136].

Soit p et q deux quaternions définis comme suit :

$$p = \begin{bmatrix} p_0 \\ p_v \end{bmatrix} = \begin{bmatrix} p_0 \\ p_x \\ p_y \\ p_z \end{bmatrix} \qquad q = \begin{bmatrix} q_0 \\ q_v \end{bmatrix} = \begin{bmatrix} q_0 \\ q_x \\ q_y \\ q_z \end{bmatrix} \tag{B.1}$$

La matrice de rotation associée au quaternion q est définie telle que :

$$R = R\{q\} = \begin{bmatrix} q_0^2 + q_x^2 - q_y^2 - q_z^2 & 2(q_x q_y - q_0 q_z) & 2(q_x q_z + q_0 q_y) \\ 2(q_x q_y + q_0 q_z) & q_0^2 - q_x^2 + q_y^2 - q_z^2 & 2(q_y q_z - q_0 q_x) \\ 2(q_x q_z - q_0 q_y) & 2(q_y q_z + q_0 q_x) & q_0^2 - q_x^2 - q_y^2 + q_z^2 \end{bmatrix} \tag{B.2}$$

Le produit de p et q, noté $\otimes$, est :

$$\begin{aligned} p \otimes q &= \begin{bmatrix} p_0 q_0 - p_v^T q_v \\ p_0 q_v + q_0 p_v + p_v \times q_v \end{bmatrix} \\ &= \begin{bmatrix} p_0 q_0 - p_x q_x - p_y q_y - p_z q_z \\ p_0 q_x + p_x q_0 + p_y q_z - p_z q_y \\ p_0 q_y - p_x q_z + p_y q_0 + p_z q_x \\ p_0 q_z + p_x q_y - p_y q_x + p_z q_0 \end{bmatrix} \end{aligned} \tag{B.3}$$

Soit $a \in \mathbb{R}^{3\times 1}$, les opérateurs $\Omega(.)$ et $\mathbb{E}(.)$ sont définis tels que :

$$\Omega(a) = \begin{bmatrix} 0 & -a^T \\ a & -[a]_\times \end{bmatrix}, \qquad \mathbb{E}(q) = \begin{bmatrix} -q_v^T \\ q_0 I_3 + [q_v]_\times \end{bmatrix} \tag{B.4}$$

Les dérivations relatives aux quaternions sont données par les équations suivantes :

$$\begin{gathered} \frac{\partial(q\otimes a\otimes q^*)}{\partial a} = \frac{Ra}{\partial a} = R \\ \frac{\partial(q\otimes a\otimes q^*)}{\partial q} = \frac{Ra}{\partial q} = 2\begin{bmatrix} q_0 a + [q_v]_\times a & q_v^T a I_3 + q_v a^T - a q_v^T - q_0 [a]_\times \end{bmatrix} \in \mathbb{R}^{3\times 4} \end{gathered} \tag{B.5}$$

Annexe C. EKF pour la navigation des projectiles

L'annexe C présente les dynamiques d'évolution des quaternions et la matrice jacobienne d'évolution de l'EKF présenté dans la partie *(2.4.3)* de ce document.

C.1. Dynamique d'évolution des quaternions

La dynamique d'évolution du quaternion $q = \begin{bmatrix} q_0 & q_v \end{bmatrix}^T$ avec $\omega \in \mathbb{R}^{3\times 1}$ est donnée par l'équation suivante :

$$\dot{q} = \frac{1}{2} q \otimes \begin{bmatrix} 0 \\ \omega \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 0 & -\omega^T \\ \omega & -[\omega]_\times \end{bmatrix} q = \frac{1}{2} \Omega(\omega) q = \frac{1}{2} \mathbb{E}(q) \omega \tag{C.1.1}$$

avec $\otimes$, $\Omega(.)$ et $\mathbb{E}(.)$ des opérateurs définis dans l'annexe *B*.
La dynamique d'évolution du quaternion q en temps discret est alors :

$$\begin{array}{rcl} q_{k+1} & = & q_k + \frac{1}{2}\Omega(\omega_{s_k}) q_k \Delta_t \\ & = & q_k + \frac{1}{2}\Omega(\tilde{\omega}_{s_k} - b_{\omega_{s_k}} - W_{\omega_{s_k}}) q_k \Delta_t \\ & = & \left(I_{4\times 4} + \frac{\Delta_t}{2}\Omega(\tilde{\omega}_{s_k} - b_{\omega_{s_k}})\right) q_k - \frac{\Delta_t}{2}\Omega(W_{\omega_{s_k}}) q_k \end{array} \tag{C.1.2}$$

La dynamique d'évolution de q_b en temps continu est :

$$\dot{q_b} = \frac{1}{2} q_b \otimes \begin{bmatrix} 0 \\ W_{R_b} \end{bmatrix} = \frac{1}{2} \Omega(W_{R_b}) q_b = \frac{1}{2} \mathbb{E}(q_b) W_{R_b} \tag{C.1.3}$$

alors la dynamique d'évolution du quaternion q_b en temps discret est :

$$q_{b_{k+1}} = q_{b_k} + \frac{1}{2} \Omega(W_{R_b}) q_{b_k} \Delta_t = q_{b_k} + \frac{1}{2} \mathbb{E}(q_{b_k}) W_{R_b} \Delta_t \tag{C.1.4}$$

C.2. Matrice jacobienne d'évolution

La matrice jacobienne ϕ_k est définie telle que :

$$\phi_k \approx I_{23} + \frac{\partial f}{\partial x}|_{x_k, u_k} \Delta t \tag{C.2.1}$$

Les dérivées partielles de la dynamique d'évolution $f(.)$ (2.86) par rapport aux états permet d'identifier la matrice jacobienne ϕ_k de sorte que :

$$\begin{array}{c} \frac{\partial q}{\partial q} = \frac{1}{2}\Omega(\tilde{\omega}_s - b_{\omega_s}) \\ \frac{\partial q}{\partial b_{\omega_s}} = \frac{\partial}{\partial b_{\omega_s}}\left(\frac{1}{2}\mathbb{E}(q)(\tilde{\omega}_s - b_{\omega_s})\right) = -\frac{1}{2}\mathbb{E}(q) \\ \frac{\partial v}{\partial q} = \frac{\partial q \otimes (\tilde{a}_s - b_{a_s}) \otimes q^*}{\partial q} \\ \frac{\partial v}{\partial b_{a_s}} = -R \\ \frac{\partial p}{\partial v} = I_3 \end{array} \tag{C.2.2}$$

La représentation matricielle permet d'identifier la matrice jacobienne ϕ_k telle que :

$$\phi_k = I_{23\times 23} + \begin{bmatrix} \frac{1}{2}\Omega(\tilde{\omega}_{sk} - b_{\omega_{sk}}) & 0_{4\times 3} & 0_{4\times 3} & -\frac{1}{2}\mathbb{E}(q_k) & 0_{4\times 3} & 0_{4\times 7} \\ \frac{\partial}{\partial q}(q \otimes (\tilde{a}_s - b_{a_s}) \otimes q^*) & 0_3 & 0_3 & 0_3 & -R_k & 0_{3\times 7} \\ 0_{3\times 4} & I_3 & 0_3 & 0_3 & 0_3 & 0_{3\times 7} \\ & & & 0_{13\times 23} & & \end{bmatrix} \Delta_t \tag{C.2.3}$$

et avec la matrice $\frac{\partial}{\partial q}(q \otimes a \otimes q^*)$ définie dans l'annexe *B*.

Annexe D. Imp.RIEKF corrigé par les mesures des magnétomètres

L'annexe D présente l'identification des matrices jacobiennes d'évolution A_t et G_W ainsi que la matrice d'observation H de l'Imp.RIEKF présenté dans la partie *(3.3.1)* de ce document.

D.1. Identification des matrices jacobiennes d'évolution

Les matrices A_t et G_w sont identifiées d'après la dynamique de l'erreur linéarisée :

$$\frac{d}{dt}\xi_t = A_t\xi_t + G_w W \tag{D.1.1}$$

D'après les propriétés de l'opérateur $[.]_\times$, la dynamique d'évolution de l'erreur invariante d'orientation s'écrit :

$$\begin{array}{rcl} \frac{d}{dt}\eta_R & = & \frac{d}{dt}(\widehat{R}R^T) = \widehat{R}^{-1}R^T + \widehat{R}(R^{-1})^T \\ & = & \widehat{R}[\tilde{\omega}_s - \widehat{b}_{\omega_s}]_\times R^T - \widehat{R}[\tilde{\omega}_s - b_{\omega_s} - W_{\omega_s}]_\times R^T \\ & = & \widehat{R}[W_{\omega_s} - (\widehat{b}_{\omega_s} - b_{\omega_s})]_\times R^T = \widehat{R}[W_{\omega_s} - \xi_{b_\omega}]_\times R^T \\ & = & \widehat{R}[W_{\omega_s} - \xi_{b_\omega})]_\times \widehat{R}^T\widehat{R}R^T = [\widehat{R}(W_{\omega_s} - \xi_{b_\omega})]_\times \eta_R \end{array} \tag{D.1.2}$$

L'erreur invariante d'orientation est linéarisée sur l'espace euclidien associé tel que $\eta_R \approx I_{3\times 3} + [\xi_R]_\times$. La dynamique de l'erreur linéarisée d'orientation est alors :

$$\frac{d}{dt}\xi_R = \widehat{R}(W_{\omega_s} - \xi_{b_\omega}) \tag{D.1.3}$$

D'après la définition de l'erreur d'orientation et des biais des accéléromètres, l'erreur invariante de vitesse est :

$$\begin{array}{rcl} \frac{d}{dt}\eta_v & = & (\widehat{v})^{-1} - (\widehat{R}R^T)^{-1}v - \widehat{R}R^T v^{-1} \\ & = & \widehat{R}(W_{a_s} - \xi_{b_a}) + g - (\eta_R)^{-1}v - \eta_R g \\ & \approx & \widehat{R}(W_{a_s} - \xi_{b_a}) - [\widehat{R}(W_{\omega_s} - \xi_{b_\omega})]_\times v - [\xi_R]_\times g \\ & \approx & \widehat{R}(W_{a_s} - \xi_{b_a}) + [v]_\times \widehat{R}(W_{\omega_s} - \xi_{b_\omega}) + [g]_\times \xi_R \approx \frac{d}{dt}\xi_v \end{array} \tag{D.1.4}$$

Ainsi, la dynamique de l'erreur linéarisée de vitesse est :

$$\frac{d}{dt}\xi_v = \widehat{R}(W_{a_s} - \xi_{b_a}) + [v]_\times \widehat{R}(W_{\omega_s} - \xi_{b_\omega}) + [g]_\times \xi_R \tag{D.1.5}$$

Par définition de l'erreur invariante de vitesse et d'orientation, la dynamique de l'erreur invariante de position est :

$$\begin{array}{rcl} \frac{d}{dt}\eta_p & = & (\widehat{p})^{-1} - (\widehat{R}R^T)^{-1}p - \widehat{R}R^T p^{-1} \\ & = & \widehat{v} - (\widehat{R}R^T)^{-1}p - \widehat{R}R^T v \\ & = & (\widehat{v} - \widehat{R}R^T v) - (\widehat{R}R^T)^{-1}p = \eta_v - (\eta_R)^{-1}p \\ & \approx & \xi_v - [\widehat{R}(W_{\omega_s} - \xi_{b_\omega})]_\times p \\ & \approx & \xi_v + [p]_\times \widehat{R}(W_{\omega_s} - \xi_{b_\omega}) \approx \frac{d}{dt}\xi_p \end{array} \tag{D.1.6}$$

Ainsi, la dynamique de l'erreur linéarisée de position est :

$$\frac{d}{dt}\xi_p = \xi_v + [p]_\times \widehat{R}(W_\omega - \xi_{b_\omega}) \tag{D.1.7}$$

Pour conclure, sachant (D.1.3) - (D.1.7) et d'après la dynamique d'évolution des erreurs de biais,

$$\frac{d}{dt}\xi_{b_\omega} = W_{b_{\omega_s}}, \qquad \frac{d}{dt}\xi_{b_a} = W_{b_{a_s}}, \qquad \frac{d}{dt}\xi_{b_h} = W_{b_{h_s}}, \tag{D.1.8}$$

alors la représentation sous forme matricielle de la dynamique d'évolution de l'erreur linéarisée permet d'identifier les matrices A_t et G_W de sorte que :

$$A_t = \begin{bmatrix} 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & -R & 0_{3\times3} & 0_{3\times3} \\ [g]_\times & 0_{3\times3} & 0_{3\times3} & -[v]_\times R & -R & 0_{3\times3} \\ 0_{3\times3} & I_{3\times3} & 0_{3\times3} & -[p]_\times R & 0_{3\times3} & 0_{3\times3} \\ & & & 0_{9\times18} & & \end{bmatrix}, \quad G_w = \begin{bmatrix} R & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ [v]_\times R & R & 0_{3\times3} & 0_{3\times3} \\ [p]_\times R & 0_{3\times3} & R & 0_{3\times3} \\ & 0_{9\times9} & & I_{9\times9} \end{bmatrix} \tag{D.1.9}$$

D.2. Identification de la matrice d'observation et de la matrice de covariance du bruit de mesure

La matrice d'observation H et la matrice de covariance du bruit de mesure N sont identifiées d'après le modèle d'observation linéarisé :

$$h(x, W_{h_s}) - h(\widehat{x}, 0) = H\xi_t + NW_{h_s} + o(\xi_t) + o(W_{h_s}) \tag{D.2.1}$$

D'après le modèle de mesure de l'Imp.RIEK (3.33),

$$h(x, W_{h_s}) - h(\widehat{x}, 0) = R(\tilde{h}_s - b_{h_s} - W_{h_s}) - \widehat{R}(\tilde{h}_s - \widehat{b}_{h_s}) \tag{D.2.2}$$

alors d'après les définitions des erreurs :

$$\begin{aligned} h(x, W_{h_s}) - h(\widehat{x}, 0) &= (\eta_R^{-1}\widehat{R} - \widehat{R})\tilde{h}_s + \widehat{R}\widehat{b}_{h_s} - \eta_R^{-1}\widehat{R}b_{h_s} - \eta_R^{-1}\widehat{R}W_{h_s} \\ &= -[\xi_R]_\times \widehat{R}\tilde{h}_s + \widehat{R}(\widehat{b}_{h_s} - b_{h_s}) + \widehat{R}W_{h_s} + o(\xi_t) + o(W_{h_s}) \\ &= [\widehat{R}\tilde{h}_s]_\times \xi_R + \widehat{R}\xi_{b_h} + \widehat{R}W_{h_s} \end{aligned} \tag{D.2.3}$$

La matrice d'observation H et la matrice de covariance du bruit de mesure N sont identifiées d'après l'équation (D.2.1) de sorte que :

$$\begin{aligned} H &= \begin{bmatrix} [R\tilde{h}_s]_\times & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & R \end{bmatrix} \\ N &= cov(RW_{h_s}) = Rcov(W_{h_s})R^T. \end{aligned} \tag{D.2.4}$$

BIBLIOGRAPHIE

[1] NEXTER, « AMMUNITION Nexter Catalogue 2018 », Report, 2018. adresse : https://www.nexter-group.fr/sites/default/files/2020-05/20180604%20Nexter%20-%20Catalogue%20Ammunition.pdf.

[2] P. B. RUFFIN et S. J. BURGETT, « Recent progress in MEMS technology development for military applications », *Smart Structures and Materials 2001 : Smart Electronics and MEMS*, t. 4334, p. 1-12, 2001.

[3] T. G. BROWN, B. DAVIS, D. HEPNER et al., « Strap-down microelectromechanical (MEMS) sensors for high-g munition applications », *IEEE Transactions on Magnetics*, t. 37, *1*, p. 336-342, 2001.

[4] P. D. GROVES, « Principles of GNSS, inertial, and multisensor integrated navigation systems », *IEEE Aerospace and Electronic Systems Magazine*, t. 30, *2*, p. 26-27, 2015.

[5] F. LIU, Z. SU, H. ZHAO, Q. LI et C. LI, « Attitude measurement for high-spinning projectile with a hollow MEMS IMU consisting of multiple accelerometers and gyros », *Sensors*, t. 19, *8*, p. 1799, 2019.

[6] A. FIOT, S. CHANGEY et N. PETIT, « Attitude estimation for artillery shells using magnetometers and frequency detection of accelerometers », *Control Engineering Practice*, t. 122, p. 105 080, 2022.

[7] M. STEFER, E. PECHEUR et L. BERNARD, « Instrumentation of the 40mm-projectile for roll rate and roll angle estimation », in *2016 IEEE 2nd International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI)*, IEEE, 2016, p. 1-5.

[8] N. JARDAK, R. ADAM et S. CHANGEY, « A Gyroless Algorithm with Multi-Hypothesis Initialization for Projectile Navigation », *Sensors*, t. 21, *22*, p. 7487, 2021.

[9] S. Changey, E. Pecheur, L. Bernard, E. Sommer, P. Wey et C. Berner, « Real time estimation of projectile roll angle using magnetometers : In-flight experimental validation », p. 371-376, 2012. doi : 10.1109/PLANS.2012.6236904.

[10] J. Zhao, « A review of wearable IMU (Inertial-Measurement-Unit)-based pose estimation and drift reduction technologies », in *Journal of Physics : Conference Series*, IOP Publishing, t. 1087, 2018, p. 042 003.

[11] M. Narasimhappa, A. D. Mahindrakar, V. C. Guizilini, M. H. Terra et S. L. Sabat, « An improved Sage Husa adaptive robust Kalman Filter for denoising the MEMS IMU drift signal », in *2018 Indian Control Conference (ICC)*, IEEE, 2018, p. 229-234.

[12] C. J. Hegarty, « GNSS signals—An overview », in *2012 IEEE International Frequency Control Symposium Proceedings*, IEEE, 2012, p. 1-7.

[13] G. Lachapelle, « High sensitivity GNSS limitations in RF perturbed environments », *NATO STO lecture series SET-197, navigation sensors and systems in GNSS degraded and denied environments*, 2013.

[14] G. T. Schmidt, « Navigation sensors and systems in GNSS degraded and denied environments », *Chinese journal of aeronautics*, t. 28, *1*, p. 1-10, 2015.

[15] G. X. Gao, M. Sgammini, M. Lu et N. Kubo, « Protecting GNSS receivers from jamming and interference », *Proceedings of the IEEE*, t. 104, *6*, p. 1327-1338, 2016.

[16] A. Ruegamer, D. Kowalewski et al., « Jamming and spoofing of GNSS signals–an underestimated risk », *Proc. Wisdom Ages Challenges Modern World*, t. 3, p. 17-21, 2015.

[17] M. L. Psiaki et T. E. Humphreys, « GNSS spoofing and detection », *Proceedings of the IEEE*, t. 104, *6*, p. 1258-1270, 2016.

[18] H. Sheng et T. Zhang, « MEMS-based low-cost strap-down AHRS research », *Measurement*, t. 59, p. 63-72, 2015.

[19] G. Courtier, R. Adam, P.-J. Lapray, E. Pecheur, S. Changey et J.-P. Lauffenburger, « Image-based navigation system using skylight polarization for an unmanned ground vehicle », in *Unmanned Systems Technology XXIV*, SPIE, t. 12124, 2022, p. 173-182.

[20] D. Simon, « Kalman filtering », *Embedded systems programming*, t. 14, *6*, p. 72-79, 2001.

[21] T. Karvonen et al., « Stability of linear and non-linear Kalman filters », 2014.

[22] G. Welch, G. Bishop et al., « An introduction to the Kalman filter », 1995.

[23] M. I. Ribeiro, « Kalman and extended kalman filters : Concept, derivation and properties », *Institute for Systems and Robotics*, t. 43, p. 46, 2004.

[24] A. Barrau et S. Bonnabel, « Extended Kalman filtering with nonlinear equality constraints : A geometric approach », *IEEE Transactions on Automatic Control*, t. 65, *6*, p. 2325-2338, 2019.

[25] K. Reif, S. Gunther, E. Yaz et R. Unbehauen, « Stochastic stability of the discrete-time extended Kalman filter », *IEEE Transactions on Automatic control*, t. 44, *4*, p. 714-728, 1999.

[26] B. Ni et Q. Zhang, « Stability of the Kalman filter for continuous time output error systems », *Systems & Control Letters*, t. 94, p. 172-180, 2016.

[27] Y. Yang et G. Huang, « Observability analysis of aided INS with heterogeneous features of points, lines, and planes », *IEEE Transactions on Robotics*, t. 35, *6*, p. 1399-1418, 2019.

[28] G. Huang et S. Mourikis Anastasiosand Roumeliotis, « Observability-based Rules for Designing Consistent EKF-SLAM Estimators », *The International Journal of Robotics Research*, t. 29, *5*, p. 502-528, 2010. doi : 10.1177/0278364909353640. eprint : https://doi.org/10.1177/0278364909353640. adresse : https://doi.org/10.1177/0278364909353640.

[29] K. W. Lee, W. S. Wijesoma et J. I. Guzman, « On the observability and observability analysis of SLAM », in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2006, p. 3569-3574.

[30] J. Hesch, D. Kottas, S. Bowman et S. Roumeliotis, « Observability-constrained vision-aided inertial navigation », *University of Minnesota, Dept. of Comp. Sci. & Eng., MARS Lab, Tech. Rep*, t. 1, p. 6, 2012.

[31] G. Huang, A. Mourikis et S. Roumeliotis, « Genearalized analysis and improvement of the consistency for EKF-based SLAM », *University of Minnesota*, p. 2008-0001, 2008.

[32] L. L. Wells, « The projectile GRAM SAASM for ERGM and Excalibur », in *IEEE 2000. Position Location and Navigation Symposium (Cat. No. 00CH37062)*, IEEE, 2000, p. 106-111.

[33] L. D. Fairfax et F. E. Fresconi, « Position estimation for projectiles using low-cost sensors and flight dynamics », *Journal of Aerospace Engineering*, t. 27, *3*, p. 611-620, 2014.

[34] H. Zhao et Z. Li, « Ultra-tight GPS/IMU integration based long-range rocket projectile navigation », *Defence Science Journal*, t. 66, *1*, p. 64-70, 2016.

[35] J. Vandersteen, S. Bennani et C. Roux, « Robust Rocket Navigation with Sensor Uncertainties : Vega Launcher Application », *Journal of Spacecraft and Rockets*, t. 55, *1*, p. 153-166, 2018.

[36] A. Radi, S. Zahran et N. El-Sheimy, « GNSS Only Reduced Navigation System Performance Evaluation for High-Speed Smart Projectile Attitudes Estimation », in *2021 International Telecommunications Conference (ITC-Egypt)*, IEEE, 2021, p. 1-4.

[37] M. B. Aykenar, I. C. Boz, G. Soysal et M. Efe, « A Multiple Model Approach for Estimating Roll Rate of a Very Fast Spinning Artillery Rocket », in *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*, IEEE, 2020, p. 1-7.

[38] D. Long, J. Lin, X. Zhang et J. Li, « Orientation estimation algorithm applied to high-spin projectiles », *Measurement Science and Technology*, t. 25, *6*, p. 065 001, 2014.

[39] H. Zhao et Z. Su, « Real-time estimation of roll angle for trajectory correction projectile using radial magnetometers », *IET radar, sonar & navigation*, t. 14, *10*, p. 1559-1570, 2020.

[40] L. An, L. Wang, N. Liu, J. Fu et Y. Zhong, « A novel method for estimating pitch and yaw of rotating projectiles based on dynamic constraints », *Sensors*, t. 19, *23*, p. 5096, 2019.

[41] S. Changey, D. Beauvois et V. Fleck, « A mixed extended-unscented filter for attitude estimation with magnetometer sensor », in *2006 American Control Conference*, IEEE, 2006, 6-pp.

[42] J. M. Maley, « Efficient attitude estimation for a spin-stabilized projectile », *Journal of Guidance, Control, and Dynamics*, t. 39, *2*, p. 339-350, 2016.

[43] C. Combettes, S. Changey, R. Adam et E. Pecheur, « Attitude and velocity estimation of a projectile using low cost magnetometers and accelerometers », in *2018 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, IEEE, 2018, p. 650-657.

[44] A. Fiot, S. Changey et N. C. Petit, « Estimation of air velocity for a high velocity spinning projectile using transerse accelerometers », in *2018 AIAA Guidance, Navigation, and Control Conference*, 2018, p. 1349.

[45] S. Changey, V. Fleck et D. Beauvois, « Projectile attitude and position determination using magnetometer sensor only », *Proc. SPIE*, t. 5803, mars 2005. doi : 10.1117/12.602099.

[46] Z. Zhao, G. Wen, X. Zhang et D. Li, « Model-based estimation for pose, velocity of projectile from stereo linear array image », *Measurement Science Review*, t. 12, *3*, p. 104, 2012.

[47] L. Hsiang-Yueh et K. Hao-Yuan, « Projectile Flight Trajectory and Position Estimation System Based on Stereo Vision », *Sensors and Materials*, t. 31, *11*, p. 3483-34, 2019. doi : https://doi.org/10.18494/SAM.2019.

[48] T. G. Dietterich, « Machine learning », *Annual review of computer science*, t. 4, *1*, p. 255-306, 1990.

[49] E. Charniak, *Introduction au deep learning*. Dunod, 2021.

[50] S. Sharma, S. Sharma et A. Athaiya, « Activation functions in neural networks », *towards data science*, t. 6, *12*, p. 310-316, 2017.

[51] D. P. Kingma et J. Ba, « Adam : A method for stochastic optimization », *arXiv preprint arXiv :1412.6980*, 2014.

[52] P. Svenmarck, L. Luotsinen, M. Nilsson et J. Schubert, « Possibilities and challenges for artificial intelligence in military applications », in *Proceedings of the NATO Big Data and Artificial Intelligence for Military Decision Making Specialists' Meeting*, 2018, p. 1-16.

[53] Y. Zhang, Z. Dai, L. Zhang, Z. Wang, L. Chen et Y. Zhou, « Application of Artificial Intelligence in Military : From Projects View », in *2020 6th International Conference on Big Data and Information Analytics (BigDIA)*, IEEE, 2020, p. 113-116.

[54] A. Carlo, « Artificial Intelligence in the Defence Sector », in *International Conference on Modelling and Simulation for Autonomous Systems*, Springer, 2020, p. 269-278.

[55] M. Hoijtink et A. Planqué-van Hardeveld, « Machine learning and the platformization of the military : A study of google's machine learning platform TensorFlow », *International Political Sociology*, t. 16, *2*, olab036, 2022.

[56] D. C. Tarraf, W. Shelton, E. Parker et al., « The Department of Defense posture for artificial intelligence : Assessment and recommendations », Rand National Defense Research Inst Santa Monica CA United States, rapp. tech., 2019.

[57] S. Zeldam, « Automated failure diagnosis in aviation maintenance using explainable artificial intelligence (XAI) », mém. de mast., University of Twente, 2018.

[58] V. Ustun, R. Kumar, A. Reilly, S. Sajjadi et A. Miller, « Adaptive synthetic characters for military training », *arXiv preprint arXiv :2101.02185*, 2021.

[59] P. Svenmarck, L. Luotsinen, M. Nilsson et J. Schubert, « Possibilities and challenges for artificial intelligence in military applications », in *Proceedings of the NATO Big Data and Artificial Intelligence for Military Decision Making Specialists' Meeting*, 2018, p. 1-16.

[60] D. Ventre, *Artificial Intelligence, Cybersecurity and Cyber Defence*. John Wiley & Sons, 2020.

[61] T. Farrell, T. Terry et O. Frans, *A transformation gap ? : American innovations and European military change*. Stanford University Press, 2010.

[62] N. J. Wheeler, « Guardian angel or global gangster : A review of the ethical claims of international society », *Political Studies*, t. 44, *1*, p. 123-135, 1996.

[63] M. des Armées (France), « L'intelligence Artificielle au Service de la Défense », Report, 2019.

[64] J. Liu et G. Guo, « Vehicle localization during GPS outages with extended Kalman filter and deep learning », *IEEE Transactions on Instrumentation and Measurement*, t. 70, p. 1-10, 2021.

[65] Y. Yao, X. Xu, C. Zhu et C.-Y. Chan, « A hybrid fusion algorithm for GPS/INS integration during GPS outages », *Measurement*, t. 103, p. 42-51, 2017.

[66] J. Li, N. Song, G. Yang, M. Li et Q. Cai, « Improving positioning accuracy of vehicular navigation system during GPS outages utilizing ensemble learning algorithm », *Information Fusion*, t. 35, p. 1-10, 2017.

[67] C. Shen, Y. Zhang, J. Tang, H. Cao et J. Liu, « Dual-optimization for a MEMS-INS/GPS system during GPS outages based on the cubature Kalman filter and neural networks », *Mechanical Systems and Signal Processing*, t. 133, p. 106 222, 2019.

[68] W. Fang, J. Jiang, S. Lu et al., « A LSTM algorithm estimating pseudo measurements for aiding INS during GNSS signal outages », *Remote sensing*, t. 12, *2*, p. 256, 2020.

[69] J. J. Wang, J. Wang, D. Sinclair et L. Watts, « A neural network and Kalman filter hybrid approach for GPS/INS integration », in *Proceedings of the Korean Institute of Navigation and Port Research Conference*, Korean Institute of Navigation et Port Research, t. 1, 2006, p. 277-282.

[70] A. AbdulMajuid, O. Mohamady, M. Draz et G. El-bayoumi, « GPS-Denied Navigation Using Low-Cost Inertial Sensors and Recurrent Neural Networks », *arXiv preprint arXiv :2109.04861*, 2021.

[71] Z. Shi, F. Zhao, X. Wang et Z. Jin, « Deep Kalman-based Trajectory Estimation of Moving Target from Satellite Images », in *2022 IEEE 10th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*, IEEE, t. 10, 2022, p. 71-75.

[72] D. Kim, K. Min, H. Kim et K. Huh, « Vehicle sideslip angle estimation using deep ensemble-based adaptive Kalman filter », *Mechanical Systems and Signal Processing*, t. 144, p. 106 862, 2020.

[73] L. Vargas-Meléndez, B. L. Boada, M. J. L. Boada, A. Gauchía et V. Díaz, « A sensor fusion method based on an integrated neural network and Kalman filter for vehicle roll angle estimation », *Sensors*, t. 16, *9*, p. 1400, 2016.

[74] J. Liu, W. Wang, X. Gong, X. Que et H. Yang, « A hybrid model based on Kalman filter and neutral network for traffic prediction », in *2012 IEEE 2nd International Conference on Cloud Computing and Intelligence Systems*, IEEE, t. 2, 2012, p. 533-536.

[75] T. Bao, Y. Zhao, S. A. R. Zaidi, S. Xie, P. Yang et Z. Zhang, « A deep Kalman filter network for hand kinematics estimation using sEMG », *Pattern Recognition Letters*, t. 143, p. 88-94, 2021.

[76] S. Bi, L. Ma, T. Shen, Y. Xu et F. Li, « Neural network assisted Kalman filter for INS/UWB integrated seamless quadrotor localization », *PeerJ Computer Science*, t. 7, e630, 2021.

[77] X. Dai, V. Nateghi, H. Fourati et C. Prieur, « Q-learning-based noise covariance adaptation in Kalman filter for MARG sensors attitude estimation », in *2022 IEEE International Symposium on Inertial Sensors and Systems (INERTIAL)*, IEEE, 2022, p. 1-6.

[78] X. Dai, H. Fourati et C. Prieur, « A Dynamic Grid-based Q-learning for Noise Covariance Adaptation in EKF and its Application in Navigation », in *2022 IEEE 61st Conference on Decision and Control (CDC)*, IEEE, 2022, p. 4984-4989.

[79] D.-J. Jwo et H.-C. Huang, « Neural network aided adaptive extended Kalman filtering approach for DGPS positioning », *The journal of navigation*, t. 57, *3*, p. 449-463, 2004.

[80] L. Abdrazakov et D. Yudin, « Neural Network Adaptation of the Kalman Filter for Odometry Fusion », in *Proceedings of the Fifth International Scientific Conference "Intelligent Information Technologies for Industry"(IITI'21)*, Springer, 2022, p. 44-54.

[81] S. Jouaber, S. Bonnabel, S. Velasco-Forero et M. Pilte, « NNAKF : A Neural Network Adapted Kalman Filter for Target Tracking », in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2021, p. 4075-4079.

[82] M. Brossard, A. Barrau et S. Bonnabel, « AI-IMU dead-reckoning », *IEEE Transactions on Intelligent Vehicles*, t. 5, *4*, p. 585-595, 2020.

[83] S. Hosseinyalamdary, « Deep Kalman filter : Simultaneous multi-sensor integration and modelling ; A GNSS/IMU case study », *Sensors*, t. 18, *5*, p. 1316, 2018.

[84] G. Revach, N. Shlezinger, X. Ni, A. L. Escoriza, R. J. Van Sloun et Y. C. Eldar, « KalmanNet : Neural network aided Kalman filtering for partially known dynamics », *IEEE Transactions on Signal Processing*, t. 70, p. 1532-1547, 2022.

[85] Y.-t. Bai, X.-y. Wang, X.-b. Jin, Z.-y. Zhao et B.-h. Zhang, « A neuron-based Kalman filter with nonlinear autoregressive model », *Sensors*, t. 20, *1*, p. 299, 2020.

[86] C. Chen, C. X. Lu, B. Wang, N. Trigoni et A. Markham, « DynaNet : Neural Kalman dynamical model for motion estimation and prediction », *IEEE Transactions on Neural Networks and Learning Systems*, t. 32, *12*, p. 5479-5491, 2021.

[87] Z. Shi, M. Xu, Q. Pan, B. Yan et H. Zhang, « LSTM-based flight trajectory prediction », in *2018 International joint conference on neural networks (IJCNN)*, IEEE, 2018, p. 1-8.

[88] L. Rossi, M. Paolanti, R. Pierdicca et E. Frontoni, « Human trajectory prediction and generation using LSTM models and GANs », *Pattern Recognition*, t. 120, p. 108 136, 2021.

[89] K. A. Sørensen, P. Heiselberg et H. Heiselberg, « Probabilistic maritime trajectory prediction in complex scenarios using deep learning », *Sensors*, t. 22, *5*, p. 2058, 2022.

[90] E. Barsoum, J. Kender et Z. Liu, « HP-GAN : Probabilistic 3D human motion prediction via GAN », in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, p. 1418-1427.

[91] A. Al-Molegi, M. Jabreel et B. Ghaleb, « STF-RNN : Space time features-based recurrent neural network for predicting people next location », in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, IEEE, 2016, p. 1-7.

[92] S. H. Park, B. Kim, C. M. Kang, C. C. Chung et J. W. Choi, « Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder-decoder architecture », in *2018 IEEE intelligent vehicles symposium (IV)*, IEEE, 2018, p. 1672-1678.

[93] N. E. Gaiduchenko, P. A. Gritsyk et Y. I. Malashko, « Multi-Step Ballistic Vehicle Trajectory Forecasting Using Deep Learning Models », in *2020 International Conference Engineering and Telecommunication (En&T)*, IEEE, 2020, p. 1-6.

[94] L.-h. Hou et H.-j. Liu, « An end-to-end LSTM-MDN network for projectile trajectory prediction », in *Intelligence Science and Big Data Engineering. Big Data and Machine Learning : 9th International Conference, IScIDE 2019, Nanjing, China, October 17–20, 2019, Proceedings, Part II 9*, Springer, 2019, p. 114-125.

[95] N. Nikhil et B. Tran Morris, « Convolutional Neural Network for trajectory prediction », in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018.

[96] P. Wey, D. Corriveau, T. A. Saitz, W. de Ruijter et P. Strömbäck, « BALCO 6/7-DoF trajectory model », in *29th International Symposium on Ballistics*, t. 1, 2016, p. 151-162.

[97] D. Corriveau, « Validation of the NATO Armaments Ballistic Kernel for use in small-arms fire control systems », *Defence technology*, t. 13, *3*, p. 188-199, 2017.

[98] M. Zeidler, « Application des techniques de contrôle des écoulements au pilotage des projectiles : Contrôle fluidique d'un projectile gyrostabilisé de 155 mm par effet Coanda », thèse de doct., Université de Lille, 2015.

[99] A. Barrau, « Non-linear state error based extended Kalman filters with applications to navigation », thèse de doct., Mines Paristech, 2015.

[100] P. Chauchat, « Smoothing algorithms for navigation, localisation and mapping based on high-grade inertial sensors », thèse de doct., Université Paris sciences et lettres, 2020.

[101] A. Barrau et S. Bonnabel, « The invariant extended Kalman filter as a stable observer », *IEEE Transactions on Automatic Control*, t. 62, *4*, p. 1797-1812, 2016.

[102] R. Hartley, M. Ghaffari, R. M. Eustice et J. W. Grizzle, « Contact-aided invariant extended Kalman filtering for robot state estimation », *The International Journal of Robotics Research*, t. 39, *4*, p. 402-430, 2020.

[103] A. Barrau et S. Bonnabel, « Invariant kalman filtering », *Annual Review of Control, Robotics, and Autonomous Systems*, t. 1, p. 237-257, 2018.

[104] S. Teng, M. W. Mueller et K. Sreenath, « Legged robot state estimation in slippery environments using invariant extended Kalman filter with velocity update », in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2021, p. 3104-3110.

[105] S. Bonnabel, « Observateurs asymptotiques invariants : théorie et exemples », thèse de doct., École Nationale Supérieure des Mines de Paris, 2007.

[106] A. Barrau et S. Bonnabel, « Invariant particle filtering with application to localization », *53rd IEEE Conference on Decision and Control*, p. 5599-5605, 2014.

[107] S. Bonnabel, « Une approche géométrique pour certains problèmes de filtrage non-linéaires. », thèse de doct., Université Pierre et Marie Curie, 2014.

[108] M. Brossard, « Deep learning, inertial measurements units, and odometry : some modern prototyping techniques for navigation based on multi-sensor fusion », thèse de doct., Université Paris sciences et lettres, 2020.

[109] F. Paulin, « Introduction aux groupes de Lie pour la physique », 2018. adresse : `https://www.math.u-psud.fr/~paulin/notescours/cours_centrale.pdf`.

[110] B. S. Marc Briant Lucas Chesnel, « Des Groupes de Lie et de leurs applications en Physique Fondamentale », 2008.

[111] E. Eade, « Lie groups for computer vision », *Cambridge Univ., Cam-bridge, UK, Tech. Rep*, t. 2, 2014.

[112] K. S. Phogat et D. E. Chang, « Invariant extended Kalman filter on matrix Lie groups », *Automatica*, t. 114, p. 108 812, 2020.

[113] S. Bonnable, P. Martin et E. Salaün, « Invariant extended Kalman filter : theory and application to a velocity-aided attitude estimation problem », in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, IEEE, 2009, p. 1297-1304.

[114] A. Fiot, « Attitude estimation of an artillery shell in free-flight from accelerometers and magnetometers », thèse de doct., Université Paris sciences et lettres, 2020.

[115] J. D. Hol, T. B. Schon et F. Gustafsson, « A new algorithm for calibrating a combined camera and IMU sensor unit », in *2008 10th International Conference on Control, Automation, Robotics and Vision*, IEEE, 2008, p. 1857-1862.

[116] M. Bloesch, C. Gehring, P. Fankhauser, M. Hutter, M. A. Hoepflinger et R. Siegwart, « State estimation for legged robots on unstable and slippery terrain », in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2013, p. 6058-6064.

[117] J. Sola, « Quaternion kinematics for the error-state Kalman filter », *arXiv preprint arXiv :1711.02508*, 2017.

[118] X. Zhu et J. Angeles, « A Reparametrization of the Rotation Matrix in Rigid-Body Dynamics », *Journal of Applied Mechanics*, t. 82, *5*, p. 051 003, 2015.

[119] Z. Chen, K. Jiang et J. C. Hung, « Local observability matrix and its application to observability analyses », in *IECON'90 : 16th Annual Conference of IEEE Industrial Electronics Society*, IEEE, 1990, p. 100-103.

[120] R. Chauhan, K. K. Ghanshala et R. Joshi, « Convolutional Neural Network (CNN) for image detection and recognition », in *2018 first international conference on secure cyber computing and communication (ICSCCC)*, IEEE, 2018, p. 278-282.

[121] Z. Liang, A. Powell, I. Ersoy et al., « CNN-based image analysis for malaria diagnosis », in *2016 IEEE international conference on bioinformatics and biomedicine (BIBM)*, IEEE, 2016, p. 493-496.

[122] A. Sherstinsky, « Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network », *Physica D : Nonlinear Phenomena*, t. 404, p. 132 306, 2020.

[123] Y. Yu, X. Si, C. Hu et J. Zhang, « A review of recurrent neural networks : LSTM cells and network architectures », *Neural computation*, t. 31, *7*, p. 1235-1270, 2019.

[124] R. C. Staudemeyer et E. R. Morris, « Understanding LSTM–a tutorial into Long Short-Term Memory recurrent neural networks », *arXiv preprint arXiv :1909.09586*, 2019.

[125] S. Hochreiter et J. Schmidhuber, « Long Short-Term Memory », *Neural computation*, t. 9, *8*, p. 1735-1780, 1997.

[126] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber et al., « Gradient flow in recurrent nets : the difficulty of learning long-term dependencies », 2001.

[127] A. Graves, S. Fernández et J. Schmidhuber, « Bidirectional LSTM networks for improved phoneme classification and recognition », in *Artificial Neural Networks : Formal Models and Their Applications–ICANN 2005 : 15th International Conference, Warsaw, Poland, September 11-15, 2005. Proceedings, Part II 15*, Springer, 2005, p. 799-804.

[128] L. Ziyin, T. Hartwig et M. Ueda, « Neural networks fail to learn periodic functions and how to fix it », *Advances in Neural Information Processing Systems*, t. 33, p. 1583-1594, 2020.

[129] T. Deng, A. Cheng, W. Han et H.-X. Lin, « Visibility Forecast for Airport Operations by LSTM Neural Network. », in *ICAART (2)*, 2019, p. 466-473.

[130] S. Kim, I. Petrunin et H.-S. Shin, « A Review of Kalman Filter with Artificial Intelligence Techniques », in *2022 Integrated Communication, Navigation and Surveillance Conference (ICNS)*, IEEE, 2022, p. 1-12.

[131] K. Weiss, T. M. Khoshgoftaar et D. Wang, « A survey of transfer learning », *Journal of Big data*, t. 3, *1*, p. 1-40, 2016.

[132] F. Zhuang, Z. Qi, K. Duan et al., « A comprehensive survey on transfer learning », *Proceedings of the IEEE*, t. 109, *1*, p. 43-76, 2020.

[133] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang et C. Liu, « A survey on deep transfer learning », in *Artificial Neural Networks and Machine Learning–ICANN 2018 : 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27*, Springer, 2018, p. 270-279.

[134] N. Agarwal, A. Sondhi, K. Chopra et G. Singh, « Transfer learning : Survey and classification », *Smart Innovations in Communication and Computational Sciences : Proceedings of ICSICCS 2020*, p. 145-155, 2021.

[135] L. Marković, M. Kovač, R. Milijas, M. Car et S. Bogdan, « Error state extended Kalman filter multi-sensor fusion for unmanned aerial vehicle localization in GPS and magnetometer denied indoor environments », in *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, 2022, p. 184-190.

[136] W. Youn et S. A. Gadsden, « Combined quaternion-based error state Kalman filtering and smooth variable structure filtering for robust attitude estimation », *IEEE Access*, t. 7, p. 148 989-149 004, 2019.

Titre : Étude des méthodes d'intelligence artificielle pour la navigation des projectiles

Mot clés : Navigation, Projectile, Filtre de Kalman, Intelligence Artificielle

Résumé : La navigation des projectiles se base sur les mesures d'une Unité de Mesure Inertielle *(IMU - Inertial Measurement Unit)* et d'un récepteur GNSS *(Global Navigation Satellite Systems)*, fusionnées par des filtres de Kalman. Toutefois, ces capteurs présentent plusieurs limitations notamment en termes de précision et de disponibilité. L'objectif de cette thèse est d'évaluer l'apport de l'intelligence artificielle (IA) pour optimiser les méthodes classiques de navigation afin d'estimer avec précision la trajectoire d'un projectile.

Trois méthodes d'estimation de la trajectoire d'un projectile ont été mises en œuvre. La première approche consiste à ajuster un paramètre de bruit d'un filtre de Kalman par un réseau de neurones afin de tenir compte des erreurs et des corrélations des mesures. La deuxième méthode vise à remplacer tous les modèles mathématiques d'évolution de la trajectoire d'un projectile par un réseau de neurones récurrents. La troisième solution mise en œuvre évalue les performances d'estimation d'un filtre de Kalman où au moins l'un des modèles est estimé par un réseau de neurones.

Les résultats obtenus montrent l'intérêt de l'IA pour la navigation des projectiles, notamment lorsque les capteurs classiquement utilisés comme le récepteur GNSS ne sont pas disponibles. Toutefois, l'IA n'est pas optimale pour estimer l'orientation d'un projectile.

Title: Study of artificial intelligence methods for projectile navigation

Keywords: Navigation, Projectile, Kalman Filter, Artificial Intelligence

Abstract: Projectile navigation is based on the Inertial Measurement Unit and the GNSS *(Global Navigation Satellite Systems)* receiver readings fused by Kalman filters. However, these sensors present several limitations in terms of accuracy and availability. The goal of this thesis is to evaluate Artificial Intelligence (AI) contribution to optimize classical navigation methods to accurately estimate a projectile trajectory.

Three methods are implemented to estimate a projectile trajectory. The first approach aims to adjust a noise parameter of a Kalman filter by a neural network to take into account the measurement errors and correlations. The second approach aims to replace all projectile trajectory mathematical models by a recurrent neural network. The third solution implemented evaluates the accuracy of a Kalman filter when at least one model is estimated by a neural network.

The reported results highlight the interest of AI for projectile navigation, especially when conventional sensors such as GNSS receivers are not available. However, AI is not optimal to estimate projectile orientation.