sensors

MDPI

Article

LSTM-Based Projectile Trajectory Estimation in a GNSS-Denied Environment †

Alicia Roux [1,2,*], Sébastien Changey [1], Jonathan Weber [2] and Jean-Philippe Lauffenburger [2]

[1] French-German Research Institute of Saint-Louis, 5 Rue du Général Casssagnou, 68300 Saint-Louis, France; sebastien.changey@isl.eu
[2] Institut de Recherche en Informatique, Mathématiques, Automatique et Signal (IRIMAS), Université de Haute-Alsace, 2 Rue des Frères Lumière, 68100 Mulhouse, France; jonathan.weber@uha.fr (J.W.); jean-philippe.lauffenburger@uha.fr (J.-P.L.)
* Correspondence: alicia.roux@uha.fr
† Conference on Artificial Intelligence for Defense, DGA Maîtrise de l'Information, 16–17 November 2022, Rennes, France.

Abstract: This paper presents a deep learning approach to estimate a projectile trajectory in a GNSS-denied environment. For this purpose, Long-Short-Term-Memories (LSTMs) are trained on projectile fire simulations. The network inputs are the embedded Inertial Measurement Unit (IMU) data, the magnetic field reference, flight parameters specific to the projectile and a time vector. This paper focuses on the influence of LSTM input data pre-processing, i.e., normalization and navigation frame rotation, leading to rescale 3D projectile data over similar variation ranges. In addition, the effect of the sensor error model on the estimation accuracy is analyzed. LSTM estimates are compared to a classical Dead-Reckoning algorithm, and the estimation accuracy is evaluated via multiple error criteria and the position errors at the impact point. Results, presented for a finned projectile, clearly show the Artificial Intelligence (AI) contribution, especially for the projectile position and velocity estimations. Indeed, the LSTM estimation errors are reduced compared to a classical navigation algorithm as well as to GNSS-guided finned projectiles.

Keywords: long-short-term-memory; projectile trajectory; navigation

Citation: Roux, A.; Changey, S.; Weber, J.; Lauffenburger, J.-P. LSTM-Based Projectile Trajectory Estimation in a GNSS-Denied Environment. *Sensors* **2023**, *23*, 3025. https://doi.org/10.3390/s23063025

Academic Editors: Chee Kiat Seow, Henrik Hesse, Kai Wen, Soon Yim Tan and Yunjia Wang

Received: 6 February 2023
Revised: 27 February 2023
Accepted: 5 March 2023
Published: 10 March 2023

1. Introduction

Projectile navigation is mainly based on IMU (Inertial Measurement Unit) and GNSS (Global Navigation Satellite System) measurements due to the high dynamic constraints imposed on projectiles and the low-cost sensor requirements. Classically, IMU and GNSS measurements are combined with Kalman Filters to estimate a trajectory. The IMU measurements are integrated to predict the trajectory in order to be corrected by the GNSS receiver measurements [1–4]. Nevertheless, GNSS signals are not always available due to the environment configuration and are vulnerable to jamming and spoofing [5–7]. For this purpose, users aim to exclude these measurements for trajectory estimation [8–10].

Moreover, Artificial Intelligence (AI) is increasingly used for defense applications such as surveillance, reconnaissance, tracking or navigation [11–15]. Indeed, AI is an interesting approach to correct model approximations, to limit the influence of incomplete or incorrect measurements or to determine complex models from system data. Therefore, this paper presents an AI-based projectile trajectory estimation method in a GNSS-denied environment using only the embedded IMU and pre-flight parameters specific to the ammunition.

Considering that a trajectory is a time series, AI provides interesting approaches for its estimation. Indeed, Recurrent Neural Networks (RNNs) are perfectly adapted to time series prediction as illustrated in [16–18]. RNNs are composed by feedback loops, i.e., they memorize past data through hidden states to predict future data. However, the simplest form of RNNs exhibits convergence issues during the training step such as vanishing or

exploding gradient problems. So another form of recurrent network is considered: the Long Short-Term Memory (LSTM) [19,20]. A LSTM includes a memory cell in addition to hidden states, in order to capture both long-term and short-term time dependencies.

This paper presents an AI-based solution to estimate a projectile trajectory in a GNSS-denied environment. In summary, the main contributions of this work are:

- to detail an LSTM-based approach to estimate projectile positions, velocities and Euler angles from the embedded IMU, the magnetic field reference, pre-flight parameters and a time vector.
- to present BALCO (BALlistic COde) [21] used to generate the dataset. This simulator provides true-to-life trajectories of several projectiles according to the ammunition parameters.
- to investigate different normalization forms of the LSTM input data in order to evaluate their contribution on the estimation accuracy. For this purpose, several LSTMs are trained with different input data normalizations.
- to study the impact of the local navigation frame rotation on the estimation accuracy. Rotating the local navigation frame during the training step allows having similar variation ranges along the three axes, especially for the lateral position, which is extremely small compared to the two other axes. This method shares the same goals as normalization but without any information loss.
- to examine the influence of inertial sensor models on estimation accuracy. For this purpose, two inertial sensor error models are studied in order to evaluate their influence on LSTM predictions.
- to compare the LSTM accuracy to a Dead-Reckoning, performed on finned mortar trajectory. Estimation methods are evaluated through error criteria based on the Root Mean Square Error and the impact point error.

The outline of the paper is as follows. Section 2 presents an introduction to projectile navigation, AI applications in the military field and to the LSTM basics. Section 3 focuses on the projectile trajectory dataset and LSTM specifications. Section 4 presents the data pre-processing; the input data normalization and the local navigation frame rotation during training. Finally, Section 5 presents projectile trajectory estimation results by analyzing the influence of the local navigation frame rotation, input data normalization and sensor model on estimation accuracy.

2. Related Work

This part presents the traditional projectile navigation methods and the sensors used, the applications of artificial intelligence for defense and the LSTM operating principle.

2.1. Model-Based Projectile Trajectory Estimation

Projectile navigation requires sensors able to resist to extreme conditions (acceleration shocks around 50,000 g along the longitudinal axis, high rotation rates around 15,000 rpm for a 155 mm shell) as well as to be relatively small and inexpensive due to the space and cost constraints imposed on projectiles [22–24]. For this purpose, projectile navigation mainly uses IMUs (Inertial Measurement Units) composed by accelerometers, gyrometers, and magnetometers as well as GNSS (Global navigation satellite system) data. On one hand, the IMU measurement integration, performed at high frequency ($\sim$1000 Hz), provides an accurate short-term projectile trajectory estimation, but deviates at long term due to sensor drift [1]. On the other hand, GNSS receivers provide accurate long-term position information at a significantly lower frequency ($\sim$10 Hz), but can be easily spoofed and jammed [5–7]. Due to their evident complementarity, IMUs and GNSS are classically fused by different types of Kalman filters for trajectory estimation, as in [2–4].

One challenge with high-speed spinning projectiles is to accurately estimate altitudes. Depending on projectile specifications, different methods can be considered, as in [25,26] by exploiting GNSS signals or in [9,22] by using accelerometers, gyrometers or magnetometers. Nevertheless gyrometers are commonly omitted because, under cost limits, many of them

saturate and do not resist to launch phases. Therefore, magnetometers, less expensive and able to resist to high accelerations, are often used [27–29] with different kinds of Kalman filters (Extended Kalman Filter, Adaptive Extended Kalman Filter, Mixed Extended-Unscented Filter).

As mentioned in the introduction (see Section 1), GNSS signals are easily spoofed and jammed. Therefore, more and more, approaches are proposed in a GNSS-denied environment in order to estimate a projectile trajectory, using only inertial measurements, as in [8,30–32].

2.2. AI-Based Trajectory Estimation

AI methods are increasingly used in the military field especially for:

- *Surveillance and target recognition* where machine learning algorithms applied to computer vision detect, identify and track objects of interest.
  For example, the Maven project, presented by the US Department of Defense (DoD), focuses on automatic target identification and localization from images collected by Aerial Vehicles [11].
- *Predictive maintenance* to establish the optimal time to change a part of a system, as the US Army does on F-16 aircraft [12].
- *Military training* where AI is used in training simulation software to improve efficiency through various scenarios, such as AIMS (Artificial Intelligence for Military Simulation) [13].
- *Analysis and decision support* to extract and deal with relevant elements in an information flow, to analyze a field or to predict events. The Defense Advanced Research Projects Agency (DARPA) aims to equip US Army helicopter pilots with augmented reality helmets to support them in operations [14].
- *Cybersecurity*, as military systems are strongly sensitive to cyberattacks leading to loss and theft of critical information. To this end, the DeepArmor program from SparkCognition uses AI to protect, detect and block networks, computer programs and data from cyber threats [15].

Although widely integrated in the military development programs, AI is relatively uncommonly applied to projectile trajectory estimation although some kinds of networks such as recurrent networks are perfectly adapted to this task.

2.2.1. Recurrent Neural Networks (RNNs)

Recurrent Neural Networks (RNNs) are a class of neural networks perfectly adapted for time series prediction [19,20] as they exploit feedback loops, i.e., an RNN cell output at the previous time is used as input at the current time.

As illustrated in Figure 1, an RNN exploits a time series $x = [x_0, x_1, ..., x_\tau] \in \mathbb{R}^{\tau \times F}$ of length τ with F input features to predict an output y. Each prediction y_t is determined by one RNN cell from the current input x_t and the previous output h_{t-1}, also called hidden state, which memorizes the past information [19,20].

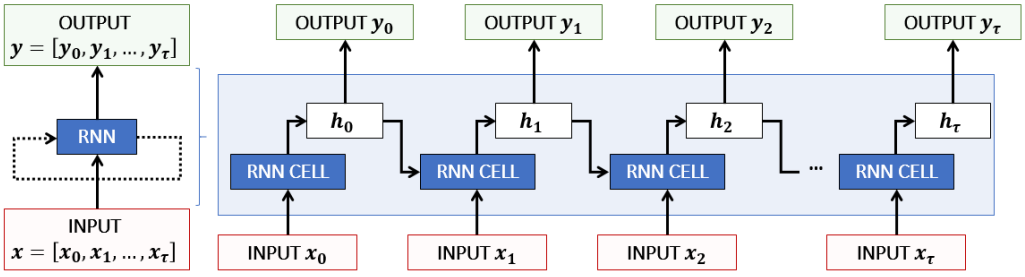

Figure 1. RNN layer overview, roll and unroll: many-to-many representation.

Vanilla RNN, the simplest RNN structure, suffers from gradient vanishing and explosion during the training step [33,34]. In the gradient vanishing case, backpropagation from the last layer to the first layer leads to a gradient reduction. Then, the first layer weights are no longer updated during training and the Vanilla RNN does not learn any features. In the gradient explosion case, gradients become increasingly large leading to huge weight updates and thus resulting in Vanilla RNN divergence.

Moreover, to predict y_t at timestamp t, the Vanilla RNN uses only the input x_t at the current time and the hidden state h_{t-1} at the previous time, containing short-term past features. For this reason, Vanilla RNN is ineffective to memorize long-term past events. To overcome these issues, memory cells are added to the Vanilla RNN, forming the Long Short-Term Memory (LSTM) [19].

2.2.2. Long Short-Term Memory Cell

Based on the recurrent network overview presented above, this paragraph focuses only on the LSTM cell operating principle. An LSTM is composed by several cells to deal with short and long-term memory. As shown in Figure 2, to predict y_t at timestamp t, an LSTM uses the input data x_t at the current time, the hidden state h_{t-1} at the previous time to memorize short-term past events, and the memory cell state c_{t-1} at the previous time to memorize the long-term past events. An LSTM cell is composed by three gates:

– the *forget gate* filters, through a Sigmoid function σ, data contained in the concatenation of x_t and h_{t-1}. Data are forgotten for values close to 0 and are memorized for values close to 1. The *forget gate* model is:

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \tag{1}$$

– the *input gate* extracts relevant information from $[h_{t-1}, x_t]$ by applying a Sigmoid σ and a Tanh function. The *input gate* is represented by the following:

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_f) \qquad \tilde{C}_t = tanh(W_c.[h_{t-1}, x_t] + b_c). \tag{2}$$

The memory cell c_t is updated from the *forget gate* f_t and the *input gate* i_t and $\tilde{C}_t$, to memorize pertinent data:

$$c_t = f_t \times c_{t-1} + i_t \times \tilde{C}_t \tag{3}$$

– the *output gate* defines the next hidden state h_t containing information about previous inputs. The hidden state h_t is updated with the memory cell c_t normalized by a Tanh function and $[h_{t-1}, x_t]$ normalized by a Sigmoid function:

$$h_t = \sigma(W_h.[h_{t-1}, x_t] + b_h) \times tanh(c_t) \tag{4}$$

with $W_{(.)}$ and $b_{(.)}$, the different gate weights and biases.

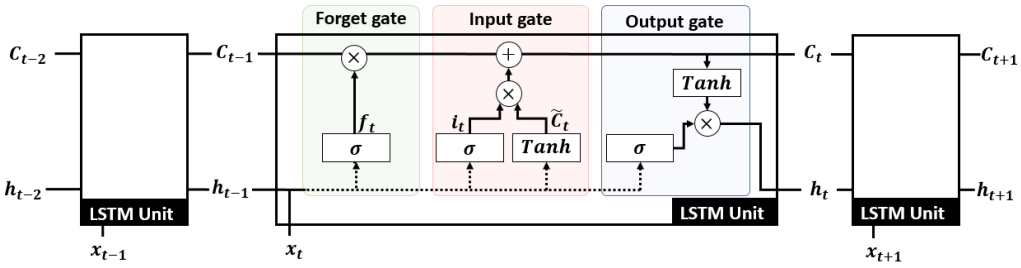

Figure 2. LSTM cell operating principle composed by three gates.

Currently, few works have appied recurrent networks in the military context. They are commonly used for aircraft navigation [16,17], vehicle trajectory estimation [35], maritime route prediction [36] or human motion prediction [18,37]. It is, however, interesting to

mention [38], which focused on projectile trajectory estimation based on LSTMs trained from incomplete and noisy radar measurements.

3. Problem Formulation

This part presents the projectile fire simulation dataset generated by BALCO (BALlistic COde) [21] and the LSTM input data used to estimate projectile trajectories.

3.1. The Projectile Trajectory Dataset BALCO (BALlistic COde

Results reported in this paper exploit a projectile fire dataset generated by BALCO [21,39], i.e., a high fidelity projectile trajectory simulator based on motion equations with six to seven degrees of freedom and discretized by a seventh order Runge-Kutta method. BALCO enables the consideration of different earth models (flat earth, spherical, ellipsoidal), different atmospheric models (standard atmosphere or defined by the user) or different aerodynamic models (axisymmetric or non-axisymmetric projectiles, aerodynamic coefficients described in correspondence tables or polynomials). BALCO accuracy is validated in comparison to the reference program PRODAS (Projectile Rocket Ordnance Design and Analysis System) by considering different projectile types, various initial conditions and different meteorological conditions.

In order to estimate projectile trajectories, three reference frames are considered. The *local navigation frame* $\boldsymbol{n}$ (black frame in Figure 3) tangent to the Earth and assumed fixed during the projectile flight.

The *body frame* $\boldsymbol{b}$ (red frame in Figure 3), which is an ideal hypothetical frame placed exactly at the projectile gravity center, in which the IMU must be placed, providing perfect inertial measurements.

The *sensor frame* $\boldsymbol{s}$ (green frame in Figure 3) rigidly fixed to the projectile and misaligned with the projectile gravity center, considered as the frame where the inertial measurements are performed.

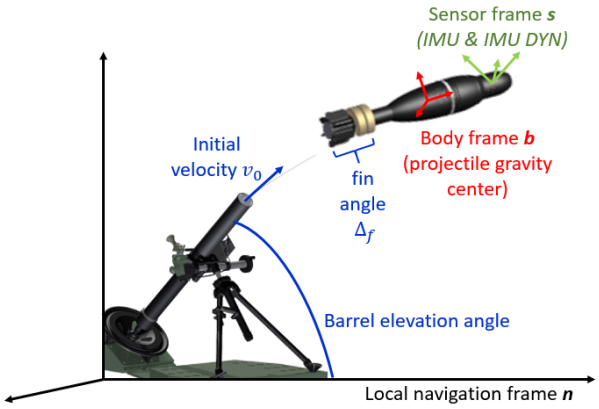

Figure 3. Navigation frames (black—local navigation frame $\boldsymbol{n}$, red—body frame $\boldsymbol{b}$, green—sensor frame $\boldsymbol{s}$) and flight parameters for a finned projectile (fin angle δ_f, initial velocity v_0, barrel elevation angle α).

Results reported in this paper are applied to the estimation of a finned mortar trajectory. The finned projectile dataset, generated by BALCO, includes 5000 fire simulations and where each one includes:

- the **inertial measurements in the body frame $\boldsymbol{b}$ and in the sensor frame $\boldsymbol{s}$**, i.e., gyrometer $\omega \in \mathbb{R}^3$, accelerometer $a \in \mathbb{R}^3$ and magnetometer $h \in \mathbb{R}^3$ measurements. Three kinds of inertial measurements are available:
  - the *Perfect IMU measurements* performed in the body frame $\boldsymbol{b}$ (red frame in Figure 3), in the ideal case where all the three inertial sensors are perfectly aligned with the projectile gravity center and where no sensor default model is taken into

account providing ideal inertial measurements, i.e., without any noise or bias. These measurements are not exploited in this work but are necessary to provide realistic inertial data.

– the *IMU measurements* performed in the sensor frame $\boldsymbol{s}$ (green frame in Figure 3): issued from the *Perfect IMU measurements* where a sensor error model is added. This error model, specific to each sensor axis, includes a misalignment, a sensitivity factor, a bias and a noise (assumed zero mean white Gaussian noise). Thus, this measurement accurately models an IMU embedded in a finned projectile.
– the *IMU DYN measurements* performed in the sensor frame $\boldsymbol{s}$ (green frame in Figure 3): issued from *IMU measurements* to which a transfer function is added to each sensor. For each sensor, *IMU DYN measurements* are modeled by:

$$y_{sensor,IMU\ DYN} = \frac{1}{1 + as + bs^2} y_{sensor,IMU} \tag{5}$$

with $y_{sensor,IMU}$ the *IMU measurements* of the considered sensor, $y_{sensor,IMU\ DYN}$ the corresponding *IMU DYN measurements* and with a and b, the coefficients of the sensor transfer function defined via BALCO. This sensor model allows to model the response of the three sensors over the operating range.

- the **magnetic field reference** $h_n \in \mathbb{R}^3$ in the local navigation frame $\boldsymbol{n}$, assumed constant during the projectile flight.
- **flight parameters** , which are, in the case of a finned projectile, the fin angle δ_f to stabilize projectiles, the initial velocity v_0 at barrel exit and the barrel elevation angle α, relatively important to obtain ballistic trajectories with short ranges.
- a **time vector** $k\Delta_t$ where Δ_t is the IMU sampling period: $\Delta_t = 1 \times 10^{-3} s$.
- the **reference trajectory**, i.e., the projectile position $p \in \mathbb{R}^3$, velocity $v \in \mathbb{R}^3$ and Euler angles $\Psi \in \mathbb{R}^3$ in the local navigation frame $\boldsymbol{n}$ at the IMU frequency. This trajectory is used to evaluate the LSTMs accuracy and to compute errors.

3.2. Data Characteristics and LSTM Requirements

The LSTM predictions at time t are obtained from three-dimensional input data of size $(B_{atch\ size}, S_{eq\ len}, I_{n\ Features})$, with $B_{atch\ size}$ the number of sequences considered, $S_{eq\ len}$ the number of time steps in the sequence and $I_{n\ Features}$ the number of features describing each time step. The input features are $I_{n\ Features} = (\mathcal{M}, \mathcal{P}, \mathcal{T}) \in \mathbb{R}^{16}$, such as the following:

- $\mathcal{M} \in \mathbb{R}^{12}$ *the inertial data*, including *IMU* or *IMU DYN measurements* in the sensor frame $\boldsymbol{s}$ and the reference magnetic field $h_n \in \mathbb{R}^3$ in the local navigation frame $\boldsymbol{n}$ presented in Section 3.1,
- $\mathcal{P} \in \mathbb{R}^3$ *the flight parameters*. In the case of a finned projectile, the three flight parameters are the fin angle δ_f, the initial velocity v_0 and the the barrel elevation angle α.
- $\mathcal{T} \in \mathbb{R}^1$ *the time vector*, such as $\mathcal{T} = k\Delta_t$ with k the considered time step and Δ_t the IMU sampling period.

Various LSTMs are trained and differ depending on the output features learned. Indeed, according to the input data of size $(B_{atch\ size}, S_{eq\ len}, I_{n\ Features})$, LSTMs estimate a projectile trajectory modeled by a vector of size $(B_{atch\ size}, O_{ut\ Features})$ and where $O_{ut\ Features}$ represents the number of output features. The output features $O_{ut\ Features}$ are 9 or 3, depending on the type of LSTM considered. Thus, the following notations are used:

- $LSTM_{ALL}$ trained to estimate 9 output features which are the projectile position $p \in \mathbb{R}^3$, velocity $v \in \mathbb{R}^3$ and Euler angles $\Psi \in \mathbb{R}^3$ in the navigation frame $\boldsymbol{n}$.
- $LSTM_{POS}$ trained to estimate 3 output features,which are the projectile position $p \in \mathbb{R}^3$ expressed in the navigation frame $\boldsymbol{n}$.
- $LSTM_{VEL}$ trained to estimate 3 output features which are the projectile velocity $v \in \mathbb{R}^3$ expressed in the navigation frame $\boldsymbol{n}$.
- $LSTM_{ANG}$ trained to estimate 3 output features which are the projectile Euler angles $\Psi \in \mathbb{R}^3$ in the navigation frame $\boldsymbol{n}$.

4. LSTM Input Data Preprocessing

This section details the two input data pre-processing methods in order to study their influence on estimation accuracy. To manage the different projectile dynamics along the three navigation axes, two data preprocessing methods are investigated: the LSTM input data normalization and local navigation frame rotation allowing to rescale each component of a 3D value on similar variation ranges. To this end, LSTMs presented in Section 3.2 are declined in 8 versions, reported in Table 1, to study the influence of the Min/Max $MM(.)$ and the Standard Deviation $STD(.)$ normalization, and the local navigation frame rotation on estimation accuracy.

Table 1. Version specifications: Influence of the normalization and the local navigation frame rotation.

Name	Normalization	Rotation
V_1	No	No
V_2	$MM(\mathcal{T}), MM(\mathcal{M}), MM(\mathcal{P})$	No
V_3	$MM(\mathcal{T}, \mathcal{M}, \mathcal{P})$	No
V_4	$STD(\mathcal{T}), STD(\mathcal{M}), STD(\mathcal{P})$	No
V_5	$STD(\mathcal{T}, \mathcal{M}, \mathcal{P})$	No
V_6	No	Yes
V_7	$MM(\mathcal{T}, \mathcal{M}, \mathcal{P})$	Yes
V_8	$STD(\mathcal{T}, \mathcal{M}, \mathcal{P})$	Yes

4.1. LSTM Input Data Normalization

Network input data normalization is a preprocessing data approach to rescale input data on similar variation ranges while preserving the same distribution and ratios as the original data. Input data normalization is used to prevent some input data features from having a greater influence than other features during training and to improve gradient backpropagation convergence. In other words, input data with different ranges can lead to lower network estimation performance. The small input values have a small influence during prediction, and therefore the network weights are updated according to the high input values, which can lead to a significant network weight update and therefore a slower network convergence or the network convergence to a local minimum.

According to Table 1, two kinds of normalization are used:

- *Min/Max normalization* $MM(.)$: $x_{MM} = \frac{x - x_{min}}{x_{max} - x_{min}}$ with x_{max} and x_{min} the maximum and minimum of x.
- *Standard Deviation normalization* $STD(.)$: $x_{STD} = \frac{x - \mu_x}{\sigma_x}$ with x the quantity to normalize, $\mu_x = \mu(x)$ its mean and $\sigma_x = \sigma(x)$ its standard deviation. Thus, x_{STD} is a quantity with a zero-mean and a standard deviation of one.

In order to study the impact of input data normalization on estimation accuracy, versions V_2 and V_4 use normalization by features while versions V_3, V_5, V_7 and V_8 use normalization for all features. Moreover, the normalization factors x_{max}, x_{min}, μ_x and σ_x are computed before the training step on the training dataset, as in the following:

$$x_{max} = \frac{1}{N_{sim}} \sum_{i=1}^{N_{sim}} max\{\chi_i\}, \quad x_{min} = \frac{1}{N_{sim}} \sum_{i=1}^{N_{sim}} min\{\chi_i\} \tag{6}$$

$$\mu_x = \frac{1}{N_{sim}} \sum_{i=1}^{N_{sim}} \mu(\chi_i), \quad \sigma_x = \frac{1}{N_{sim}} \sum_{i=1}^{N_{sim}} \mu(\chi_i) \tag{7}$$

with N_{sim} the number of simulation in the training dataset and with χ_i the considered quantities of the simulation n° i, which are $\chi_i = \begin{bmatrix} \mathcal{M} & \mathcal{P} & \mathcal{T} \end{bmatrix}$ for versions V_3, V_5, V_7 and V_8, and $\chi_i = \mathcal{M}$ or $\chi_i = \mathcal{P}$ or $\chi_i = \mathcal{T}$ for versions V_2 and V_4.

4.2. Local Navigation Frame Rotation

The local navigation frame rotation aims to rotate the local navigation frame $\boldsymbol{n}$ by a fixed angle γ (local rotated navigation frame $\boldsymbol{n}_\gamma$), such as $x_\gamma = R_\gamma x$ with $x \in \mathbb{R}^3$ defined in

$\boldsymbol{n}$, $x_\gamma \in \mathbb{R}^3$ expressed in $\boldsymbol{n}_\gamma$ and $R_\gamma \in SO(3)$ the transition matrix from the local navigation frame $\boldsymbol{n}$ to the local rotated navigation frame $\boldsymbol{n}_\gamma$ as in the following:

$$R_\gamma = \begin{bmatrix} cos(\gamma) & -sin(\gamma) & 0 \\ sin(\gamma) & cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos(\gamma) & 0 & sin(\gamma) \\ 0 & 1 & 0 \\ -sin(\gamma) & 0 & cos(\gamma) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\gamma) & -sin(\gamma) \\ 0 & sin(\gamma) & cos(\gamma) \end{bmatrix}. \quad (8)$$

The navigation frame rotation allows a quantity $x \in \mathbb{R}^3$ expressed in the navigation frame $\boldsymbol{n}$ to modify its three components in order to have a similar amplitude order for the three components. The local navigation frame rotation provides similar variation ranges of a quantity along the three axes. This approach is used to ensure that the LSTMs adequately estimate a quantity with small magnitudes along one axis, even though the variations are considerably larger along the other two axes.

As illustrated in Figure 4, the variation range of the projectile position along the y-axis is significantly smaller than along the x and z-axes and thus, the expressed position in the local rotated navigation frame $\boldsymbol{n}_\gamma$ provides similar amplitudes along the three axes. For example, projectile position variation ranges along the x and z-axes are around several kilometers, while the position along the y-axis varies by a few meters. As illustrated in Figure 4, expressed position in the rotated navigation frame $\boldsymbol{n}_\gamma$ provides similar amplitudes along the three axes.

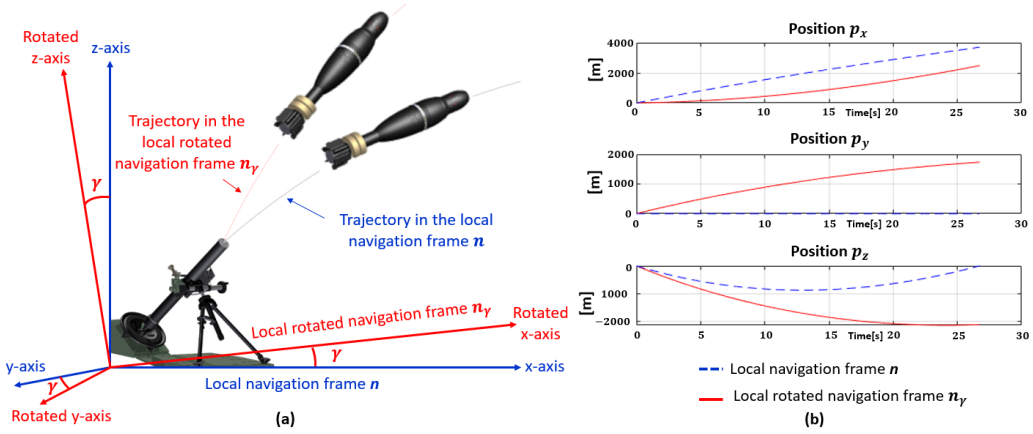

Figure 4. (**a**) Local navigation frame $\boldsymbol{n}$ and local rotated navigation frame $\boldsymbol{n}_\gamma$. (**b**) Projectile position in the local navigation frame $\boldsymbol{n}$ (blue dashed line), projectile position in the local rotated navigation frame $\boldsymbol{n}_\gamma$ (red solid line).

All quantities expressed in the local navigation frame are rotated, i.e., the projectile position p, velocity v and Euler angles Ψ. Moreover, the angle γ is fixed for all trajectories in the dataset and is determined according to the data used in this paper and is the same to express the position, velocity or Euler angles in the rotated navigation frame. This angle is determined according to the data used in this paper in particular to have similar magnitudes for the three positions, as for the velocity. During the training step, labels are expressed in the local rotated navigation frame $\boldsymbol{n}_\gamma$ and LSTMs predict trajectories in $\boldsymbol{n}_\gamma$. During the test step, LSTMs estimate projectile trajectories in the local rotated navigation frame $\boldsymbol{n}_\gamma$, and then, estimations are moved back to the initial local navigation frame $\boldsymbol{n}$.

5. Results and Analysis

This part of the paper reports LSTM results applied to finned projectiles. A first section focuses on the influence of the normalization and the local navigation frame rotation on the estimation accuracy for short training. A second section validates LSTMs on a large dataset by focusing on the impact of the local navigation frame rotation and the IMU model.

The LSTMs' performances are evaluated in comparison to a classical navigation algorithm, i.e., a Dead-Reckoning. This algorithm integrates gyrometer ω and accelerometer a measurements to estimate at each discrete time k:

$$\begin{aligned} R_k &= R_{k-1}[\omega_k \Delta_t]_\times, \\ v_k &= v_{k-1} + (R_{k-1}a_k + g)\Delta_t, \\ p_k &= p_{k-1} + v_{k-1}\Delta_t + \tfrac{1}{2}(R_{k-1}a_k + g)\Delta_t^2, \end{aligned} \tag{9}$$

with $R_k \in SO(3)$ the rotation matrix from the sensor frame $\boldsymbol{s}$ to the local navigation frame $\boldsymbol{n}$, $g \in \mathbb{R}^3$ the constant gravity vector, $p_k \in \mathbb{R}^3$ and $v_k \in \mathbb{R}^3$, respectively, the projectile position and velocity, and $[.]_\times$ the skew matrix. This algorithm is generally used for Kalman filters for the prediction step to estimate trajectory, as presented in [2–4,8,24,40].

5.1. Impact of the Input Data Normalization and the Local Navigation Frame Rotation

This section reports the estimated trajectories of a finned projectile according to $LSTM_{ALL}$, $LSTM_{POS}$, $LSTM_{VEL}$ and $LSTM_{ANG}$, each declined in the 8 versions V_1–V_8 presented in Section 3.2. The training parameters are summarized in the Table 2.

Table 2. Training characteristics of $LSTM_{ALL,\ POS,\ VEL,\ ANG,\ V_{1-8}}$.

Dataset	Training Dataset: Test Dataset:	100 Simulations (Validation: 10 Simulations) 20 Simulations
Input data	Batch size: Input data :	64 ($S_{eq\ len}$: 20 timesteps) $I_{n\ Features} = (\mathcal{M}, \mathcal{P}, \mathcal{T}) \in \mathbb{R}^{16}$ (with *IMU measurements*)
Training	Cost function: Optimization algorithm: LSTM layer:	Mean Squared Error (MSE) ADAM [41] (Learning rate : 1×10^{-4}) 2 (Hidden units: 64–128)

$S_{eq\ len}$ corresponds to 20 samples representing a window of $0.02s$ as the sensor sampling period is $\Delta_t = 1 \times 10^{-3}s$. This parameter is adjusted according to the input data used.

5.1.1. Qualitative Validation: One Finned Projectile Fire Simulation

Figure 5 focuses on the estimated positions and orientation for one projectile shot. For readability reasons, three estimation methods are first compared: the Dead-Reckoning (9), $LSTM_{ALL,V_1}$ and $LSTM_{ALL,V_6}$ (local navigation frame rotation).

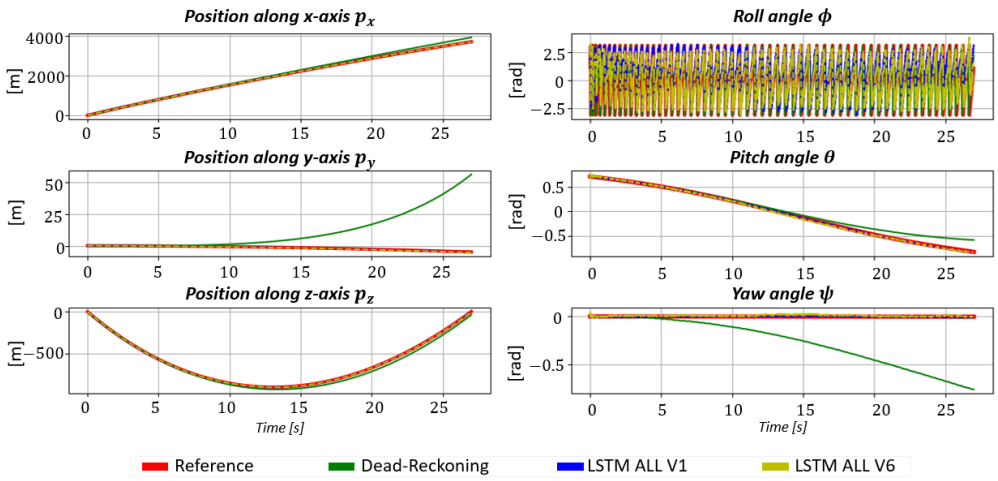

Figure 5. Estimated projectile position [m] and Euler angles [rad] obtained by the Dead-Reckoning (green), $LSTM_{ALL,\ V_1}$ (blue), $LSTM_{ALL,\ V_6}$ (yellow) and the reference (red).

As shown in Figure 5, positions estimated by the LSTMs are significantly more accurate than the Dead-Reckoning. Nevertheless, LSTMs are only accurate in estimating the

pitch and yaw angle. Errors in the roll angle estimation are due to the finned projectile rotation rate. The LSTMs fail to fully capture the roll angle dynamics. Moreover, the local navigation frame rotation improves projectile position estimation but slightly degrades pitch angle estimation.

5.1.2. Quantitative Evaluation: Analysis on the Whole Test Dataset

To validate the previous observations, $LSTM_{ALL,\ POS,\ VEL,\ ANG,\ V_{1-8}}$ are evaluated on the N_{sim} simulations in the test dataset according to two criteria based on the Root Mean Square Error (RMSE) defined as $RMSE_x = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\left(x_{k,ref} - \hat{x}_k\right)^2}$, with $\hat{x}$ the estimate, x_{ref} the reference and N the number of samples for one simulation.

The two evaluation criteria are as follows:

- *Success Rate* $\mathcal{C}_1$*:* number of simulations where a LSTM RMSE is strictly smaller than the Dead-Reckoning.
- *Error Rate* $\mathcal{C}_2$*:* percentage of LSTM error compared to Dead-Reckoning errors.

$$\mathcal{C}_1 = \sum_{k=1}^{N_{sim}} RMSE_{LSTM} < RMSE_{DR}, \qquad \mathcal{C}_2 = \frac{100}{N_{sim}} \sum_{k=1}^{N_{sim}} \frac{RMSE_{LSTM}}{RMSE_{LSTM} + RMSE_{DR}} \tag{10}$$

The position, velocity and orientation success rate $\mathcal{C}_1$ and the error rates $\mathcal{C}_2$ are presented in Figures 6–8. Figures 5–8 has been modified for better readability.

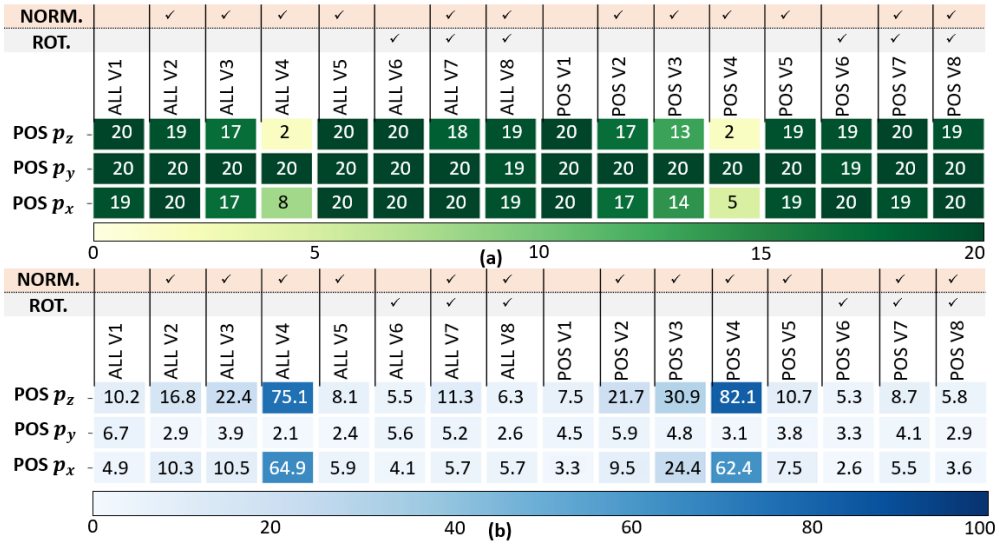

Figure 6. Position analysis: (**a**) Success Rate $\mathcal{C}_1$, (**b**) Error Rate $\mathcal{C}_2$ (in %) of $LSTM_{ALL,\ V_1-V_8}$ and $LSTM_{POS,\ V_1-V_8}$.

Position analysis results (see Figure 6): The LSTMs outperform Dead-Reckoning for position estimation, especially along the y-axis. Normalizations affect position estimates differently as $STD(\mathcal{T}), STD(\mathcal{M}), STD(\mathcal{P})$ (V_4) and $MM(\mathcal{T}, \mathcal{M}, \mathcal{P})$ (V_3) are not appropriate to this application. In addition, the normalization leads to less accuracy as it implies a loss of information. Finally, rotating the local navigation frame improves the accuracy according to $\mathcal{C}_2$ along the three axis.

Veclocity analysis results (see Figure 7): As previously, the LSTMs clearly outperform Dead-Reckoning for velocity estimation. Specialized networks $LSTM_{VEL}$ are a bit better than $LSTM_{ALL}$. The STD normalization for all features V_5 exhibits the best results among the different normalization options investigated, especially for velocity along the z-axis. Moreover, rotating the local navigation frame V_6 significantly improves the projectile velocity estimation along all the three axes.

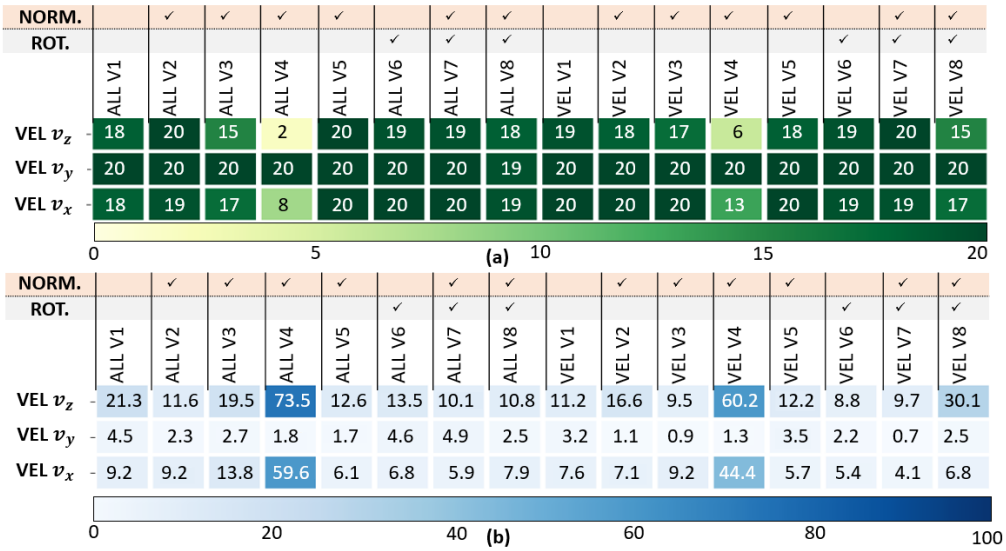

Figure 7. Velocity analysis: (**a**) Success Rate $\mathcal{C}_1$, (**b**) Error Rate $\mathcal{C}_2$ (in %) of $LSTM_{ALL,\ V_1-V_8}$ and $LSTM_{VEL,\ V_1-V_8}$.

Euler angles analysis results (see Figure 8): The LSTMs deteriorate the roll ϕ angle estimation compared to the Dead-Reckoning, but accurately estimate the yaw angle ψ. As previously, the $STD(\mathcal{T},\mathcal{M},\mathcal{P})$ (V_5) normalization of $LSTM_{ALL}$ exhibits the best performances for the three Euler angles estimation as well as the local navigation frame rotation.

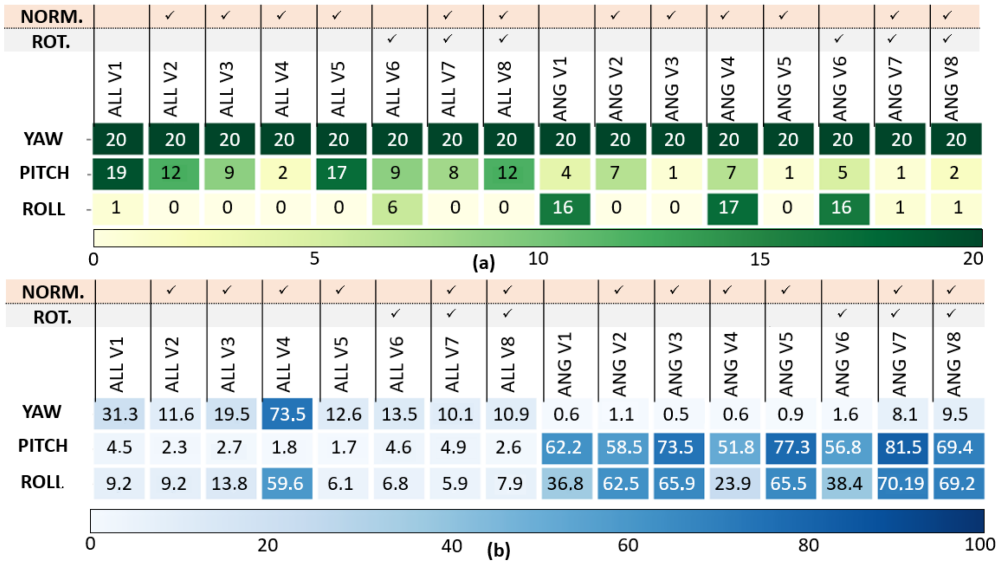

Figure 8. Orientation analysis: (**a**) Success Rate $\mathcal{C}_1$, (**b**) Error Rate $\mathcal{C}_2$ (in %) of $LSTM_{ALL,\ V_1-V_8}$ and $LSTM_{ANG,\ V_1-V_8}$.

In summary, the LSTM end-to-end estimation is particularly appropriate for projectile position and velocity estimation. Moreover, from this evaluation study, it can be concluded that specialized networks do not significantly improve the estimation accuracy and $STD(\mathcal{T},\mathcal{M},\mathcal{P})$ (V_5) normalization is more appropriate to estimate a projectile trajectory compared to other normalizations. Finally, rotating the local navigation frame is an efficient method to optimize projectile position and velocity estimation.

5.2. Impact of Inertial Measurement Type and Local Navigation Frame Rotation on Estimation Accuracy

The dataset presented in Section 3.1 contains two kinds of inertial readings; *IMU measurement*, used so far, and *IMU DYN measurement*, where sensors are characterized by a 2nd order model.

This section focuses on the impact of inertial data and navigation frame rotation on LSTM estimation accuracy. To this end, four LSTMs are trained to estimate positions, velocities and orientations of a finned projectile: $LSTM_{IMU,\ V_1}$ (no rotation) , $LSTM_{IMU\ DYN,\ V_1}$ (no rotation), $LSTM_{IMU,\ V_6}$ (navigation frame rotation) and $LSTM_{IMU\ DYN,\ V_6}$(navigation frame rotation). LSTM specifications are given in Table 3.

Table 3. Training characteristics of $LSTM_{IMU,\ V_1}$, $LSTM_{IMU\ DYN,\ V_1}$, $LSTM_{IMU,\ V_6}$, $LSTM_{IMU\ DYN,\ V_6}$.

Dataset	Training Dataset: Test Dataset:	4000 Simulations (Validation: 400 Simulations) 400 Simulations
LSTM name	No normalization & No rotation	$LSTM_{IMU,\ V_1}$ (with *IMU measurements*) $LSTM_{IMU\ DYN,\ V_1}$ (with *IMU DYN measurements*)
	No normalization & Rotation	$LSTM_{IMU,\ V_6}$ (with *IMU measurements*) $LSTM_{IMU\ DYN,\ V_6}$ (with *IMU DYN measurements*)
Input data	Batch size: Input data :	64 ($S_{eq\ len}$: 20 timestamp) $I_{n\ Features} = (\mathcal{M}, \mathcal{P}, \mathcal{T}) \in \mathbb{R}^{16}$
Training	Cost function: Optimization algorithm: LSTM layer:	Mean Squared Error (MSE) ADAM [41] (Learning rate: 1×10^{-4}) 2 (Hidden units: 64–128)

5.2.1. Impact of the Local Navigation Frame Rotation and IMU Measurement

This section focuses on $LSTM_{IMU,\ V_1}$ (no rotation), $LSTM_{IMU,\ V_6}$ (rotation) and the Dead-Reckoning algorithm (9) for position, velocity and Euler angles estimation. Network characteristics are presented in Table 3.

Figure 9 presents the average error distributions $\bar{e}$ and the corresponding standard deviations σ evaluated for positions and Euler angles, with the three estimation methods considered, such as

$$\bar{e} = \frac{1}{N}\sum_{k=1}^{N}\left(x_{ref} - \widehat{x}\right), \tag{11}$$

$$\sigma = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\left(\left[x_{ref} - \widehat{x}\right] - \bar{e}\right)^2} \quad , \tag{12}$$

where x_{ref} is the reference, $\widehat{x}$ the estimate and N the number of samples in the considered simulation.

According to Figure 9, the Dead-Reckoning mean error dispersion (red) is very large compared to those of LSTMs (green and blue). Thus, LSTMs are perfectly adapted to estimate finned projectile positions and velocities. Focusing on p_z, v_x and v_z, the $LSTM_{IMU,\ V_1}$ average errors are not centered on zero compared to $LSTM_{IMU,\ V_6}$ (rotation). Thus, the local navigation frame rotation improves these estimates. As previously observed, the LSTMs fail to estimate the projectile roll angle even if the finned projectile rotation rate is low. Furthermore, the centering and dispersion of angle errors indicate that the LSTMs suffer to estimate the projectile orientation despite the yaw angle accuracy.

5.2.2. Impact of the Local Navigation Frame Rotation and IMU DYN Measurement

Figure 10 presents the average error $\bar{e}$ (12) distributions for positions, velocities, and Euler angles estimated by $LSTM_{IMU\ DYN,\ V_1}$ (no rotation), $LSTM_{IMU\ DYN,\ V_6}$ (rotation), and the Dead-Reckoning (9).

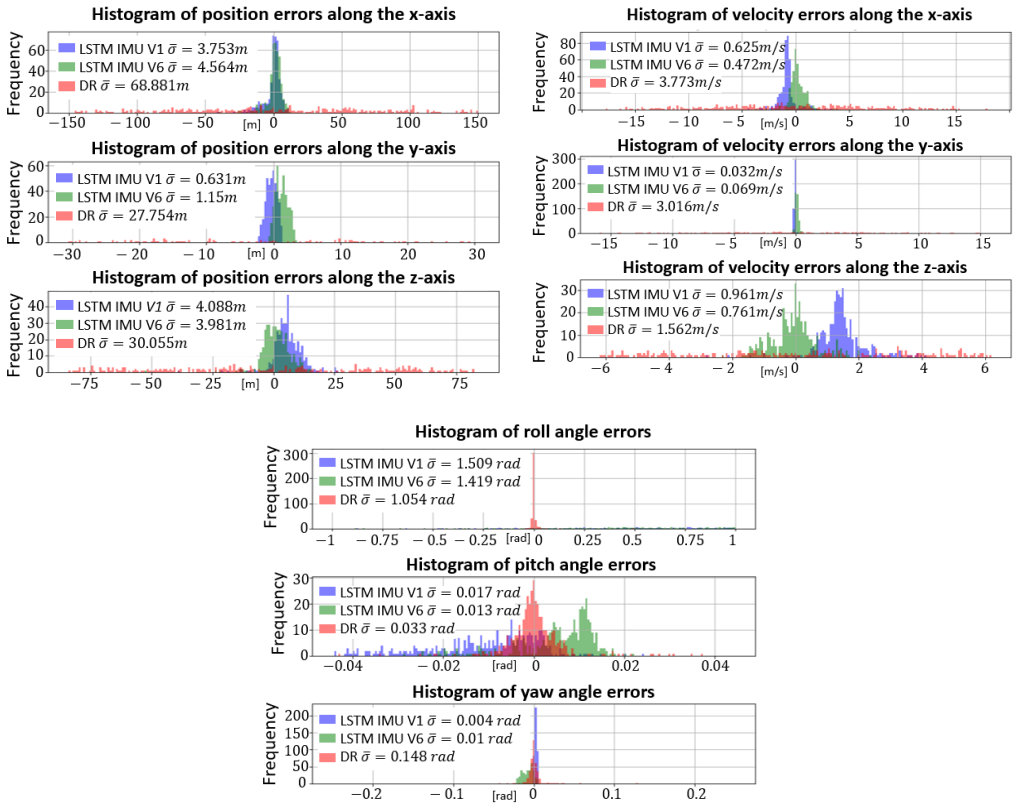

Figure 9. Average position, velocity and orientation error histogram obtained by $LSTM_{IMU,\ V_1}$ (blue), $LSTM_{IMU,\ V_6}$ (green) and Dead-Reckoning (red).

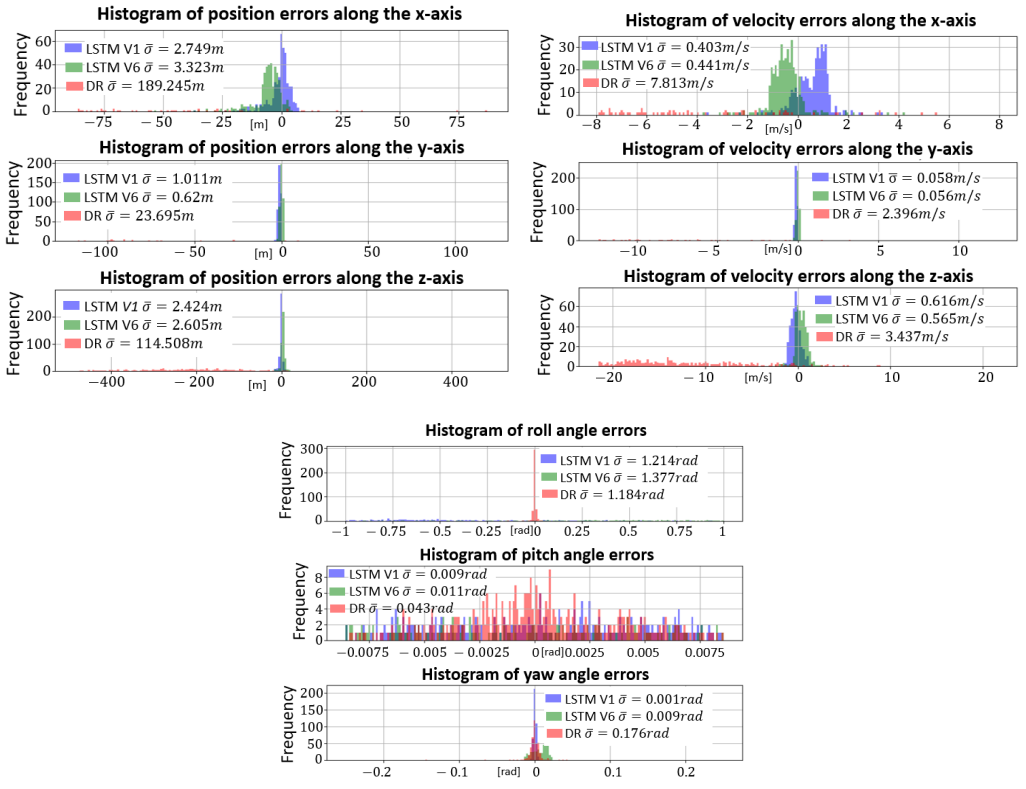

Figure 10. Average position, velocity and orientation error histogram obtained by $LSTM_{IMU\ DYN,\ V_1}$ (blue), $LSTM_{IMU\ DYN,\ V_6}$ (green) and Dead-Reckoning (red).

According to Figure 10, the LSTMs accurately estimate the projectile position and velocity over a large dataset, despite dynamic inertial data causing, in contrast, the Dead-Reckoning divergence. Furthermore, the error distribution centering analysis allows to conclude that the local navigation frame rotation improves the estimation of p_y, v_x and v_y. As for previous orientation estimation results, both LSTMs fail to estimate the projectile roll angle ϕ.

5.2.3. Evaluation Metric

The performance of $LSTM_{IMU\ V_1}$, $LSTM_{IMU\ V_6}$, $LSTM_{IMU\ DYN\ V_1}$, $LSTM_{IMU\ DYN\ V_6}$, and the Dead-Reckoning (9) are evaluated using two evaluation criteria computed for each simulation in the test dataset:

- *Mean Absolute Error:*

$$MAE = \frac{1}{N}\sum_{k=1}^{N}|x - \hat{x}| \tag{13}$$

  with x the reference, $\hat{x}$ the estimate N, the number of samples.
- *SCORE:* Number of simulations in the test dataset where the considered method obtains the smallest RMSE.

The average of each criteria are evaluated on the dataset as:

$$\mathcal{C}_\chi = \frac{1}{N_{sim}}\sum_{k=1}^{N_{sim}}\chi_k \tag{14}$$

with χ the selected evaluation criterion and N_{sim} the number of simulations in the test dataset.

$\mathcal{C}_{MAE}$ Criterion analysis: Figure 11 presents the MAE average $\mathcal{C}_{MAE}$, evaluated on the whole test dataset for $LSTM_{IMU,\ V_1}$, $LSTM_{IMU,\ V_6}$, $LSTM_{IMU\ DYN,\ V_1}$, $LSTM_{IMU\ DYN,\ V_6}$ and the Dead-Reckoning.

The LSTMs accurately estimate position and velocity, both with *IMU* and *IMU DYN measurement*, with errors around a few meters for the positions. This criterion confirms that the local navigation frame rotation improves p_z, v_x and v_z estimation for LSTMs trained with *IMU measurement*, and p_y, v_x and v_y estimation for LSTMs trained with *IMU DYN measurement*. As expected, LSTMs are not adapted to estimate the projectile roll angle contrary to the pitch and the yaw angle.

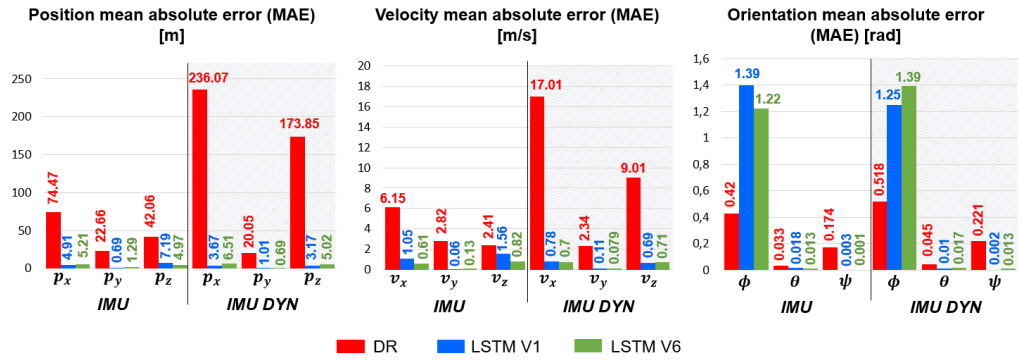

Figure 11. Position, velocity and orientation MAE average $\mathcal{C}_{MAE}$ obtained by $LSTM_{IMU,\ V_1}$, $LSTM_{IMU,\ V_6}$, $LSTM_{IMU\ DYN,\ V_1}$, $LSTM_{IMU\ DYN,\ V_6}$ and the Dead-Reckoning.

$\mathcal{C}_{SCORE}$ Criterion analysis: Figure 12 presents the score, $\mathcal{C}_{SCORE}$, evaluated on the whole test dataset for $LSTM_{IMU,\ V_1}$, $LSTM_{IMU,\ V_6}$, $LSTM_{IMU\ DYN,\ V_1}$, $LSTM_{IMU\ DYN,\ V_6}$ and the Dead-Reckoning.

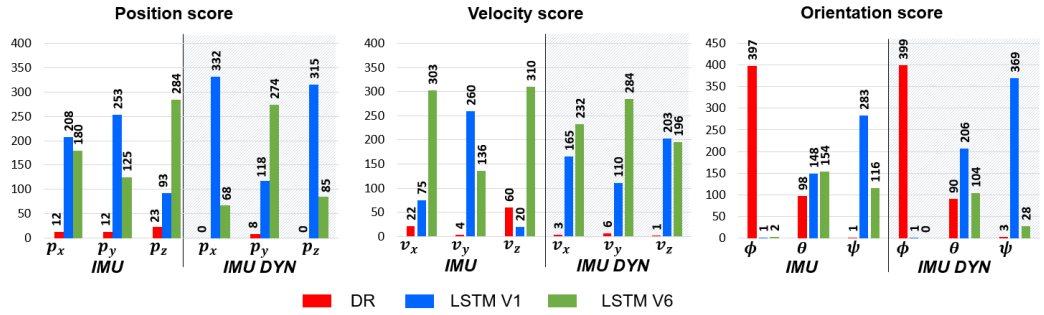

Figure 12. Position, velocity and orientation score obtained by $LSTM_{IMU,\,V_1}$, $LSTM_{IMU,\,V_6}$, $LSTM_{IMU\ DYN,\,V_1}$, $LSTM_{IMU\ DYN,\,V_6}$ and the Dead-Reckoning.

According to Figure 12, an LSTM is an accurate approach to estimate projectile position and velocity in a GNSS-denied environment. However, a LSTM is not the optimal method for the orientation estimation. Furthermore, an LSTM is able to generalize the learned features over a large projectile fire dataset as well as learn and predict trajectories from different sensor models. Finally, $\mathcal{C}_{MAE}$ and $\mathcal{C}_{SCORE}$ analysis confirms that the local navigation frame rotation is an appropriate method to optimize p_z, v_x and v_z for LSTMs trained with *IMU measurement*, and p_y, v_x, and v_y for LSTMs trained with *IMU DYN measurement*.

5.2.4. Errors at Impact Point

This section focuses on the errors at the impact point, i.e., position errors (p_x, p_y) at the final time of a shot. Figure 13a shows the impact point errors of LSTMs with *IMU* and *IMU DYN measurements* and the Dead-Reckoning. Figure 13b presents where the impact point errors are located in different error zones.

Whatever the inertial sensor model, The Dead-Reckoning impact point errors are greater than 100 m, contrary to LSTMs. Focusing on *IMU measurements*, the majority of $LSTM_{IMU,\,V_6}$ (rotation) impact point errors are less than 5 m contrary to $LSTM_{IMU,\,V_1}$ (no rotation) where errors are lower than 20 m. Thus, the local navigation frame rotation allows to strongly reduce errors at the impact point. Focusing on *IMU DYN measurements*, the local navigation frame rotation deteriorates estimation accuracy. Indeed, 266 simulations have impact point errors less than 5 m with $LSTM_{IMU\ DYN,\,V_1}$ while 180 simulations have impact point errors less than 5 m with $LSTM_{IMU\ DYN,\,V_6}$.

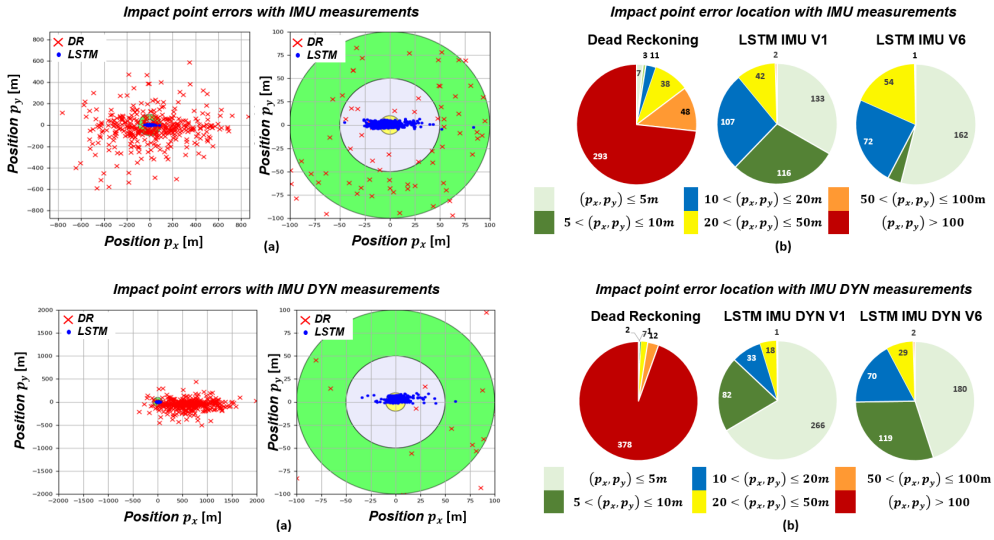

Figure 13. (**a**) Errors at impact point obtained by $LSTM_{IMU,\,V_1}$ and $LSTM_{IMU\ DYN,\,V_1}$ (blue cross), $LSTM_{IMU,\,V_6}$ and $LSTM_{IMU\ DYN,\,V_6}$ (green cross) and the Dead-Reckoning (red dot). (**b**) Impact point error location.

In summary, an LSTM is an accurate approach to estimate projectile position as errors are less than ten meters in a GNSS-denied environment. These results are comparable to those obtained by commercial GNSS-guided mortars. Moreover, the local navigation frame rotation is useful with *IMU measurement* and allows to minimize position errors.

6. Conclusions

This paper presents a deep learning approach to estimate a gun-fired projectile trajectory in a GNSS-denied environment. Long-Short-Term-Memories (LSTMs) are trained from the embedded IMU, the magnetic field reference, flight parameters specific to the projectile (the fin angle, the initial velocity, the barrel elevation angle) and a time vector. The impact of three preprocessing methods are analyzed: the input data normalizations, the local navigation frame rotation and the inertial sensor model. According to the reported results, the LSTMs accurately estimate projectile positions and velocities compared to a conventional navigation algorithm as errors are around ten meters, similar to the GNSS-guided projectile accuracy. Nevertheless, LSTM suffers in the estimation of the projectile orientation, especially for the high dynamic roll angle. The input data normalization provides no interesting results while the local navigation frame rotation optimizes the position and velocity estimation. Moreover, the results prove that the LSTMs generalize the learned features on large datasets independently of the inertial sensor model considered. Based on results reported in this paper, the next step is to implement a deep Kalman filter by considering the LSTM predictions as observations.

Author Contributions: Methodology, A.R.; Validation, A.R.; Investigation, A.R.; Writing—original draft, A.R.; Supervision, S.C., J.W. and J.-P.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Groves, P.D. Principles of GNSS, inertial, and multisensor integrated navigation systems. *IEEE Aerosp. Electron. Syst. Mag.* **2015**, *30*, 26–27. [CrossRef]
2. Zhao, H.; Li, Z. Ultra-tight GPS/IMU integration based long-range rocket projectile navigation. *Def. Sci. J.* **2016**, *66*, 64–70. [CrossRef]
3. Fairfax, L.D.; Fresconi, F.E. Loosely-coupled GPS/INS state estimation in precision projectiles. In Proceedings of the 2012 IEEE/ION Position, Location and Navigation Symposium, Myrtle Beach, SC, USA, 23–26 April 2012; pp. 620–624.
4. Wells, L.L. The projectile GRAM SAASM for ERGM and Excalibur. In Proceedings of the IEEE 2000. Position Location and Navigation Symposium (Cat. No. 00CH37062), San Diego, CA, USA, 13–16 March 2000; pp. 106–111.
5. Duckworth, G.L.; Baranoski, E.J. Navigation in GNSS-denied environments: Signals of opportunity and beacons. In Proceedings of the NATO Research and Technology Organization (RTO) Sensors and Technology Panel (SET) Symposium, 2007 ; pp. 1–14.
6. Ruegamer, A.; Kowalewski, D. Jamming and spoofing of gnss signals–an underestimated risk?! *Proc. Wisdom Ages Chall. Mod. World* **2015**, *3*, 17–21.
7. Schmidt, D.; Radke, K.; Camtepe, S.; Foo, E.; Ren, M. A survey and analysis of the GNSS spoofing threat and countermeasures. *ACM Comput. Surv. (CSUR)* **2016**, *48*, 1–31. [CrossRef]
8. Roux, A.; Changey, S.; Weber, J.; Lauffenburger, J.P. Projectile trajectory estimation: Performance analysis of an Extended Kalman Filter and an Imperfect Invariant Extended Kalman Filter. In Proceedings of the 2021 9th International Conference on Systems and Control (ICSC), Caen, France, 24–26 November 2021; pp. 274–281.
9. Combettes, C.; Changey, S.; Adam, R.; Pecheur, E. Attitude and velocity estimation of a projectile using low cost magnetometers and accelerometers. In Proceedings of the 2018 IEEE/ION Position, Location and Navigation Symposium (PLANS), Monterey, CA, USA, 23–26 April 2018; pp. 650–657.
10. Fiot, A.; Changey, S.; Petit, N. Attitude estimation for artillery shells using magnetometers and frequency detection of accelerometers. *Control. Eng. Pract.* **2022**, *122*, 105080. [CrossRef]

11. Tarraf, D.C.; Shelton, W.; Parker, E.; Alkire, B.; Carew, D.G.; Grana, J.; Levedahl, A.; Léveillé, J.; Mondschein, J.; Ryseff, J.; et al. *The Department of Defense Posture for Artificial Intelligence: Assessment and Recommendations*; Technical Report; Rand National Defense Research Institute: Santa Monica, CA, USA, 2019.
12. Zeldam, S. Automated Failure Diagnosis in Aviation Maintenance Using Explainable Artificial Intelligence (XAI). Master's Thesis, University of Twente, Enschede, The Netherlands, 2018.
13. Ustun, V.; Kumar, R.; Reilly, A.; Sajjadi, S.; Miller, A. Adaptive synthetic characters for military training. *arXiv* **2021**, arXiv:2101.02185.
14. Svenmarck, P.; Luotsinen, L.; Nilsson, M.; Schubert, J. 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, Bordeaux, France, 30 June–1 July 2018; pp. 1–16.
15. Ventre, D. *Artificial Intelligence, Cybersecurity and Cyber Defence*; John Wiley & Sons: Hoboken, NJ, USA, 2020.
16. Shi, Z.; Xu, M.; Pan, Q.; Yan, B.; Zhang, H. LSTM-based flight trajectory prediction. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), 8–13 July 2018; pp. 1–8.
17. Gaiduchenko, N.E.; Gritsyk, P.A.; Malashko, Y.I. Multi-Step Ballistic Vehicle Trajectory Forecasting Using Deep Learning Models. In Proceedings of the 2020 International Conference Engineering and Telecommunication (En&T), Dolgoprudny, Russia, 25–26 November 2020; pp. 1–6.
18. Al-Molegi, A.; Jabreel, M.; Ghaleb, B. STF-RNN: Space time features-based recurrent neural network for predicting people next location. In Proceedings of the 2016 IEEE Symposium Series on Computational Intelligence (SSCI), Athens, Greece, 6–9 December 2016; pp. 1–7.
19. Sherstinsky, A. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Phys. D Nonlinear Phenom.* **2020**, *404*, 132306. [CrossRef]
20. Yu, Y.; Si, X.; Hu, C.; Zhang, J. A review of recurrent neural networks: LSTM cells and network architectures. *Neural Comput.* **2019**, *31*, 1235–1270. [CrossRef] [PubMed]
21. Wey, P.; Corriveau, D.; Saitz, T.A.; de Ruijter, W.; Strömbäck, P. BALCO 6/7-DoF trajectory model. In Proceedings of the 29th International Symposium on Ballistics, Edinburgh, Scotland, 9–13 May 2016; Volume 1, pp. 151–162.
22. Liu, F.; Su, Z.; Zhao, H.; Li, Q.; Li, C. Attitude measurement for high-spinning projectile with a hollow MEMS IMU consisting of multiple accelerometers and gyros. *Sensors* **2019**, *19*, 1799. [CrossRef] [PubMed]
23. Rogers, J.; Costello, M. Design of a roll-stabilized mortar projectile with reciprocating canards. *J. Guid. Control Dyn.* **2010**, *33*, 1026–1034. [CrossRef]
24. Jardak, N.; Adam, R.; Changey, S. A Gyroless Algorithm with Multi-Hypothesis Initialization for Projectile Navigation. *Sensors* **2021**, *21*, 7487. [CrossRef] [PubMed]
25. Radi, A.; Zahran, S.; El-Sheimy, N. GNSS Only Reduced Navigation System Performance Evaluation for High-Speed Smart Projectile Attitudes Estimation. In Proceedings of the 2021 International Telecommunications Conference (ITC-Egypt), Alexandria, Egypt, 13–15 July 2021; pp. 1–4.
26. Aykenar, M.B.; Boz, I.C.; Soysal, G.; Efe, M. A Multiple Model Approach for Estimating Roll Rate of a Very Fast Spinning Artillery Rocket. In Proceedings of the 2020 IEEE 23rd International Conference on Information Fusion (FUSION), Rustenburg, South Africa, 6–9 July 2020; pp. 1–7.
27. Changey, S.; Pecheur, E.; Bernard, L.; Sommer, E.; Wey, P.; Berner, C. Real time estimation of projectile roll angle using magnetometers: In-flight experimental validation. In Proceedings of the 2012 IEEE/ION Position, Location and Navigation Symposium, Myrtle Beach, SC, USA, 23–26 April 2012; pp. 371–376.
28. Zhao, H.; Su, Z. Real-time estimation of roll angle for trajectory correction projectile using radial magnetometers. *IET Radar Sonar Navig.* **2020**, *14*, 1559–1570. [CrossRef]
29. Changey, S.; Beauvois, D.; Fleck, V. A mixed extended-unscented filter for attitude estimation with magnetometer sensor. In Proceedings of the 2006 American Control Conference, Minneapolis, MN, USA, 14–16 June 2006; p. 6.
30. Schmidt, G.T. Navigation sensors and systems in GNSS degraded and denied environments. *Chin. J. Aeronaut.* **2015**, *28*, 1–10. [CrossRef]
31. Fiot, A.; Changey, S.; Petit, N.C. Estimation of air velocity for a high velocity spinning projectile using transerse accelerometers. In Proceedings of the 2018 AIAA Guidance, Navigation, and Control Conference, Kissimmee, FL, USA, 8–12 January 2018; p. 1349.
32. Changey, S.; Fleck, V.; Beauvois, D. Projectile attitude and position determination using magnetometer sensor only. In *Proceedings of the Intelligent Computing: Theory and Applications III*; SPIE: Orlando, FL, USA, 2005; Volume 5803, pp. 49–58.
33. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
34. Hochreiter, S.; Bengio, Y.; Frasconi, P.; Schmidhuber, J. Gradient flow in recurrent nets: The difficulty of learning long-term dependencies. In *A Field Guide to Dynamical Recurrent Neural Networks*; IEEE Press: Piscataway, NJ, USA, 2001.
35. Park, S.H.; Kim, B.; Kang, C.M.; Chung, C.C.; Choi, J.W. Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder-decoder architecture. In Proceedings of the 2018 IEEE intelligent vehicles symposium (IV), Changshu, China, 26–30 June 2018; pp. 1672–1678.
36. Sørensen, K.A.; Heiselberg, P.; Heiselberg, H. Probabilistic maritime trajectory prediction in complex scenarios using deep learning. *Sensors* **2022**, *22*, 2058. [CrossRef] [PubMed]

37. Barsoum, E.; Kender, J.; Liu, Z. Hp-gan: Probabilistic 3d human motion prediction via gan. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Salt Lake City, UT, USA, 18–22 June 2018; pp. 1418–1427.
38. Hou, L.h.; Liu, H.j. An end-to-end lstm-mdn network for projectile trajectory prediction. In Proceedings of the Intelligence Science and Big Data Engineering, Big Data and Machine Learning: 9th International Conference, IScIDE 2019, Nanjing, China, 17–20 October 2019; pp. 114–125.
39. Corriveau, D. Validation of the NATO Armaments Ballistic Kernel for use in small-arms fire control systems. *Def. Technol.* **2017**, *13*, 188–199. [CrossRef]
40. Roux, A.; Changey, S.; Weber, J.; Lauffenburger, J.P. Cnn-based invariant extended kalman filter for projectile trajectory estimation using imu only. In Proceedings of the 2021 International Conference on Control, Automation and Diagnosis (ICCAD), Grenoble, France, 3–5 November 2021; pp. 1–6.
41. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.