Journal of Machine Learning and Information Security
https://www.sciltp.com/journals/jmlis

Article

Phy-Ture: A Physics-Guided Fusion and FeaTure Processing Framework for Unmanned Helicopter Dynamics Modeling

Bohan Zhang, Chao Chen, Jianglan Fu, Haiwei Chen, Yiyang Ye, Beichen Shao, Mingyan Li, Quanwang Wu and Hongyu Huang *

College of Computer Science, Chongqing University, Chongqing 400044, China
* Correspondence: hyhuang@cqu.edu.cn

How To Cite: Zhang, B.; Chen, C.; Fu, J.; et al. Phy-Ture: A Physics-Guided Fusion and FeaTure Processing Framework for Unmanned Helicopter Dynamics Modeling. *Journal of Machine Learning and Information Security* **2026**, *2*(3), 19. https://doi.org/10.53941/jmlis.2026.100019

Received: 7 June 2026
Revised: 5 September 2026
Accepted: 9 September 2026
Published: 15 September 2026

Abstract: Accurate dynamics modeling for unmanned helicopters remains challenging because rotor aerodynamics, structural vibrations, and actuator dynamics are strongly coupled, nonlinear, and rapidly time varying during flight. To address this challenge, we propose a physics-guided fusion and feature processing framework for unmanned helicopter dynamics modeling (Phy-Ture). In this framework, the sequential dynamics estimates from the first-principles based model are fused with raw state-control input sequences to implement physics-guided feature augmentation. Subsequently, a multiscale temporal feature enhancement module is incorporated to reinforce the framework's representational capability for both the short-term transient behaviors and long-term evolutionary trends of the unmanned helicopter. Furthermore, a time-series decomposition and prediction module is introduced to explicitly decouple and model dynamic components across distinct time scales, enabling the precise prediction and compensation of dynamic residuals. Experiments on the Stanford Autonomous Helicopter Project dataset show that Phy-Ture reduces the mean RMSE by approximately 48% relative to the best-performing compared baseline under the benchmark protocol and by approximately 56% on held-out maneuver classes under the maneuver-class-based 10-fold evaluation. These results support the effectiveness of combining a structured physical baseline with adaptive temporal feature learning within the evaluated dataset and maneuver classes.

Keywords: unmanned helicopter; nonlinear dynamic system modeling; cross-attention; feature enhancement; temporal series decomposition

1. Introduction

Over the past decade, unmanned helicopters have attracted increasing attention in emergency response, precision agriculture, urban logistics, and military missions. Their ability to take off and land vertically, hover, and execute aggressive maneuvers enables various missions that other platforms cannot match [1–3]. Generally, UAV configurations are commonly classified into fixed-wing vehicles, multirotor vehicles, and helicopters. Unlike fixed-wing UAVs, unmanned helicopters dispense with runways and can sustain stationary hover, enabling close-range inspection, vertical cargo delivery, and confined-space maneuvers that fixed-wing platforms cannot perform [4–6]. Compared with multirotor UAVs, unmanned helicopters' larger rotor disks and additional control degrees of freedom confer decisive advantages for wide-envelope, heavy-payload missions [7–9]. To exploit these advantages, unmanned helicopters require precise, efficient, and robust control laws. These control laws depend on a high-precision real-time dynamics model. However, developing such a model faces several challenges. First, unmanned helicopters are inherently high-order, time-varying, underactuated and nonlinear, which makes it difficult to accurately represent their dynamic behavior. Second, strong couplings among the main rotor, fuselage and tail give rise to unsteady aerodynamic forces, structural vibrations and coupled engine dynamics, which complicate the overall dynamics and

hinder precise modeling. Third, it is difficult to identify aerodynamic and damping parameters. Furthermore, they may drift during flight, which leads to mismatches between simulation and actual system behavior, thereby reducing the accuracy of dynamics modeling. Taken together, these factors make unmanned-helicopter dynamics modeling a challenging research problem, reflecting the broader difficulty of characterizing complex nonlinear behavior in practical engineering systems [10–13].

Dynamics modeling of an unmanned helicopter seeks to map its measured flight states and control inputs to instantaneous linear and angular accelerations. Early attempts to model unmanned helicopter dynamics can be broadly classified into first-principles models [14–16] and traditional system identification models [17–19]. However, deriving accurate dynamics from first-principles is hindered by the helicopter's complex geometry and the unsteady nonlinear airflow around its rotors. At the same time, the inability to measure certain key states renders parameter estimation via traditional system identification methods unreliable. To reconcile these weaknesses, many researchers have adopted gray-box or hybrid strategies, consistently outperforming either method alone [20–23]. Tischler et al. [20] derived a 13-state linear representation and identified its parameters from frequency sweep data using CIFER. Chaturvedi et al. [22] developed a gray-box dynamics model that embeds first-principles rigid-body equations and uses adaptive system identification laws to estimate unknown inertia and input nonlinearities in real time. Tang et al. [23] derived a nonlinear helicopter dynamics model from first-principles equations and tuned its unknown parameters using an adaptive genetic-algorithm identification scheme. Huang et al. [24] combined system identification with an integrator-enhanced general internal model controller, achieving asymptotic error convergence in real flight tests. More recently, Pasquali et al. identified open-loop transfer functions from highly correlated closed-loop data and used their rational approximations to construct a reduced-order helicopter representation, demonstrating a practical route to compact helicopter modeling under feedback operation [25]. Although these studies illustrate the potential advantages of traditional hybrid modeling approaches, the practical calibration of such models remains a significant challenge. Parameters such as stability derivatives, hinge stiffness and rotor-body inertia are costly to measure and are usually inferred from limited flight data [26–28]. To make the problem computationally tractable, many existing studies rely on linear approximations and other simplifying assumptions. However, these simplifications can introduce modeling bias and lead to reduced accuracy when applied under realistic flight conditions, motivating physics-informed learning approaches that integrate system knowledge with data-driven methods [29–32]. More broadly, recent studies of nonlinear dynamical systems have also reported complex behaviors such as multi-scroll hidden attractors and coexisting attractors, further illustrating the rich dynamical structures that can arise from nonlinear interactions [33,34].

With the rapid growth of onboard computing power, high-frequency flight data logging, and related signal-processing techniques [35], data-driven modeling has become an appealing alternative to traditional approaches [36–38]. For helicopters, Punjani et al. [39] treat the system identification as high-dimensional regression and introduce a ReLU-based deep network that outperforms earlier identification methods across aerobatic maneuvers. Building on this idea, Chen et al. [40] trained a deep convolutional neural network on raw flight logs and achieved higher accuracy in unmanned helicopter dynamics modeling. Extending the approach to multirotors, Liu et al. [41] employed an LSTM to deliver real-time attitude estimates from quadrotor flight data. Amiruddin et al. [42] tested several deep learning architectures for quadcopter system identification using experimental flight data and found that a CNN-LSTM network achieved the best performance. These models require little aeromechanical expertise and avoid many of the assumptions imposed by analytical formulations, yielding high predictive accuracy on data similar to that used for training. However, their parameters lack direct physical meaning, and large data requirements can lead to overfitting and poor generalization when the flight regime deviates from the training distribution [43–46].

To improve the modeling accuracy and generalization of unmanned helicopter dynamics, recent work combines physical priors with deep learning for flight-dynamics modeling [47–49]. Kang et al. [50] fuse a first-principles rigid-body model with a deep convolutional network that learns residual dynamics from flight logs, producing a high-precision and broadly generalizable helicopter model. Zhang et al. [51] augment a first-principles helicopter model with a CNN-LSTM residual learner, resulting in a hybrid modeling framework that effectively combines the interpretability of physics-based models with the accuracy of data-driven methods. Mohajerin et al. [52] combine a simplified physics-based motion model with recurrent neural networks and achieve high-precision predictions for helicopter and quadrotor dynamics. By combining a simplified ground-effect aerodynamic model with a ReLU-based convolutional network, Xia [53] obtain more accurate predictions of small-helicopter rotational dynamics and enhanced landing control. Ma et al. [49] integrate a low-precision physics model with a residual-transformer autoencoder and a physics-informed multi-step loss, resulting in a high-precision aircraft dynamics model which is suitable for diverse flight regimes.

Most of the above methods follow a residual strategy in which the network learns the discrepancy between sensor measurements and a physics baseline. Although these hybrids generally outperform purely data-driven models, two limitations remain. First, existing fusion strategies are mostly unstructured (e.g., direct concatenation or simple residual correction), which can propagate physics model bias and measurement noise when the physical estimator is inaccurate under high-dynamic maneuvers. Second, temporal modeling is frequently insufficient for helicopter dynamics: conventional CNN/LSTM-style backbones struggle to jointly capture short-term transients and long-horizon evolution in highly nonlinear, time-varying flight regimes.

To address these limitations, we propose a physics-guided fusion and feature processing framework (Phy-Ture) for unmanned helicopter dynamics modeling. It first computes baseline linear and angular accelerations from a first-principles model, then introduces a physics-augmented fusion (PAF) module to adaptively integrate state-control history and physics baseline estimates through multi-head cross-attention in a shared latent space. Next, a multiscale temporal feature enhancement (MSTFE) module strengthens cross-scale temporal interactions to better represent both rapid and slowly varying dynamics. Finally, a time-series decomposition and prediction (TSDP) module separates the fused features into trend and seasonal components, and predicts the residual dynamics for compensation. The predicted residual is added back to the physical baseline to produce the final acceleration output.

Compared with representative physics-guided helicopter dynamics models that primarily combine a physical baseline with residual learning, Phy-Ture additionally performs feature-level interaction between sequential physical estimates and state–control representations and incorporates multiscale temporal processing before residual prediction.

The main contributions of this paper are listed as follows.

- We design a physics-guided fusion and feature processing framework tailored to unmanned helicopter dynamics, achieving nearly 50% reduction in RMSE on the Stanford unmanned helicopter dataset.
- We integrate a cross-attention-based fusion mechanism for physical estimates and state-control sequences, which adaptively extracts residual-relevant features from physical priors and fuses them with state-control features, effectively suppressing bias propagation from imperfect physical baselines and boosting modeling accuracy.
- We incorporate a multiscale temporal feature enhancement module with a lightweight two-layer SCINet backbone, which can capture richer multiscale dynamics than CNN or LSTM backbones and improves prediction accuracy in every flight maneuver tested.
- We show that decomposing the fused sequence into trend and seasonal components provides a more effective representation for residual prediction, leading to notably improved modeling accuracy for unmanned helicopters.

Following this introduction, Section 2 describes the mathematical foundation. Section 3 details the proposed Phy-Ture and its components. Section 4 presents the experimental studies. Last, Section 5 concludes this article.

2. Preliminaries

2.1. Attitude Representation and Coordinate Transformation

Before developing the dynamics model, we present the coordinate frames and the rotation matrices that relate to them. Figure 1 depicts the unmanned helicopter considered in this study. We regard the unmanned helicopter as a rigid body and define two coordinate frames:

- Inertial frame: This frame is centered at the helicopter's initial position. The x_i-axis points north, the y_i-axis points east, and the z_i-axis points downward, forming a local North–East–Down (NED) frame. Earth curvature and rotation are neglected.
- Body-fixed frame: This frame is rigidly attached to the vehicle body, with its origin o_b at the center of mass. The x_b-axis points to the nose, the y_b-axis points starboard, and the z_b-axis points downward.

Figure 1. An unmanned helicopter that was independently developed by Stanford University. This photograph shows the Synergy N9 helicopter in flight, serving as the physical platform for experimental validation and the definition of the reference coordinate systems.

Since state variables are measured or defined in different frames, a rotation matrix is required to transfer vectors from one frame to the other. Attitude is parameterized by a unit quaternion q, and q is expressed as:

$$q = q_1 i + q_2 j + q_3 k + q_4, \tag{1}$$

$$\|q\|^2 = q_1^2 + q_2^2 + q_3^2 + q_4^2 = 1, \tag{2}$$

where $[q_1, q_2, q_3]^\mathsf{T} \in \mathbb{R}^3$ is the vector part and q_4 is the scalar part.

Based on the foregoing, we adopt the notation R^{to}_{from}, where the subscript $from$ denotes the source frame and the superscript to denotes the target frame. In particular, the body-to-inertial rotation matrix R^{in}_{b}, which maps variables from the body-fixed frame into the inertial frame, can be written as

$$R^{in}_{b}(q) = \begin{bmatrix} 1-2(q_2^2+q_3^2) & 2(q_1q_2-q_3q_4) & 2(q_1q_3+q_2q_4) \\ 2(q_1q_2+q_3q_4) & 1-2(q_1^2+q_3^2) & 2(q_2q_3-q_1q_4) \\ 2(q_1q_3-q_2q_4) & 2(q_2q_3+q_1q_4) & 1-2(q_1^2+q_2^2) \end{bmatrix}. \tag{3}$$

The inertial-to-body rotation matrix R^{b}_{in}, which maps variable from the inertial frame into the body-fixed frame, can be written as

$$R^{b}_{in}(q) = (R^{in}_{b}(q))^\mathsf{T}. \tag{4}$$

2.2. Problem Formulation

Unmanned helicopter dynamics modeling aims to learn a mapping from flight states and control inputs to instantaneous linear and angular accelerations. The state variables include position $p = [x, y, z]^\mathsf{T}$, Euler angles $\phi = [\phi_x, \phi_y, \phi_z]^\mathsf{T}$, linear velocity $v = [v_x, v_y, v_z]^\mathsf{T}$, and angular velocity $\omega = [\omega_x, \omega_y, \omega_z]^\mathsf{T}$. Here, ϕ denotes the Euler-angle attitude coordinates included in the state history, whereas q is the equivalent unit-quaternion representation used to construct the rotation matrices in Equations (3) and (4). The control input is $u = [u_1, u_2, u_3, u_4]^\mathsf{T}$, where u_1 and u_2 denote cyclic commands, u_3 is the tail-rotor command, and u_4 is the collective command.

Considering both translational and rotational dynamics, we describe the complete nonlinear model as

$$\dot{v} = -\omega \times v + R^{b}_{\text{in}} g + Av + C_1 u_v + D_1 + \delta_v, \tag{5}$$

$$\dot{\omega} = I^{-1}(-\omega \times (I\omega)) + B\omega + C_2 u_\omega + D_2 + \delta_\omega, \tag{6}$$

where δ_v and δ_ω denote aggregated uncertainties that cannot be accurately captured by the analytical model, including rotor–fuselage coupling effects, unsteady aerodynamics, actuator dynamics, and modeling simplifications.

Based on Equations (5) and (6), we adopt a two-level hybrid framework composed of a physical baseline and a residual compensation model. The physical baseline is obtained by removing δ_v and δ_ω and identifying model coefficients offline; these physical parameters are fixed during neural-network training. The residual model learns only the discrepancy between measured dynamics and physical baseline prediction.

The six-dimensional acceleration target is defined as

$$\mathbf{a}_t = \begin{bmatrix} \dot{v}_t^\mathsf{T} \\ \dot{\omega}_t^\mathsf{T} \end{bmatrix} \in \mathbb{R}^6, \tag{7}$$

and the physical baseline output at time t is denoted by

$$\mathbf{a}_t^p = \begin{bmatrix} \dot{v}_{p,t}^\mathsf{T} \\ \dot{\omega}_{p,t}^\mathsf{T} \end{bmatrix}. \tag{8}$$

Accordingly, the residual to be estimated is

$$\mathbf{r}_t = \mathbf{a}_t - \mathbf{a}_t^p. \tag{9}$$

To incorporate temporal context, we stack the past H samples into a sliding window:

$$\begin{aligned} S_{t-H+1:t} = [\, & p_{t-H+1},\ \phi_{t-H+1},\ v_{t-H+1},\ \omega_{t-H+1},\ u_{t-H+1}; \\ & \ldots;\ p_t,\ \phi_t,\ v_t,\ \omega_t,\ u_t\,], \end{aligned} \tag{10}$$

and compute the corresponding physical baseline sequence $P_{t-H+1:t}$ by applying the physical baseline model to the same sliding window defined above. The residual compensation network is written as

$$\widehat{\mathbf{r}}_t \;=\; \mathbf{F}(\mathbf{S}_{t-H+1:t}, \mathbf{P}_{t-H+1:t};\, \boldsymbol{\theta}), \tag{11}$$

and the final hybrid prediction is

$$\widehat{\mathbf{a}}_t = \mathbf{a}_t^p + \widehat{\mathbf{r}}_t. \tag{12}$$

Given a training set $\mathcal{D} = \{(\mathbf{S}_{t-H+1:t}^{(i)}, \mathbf{P}_{t-H+1:t}^{(i)}, \mathbf{a}_t^{(i)})\}_{i=1}^{N}$, the learnable parameters are optimized by minimizing MSE:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \frac{1}{N}\sum_{i=1}^{N} \left\| \mathbf{a}_t^{(i)} - \left(\mathbf{a}_t^{p,(i)} + \widehat{\mathbf{r}}_t\right) \right\|_2^2. \tag{13}$$

By solving the optimization problem in Equation (13), we obtain a physics-guided hybrid dynamics model that retains the first-principles prediction as a physical baseline while learning residual corrections for complex nonlinear and time-varying unmodeled dynamics.

3. Methodology

3.1. Framework of the Phy-Ture

The overall architecture of the proposed Phy-Ture physics-guided hybrid dynamics modeling framework is illustrated in Figure 2. Built upon the residual learning paradigm defined in the preliminaries, Phy-Ture consists of four sequentially connected functional modules with a clear end-to-end data flow: (a) First-Principles based Physical model Calculation (FPPC) module, which generates baseline acceleration predictions from the rigid-body dynamics model; (b) Physics-Augmented Fusion (PAF) module, which adaptively fuses the state-control sequence with physical baseline estimates; (c) Multiscale Temporal Feature Enhancement (MSTFE) module, which captures the multiscale dynamic characteristics of the unmanned helicopter; and (d) Time Series Decomposition and Prediction (TSDP) module, which outputs the final residual dynamics prediction.

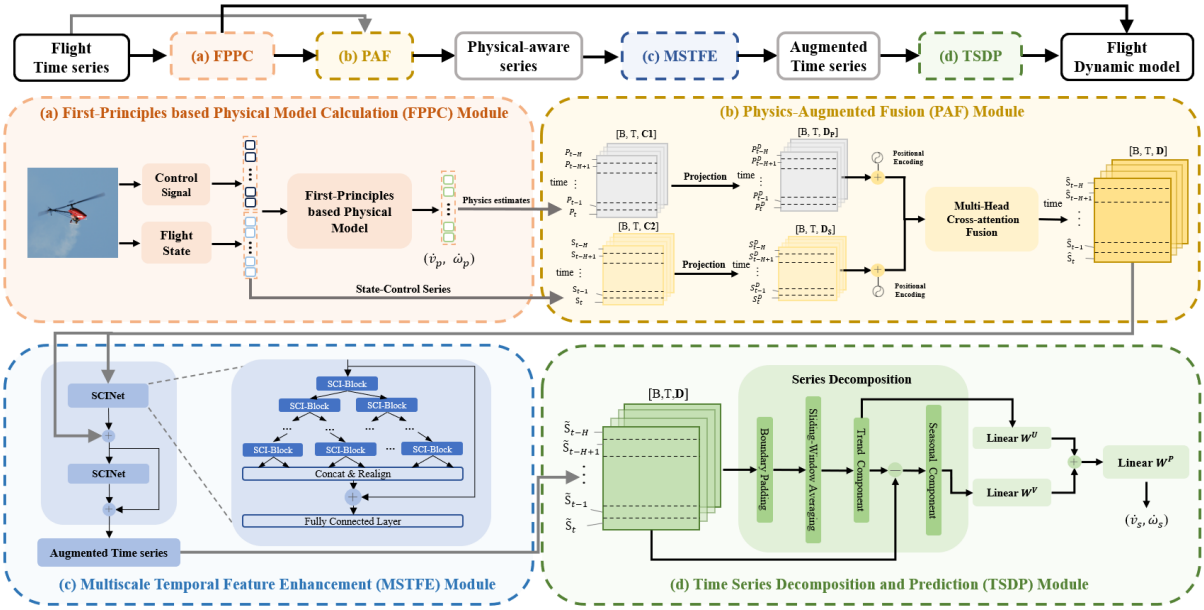

Figure 2. Overview of the proposed Phy-Ture framework for unmanned helicopter dynamics.This diagram illustrates the end-to-end data flow of the framework, sequentially connecting the first-principles based physical model calculation (FPPC), physics-augmented fusion (PAF), multiscale temporal feature enhancement (MSTFE), and time series decomposition and prediction (TSDP) modules.

Different from purely data-driven pipelines that directly map flight states to dynamics outputs without an explicit physical baseline, Phy-Ture uses the first-principles model as a structured physical baseline for the helicopter's rigid-body dynamics and learns residual corrections for the remaining unmodeled dynamics.

This design improves prediction accuracy in the evaluated nonlinear flight regimes while retaining the first-principles prediction as a structured physical baseline for residual learning.

In the FPPC module, we employ a first-principles-based 6-degree-of-freedom (6-DoF) rigid-body physical model of the unmanned helicopter. Given the measured flight states and control inputs, this module generates structured baseline estimates of linear and angular accelerations. These baseline estimates are then fed into the subsequent PAF module as physical baseline information.

Next, to integrate physical baseline estimates with state-control sequence features, the PAF module first projects both sequences into a shared latent space.

The resulting physics-guided feature sequence is then fed into the MSTFE module, which performs multiscale feature interaction to amplify latent temporal features before trend-seasonal decomposition. Through its multiscale interaction enhancement mechanism, MSTFE enables the network to more effectively capture the strong nonlinear and rapidly time-varying dynamic characteristics of the helicopter system, thereby supporting accurate prediction in complex flight regimes.

Finally, the enriched temporal feature representations are fed into the TSDP module, which performs trend-seasonal decomposition and predicts the acceleration residual for compensation. This decomposition operation decouples the long-term trend and short-term periodic components of the dynamics, simplifies the residual learning task, sharpens the model's focus on inherent dynamic structures, and improves prediction accuracy across diverse flight maneuvers.

3.2. First-Principles Based Physical Model Calculation

The FPPC module serves as the physical baseline predictor in the proposed hybrid framework. It uses a first-principles analytical approach to estimate linear and angular accelerations within the full state-control space. The physical model is expressed in Equations (14) and (15) below.

$$\dot{v}_p = -\omega \times v + R_{\text{in}}^b g + Av + C_1 u_v + D_1, \tag{14}$$

$$\dot{\omega}_p = I^{-1}(-\omega \times (I\omega)) + B\omega + C_2 u_\omega + D_2, \tag{15}$$

where R_{in}^b is the rotation matrix, $g = [\,0, 0, g_0\,]^\mathsf{T}$ is the gravity vector, and I denotes the inertia tensor. The control inputs are partitioned to $u_v = [\,0, 0, u_4\,]^\mathsf{T}$ and $u_\omega = [\,u_1, u_2, u_3\,]^\mathsf{T}$. Matrices A, B, C_1, C_2 and bias vectors D_1, D_2 are parameters of the physical model, which are identified via least-squares optimization. This physical model is derived by linearizing the general rigid-body equations with small-disturbance theory, a formulation widely used in both fixed-wing and rotary-wing flight-dynamics studies [54].

Given the difficulty of measuring certain state variables and the incomplete characterization of rotorcraft aerodynamics, the first-principles based physical model incorporates several simplifying assumptions.

First, in real aircraft dynamics, the inertia matrix is generally a symmetric 3×3 matrix that includes both principal axis inertia and cross-axis coupling inertia. To simplify the computations, the inertia matrix I is assumed diagonal, which eliminates coupling terms between the axes. Second, in this physical model, the aerodynamic forces and aerodynamic moments are regarded as linear relationships directly proportional to the velocity: the aerodynamic force is written as Av and the aerodynamic moment as $B\omega$. This linearization significantly reduces the complexity of the equation, but the error becomes more obvious when flying at high speed or with a large angle of attack. Last, the control inputs are assumed independent across axes. The forces and moments are linearly dependent only on the corresponding inputs, mapped respectively through the constant matrices C_1 and C_2.

The physical baseline uses a diagonal inertia matrix, linear force and moment mappings, and decoupled control channels.

We form the flight state-control series over the past H time steps as $S_{t-H+1:t}$. Applying the physical model to this series produces the matching physical baseline sequence

$$P_{t-H+1:t} = [\dot{v}_{t-H+1}^p, \dot{\omega}_{t-H+1}^p; \ldots; \dot{v}_t^p, \dot{\omega}_t^p]. \tag{16}$$

The two sequences, i.e., $S_{t-H+1:t}$ and $P_{t-H+1:t}$, are then fed into the PAF module to construct physics-guided representations for residual compensation learning.

3.3. Physics-Augmented Fusion

The physics-augmented fusion (PAF) module takes the flight state–control series $S_{t-H+1:t}$ together with its corresponding physical baseline series $P_{t-H+1:t}$ as inputs, and adaptively fuses them to generate a physics-guided representation for downstream residual modeling.

First, each series is projected into a shared latent space via learnable linear layers:

$$S_d = S_{t-H+1:t} W^{S_d}, \tag{17}$$

$$P_d = P_{t-H+1:t} W^{P_d}, \tag{18}$$

where $W^{S_d} \in \mathbb{R}^{D_S \times d_{\text{model}}}$ and $W^{P_d} \in \mathbb{R}^{D_P \times d_{\text{model}}}$. Here, D_S and D_P are the channel dimensions of $S_{t-H+1:t}$ and $P_{t-H+1:t}$, and d_{model} denotes the dimension of the shared latent space. Through these projections, the two sequences are mapped into a unified feature space that facilitates effective fusion.

Subsequently, to ensure that the downstream model can explicitly capture the temporal order of the data, we apply positional encoding independently to each projected series before fusion. The encoding method we adopt is the sinusoidal position encoding proposed by Vaswani et al. [55], which introduces deterministic position-dependent variations into the representations. The corresponding encoding formula is:

$$\text{PE}(p, 2i) = \sin\Big(\frac{p}{10000^{2i/d_{model}}}\Big), \tag{19}$$

$$\text{PE}(p, 2i+1) = \cos\Big(\frac{p}{10000^{2i/d_{model}}}\Big), \tag{20}$$

where $p \in \{0, \ldots, H-1\}$ is the timestep index, and $i \in \{0, \ldots, d_{\text{model}}/2 - 1\}$ is the channel index.

The resulting position vectors are added element-wise to the original feature vectors, producing the position-aware series S' and P'. Since the encoding does not introduce additional trainable parameters, its computation cost can be neglected. This parameter-free design preserves both absolute and relative temporal offsets and generalizes to sequences longer than those seen during training.

As shown in Figure 3, we fuse the position-aware state-control series $S' \in \mathbb{R}^{H \times d_{\text{model}}}$ and the position-aware physics baseline series $P' \in \mathbb{R}^{H \times d_{\text{model}}}$ via multi-head cross-attention. Here, h denotes the number of cross-attention heads.

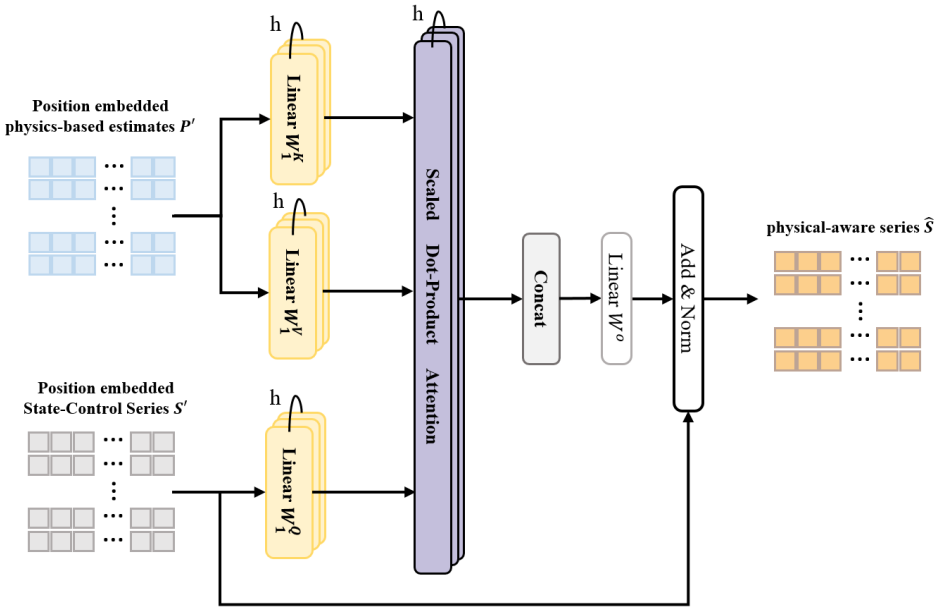

Figure 3. Multi-head cross-attention mechanism for fusing physical baseline estimates and state-control representations.

For each head $i = 1, \ldots, h$, we compute

$$Q_i = S' W_i^Q, \quad K_i = P' W_i^K, \quad V_i = P' W_i^V, \tag{21}$$

with $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_k}$. Here, d_k denotes the per-head embedding dimension, typically $d_k = d_{\text{model}}/h$, so that $Q_i, K_i, V_i \in \mathbb{R}^{H \times d_k}$. Then, each head performs scaled dot-product attention to compute an output of shape $\mathbb{R}^{H \times d_k}$:

$$\text{head}_i = \text{Softmax}\Big(\frac{Q_i K_i^\top}{\sqrt{d_k}}\Big) V_i. \tag{22}$$

The outputs of all heads are concatenated and projected back to d_{model} dimensions:

$$\text{MultiHead}(Q, K, V) = Concat(\text{head}_1, \ldots, \text{head}_h)\, W^O, \tag{23}$$

where $W^O \in \mathbb{R}^{(h\,d_k)\times d_{\text{model}}}$ is the output projection matrix. Subsequently, a residual connection around S' followed by layer normalization yields the fused series:

$$\hat{S}_{t-H+1:t} = \text{LayerNorm}(\text{MultiHead}(S', P') \;+\; S'), \tag{24}$$

Compared with feature concatenation or pointwise gated fusion, multi-head cross-attention enables each state–control representation to selectively aggregate relevant information from the physical baseline sequence across the observation history. In contrast to pointwise gating, which mainly modulates the relative contributions of aligned features at each time step, cross-attention explicitly models query–key dependencies between the two sequences and can therefore capture cross-temporal interactions between state–control features and physical baseline estimates. The residual connection preserves the original state–control representation.

3.4. Multiscale Temporal Feature Enhancement

In this module, MSTFE adopts the sample-splitting and cross-scale interaction mechanism of SCINet [56] to extract multiscale temporal features for the downstream stage.

Figure 2c illustrates SCINet's cascaded SCI-Block architecture. These blocks capture dynamic temporal dependencies at multiple resolutions via one-dimensional convolutional operations, thereby improving the time series representation capabilities for unmanned helicopters.

For clarity, we use $\hat{F} \in \mathbb{R}^{H\times d_{\text{model}}}$ to denote the input feature sequence of the SCI-Block. As shown in Figure 4, the SCI-Block downsamples $\hat{F}$ by separating the series into two subseries, $\hat{F}_{\text{odd}}$ and $\hat{F}_{\text{even}}$, according to the parity indices. The SCI-Block then applies distinct convolutional kernels to $\hat{F}_{\text{odd}}$ and $\hat{F}_{\text{even}}$. Since the kernels are independent, the extracted features encode unique yet complementary temporal relations and thus possess richer representational power. The information is then exchanged between the two subseries by learning the affine transformation parameters in a mutually adaptive manner. Specifically, $\hat{F}_{\text{even}}$ and $\hat{F}_{\text{odd}}$ are first projected to hidden states by two separate 1D convolutions, denoted as ϕ and ψ, converted to the formats of exp, and then interact through element-wise production $\odot$ with $\hat{F}_{\text{even}}$ and $\hat{F}_{\text{odd}}$, respectively. The process is formalized as follows:

$$\hat{F}^s_{\text{odd}} = \hat{F}_{\text{odd}} \;\odot\; \exp(\phi(\hat{F}_{\text{even}})), \tag{25}$$

$$\hat{F}^s_{\text{even}} = \hat{F}_{\text{even}} \;\odot\; \exp(\psi(\hat{F}_{\text{odd}})). \tag{26}$$

Subsequently, the SCI-Block feeds the scaled features $\hat{F}^s_{\text{even}}$ and $\hat{F}^s_{\text{odd}}$ to two 1D convolutional modules, projecting them to hidden states that are then added back to the original $\hat{F}^s_{\text{even}}$ and $\hat{F}^s_{\text{odd}}$, respectively. The operation is formalized as follows:

$$\hat{F}'_{\text{odd}} = \hat{F}^s_{\text{odd}} + \rho(\hat{F}^s_{\text{even}}), \tag{27}$$

$$\hat{F}'_{\text{even}} = \hat{F}^s_{\text{even}} + \eta(\hat{F}^s_{\text{odd}}), \tag{28}$$

where ρ and η denote two distinct 1D convolutional modules.

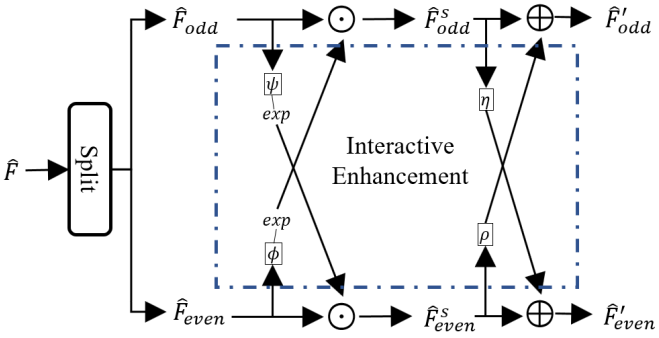

Figure 4. The process of interactive enhancement within SCI-Block.

As illustrated at the right side of Figure 2c, multiple SCI-Blocks are arranged in a binary tree structure. This structure gives each SCI-Block both local and global views of the entire time series, helping it extract salient temporal features. Since the series is progressively down-sampled and processed at successive levels, the network can learn representations at different temporal resolutions. Information from earlier levels is cumulatively propagated upward, hence deeper blocks inherit and refine the finer-grained temporal details captured by the preceding layers.

After all interactions are completed, SCINet realigns the extracted features into a new series representation and adds this residual to the original time series, producing an enhanced series denoted by $\tilde{\mathbf{S}}_{t-H+1:t}$ with greater predictive capacity.

By capturing temporal dependencies at multiple scales, MSTFE provides the downstream module with a richer temporal representation. Since MSTFE itself does not rely on helicopter-specific operators, the architecture is not intrinsically tied to helicopter dynamics and could, in principle, be adapted to other aircraft platforms by redefining the physical baseline and the corresponding input–output variables.

3.5. Time Series Decomposition and Prediction

Since the time series of unmanned helicopters encompasses multiscale components, such as trend changes and periodic fluctuations, directly predicting the mixed signal would require a single model to simultaneously capture the dynamic coupling relationships of these two time scales.

To address this problem, we pad the input series $\tilde{S}_{t-H+1:t}$ to avoid edge distortion and apply a sliding-window moving average of width w to extract the trend component:

$$U = \mathrm{MA}_w(\tilde{S}_{t-H+1:t}), \tag{29}$$

and compute the seasonal component by

$$V = \tilde{S}_{t-H+1:t} \; - \; U \, . \tag{30}$$

Next, since the trend and seasonal components exhibit distinct patterns and statistical properties, we apply separate linear mappings to each, allowing the model to adaptively transform and weight long-term and periodic signals. Let $L = H$ and $U, V \in \mathbb{R}^{L \times d_{\mathrm{model}}}$. The temporal projections are defined as follows:

$$U_p \; = \; (W^U)^{\mathsf{T}} U, \tag{31}$$

$$V_p \; = \; (W^V)^{\mathsf{T}} V, \tag{32}$$

where $W^U, W^V \in \mathbb{R}^{L \times P}$ and $U_p, V_p \in \mathbb{R}^{P \times d_{\mathrm{model}}}$. In practice, these mappings are applied independently to each feature channel, enabling the model to separately transform and emphasize long-term trends and periodic fluctuations before fusion.

The two transformed feature streams are fused by element-wise addition and passed through a final linear projection $W^P \in \mathbb{R}^{d_{\mathrm{model}} \times 6}$ to generate the predicted residual terms for linear and angular accelerations, denoted as $\dot{v}_s$ and $\dot{\omega}_s$, respectively. Adding this data-driven residual prediction element-wise to the corresponding physical-baseline output produces the final hybrid acceleration prediction.

4. Performance Evaluation

4.1. Flight Dataset and Metrics

To comprehensively validate Phy-Ture for unmanned helicopter dynamics, we evaluate Phy-Ture on the public flight dataset released by the Stanford Autonomous Helicopter Project [57]. In this dataset, an expert pilot repeatedly flew a Synergy N9 helicopter through 20 canonical aerobatic maneuvers. The helicopter weighs 4.58 kg with a 720 mm main rotor blade length. The onboard sensor suite recorded the helicopter's position, attitude, linear and angular velocities, and their corresponding accelerations. Meanwhile, a microcontroller captured, in parallel, four primary control signals transmitted by the Spektrum RC DX7 to the aircraft's receiver.

As shown in Figure 5, the Synergy N9 dataset contains 20 distinct aerobatic maneuvers, with durations ranging from approximately 140 s to 510 s and flight distances from 1231 m to 5863 m. State and control signals were synchronized and smoothed using a Kalman smoother at 100 Hz, yielding 7376 s of usable flight data.

For model development and evaluation, each flight log is first organized into 10-s trajectory segments, which are assigned to the training, validation, and test partitions in a 70%/20%/10% ratio. All subsequent sample construction and data augmentation, including temporally overlapping segments generated with a 5-s offset, are performed independently within each partition. This procedure ensures that no constructed sample crosses a partition boundary and that no raw timestamp is shared across the training, validation, and test sets. Beyond this benchmark split, a stricter evaluation of generalization is provided by the maneuver-class-based 10-fold protocol in Section 4.4, where entire maneuver classes are held out before sample construction and neural-network training. For reproducibility, the benchmark partition is generated using a fixed random seed of 2030. The resulting training, validation, and test partitions are kept identical for all compared methods, and overlapping samples are constructed only after the partition has been fixed.

All state and control variables are normalized using statistics computed exclusively from the training partition. The training set is used to optimize the network parameters, the validation set for hyperparameter selection, and the test set exclusively for final performance evaluation.

The input to all methods is the state–control sequence $S_{t-H+1:t}$, comprising position, attitude, linear velocity, angular velocity and four control signals. All compared methods use a 10-s input history and perform instantaneous dynamics prediction, with the linear and angular accelerations at the current time t as the prediction targets rather than future-horizon forecasts.

In the original dataset, position is expressed in meters, Euler angles in radians, linear velocity in m/s, angular velocity in rad/s, linear acceleration in m/s^2, and angular acceleration in rad/s^2. The four control commands are approximately normalized to the range $[-1, 1]$.

Given the regression setting, we select RMSE as the evaluation metric for each method's dynamics modeling performance. RMSE shares the targets' physical units, enabling direct interpretation, and penalizes large deviations while remaining comparable across methods.

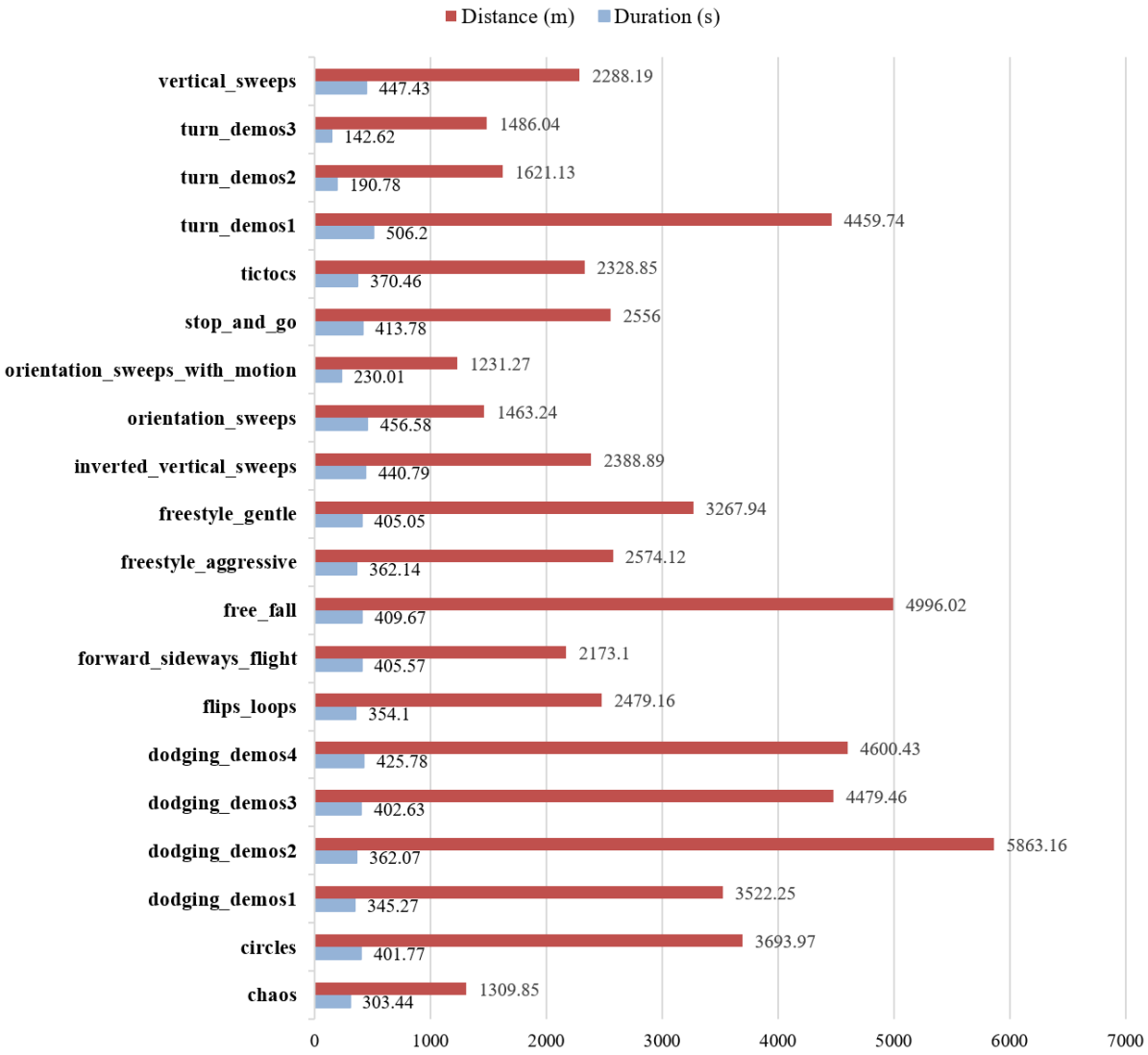

Figure 5. Maneuver duration and flight distance of 20 distinct aerobatic maneuvers in the Stanford Autonomous Helicopter Project.

4.2. Physical Model Calibration and Network Training

Before training the data-driven components, we estimate the coefficients in Equations (14) and (15) via least squares. Specifically, the calibration is performed using only the training partition. Let $\Theta_v = \{A, C_1, D_1\}$ and $\Theta_\omega = \{B, C_2, D_2\}$. The translational and rotational coefficients are obtained by solving the following two independent unweighted least-squares problems:

$$\widehat{\Theta}_v = \arg\min_{\Theta_v} \sum_{i \in \mathcal{D}_{\text{train}}} \|\dot{v}_i - \dot{v}_{p,i}(\Theta_v)\|_2^2 ,$$
$$\widehat{\Theta}_\omega = \arg\min_{\Theta_\omega} \sum_{i \in \mathcal{D}_{\text{train}}} \|\dot{\omega}_i - \dot{\omega}_{p,i}(\Theta_\omega)\|_2^2 . \tag{33}$$

The calibration is performed in the original physical units before neural-input normalization. Once identified, the physical coefficients are fixed throughout neural-network training and evaluation and are shared by all residual-hybrid methods.

The obtained physical-model parameter A is $\mathrm{diag}[-0.05, -0.06, -1.42]$. Parameter B is $\mathrm{diag}[-5.47, -5.32, -3.43]$. The control matrices are $C_1 = \mathrm{diag}[0, 0, -0.02]$ and $C_2 = \mathrm{diag}[0.02, 0.01, 0.02]$. The control-related vectors are $D_1 = [0, 0, -0.51]^\mathsf{T}$ and $D_2 = [-1.23, -0.21, 0.14]^\mathsf{T}$.

Then, with the physical coefficients held fixed, we train the remaining data-driven blocks of Phy-Ture. In the PAF module, both the state–control and physical baseline sequences are mapped into a 64-dimensional latent space and fused via multi-head cross-attention with four parallel heads. The MSTFE module stacks two multiscale interaction layers, each with three hierarchical interaction levels that split the input into odd and even subsequences and enable mutual enhancement through cross-interactions. The TSDP module applies a moving-average window of length twenty-five to decompose the series into trend and seasonal components. At the 100-Hz sampling rate, $w = 25$ corresponds to a 0.25-s local averaging interval, providing a short-term smoothing scale while retaining transient variations within the input sequence.

All experiments are conducted on a single NVIDIA GeForce RTX 2080 Ti GPU (NVIDIA Corporation, Santa Clara, CA, USA) using PyTorch 2.5.0. Weights and biases are updated with the Adam optimizer. The initial learning rate is set to 1×10^{-3}. We employ a combined warm-up and cosine-annealing schedule for the learning rate. Specifically, during the first ten epochs the rate is warmed up by increasing it linearly from zero to its peak value. After that it follows the cosine decay curve, gradually falling towards zero by the final training epoch. Moreover, to guard against overfitting we apply dropout with probability 0.2 after each convolutional block and inside the multi-head attention layers. No early stopping is used in the experiments, and each neural model is trained for its predefined number of epochs.

For fair comparison, all methods use the same state–control history, target definition, data partitions, and preprocessing procedure. For each baseline, we follow the architecture and recommended hyperparameter configuration reported in the corresponding paper or official implementation whenever these details are available. For training settings that are not specified in the original work, we adopt the same training protocol as Phy-Ture to ensure a consistent comparison. Any additional hyperparameter selection required for the present dataset is performed exclusively on the validation partition. Residual-hybrid methods use the same fixed physical predictor, whereas purely data-driven baselines receive only the state–control inputs. Each neural method is evaluated over 10 random seeds, and the mean and standard deviation of RMSE are reported.

With respect to the input length H, the cross-attention operation in PAF has a computational complexity of $\mathcal{O}(H^2 d)$, where d is the latent dimension, whereas the temporal convolution and decomposition operations scale linearly with H for fixed architectural widths. Phy-Ture performs a single feed-forward residual prediction without autoregressive rollout.

4.3. Performance Comparison of Phy-Ture and State-of-the-Art Approaches on Benchmark Dataset

We compare Phy-Ture with representative benchmark approaches, including CNN hybrid [50], CNN–LSTM hybrid [51], TCN [58], TimesNet [59], TimeMixer [60], DLinear [61], and Conv–Informer [62]. In addition, we report the standalone physical baseline to quantify the modeling discrepancy that remains before residual compensation. Table 1 summarizes the results on the Stanford helicopter dataset.

As shown in Table 1, Phy-Ture attains the lowest RMSE on all six output channels. The standalone physical baseline yields RMSEs of 6.03, 5.91, and 6.53 for the three linear-acceleration channels and 2.48, 2.63, and 1.67 for the three angular-acceleration channels, respectively, with a mean summary score of 4.21. These errors quantify the substantial residual discrepancy that must be compensated by the data-driven component under the simplified physical formulation. Across 10 independent runs, Phy-Ture achieves a mean summary score of 0.32 ± 0.01, compared with 0.62 ± 0.03 for the strongest benchmark, CNN–LSTM hybrid, corresponding to an approximately 48% reduction in the mean value. Linear accelerations $\dot{v}_x$, $\dot{v}_y$, and $\dot{v}_z$ decrease from 0.73 ± 0.04, 0.72 ± 0.02, and 0.63 ± 0.03 under the CNN–LSTM hybrid to 0.37 ± 0.02, 0.32 ± 0.01, and 0.42 ± 0.02, respectively. Angular accelerations $\dot{\omega}_x$, $\dot{\omega}_y$, and $\dot{\omega}_z$ similarly decrease from 0.44 ± 0.03–0.69 ± 0.03 to 0.23 ± 0.01–0.31 ± 0.01. These gains also hold relative to recent time-series baselines such as TimesNet and TimeMixer, whose mean summary

scores are 0.70 ± 0.04 and 0.68 ± 0.03, respectively. The relatively small standard deviations of Phy-Ture across all six channels further indicate that its prediction performance remains stable across different random initializations.

Table 1. RMSE comparison on the Stanford helicopter dataset. Neural-model results are reported as mean ± standard deviation over 10 independent runs, while the deterministic physical baseline is reported as a single value.

Method	$\dot{v}_x$	$\dot{v}_y$	$\dot{v}_z$	$\dot{\omega}_x$	$\dot{\omega}_y$	$\dot{\omega}_z$	**Mean**
Physical baseline	6.03	5.91	6.53	2.48	2.63	1.67	4.21
CNN hybrid	1.76 ± 0.08	1.43 ± 0.06	1.73 ± 0.07	1.33 ± 0.05	1.58 ± 0.06	1.67 ± 0.07	1.58 ± 0.06
CNN–LSTM hybrid	0.73 ± 0.04	0.72 ± 0.02	0.63 ± 0.03	0.44 ± 0.03	0.69 ± 0.03	0.49 ± 0.02	0.62 ± 0.03
TCN	1.46 ± 0.06	1.13 ± 0.05	1.41 ± 0.05	0.75 ± 0.04	0.77 ± 0.04	0.70 ± 0.03	1.04 ± 0.08
TimesNet	1.06 ± 0.06	0.74 ± 0.03	0.76 ± 0.04	0.58 ± 0.03	0.57 ± 0.02	0.47 ± 0.03	0.70 ± 0.04
TimeMixer	0.76 ± 0.03	0.61 ± 0.03	0.78 ± 0.04	0.65 ± 0.03	0.68 ± 0.04	0.62 ± 0.03	0.68 ± 0.03
DLinear	2.64 ± 0.11	2.15 ± 0.09	1.93 ± 0.08	0.79 ± 0.04	0.82 ± 0.04	0.67 ± 0.03	1.50 ± 0.07
Conv–Informer	1.32 ± 0.05	1.09 ± 0.04	1.23 ± 0.05	1.01 ± 0.04	0.96 ± 0.04	0.75 ± 0.04	1.06 ± 0.04
Phy-Ture (Ours)	0.37 ± 0.02	0.32 ± 0.01	0.42 ± 0.02	0.31 ± 0.01	0.29 ± 0.01	0.23 ± 0.01	0.32 ± 0.01

To complement the RMSE-based comparison, we further evaluate the mean absolute error (MAE) and maximum absolute error (MaxAE) of Phy-Ture and the strongest benchmark baseline, CNN–LSTM hybrid. MAE characterizes the average magnitude of the prediction error and is less sensitive to occasional large deviations than RMSE, whereas MaxAE measures the largest instantaneous absolute prediction error observed on the benchmark test set. Together, these two metrics provide complementary assessments of typical prediction accuracy and worst-case prediction deviation. Table 2 summarizes the results.

Table 2. MAE and MaxAE comparison between Phy-Ture and the strongest benchmark baseline.

Method	Metric	$\dot{v}_x$	$\dot{v}_y$	$\dot{v}_z$	$\dot{\omega}_x$	$\dot{\omega}_y$	$\dot{\omega}_z$
CNN–LSTM hybrid	MAE	0.52	0.42	0.51	0.36	0.40	0.40
CNN–LSTM hybrid	MaxAE	3.66	3.73	3.11	2.09	3.45	2.44
Phy-Ture	MAE	0.24	0.19	0.29	0.22	0.19	0.20
Phy-Ture	MaxAE	1.89	1.57	2.23	1.55	1.50	1.38

Phy-Ture achieves lower MAE than the CNN–LSTM hybrid across all six output channels. For the translational dynamics, the MAE of $\dot{v}_x$, $\dot{v}_y$, and $\dot{v}_z$ decreases from 0.52, 0.42, and 0.51 to 0.24, 0.19, and 0.29, respectively. For the rotational dynamics, the corresponding MAE decreases from 0.36, 0.40, and 0.40 to 0.22, 0.19, and 0.20. These results confirm that the performance improvement of Phy-Ture is not limited to the RMSE metric, but is also reflected in consistently lower average absolute prediction errors across all output dimensions.

Phy-Ture also achieves lower MaxAE across all six output channels, indicating a reduction in the largest instantaneous prediction deviations observed on the benchmark test set. For the translational accelerations, MaxAE decreases from 3.66, 3.73, and 3.11 under the CNN–LSTM hybrid to 1.89, 1.57, and 2.23, respectively. For the angular accelerations, the corresponding values decrease from 2.09, 3.45, and 2.44 to 1.55, 1.50, and 1.38. Therefore, the improvement of Phy-Ture is reflected not only in aggregate RMSE and average absolute error, but also in reduced worst-case instantaneous errors. Together with the per-maneuver RMSE results and the time-domain comparisons presented below, these findings provide a more comprehensive characterization of the prediction errors across different flight regimes.

Figure 6 shows the RMSE of each model on every individual test flight segment. Across the evaluated flight maneuvers, Phy-Ture consistently achieves lower errors than the compared methods over a wide range of motion patterns. For linear accelerations (Figure 6a–c), Phy-Ture shows marked gains in both relatively steady maneuvers such as vertical_sweeps and orientation_sweeps and more aggressive maneuvers such as freestyle_aggressive and dodging_demos. This consistent advantage across diverse flight patterns indicates that Phy-Ture can capture both gradual state evolutions and rapid transients. Similarly, for angular accelerations (Figure 6d–f), Phy-Ture maintains lower RMSE in highly dynamic maneuvers such as dodging_demos and chaos.

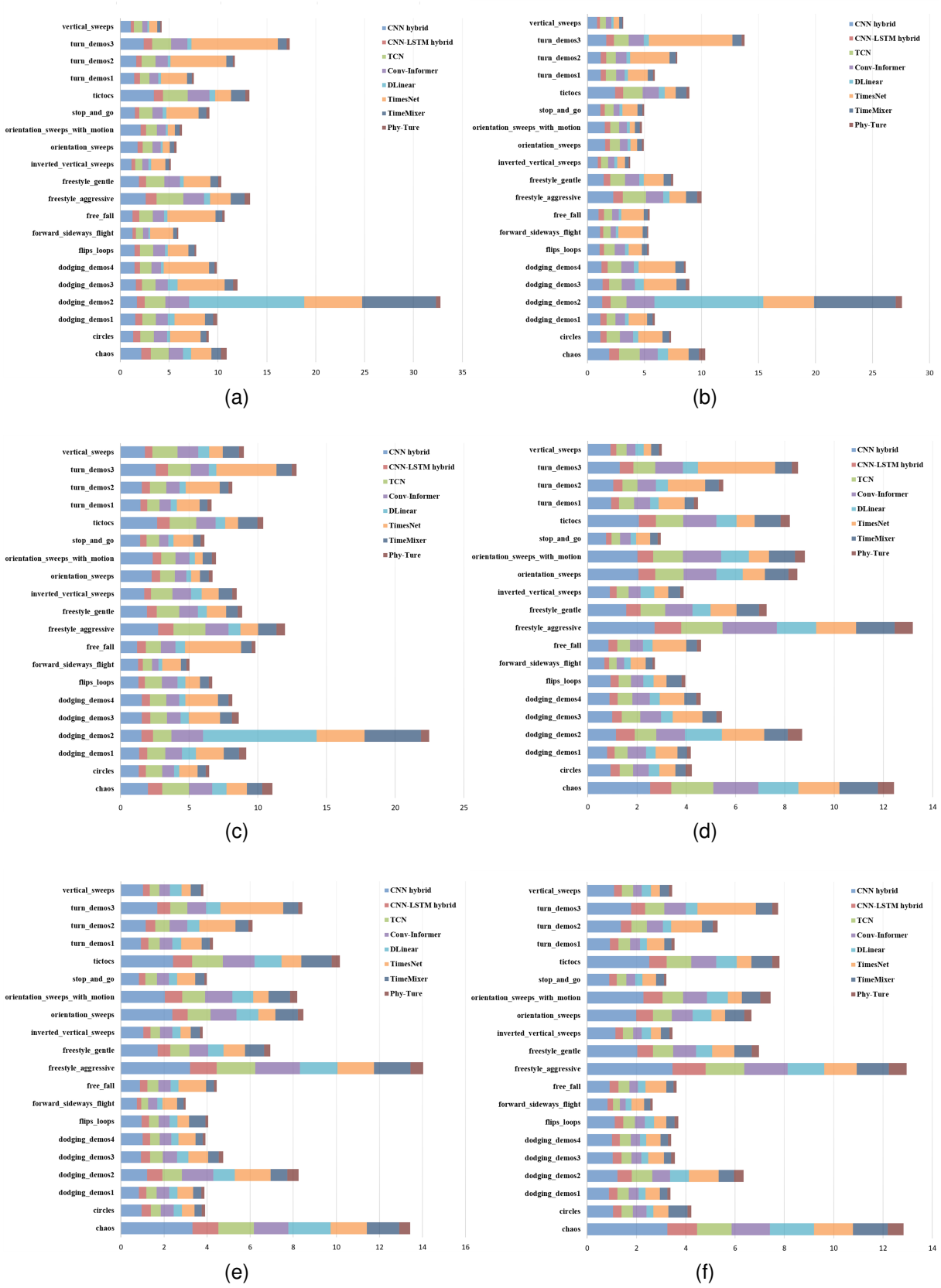

Figure 6. Comparison of Phy-Ture with benchmark approaches. (**a**) The RMSE of linear acceleration in x-axis. (**b**) The RMSE of linear acceleration in y-axis. (**c**) The RMSE of linear acceleration in z-axis. (**d**) The RMSE of angular acceleration in x-axis. (**e**) The RMSE of angular acceleration in y-axis. (**f**) The RMSE of angular acceleration in z-axis.

Notably, models such as DLinear and TimeMixer perform competitively on certain maneuvers with relatively smooth dynamics, while their errors increase in more dynamically varying regimes, indicating greater sensitivity to changes in maneuver dynamics. By contrast, Phy-Ture maintains relatively low errors across both gradual and abrupt motion patterns. These per-segment results complement the aggregate RMSE comparison and show that the performance advantage of Phy-Ture is maintained across diverse maneuver regimes.

In addition, we plot the measured and predicted linear and angular accelerations in Figures 7 and 8. It can be seen that, in addition to lowering the overall RMSE, Phy-Ture closely follows the observed accelerations over the displayed time segments, including sharp positive surges and steep negative drops. While certain segments still exhibit phase lag or amplitude discrepancies, Phy-Ture generally maintains closer alignment with the ground truth. By contrast, certain baselines show reduced responsiveness during abrupt maneuvers or visible temporal lag in some rapidly varying segments. Moreover, Phy-Ture maintains relatively stable predictions during flat and low-activity intervals.

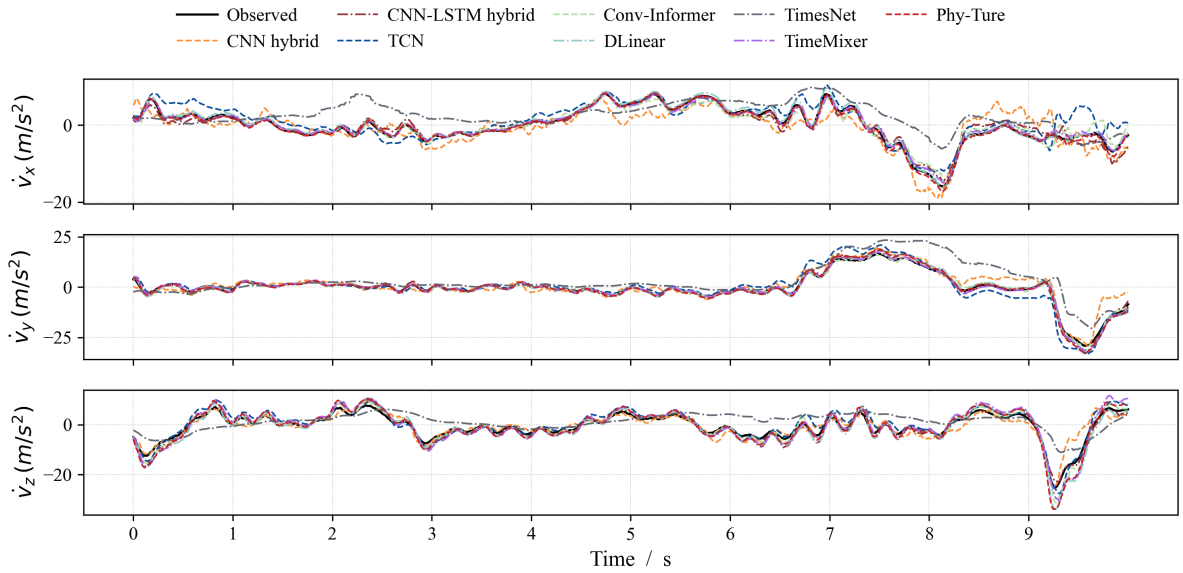

Figure 7. The predictive and observed linear accelerations curves at a certain time segment in the turn_demos1 trajectory.

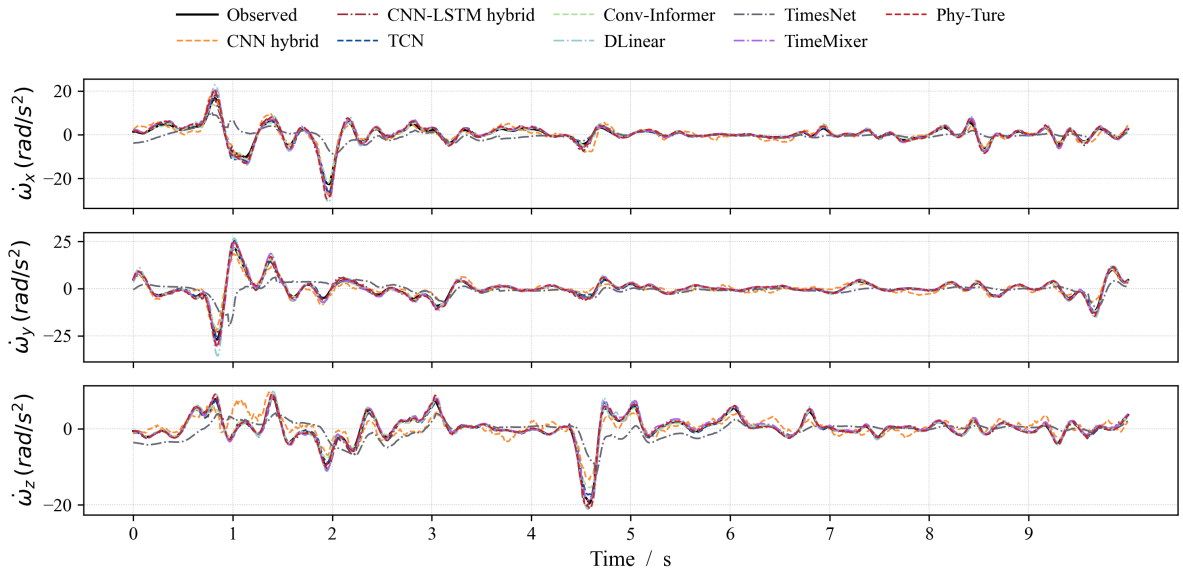

Figure 8. The predictive and observed angular accelerations curves at a certain time segment in the turn_demos1 trajectory.

4.4. Evaluation of Generalization Performance

To assess neural-model transfer to held-out flight trajectories and reduce dependence on a particular hold-out choice, we adopt a maneuver-class-based 10-fold protocol that is independent of the randomized 70%/20%/10% benchmark split. The complete dataset contains 20 maneuver classes, which are grouped into 10 mutually exclusive folds of two classes each. The 20 maneuver classes are sorted alphabetically by their canonical names in the dataset

and sequentially paired into 10 fixed folds. This deterministic partitioning is applied uniformly to all compared methods. In each fold, all samples from the two held-out classes are excluded before physical-model calibration, neural normalization, window construction, and network training. The physical model and residual network are then trained using only the remaining 18 maneuver classes, and evaluation is performed exclusively on the held-out pair. This procedure is repeated 10 times so that every maneuver class is evaluated once as held-out data. The reported results are aggregated across the 10 held-out folds.

As shown in Table 3, Phy-Ture achieves the lowest RMSE across all six output channels. Its mean RMSE is 0.42, compared with 0.95 for the strongest baseline, CNN–LSTM hybrid, corresponding to a reduction of approximately 56%.

Table 3. RMSE under the maneuver-class-based 10-fold evaluation.

Method	$\dot{v}_x$	$\dot{v}_y$	$\dot{v}_z$	$\dot{\omega}_x$	$\dot{\omega}_y$	$\dot{\omega}_z$	Mean
CNN hybrid	2.25	1.85	2.10	1.95	2.15	2.20	2.08
CNN–LSTM hybrid	0.88	0.79	1.03	0.89	1.06	1.03	0.95
TCN	1.95	1.87	2.13	1.39	1.61	1.28	1.71
TimesNet	1.35	1.20	1.12	1.18	1.30	0.96	1.19
TimeMixer	1.15	0.95	1.37	1.31	1.56	1.07	1.24
DLinear	1.67	1.55	1.32	0.83	0.88	0.92	1.20
Conv–Informer	1.65	1.15	1.25	1.45	1.40	1.10	1.33
Phy-Ture (Ours)	0.44	0.37	0.49	0.41	0.35	0.46	0.42

The CNN hybrid and CNN–LSTM hybrid similarly adopt physical-baseline-based residual learning. In contrast, Phy-Ture further introduces adaptive fusion between physical estimates and state–control features together with multiscale temporal processing. The angular-acceleration RMSEs of CNN–LSTM hybrid range from 0.89 to 1.06, compared with 0.35–0.46 for Phy-Ture. These results further support the effectiveness of the proposed physics-guided fusion and multiscale temporal modeling strategy.

Among the data-driven time-series baselines, TimesNet and TimeMixer achieve mean RMSEs of 1.19 and 1.24, respectively, while Phy-Ture remains below 0.50 on every output channel. Overall, the 10-fold results demonstrate that Phy-Ture maintains consistent predictive performance on maneuver classes excluded from training under the maneuver-class-based 10-fold protocol.

4.5. Ablation Study

To quantify the contribution of each key component in Phy-Ture, we perform a series of ablation experiments on the Stanford helicopter dataset. Three variants of the full model are evaluated, each with one module removed in turn:

- **Without PAF.** The PAF module is removed, and the only input to the MSTFE module is the position-encoded state-control feature sequence, without any physical prior fusion.
- **Without MSTFE.** The MSTFE module is removed so that the fused features bypass temporal feature enhancement and are directly fed into the decomposition–prediction head.
- **Without TSDP.** The series decomposition and dual-branch prediction are replaced by a single linear predictor.

Table 4 summarizes the RMSE of each model variant on linear acceleration channels ($\dot{v}_x, \dot{v}_y, \dot{v}_z$) and angular acceleration channels ($\dot{\omega}_x, \dot{\omega}_y, \dot{\omega}_z$). The results show that removing any of the three modules degrades the overall modeling performance, indicating that each component contributes to the predictive accuracy of the complete framework.

Table 4. Ablation study on different components of the phy-ture framework.

Method	$\dot{v}_x$	$\dot{v}_y$	$\dot{v}_z$	$\dot{\omega}_x$	$\dot{\omega}_y$	$\dot{\omega}_z$	Mean
Without PAF	0.48	0.34	0.45	0.43	0.32	0.38	0.40
Without MSTFE	2.84	1.81	1.80	0.32	0.34	0.30	1.24
Without TSDP	0.51	0.58	0.55	0.54	0.48	0.48	0.52
Phy-Ture (Ours)	0.37	0.32	0.42	0.31	0.29	0.23	0.32

Ablating the physics-augmented fusion module increases the global mean RMSE across all six channels from 0.32 to 0.40. Specifically, the RMSE of the linear-acceleration channels increases by 0.02–0.11, while that of the angular-acceleration channels increases by 0.03–0.15. These results support the contribution of adaptive physics–data fusion to residual prediction and show that removing PAF consistently degrades performance across both translational and rotational channels.

Removing the multiscale temporal feature enhancement module causes a substantial increase in the linear-acceleration RMSE, which rises to 2.84, 1.81, and 1.80 for $\dot{v}_x$, $\dot{v}_y$, and $\dot{v}_z$, respectively, while the angular-acceleration RMSE changes only moderately. This asymmetric degradation indicates that MSTFE contributes particularly strongly to the prediction of translational dynamics in the present dataset, while the rotational channels are comparatively less sensitive to its removal.

Removing MSTFE produces a markedly asymmetric degradation across the six output channels. To quantify this effect, we compute the relative RMSE increase caused by removing MSTFE for each output channel. Relative to the complete Phy-Ture model, the RMSE increases by approximately 667.6%, 465.6%, and 328.6% for $\dot{v}_x$, $\dot{v}_y$, and $\dot{v}_z$, respectively, whereas the corresponding increases for $\dot{\omega}_x$, $\dot{\omega}_y$, and $\dot{\omega}_z$ are 3.2%, 17.2%, and 30.4%. These results show that the contribution of MSTFE is concentrated primarily on the translational channels rather than being uniform across all six outputs. One possible interpretation is that the translational residuals in the present dataset contain stronger temporal structures that benefit from multiscale feature interaction, whereas the rotational residuals can already be modeled comparatively well by the remaining physical-baseline, fusion, and decomposition components. The ablation results therefore suggest that MSTFE primarily enhances the representation of multiscale temporal patterns associated with translational residual dynamics.

Finally, replacing the time-series decomposition and prediction module with a single linear projection layer raises the global mean RMSE from 0.32 to 0.52, corresponding to a 62.5% relative increase. This result supports the effectiveness of trend–seasonal decomposition and dual-branch prediction for improving residual modeling accuracy across both translational and rotational channels.

Overall, the ablation results show that PAF, MSTFE, and TSDP each contribute to the performance of Phy-Ture, and their combined use yields the lowest RMSE across all six output channels.

4.6. Sensitivity Analysis of the Moving-Average Window

To examine the influence of the moving-average window width in TSDP, we conduct a sensitivity analysis by varying w over $\{5, 15, 25, 35, 45\}$ while keeping the data partition, physical baseline, network architecture, and all other training configurations unchanged. At the sampling rate of 100 Hz, these settings correspond to local averaging intervals of 0.05 s, 0.15 s, 0.25 s, 0.35 s, and 0.45 s, respectively. Table 5 summarizes the resulting prediction errors.

Table 5. Sensitivity analysis of the moving-average window width w in TSDP.

w	$\dot{v}_x$	$\dot{v}_y$	$\dot{v}_z$	$\dot{\omega}_x$	$\dot{\omega}_y$	$\dot{\omega}_z$	Mean
5	0.44	0.40	0.49	0.36	0.35	0.29	0.39
15	0.40	0.31	0.44	0.33	0.31	0.22	0.34
25	0.37	0.32	0.42	0.31	0.29	0.23	0.32
35	0.37	0.32	0.43	0.32	0.33	0.24	0.34
45	0.41	0.39	0.43	0.35	0.39	0.26	0.37

As shown in Table 5, the overall prediction error decreases as the moving-average window increases from $w = 5$ to $w = 25$ and then increases for larger window sizes. The best overall performance is obtained at $w = 25$, with a mean summary score of 0.32. A relatively small window provides limited separation between the slowly varying trend and short-term fluctuations, whereas an excessively large window may smooth out transient dynamic variations. Performance remains relatively stable around $w = 25$, particularly for $w = 15$ and $w = 35$, indicating that Phy-Ture is not highly sensitive to moderate variations in the moving-average window width. Based on the best overall performance, $w = 25$ is adopted in all other experiments.

4.7. Comparison of Physical Information Fusion Strategies

To evaluate the effectiveness of the multi-head cross-attention fusion in Phy-Ture, we compare it against three alternative physics-integration strategies. All variants share the same network backbone and differ only in how the physics baseline estimates are merged with the state-control series:

- **State-Control Only.** Uses only the state-control input sequence as network input without any physical prediction information, serving as a purely data-driven baseline for dynamics modeling.
- **Direct Concatenation.** Directly concatenates physical baseline estimates with the state-control sequence in the feature dimension for subsequent temporal modeling, without adopting any residual learning paradigm.
- **Residual Correction.** Takes physical model predictions as the baseline output and learns the residual between predictions and ground truth, with no explicit physics-data fusion or physics-guided modeling.
- **Phy-Ture.** Adaptively fuses physical prediction sequences and state-control sequences to obtain a physics-guided feature representation, and performs residual learning to compensate for complex nonlinear and time-varying dynamic errors.

Table 6 reports the RMSE on linear ($\dot{v}_x, \dot{v}_y, \dot{v}_z$) and angular ($\dot{\omega}_x, \dot{\omega}_y, \dot{\omega}_z$) accelerations for each fusion strategy.

Table 6. Comparison of physical fusion methods

Method	$\dot{v}_x$	$\dot{v}_y$	$\dot{v}_z$	$\dot{\omega}_x$	$\dot{\omega}_y$	$\dot{\omega}_z$	Mean
State–Control Only	0.69	0.67	0.70	0.55	0.65	0.52	0.63
Direct Concatenation	0.57	0.54	0.48	0.49	0.53	0.46	0.51
Residual Correction	0.48	0.34	0.45	0.43	0.32	0.38	0.40
Phy-Ture (Ours)	0.37	0.32	0.42	0.31	0.29	0.23	0.32

As shown in Table 6, State–Control Only achieves a mean RMSE of 0.63 using only the state–control sequence. This result provides a purely data-driven reference under the same backbone and experimental setting.

Direct Concatenation introduces the physical baseline estimates by concatenating them with the state–control sequence, reducing the mean RMSE from 0.63 to 0.51. This improvement indicates that the physical baseline estimates provide useful complementary information. However, its performance remains inferior to residual correction and adaptive fusion, suggesting that simple feature concatenation is less effective than explicitly structuring the interaction between physical and data-driven information.

Residual Correction takes the physical model as a baseline and learns the dynamic residual, further reducing the mean RMSE to 0.40. This result indicates that learning the discrepancy relative to the physical baseline is more effective than directly concatenating physical baseline estimates with the state–control sequence.

The proposed Phy-Ture further introduces adaptive physics–data fusion within the residual-learning framework and achieves the lowest mean RMSE of 0.32. The improvement is particularly evident on the angular-acceleration channels, where Phy-Ture consistently achieves lower RMSE than the other evaluated fusion strategies.

Overall, the comparison shows that the manner in which physical information is incorporated has a substantial impact on prediction accuracy. Among the evaluated strategies, combining residual learning with adaptive physics–data fusion yields the best performance across all six output channels.

5. Conclusions

This paper presents Phy-Ture, a physics-guided fusion and feature processing framework for unmanned helicopter dynamics modeling. Starting from a formal problem definition over flight states, control inputs, and acceleration targets, we adopt a two-level hybrid strategy: the first-principles model serves as a structured baseline predictor, while the neural network focuses on learning nonlinear and time-varying residual errors that are difficult to capture analytically. This design retains the first-principles model as a structured physical reference while using data-driven residual learning to improve prediction accuracy across the flight regimes evaluated in this study.

On top of the residual-learning framework, Phy-Ture integrates three key mechanisms: physics-augmented fusion (PAF), multiscale temporal feature enhancement (MSTFE), and time-series decomposition and prediction (TSDP). Together, these modules support the modeling of dynamic processes across multiple time scales within the proposed residual-learning framework. Experiments on the Stanford Autonomous Helicopter Project, including benchmark comparison, maneuver-class-based 10-fold generalization evaluation, fusion-strategy comparison, and

ablation analysis, show that Phy-Ture achieves the lowest RMSE across all six linear and angular acceleration channels. Compared with the strongest baseline, the mean RMSE is reduced by approximately 48% in the benchmark evaluation and 56% in the 10-fold maneuver-class evaluation.

A systematic sensitivity analysis of the identified physical parameters is not included in the present study, so the influence of physical-parameter uncertainty on the final hybrid prediction remains to be quantified.

Future work will therefore investigate physical-parameter sensitivity and extend the current reduced-order physical model to account for effects omitted in the present formulation, including cross-axis inertia coupling, nonlinear rotor aerodynamics, and actuator coupling, while further evaluating the framework under additional flight conditions and rotorcraft platforms.

Author Contributions

B.Z.: Conceptualization, Methodology, Software, Formal analysis, Investigation, Writing—original draft; C.C.: Conceptualization, Supervision, Project administration, Funding acquisition, Writing—review & editing; J.F.: Investigation, Validation; H.C.: Methodology, Validation, Formal analysis, Writing—review & editing; Y.Y.: Investigation, Validation, Visualization; B.S.: Software, Data curation; M.L.: Methodology, Validation, Formal analysis, Writing—review & editing; Q.W.: Investigation, Data curation, Visualization, Validation; H.H.: Conceptualization, Methodology, Software, Formal analysis, Investigation, Writing—original draft. All authors have read and agreed to the published version of the manuscript.

Funding

This study was supported by the National Natural Science Foundation of China (Grant Nos. 62322601 and 62676053).

Institutional Review Board Statement

Not applicable.

Informed Consent Statement

Not applicable.

Data Availability Statement

Data will be made available on request.

Conflicts of Interest

The authors declare no conflict of interest.

Use of AI and AI-Assisted Technologies

During the preparation of this work, the authors used ChatGPT (OpenAI) for English-language polishing and wording refinement. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article.

References

1. Xian, B.; Zhang, X.; Zhang, H.; et al. Robust Adaptive Control for a Small Unmanned Helicopter Using Reinforcement Learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *33*, 7589–7597. https://doi.org/10.1109/tnnls.2021.3085767.
2. Zhao, Z.; Wu, J.; Mu, C.; et al. Neural-Network-Based Adaptive Fixed-Time Control for a 2-DOF Helicopter System With Input Quantization and Output Constraints. *IEEE Trans. Neural Netw. Learn. Syst.* **2025**, *36*, 7065–7076. https://doi.org/10.1109/tnnls.2024.3403145.
3. Zhao, Z.; Zhang, J.; Liu, Z.; et al. Adaptive Neural Network Control of an Uncertain 2-DOF Helicopter With Unknown Backlash-Like Hysteresis and Output Constraints. *IEEE Trans. Neural Netw. Learn. Syst.* **2023**, *34*, 10018–10027. https://doi.org/10.1109/tnnls.2022.3163572.
4. Yan, C.; Wang, C.; Xiang, X.; et al. Collision-Avoiding Flocking With Multiple Fixed-Wing UAVs in Obstacle-Cluttered Environments: A Task-Specific Curriculum-Based MADRL Approach. *IEEE Trans. Neural Netw. Learn. Syst.* **2024**, *35*, 10894–10908. https://doi.org/10.1109/tnnls.2023.3245124.
5. Singh, R.; Bhushan, B. Evolving Intelligent System for Trajectory Tracking of Unmanned Aerial Vehicles. *IEEE Trans. Autom. Sci. Eng.* **2022**, *19*, 1971–1984. https://doi.org/10.1109/tase.2021.3072339.

6. Wei, Z.; Shi, J.; Wang, Y.; et al. Fixed-Time Attitude Control of Flying-Wing Unmanned Aerial Vehicles Based on Plasma Control Surface. *Aerosp. Sci. Technol.* **2025**, *159*, 110016. https://doi.org/10.1016/j.ast.2025.110016.
7. Dong, L.; Jiang, F.; Wang, M.; et al. Deep Progressive Reinforcement Learning-Based Flexible Resource Scheduling Framework for IRS and UAV-Assisted MEC System. *IEEE Trans. Neural Netw. Learn. Syst.* **2025**, *36*, 2314–2326. https://doi.org/10.1109/tnnls.2023.3341067.
8. Xi, Y.Z.; Liao, G.X.; Lu, N.; et al. Study on the Aeromagnetic System between Fixed-Wing UAV and Unmanned Helicopter. *Minerals* **2023**, *13*, 700. https://doi.org/10.3390/min13050700.
9. Hua, H.; Fang, Y. A Novel Learning-Based Trajectory Generation Strategy for a Quadrotor. *IEEE Trans. Neural Netw. Learn. Syst.* **2024**, *35*, 9068–9079. https://doi.org/10.1109/tnnls.2022.3217814.
10. Dai, J.; Nie, H.; Ying, J.; et al. Modeling and Tracking Control of Unmanned Helicopter. In Proceedings of the 2020 IEEE 6th International Conference on Control Science and Systems Engineering (ICCSSE), Beijing, China, 17–19 July 2020; pp. 149–155. https://doi.org/10.1109/iccsse50399.2020.9171959.
11. Gu, X.; Xian, B.; Wang, Y. Geometry-Based Adaptive Tracking Control for an Underactuated Small-Size Unmanned Helicopter. *IEEE Trans. Syst. Man Cybern. Syst.* **2023**, *53*, 7489–7500. https://doi.org/10.1109/tsmc.2023.3298034.
12. He, Z.; He, X.; Javaid, U. Attitude Control of Unmanned Tandem Helicopter with Active Disturbance Rejection Capability. In Proceedings of the 2024 IEEE International Conference on Robotics and Biomimetics (ROBIO), Bangkok, Thailand, 10–14 December 2024; pp. 2105–2110. https://doi.org/10.1109/robio64047.2024.10907376.
13. Wang, G.; Ye, X.; Zhao, B. A Novel Remote Sensing Image Encryption Scheme Based on Block Period Arnold Scrambling. *Nonlinear Dyn.* **2024**, *112*, 17477–17507. https://doi.org/10.1007/s11071-024-09953-6.
14. Kim, S.K.; Tilbury, D.M. Mathematical Modeling and Experimental Identification of an Unmanned Helicopter Robot with Flybar Dynamics. *J. Robot. Syst.* **2004**, *21*, 95–116. https://doi.org/10.1002/rob.20002.
15. Han, S.; Wang, G.; Yan, Y. Dynamic Analysis and Model Identification of an Unmanned Helicopter. In Proceedings of the 2nd Asian/Australian Rotorcraft Forum and the 4th International Basic Research Conference on Rotorcraft Technology, Tianjin, China, 8–11 September 2013.
16. Fang, X.; Wu, A.; Shang, Y.; et al. A Novel Sliding Mode Controller for Small-Scale Unmanned Helicopters with Mismatched Disturbance. *Nonlinear Dyn.* **2016**, *83*, 1053–1068. https://doi.org/10.1007/s11071-015-2387-4.
17. Pan, Y.; Song, P.; Li, K.; et al. Attitude Dynamics Modeling of a Small-Scale Unmanned Helicopter. In Proceedings of the 2012 IEEE International Conference on Intelligent Control, Automatic Detection and High-End Equipment (ICADE), Beijing, China, 27–29 July 2012; pp. 84–88. https://doi.org/10.1109/icade.2012.6330104.
18. Khadeeja Nusrath, T.K.; Lulu, V.P.; Singh, J. System Identification of Flybar-Less Rotorcraft UAV. *Aircr. Eng. Aerosp. Technol.* **2020**, *92*, 1483–1493. https://doi.org/10.1108/aeat-05-2019-0100.
19. Liu, S.Y.; Zhou, J.; Shi, J.Y.; et al. Small Unmanned Helicopter System Identification Based on the Weighted Least Square Method and Improved Grey Wolf Optimisation Algorithm. *Aeronaut. J.* **2025**, *129*, 1886–1902. https://doi.org/10.1017/aer.2024.159.
20. Tischler, M.B.; Cauffman, M.G. Frequency-Response Method for Rotorcraft System Identification: Flight Applications to BO 105 Coupled Rotor/Fuselage Dynamics. *J. Am. Helicopter Soc.* **1992**, *37*, 3–17. https://doi.org/10.4050/jahs.37.3.
21. La Civita, M.; Messner, W.C.; Kanade, T. Modeling of small-scale helicopters with integrated first-principles and system-identification techniques. In Proceedings of the AHS International, 58th Annual Forum Proceedings, Montreal, QC, Canada, 11–13 June 2002; pp. 2505–2516. https://doi.org/10.4050/vfs-f58-00109.
22. Chaturvedi, N.A.; Sanyal, A.K.; Chellappa, M.; et al. Adaptive tracking of angular velocity for a planar rigid body with unknown models for inertia and input nonlinearity. *IEEE Trans. Control Syst. Technol.* **2006**, *14*, 613–627. https://doi.org/10.1109/tcst.2006.876628.
23. Tang, S.; Zheng, Z.; Qian, S.; et al. Nonlinear system identification of a small-scale unmanned helicopter. *Control Eng. Pract.* **2014**, *25*, 1–15. https://doi.org/10.1016/j.conengprac.2013.12.004.
24. Huang, L.; Pei, H.; Cheng, Z. System Identification and Improved Internal Model Control for Yaw of Unmanned Helicopter. *Asian J. Control* **2023**, *25*, 1619–1638. https://doi.org/10.1002/asjc.2963.
25. Pasquali, C.; Serafini, J.; Gennaretti, M.; et al. Reduced-Order Helicopter Model Identification from Closed-Loop Data. *Aerosp. Sci. Technol.* **2024**, *153*, 109419. https://doi.org/10.1016/j.ast.2024.109419.
26. Zhou, J.; Liu, S.; Lu, J.; et al. An Improved Dynamic Model Identification Method for Small Unmanned Helicopter. *Aircr. Eng. Aerosp. Technol.* **2024**, *96*, 175–183. https://doi.org/10.1108/aeat-05-2023-0145.
27. Mazzeo, F.; Pavel, M.D.; Fattizzo, D.; et al. Numerical Modeling, Trim, and Linearization of a Side-by-Side Helicopter in Hovering Conditions. *Aerospace* **2024**, *11*, 927. https://doi.org/10.3390/aerospace11110927.
28. Mazzeo, F.; Pavel, M.D.; Fattizzo, D.; et al. Flight Dynamic Modeling and Stability of a Small-Scale Side-by-Side Helicopter for Urban Air Mobility. *Aerosp. Sci. Technol.* **2024**, *148*, 109117. https://doi.org/10.1016/j.ast.2024.109117.
29. Wang, M.; Lou, X.; Wu, W.; et al. Koopman-Based MPC With Learned Dynamics: Hierarchical Neural Network Approach. *IEEE Trans. Neural Netw. Learn. Syst.* **2024**, *35*, 3630–3639. https://doi.org/10.1109/tnnls.2022.3194958.
30. Huang, X.; Zhang, H.T.; Wang, J. A Data-Driven Bayesian Koopman Learning Method for Modeling Hysteresis Dynamics. *IEEE Trans. Neural Netw. Learn. Syst.* **2024**, *35*, 15615–15623. https://doi.org/10.1109/tnnls.2023.3288752.

31. Mohammadzahri, M.; Khaleghifar, A.; Ghodsi, M.; et al. A Discrete Approach to Feedback Linearization, Yaw Control of an Unmanned Helicopter. *Unmanned Syst.* **2023**, *11*, 57–66. https://doi.org/10.1142/s2301385023500012.
32. Zhu, Y.; Li, C.; Zeng, R. A Survey of Learning in Optimal Control and Differential Game. *J. Mach. Learn. Inf. Secur.* **2026**, *2*, 4. https://doi.org/10.53941/jmlis.2026.100004.
33. Hou, Y.; Ye, X.; Zhang, X.; et al. Generating Multi-Scroll Hidden Attractors via Sigmoid Function. *Chaos Solitons Fractals* **2026**, *212*, 119023. https://doi.org/10.1016/j.chaos.2026.119023.
34. Zhang, X.; Hou, Y.; Wang, G.; et al. An Improved Sprott-E Chaotic System with Coexisting Attractors for Regional DNA Image Encryption. *Nonlinear Dyn.* **2026**, *114*, 1105. https://doi.org/10.1007/s11071-026-13012-7.
35. Mercorelli, P. A Wavelet Based Algorithm without a Priori Knowledge of Noise Level for Gross Errors Detection. In Proceedings of the 2013 International Conference on Advances in Intelligent Systems in Bioinformatics, Jakarta, Indonesia, 11 October 2013; pp. 7–12.
36. Peng, Z.; Wang, D.; Wang, J. Data-Driven Adaptive Disturbance Observers for Model-Free Trajectory Tracking Control of Maritime Autonomous Surface Ships. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *32*, 5584–5594. https://doi.org/10.1109/tnnls.2021.3093330.
37. Xiong, S.; Hou, Z. Data-Driven Formation Control for Unknown MIMO Nonlinear Discrete-Time Multi-Agent Systems With Sensor Fault. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *33*, 7728–7742. https://doi.org/10.1109/tnnls.2021.3087481.
38. Wang, C.; Huo, X.; Ma, K.; et al. PID-Like Model Free Adaptive Control With Discrete Extended State Observer and Its Application on an Unmanned Helicopter. *IEEE Trans. Ind. Inform.* **2023**, *19*, 11265–11274. https://doi.org/10.1109/tii.2023.3245223.
39. Punjani, A.; Abbeel, P. Deep Learning Helicopter Dynamics Models. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 26–30 May 2015; pp. 3223–3230. https://doi.org/10.1109/icra.2015.7139643.
40. Chen, S.; Cao, Y.; Kang, Y.; et al. Deep CNN Identifier for Dynamic Modelling of Unmanned Helicopter. In Proceedings of the Neural Information Processing (ICONIP), Guangzhou, China, 14–18 November 2017; pp. 51–60. https://doi.org/10.1007/978-3-319-70136-3_6.
41. Liu, Y.; Zhou, Y.; Li, X. Attitude Estimation of Unmanned Aerial Vehicle Based on LSTM Neural Network. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–6. https://doi.org/10.1109/ijcnn.2018.8489118.
42. Amiruddin, B.P.; Iskandar, E.; Fatoni, A.; et al. Deep Learning Based System Identification of Quadcopter Unmanned Aerial Vehicle. In Proceedings of the 2020 3rd International Conference on Information and Communications Technology (ICOIACT), Yogyakarta, Indonesia, 24–25 November 2020; pp. 165–169. https://doi.org/10.1109/icoiact50329.2020.9332059.
43. Candon, M.; Hale, E.; Balajewicz, M.; et al. Parameterization of nonlinear aeroelastic reduced order models via direct interpolation of Taylor partial derivatives. *Nonlinear Dyn.* **2024**, *112*, 17649–17670. https://doi.org/10.1007/s11071-024-09976-z.
44. Döring, F.A.; Wartmann, J.; Seher-Weiß, S. Data-Driven Approach for Fixed-Frame Rotorcraft Mast-Moment Estimation. *J. Guid. Control Dyn.* **2024**, *47*, 2301–2315. https://doi.org/10.2514/1.g007399.
45. Bejani, M.M.; Ghatee, M. A Systematic Review on Overfitting Control in Shallow and Deep Neural Networks. *Artif. Intell. Rev.* **2021**, *54*, 6391–6438. https://doi.org/10.1007/s10462-021-09975-1.
46. Hirtopanu, T.; Wang, Z.; Serrano, A.; et al. Uncertainty-Gated Mixture Modeling for Anomaly Detection in Human-in-the-Loop Vehicle Systems. *J. Mach. Learn. Inf. Secur.* **2026**, *2*, 10. https://doi.org/10.53941/jmlis.2026.100010.
47. Saviolo, A.; Li, G.; Loianno, G. Physics-Inspired Temporal Learning of Quadrotor Dynamics for Accurate Model Predictive Trajectory Tracking. *IEEE Robot. Autom. Lett.* **2022**, *7*, 10256–10263. https://doi.org/10.1109/lra.2022.3192609.
48. Yu, R.; Wang, R. Learning Dynamical Systems From Data: An Introduction to Physics-Guided Deep Learning. *Proc. Natl. Acad. Sci. USA* **2024**, *121*, e2311808121. https://doi.org/10.1073/pnas.2311808121.
49. Ma, J.; Zhu, Q.; Xue, T.; et al. Aircraft Dynamics Modeling at High Angles of Attack Incorporating Residual Transformer Autoencoder and Physical Mechanisms. *Aerosp. Sci. Technol.* **2025**, *160*, 110045. https://doi.org/10.1016/j.ast.2025.110045.
50. Kang, Y.; Chen, S.; Wang, X.; et al. Deep Convolutional Identifier for Dynamic Modeling and Adaptive Control of Unmanned Helicopter. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 524–538. https://doi.org/10.1109/tnnls.2018.2844173.
51. Zhang, H.; Liu, J. A Novel Residual Hybrid Dynamic Model for Unmanned Helicopters: Combining Both Physical and Deep Learning Models. *Nonlinear Dyn.* **2024**, *112*, 12235–12252. https://doi.org/10.1007/s11071-024-09689-3.
52. Mohajerin, N.; Waslander, S.L. Multistep Prediction of Dynamic Systems With Recurrent Neural Networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 3370–3383. https://doi.org/10.1109/tnnls.2019.2891257.
53. Xia, H. Modeling and Control Strategy of Small Unmanned Helicopter Rotation Based on Deep Learning. *Syst. Soft Comput.* **2024**, *6*, 200146. https://doi.org/10.1016/j.sasc.2024.200146.
54. Innocenti, M. Helicopter Flight Dynamics: The Theory and Application of Flying Qualities and Simulation Modeling. *J. Guid. Control Dyn.* **1999**, *22*, 383–384. https://doi.org/10.2514/2.4396.
55. Vaswani, A.; Shazeer, N.; Parmar, N.; et al. Attention Is All You Need. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 4–9 December 2017; pp. 6000–6010.
56. Liu, M.; Zeng, A.; Chen, M.; et al. SCINet: Time Series Modeling and Forecasting with Sample Convolution and Interaction. In Proceedings of the Neural Information Processing Systems (NeurIPS), New Orleans, LA, USA, 28 November–9 December 2022.

57. Abbeel, P.; Coates, A.; Ng, A.Y. Autonomous Helicopter Aerobatics through Apprenticeship Learning. *Int. J. Robot. Res.* **2010**, *29*, 1608–1639. https://doi.org/10.1177/0278364910371999.
58. Chen, S.; Kang, Y.; Di, J.; et al. Convex Temporal Convolutional Network–Based Distributed Cooperative Learning Control for Multiagent Systems. *IEEE Trans. Neural Netw. Learn. Syst.* **2023**, *34*, 5234–5243. https://doi.org/10.1109/tnnls.2022.3216327.
59. Wu, H.; Hu, T.; Liu, Y.; et al. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. In Proceedings of the International Conference on Learning Representations (ICLR), Kigali, Rwanda, 1–5 May 2023. https://arxiv.org/abs/2210.02186.
60. Wang, S.; Wu, H.; Shi, X.; et al. TimeMixer: Decomposable Multiscale Mixing for Time Series Forecasting. In Proceedings of the International Conference on Learning Representations (ICLR), Vienna, Austria, 7–11 May 2024. https://arxiv.org/abs/2405.14616.
61. Zeng, A.; Chen, M.; Zhang, L.; et al. Are Transformers Effective for Time Series Forecasting? In Proceedings of the AAAI Conference on Artificial Intelligence, Washington, DC, USA, 7–14 February 2023. https://doi.org/10.1609/aaai.v37i9.26317.
62. Wang, Y.; Dou, Y.; Peng, C.; et al. Multi Step Prediction Method of Ship Pitch Based on Conv-Informer Model. In Proceedings of the OCEANS 2023—Limerick, Limerick, Ireland, 5–8 June 2023; pp. 1–6. https://doi.org/10.1109/oceanslimerick52467.2023.10244592.