A Physics-Informed Neural Network Approach to Augmented Dynamics Visual Servoing of Multi-rotors

Archit Krishna Kamath[1], Sreenatha G. Anavatti[2] and Mir Feroskhan[1]

***Abstract*—This paper presents a visual servoing strategy that integrates the capabilities of a Physics-informed Neural Network (PINN) to estimate system uncertainties and inaccuracies with a dynamics-centered visual servoing technique for multi-rotors. The proposed method effectively combines these approaches, eliminating the need for inverse Jacobian calculations to determine multi-rotor motion by directly relating pixel variations to the multi-rotor's torque and thrust inputs, while also strengthening the method's robustness through the utilization of the PINN to model and address uncertainties in camera and multi-rotor parameters, as well as the modeling inaccuracies inherent in the dynamics-centered visual servoing technique. In contrast to existing state-of-the-art data-driven approaches, the proposed PINN approach requires, on average, 65% less labeled data to characterize uncertainties and inaccuracies. To ensure real-time implementation of the visual servoing model, the PINN-learned model is combined with an adaptive horizon monotonically weighted nonlinear model predictive controller (NMPC), capable of processing control efforts at rates 10 times faster than existing Tube MPC and Adaptive MPC strategies. These findings are validated through real-time trajectory tracking experiments, which not only highlight the effectiveness of the proposed approach in approximating modeling inaccuracies but also its capability in handling uncertainties upto 70% in camera parameters.**

***Index Terms*—Visual Servoing; Physics-informed Neural Network; Nonlinear Model Predictive Control; Autonomous Systems; Multi-rotor Control; Unmanned Aerial Vehicle (UAV).**

I. Introduction

THE progress in computer vision (CV) algorithms has spurred researchers worldwide to investigate ways to enhance the interaction between robotic systems and their surroundings. Numerous industrial applications, such as search and rescue, surveillance, driver assistance, and condition monitoring, have experienced substantial improvements through the seamless integration of CV algorithms with various robotic platforms, leading to efficient task execution [1], [2]. In this regard, multi-rotor unmanned aerial vehicles (UAVs) have emerged as a fitting choice for these applications due to their ability to vertically take off and land, achieve precise hovering, and perform agile maneuvers [3].

[1] Archit Krishna Kamath and Mir Feroskhan are with the School of Mechanical and Aerospace Engineering, Nanyang Technological University Singapore, Singapore, 639798. archit.kamath@ntu.edu.sg, mir.feroskhan@ntu.edu.sg

[2] Sreenatha G. Anavatti is with the School of Engineering and Information Technology, University of New South Wales Canberra, Australia, ACT2610. a.sreenatha@adfa.edu.au

The implementation of CV algorithms to control the motion of multi-rotors is known as visual servoing, and it is traditionally categorized into three main types: Position-Based Visual Servoing (PBVS), Image-Based Visual Servoing (IBVS), and Hybrid Visual Servoing (HVS) [4]–[6]. Extensive research has demonstrated the effectiveness of these approaches, particularly when combined with diverse controllers such as sliding mode strategies [7]–[9], model predictive strategies [10], and adaptive control strategies [11]. Nevertheless, these conventional visual servoing algorithms face several significant drawbacks that hinder their effectiveness, including dependency on inverse Jacobians for computing robot motions, leading to control singularities and increased computational demands during implementation, susceptibility to image noise even with robust robotic controllers, potential issues with target and actuator saturation, and errors arising from uncertain camera intrinsic parameters [12], [13].

Authors in [14] proposed an IBVS approach for manipulator control, utilizing a recurrent neural network controller to eliminate the need for inverse Jacobians. Similarly, in [15], a sampling-based model predictive controller (MPC) was introduced for visual servoing of a 6DoF manipulator, considering constraints and avoiding the use of inverse Jacobians for motion computation. However, it is important to highlight that the control methods suggested in [14], [15] involve a significant level of recursion, resulting in computational complexity when applied to systems like the multi-rotor. In another study, presented in [16], researchers introduced a dynamics-based visual servoing approach for a quadrotor, incorporating pixel velocities with the quadrotor states. The aim was to avoid the need for computing an inverse Jacobian. However, the mapping between pixel variations and the quadrotor's torque and thrust commands is not direct. It involves a series of cascaded intermediate blocks that first relate pixel position variations to pixel velocity and then relate pixel velocity variations to the quadrotor's thrust and torque commands. This introduces the requirement for additional outer loop control blocks in the multi-rotor's firmware, leading to increased computational burden and susceptibility to system noises. Therefore, it becomes essential to devise a dynamics-based visual servoing approach that can directly map pixel variations to the multi-rotor's input commands, eliminating the need for computing inverse Jacobians to estimate its motions. Such a mapping also enables the merging of image noises with the system uncertainties, allowing for the use of a single robust robotic controller to address both of these issues simultaneously.

Despite having such a mapping, the visual servoing algorithm remains susceptible to errors caused by uncertainties in camera parameters. To address this issue, researchers in [17] build upon the concept of a depth-independent interaction matrix developed in [16], proposing a novel algorithm for online estimation of unknown parameters. This approach combines the Slotine-Li method with the idea of motion in computer vision. Similarly, in [18], a related problem is tackled, where the extrinsic parameters of a camera are considered unknown, and a two-stage adaptive control law is developed for IBVS on a nonholonomic mobile robot. However, it is noteworthy that the development of such algorithms to support dynamics-based visual servoing approaches is currently lacking. Additionally, none of the aforementioned techniques address the challenges related to target loss and actuator saturation, which are common issues in visual servoing applications.

Authors in [19] tackle the challenges of target loss and actuator saturation by introducing a novel fuzzy sliding mode controller for IBVS of an omnidirectional mobile robot. However, a major drawback of this approach is its recursive nature, with the overall system's stability highly dependent on the evolution of the fuzzy network utilized. To address the issues of target loss and actuator saturation in visual servoing of a multi-rotor, researchers in [20]–[22] propose a robust nonlinear Model Predictive Control (NMPC) strategy for visual servoing of a quadrotor. While these approaches have demonstrated efficient real-time performance, NMPC methods require a complete nonlinear dynamical model for effective control. Incorporating camera and system uncertainties, as well as modeling inaccuracies, into the NMPC's cost function, can significantly increase the computational burden. Moreover, NMPC approaches necessitate the inclusion of terminal constraints to ensure stability, further exacerbating the computational requirements.

To efficiently handle system uncertainties and uphold constraints in visual servoing, one promising solution is to adopt a completely data-driven approach for modeling the dynamic visual servoing model, as suggested in [23]. Nevertheless, these approaches necessitate a substantial amount of labeled data to accurately capture the nonlinear dynamics of the system. In certain scenarios, such as visual servoing of multi-rotors, obtaining a sufficient volume of labeled data might be challenging and impractical. To address the collective shortcomings of the aforementioned approaches, this work makes the following contributions:

1) **Learning framework for visual servoing with limited training data**: Creating a framework that employs a physics-informed neural network to learn augmented dynamics, a direct correlation between pixel variations and multi-rotor thrust and torque inputs, facilitating efficient visual servoing while requiring, on average, 65% less training data compared to state-of-the-art data-driven approaches mentioned in [24] for simple circular trajectories.
2) **Catering to parametric uncertainties**: This paper presents results showcasing the performance of the proposed physics-informed neural network approach in visually servoing a ground vehicle across various trajectories with up to 70% parameter uncertainty. A comparison is drawn between the performance of this approach and state-of-the-art KNODE-MPC [25] and GP-MPC [26] approaches. Notably, despite the uncertainties, the proposed approach demonstrates 42.14% and 15.85% higher accuracy in tracking simple circular trajectories compared to the GP-MPC and KNODE-MPC approaches.
3) **Catering to modeling inaccuracies for real-time implementation**: The paper also emphasizes a crucial aspect of the proposed PINN approach as a dynamics approximator, incorporating real-time datasets to address modeling inaccuracies. This is showcased through real-time tracking experiments, revealing that the proposed approach is 43.82% and 14.57% more accurate compared to state-of-the-art KNODE-MPC [25] and GP-MPC [26] approaches.
4) **Real-time implementability**: To ensure real-time implementability of the learned model, the paper suggests coupling it with a monotonically weighted nonlinear model predictive controller (NMPC). This NMPC formulation integrates constraints imposed on the physics-aware augmented dynamics visual servoing model, effectively addressing concerns related to target loss and actuator saturation. The feasibility of real-time implementation is assessed using two popular state-of-the-art techniques, namely the Tube MPC [27] and the Adaptive MPC [28]. In the experimental results section of the paper, it is demonstrated that the proposed approach is nearly 10 times faster compared to the Tube MPC and Adaptive MPC approaches, rendering it a practical choice for real-time implementation.
5) **Comparison between Synthetic and Real-Time Training Data**: Typically, physics-informed neural networks incorporate a real-data loss term, necessitating experimental data to approximate system modeling inaccuracies. This set of experimental data can be generated synthetically using simulators like Gazebo, which feature physics engines, or obtained through actual real-time experiments, or a combination of both. This paper demonstrates that the proposed physics-informed neural network exhibits the ability to model system dynamics with reasonable precision, even with a moderate amount of available real-time experimental data. As a result, the trained network can be effectively employed to implement control strategies in real-time with high accuracy.

The paper's structure is as follows: Section II provides a concise introduction to Physics-Informed Neural Networks (PINNs), followed by the mathematical modeling of the augmented dynamics visual servoing approach in Section III. Section IV details the design of the monotonically weighted NMPC, followed by the overall implementation methodology in Section V. Section VI presents experimental validation for the proposed approach, encompassing hardware description, trajectory tracking performance analysis, analysis considering uncertainty in camera parameters, comparison with existing data-driven approaches, and real-time implementability analy-

sis. Finally, concluding remarks are provided in Section VII.

II. Physics-Informed Neural Networks

Physics-informed neural networks (PINNs) were introduced in 2019 for solving nonlinear differential equations [29]. PINNs can be used to approximate the dynamics of nonlinear systems with limited amount of labeled data as compared to complete data-driven models. The working of a PINN in solving the nonlinear dynamics of a system can be understood by considering a first order nonlinear differential equation given by:

$$\dot{x}(t) = f(x(t), u(t)) \tag{1}$$

Assuming that the function $f(x(t), u(t))$ is Lipschitz's continuous and $x(t) = x_0$ at $t = 0$ then the solution of the differential equation (1) over the time-interval T is given as $x[k+1] = F(T, x[k], u[k])$ where the function $F(T, x[k], u[k])$ is the actual solution of the nonlinear dynamics represented by (1) which can only be obtained if the function $f(x(t), u(t))$ is completely known. In real-time systems, the complete knowledge of the function $f(x(t), u(t))$ is not available and is often accompanied by uncertainties and inaccuracies in modeling. Only a partial understanding of the function $f(x(t), u(t))$ is known, referred to as the nominal dynamics of the system $f_N(x(t), u(t))$. Hence, the function $F(T, x[k], u[k])$ will have to be approximated with the help of a PINN, $\mathbf{\Phi}(T, x[k], u[k]; \mathbf{w})$, which is a fully connected neural network with $\mathbf{w}$ representing its trainable weight matrix and $(T, x[k], u[k])$ representing its inputs.

Typically, dynamical models of real-time systems encompass system uncertainties as well as modeling inaccuracies. Conventional data-driven approaches approximate $F(T, x[k], u[k])$ using large volumes of labeled data i.e. $(T, x[k], u[k])$ pairs, making them cumbersome. Additionally, the knowledge of the nominal dynamics $f_N(x(t), u(t))$ is not put to use in these approaches. PINNs, on the other hand, tend to incorporate the knowledge of $f_N(x(t), u(t))$ in approximating $F(T, x[k], u[k])$ making them data efficient.

To model system uncertainties, a physics loss $\mathcal{E}_p$ is computed. The expression for $\mathcal{E}_p \ \forall \ (x[i], u[i]) \in \mathcal{P}$ is given as:

$$\mathcal{E}_p = \frac{1}{N_p} \sum_{i=1}^{N_p} ||\dot{\mathbf{\Phi}}(T, x[i], u[i]; \mathbf{w}) - (f_N(x[i], u[i]) + f_p(x[i], u[i]))||^2 \tag{2}$$

where $\mathcal{P}$ represents the physics loss dataset which is obtained by considering $f_p(x(t), u(t))$ as an additive parametric uncertainty which can be sampled from standard distributions, such as a normal distribution parametrized by $\mathcal{N}(\mu, \sigma)$, μ and σ being the mean and standard deviations, respectively. N_p represents the size of the dataset $\mathcal{P}$ and the term $\dot{\mathbf{\Phi}}(T, x[i], u[i]; \mathbf{w})$ is the numerical derivative of the neural network which can be obtained by using the readily available autograd function.

The modeling inaccuracies are typically accounted for by using a real-data loss term $\mathcal{E}_r \ \forall \ (x[j], u[j], y[j]) \in \mathcal{D}$ which is defined as:

$$\mathcal{E}_r = \frac{1}{N_r} \sum_{j=1}^{N_r} ||\mathbf{\Phi}(T, x[j], u[j]; \mathbf{w}) - y[j]||^2 \tag{3}$$

where $\mathcal{D}$ represents the real-time dataset which is obtained by conducting real-time experiments on the dynamical system. Typically, a Proportional-Integral-Derivative (PID) controller is used to stabilize the nonlinear system to conduct these experiments and the data from these trails is recorded in the dataset $\mathcal{D}$. The output of the actual system for the control effort $u[j]$ and state feed-back $x[j]$ is represented by $y[j]$ and the size of the real-time dataset is given by N_r. The total composite loss for the PINN is given as $\mathcal{E}_p + \mathcal{E}_r$ which will be used alongside the back-propagation algorithm to train the network and obtain suitable values for the weight matrix $\mathbf{w}$.

In summary, a PINN is a powerful approach that combines neural network learning with physics-based constraints to improve predictions, especially in scenarios where traditional data-driven methods face challenges. PINNs excel in preserving the nonlinear nature of the modelled system, offering advantages such as the ability to build upon a nominal model. This unique capability allows the model to handle parametric uncertainty and modeling inaccuracies through distinct loss functions, denoted as $\mathcal{E}_p$ and $\mathcal{E}_r$. The resulting function approximator not only demonstrates accuracy in extrapolation but also requires significantly less data compared to conventional data-driven approaches. Section VII of the paper will thoroughly exploring these advantages and presenting the findings.

III. Mathematical Modeling

A. Nominal Model

This paper employs the idea presented in [30] to derive the nominal model for augmented dynamics visual servoing of a 6DoF multi-rotor. The state space representation of the multi-rotor model is depicted below [31], [32]:

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \\ \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -g \\ a_1\dot{\theta}\dot{\psi} \\ a_2\dot{\psi}\dot{\phi} \\ a_3\dot{\phi}\dot{\theta} \end{bmatrix} + \begin{bmatrix} \frac{1}{m} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{m} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{m} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{J_x} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{J_y} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{J_z} \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ u_z \\ u_\phi \\ u_\theta \\ u_\psi \end{bmatrix}$$
$$\implies \begin{bmatrix} \ddot{\boldsymbol{\eta}}_o \\ \ddot{\boldsymbol{\eta}}_i \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{f}}_o \\ \bar{\mathbf{f}}_i \end{bmatrix} + \begin{bmatrix} \mathbf{J}_o & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_i \end{bmatrix} \begin{bmatrix} \bar{\mathbf{u}}_o \\ \bar{\mathbf{u}}_i \end{bmatrix} + \begin{bmatrix} \mathbf{D}_o \\ \mathbf{D}_i \end{bmatrix} \tag{4}$$

where $[x, \ y, \ z] \in \mathbb{R}^3$ and $[\phi, \ \theta, \ \psi] \in \mathbb{R}^3$ represent the position, altitude, roll, pitch and yaw of the multi-rotor, satisfying the conditions $-\frac{\pi}{2} < \phi < \frac{\pi}{2}$, $-\frac{\pi}{2} < \theta < \frac{\pi}{2}$ and $-\pi < \psi < \pi$. The terms $[u_x, \ u_y, \ u_z] \in \mathbb{R}^3$ and $[u_\phi, \ u_\theta, \ u_\psi]$ represent the auxiliary control efforts and the main control effort in the outer loop and the inner loop of the multi-rotor, respectively. The terms $g \in \mathbb{R}^+$, $m \in \mathbb{R}^+$, $J_x \in \mathbb{R}^+$, $J_y \in \mathbb{R}^+$ and $J_z \in \mathbb{R}^+$ represent the acceleration due to gravity, mass of the multi-rotor and the inertia in the x, y and z directions, respectively. The terms a_1, a_2 and a_3 are defined as $a_1 = \frac{J_y - J_z}{J_x}$, $a_2 = \frac{J_z - J_x}{J_y}$ and $a_3 = \frac{J_x - J_y}{J_z}$.

The terms $\boldsymbol{\eta}_o \in \mathbb{R}^3 = [x \ y \ z]$ and $\boldsymbol{\eta}_i \in \mathbb{R}^3 = [\phi \ \theta \ \psi]$ represent the outer and inner loop state variables, while $\bar{\mathbf{f}}_o \in \mathbb{R}^3 = [0 \ 0 \ -g]^T$, $\bar{\mathbf{f}}_i \in \mathbb{R}^3 = [a_1\dot{\theta}\dot{\psi} \ \ a_2\dot{\psi}\dot{\phi} \ \ a_3\dot{\phi}\dot{\theta}]^T$, $\mathbf{J}_o \in \mathbb{R}^{3\times3} = \mathbf{diag}[\frac{1}{m} \ \frac{1}{m} \ \frac{1}{m}]$, and $\mathbf{J}_i \in \mathbb{R}^{3\times3} = \mathbf{diag}[\frac{1}{J_x} \ \frac{1}{J_y} \ \frac{1}{J_z}]$ represent the nonlinear dynamics of the multi-rotor. Additionally, $\bar{\mathbf{u}}_o \in \mathbb{R}^3 = [u_x \ u_y \ u_z]^T$ and $\bar{\mathbf{u}}_i \in \mathbb{R}^3 = [u_\phi \ u_\theta \ u_\psi]^T$

denote the auxiliary control effort in the outer loop and the actual control efforts of the inner loop of the multi-rotor. Furthermore, $\mathbf{D}_o \in \mathbb{R}^3$ and $\mathbf{D}_i \in \mathbb{R}^3$ represent the bounded disturbance in the outer loop and inner loop state variables, respectively. The relationship between the actual thrust input (u_T) of the outer loop and the auxiliary control efforts ($[u_x,\ u_y,\ u_z]$) and the desired values of the inner loop state variables (ϕ^d and θ^d) can be obtained using the relationships provided in [9].

To obtain the augmented dynamical multi-rotor model it is necessary to establish a relationship between pixel variations and the multi-rotor's state variables. To do so, consider the relationship between the motion of 'n' points of interest (POIs) in the camera plane and their velocities in the 3D plane:

$$\dot{\mathbf{r}} = \mathbf{\Lambda}\bar{\mathbf{V}} \tag{5}$$

where $\dot{\mathbf{r}} \in \mathbb{R}^{2n}$ represents the velocity of the 'n' POIs, $\bar{\mathbf{V}} \in \mathbb{R}^6 = \begin{bmatrix}\boldsymbol{\nu} & \boldsymbol{\omega}\end{bmatrix}^T$ with $\boldsymbol{\nu} \in \mathbb{R}^3$ and $\boldsymbol{\omega} \in \mathbb{R}^3$ representing the 3D linear and angular velocities of each POI in $\mathbf{r} = [u_1\ v_1\ u_2\ v_2\ \cdots\ u_n\ v_n]^T$. The matrix $\mathbf{\Lambda} \in \mathbb{R}^{2n\times 6}$ represents the stacked camera interaction matrix for 'n' POIs which is defined as shown:

$$\mathbf{\Lambda} = \begin{bmatrix} \frac{-f}{\mathbf{P}_{z1}} & 0 & \frac{u_1}{\mathbf{P}_{z1}} & \frac{u_1 v_1}{f} & -\left(f+\frac{u_1^2}{f}\right) & v_1 \\ 0 & \frac{-f}{\mathbf{P}_{z1}} & \frac{v_1}{\mathbf{P}_{z1}} & \left(f+\frac{v_1^2}{f}\right) & -\frac{u_1 v_1}{f} & -u_1 \\ \frac{-f}{\mathbf{P}_{z2}} & 0 & \frac{u_2}{\mathbf{P}_{z2}} & \frac{u_2 v_2}{f} & -\left(f+\frac{u_2^2}{f}\right) & v_2 \\ 0 & \frac{-f}{\mathbf{P}_{z2}} & \frac{v_2}{\mathbf{P}_{z2}} & \left(f+\frac{v_2^2}{f}\right) & -\frac{u_2 v_2}{f} & -u_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{-f}{\mathbf{P}_{zn}} & 0 & \frac{u_n}{\mathbf{P}_{zn}} & \frac{u_n v_n}{f} & -\left(f+\frac{u_n^2}{f}\right) & v_n \\ 0 & \frac{-f}{\mathbf{P}_{zn}} & \frac{v_n}{\mathbf{P}_{zn}} & \left(f+\frac{v_n^2}{f}\right) & -\frac{u_n v_n}{f} & -u_n \end{bmatrix}$$

where $f \in \mathbb{R}$ represents the focal length, u_i, v_i and $\mathbf{P}_{zi}$ represents the pixel address and the depth of the i^{th} POI, respectively. Since pixel accelerations will be needed for the augmented dynamical model (5) needs to be differentiated with respect to time as follows:

$$\ddot{\mathbf{r}} = \mathbf{\Lambda}\bar{\mathbf{A}} + \begin{bmatrix} \bar{\mathbf{V}}^T\Delta_1\bar{\mathbf{V}} \\ \bar{\mathbf{V}}^T\Xi_1\bar{\mathbf{V}} \\ \bar{\mathbf{V}}^T\Delta_2\bar{\mathbf{V}} \\ \bar{\mathbf{V}}^T\Xi_2\bar{\mathbf{V}} \\ \vdots \\ \bar{\mathbf{V}}^T\Delta_n\bar{\mathbf{V}} \\ \bar{\mathbf{V}}^T\Xi_n\bar{\mathbf{V}} \end{bmatrix} + \begin{bmatrix} \mathbf{D}_\mathbf{r}^\mathbf{1} \\ \mathbf{D}_\mathbf{r}^\mathbf{2} \\ \vdots \\ \mathbf{D}_\mathbf{r}^\mathbf{n} \end{bmatrix} \tag{6}$$

where $\bar{\mathbf{A}} \in \mathbb{R}^6 = \begin{bmatrix}\mathbf{a} & \boldsymbol{\alpha}\end{bmatrix}^T$ with $\mathbf{a} \in \mathbb{R}^3$, $\boldsymbol{\alpha} \in \mathbb{R}^3$ representing the 3D linear and angular accelerations of the POIs, respectively and $\Delta_i \in \mathbb{R}^{6\times 6}$ and $\Xi_i \in \mathbb{R}^{6\times 6}$ are expressed as in (7) and $\mathbf{D}_\mathbf{r}^\mathbf{i} \in \mathbb{R}^2$ represents the bounded image disturbance vector associated with the i^{th} POI.

To obtain the augmented dynamical model, the outer loop dynamics of (4) and (6) can be combined as follows:

$$\ddot{\boldsymbol{\eta}}_o = \bar{\mathbf{f}}_o + \mathbf{J}_o\bar{\mathbf{u}}_o + \mathbf{D}_o \tag{8}$$

Since pixel acceleration in (6) is for 2D pixel values and the outer loop dynamics of the multi-rotor has three variables, the relationship between them will have to be obtained by splitting the outer loop dynamics into two subsections as shown:

$$\begin{bmatrix}\ddot{\eta}_{dep} \\ \ddot{\boldsymbol{\eta}}_{pos}\end{bmatrix} = \begin{bmatrix}\bar{f}_{dep} \\ \bar{\mathbf{f}}_{pos}\end{bmatrix} + \begin{bmatrix}J_{dep} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_{pos}\end{bmatrix}\begin{bmatrix}\bar{u}_{dep} \\ \bar{\mathbf{u}}_{pos}\end{bmatrix} + \begin{bmatrix}D_{dep} \\ \mathbf{D}_{pos}\end{bmatrix} \tag{9}$$

where the subscripts dep and pos represent the depth and the position subsystem, respectively. For instance, if the camera setup is as depicted in Fig. 1, the dep parameter would align with the z-axis of the multi-rotor and the components of $\boldsymbol{\eta}_{pos} \in \mathbb{R}^2$ would align with the x and y of the multi-rotor.

The augmented dynamical model can be obtained using the pos subsystem alongside (6). To relate the variables of (6) with that of (9) a rotation matrix $\mathbf{R}_{cam}^{UAV} \in \mathbb{R}^{2\times 2}$ is considered that relates the multi-rotor's body frame and the camera frame. The product $\mathbf{R}_{cam}^{UAV}\dot{\boldsymbol{\eta}}_{pos}\mathbb{R}^2$ and $\mathbf{R}_{cam}^{UAV}\ddot{\boldsymbol{\eta}}_{pos} \in \mathbb{R}^2$ are related to $\bar{\mathbf{V}}$ and $\bar{\mathbf{A}}$ as:

$$\begin{aligned}\bar{\mathbf{V}} &= \begin{bmatrix}\mathbf{R}_{cam}^{UAV}\dot{\boldsymbol{\eta}}_{pos} & 0 & 0 & 0 & 0\end{bmatrix}^T \\ \bar{\mathbf{A}} &= \begin{bmatrix}\mathbf{R}_{cam}^{UAV}\ddot{\boldsymbol{\eta}}_{pos} & 0 & 0 & 0 & 0\end{bmatrix}^T\end{aligned} \tag{10}$$

Substituting (10) in (6) and eliminating the zero columns:

$$\ddot{\mathbf{r}} = \mathbf{\Lambda}\begin{bmatrix}\mathbf{R}_{cam}^{UAV}\ddot{\boldsymbol{\eta}}_{pos} \\ 0 \\ 0 \\ 0 \\ 0\end{bmatrix} + \begin{bmatrix} \bar{\mathbf{V}}^T\Delta_1\bar{\mathbf{V}} \\ \bar{\mathbf{V}}^T\Xi_1\bar{\mathbf{V}} \\ \bar{\mathbf{V}}^T\Delta_2\bar{\mathbf{V}} \\ \bar{\mathbf{V}}^T\Xi_2\bar{\mathbf{V}} \\ \vdots \\ \bar{\mathbf{V}}^T\Delta_n\bar{\mathbf{V}} \\ \bar{\mathbf{V}}^T\Xi_n\bar{\mathbf{V}} \end{bmatrix} + \begin{bmatrix} \mathbf{D}_\mathbf{r}^\mathbf{1} \\ \mathbf{D}_\mathbf{r}^\mathbf{2} \\ \vdots \\ \mathbf{D}_\mathbf{r}^\mathbf{n} \end{bmatrix} \tag{11}$$

For ease of representation, (11) can be rewritten as:

$$\ddot{\mathbf{r}} = \mathbf{F}_{aug} + \mathbf{J}_{aug}\bar{\mathbf{u}}_{pos} + \mathbf{D}_{aug} \tag{12}$$

where $\mathbf{F}_{aug} \in \mathbb{R}^{2n}$, $\mathbf{J}_{aug} \in \mathbb{R}^{2n\times 2}$ and $\mathbf{D}_{aug} \in \mathbb{R}^{2n}$ (transformed bounded disturbance vector) are functions of $\mathbf{r}$, $\dot{\mathbf{r}}$, $\boldsymbol{\eta}_{pos}$ and $\dot{\boldsymbol{\eta}}_{pos}$. From (9) and (12), the overall outer loop dynamics are given as:

$$\begin{bmatrix}\ddot{\eta}_{dep} \\ \ddot{\mathbf{r}}\end{bmatrix} = \begin{bmatrix}\bar{f}_{dep} \\ \mathbf{F}_{aug}\end{bmatrix} + \begin{bmatrix}J_{dep} & \mathbf{0}_{2n\times 1} \\ \mathbf{0} & \mathbf{J}_{aug}\end{bmatrix}\begin{bmatrix}\bar{u}_{dep} \\ \bar{\mathbf{u}}_{pos}\end{bmatrix} + \begin{bmatrix}D_{dep} \\ \mathbf{D}_{aug}\end{bmatrix} \tag{13}$$

Therefore, the outer loop control is now split into two subsystems: the depth subsystem and the augmented dynamics position subsystem. The state of the multi-rotor in each of these subsystems is influenced by the camera's orientation in relation to the multi-rotor. The complete state model for the augmented dynamics visual servoing of a multi-rotor is:

$$\underbrace{\begin{bmatrix}\ddot{\eta}_{dep} \\ \ddot{\mathbf{r}} \\ \ddot{\boldsymbol{\eta}}_i\end{bmatrix}}_{\dot{\boldsymbol{\eta}}_{\text{ADVS}}} = \underbrace{\begin{bmatrix}\bar{f}_{dep} \\ \mathbf{F}_{aug} \\ \bar{\mathbf{f}}_i\end{bmatrix} + \begin{bmatrix}J_{dep} \\ \mathbf{J}_{aug} \\ \mathbf{J}_i\end{bmatrix}\begin{bmatrix}\bar{u}_{dep} \\ \bar{\mathbf{u}}_{pos} \\ \bar{\mathbf{u}}_i\end{bmatrix}}_{\mathbf{f}_{N-\text{ADVS}}(\eta_{dep},\mathbf{r},\boldsymbol{\eta}_i,\bar{u}_{dep},\bar{\mathbf{u}}_{pos},\bar{\mathbf{u}}_i)} + \underbrace{\begin{bmatrix}D_{dep} \\ \mathbf{D}_{aug} \\ \mathbf{D}_i\end{bmatrix}}_{\mathbf{D}_{ADVS}}$$

$$\implies \dot{\boldsymbol{\eta}}_{\text{ADVS}} = \mathbf{f}_{\text{ADVS}}(\eta_{dep}, \mathbf{r}, \boldsymbol{\eta}_i, \bar{u}_{dep}, \bar{\mathbf{u}}_{pos}, \bar{\mathbf{u}}_i) \tag{14}$$

where $\dot{\boldsymbol{\eta}}_{\text{ADVS}} \in \mathbb{R}^{4n+8}$ represents the overall states of the augmented dynamics model, $\mathbf{f}_{N-\text{ADVS}} \in \mathbb{R}^{4n+8}$ represents the nominal model for the augmented dynamics and $\mathbf{D}_{ADVS} \in \mathbb{R}^{4n+8}$ represents the merged disturbance term.

$$\Delta_i = \begin{bmatrix} 0 & 0 & \frac{f}{\mathbf{P}_{z_i}^2} & -\frac{v_i}{\mathbf{P}_{z_i}} & \frac{3u_i}{2\mathbf{P}_{z_i}} & 0 \\ 0 & 0 & 0 & -\frac{u_i}{2\mathbf{P}_{z_i}} & 0 & -\frac{f}{2\mathbf{P}_{z_i}} \\ \frac{f}{\mathbf{P}_{z_i}^2} & 0 & \frac{2u_i}{\mathbf{P}_{z_i}^2} & \frac{2u_iv_i}{f\mathbf{P}_{z_i}} & -\frac{f}{2\mathbf{P}_{z_i}} - \frac{2u_i^2}{f\mathbf{P}_{z_i}} & \frac{v_i}{\mathbf{P}_{z_i}} \\ -\frac{v_i}{\mathbf{P}_{z_i}} & -\frac{u_i}{2\mathbf{P}_{z_i}} & \frac{2u_iv_i}{f\mathbf{P}_{z_i}} & u_i + \frac{2u_iv_i^2}{f^2} & -\frac{v_i}{2} - \frac{2u_i^2v_i}{f^2} & \frac{f}{2} - \frac{u_i^2}{2f} + \frac{v_i^2}{f} \\ \frac{3u_i}{2\mathbf{P}_{z_i}} & 0 & -\frac{f}{2\mathbf{P}_{z_i}} - \frac{2u_i^2}{f\mathbf{P}_{z_i}} & -\frac{v_i}{2} - \frac{2u_i^2v_i}{f^2} & 2u_i + \frac{2u_i^3}{f^2} & -\frac{3u_iv_i}{2f} \\ 0 & -\frac{f}{2\mathbf{P}_{z_i}} & \frac{v_i}{\mathbf{P}_{z_i}} & \frac{f}{2} - \frac{u_i^2}{2f} + \frac{v_i^2}{f} & -\frac{3u_iv_i}{2f} & -u_i \end{bmatrix}$$

$$\Xi_i = \begin{bmatrix} 0 & 0 & 0 & 0 & \frac{2v_i}{\mathbf{P}_{z_i}} & \frac{f}{2\mathbf{P}_{z_i}} \\ 0 & 0 & \frac{f}{\mathbf{P}_{z_i}^2} & -\frac{3v_i}{2\mathbf{P}_{z_i}} & \frac{u_i}{\mathbf{P}_{z_i}} & 0 \\ 0 & \frac{f}{\mathbf{P}_{z_i}^2} & \frac{2v_i}{\mathbf{P}_{z_i}^2} & \frac{f}{2\mathbf{P}_{z_i}} + \frac{2v_i^2}{f\mathbf{P}_{z_i}} & -\frac{u_iv_i}{f\mathbf{P}_{z_i}} & -\frac{u_i}{\mathbf{P}_{z_i}} \\ 0 & -\frac{3v_i}{2\mathbf{P}_{z_i}} & -\frac{f}{2\mathbf{P}_{z_i}} + \frac{2v_i^2}{f\mathbf{P}_{z_i}} & 2v_i + \frac{2v_i^3}{f^2} & -\frac{u_i}{2} - \frac{2u_iv_i^2}{f^2} & -\frac{3u_iv_i}{2f} \\ \frac{f}{2\mathbf{P}_{z_i}} & \frac{u_i}{v_i} & -\frac{u_iv_i}{f\mathbf{P}_{z_i}} & -\frac{u_i}{2} - \frac{2u_iv_i^2}{f^2} & v_i + \frac{2u_i^2v_i}{f^2} & \frac{f}{2} + \frac{u_i^2}{f} - \frac{v_i^2}{2f} \\ \frac{f}{2\mathbf{P}_{z_i}} & 0 & -\frac{u_i}{\mathbf{P}_{z_i}} & \frac{3u_iv_i}{2f} & \frac{f}{2} + \frac{u_i^2}{f} - \frac{v_i^2}{2f} & -v_i \end{bmatrix} \tag{7}$$

The merged disturbance term can encompass system uncertainties, image noises as well as modeling inaccuracies. $\mathbf{f}_{\text{ADVS}}(\eta_{dep}, \mathbf{r}, \boldsymbol{\eta}_i, \bar{u}_{dep}, \bar{\mathbf{u}}_{pos}, \bar{\mathbf{u}}_i) \in \mathbb{R}^{4n+8}$ represents the complete nonlinear model of the augmented dynamics visual servoing approach.

The equation $\dot{\boldsymbol{\eta}}_{\text{ADVS}} = \mathbf{f}_{N-\text{ADVS}}(\eta_{dep}, \mathbf{r}, \boldsymbol{\eta}_i, \bar{u}_{dep}, \bar{\mathbf{u}}_{pos}, \bar{\mathbf{u}}_i) + \mathbf{D}_{ADVS}$ is obtained by converting each of the second order differential equation into two first order differential equations. For future sections of this work $\mathbf{f}_{N-\text{ADVS}}(\eta_{dep}, \mathbf{r}, \boldsymbol{\eta}_i, \bar{u}_{dep}, \bar{\mathbf{u}}_{pos}, \bar{\mathbf{u}}_i)$ and $\mathbf{f}_{\text{ADVS}}(\eta_{dep}, \mathbf{r}, \boldsymbol{\eta}_i, \bar{u}_{dep}, \bar{\mathbf{u}}_{pos}, \bar{\mathbf{u}}_i)$ will be referred to as $\mathbf{f}_{N-\text{ADVS}}$ and $\mathbf{f}_{\text{ADVS}}$ which are continuous-time functions representing the nominal and the total dynamics of the system, respectively. As explained in the preceding section of this work, the nonlinear dynamical differential equation (14) can be solved as an initial value problem as the values of the state variables η_{dep}, $\mathbf{r}$ and $\boldsymbol{\eta}_i$ at $t = 0$ is known. The solution of (14), for a sampling time-interval of T, is:

$$\begin{aligned} &\int_{t=k}^{t=k+1} d\boldsymbol{\eta}_{\text{ADVS}} = \int_{t=0}^{t=T} \mathbf{f}_{\text{ADVS}} dt \\ &\implies \boldsymbol{\eta}_{\text{ADVS}}[k+1] = \boldsymbol{\eta}_{\text{ADVS}}[k] + \int_{t=0}^{t=T} \mathbf{f}_{\text{ADVS}} dt \\ &\implies \boldsymbol{\eta}_{\text{ADVS}}[k+1] = \mathbf{F}_{\text{ADVS}}(T, \boldsymbol{\eta}_{\text{ADVS}}[k], \bar{\mathbf{u}}[k]) \\ &\implies \boldsymbol{\eta}_{\text{ADVS}}[k+1] = \mathbf{F}_{\text{ADVS}}(T, k) \end{aligned} \tag{15}$$

where $\boldsymbol{\eta}_{\text{ADVS}}[k] = [\eta_{dep}[k] \ \mathbf{r}[k] \ \boldsymbol{\eta}_i[k]]$ and $\bar{\mathbf{u}}[k] = [\bar{u}_{dep}[k] \ \bar{\mathbf{u}}_{pos}[k] \ \bar{\mathbf{u}}_i[k]]$. The following section of this work describes the development of a PINN structure that can be used to accurately approximate the function $\mathbf{F}_{\text{ADVS}}$ using the nominal dynamics $\mathbf{f}_{N-\text{ADVS}}$ and additional datasets to capture system uncertainties and modeling inaccuracies.

Remark 1: The transformed bounded disturbance vector i.e. $\mathbf{D}_{aug}$ combines the system disturbance $\mathbf{D}_{pos}$ and the bounded image noise $\mathbf{D}_{\mathbf{r}}^{\mathbf{i}}$ for each POI being tracked. This merger facilitates in designing of a single controller to tackle both.

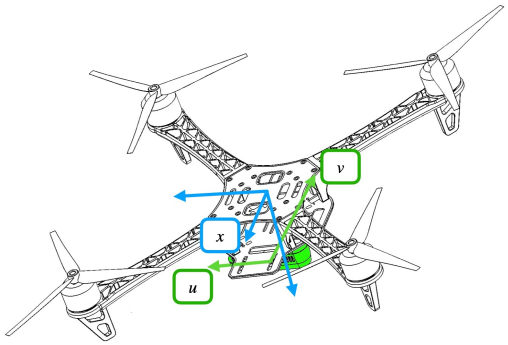


Fig. 1: Multi-rotor and camera setup.

B. System Uncertainties and Modeling Inaccuracies

To approximate the system uncertainties and modeling inaccuracies consider a fully connected neural network $\boldsymbol{\Phi}_{\text{ADVS}}(T, k; \mathbf{w}) = \boldsymbol{\Phi}_{\text{ADVS}}(T, \boldsymbol{\eta}_{\text{ADVS}}[k], \bar{\mathbf{u}}[k]; \mathbf{w})$ to approximate the function $\mathbf{F}_{\text{ADVS}}(T, k)$. System uncertainties are estimated using the physics loss term which is modelled as an additive parametric uncertainty and encompasses uncertainties in the camera parameters as well as multi-rotor's mass and inertia terms. Such deviations can be sampled from standard distributions, such as a normal distribution parametrized by $\mathcal{N}(\mu, \sigma)$, μ and σ being the mean and standard deviations, respectively. Since this additive term is included alongside the nominal dynamics, the physics loss $\mathcal{E}_p \ \forall \ (\boldsymbol{\eta}_{\text{ADVS}}[i], \bar{\mathbf{u}}[i]) \in \mathcal{P}$ of the PINN is obtained as follows:

$$\mathcal{E}_p = \frac{1}{N_p} \sum_{i=1}^{N_p} ||\dot{\boldsymbol{\Phi}}_{\text{ADVS}}(T, i; \mathbf{w}) - (\mathbf{f}_{N-\text{ADVS}} + \mathbf{f}_{p-\text{ADVS}})||^2 \tag{16}$$

where $\mathbf{f}_{p-\text{ADVS}}$ is the additive parametric uncertainty function which is sampled from a standard normal distribution, $\mathcal{P}$ represents the physics loss dataset and N_p represents the size of the physics loss dataset. The function $\mathbf{f}_{N-\text{ADVS}}+\mathbf{f}_{p-\text{ADVS}}$ is computed at every i^{th} time sample by substituting the values of $\boldsymbol{\eta}_{\text{ADVS}}[i], \bar{\mathbf{u}}[i]$ in $\mathbf{f}_{N-\text{ADVS}}$ and $\mathbf{f}_{p-\text{ADVS}}$.

Modeling inaccuracies can arise due to external environmental conditions, such as wind, as well as nonlinear terms corresponding to the system's unmodeled internal dynamics, such as frictional effects due to the rotor's inertia and the elasticity of the propellers. These effects are typically tough to model into the multi-rotor dynamics and can only be captured by performing real-time tests on the system. To facilitate the addition of such effects, a non-parametric function $\mathbf{f}_{r-\text{ADVS}}$ has to be added to the existing nominal dynamics $\mathbf{f}_{N-\text{ADVS}}$ and the parametric uncertainty term $\mathbf{f}_{p-\text{ADVS}}$. Hence, the function $\mathbf{f}_{\text{ADVS}} = \mathbf{f}_{N-\text{ADVS}} + \mathbf{f}_{p-\text{ADVS}} + \mathbf{f}_{r-\text{ADVS}}$. Including the non-parametric function into the system dynamics, (15) is re-written as:

$$\begin{aligned}\int_{t=k}^{t=k+1} d\boldsymbol{\eta}_{\text{ADVS}} &= \int_{t=0}^{t=T} \mathbf{f}_{\text{ADVS}} dt \\ \boldsymbol{\eta}_{\text{ADVS}}[k+1] &= \boldsymbol{\eta}_{\text{ADVS}}[k] + \int_{t=0}^{t=T} (\mathbf{f}_{N-\text{ADVS}} + \mathbf{f}_{p-\text{ADVS}}) dt \\ &\quad + \mathbf{f}_{r-\text{ADVS}} = \mathbf{F}_{\text{ADVS}_{\text{true}}}(T,k) \end{aligned} \tag{17}$$

where the expression $\mathbf{F}_{\text{ADVS}_{\text{true}}}(T,k) = \boldsymbol{\eta}_{\text{ADVS}}[k] + \int_{t=0}^{t=T}(\mathbf{f}_{N-\text{ADVS}} + \mathbf{f}_{p-\text{ADVS}})dt + \mathbf{f}_{r-\text{ADVS}}$ now represents the true real-time dynamics of the system under consideration, inclusive of the nominal model, physics-loss function and the real data loss function. Since the objective of the PINN is to approximate $\boldsymbol{\eta}_{\text{ADVS}}[k+1] = \mathbf{F}_{\text{ADVS}_{\text{true}}}(T,k)$ to $\boldsymbol{\eta}_{\text{ADVS}}[k+1] = \boldsymbol{\Phi}_{\text{ADVS}}(T,k;\mathbf{w})$ the real data loss function will have to be approximated using the loss function $\mathcal{E}_r \ \forall \ (\boldsymbol{\eta}_{\text{ADVS}}[j], \bar{\mathbf{u}}[j], \mathbf{y}[j]) \in \mathcal{D}$ which is defined as:

$$\begin{aligned}\mathcal{E}_r &= \frac{1}{N_r}\sum_{j=1}^{N_r} ||\boldsymbol{\Phi}_{\text{ADVS}}(T,j;\mathbf{w}) - \mathbf{F}_{\text{ADVS}_{\text{true}}}(T,j)||^2 \\ &= \frac{1}{N_r}\sum_{j=1}^{N_r} ||\boldsymbol{\Phi}_{\text{ADVS}}(T,j;\mathbf{w}) - \mathbf{y}[j]||^2 \end{aligned} \tag{18}$$

where $\mathcal{D}$ represents the real-time dataset and N_r represents the size of $\mathcal{D}$ and $\mathbf{y}[j] = \mathbf{F}_{\text{ADVS}_{\text{true}}}(T,j)$ represents the output state vector of the system obtained by conducting real-time experiments for gathering $\mathcal{D}$.

The experimental section of this paper explains details on the method used to collect the datasets $\mathcal{P}$ and $\mathcal{D}$. The cumulative loss function that is used to train the PINN is obtained by considering $\mathcal{E}_T = \mathcal{E}_p + \mathcal{E}_r$. The weight matrix $\mathbf{w}$ is obtained using the back-propagation algorithm and is tuned to minimize $\mathcal{E}_T$. Once fully trained the PINN $\mathbf{F}_{\text{ADVS}}(T,k)$ can be used as a replacement for $\boldsymbol{\Phi}_{\text{ADVS}}(T,k;\mathbf{w})$ and the overall system transforms into a physics-aware augmented dynamical model that can be used for visual servoing applications. The discrete-time representation of this trained PINN model is:

$$\boldsymbol{\eta}_{\text{ADVS}}[k+1] = \boldsymbol{\Phi}_{\text{ADVS}}(T,k;\mathbf{w}) \tag{19}$$

where $\boldsymbol{\Phi}_{\text{ADVS}}(T,k;\mathbf{w}) = \boldsymbol{\Phi}_{\text{ADVS}}(T,\boldsymbol{\eta}_{\text{ADVS}}[k],\bar{\mathbf{u}}[k];\mathbf{w})$. The following section of this work presents the control framework that facilitates the implementation of the physics-aware augmented dynamics visual servoing approach on a multi-rotor.

IV. Control Design

A. Monotonically Weighted NMPC

To effectively control the physics-aware augmented dynamics visual servoing model derived in (19), it is necessary to employ a discrete-time controller that not only guarantees system stability but also ensures adherence to prescribed input and state constraints. The inclusion of an input constraint in the control design helps prevent issues related to actuator saturation, while the state constraints assist in avoiding problems such as target loss, as the term associated with image pixels, denoted as $\mathbf{r}$, is one of the states present in $\boldsymbol{\eta}_{\text{ADVS}}$.

An approach frequently employed is the development of a nonlinear model predictive controller (NMPC) that considers these constraints, ensuring both effective tracking performance and system stability. However, conventional NMPC formulations often require significant computational resources, making them impractical for implementation on systems with limited on-board computation capabilities, such as multi-rotors. The computational burden is further exacerbated when incorporating terminal constraints into the NMPC formulation to guarantee stability.

An effective method to solve this issue is to use a monotonically weighted NMPC in which the cost function is modified such that each term in the cost function is associated with an increasing weight [33]. The computational burden of implementing the monotonically increasingly weighted NMPC is reduced as the formulation eliminates the use of terminal constraints. The cost function for the monotonically increasingly weighted NMPC formulation for the dynamical model (19) is given as:

$$\begin{aligned}J(\boldsymbol{\eta}_{\text{ADVS}}[k], \bar{\mathbf{u}}[k]) = \sum_{i=1}^{T}\left(\frac{i}{T}\right)^m (||\boldsymbol{\eta}^d_{\text{ADVS}}[k+i] \\ - \boldsymbol{\eta}_{\text{ADVS}}[k+i]||^2_Q + ||\bar{\mathbf{u}}[k+i]||^2_R) \end{aligned} \tag{20}$$

where $\bar{\mathbf{u}}[k] = [\bar{u}_{dep}[k] \ \bar{\mathbf{u}}_{pos}[k] \ \bar{\mathbf{u}}_i[k]]$, $m \in \mathbb{N}$ is an integer, $\boldsymbol{\eta}^d_{\text{ADVS}}[k]$ is the desired value of the system states $\boldsymbol{\eta}_{\text{ADVS}}[k]$ at the k^{th} time sample, $||\boldsymbol{\eta}_{\text{ADVS}}||_{\mathbf{Q}} = \sqrt{\boldsymbol{\eta}^T_{\text{ADVS}}\mathbf{Q}\boldsymbol{\eta}_{\text{ADVS}}}$ and $||\bar{\mathbf{u}}||_{\mathbf{R}} = \sqrt{\bar{\mathbf{u}}^T\mathbf{R}\bar{\mathbf{u}}}$ with $\mathbf{Q} \in \mathbb{R}^{(4n+8)\times(4n+8)}$ and $\mathbf{R} \in \mathbb{R}^{6\times 6}$ are positive definite matrices used to compute the weighted norm.

The NMPC formulation can now be derived as:

$$\begin{aligned}&\min_{\bar{\mathbf{u}}[k]} \ J(\boldsymbol{\eta}_{\text{ADVS}}[k], \bar{\mathbf{u}}[k]) \\ &\text{Subject to :} \begin{cases} \boldsymbol{\eta}_{\text{ADVS}}[k+1] = \boldsymbol{\Phi}_{\text{ADVS}}(T,k;\mathbf{w}) \\ \forall \ \boldsymbol{\eta}_{\text{ADVS}}[k] \in \mathcal{X} \\ \forall \ \bar{\mathbf{u}}[k] \in \mathcal{U} \\ \forall \ i = 1,2,\cdots,T \end{cases}\end{aligned} \tag{21}$$

where $\mathcal{X}$ and $\mathcal{U}$ represent the state and the input constraints, respectively. The idea is to evaluate an optimal value of the

control effort $\bar{\mathbf{u}}[k]$ at the k^{th} instance that will minimize the cost function $J(\boldsymbol{\eta}_{\text{ADVS}}[k], \bar{\mathbf{u}}[k])$. Typically, this control effort is denoted as $\bar{\mathbf{u}}^*[k]$ and the optimal cost at the k^{th} instance is represented as $J^*(\boldsymbol{\eta}_{\text{ADVS}}[k], \bar{\mathbf{u}}^*[k])$. the above optimization problem can be solved using nonlinear programming techniques such as interior point-optimization (IP-OPT) which are readily available in open-source solvers [34].

Assumptions: The following assumptions are made:

1) There exists a value of $\bar{\mathbf{u}}[k] \in \mathcal{U}$ for which $\boldsymbol{\Phi}_{\text{ADVS}}(T, k; \mathbf{w}) \in \mathcal{X} \; \forall \; \boldsymbol{\eta}_{\text{ADVS}}[k] \in \mathcal{X}$.
2) $\mathcal{X}$ and $\mathcal{U}$ are closed and bounded sets containing the origin in its interior.
3) The PINN is trained such that $\boldsymbol{\Phi}_{\text{ADVS}}(T, k; \mathbf{w}) = \mathbf{0}$ when $\bar{\mathbf{u}}[k] = \mathbf{0}$ and $\boldsymbol{\eta}_{\text{ADVS}}[k] = \mathbf{0}$ simultaneously.

B. Adaptive Horizon for Monotonically Weighted NMPC

While monotonically weighted NMPC is beneficial in circumventing stringent terminal constraints, it necessitates a sufficiently long prediction horizon to maintain stability when the states are far from the equilibrium point. Moreover, this method relies on prior knowledge of an appropriate prediction horizon for stabilizing the system. In the following formulation, we present an approach that adjusts the prediction horizon based on specific requirements. By reducing the length of the prediction horizon, the computation time for solving the optimization problem can be decreased. To achieve this mechanism consider $J^*(\boldsymbol{\eta}_{\text{ADVS}}[k], \bar{\mathbf{u}}^*[k])$ and $J^*(\boldsymbol{\eta}_{\text{ADVS}}[k-1], \bar{\mathbf{u}}^*[k-1])$ which are the optimal costs obtained at the k^{th} and $(k-1)^{th}$ instances. At the k^{th} instance, consider computing a contraction factor κ_f as:

$$\kappa_f = \frac{J^*(\boldsymbol{\eta}_{\text{ADVS}}[k-1], \bar{\mathbf{u}}^*[k-1]) - J^*(\boldsymbol{\eta}_{\text{ADVS}}[k], \bar{\mathbf{u}}^*[k])}{J^*(\boldsymbol{\eta}_{\text{ADVS}}[k-1], \bar{\mathbf{u}}^*[k-1])} \tag{22}$$

The factor κ_f indicates the extent to which the current cost has contracted/expanded with respect to the previous cost. A positive value of κ_f indicates a contraction in cost where as a negative κ_f represents an expansion in the optimal cost. Typically, the prediction horizon length T is to be designed such that κ_f is bounded within $(\kappa_{min},\ \kappa_{max})$ where the values of κ_{min} and κ_{max} are determined heuristically by simulating the model (19) alongside the proposed NMPC in (21). The adaptive horizon logic is used to determine the value of T at each time sample $k+1$ which is defined as follows:

$$T_{k+1} = \begin{cases} T_k & \text{if } \kappa_{min} < \kappa_f < \kappa_{max} \\ T_k - 1 \text{ if } \kappa_f > \kappa_{max} \;\&\; & T_k > T_{min} \\ T_k + 1 \text{ if } \kappa_f < \kappa_{min} \;\&\; & T_k < T_{max} \end{cases} \tag{23}$$

where T_{k+1} and T_k represent the values of T at the $(k+1)^{th}$ and k^{th} time samples, respectively. T_{min} and T_{max} denote the minimum and maximum acceptable prediction horizon lengths, ensuring feasibility. The stated condition must be satisfied for all time samples where $k \geq 2$.

Remark 2: The horizon adaptation, as outlined in (23), is executed at the conclusion of each time instance k (after solving the NMPC formulation for that particular time instant) to determine the value of T for the subsequent time interval. In summary, the initial value of T at $k = 1$ is arbitrarily assumed and remains unchanged for $k = 2$. This fixed value is utilized to derive the quantities $J^*(\boldsymbol{\eta}_{\text{ADVS}}[1], \bar{\mathbf{u}}[1])$ and $J^*(\boldsymbol{\eta}_{\text{ADVS}}[2], \bar{\mathbf{u}}[2])$ for the respective time instances, which are then employed to calculate the contraction factor at $k = 2$ using (22). Subsequently, this computed value is used to determine the expression for T at $k = 3$ and so forth.

C. Stability Analysis

Theorem: Assuming the physics-aware augmented dynamics visual servoing model for the multi-rotor, as presented in (19), is subject to input constraints $\mathcal{U}$ and state constraints $\mathcal{X}$, the monotonically weighted NMPC formulation outlined in (21), along with the horizon adaptation rule obtained in

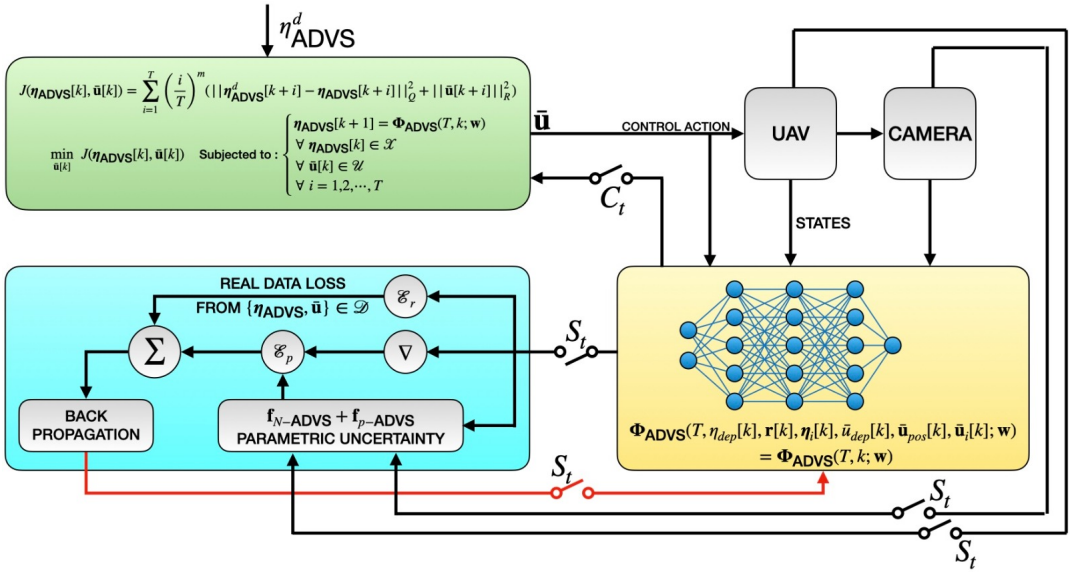


Fig. 2: Overall implementation block diagram.

(23), will asymptotically drive the system states to the origin. This holds true for all values of $\boldsymbol{\eta}_{\text{ADVS}}$ that originate within $\mathcal{X}$, given that Assumptions 1, 2, and 3 are fulfilled.

Proof: To establish the stability of the aforementioned controller, we investigate the convergence of the control effort under three major cases, namely: Increasing horizon ($T_{k+1} = T_k + 1$), Unchanged horizon ($T_{k+1} = T_k$), and Decreasing horizon ($T_{k+1} = T_k - 1$). The proof for the three cases can be found in the supplementary material attached to this paper.

V. Implementation Methodology

A. Physics Loss and Real-time Datasets

As mentioned in the preceding sections of this work the physics loss dataset is sampled from a normal distribution $\mathcal{N}$. For the tests conducted in this work the physics loss dataset $\mathcal{P}$ was sampled from a zero mean normal distribution and a unit constant standard deviation. The real-time dataset is obtained by conducting autonomous landing tasks partly in the ROS based simulator Gazebo and partly in the real-time test scenario which is depicted in the following section of this work. A conventional PID controller is implemented on the nominal dynamics $\mathbf{f}_{\text{N-ADVS}}$ when conducting the landing trials and 50% of the dataset $\mathcal{D}$ is populated using the data obtained from Gazebo and the rest from the real-time test scenarios. The physics engine in the Gazebo test environment helps in injecting the multi-rotor model with a Gaussian noise defined by $\mathcal{N}(0, 1)$ and a polynomial aerodynamic drag force to simulate a real-time test scenario. Furthermore, the platform supports addition of propeller elasticity and asymmetric propeller voltages which were considered when preparing the dataset $\mathcal{D}$. To account for uncertainties and inaccuracies in the camera parameters, trials were conducted with un-calibrated and partially calibrated camera matrices in Gazebo as well as real-time test scenarios. These outputs are also amalgamated into the dataset $\mathcal{D}$. A total of 3000 data points are gathered for $\mathcal{P}$ and 6000 data points are gathered for $\mathcal{D}$ with 3000 points coming from Gazebo and 3000 points from real-time flights.

B. Overall Implementation of PINN and NMPC

The overall system implementation is depicted in Fig. 2. The PINN is trained in an off-line mode with S_t turned ON and C_t OFF. the trained PINN is then deployed to support the NMPC formulation presented earlier for real-time flight by turning C_t ON and S_t OFF. The training of the PINN is based on batch update policy using the memory-efficient quasi-newton L-BFGS optimizer [35].

VI. Experimental Results

A. Hardware Description

The effectiveness of the proposed approach is verified through experiments conducted on a quadrotor system, shown in Fig. 3. This quadrotor is assembled using a DJI F450 frame and incorporates a carbon fibre landing gear to improve shock absorption during landings. The flight controller firmware is run on a Pixhawk 6X flight controller and the application layer runs on a NVIDIA Jetson Nano board. For visual perception, the quadrotor is equipped with a Logitech BRIO 500 camera facing downwards, boasting a 4K ultra-high-definition sensor, ensuring clear and sharp image capture even under low-light conditions. Moreover, to enhance depth estimation, a LiDAR sensor is employed in conjunction with the downward-facing camera, operating within the same reference frame.

Fig. 3: Hardware setup.

Parameters	Value	Parameters	Value
m	1.5 kg	g	9.8 m/sec^2
J_x	0.0150 kg.m^2	J_y	0.00150 kg.m^2
J_z	0.0300 kg.m^2	f	27 mm

TABLE I: Hardware parameters.

Table I provides the essential parameters of the quadrotor and camera setup employed in this work. For the NMPC, a prediction horizon of 1 second and a publishing frequency of 1 kHz are chosen, enabling the controller to be deployed on the actual Pixhawk based hardware. The outer-loop of the Pixhawk hardware operates at 50 Hz, while the inner attitude loop functions at 250 Hz. NMPC implementation on the Pixhawk flight controller is performed using the logic presented in [36]. The number of POIs to be tracked is determined as $n = 4$, leading to the selection of positive definite diagonal matrices $\mathbf{Q} \in \mathbb{R}^{24\times24}$ and $\mathbf{R} \in \mathbb{R}^{6\times6}$ for the cost function and the value of m is set to 3. To evaluate the effectiveness of the proposed PINN framework, it is compared to two state-of-the-art KNODE-MPC [25] techniques.

B. Target Tracking Performance

The proposed approach's efficacy is assessed through a moving target tracking task. For this task, an automatic ground vehicle (AGV) affixed with an ArUco marker is utilized to aid in target tracking. The AGV is programmed to follow a lemniscate trajectory with a radius of 1m and a constant speed of 0.5m/sec. The quadrotor's objective is to track the AGV while maintaining a visual contact with the AGV at an altitude of 4m. The PINN is trained with 2000 samples of physics loss data and 2000 samples of real-time data to complete this task. Fig. 4 represents the 3D tracking trajectory of the quadrotor and the associated motion in the AGV. Similar experimental trails are performed for the analysis presented in the following sections of this work.

C. Effect of Data-skewness on PINN

The quadrotor's tracking performance is significantly affected by the accuracy of the PINN, which approximates the

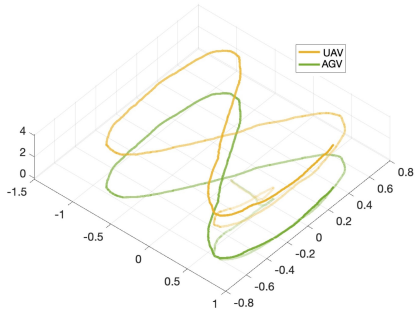


Fig. 4: 3D trajectory of quadrotor (in orange) tracking the AGV (in green) using proposed PINN approach.

augmented dynamical model. Therefore, it is crucial to assess the PINN's performance by adjusting the dataset used during training. A valuable metric for evaluating this characteristic is referred to as data-skewness, denoted by $\epsilon = \frac{N_r}{N_p}$. This ratio represents the proportion of real-time data points used to train the PINN compared to the physics loss data points.

A nil data-skewness corresponds to $\epsilon = 1$, indicating an equal number of real-time data points and physics loss data points used for training. In contrast, a high data-skewness of $\epsilon = \frac{1}{32}$ indicates that the real-time data points are fewer compared to the physics loss data points. Consequently, a high data-skewness may lead to the PINN inaccurately capturing modeling inaccuracies, resulting in larger tracking errors during real-time experiments. To analyze the effect of data-skewness on the proposed PINN, the value of ϵ is varied between 1 to $\frac{1}{32}$ following a geometric progression of common ratio $\frac{1}{2}$. The precision of the PINN for each value of the data-skewness is measured using the percentage tracking error which is evaluated using the percentage Dynamic Time Warping (%DTW) algorithm [37]. These trials are conducted for tracking the previously mentioned lemniscate trajectory and the performance of the proposed PINN approach is evaluated with the KNODE-MPC [25].

Skewness (ϵ)	Proposed	KNODE-MPC [25]
1/32	110.39	121.45
1/16	103.28	106.28
1/8	70.32	80.32
1/4	58.56	66.78
1/2	38.22	43.19
1/1	10.28	21.88

TABLE II: Effect of data-skewness between KNODE-MPC [25] and proposed PINN approach based on %DTW.

Table II displays the %DTW comparison between the proposed approach and KNODE-MPC [25] when tracking a lemniscate with a radius of 1m. The PINN is trained using a total dataset size of $N_p + N_r = 4000$. The values in the table represent the relative %DTW errors, normalized with respect to the nominal value achieved during the tracking task while using a conventional PID controller to control only the nominal dynamics. As discussed earlier, it is evident that the %DTW errors are higher under a high data-skewness condition of $\epsilon = \frac{1}{32}$ compared to the nil data-skewness condition of $\epsilon = 1$. This observation indicates that the PINN struggles to accurately capture the model, underscoring the importance of employing equally balanced datasets for PINN training. Moreover, the proposed approach demonstrates a greater reduction in %DTW errors compared to the KNODE-MPC [25] approach. Specifically, for a data-skewness of $\epsilon = 1$, the proposed approach achieves an error reduction of 53.23%.

Further, extending the analysis discussed earlier, we assess the training time for both the KNODE-MPC [25] and the proposed approach under different levels of data-skewness (ϵ). The training durations for the proposed approach and KNODE-MPC [25] are detailed in Table III and are measured in seconds. It is crucial to highlight that these comparisons are performed using training data gathered for tracking a lemniscate trajectory and are executed on a ground station computer featuring an Intel Iris XE graphics card.

Skewness (ϵ)	Proposed	KNODE-MPC [25]
1/32	268	397
1/16	426	680
1/8	551	1008
1/4	724	1249
1/2	908	1832
1/1	1193	2633

TABLE III: Comparing the effect of data-skewness on training time (sec) between KNODE-MPC [25] and proposed approach.

From the entries in Table III, it can be observed that with an increase in the data-skewness, the training time required for convergence for both approaches increases. For the proposed approach, this increase is about 35.3% with a 100% increase in the data-skewness factor, and about 48.76% with the KNODE-MPC [25]. Additionally, the average training time difference between the two approaches increases from 48% for a data-skewness of $\frac{1}{32}$ to 120.7% for a data-skewness of 1. This is due to the fact that an increase in data-skewness implies that there is an increase in the number of data points depicting modeling inaccuracies obtained from the dataset $\mathcal{D}$, which consequently introduces data points that can only be fit using approximators with physics-based constraints such as the proposed approach.

Trajectory	Circle	Cardioid	Lemniscate
GP MPC [26]	78	83.24	92.79
KNODE MPC [25]	53.63	72.18	80.65
Proposed	45.13	46.87	50.22

TABLE IV: %DTW errors for GP MPC [26], KNODE MPC [25] and proposed PINN approach.

D. Uncertainty in Camera Parameters

Since camera uncertainties are modelled as a part of the physics loss dataset, the PINN is able to maintain an efficient tracking performance alongside the monotonically weighted NMPC approach, provided the data-skewness is maintained close to nil i.e. $\epsilon = 1$. To analyze this phenomena, the tracking task is repeated using a camera with an uncertainty of 70.38% in its intrinsic parameters. Three trajectories are executed using the AGV for this trial, namely: a lemniscate, a circle and a cardioid with a radius of 2m and the quadrotor maintaining its altitude at 4m. The AGV moves at a speed of 0.5m/sec for each

trajectory. The effectiveness of the proposed PINN approach is compared with the existing KNODE-MPC [25] and GP-MPC [26] approaches and the %DTW errors are shown in Table IV.

Table IV presents %DTW values that have been normalized with respect to the tracking performance of a monotonically weighted NMPC applied to the nominal dynamics. Generally, when tracking a lemniscate trajectory compared to circular or cardioid trajectories, the errors are higher. This difference is attributed to the lemniscate's increased complexity, which places higher demands on motor control, particularly necessitating increased yawing to compensate for environmental disturbances. However, it is important to note that the proposed approach achieves a lower %DTW when tracking trajectories compared to existing approaches, despite the presence of uncertainties in the camera parameters.

E. AGV speed and radius variations on tracking performance

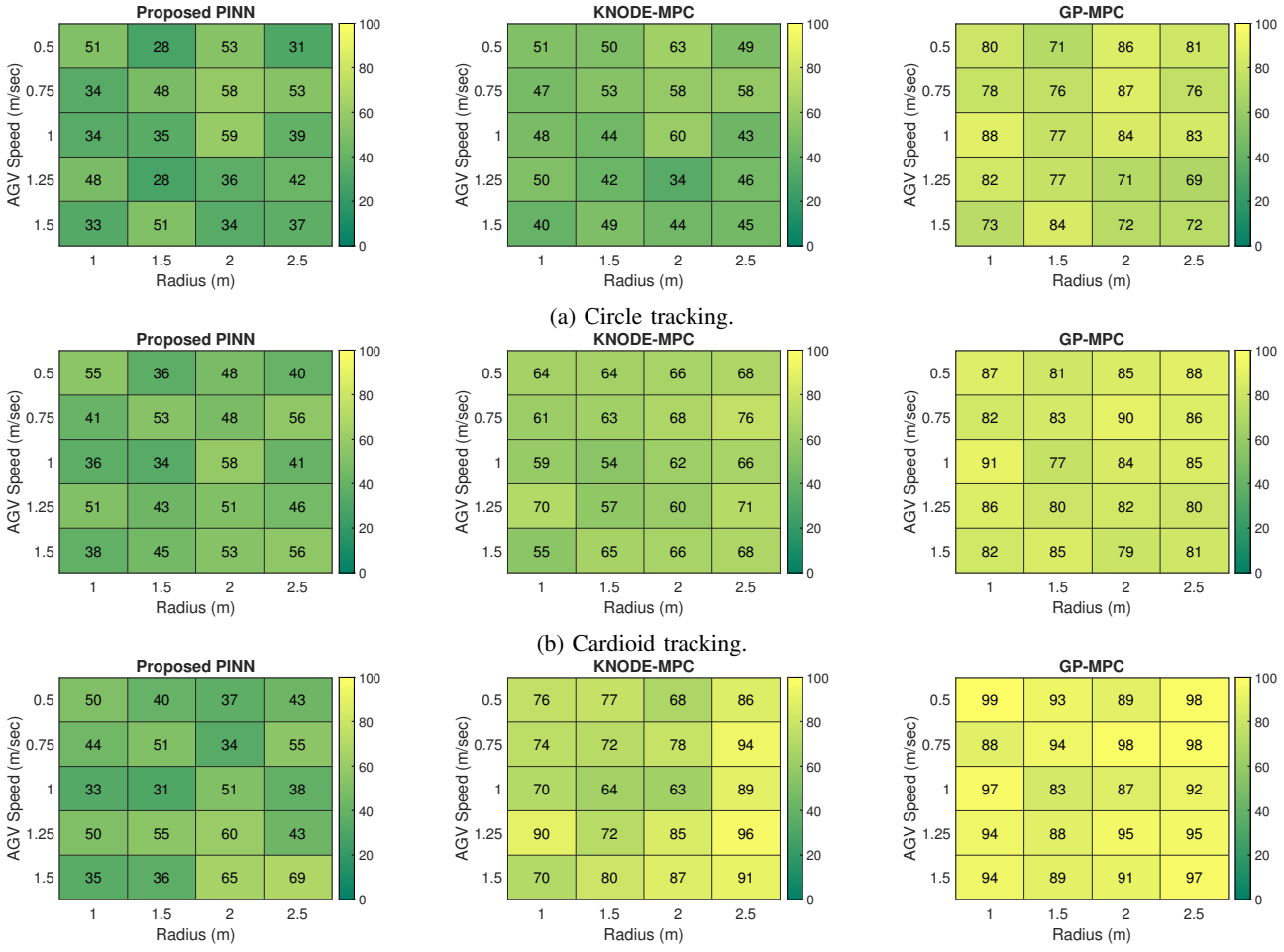
To further assess the effectiveness of the proposed approach, the tracking task is repeated for various AGV speeds and radii, and the resulting %DTW errors normalized to the nominal MPC are illustrated using heatmaps depicted in Fig. 5. Comparing the %DTW errors normalized in the heatmaps, the proposed approach consistently exhibits superior performance compared to the KNODE-MPC [25] and GP-MPC [26]. It is observed that the %DTW values are higher when tracking the lemniscate trajectory in comparison to tracking the cardioid and circle trajectories, for all three cases. Across speeds ranging from 0.5m/sec to 1.5m/sec, the proposed approach outperforms both KNODE-MPC and GP-MPC. The average %DTW error values for all three algorithms and trajectories are summarized in Table V.

Trajectory	Circle	Cardioid	Lemniscate
GP MPC [26]	74.05	83.7	88.65
KNODE MPC [25]	48.7	64.15	73.7
Proposed	41.6	46.45	46

TABLE V: Average %DTW errors for GP MPC [26], KNODE MPC [25] and proposed approach when tracking different trajectories for AGV speed variations between 0.5m/sec to 1.5m/sec and radius variations between 1m to 2.5m.

F. Comparison with data-driven approaches

In this section, a comparison is presented between the proposed approach and an existing data-driven method. Specif-



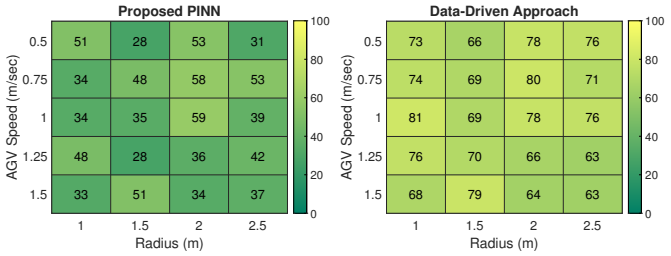
(a) Circle tracking.

(b) Cardioid tracking.

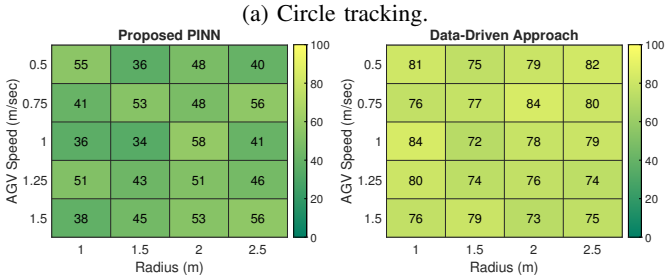
(c) Lemniscate tracking.

Fig. 5: Heatmap of %DTW between KNODE-MPC [25], GP-MPC [26] and Proposed PINN approach when tracking a circle, cardioid and lemniscate trajectory at varying speeds and radius.

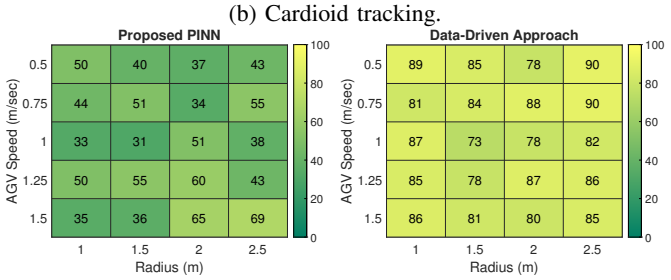
ically, the study evaluates the data-driven approach outlined in [24] for this comparative analysis. The results are presented in two sets, illustrating the effectiveness of the proposed approach in comparison to [24]. The first set includes heatmaps depicting the %DTW of both approaches when tracking a circle, lemniscate, and a cardioid. It is important to note that the number of data points considered for training in both approaches remains constant, i.e., $N_p + N_r = 4000$. Furthermore, the results are presented with a data-skewness of 1 for both networks, as it has been observed that the PINN's lowest accuracy occurs when $\epsilon = 1$.



(a) Circle tracking.



(b) Cardioid tracking.



(c) Lemniscate tracking.

Fig. 6: Heatmap of %DTW between a data-driven approach [24] and the Proposed approach when tracking a circle, lemniscate and cardioid trajectory at varying speeds and radius.

In Fig. 6, the %DTW results are presented for both the proposed approach and the data-driven method outlined in [24]. The training involved 4000 data points, with 2000 sourced from the physics loss dataset and the remaining 2000 from the real-time dataset. Similar to previous procedures, the %DTW errors in the heatmap are normalized to the nominal MPC in both cases, assuming an uncertainty of 70.98% in the intrinsic parameters of the camera during the tracking of the moving AGV. It is evident from the results that the proposed approach consistently outperforms the data-driven approach [24] in tracking the three trajectories, as indicated by the lower %DTW values. This superiority can be attributed to the enhanced function approximation capability of PINNs compared to comprehensive data-driven approaches when dealing with a limited number of data points.

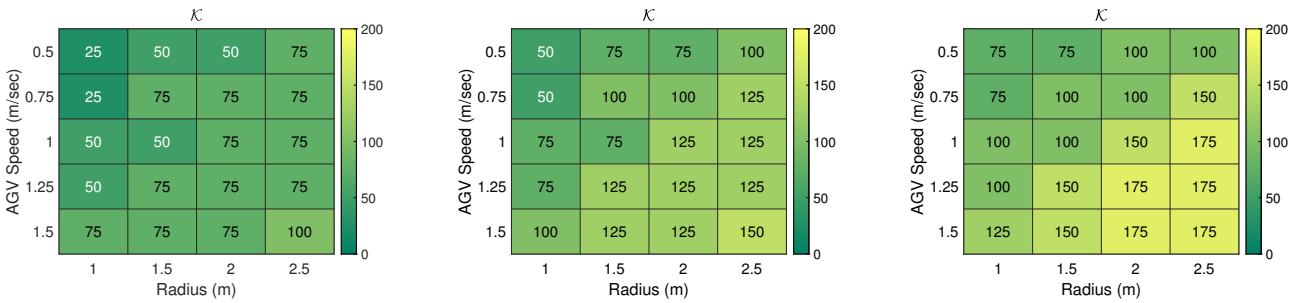
This does not necessarily imply that data-driven approaches lack proficiency in function approximation. In situations where the number of training data points is increased, the data-driven approach [24] possesses the capability to achieve performance comparable to that of the proposed approach. To illustrate this point, the second set of results are presented, wherein the dataset size used to train the data-driven approach is increased to obtain an equivalent %DTW as that of the proposed approach. The paper introduces this result to showcase the percentage increase in dataset size required for a complete data-driven approach to be equally accurate as the proposed approach in tracking targets. It is important to note that the data-skewness is maintained at 1 for this analysis, and the accuracy of the proposed approach is assumed to remain unchanged, serving as a benchmark for comparison with [24].

Fig. 7 presents heatmaps illustrating the fraction $\mathcal{K}$, indicating the percentage increase in data points needed to ensure that the data-driven approach outlined in [24] can achieve a %DTW comparable to the proposed method. This comparison extends beyond trajectory contour analysis to encompass the speed of the AGV. Notably, tracking a circular trajectory demands a significantly lower data increment factor than a lemniscate trajectory. On average, attaining a %DTW equivalent to the proposed approach requires an increment of 65% for a circular trajectory, 126.25% for a lemniscate trajectory and 101.25% for a cardioid trajectory. In each scenario, it is evident that the factor $\mathcal{K} > 50\%$, indicating that the data-driven approach necessitates a larger dataset to approximate the system dynamics compared to the proposed approach. This not only extends the training time for the network prior to deployment but also increases the storage space needed for enhancing the approximated system dynamics.

This phenomenon can be explained by the inherent nature of the PINN, which exhibit a propensity to fit data onto a nominal model. This characteristic contributes to a more effective approximation of system dynamics compared to complete data-driven approaches. The latter often involve fitting given datasets onto a set of arbitrary nonlinear equations that closely mimic the system dynamics only within the range where the training data was collected. In essence, data-driven approaches prove to be less adept at extrapolations, particularly in scenarios characterized by uncertain parameters and modeling inaccuracies.

G. Feasibility of Real-time Implementation

This subsection aims to validate the feasibility of integrating the proposed physics-aware model with the monotonically weighted NMPC strategy for real-time experiments. Specifically, the proposed approach is compared to existing state-of-the-art MPC strategies, namely the Tube MPC [27] and the Adaptive MPC [28], both proven to be effective for real-time implementation in problems similar to the one considered in this work. The experiments are conducted on a NVIDIA Jetson Nano companion computer, equipped with a 128-core Maxwell GPU, making it well-suited for real-time assessments. Table VI illustrates the latency comparison between the



(a) Circle tracking. (b) Cardioid tracking. (c) Lemniscate tracking.

Fig. 7: Heatmap depicting the percentage $\mathcal{K}$ increase in data necessary for the data-driven approach in [24] to achieve performance parity with the proposed approach.

proposed approach and the approaches presented in [27] and [28], providing measurements in seconds for a single forward-pass calculation. It is important to note that these comparisons are performed while tracking a lemniscate trajectory of 2.5m at an AGV speed of 1.5m/sec. Additionally, the Tube MPC and the Adaptive MPC are tuned, and constraints are adjusted to achieve nearly equivalent %DTW as that of the proposed approach for the similar tracking scenario.

Examining the entries in Table VI, it is evident that the time required for completing a single forward-pass is nearly 10 times lower when utilizing the proposed algorithm compared to the algorithms presented in [27] and [28]. This can be attributed to the following reasons:

- The Tube MPC [27] is a conventional robust Model Predictive Control method that depends on a nominal dynamics model and imposes conservative bounds on disturbances affecting the state variables. While this strategy ensures stable behavior even in worst-case scenarios, it achieves this by introducing additional constraints to the optimization problem. As a result of this approach, the computation time required for a single forward-pass is significantly higher compared to the proposed approach.
- The Adaptive MPC [28] approach employs control parameter adaptation to emulate the reference model, thereby ensuring robustness against parametric uncertainties. However, these methods are limited in their ability to address modeling inaccuracies. To address this limitation, modeling inaccuracies must be incorporated into the MPC formulation as a disturbance constraint. This addition, in turn, increases the time required for a single forward-pass to achieve a similar %DTW as the proposed approach.

Algorithm	Mean	Median
Tube MPC [27]	9.53×10^{-3}	9.19×10^{-3}
Adaptive MPC [28]	3.34×10^{-3}	3.05×10^{-3}
Proposed	6.37×10^{-4}	3.15×10^{-4}

TABLE VI: Latency comparison between the proposed approach, [27] and [28] method for a single forward-pass.

Despite the mentioned limitations, both the Tube MPC and the Adaptive MPC have found widespread application in real-time implementations in the past. In contrast, the proposed approach offers two significant advantages over these methods. Firstly, the monotonically weighted NMPC with its adaptive horizon helps reduce the estimation horizon for each pass, thereby decreasing computation time. Secondly, the PINN in the proposed approach effectively approximates both the modeling inaccuracies and the parametric uncertainties of the system, eliminating the need to add these as constraints to maintain high tracking accuracy. This further reduces the computational burden for real-time implementation. These advantages position the proposed approach as a suitable alternative to the existing state-of-the-art methods in terms of real-time implementability.

Remark 3: The rationale behind selecting the proposed approach as the benchmark %DTW for comparing its performance with the existing Tube MPC [27] and Adaptive MPC [28] approaches stems from the observation that the %DTW for tracking a lemniscate was found to be higher in the case of [27] and [28] with their default gain and constraint settings. It is important to note that these specific details are not included in the results section due to space constraints.

H. Effect of Gazebo and Real-time experimental data on tracking performance

In the context of the previously presented findings, a data-skewness of $\epsilon = 1$ was employed, involving the use of 2000 data points from $\mathcal{P}$ and an additional 2000 from $\mathcal{D}$ for training the PINN. The standard configuration for $\mathcal{D}$ consisted of 10% of data acquired through ROS Gazebo, while the remaining 90% was obtained from real-time hardware experiments. To illustrate the impact of increasing the influence of Gazebo on these outcomes, the %DTW is presented in Table VII for the proposed approach when tracking a lemniscate trajectory of radius 2.5m with varying proportions of Gazebo and real-time experimental data in $\mathcal{D}$. Note that that the %DTW values displayed in Table VII have been normalized to the nominal MPC, as previously specified. The comparison parameter employed is the ratio $N_{r_{\text{Gazebo}}}/N_{r_{\text{Exp}}}$, where $N_{r_{\text{Gazebo}}}$ represents the number of data points derived from ROS Gazebo, and $N_{r_{\text{Exp}}}$ denotes the data points acquired from real-time experiments.

The data entries in Table VII clearly indicate that an increase in Gazebo data within $\mathcal{D}$, relative to real-time experimental

$N_{r_{\text{Gzebo}}}/N_{r_{\text{Exp}}}$	0.5	0.75	1	1.25	1.5
10/90	43	55	38	43	69
30/70	47	60	43	48	73
50/50	50	64	47	52	76
70/30	55	71	54	59	83
90/10	61	77	68	63	85

TABLE VII: %DTW for tracking a lemniscate of radius 2.5m with varying values of the ratio $N_{r_{\text{Gzebo}}}/N_{r_{\text{Exp}}}$ and AGV speed used for training the PINN.

data, leads to an augmentation in %DTW for the lemniscate tracking scenario. Nevertheless, it is important to note that the average %DTW values are approximately 51 for a speed of 0.5m/sec, 65 for a speed of 0.75m/sec, 50 for a speed of 1m/sec, 53 for a speed of 1.25m/sec, and 77 for a speed of 1.5m/sec. These values are not significantly higher than the baseline scenario with $N_{r_{\text{Gazebo}}}/N_{r_{\text{Exp}}} = 10/90$. This can be attributed to the fact that the PINN incorporates physics-based constraints to model the system dynamics, making it robust against uncertainties encountered during real-time flight tests. Despite a reduced volume of real-time experimental data in $\mathcal{D}$, the outcomes of the tracking experiments are reasonably satisfactory. This suggests that PINNs possess the capability to model system dynamics with reasonable precision, even with a moderate amount of available real-time experimental data. Consequently, the trained network can be effectively utilized to implement control strategies in real-time with high precision.

VII. Conclusion

This study introduced a novel visual servoing approach that merges the capabilities of PINN for estimating system uncertainties and inaccuracies with a dynamics-centered visual servoing technique tailored for multi-rotors. This integrated method effectively combines these strategies, eliminating the requirement for inverse Jacobian calculations to determine multi-rotor motion. Instead, it directly correlates pixel variations with the multi-rotor's torque and thrust inputs. Furthermore, the method enhances robustness by utilizing PINN to model and address uncertainties in camera and multi-rotor parameters, as well as the inherent modeling inaccuracies in the dynamics-centered visual servoing technique. To ensure real-time feasibility, the acquired augmented dynamical model is integrated with a monotonically weighted NMPC framework, facilitating trajectory tracking tasks. Subsequently, the results are compared with the state-of-the-art literature for validation.

The results demonstrate that, for a data-skewness of 1, the proposed approach achieves a 120% faster training speed compared to [25]. Furthermore, it exhibits an average %DTW of 46.45%, 41.6%, and 46% when tracking cardioid, circle, and lemniscate trajectories, respectively. These values are notably lower than those achieved by the approaches presented in [25] and [26]. Additionally, when compared with a purely data-driven approach from [24], the proposed approach requires, on average, a 97% increase in the number of data points to achieve an equivalent %DTW for trajectory tracking. This outcome is attributed to the PINN's ability to incorporate nominal dynamics into its approximation, enhancing accuracy and reducing the amount of labeled data required. In terms of real-time implementability, the proposed approach achieves nearly 10 times lower latency than current state-of-the-art techniques, establishing it as a viable and elegant solution for real-time experiments.

Acknowledgments

This research is supported by A*STAR under its RIE2025 MANUFACTURING, TRADE AND CONNECTIVITY (MTC) INDUSTRY ALIGNMENT FUND -PRE POSITIONING (IAF-PP), with Award No. M23L5a0002. The authors express special gratitude to Dr. Tanmoy Dam and Dr. T. Thanaraj for their invaluable assistance in refining the article. They also extend their appreciation to the three reviewers, the associate editor and the editor-in-chief of the IEEE Transactions on Cybernetics for their constructive feedback, which has significantly contributed to improving the paper's readability and technical content.

References

[1] X. Liang, H. Wang, Y.-H. Liu, B. You, Z. Liu, Z. Jing, and W. Chen, "Fully uncalibrated image-based visual servoing of 2dofs planar manipulators with a fixed camera," *IEEE Transactions on Cybernetics*, vol. 52, no. 10, pp. 10 895–10 908, 2022.

[2] W. Lin, C. Liu, H. Guo, and H. Gao, "Hybrid visual-ranging servoing for positioning based on image and measurement features," *IEEE Transactions on Cybernetics*, 2022.

[3] K. Singh, M. Mehndiratta, and M. Feroskhan, "Quadplus: Design, modeling, and receding-horizon-based control of a hyperdynamic quadrotor," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, no. 3, pp. 1766–1779, 2021.

[4] K. Jo and D. Chwa, "Robust hybrid visual servoing of omnidirectional mobile manipulator with kinematic uncertainties using a single camera," *IEEE Transactions on Cybernetics*, 2023.

[5] G. Fu, L. Fang, L. Liu, X. Zhu, and Y. Wang, "An image-based visual servoing control method for uavs based on fuzzy logic," *Advances in Mechanical Engineering*, vol. 15, no. 4, p. 16878132231167238, 2023.

[6] G. Wang, J. Qin, Q. Liu, Q. Ma, and C. Zhang, "Image-based visual servoing of quadrotors to arbitrary flight targets," *IEEE Robotics and Automation Letters*, vol. 8, no. 4, pp. 2022–2029, 2023.

[7] A. K. Kamath, V. K. Tripathi, S. C. Yogi, and L. Behera, "Vision-based fast-terminal sliding mode super twisting controller for autonomous landing of a quadrotor on a static platform," in *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 2019, pp. 1–6.

[8] X. Sun, W. Zhu, H. Du, W. Chen, and C.-C. Chen, "Image-based fast finite-time target tracking control of onboard inertially stabilized camera platform," *IEEE Transactions on Industrial Electronics*, 2024.

[9] A. K. Kamath, T. Dam, H. L. Maurya, P. Singh, R. R. Nair, and S. Nahavandi, "Modelling and sliding mode control of a stereo vision augmented 6 dof quadrotor system," in *2023 21st International Conference on Advanced Robotics (ICAR)*. IEEE, 2023, pp. 425–430.

[10] P. Roque, E. Bin, P. Miraldo, and D. V. Dimarogonas, "Fast model predictive image-based visual servoing for quadrotors," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 7566–7572.

[11] A. Miranda-Moya, H. Castañeda, J. Gordillo, and H. Wang, "Ibvs based on adaptive sliding mode control for a quadrotor target tracking under perturbations," *Mechatronics*, vol. 88, p. 102909, 2022.

[12] B. Li, X. Zhang, Y. Fang, and W. Shi, "Visual servoing of wheeled mobile robots without desired images," *IEEE Transactions on Cybernetics*, vol. 49, no. 8, pp. 2835–2844, 2018.

[13] Z. Cao, X. Chen, Y. Yu, J. Yu, X. Liu, C. Zhou, and M. Tan, "Image dynamics-based visual servoing for quadrotors tracking a target with a nonlinear trajectory observer," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 1, pp. 376–384, 2017.

[14] Y. Zhang and S. Li, "A neural controller for image-based visual servoing of manipulators with physical constraints," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 11, pp. 5419–5429, 2018.

[15] I. S. Mohamed, G. Allibert, and P. Martinet, "Sampling-based mpc for constrained vision based control," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 3753–3758.
[16] N. Lai, Y. Chen, J. Liang, B. He, H. Zhong, H. Zhang, and Y. Wang, "Image dynamics-based visual servo control for unmanned aerial manipulatorl with a virtual camera," *IEEE/ASME Transactions on Mechatronics*, vol. 27, no. 6, pp. 5264–5274, 2022.
[17] H. Wang, Y.-H. Liu, and D. Zhou, "Adaptive visual servoing using point and line features with an uncalibrated eye-in-hand camera," *IEEE Transactions on Robotics*, vol. 24, no. 4, pp. 843–857, 2008.
[18] X. Zhang, Y. Fang, B. Li, and J. Wang, "Visual servoing of nonholonomic mobile robots with uncalibrated camera-to-robot parameters," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 1, pp. 390–400, 2016.
[19] S. Lee and D. Chwa, "Dynamic image-based visual servoing of monocular camera mounted omnidirectional mobile robots considering actuators and target motion via fuzzy integral sliding mode control," *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 7, pp. 2068–2076, 2020.
[20] K. Zhang, Y. Shi, and H. Sheng, "Robust nonlinear model predictive control based visual servoing of quadrotor uavs," *IEEE/ASME Transactions on Mechatronics*, vol. 26, no. 2, pp. 700–708, 2021.
[21] M. Mohammad Hossein Fallah and F. Janabi-Sharifi, "Conjugated visual predictive control for constrained visual servoing," *Journal of Intelligent & Robotic Systems*, vol. 101, pp. 1–21, 2021.
[22] H. Sheng, E. Shi, and K. Zhang, "Image-based visual servoing of a quadrotor with improved visibility using model predictive control," in *2019 IEEE 28th international symposium on industrial electronics (ISIE)*. IEEE, 2019, pp. 551–556.
[23] D. Nan, J. Li, Y. Weng, L. Lian, C. Yu, and S. Li, "Data-driven adaptive pid control of unknown quadrotor uavs," in *2020 IEEE 9th Data Driven Control and Learning Systems Conference (DDCLS)*. IEEE, 2020, pp. 953–958.
[24] G. Torrente, E. Kaufmann, P. Föhn, and D. Scaramuzza, "Data-driven mpc for quadrotors," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3769–3776, 2021.
[25] K. Y. Chee, T. Z. Jiahao, and M. A. Hsieh, "Knode-mpc: A knowledge-based data-driven predictive control framework for aerial robots," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2819–2826, 2022.
[26] Y. Zheng, T. Zhang, S. Li, G. Zhang, and Y. Wang, "Gp-based mpc with updating tube for safety control of unknown system," *Digital Chemical Engineering*, vol. 4, p. 100041, 2022.
[27] X. Ping, J. Yao, B. Ding, and Z. Li, "Tube-based output feedback robust mpc for lpv systems with scaled terminal constraint sets," *IEEE Transactions on Cybernetics*, vol. 52, no. 8, p. 7563–7576, Aug. 2022. [Online]. Available: http://dx.doi.org/10.1109/TCYB.2020.3041334
[28] M. Yuan, Y. Wang, L. Li, T. Chai, and W. T. Ang, "Safety-based speed control of a wheelchair using robust adaptive model predictive control," *IEEE Transactions on Cybernetics*, p. 1–11, 2023. [Online]. Available: http://dx.doi.org/10.1109/TCYB.2023.3309369
[29] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational physics*, vol. 378, pp. 686–707, 2019.
[30] A. K. Kamath, S. C. Yogi, L. Behera, and S. Nahavandi, "Vision augmented 3 dof quadrotor control using a non-singular fast-terminal sliding mode modified super-twisting controller," in *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2021, pp. 2030–2036.
[31] L. Chen, J. Xiao, R. C. H. Lin, and M. Feroskhan, "Angle-constrained formation maneuvering of unmanned aerial vehicles," *IEEE Transactions on Control Systems Technology*, 2023.
[32] S. Seshasayanan, S. De, and S. R. Sahoo, "Robust attitude control with fixed exponential rate of convergence and consideration of motor dynamics for tilt quadrotor using quaternions," *IEEE Transactions on Automation Science and Engineering*, 2024.
[33] K. Chacko, J. Sivaramakrishnan, and I. N. Kar, "Computationally efficient nonlinear mpc for discrete system with disturbances," *International Journal of Control, Automation and Systems*, vol. 20, no. 6, pp. 1951–1960, 2022.
[34] P. Zhou, X. Sun, and T. Chai, "Enhanced nmpc for stochastic dynamic systems driven by control error compensation with entropy optimization," *IEEE Transactions on Control Systems Technology*, 2023.
[35] J. Rafati and R. F. Marica, "Quasi-newton optimization methods for deep learning applications," *Deep Learning Applications*, pp. 9–38, 2020.
[36] B. B. Kocer, M. E. Tiryaki, M. Pratama, T. Tjahjowidodo, and G. G. L. Seet, "Aerial robot control in close proximity to ceiling: A force estimation-based nonlinear mpc," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 2813–2819.
[37] R. Tavenard, J. Faouzi, G. Vandewiele, F. Divo, G. Androz, C. Holtz, M. Payne, R. Yurchak, M. Rußwurm, K. Kolar *et al.*, "Tslearn, a machine learning toolkit for time series data," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 4686–4691, 2020.

Archit Krishna Kamath (Member, IEEE) received his Ph.D. and Master's degrees specializing in Control and Automation from the Indian Institute of Technology Kanpur (IIT Kanpur) in 2023 and 2018, respectively and a Bachelor's degree in Electrical and Electronics Engineering from M.S. Ramaiah Institute of Technology, Bengaluru, Karnataka, India, in 2015. He presently serves as a Research Fellow at the eVTOL Research and Innovation Center within the School of Mechanical and Aerospace Engineering at NTU Singapore and research pursuits encompass various areas, including modeling and sliding mode control of both conventional and over-actuated quadrotors, utilization of physics-informed neural networks and graph neural networks in aerial robot control, and decentralized control of drone swarms. He is a member of IEEE and a TC member for the Robotics and Intelligent Sensing Technical Committee on the IEEE SMC Society.

Sreenatha G. Anavatti recieved his Bachelor's degree in Mechanical Engineering from Mysore University, India, in 1984, followed by a Ph.D. in Aerospace Engineering from the Indian Institute of Science, Bangalore, in 1990. He began his academic career as an Assistant Professor at the Indian Institute of Bombay, Mumbai, from 1991 to 1997, progressing to Associate Professor for a year starting in 1997. In 1998, he joined the School of Engineering and Information Technology at the University of New South Wales, ADFA Campus, Canberra, Australia, where he currently holds the position of Senior Lecturer, having been promoted from Lecturer in 2001. Throughout his career, Prof. Anavatti has taught various courses such as Flight Mechanics, Flight Control Systems, Missile Guidance, Spacecraft Dynamics, Orbital Mechanics, Classical and Modern Control Theory. His research interests span control systems, flight dynamics, robotics, aero-elasticity, artificial neural networks, fuzzy systems, and unmanned systems. As of 2024, he has co-authored over 330 conference papers, journal articles, and book chapters. Additionally, he authored a book titled "Machine Learning Approaches and Applications in Applied Intelligence for Healthcare Data Analytics," published by CRC Press, Taylor & Francis Group.

Mir Feroskhan (Member, IEEE) received his Ph.D. degree in Aerospace Engineering in 2016 Florida Institute of Technology, USA and a Bachelor's degree with First Class Honours in Aerospace Engineering in 2011 from Nanyang Technological University (NTU), Singapore. Currently, he holds the position of Assistant Professor at NTU Singapore's School of Mechanical and Aerospace Engineering and leads the Intelligent Cybernetics Group. Within this role, he conducts research in diverse areas such as modeling and controlling avian-inspired drones, applying artificial intelligence to aerospace engineering, exploring multi-agent systems, addressing urban air mobility challenges, and advancing space situational awareness. Prof. Feroskhan has received notable accolades, including the A*STAR's Young Individual Research Grant in 2022 for pioneering AI-driven solutions for UAV applications and the Dr. S K Leung Excellence Award in 2023 in recognition of his exemplary teaching activities. He is an active member of IEEE and holds a position as a Technical Committee member for the Robotics and Intelligent Sensing Technical Committee within the IEEE SMC Society. Additionally, he currently serves on the editorial board for Nature's Scientific Reports.