Guidance Mechanism for Flexible Wing Aircraft Using Measurement-Interfaced Machine Learning Platform

Mohammed Abouheaf, Nathaniel Mailhot, Wail Gueaieb, and Davide Spinello

***Abstract*— The autonomous operation of flexible-wing aircraft is technically challenging and has never been presented within literature. The lack of an exact modeling framework is due to the complex nonlinear aerodynamic relationships governed by the deformations in the flexible-wing shape, which in turn complicates the controls and instrumentation setup of the navigation system. This urged for innovative approaches to interface affordable instrumentation platforms to autonomously control this type of aircraft. This work leverages ideas from instrumentation and measurements, machine learning, and optimization fields in order to develop an autonomous navigation system for a flexible-wing aircraft. A novel machine learning process based on a guiding search mechanism is developed to interface real-time measurements of wing-orientation dynamics into control decisions. This process is realized using an online value iteration algorithm that decides on two improved and interacting model-free control strategies in real-time. The first strategy is concerned with achieving the tracking objectives while the second supports the stability of the system. A neural network platform that employs adaptive critics is utilized to approximate the control strategies while approximating the assessments of their values. An experimental actuation system is utilized to test the validity of the proposed platform. The experimental results are shown to be aligned with the stability features of the proposed model-free adaptive learning approach.**

I. Introduction

Flexible-wing aircraft have been capturing increasing interests due to their relatively simple mechanical structures, flexible operation, low fabrication costs, and high payload-to-mass ratio [1]–[3]. On one side, there are no experimentally-validated dynamical models for such systems since they are characterized by highly nonlinear dynamics. Therefore, designing algorithms for the autonomous flight control of this type of aircraft is a complex process. To this end, relying on conventional methods which are based on some sort of classical mathematical models, however approximate they may be, is not an option. On another side, analytical solutions for the optimal tracking control problems often necessitate solving offline coupled differential equations simultaneously, where a subset of these equations is solved backward in-time [4]. Additionally, the complexity of the tracking control laws, resulting from adaptive systems, are not realizable or hard-to-implement using digital processing units.

This work was partially supported by Ontario Centers of Excellence (OCE). Grant number 27404.

Mohammed Abouheaf and Wail Gueaieb are with the School of Electrical Engineering & Computer Science, University of Ottawa, Ottawa, Canada. Nathaniel Mailhot and Davide Spinello are with the Department of Mechanical Engineering, University of Ottawa, Ottawa, Canada. E-mail: {mabouhea,nmailhot,wgueaieb,dspinell}@uottawa.ca

In this work, a coupled instrumentation-machine learning framework is proposed to solve the guidance control problem of a class of nonlinear dynamical systems with unknown dynamics, typical for flexible-wing aircraft. This process tackles difficulties associated with developing control solutions for partially- or fully model-based approaches in uncertain dynamical environments. Further, it solves the tracking control problem in real-time with no offline or backward solution for the underlying differential equations. Simply, it provides a simplified method to compute the tracking control laws which may be both complex and hard-to-realize in digital simulation environments. The proposed scheme was tested in real-time aboard of a Raspberry Pi equipped with an Inertial Measurement Unit (IMU) consisting of accelerometers, gyroscopes and magnetometers, in order to provide accurate orientation and motion sensing measurements (i.e., for the wing of the aircraft). A reinforcement learning process based on adaptive guided search mechanism is employed to steer the motion dynamics of a mock-up emulating the aircraft's wing movement relative to the fuselage. It employs an online value iteration process that produces real-time model-free tracking control strategies.

Flexible-wing aircraft are composed of two interacting structures, namely wing and fuselage, which are pinned at a point where they can rotate on pitch-roll axes relative to one another [5]–[8]. The control of this type of aircraft is challenging due to the flexibility of the wing which induces continuous aerodynamic variations and makes it difficult to model the vehicle's dynamics [1], [9]. Partial theoretical models have been presented for this type of aircraft in [8], [10], [11]. The approximate aerodynamic models varied in complexity and approach. Some researchers opted to decouple the dynamics into longitudinal and lateral motion frames to simplify the problem [12]. A fully decoupled aerodynamic model is developed in [13]. In another context, a fixed-wing approach that considers aerodynamic derivatives is employed in [14], [15]. Additionally, a system of nonlinear dynamical equations based on a nine-degree-of-freedom dynamic model is derived in [1], [9]. The navigation process of this aircraft is based on a weight-shift mechanism where the relative variations in the Center-of-Gravity (CoG) locations of the wing-fuselage systems, under kinematic and dynamic constraints at the interconnection points, steer the aircraft toward the desired orientations [16]–[18]. It is shown that when a weight-shift mechanism is applied, the longitudinal stability is magnified [15], [16]. Also, the stability margins corresponding to the lateral motion are shown to be larger than those of fixed-wing aircraft [8].

This is the accepted version of the paper. Published paper DOI: 10.1109/TIM.2020.2985553.

The tracking, navigation, and motion-guided systems employ instrumentation configurations along with theoretical advances for the underlying physical mechanisms. A method based on radio frequency identification technology is proposed for navigating mobile robots in [19]. An optoacoustic positioning scheme that utilizes inertial and distance measurements is developed for automated monitoring of complex manual assembly operations in [20]. It employs a particle filter that fuses inertial navigation measurements with unilateral distances measurements to space-fixed receiving devices. An artificial neural network approach is proposed to fuse differential and uncompensated measurements from global position and inertial-navigation devices respectively in [21]. This approach avoided using Kalman filtering in order to integrate the inertial navigation and global position systems. Adaptive robust tracking control approach that uses a continuous model based estimator is employed to control a flexible air-breathing hypersonic vehicle in [22]. It employs a type-2 Fuzzy structure to approximate the unknown model dynamics and the stability features are validated using a Lyapunov theorem. A comprehensive test-bed that utilizes multi-camera for operations which involve control and 3D tracking tasks of unmanned aerial vehicles is developed in [23]. The embedded navigation approach relies on a Proportional-Derivative (PD) control structure that receives navigation information form multi-camera system. The test-bed allowed real-time computations at $100\,\mathrm{Hz}$ using cameras with field programmable gate array (FPGA) modules where the embedded software is able to perform motion control and image processing. A vision-based tracking approach that uses particle swarm optimization and fuzzy logic scheme is developed to navigate an autonomous mobile robot in [24]. The fuzzy tracking system is designed using a Lyapunov framework and it benefits global search capabilities of the particle swarm optimization technique. An approach based on interval analysis is developed to solve the localization problem of a mobile robot using ultrasonic sensors in [25]. This approach manages the issues arising from data-associate step noticed within the classical localization problems using Kalman filtering.

Dynamic Programming (DP) solution techniques are employed to solve different control problems [4], [26], [27]. However, these frameworks degrade due to the curse of dimensionality associated with the state-action domains [26], [27]. Approximate Dynamic Programming (ADP) relaxed the manner at which dynamic programming problems are solved using heuristic platforms [26], [28]. These solution forms are meant to provide computational platforms to solve the control problems using temporal difference structures [29]–[31]. The control problems are solved by optimizing the performance of the underlying dynamical systems using objective cost functions. Hence, the solutions for the underlying Bellman optimality or Hamilton-Jacobi-Bellman equations lead to solutions for the optimal control problems [4], [32], [33]. These optimal forms vary in structure, as they depend either on Bellman or Hamiltonian structures, allowing different temporal difference solution forms [34], [35]. These solutions become more complicated in case of coupled hierarchical control systems or multi-agent structures [30], [36]. This is due to the existence of coupled temporal difference formulations. The ADP problems are solved using a dynamic learning environment known as Reinforcement Learning (RL) [26], [37]–[39]. In this process, the strategies taken by the agent are either rewarded or penalized based on their value-assessments using a utility or an objective cost function. The policy-value assessment mechanism is repeated until the best strategy is found.

Reinforcement learning solutions are implemented using two-step mechanisms known as value iteration and policy iteration [26], [37]. The first step solves the temporal difference equation, while the second approximates the optimal control policy [34], [35]. In value iteration, the solving value function is evaluated and then the best strategy is extracted. While in policy iteration, a strategy is evaluated in the first step then a policy improvement step is introduced [26]. The adaptive learning solutions are developed for multi-agent systems where the communications between the nodes are accomplished using graph structures in [30], [35], [40], [41]. These solution forms were able to solve coupled temporal difference equations in real-time using only neighborhood information. Adaptive critics are used as neural network approximation tools to implement these solutions for single- and multi-agent control problems [37], [38], [42], [43]. Each adaptive critics scheme involves two approximating neural networks; the actor network approximates the optimal control strategy while the critic network approximates the solving value function. A cell-mapping approach based on a reinforcement learning technique is developed for robot motion planning in [44]. The learning mechanism does not employ a dynamical model for the robot as it builds experience-knowledge about the dynamical environment and robots dynamics. It employs a transformation based on cell-to-cell transitions in order to reduce time used to build experience about the dynamical environment.

The proposed work advances initial research investigations aimed to design intelligent flight controllers for flexible-wing aircraft using a linear actuation mechanism and machine learning process [45]. However, this development relied on a geometric model of the aircraft and the control strategies are mapped to actuation lengths. Herein, an experimental mock-up system that is composed of servo actuation winch motors acting on the mast and wing of a flexible-wing aircraft are employed to achieve autonomous maneuvers. In [45], the desired strategies are only evaluated in terms of the tracking error signals, while the proposed mechanism finds control decisions based on tracking error signals, their dynamics, and the dynamical measures of the key orientation parameters to support the overall stability of the aircraft. Finally in contrast to [45], this work provides a flexible and innovative model-free guided search learning process based on a real-time value iteration scheme. The machine learning search method opens the door to reflect the designer considerations, regarding data driven structures, in the architecture of the learning process.

The contributions of this work are four-fold. First, an innovative experimental platform is developed for autonomous control of flexible-wing aircraft. This is known to be a complicated task which was not realized using online model-free control systems before introducing this work. Second, a new machine learning process is proposed. It uses real-time measurements and it conditions the optimization objectives in order to guide the search for the best control strategies in an online fashion with high success rate. Third, it provides ideas related to the online solutions of the optimal tracking control problems, which are solved basically in an offline fashion. This approach does not employ any model-dependent control strategies and it is easily integrated into an off-the-shelf computing unit, such as a Raspberry Pi. Finally, the research presented herein provides a flexible framework that can be generalized for controlling complex nonlinear dynamical systems of the same class as flexible-wing aircraft.

The remaining sections of the paper are arranged as follows. The operation of flexible-wing aircraft is briefly highlighted in Section II. The measurement scheme and the adopted real-time sensory devices are detailed in Section III. The different control and optimization objectives are presented in Section IV. This is followed by the development of the machine learning process and its adaptive critics implementation in Section V. The digital experimental results are analyzed in Section VI in order to evaluate the validity of the proposed control setup. Finally, conclusions are drawn in Section VII.

II. Operation of Flexible-wing Aircraft

In this section, the basic kinematic relationships concerned with pitch motion control of the flexible-wing aircraft are detailed out. The two-mass system (i.e., wing and fuselage) attaches the two bodies by a hang block, a joint that connects the mast of the fuselage and the keel tube of the wing at a common point known as the hang point. The hang block is a mechanical joint which allows for two degrees of motion relative to each body. The wing system may roll freely about the longitudinal axis of the keel tube, while the keel bar may pitch about the axis which is perpendicular to both the mast and keel tubes. During manned flight, the pilot who is positioned within the fuselage, modifies the wings orientation with respect to the fuselage by manipulating the control bar that is rigidly attached to the wing. To modify the vehicle's orientation, the pilot applies a force to the control bar to shift the wing with respect to the fuselage. By the principle of weight shift, this in turn modifies the lift profile of the wing, resulting in a new wing orientation with respect to the fuselage.

The work presented in this paper is a first phase of a long-term project aiming at devising a model-free control algorithm for a commercial flexible-wing cargo aircraft. The first phase focuses on controlling the wing's orientation with respect to the fuselage when the aircraft is either stalled on the ground or is in a taxi mode on the runway. To this end, an experimental mock-up is set up to emulate the relative actuation mechanism between the wing and fuselage. The weight-shift kinematics of this mechanism is schematically depicted in Figure 1. A pair of servomotor winches are mounted at point M on the mast, and connected to points K and K' on the keel tube. The orientation of the wing with respect to the fuselage is represented by the pair of rotation angles, θ and ϕ corresponding to pitch and roll of the wing. It is important to note that the wing does not rotate along the yaw axis, represented by ψ, as it is constrained by the hang point on that axis. The set point of the control bar corresponds to the resting orientation of the wing in stable steady altitude and steady speed flight. Pull-only forces applied to the wing by the winch servomotors are represented by the pair of position vectors f_{fore} and f_{aft}, and act in the free motion axis of pitch perpendicular to the keel tube with unknown angles proportional to $\angle KK'M$ and $\angle K'KM$, respectively. For this work, it is assumed that the wing control in the pitch axis is sufficiently decoupled from the roll axis, such that the servomotor winch actions due to their geometric configuration do not induce the roll motion of the wing.

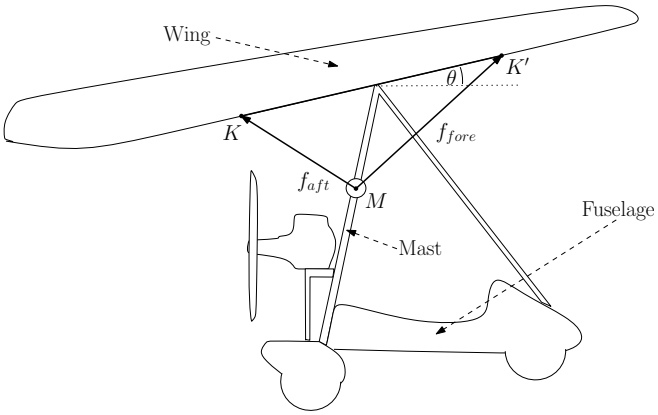


Fig. 1: Schematic of weight shift kinematics of flexible-wing aircraft with two servo winches acting on wing to affect pitch.

III. Instrumentation and Measurement Platform

The experimental mock-up emulating the relative motion between the fuselage and the wing is shown in Figure 2. A block diagram of how the system's interconnected blocks are interfaced is revealed in Figure 3. Efforts were made to ensure each hardware device and supporting software is either open-source or has publicly available documentation. An Emlid Navio2 [46] board attached to a Raspberry Pi 3 [47] is selected as the controller's computational unit due its relatively convenient and easy user setup and prevalence as a hardware of choice for open-source flight controller platforms, such as ArduPilot [48] and ROS [49]. Navio2 provides redundant wing orientation estimates through the combination of sensor readings from either the main on-board InvenSense MPU-9250 9-Degree-of-Freedom (DOF) Inertial Measurement Unit (IMU) [50] or the secondary Microelectronics LSM9DS1 9 DOF IMU [51], as well as

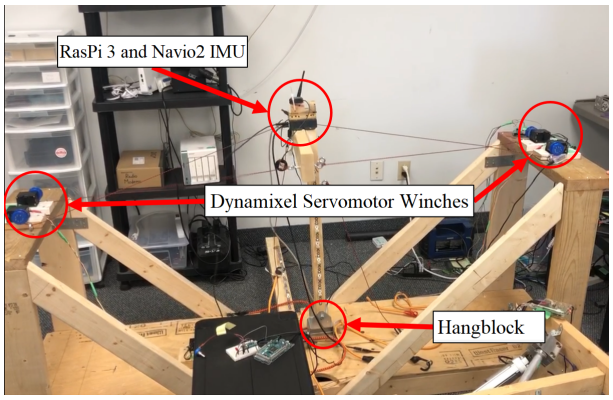


Fig. 2: Experimental mock-up used to emulate the motion of the fuselage about the wing in ground test. Note that the wing-mast system is oriented upside down; i.e., the top surface of the middle bar, representing the wing, is facing the floor, while the mast likewise sits above it. The figure also shows the Navio2+Raspi IMU and controller, servomotor winches, and hangblock.

global positioning system (GPS). Each IMU has a three-axis magnetometer, three-axis gyroscope, and three-axis accelerometer. The IMU signals are filtered by a programmable digital low pass filter, which is then fed to a portable mini-computer (Raspberry Pi 3 model B). The unit has a 1.2GHz 64-bit quad-core ARMv8 CPU, 1GB of RAM and a Cortex-M3 co-processor.

Angular position measurements of the wing orientation were sampled at 20 Hz, a rate far below the maximum of 8 kHz supported by the dual IMUs. The total root mean square (RMS) noise of the MPU-9250 used for inertial measurement feedback is provided to be $0.1\,^{\circ}/\mathrm{s}$. The measurements data are filtered by a low-pass filter with a cut-off frequency of 250 Hz. The control of the platform was actuated by two Dynamixel XM-430 servomotors [52]. The servomotors receive current input commands at a rate of 20 Hz from the controller embedded within the Raspberry Pi 3. In reference to Figure 2, the system is arranged as if the aircraft wing was upside down, where the hangblock and wing are closest to the ground, and the mast sits above it. This setup was chosen for its convenience but it should have no effect on the validity of the experimental results. The mock-up allows for the mast to move relative to the wing. The servomotor winches are rigidly attached to the wing keel, and the attachment points are at opposite points along the mast. The results presented in Section VI are captured from experiments with this mock-up system where two servomotors are mounted directly to the wing. Desired reference position commands are programmed prior to the experiments. The learning algorithm decides the best control policy using IMU observations, in real time, and transmits current signals, based torque commands, to the servomotors to track the reference signal.

IV. The Control and Optimization Problem

In the following section, ideas behind the online machine learning process are first developed. Thereafter, the governing Bellman or temporal difference equations are derived. Further, this section discusses how the different control tasks (i.e., tracking and stabilization) are coordinated simultaneously in real-time.

A. Optimal Control Structure

Due to the aircraft's complex nonlinear dynamics, it is necessary to avoid building a control or computational approach that relies explicitly on existence of an aerodynamic model. As such, any control solution would better be based on robust model-free computational mechanisms.

The wing's pitch control mechanism is realized using two interacting control objectives. The first looks for a control strategy that receives real-time tracking error signals, arranged using pre-designed criteria, and provides a control signal that minimizes the tracking error. The second searches for an auxiliary control policy that supports the stability of the overall system during the maneuvers. The full autonomous control system is schematically represented in Figure 4. These control objectives are integrated together and implemented using a guided search algorithm that is developed herein based on an innovative machine learning process (as shall be discussed later).

The structure of the tracking error vector $E_\ell \in \mathbb{R}^o$ reflects the design objectives corresponding to the tracking control strategy π_E which in turn decides the tracking control signal $u_\ell^{\pi_E} \in \mathbb{R}^m$. The error vector E_ℓ relies on various dynamic forms of the tracking error signals e_ℓ (ℓ is a time-index). On another side, the stabilizing control policy π_X maps the observable key measurements in vector $X_\ell \in \mathbb{R}^n$ to a state feedback control signal $u_\ell^{\pi_X} \in \mathbb{R}^s$ in order to support the system's stability during the navigation process. The overall control $u_\ell^F \in \mathbb{R}^{\max\{s,m\}}$ which is applied to the actuation system, results from combining the dynamical effects from both control policies (i.e., π_E and π_X). The control signals u^{π_E}, u^{π_X}, and hence u^F, are the actuation signals responsible to move the control bar of the flexible-wing aircraft. It is worth to note that in order to generalize the proposed approach, the control signals u^{π_E} and u^{π_X} could have different vector sizes and the total control signal u^F sums the dynamical effects of the matched actuation signals in both control laws. In this work, the control signals which are fed to the actuation systems are scalars. This simplifies the form of the collective control signal u^F.

Performance indices $J^{\pi_E}(E_\ell)$ and $J^{\pi_X}(X_\ell)$ are used to asses the usefulness of the attempted policies π_E and π_X, respectively, so that

$$\begin{aligned} J^{\pi_E}(\boldsymbol{E}_\ell) &= \sum_{i=\ell}^{\infty} C^E\left(\boldsymbol{E}_i, \boldsymbol{u}_i^{\pi_E}\right) \\ J^{\pi_X}(\boldsymbol{X}_\ell) &= \sum_{i=\ell}^{\infty} C^X\left(\boldsymbol{X}_i, \boldsymbol{u}_i^{\pi_X}\right), \end{aligned} \tag{1}$$

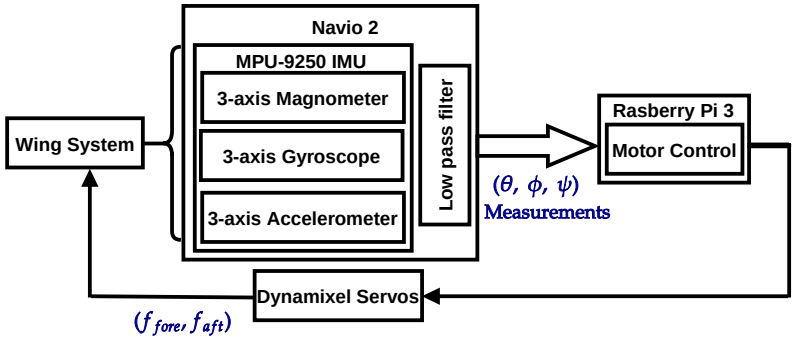


Fig. 3: Flow schematic of experimental instrumentation.

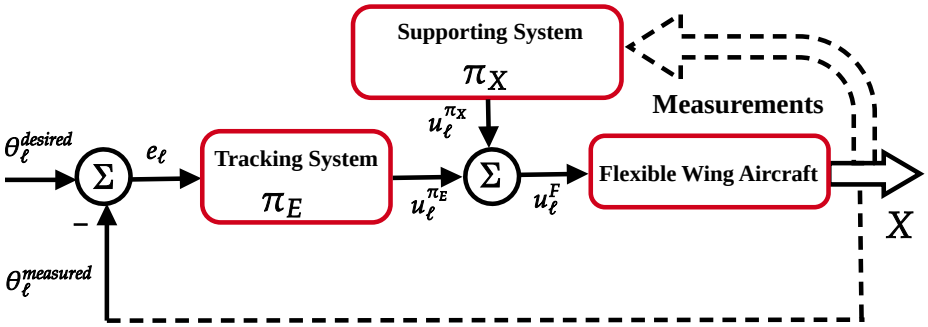


Fig. 4: Feedback control loop.

where C^E and C^X are cost functions associated with the performance indices $J^{\pi_E}(\boldsymbol{E}_\ell)$ and $J^{\pi_X}(\boldsymbol{X}_\ell)$, respectively. They are given by

$$C^E(\boldsymbol{E}_\ell, \boldsymbol{u}_\ell^{\pi_E}) = \frac{1}{2}\left(\boldsymbol{E}_\ell^T\, \boldsymbol{Q}^E\, \boldsymbol{E}_\ell + \boldsymbol{u}_\ell^{\pi_E\, T}\, \boldsymbol{R}^E\, \boldsymbol{u}_\ell^{\pi_E}\right)$$
$$C^X(\boldsymbol{X}_\ell, \boldsymbol{u}_\ell^{\pi_X}) = \frac{1}{2}\left(\boldsymbol{X}_\ell^T\, \boldsymbol{Q}^X\, \boldsymbol{X}_\ell + \boldsymbol{u}_\ell^{\pi_X\, T}\, \boldsymbol{R}^X\, \boldsymbol{u}_\ell^{\pi_X}\right),$$

where $\boldsymbol{Q}^E \in \mathbb{R}^{o\times o}$, $\boldsymbol{R}^E \in \mathbb{R}^{m\times m}$, $\boldsymbol{Q}^X \in \mathbb{R}^{n\times n}$, and $\boldsymbol{R}^X \in \mathbb{R}^{s\times s} > 0$ are symmetric positive definite weighting matrices.

The objective functions C^E and C^X have convex forms and quadratic dependencies on the different policies and real-time measurements. These forms motivate and enable Lyapunov stability proofs for the underlying temporal difference solutions [31], [53].

Remark 1: The design of the optimization problem could variate from the one that is considered herein. This is decided by the designer which judges the different segments of the optimization problem and how they are hierarchically organized. In this development, it depends on the way key measurements are allowed and collected for the control design and on the dynamics which the controller needs to consider or regulate during the navigation process.

B. Mathematical Solution Framework

The control solution developed herein maps the optimization objectives mentioned above into temporal difference forms using a discrete-time optimal control framework [4]. Quadratic forms of solving value functions $V^{\pi_E}(\boldsymbol{E}_\ell)$ and $V^{\pi_X}(\boldsymbol{X}_\ell)$ are advised to evaluate the quality of the computed control strategies. They are motivated based on the structures of the indices $J^{\pi_E}(\boldsymbol{E}_\ell)$ and $J^{\pi_X}(\boldsymbol{X}_\ell)$ and the associated cost functions C^E and C^X, respectively, such that

$$V^{\pi_E}(\boldsymbol{E}_\ell) = \frac{1}{2}\boldsymbol{E}_\ell^T\, \boldsymbol{S}^E\, \boldsymbol{E}_\ell \quad \equiv \quad J^{\pi_E}(\boldsymbol{E}_\ell)$$
$$V^{\pi_X}(\boldsymbol{X}_\ell) = \frac{1}{2}\boldsymbol{X}_\ell^T\, \boldsymbol{S}^X\, \boldsymbol{X}_\ell \quad \equiv \quad J^{\pi_X}(\boldsymbol{X}_\ell),$$

where $S^E \in \mathbb{R}^{o\times o}$ and $S^X \in \mathbb{R}^{n\times n}$ are solution matrices. They play a major role in the guided search policies found using the online adaptive learning process.

The matrices S^E and S^X are computed using a temporal difference solution framework in real-time, and hence are relevant to the choice of the best policies π_E and π_X, respectively, using the interactive learning process.

The value functions V^{π_E} and V^{π_X} are utilized to form Bellman equations (temporal difference solution structures) for the underlying control mechanism, such that

$$V^{\pi_E}(\boldsymbol{E}_\ell) = C^E(\boldsymbol{E}_\ell, \boldsymbol{u}_\ell^{\pi_E}) + V^{\pi_E}(\boldsymbol{E}_{\ell+1})$$
$$V^{\pi_X}(\boldsymbol{X}_\ell) = C^X(\boldsymbol{X}_\ell, \boldsymbol{u}_\ell^{\pi_X}) + V^{\pi_X}(\boldsymbol{X}_{\ell+1}). \qquad (2)$$

These Bellman equations indicate that two interacting optimization processes are solved simultaneously, where it is required to drive the tracking error $\boldsymbol{E}$ to zero and, at the same time, optimize the dynamics $\boldsymbol{X}$ along the navigation trajectories. Further, Bellman equations (2) enable the integration between two environments. The first is related to solving

the navigation control problem, while the second enables an approximate dynamic programming solution (i.e., machine learning solution) for the problem. The online adaptive learning control process starts with an initial control strategy for each control decision and then the learning process, employing the above Bellman equations, directs the control strategies (i.e., learn better control decisions) using a value iteration process which is guaranteed to converge.

Herein, the error vector $\boldsymbol{E}_\ell$ is structured as follows:

$$\boldsymbol{E}_\ell = \begin{bmatrix} e_\ell & e_{\ell-1} & e_{v\ell} & e_{v\ell-1} & e_{s\ell} & s_{s\ell-1} \end{bmatrix}^T$$

where $e_{v\ell}$ and $e_{v\ell-1}$ are error derivatives, with respect to time, evaluated at time ℓ and $\ell-1$, respectively. $e_{s\ell}$ and $s_{s\ell-1}$ are the moving averages calculated as $e_{s\ell} = \dfrac{1}{N} \displaystyle\sum_{i=\ell-N}^{\ell} e_i$ and $e_{s\ell-1} = \dfrac{1}{N} \displaystyle\sum_{i=\ell-N-1}^{\ell-1} e_i$, respectively.

The way the error vector $\boldsymbol{E}_\ell$ is calculated, combines many sub-objectives which are optimized together. It minimizes the local tracking error e_ℓ while considering its previous instance $e_{\ell-1}$. Further, it smoothens the control decision by looking backward in-time to include error derivatives evaluated at time instances ℓ and $\ell-1$, as well as evaluating average of the errors across longer time-intervals N. It is worth to mention that the vector formulation $\boldsymbol{E}_\ell$ enables more advanced forms of equivalent discrete Proportional-Integral-Derivative (PID) structures (i.e., dependence on $e_\ell, e_{\ell-1}$, and $e_{\ell-2}$) or even considers higher-order derivatives and integral equivalents (i.e., dependence on $e_\ell, e_{\ell-1}, e_{\ell-2}, \ldots,$ and $e_{\ell-N}$).

The solution of the optimal tracking problem relies on solving a number of differential equations, a subgroup of which is solved offline and at the same time they do not allow dynamical forms of the error signals. Herein, the formulation of the adaptive learning mechanism allows for an online solution as well as using a variety of tracking error dynamical forms.

The experimental setup, described in Section III, provides measurements related to the aerodynamic orientation of the wing. This made it convenient to choose the states $\boldsymbol{X}_\ell$ as

$$\boldsymbol{X}_\ell = \begin{bmatrix} \theta_\ell & \theta_{v\ell} & \theta_{a\ell} \end{bmatrix}^T,$$

where $\theta_\ell, \theta_{v\ell}$, and $\theta_{a\ell}$ are the pitch attitude, pitch velocity, and pitch acceleration, respectively.

The desired control policies $\boldsymbol{\pi}_E$ and $\boldsymbol{\pi}_X$ are linear feedback polices which are used to decide on the different control signals, such that

$$\boldsymbol{u}_\ell^{\pi_E} = \boldsymbol{\pi}_E \, \boldsymbol{E}_\ell, \quad \boldsymbol{u}_\ell^{\pi_X} = \boldsymbol{\pi}_X \, \boldsymbol{X}_\ell, \tag{3}$$

Next, we will explain how to compute the real-time negative feedback control strategies $\boldsymbol{\pi}_E$ and $\boldsymbol{\pi}_X$.

V. Machine Learning Platform

The online computational machine learning framework is driven by the design of the control problem. First, we will present a computational platform based on an innovative value iteration implementation for an online reinforcement learning solution. Then, adaptive critics are employed to provide neural network implementation for the approximate dynamic programming solution.

A. Guided Search Process

The policies $\boldsymbol{\pi}_E$ and $\boldsymbol{\pi}_X$ are guided toward the intended optimization objectives stated by the designer as follows:

$$\boldsymbol{u}^{\pi_E} = -\boldsymbol{P}^E \frac{\partial V^{\pi_E}(\boldsymbol{E}_\ell)}{\partial \boldsymbol{E}_\ell}, \quad \boldsymbol{u}^{\pi_X} = -\boldsymbol{P}^X \frac{\partial V^{\pi_X}(\boldsymbol{X}_\ell)}{\partial \boldsymbol{X}_\ell}, \tag{4}$$

where $\boldsymbol{P}^E \in \mathbb{R}^{m \times o}$ and $\boldsymbol{P}^X \in \mathbb{R}^{s \times n}$ are guiding search vectors which carry the intentions and dynamic preferences of the control mechanism designer.

The vectors $\boldsymbol{P}^E$ and $\boldsymbol{P}^X$ contain selective dynamic forms (i.e., introduced by the designer to reflect their objectives regarding the entries in vectors $\boldsymbol{E}$ and $\boldsymbol{X}$) as will be highlighted in the experimental analysis. This in turn promotes flexibility for a new class of online learning processes with guided search features.

The control polices $\boldsymbol{\pi}_E = -\boldsymbol{P}^E \, \boldsymbol{S}^E$ and $\boldsymbol{\pi}_X = -\boldsymbol{P}^X \, \boldsymbol{S}^X$ (i.e., these associated with $\dfrac{\partial V^{\pi_E}(\boldsymbol{E}_\ell)}{\partial \boldsymbol{E}_\ell}$ and $\dfrac{\partial V^{\pi_X}(\boldsymbol{X}_\ell)}{\partial \boldsymbol{X}_\ell}$) are applied in real-time using a value iteration process as will be explained next.

B. Value Iteration Algorithm

An online reinforcement learning solution is developed using the aforementioned Bellman equations (2). The solution is realized using a two-step value iteration process. The first updates the solving value functions (i.e., $\boldsymbol{S}^E$ and $\boldsymbol{S}^X$) using (2) while the second extracts the new or improved policies (i.e., $\boldsymbol{\pi}_E$ and $\boldsymbol{\pi}_X$). The online value iteration process is detailed out in Algorithm 1. Bellman equations provide the temporal difference platform necessary to guide the online learning process beyond the initially selected polices $\boldsymbol{\pi}_E^0$ and $\boldsymbol{\pi}_X^0$. Note that, unlike policy iteration paradigms, these initial policies do not need to be admissible. Hence, the intentionally guided policies are improved along the trajectory of the aircraft.

Remark 2: Value iteration processes are proven to converge in general for single and multi-agent systems based on Lyapunov stability approaches if the underlying systems are stabilizable [30], [53], [54]. This is mainly due to the properties of convex objective cost functions (i.e., C^E and C^X in this work) and consequently Bellman solution forms (i.e., (2)). Hence, the value functions evolve in a bounded manner, such that

$$\begin{aligned} 0 \leq V^{\pi_E(0)}(\boldsymbol{E}_\ell) \leq V^{\pi_E(1)}(\boldsymbol{E}_\ell) \leq \cdots \leq \\ V^{\pi_E(t)}(\boldsymbol{E}_\ell) \leq \cdots \leq V^{\pi_E(*)}(\boldsymbol{E}_\ell), \\ 0 \leq V^{\pi_X(0)}(\boldsymbol{X}_\ell) \leq V^{\pi_X(1)}(\boldsymbol{X}_\ell) \leq \cdots \leq \\ V^{\pi_X(t)}(\boldsymbol{X}_\ell) \leq \cdots \leq V^{\pi_X(*)}(\boldsymbol{X}_\ell), \end{aligned} \tag{5}$$

where the value functions $V^{\pi_E(*)}$ and $V^{\pi_X(*)}$ are the optimal response solutions for Bellman equations (2). In other words,

the tracking error and motion dynamics are stable if the aircraft system is controllable.

The online value iteration learning process guides the solving value functions toward the best values $V^{\pi_E(*)}(\boldsymbol{E}_\ell)$ and $V^{\pi_X(*)}(\boldsymbol{X}_\ell)$, and hence best guided policies $\boldsymbol{\pi}_E^*$ and $\boldsymbol{\pi}_X^*$.

C. Adaptive Critics Implementation

Adaptive critics are employed as neural network approximation tools for the introduced online reinforcement learning solution [37]. The adaptive critics scheme is implemented using means of neural network structures, namely actor-critic networks, for each optimization control process. They provide guided search solutions for the underlying Bellman equations. The critic and actor structures approximate the solving value functions and the associated policies in an interactive manner implicitly during the iterative update of underlying Bellman equations. The actor network reflects the improvements in the guided control strategy while its quality is approximated by the critic network. The weights of the actor-critic structures are adapted using a gradient decedent approach motivated by the linear forms of the control policies.

The value functions V^{π_E} and V^{π_X} (i.e., the critic structures) are approximated so that

$$\hat{V}^{\pi_E}(\boldsymbol{E}_\ell) = \frac{1}{2}\boldsymbol{E}_\ell^T\,\boldsymbol{\Omega}_c^E\,\boldsymbol{E}_\ell, \hat{V}^{\pi_X}(\boldsymbol{X}_\ell) = \frac{1}{2}\boldsymbol{X}_\ell^T\,\boldsymbol{\Omega}_c^X\,\boldsymbol{X}_\ell,$$

where $\boldsymbol{\Omega}_c^E \in \mathbb{R}^{o\times o}$ and $\boldsymbol{\Omega}_c^X \in \mathbb{R}^{n\times n}$ are the critic approximation weights of the value functions $\hat{V}^{\pi_E}(\boldsymbol{E}_\ell)$ and $\hat{V}^{\pi_X}(\boldsymbol{X}_\ell)$, respectively. The critic network forms are motivated by the structures of the value functions V^{π_E} and V^{π_X} respectively.

In a similar fashion, the guided search polices (i.e., the actor structures) are approximated so that

$$\hat{\boldsymbol{u}}_\ell^{\pi_E} = \boldsymbol{\Omega}_a^E\boldsymbol{E}_{\boldsymbol{\ell}}, \quad \hat{\boldsymbol{u}}_\ell^{\pi_X} = \boldsymbol{\Omega}_a^X\boldsymbol{X}_{\boldsymbol{\ell}},$$

where $\boldsymbol{\Omega}_a^E \in \mathbb{R}^{m\times o}$ and $\boldsymbol{\Omega}_a^X \in \mathbb{R}^{s\times n}$ are the actor approximation weights of the policies $\boldsymbol{\pi}_E$ and $\boldsymbol{\pi}_X$, respectively.

The different approximation weights are updated using a gradient descent approach that applies a minimization criteria on the squared approximation errors. The approximation errors of the critic networks have squared forms, so that

$$\begin{aligned}\varepsilon_c^{E_\ell} &= \frac{1}{2}\left(\hat{V}^{\pi_E}(\boldsymbol{E}_\ell) - \hat{V}^{T_E}(\boldsymbol{E}_\ell)\right)^2,\\ \varepsilon_c^{X_\ell} &= \frac{1}{2}\left(\hat{V}^{\pi_X}(\boldsymbol{X}_\ell) - \hat{V}^{T_X}(\boldsymbol{X}_\ell)\right)^2,\end{aligned} \tag{6}$$

where the target values $\hat{V}^{T_E}(\boldsymbol{E}_\ell)$ and $\hat{V}^{T_X}(\boldsymbol{X}_\ell)$ are calculated using

$$\begin{aligned}\hat{V}^{T_E}(\boldsymbol{E}_\ell) &= C^E(\boldsymbol{E}_\ell, \hat{\boldsymbol{u}}_\ell^{\pi_E}) + \hat{V}^{\pi_E}(\boldsymbol{E}_{\ell+1})\\ \hat{V}^{T_X}(\boldsymbol{X}_\ell) &= C^X(\boldsymbol{X}_\ell, \hat{\boldsymbol{u}}_\ell^{\pi_X}) + \hat{V}^{\pi_X}(\boldsymbol{X}_{\ell+1}).\end{aligned}$$

The critic weights are adapted according to a gradient descent rule such that

$$\begin{aligned}\boldsymbol{\Omega}_c^{E(t+1)} &= \boldsymbol{\Omega}_c^{E(t)} - \alpha_c^E\left[\frac{\partial\varepsilon_c^{E_\ell}}{\partial\boldsymbol{\Omega}_c^E}\right]^{(t)}\\ \boldsymbol{\Omega}_c^{X(t+1)} &= \boldsymbol{\Omega}_c^{X(t)} - \alpha_c^X\left[\frac{\partial\varepsilon_c^{X_\ell}}{\partial\boldsymbol{\Omega}_c^X}\right]^{(t)},\end{aligned} \tag{7}$$

where $0 < \alpha_c^E, \alpha_c^X < 1$ are critic networks learning rates, $\frac{\partial\varepsilon_c^{E_\ell}}{\partial\boldsymbol{\Omega}_c^E} = \left(\hat{V}^{\pi_E}(\boldsymbol{E}_\ell) - \hat{V}^{T_E}(\boldsymbol{E}_\ell)\right)\boldsymbol{E}_\ell\,\boldsymbol{E}_\ell^T$, $\frac{\partial\varepsilon_c^{X_\ell}}{\partial\boldsymbol{\Omega}_c^X} = \left(\hat{V}^{\pi_X}(\boldsymbol{X}_\ell) - \hat{V}^{T_X}(\boldsymbol{X}_\ell)\right)\boldsymbol{X}_\ell\,\boldsymbol{X}_\ell^T$, and t refers to the iteration update index.

Similarly, the squared approximation errors for the actor networks are defined by

$$\varepsilon_a^{E_\ell} = \frac{1}{2}\left(\hat{\boldsymbol{u}}_\ell^{\pi_E} - \hat{\boldsymbol{u}}_\ell^{T_E}\right)^2, \quad \varepsilon_a^{X_\ell} = \frac{1}{2}\left(\hat{\boldsymbol{u}}_\ell^{\pi_X} - \hat{\boldsymbol{u}}_\ell^{T_X}\right)^2,$$

where the target values $\hat{\boldsymbol{u}}_\ell^{T_E}$ and $\hat{\boldsymbol{u}}_\ell^{T_X}$ are calculated as follows

$$\hat{\boldsymbol{u}}_\ell^{T_E} = -\boldsymbol{P}^E\,\boldsymbol{\Omega}_c^E\boldsymbol{E}_\ell, \quad \hat{\boldsymbol{u}}_\ell^{T_X} = -\boldsymbol{P}^X\,\boldsymbol{\Omega}_c^X\boldsymbol{X}_\ell.$$

Using gradient descent, the actor network weights are adapted according to the following rule

$$\begin{aligned}\boldsymbol{\Omega}_a^{E(t+1)} &= \boldsymbol{\Omega}_a^{E(t)} - \alpha_a^E\left[\frac{\partial\varepsilon_a^E}{\partial\boldsymbol{\Omega}_a^{E_\ell}}\right]^{(t)}\\ \boldsymbol{\Omega}_a^{X(t+1)} &= \boldsymbol{\Omega}_a^{X(t)} - \alpha_a^X\left[\frac{\partial\varepsilon_a^X}{\partial\boldsymbol{\Omega}_a^{X_\ell}}\right]^{(t)},\end{aligned} \tag{8}$$

where $0 < \alpha_a^E, \alpha_a^X < 1$ are actor networks learning rates, $\frac{\partial\varepsilon_a^{E_\ell}}{\partial\boldsymbol{\Omega}_a^E} = \left(\hat{\boldsymbol{u}}_\ell^{\pi_E} - \hat{\boldsymbol{u}}_\ell^{T_E}\right)\boldsymbol{E}_\ell^T$, and $\frac{\partial\varepsilon_a^{X_\ell}}{\partial\boldsymbol{\Omega}_a^X} = \left(\hat{\boldsymbol{u}}_\ell^{\pi_X} - \hat{\boldsymbol{u}}_\ell^{T_X}\right)\boldsymbol{X}_\ell^T$.

A full schematic diagram of the adaptive critics process is shown in Figure 5 for the combined control problem in hand. The different actor and critic weights are updated simultaneously in real-time following the scope of Algorithm 1.

D. Complexity of the Learning Mechanism

Algorithm 1 provides multiple merits concerning the complexity and scalability of the navigation or tracking problems. First, the learning process can be conditioned for any number of tracking signals simultaneously (i.e., for a large tracking or navigation process that contains many tracking objectives, that would be easy to implement compared to solving a high number of differential equations in an offline mode). Second, the tracking error signals could involve many dynamical forms (i.e., moving velocity, acceleration, moving average, etc) which were not possible or easy-to-implement before. Third, this scheme allows for coupled optimized dynamical processes (i.e., possible optimization of multiple coupled and interactive dynamical objectives for multiple problems without the need to independently solve each problem). Finally, the arrangement of Bellman equations enables simple and straightforward adaptive critics solution, since the control strategies hold linear forms.

Algorithm 1: Online Value Iteration Process.

Input: Desired trajectory $\theta_\ell^{desired}$, weighting matrices $\boldsymbol{Q}^E, \boldsymbol{R}^E, \boldsymbol{Q}^X$, and $\boldsymbol{R}^X$, guiding search vectors $\boldsymbol{P}^E$ and $\boldsymbol{P}^X$, tracking error evaluation interval N.

Output: Policies $\boldsymbol{\pi}_E$, $\boldsymbol{\pi}_X$, and tracking error e_ℓ, for $\ell = 0, 1, \ldots$

begin
- $\ell = 0, t = 0$ /* time and strategy indices */
- Initialize solving matrices $\boldsymbol{S}^{E\{0\}}$ and $\boldsymbol{S}^{X\{0\}}$ /* Positive definite */ and hence initial policies $\boldsymbol{\pi}_E^0$ and $\boldsymbol{\pi}_X^0$ and tracking errors interval $e_{\ell-1}, \ldots, e_{\ell-N-1}$
- Measure $\theta_\ell, \theta_{v\ell}, \theta_{a\ell}$ and then calculate the errors e_ℓ, $e_{\ell-1}$, $e_{v\ell}$, $e_{v\ell-1}$, $e_{s\ell}$, $e_{s\ell-1}$
- **repeat** /* Training/Search loop */
  - Compute the different control signals $u^{\pi_E(t)}$ and $u^{\pi_X(t)}$ using (4)
  - Measure $\theta_{\ell+1}, \theta_{v\ell+1}, \theta_{a\ell+1}$ and then calculate the errors $e_{\ell+1}$, e_ℓ, $e_{v\ell+1}$, $e_{v\ell}$, $e_{s\ell+1}$, $e_{s\ell}$
  - Evaluate the solving value functions $V^{\pi_E(t+1)}(\boldsymbol{E}_\ell) = C^E(\boldsymbol{E}_\ell, \boldsymbol{u}_\ell^{\pi_E(t)}) + V^{\pi_E(t)}(\boldsymbol{E}_{\ell+1})$ $V^{\pi_X(t+1)}(\boldsymbol{X}_\ell) = C^X(\boldsymbol{X}_\ell, \boldsymbol{u}_\ell^{\pi_X(t)}) + V^{\pi_X(t)}(\boldsymbol{X}_{\ell+1})$.
  - Extract the improved control policies ${\pi_E}^{t+1} = -\boldsymbol{P}^E S^{E(t+1)}$, ${\pi_X}^{t+1} = -\boldsymbol{P}^X S^{X(t+1)}$,
  - $\ell \leftarrow \ell + 1$ /* Update real-time index */
  - $t \leftarrow t + 1$ /* Update policy index */
- **until** *Satisfactory trajectory-tracking performance (i.e., acceptable tracking error).*

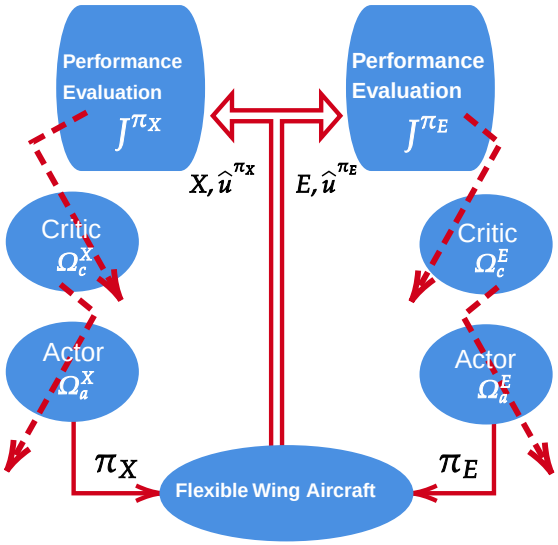


Fig. 5: Adaptive critics structure.

VI. Experimental Results

The online model-free adaptive learning mechanism is validated in a real-time environment using a two-servomotor pitch actuation system. The system generates servomotor pull actions dependent on the current-measured pitch position and the desired trajectory steering the wing. For the sake of experiments presented below, the trajectory reference is chosen to be sinusoidal (i.e., continuous pitch up/down commands).

A. Learning Environment

The learning parameters of the system are selected in order to reflect the physical constraints of the measured dynamical parameters and also encode the control design preferences that prioritize the desired system response. The weighting matrices capture the physical limitations of the dynamical parameters (i.e., the states $\theta_\ell, \theta_{v\ell}, \theta_{a\ell}, e_\ell$ and the actuation control signal u^F). They are chosen as $R^E = 10^{-7}$, $R^X = 10$,

$$Q^E = 10^{-4} \begin{bmatrix} 25 & 0 & 0 & 0 & 0 & 0 \\ 0 & 25 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.25 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 25 & 0 \\ 0 & 0 & 0 & 0 & 0 & 25 \end{bmatrix},$$

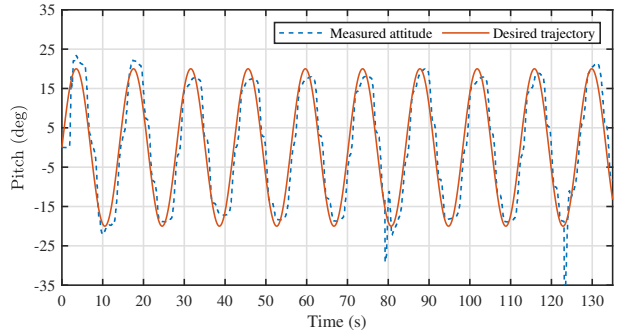
$$Q^X = 10^{-6} \begin{bmatrix} 25 & 0 & 0 \\ 0 & 25 & 0 \\ 0 & 0 & 0.0025 \end{bmatrix}.$$

Vectors P^E and P^X for the guided search policies π^E and π^X are selected so that $P^E = \begin{bmatrix} 200 & 50 & 10 & 5 & 10 & 5 \end{bmatrix}$ and $P^X = \begin{bmatrix} 10 & 10 & 5 \end{bmatrix}$. The vector P^E assigns more weight to minimizing the recently measured tracking errors e_ℓ and $e_{\ell-1}$, as opposed to the error velocity terms $e_{v\ell}$ and $e_{v\ell-1}$, to smooth down the nonlinearity transitions. Furthermore, the influences of the moving average tracking errors $e_{s\ell}$ and $e_{s\ell-1}$ on the overall performance are weighted similarly to those of the error velocities. The vector P^X reflects the gradual dynamic importance of the pitch attitude θ_ℓ and the pitch velocity $\theta_{v\ell}$ over the angular acceleration $\theta_{a\ell}$. The actor and critic learning rates are selected to be small enough in order to match the actor and critic adjustments in a smooth manner. They are set to $\alpha_c^E = 0.01$, $\alpha_c^X = 0.01$, $\alpha_a^E = 0.01$, and $\alpha_a^X = 0.05$. The desired navigation trajectory is an independent sinusoidal reference signal that takes the form $\theta_\ell^{desired} = 20 \sin(0.132\,\pi\,\ell)$ deg.

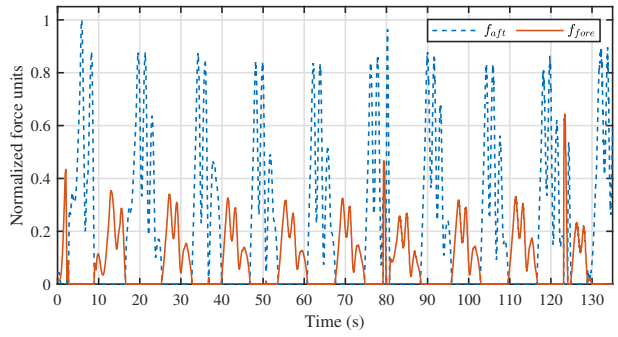
B. Test Scenarios

The adaptive learning controller is validated using three test scenarios. In the first scenario, the system is tested under nominal circumstances where no external disturbance is applied on the system. The dynamic performance of the proposed adaptive learning controller is shown in Figure 6. This experiment reveals that, the learning algorithm successfully converges to a stable control policy which prescribes correct servomotor pull-forces for the observed system state (i.e., pitch attitude). The absolute average tracking error over the test period is found to be 0.45 deg. It is worth noticing that

the convergence behavior of the proposed control mechanism is not affected by the occasional sudden erroneous readings in the feedback measurements, as revealed at around time instants $80\,\mathrm{s}$ and $123\,\mathrm{s}$ in Figure 6a. Such measurement errors are due to some imperfections in the sensing units which could not be overcame by the adopted low-pass filter. The convergence of the adaptive learning approach is clear in Figure 7, where the actor and critic weights converge throughout the online learning process. The overall normalized forces generated by the servomotors are shown in Figure 6b. Finally, the convergence of the value iteration approach, as described in Section V, is highlighted in Figure 8. It is shown that, despite the sensor reading spikes, a general bounded non-decreasing convergence pattern is observed, as expected for the underlying value iteration process.



(a)



(b)

Fig. 6: Control performance during online learning process: (a) measured vs. desired pitch attitude, (b) acting forces on the wing's keel, f_{fore} and f_{aft}.

The second scenario is set up to demonstrate the ability of the controller to reject sudden external and forced disturbances. The test case is a repetition of the first, except that this time, a sudden disturbance was applied at around $120\,\mathrm{s}$ for about $10\,\mathrm{s}$ by violently jerking the control bar around the pitch and roll axes to stimulate the effects of wind gusts on the wing during the flight. The tracking performance of the proposed controller is demonstrated in Figure 9a. The controller is started with converged control gains from a previous learning episode. As such, minimal changes in the critic weights are witnessed, as shown by the critic weights evolution in Figure 10. During disturbance, the critic reacts appropriately to generate large abrupt motor torques to counteract the induced perturbations, as shown in Figure 9b. Once the dynamics are back to their nominal conditions, after that the disturbance subsided, the controller successfully resumes the tracking of the commanded pitch.

The goal of the third experiment is to show that once the controllers converge to some optimal strategies π_E^* and π_X^*, the actor's learning mechanism can be turned off without degrading the control performance. By doing so, the controller is made to operate with static control gains. For this test case, the adopted static control policies applied are those converged from a prior experiment. The controller's performance is illustrated in Figure 11. The absolute average tracking error over the test period is found to be $0.94\ \mathrm{deg}$. This result emphasizes that the converged strategies from static "actor-only" controllers previously derived from the adaptive learning mechanism remain valid during the trajectory tracking process. Notice how this experiment also witnessed two abrupt spikes in the sensor measurements.

Unlike classical Q-leaning processes which employ multiple offline training episodes before settling on suitable control strategies, herein the proposed approach showed successful outcomes following a single real-time learning episode as shown in Figures 6a, 9a. Additionally, the learning process exhibits capability to handle large mechanical systems. This explains the superior features of the proposed approach to follow desired trajectory reaching bounded error when some converged polices are utilized to run the system as shown in Figure 11a. The maximum observed absolute average tracking error was confined to under $1\ \mathrm{deg}$ for such mechanical actuation system (without overlooking the sensory error biasing).

When experimenting with real systems (as opposed to simulation), it is typical to notice differences between the reference and the actual signals. The differences may be due to several reasons, including noise, delay caused by digital filters, and backlashes. In this particular work, the differences also come from the coupling between the roll and pitch motions. The two were assumed to be decoupled in the first-order linear approximation of the system. However, in a real system, such as the one we adopted, they are never completely decoupled. Nevertheless, it is clear from the figures that the lag between the reference and the actual signals is bounded within an acceptable margin.

Remark 3: As explained earlier, the stability of the value iteration solution was investigated and proved, provided that the system under consideration is stabilizable and convex objective cost function is adopted. Figures 6a, 9a, and 11a emphasize the bounded stability features and the ability of the online learning mechanism to retrieve stability under significant mechanical disturbances, noise, and false readings. These figures show that the tracking errors are bounded and non-increasing. This is reassured by examining the actuation input signals shown in Figures 6b, 9b, and 11b.

VII. Conclusion

The challenging autonomous navigation of flexible-wing aircraft is solved by integrating analytical and computational

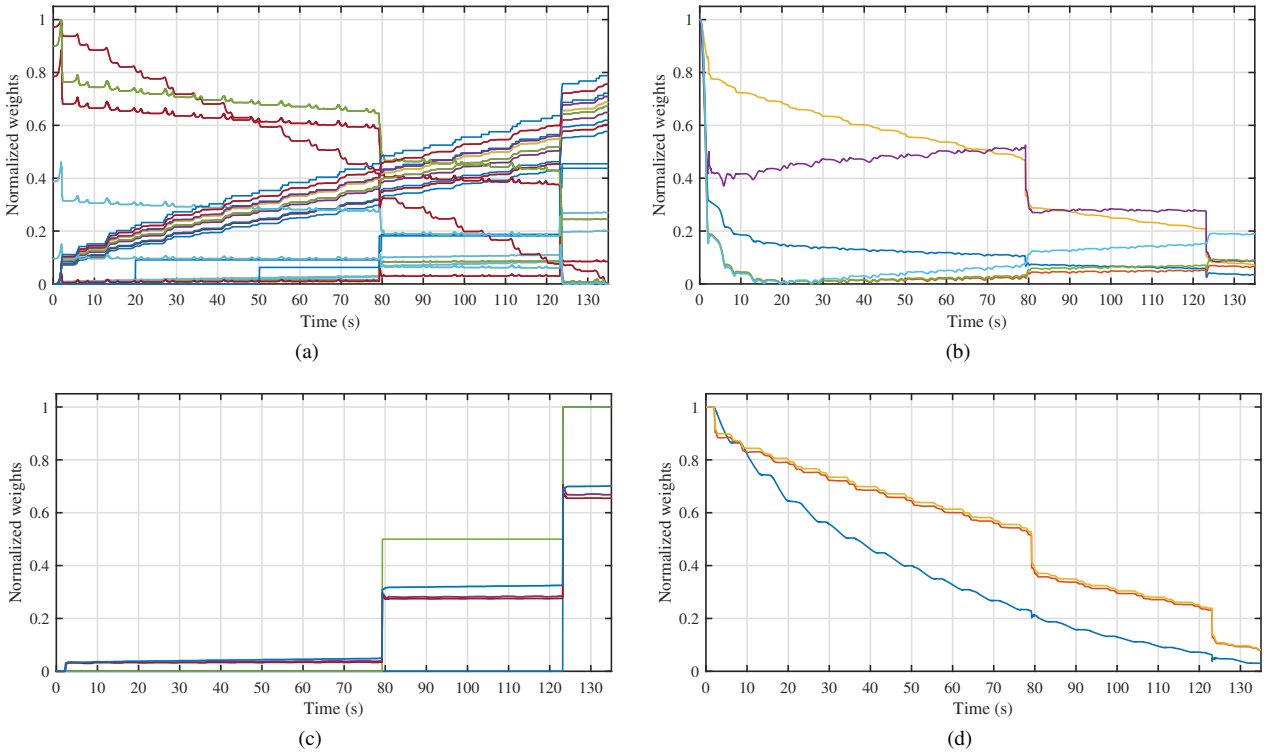


Fig. 7: Evolution of actor-critic weights during learning process: (a) tracking critic unit, (b) tracking actor unit, (c) stabilizing critic unit, (d) stabilizing actor unit.

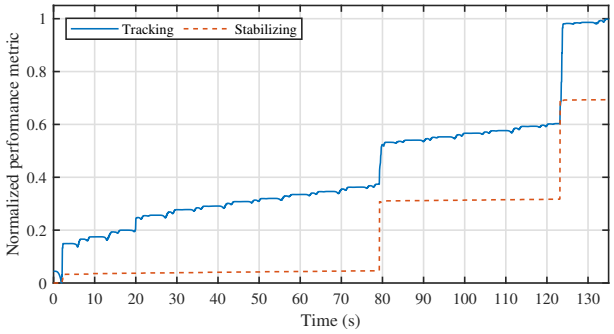


Fig. 8: Evolution of the policy evaluation metric.

solution platforms into an extensible experimental instrumentation and actuation incubator. The feedback loop receives the inertial measurements from standard measurement package Navio2 mounted on the wing system of an aircraft and decides, in real-time, on the best control strategies without acquiring any prior knowledge on the system's dynamical model. This enabled the integration of a powerful model-free control unit with a flexible and affordable measuring circuitry without over-complicating the sensory structures for such type of systems. The quality of the resultant systems depends mostly on the design of the model-free learning process rather than the precision of the sensors. This could be faced if a different model-based strategy is followed or even complicated augmented control structures are considered for this type of aircraft. This in turn, opens the door to generalize this approach for problems with similar interests. On another side, an online guided search mechanism based on a value iteration process is introduced where the learning process decides on the real-time control strategies based on the dynamic selection of the goals of the optimization problem. The experimental results coincided with the stated objectives, where the wing is subjected to severe disturbances without destabilizing the system.

REFERENCES

[1] Y. Ochi, "Modeling of flight dynamics and pilot's handling of a hang glider," in *AIAA Modeling and Simulation Technologies Conference*. American Institute of Aeronautics and Astronautics, 2017, pp. 1758–1776.

[2] M. V. Cook and E. A. Kilkenny, "An experimental investigation of the aerodynamics of the hang glider," in *Proceedings of an International Conference on Aerodynamics*, 1986.

[3] M. Abouheaf, W. Gueaieb, and F. Lewis, "Model-free gradient-based adaptive learning controller for an unmanned flexible wing aircraft," *Robotics*, vol. 7, no. 4, p. 66, 2018.

[4] F. Lewis, D. Vrabie, and V. Syrmos, *Optimal Control*, 3rd ed. New York, USA: John Wiley, 2012.

[5] E. Kilkenny, "Full scale wind tunnel tests on hang glider pilots," Cranfield Institute of Technology, College of Aeronautics, Department of Aerodynamics, Tech. Rep., 1984.

[6] E. A. Kilkenny, "An experimental study of the longitudinal aerodynamic and static stability characteristics of hang gliders," phdthesis, Cranfield University, Sep. 1986.

[7] D. Blake, "Modelling the aerodynamics, stability and control of the hang glider," Master's thesis, Centre for Aeronautics - Cranfield University, 1991.

[8] M. V. Cook, "The theory of the longitudinal static stability of the hang-glider," *The Aeronautical Journal*, vol. 98, no. 978, pp. 292–304, 1994.

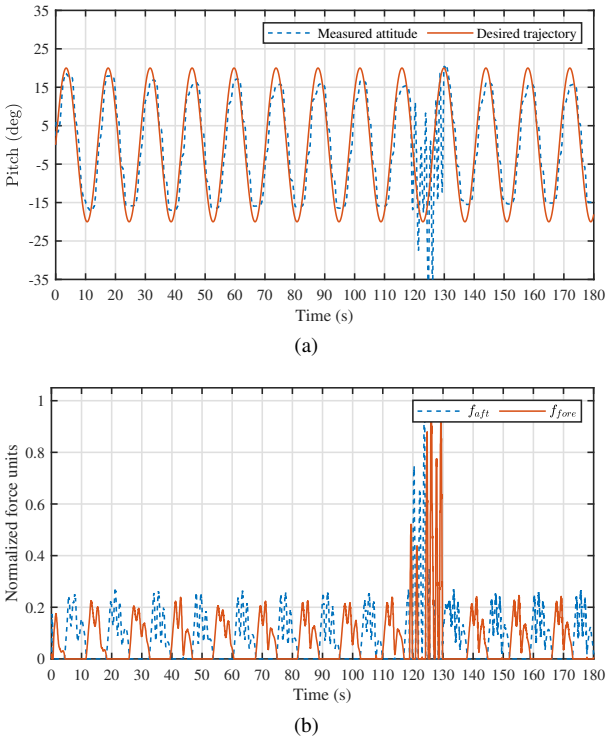


Fig. 9: Control performance while learning with a converged controller in the presence of mechanical disturbances: (a) measured vs. desired pitch attitude, (b) acting forces on the wing's keel, f_{fore} and f_{aft}.

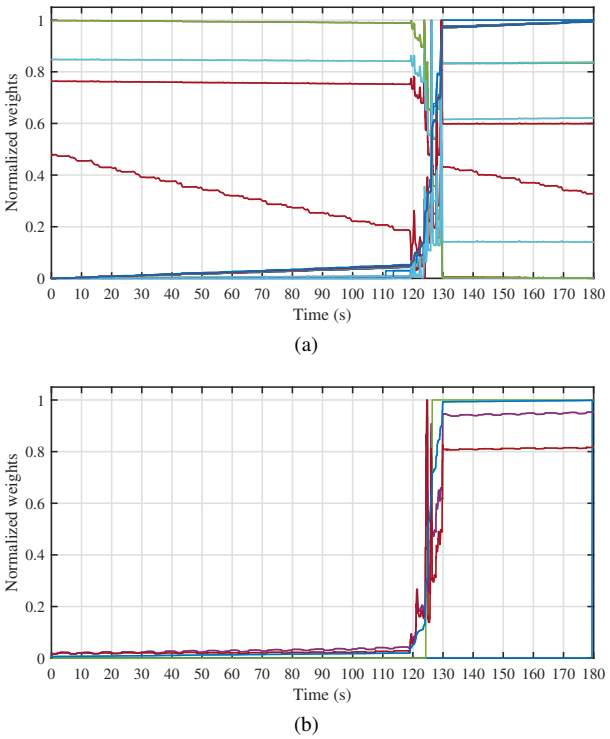


Fig. 10: Variations in critic weights in the face of mechanical disturbances: (a) tracking unit, (b) stabilizing unit.

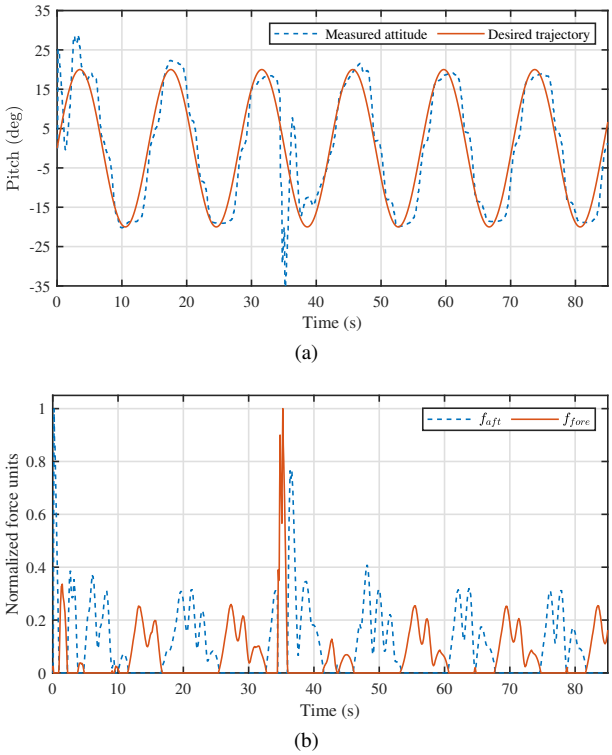


Fig. 11: Control performance with previously learned static control policies: (a) measured vs. desired pitch attitude, (b) acting forces on the wing's keel, f_{fore} and f_{aft}.

[9] Y. Ochi, "Modeling of the longitudinal dynamics of a hang glider," in *AIAA Modeling and Simulation Technologies Conference*. American Institute of Aeronautics and Astronautics, 2015, pp. 1591–1608.

[10] J. Sweeting, "An experimental investigation of hang glider stability," Master's thesis, College of Aeronautics, Cranfield University, 1981.

[11] M. Cook, *'Flight Dynamics Principles*. Butterworth-Heinemann, London, 2012.

[12] I. Kroo, *Aerodynamics, Aeroelasticity and Stability of Hang Gliders*. Stanford University, 1983.

[13] M. Spottiswoode, "A theoretical study of the lateral-directional dynamics, stability and control of the hang glider," mathesis, College of Aeronautics, Cranfield Institute of Technology, 2001.

[14] G. De Matteis, "Response of hang gliders to control," *The Aeronautical Journal*, vol. 94, no. 938, pp. 289–294, 1990.

[15] G. D. Matteis, "Dynamics of hang gliders," *Journal of Guidance Control and Dynamics*, vol. 14, no. 6, pp. 1145–1152, 1991.

[16] M. V. Cook and M. Spottiswoode, "Modelling the flight dynamics of the hang glider," *The Aeronautical Journal*, vol. 109, no. 1102, pp. I–XX, 2005.

[17] M. V. Cook, Ed., *Flight Dynamics Principles: A Linear Systems Approach to Aircraft Stability and Control*, 3rd ed., ser. Aerospace Engineering. Butterworth-Heinemann, 2013.

[18] M. Abouheaf, W. Gueaieb, and F. Lewis, "Model-free gradient-based adaptive learning controller for an unmanned flexible wing aircraft," *Robotics*, vol. 7, no. 4, p. 66, 2018.

[19] W. Gueaieb and M. S. Miah, "An intelligent mobile robot navigation technique using rfid technology," *IEEE Transactions on Instrumentation and Measurement*, vol. 57, no. 9, pp. 1908–1917, Sep. 2008.

[20] D. Esslinger, P. Rapp, S. Wiertz, H. Rendich, R. Marsden, O. Sawodny, and C. Tarín, "Accurate optoacoustic and inertial 3-d pose tracking of moving objects with particle filtering," *IEEE Transactions on Instrumentation and Measurement*, pp. 1–14, 2019.

[21] N. El-Sheimy, K. . Chiang, and A. Noureldin, "The utilization of artificial neural networks for multisensor system integration in navigation and positioning instruments," *IEEE Transactions on Instrumentation and Measurement*, vol. 55, no. 5, pp. 1606–1615, Oct 2006.

[22] X. Tao, J. Yi, Z. Pu, and T. Xiong, "State-estimator-integrated robust adaptive tracking control for flexible air-breathing hypersonic vehicle

with noisy measurements," *IEEE Transactions on Instrumentation and Measurement*, pp. 1–15, 2019.

[23] H. Deng, Q. Fu, Q. Quan, K. Yang, and K. Cai, "Indoor multi-camera based testbed for 3d tracking and control of uavs," *IEEE Transactions on Instrumentation and Measurement*, pp. 1–1, 2019.

[24] K. Das Sharma, A. Chatterjee, and A. Rakshit, "A pso–lyapunov hybrid stable adaptive fuzzy tracking control approach for vision-based robot navigation," *IEEE Transactions on Instrumentation and Measurement*, vol. 61, no. 7, pp. 1908–1914, July 2012.

[25] I. A. R. Ashokaraj, P. M. G. Silson, A. Tsourdos, and B. A. White, "Robust sensor-based navigation for mobile robots," *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 3, pp. 551–556, March 2009.

[26] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed., ser. Second. Massachusetts: MIT Press, 1998.

[27] P. Werbos, "Beyond regression: New tools for prediction and analysis in the behavior sciences," Ph.D. dissertation, Harvard University, 1974.

[28] P. J. Webros, "A menu of designs for reinforcement learning over time," in *Neural Networks for Control*, W. T. Miller, III, R. S. Sutton, and P. J. Werbos, Eds. Cambridge, MA, USA: MIT Press, 1990, pp. 67–95.

[29] M. Abouheaf and F. Lewis, "Approximate dynamic programming solutions of multi-agent graphical games using actor-critic network structures," in *International Joint Conference on Neural Networks (IJCNN)*, Aug. 2013, pp. 1–8.

[30] M. Abouheaf, F. Lewis, K. Vamvoudakis, S. Haesaert, and R. Babuska, "Multi-agent discrete-time graphical games and reinforcement learning solutions," *Automatica*, vol. 50, no. 12, pp. 3038–3053, 2014.

[31] M. I. Abouheaf and F. L. Lewis, "Multi-agent differential graphical games: Nash online adaptive learning solutions," in *52nd IEEE Conference on Decision and Control*, Dec 2013, pp. 5803–5809.

[32] R. Bellman, *Dynamic Programming*. Princeton University Press, 1957.

[33] A. Bryson, "Optimal control-1950 to 1985," *IEEE Control Systems*, vol. 16, no. 3, pp. 26–33, 1996.

[34] M. Abouheaf, F. Lewis, M. Mahmoud, and D. Mikulski, "Discrete-time dynamic graphical games: Model-free reinforcement learning solution," *Control Theory and Technology*, vol. 13, no. 1, pp. 55–69, 2015.

[35] M. Abouheaf and W. Gueaieb, "Multi-agent reinforcement learning approach based on reduced value function approximations," in *2017 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS)*, Oct 2017, pp. 111–116.

[36] T. Başar and G. J. Olsder, *Dynamic Non-cooperative Game Theory*, 2nd ed., ser. Classics in Applied Mathematics:. SIAM: Philadelphia, 1999.

[37] B. Widrow, N. K. Gupta, and S. Maitra, "Punish/reward: Learning with a critic in adaptive threshold systems," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-3, no. 5, pp. 455–465, 1973.

[38] P. Werbos, "Neural networks for control and system identification," in *28th Conference on Decision and Control*, Dec. 1989, pp. 260–265.

[39] P. J. Webros, "Neurocontrol and supervised learning: An overview and evaluation," in *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches*, D. A. White and D. A. Sofge, Eds. Van Nostrand Reinhold, Jun. 1992, pp. 65–89.

[40] M. Abouheaf and M. Mahmoud, "Policy iteration and coupled riccati solutions for dynamic graphical games," *International Journal of Digital Signals and Smart Systems*, vol. 1, no. 2, pp. 143–162, 2017.

[41] M. I. Abouheaf, F. L. Lewis, and M. S. Mahmoud, "Differential graphical games: Policy iteration solutions and coupled riccati formulation," in *2014 European Control Conference (ECC)*, June 2014, pp. 1594–1599.

[42] D. Bertsekas and J. Tsitsiklis, *Neuro-Dynamic Programming*, 1st ed. Massachusetts: Athena Scientific, 1996.

[43] L. Busoniu, R. Babuska, and B. D. Schutter, "A comprehensive survey of multi-agent reinforcement learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 2, pp. 156–172, 2008.

[44] M. Gomez Plaza, T. Martinez-Marin, S. Sanchez Prieto, and D. Meziat Luna, "Integration of cell-mapping and reinforcement-learning techniques for motion planning of car-like robots," *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 9, pp. 3094–3103, Sep. 2009.

[45] M. Abouheaf, N. Mailhot, and W. Gueaieb, "An online reinforcement learning wing-tracking mechanism for flexible wing aircraft," in *2019 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, June 2019, pp. 1–7.

[46] Emlid Limited. Emlid official open source documentation. [Online]. Available: https://github.com/emlid/emlid-docs

[47] Raspberry Pi Foundation. Raspberry Pi official open source documentation. [Online]. Available: https://github.com/raspberrypi/documentation

[48] ArduPilot Development Team and Community. ArduPilot open source autopilot. [Online]. Available: http://ardupilot.org/

[49] BYU MAGICC Lab, Provo, Utah. ROS Flight. [Online]. Available: https://rosflight.org/

[50] TDK Corp. InvenSense. TDK InvenSense MPU-9250 sensor documentation. [Online]. Available: https//www.invensense.com/products/motion-tracking-9-axis/mpu-9250/

[51] STMicroelectronics N.V. STM LSM9DS1 sensor documentation. [Online]. Available: https://www.st.com/en/mems-and-sensors/lsm9ds1.html

[52] Robotis Co. Dynamixel SDK. [Online]. Available: http://emanual.robotis.com/docs/en/software/dynamixel/dynamixel_sdk/overview/

[53] M. Abouheaf and F. Lewis, *Dynamic Graphical Games: Online Adaptive Learning Solutions Using Approximate Dynamic Programming*. World Scientific, 2014, ch. Chapter 1, pp. 1–48.

[54] T. Landelius and H. Knutsson, "Greedy adaptive critics for lqr problems: Convergence proofs," *Neural Computation - NECO*, 01 1996.