Online Actor-Critic-Based Adaptive Control for a Tailless Aircraft with Innovative Control Effectors

Shayan, K.; van Kampen, E.

DOI
10.2514/6.2021-0884

Publication date
2021

Document Version
Final published version

Published in
AIAA Scitech 2021 Forum

AIAA SciTech Forum
11–15 & 19–21 January 2021, VIRTUAL EVENT
AIAA Scitech 2021 Forum

10.2514/6.2021-0884

Online Actor-Critic-Based Adaptive Control for a Tailless Aircraft with Innovative Control Effectors

K. Shayan* and E. van Kampen.†
Delft University of Technology, 2629HS Delft, The Netherlands.

Conventional discrete reinforcement learning methods fail in providing satisfactory performance for online Flight Control Systems (FCSs). The lack of efficiency of the discrete controller in exploration for finding the optimal policy, the so-called problem of 'curse of dimensionality', results in an approach that is not suitable for online implementation. the Innovative Control Effector (ICE) aircraft is a highly maneuverable tailess aircraft with redundancy in its control effectors suite. This paper studies the experiments with adaptive critic design (ACDs) for longitudinal control of ICE aircraft. The online simulation results show the accuracy of the designed continuous RL controller in the attitude and altitude control of the aircraft using different sets of control effectors. The proposed approach also shows significant improvements in the tracking performance and control policy smoothness (e.g., compared to discrete methods).

Nomenclature

b	=	Wing span
c	=	Immediate cost value
c_α	=	Learning rates update coefficient
$\bar{c}$	=	Mean aerodynamic chord
C	=	Cost function constant
C_z, C_m	=	Longitudinal force and pitch moment coefficients
d_{TV}	=	Thrust vectoring arm
E	=	Error
F_x, F_y, F_z	=	Aerodynamic body forces
g	=	Gravitational acceleration
I_{yy}	=	Pitching moment of inertia
J	=	Value or cost-to-go function
l, m, n	=	Aerodynamic moments in body frame
M	=	Mach number
M_x, M_y, M_z	=	Aerodynamic moments
n	=	Number of hidden layer neurons
N	=	Number of inputs to the network
p, q, r	=	Pitch, roll and yaw angular rates
$\bar{q}$	=	Dynamic pressure
Q_c	=	Cost function state weight matrix
Q_m	=	Model error weight
$\boldsymbol{r}$	=	Immediate reward value
R	=	Cumulative reward
$\boldsymbol{R}$	=	Cost function control input weight matrix
S	=	Wing surface
t	=	Time

*MSc Student, Control and Simulation Division, Faculty of Aerospace Engineering, Kluyverweg 1, 2629HS Delf, The Netherlands, k.shayan@student.tudelft.nl. AIAA Member.
†Assistant Professor, Control and Simulation Division, Faculty of Aerospace Engineering, Kluyverweg 1, 2629HS Delft, The Netherlands, E.vanKampen@TUDelft.nl. AIAA Member.

T	=	Thrust force vector
u, v, w	=	Airspeed components
$\boldsymbol{u}$	=	Control input vector
V	=	True speed
$V(x)$	=	Value function
W	=	Neural networks weights
W_0	=	Initial weights
W^h, W^o	=	Hidden and output layer weights
W_c, W_a	=	Critic and actor networks weights
x	=	State vector

Subscripts

$\boldsymbol{\alpha}$	=	Angle of attack
$\boldsymbol{\alpha}$	=	Learning rate
δ	=	Control effectors deflection
γ	=	Discount factor
$\pi(s)$	=	Policy function
$\boldsymbol{\phi}, \theta, \psi$	=	Roll, pitch and yaw angles
$\boldsymbol{\phi}, \boldsymbol{\sigma}$	=	Activation Function
ρ	=	Atmospheric pressure

I. Introduction

The aim for automation in aircraft control systems calls for innovative approaches that are only possible with a revolution in the current Flight Control System (FCS) designs. Based on a recent study, loss of control (LOC) is the main reason for jet aircraft fatalities [1]. A genuinely autonomous control system can adjust its behavior after unexpected changes occurred in the system, so show adaptability features. With a combat aircraft, examples of unknown and dangerous situations can be combat damage affecting control surfaces or asymmetrical weight distribution after missed/faulty weapons release. As a result, self-learning and adaptability in unknown conditions are one of the most critical challenges in control systems designed for automation. Adaptive control is one of the most promising approaches in automatic control for the prevention of LOC. Between adaptive control methods, reinforcement learning due to its learning abilities attracts many researchers in this field.

Technological improvements in aerospace systems have spurred innovative designs both in the civilian and in the military sectors. The innovative control effector aircraft (ICE) is state-of-the-art in the design of the future fighter aircraft. The low-observability that resulted from the tailless configuration and the high maneuverability reflected in its unconventional control suite make ICE aircraft unique in its design. Maneuverability, stealth, and resilience are the primary attributes of modern-day fighter combat aircraft that also provoke the ICE aircraft design control objectives ([2–5]). A large number of highly non-linear and constantly interacting control effectors in tail-less ICE configuration, while satisfying the design requirements of the modern fighters, introduce a challenging control problem. The need for a continuous controller that ensures the online, autonomous, and contiguous intervention of FCS (particularly for failures) is then necessary [6]. The desire controller should use the combination of the control effectors in their control limits, to avoid extensive drag and effector utilization in achieving the optimal control objective. Also, the control system should be suitable for adaptability, independent of the global model of the system. Therefore, quick online adaptation to the possible changes is required.

Reinforcement Learning (RL) is the idea of active decision-making and learning by interacting with the environment. RL is relatively new to the field of Guidance, Navigation, and Control (GNC) of aerospace systems. Still, RL solutions have shown promising results in model-free adaptive optimal control problems for both discrete and continuous systems [7–12]. Adaptive optimal controllers are usually designed to find optimal solutions for user-defined performance equations. They use system identification methods to find the global model parameters and then use the model to solve optimal equations. Instead, RL controllers are introduced for adaptive control of the real-world complex systems, when a model of the system is generally unavailable, and therefore, model-based approaches become invalid. The online

adaptability is achieved by being independent of the global model updates, possible by goal-oriented characteristics of RL. Instead of instructing the agent on how to do things from the model, the agent comes up with the suitable control command using the end-goal alone. RL can also be seen as an optimal control approach. The current optimal control methods frequently require complete knowledge of the system and are generally designed to solve an optimization problem offline, e.g., by solving Hamilton-Jacobi-Bellman (HJB) equations. Unlike these controllers, the recently developed RL optimization methods allows the design of controllers that solve optimal equations independent of the model of the system dynamics [13].

RL methods are studied for discrete and continuous dynamical systems. Using the tabular RL methods for control of the continuous systems exhibits low-performance characteristics, when compared to current model-based control approaches [10]. Introducing approximation methods for these methods overcome the deficiencies of the discrete RL while keeping the model-free, and self-learning adaptability characteristics. Hence, approximate RL is conceived as a suitable approach for the development of an automated FCS, as discussed by [11, 12, 14]. The offline learned RL agent with the optimal policy can be used for online and automatic adaptability of the aircraft to, e.g., unknown dynamics changes. Recently, the performance of continuous RL methods has improved with the extension of optimally for RL methods of Adaptive Critic Designs (ACDs) categorized within the Actor-Critic (AC) framework. The technique integrates dynamic programming, temporal-difference (TD) learning, and function approximation to provide model-free and online learning in the continuous domain.

A discrete RL-based controller using the Q-learning approach was developed for the ICE model to fulfill an objective of altitude control [10]. The study shows that the performance of the discrete RL is less than that of model-based approaches. The limitations that come with the discretization of the state and action space result in the discrete RL controllers deficiency. Van Kampen et al. [15] focused on the self-learning and the adaptability of continuous RL methods. He showed their advantages in the field of re-configurable flight control, where accidental changes in the plant dynamics require the control system adaptation. He describes the prominence of RL comparing to supervised learning in being free of prior instructions for achieving a specific goal and its ability to come up with policies without restrictions of a structured model. By comparing two continuous ACD methods of HDP and Action-Dependent HDP (ADHDP), he concluded that the continuous domain brings a broader scope of applications for RL (compared to the discrete algorithms). The same study shows the high performance of the HDP approach in terms of success ratio and adaptation speed (compared to action-dependent approach).

This paper propose the implementation of one of the promising approaches of adaptive critic designs, the so-called heuristic dynamic programming method, to overcome the deficiencies of the discrete reinforcement learning control. Online, model-free, and self-learning are the main requirements for the development of an automated FCS that can be achieved by an RL controller. By continuous, it is meant here that the system is continuous in states and actions but not necessarily in time (e.g., discrete-time and continuous-time systems). Although real systems are continuous in time, however, their measurements are obtained discrete. With the advance HDP architecture, the RL controller can achieve the (near-) optimal policy with no prior knowledge about the complex dynamics of a system such as ICE aircraft. The convergence with the extension of optimally for RL methods in a continuous space is assured in literature by [16–18]. This makes the HDP architecture a suitable choice for designing an online, model-free, and adaptive controller for the ICE aircraft. In the following, by all effectors it is meant that all 12 effectors of the ICE aircraft, except for the thrust vectoring.

The main contributions of this article are presenting the capabilities of continuous RL methods in aerospace applications and automation research for flight control systems related to innovative designs. The use of these methods for ICE aircraft provides the RL research community with an implementation of the RL methods for an innovative over-actuated system and gives novel insights to the developers of the ICE automatic control systems.

The layout of this paper includes the elaboration of the following topics. The introduction of the RL and a background on adaptive and optimal flight controllers and the location of RL in these concepts are given in section II. The ICE model and configuration are described in section III. section IV includes the development of the ACD-based controller for the ICE aircraft. The simulation set-up and the results for the application of the approximate reinforcement learning controller for the ICE aircraft model are presented by section VI. Finally, the conclusion of this paper and the next steps for future studies are given by section VII.

II. The Actor-Critic Reinforcement Learning Problem

In most of the reinforcement learning problems, a state is measured, and an action corresponding to that state is chosen. Next, a transition to the next state is measured, and the reward/cost value restores for such a shift. The long-term reward/cost values are called value function. Such a formulation of the problem is based on the Markov Decision Process (MDP) modeling. The framework includes two processes of policy evaluation (value function improvement) and policy improvement (policy function update) to reach the optimal policy. Fig.1 shows a schematic of such an interaction. RL, in the context of control, involves the change of the controller's strategy based on its experience to maximize long-term rewards or minimize a cost function. In case of constraints for the control limits, the designer defines the boundaries for the learning controller. To reach optimality in an RL framework, the importance of exploration and its trade-off with exploitation should not be ignored. The smart design of the reward or cost function also plays a vital role in convergence to the correct optimal policy.

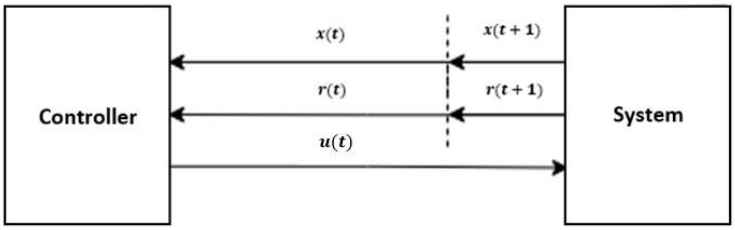


Fig. 1 A schematic diagram of the controller and system interaction in a RL problem

Within continuous RL approaches, AC methods are emphasized more for non-stationary settings that can deal with continuous state and action spaces. AC combines learning in policy space with simultaneous value function approximations that result in algorithms that are proved for convergence. An AC framework integrates three fundamental aspects and classes of RL, Dynamic Programming (DP), Temporal-Difference (TD) learning, and Function Approximation (FA). By combining the DP method with incremental TD approach, possible within approximate solutions of FAs, the AC design can overcome the deficiencies of each technique and provide an architecture suitable for online and model-free learning of large-scale processes. This structure also has shown promising results for optimal adaptive control of non-linear systems by exploiting a variety of sophisticated FA methods. In the following this method is described in more details.

A. Actor-Critic Architecture

Actor-critic methods are a group of continuous RL approaches, in which classification is done if the critic element is approximating the expected total future reward/cost or the value function, and/or the derivative of the value function. Such a scheme abandoned the knowledge of the dynamics, by approximation of the Hamilton-Jacobian-Bellman (HJB) equation. The state information obtained by the critic will be used for determining the value of the state and change the internal parameters of the critic so that this knowledge will be stored compactly within the critic parameters. The actor represents the policy π of the controller by providing actions based on the current information of the state. The optimal policy is achieved then by making the approximated value function $\hat{J}(t)$ to a goal value $J^*(t)$ using the actor training error. Fig.2 represents the schematic of the AC framework.

AC methods of approximation can be seen as an extension of Generalized Policy Iteration (GPI) [19]) to the continuous state and action spaces. In a GPI framework, the critic policy evaluation step is not performed entirely, and the current estimation of the value function would be updated towards the value of the policy [20]. However, in an actor-critic structure, both critic and actor updates would be done completely and simultaneously at each time step. Since the TD error is used as the update rule at each time step, it is not needed to wait for the end of each episode, which makes the AC structures suitable for online application. Also, no greedy selection (so selecting based on an optimization process) is performed, and it is the policy itself that determines the direction.

Within AC methods, HDP algorithms are one of the most used and popular architectures. Their simplicity in implementation while showing high performance in adaptation and convergence rate besides their high success ratio (for example, when compared with other AC methods such as ADHDP methods [15]) makes them a suitable choice for

adaptive control of complex nonlinear systems. In the below a more detail of this framework is described.

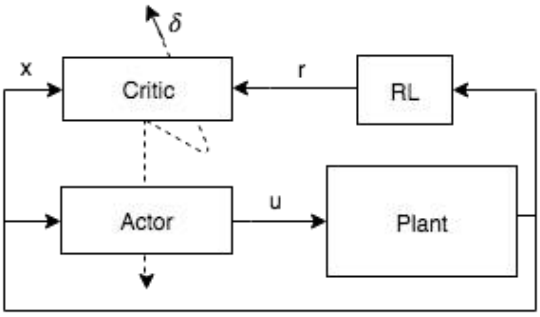


Fig. 2 Schematic of the actor-critic structure

B. Heuristic Dynamic Programming

In a HDP framework, the critic uses only the incoming state information for value-function approximation, while the actor network is providing the control input. There should be an information exchange between the two networks. Since the actor needs the information updates in the critic as its output indirectly influences also the critic output. Therefore, in such cases, an internal model of the plant is approximated and will be updated along with the critic and actor networks to identify and reflect the changes in the real plant [21]. The optimal value function is usually unknown and, therefore, cannot be used as the target value for the learning, although the recursive characteristics of it help the network move toward the optimal value function. A schematic of an HDP architecture is given in Fig.3.

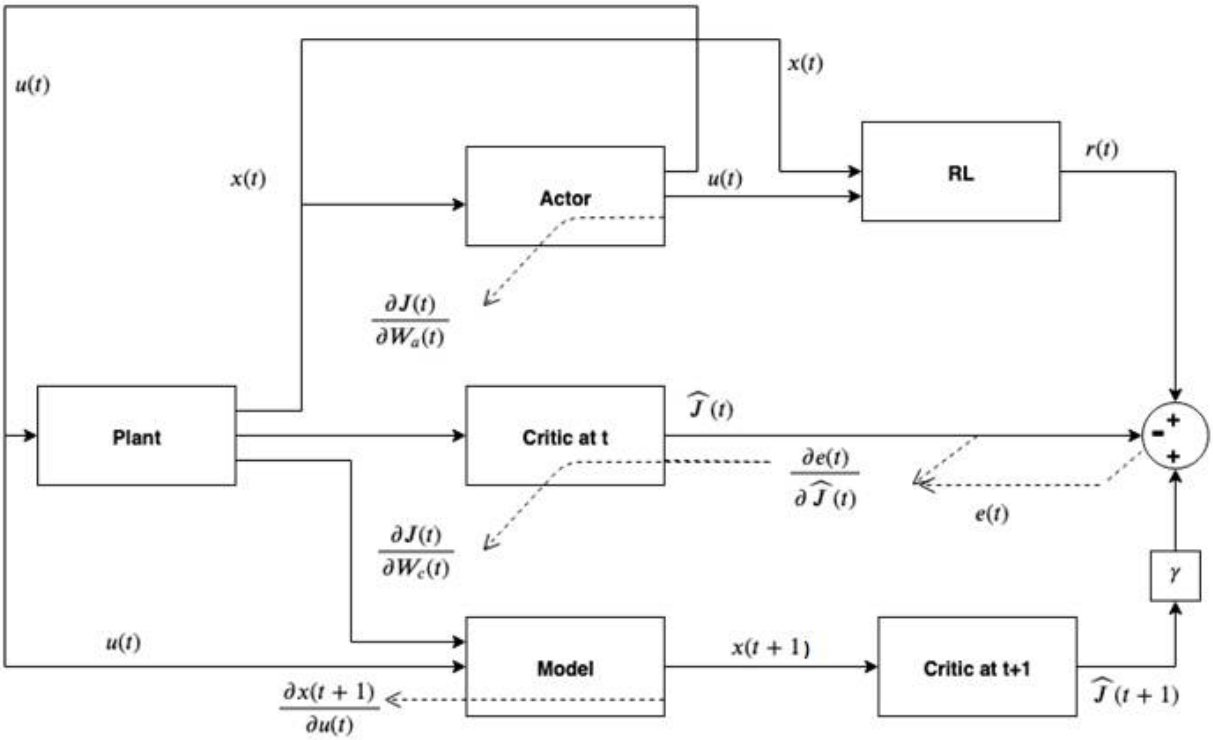


Fig. 3 HDP scheme block diagrams and updates based on [19]

For HDP, an indirect path from the actor output to the approximated plant dynamics and then through the critic will approximate the value-function, and through the same path, the actor network will be updated in the direction to achieve the zero value for actor error $E_a(t)$ [15]. While the critic is used for approximation of the value function, the unbalance in the Bellman's equation will be used as the learning error for the actor that will need to be minimized. The actor would then lead the system to the regions within the state space that the value function approximation of the critic has high or low values depending on the way the reward or cost function is defined.

It is important to note that the only place where the plant dynamics will be used is in the back-propagation step for the actor network. Also, in such an approach, there is no direct connection between the actor output to the critic network, and therefore, the critic is estimating the value-function now only based on the state information. Thus, the task of the critic is simplified, and the complexity is now partially moved to the plant dynamics. In the HDP framework, the plant network is independent, and it would influence the actor and critic networks. In the following, the actor, critic and model networks are defined for the HDP framework with this assumption that both networks are approximated with two-layer Convolutional Artificial Neural networks (CANNs) with one hidden layer. It is assumed that the networks are updated using the gradient descent method, and a general input-output definition is given that will be used later in this paper for the development of the controller.

1. Critic Network

The input to the critic network is the difference between the measured state $x(t)$ and the reference value $x_{ref}(t)$ so the tracking error $e(t)$ defined by Eq. (1).

$$e(t) = x(t) - x_{ref}(t) \tag{1}$$

The output is the estimated value function $\widehat{J}(X(t))$. The network is defined then by:

$$\begin{aligned} \widehat{J}(t) &= W_c^o(t)\boldsymbol{\phi}(W_c^h e(t)) \\ &= \sum_{i=1}^{n_c} \widehat{w}_{ci}^o(t)\boldsymbol{\phi}_i\Big(\sum_{j=1}^{N} \widehat{w}_{cij}^h(t)e_j(t)\Big) \end{aligned} \tag{2}$$

where $\boldsymbol{\phi}(t) = [\boldsymbol{\phi}_1(t), ..., \boldsymbol{\phi}_{n_c}(t)]^T \in R^{n_c}$ is the hidden layer activation function vector (e.g. tanh), W_c^h is the output layer weights defined by as:

$$W_c^h = \begin{bmatrix} w_{c11}^h & \cdots & w_{c1(N)}^h \\ \vdots & \vdots & \vdots \\ w_{cn_c1}^h & \cdots & w_{cn_c(N)}^h \end{bmatrix} \in R^{n_c \times N} \tag{3}$$

where n_c is the number of the neurons for the hidden layer and w_{cij}^h denotes the weight from the jth input value to the ith hidden neuron. The output layer weights are defined as:

$$W_c^o = [w_{c1}^o, w_{c2}^o, ..., w_{cn_c}^o] \in R^{1 \times n_c} \tag{4}$$

where w_{ci}^o is the weight from ith hidden layer to the output layer.

2. Actor Network

This NN approximates the control input for the nonlinear tracking problem. The data to the actor network is the same as the critic, and the output is the control command given as:

$$\widehat{u}(t, W_a^h, W_a^o) = [\widehat{u}_1, ..., \widehat{u}_m] \tag{5}$$

where

$$\begin{aligned}\widehat{u}(t) &= W_a^o(t)\boldsymbol{\sigma}(W_a^h(t)e(t)) \\ &= \sum_{k=1}^{n_a} \widehat{w}_{a_k}^o(t)\boldsymbol{\sigma}_k(\sum_{l=1}^{N} \widehat{w}_{a_{kl}}^h(t)e_l(t))\end{aligned} \tag{6}$$

where $\boldsymbol{\sigma}(t) = [\boldsymbol{\sigma}_1(t), ..., \boldsymbol{\sigma}_{n_a}(t)]^T \in R^{n_a}$ is the hidden layer activation function vector, n_a is the number of neurons for the hidden layer and $k = 1, ..., m$ is the kth output-layer. The weights matrix for hidden layer is defined as:

$$W_a^h = \begin{bmatrix} w_{a11}^h & \cdots & w_{a1(N)}^h \\ \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots \\ w_{an_a1}^h & \cdots & w_{an_a(N)}^h \end{bmatrix} \in R^{n_a \times N} \tag{7}$$

where $w_{a_{kl}}^h$ is the weight connecting kth input to the kth hidden neurons. The matrix of the weights between the hidden and output layer is defined as:

$$W_a^o = \begin{bmatrix} w_{a11}^o & \cdots & w_{a1(m)}^o \\ \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots \\ w_{an_a1}^o & \cdots & w_{an_a(m)}^o \end{bmatrix} \in R^{n_a \times m} \tag{8}$$

where $w_{a_k}^o$ is the weight relating kth hidden layer to the output layer.

C. Learning Rules for Actor and Critic Networks

The gradient descent rules usually are used for updating the critic and actor network weights. The direction for the critic network is toward minimizing bellman error, and for the actor, the objective is to minimize the value function estimation. In the following, the update rules for the actor and the critic are illustrated separately.

1. Critic Training

If the Bellman prediction error is defined as:

$$e_c(t) = c(t) + \gamma\widehat{J}(t+1) - \widehat{J}(t) \tag{9}$$

where the reinforcement signal is defined by:

$$c(t) = e(t)^T Q_c e(t) \tag{10}$$

then the Bellman error will be defined as:

$$E_c(t) = \frac{1}{2}e_c^2(t) \tag{11}$$

The NN weights are then tuned by the gradient descent algorithm as:

$$\widehat{W}_{t+1} = \widehat{W}_t + \Delta W, \qquad \Delta W = -\boldsymbol{\alpha}\frac{\partial E}{\partial W_t} \tag{12}$$

where $\boldsymbol{\alpha}$ is the learning rates. For the critic network, which approximates the value function and is the function of the tracking error, the tuning laws apply as:

$$\hat{w}_{c_{ij}}^h(t+1) = \hat{w}_{c_{ij}}^h(t) - \boldsymbol{\alpha}_c\frac{\partial E_c(t)}{\partial \hat{w}_{c_{ij}}^h(t)} \tag{13}$$

$$\hat{w}^o_{c_i}(t+1) = \hat{w}^o_{c_i}(t) - \boldsymbol{\alpha}_c \frac{\partial E_c(t)}{\partial \hat{w}^o_{c_i}(t)} \tag{14}$$

Applying the chain rule for gradient descent approach for the hidden layer gives:

$$\frac{\partial E_c(t)}{\partial \hat{w}^h_{c\,ij}} = \frac{\partial E_c(t)}{\partial e_c(t)} \frac{\partial e_c(t)}{\partial \hat{J}(t)} \frac{\partial \hat{J}(t)}{\partial \boldsymbol{\phi}_{c_i}(t)} \frac{\partial \boldsymbol{\phi}_{c_i}(t)}{\partial \hat{w}^h_{c\,ij}} \tag{15}$$

same for output layer gives:

$$\frac{\partial E_c(t)}{\partial \hat{w}^o_{c_i}} = \frac{\partial E_c(t)}{\partial \hat{J}(t)} \frac{\partial \hat{J}(t)}{\partial \hat{w}^o_{c_i}(t)} \tag{16}$$

The convergence for this update rule can be tricky due to the nature of the approximation with this gradient descent method.

2. Actor Training

The objective of the actor network is the minimization of the approximated value function. Therefore, the difference between the evaluated function approximation $J(t)$ and the pre-defined value-function goal $J^*(t)$ (e.g. designed by expert knowledge considering the chosen cost function $c(x)$) is used for defining the actor error $e_a(t)$ as:

$$e_a(t) = J(t) - J^*(t) \tag{17}$$

The square actor error is defined as:

$$E_a(t) = \frac{1}{2} e_a^2(t) \tag{18}$$

The same gradient descent tuning laws can be applied for the actor network as:

$$\hat{w}^h_{a_{kl}}(t+1) = \hat{w}^h_{a_{kl}}(t) - \boldsymbol{\alpha}_a \frac{\partial E_a(t)}{\partial \hat{w}^h_{a_{kl}}(t)} \tag{19}$$

$$\hat{w}^o_{a_k}(t+1) = \hat{w}^o_{a_k}(t) - \boldsymbol{\alpha}_a \frac{\partial E_a(t)}{\partial \hat{w}^o_{a_k}(t)} \tag{20}$$

where $\boldsymbol{\alpha}_a$ is the actor learning rate. The chain rule for actor gradient descent algorithms is derived for hidden layer as:

$$\frac{\partial E_a(t)}{\partial \hat{w}^h_{a_{kl}}(t)} = \frac{\partial E_a(t)}{\partial e_a(t)} \frac{\partial e_a(t)}{\partial \hat{J}(t)} \frac{\partial \hat{J}(t)}{\partial x(t)} \frac{\partial x(t)}{\partial u(t)} \frac{\partial u(t)}{\partial \hat{w}^h_{a_{kl}}(t)} \tag{21}$$

and for the output layer weights:

$$\frac{\partial E_a(t)}{\partial \hat{w}^o_{a_k}(t)} = \frac{\partial E_a(t)}{\partial \hat{J}(t)} \frac{\partial \hat{J}(t)}{\partial \hat{w}^o_{a_k}(t)} \tag{22}$$

The simultaneous update of the actor and the critic make convergence guarantee more challenging to prove [22]. To ensure sufficient exploration within the states of the system and convergence to the global optimum, the control input signal should satisfy the persistence of excitation (PE) condition. Some of the PE methods can be a simple white noise, a filtered noise signal or a sinusoidal sweep added to the control input as an action modifier.

The approach proposed above can be seen as a generic actor-critic approach in solving optimal tracking control problem [15, 22, 23]. The convergence of actor-critic methods is provided in [24, 25].

3. Model Network

To derive the control value at the next time step, in HDP control approaches, a model of the system, is required. This model can be updated at each time step based on the target states and the current values for the states. Therefore, it adopts incrementally and is only valid locally. If it is assumed that a third neural network is used for the plant approximation, the input to the plant is the current state estimation, and the action derived by the actor and the output is the estimated states. The update rule for the model network is based on the error given by:

$$e_m(t) = x_{target}(t) - x(t) \tag{23}$$

The square of the model error is given by:

$$E_m(t) = \frac{1}{2} Q_m e_m^2(t) \tag{24}$$

where Q_m is the adjustment matrix for the model. The update rules for the model then will be the same as the critic network, and to avoid duplication will not be discussed here.

III. The Innovative Control Effector Aircraft Model

The continuous RL method of HDP was used to develop a flight control system for the ICE aircraft in a simulation framework. Lockheed Martin designs this highly maneuverable tailless aircraft with an unconventional control suite configuration, which results in substantial control challenges. In this section, details about the ICE configuration and model are given.

A. ICE Aircraft Control Objectives and Challenges

The ICE aircraft designed control objectives are mainly based upon maneuverability, stealth, and resilience. The ICE aircraft configuration is designed with Relaxed Static Stability (RSS) in both pitch and yaw axes to achieve high maneuverability. From 1974, the next generation of highly maneuverable fighters introduced that take advantage of the inherently aerodynamic instability to get extraordinary agility while relying on FCC to stabilize the aircraft for the human pilot. Flying an aircraft with RSS is achieved by fitting stability augmentation devices and with the constant intervention of the FCS to maintain the aircraft level. Therefore, the continuous intervention of an automated FSC is also critical and vital for ICE design. ICE aircraft is designed without horizontal and vertical tails. In a tail-less aircraft, other advantages can be obtained apart from low RCS, as the flying-wing configuration is very efficient in producing lift, allowing for overall savings in weight and cost. Without the tail, the stabilizing moments in pitch and yaw for ICE have to be generated with the proper actuation of the redundant control surfaces, a challenge defined in the context of the ICE development objectives [3]. The ICE aircraft is also meant to be categorized as Unmanned Air Vehicles (UAVs). Therefore, improvements in automatic adaptive control systems are essential for this innovative design. In the occurrence of accidents such as the signal loss and other failures, adaptability and the ability to control the aircraft remain completely to the smart FCC on-board.

B. ICE Aircraft Control Suite

The land-based baseline configuration of the ICE simulation model is described by [26] as a supersonic single-engine, 65 deg sweep delta flying wing and a multi-role fighter. The ICE aircraft has highly redundant 13 multi-axis control effectors [26]. A schematic of the control suits for ICE is presented by Fig.4. The control effectors include Leading-edge flaps (LEF), all moving wing tips (AMT) and spoiler-slot deflectors (SSD) which are mainly used for lateral-directional control, the pitch-flaps (PF) to provide longitudinal control power and elevons and multi-axis thrust vectoring (MTV) that are used for both symmetric and asymmetric movements. The elevons that can deflect independently and asymmetrically for roll control and also contribute to pitch control by symmetric deflections. Two pairs of LEfs use for lateral-directional control in high angle of attacks (AOAs) and low speeds. AMTs are the primary sources for yaw control in high angle of attacks and can maintain their effectiveness for wide ranges in the speed envelope of the aircraft. The SSDs are more useful for yaw augmentations in a lower angle of attacks.

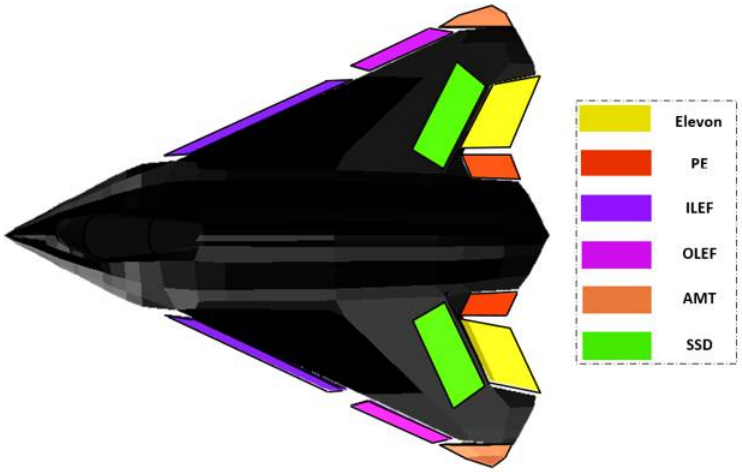


Fig. 4 ICE aircraft control suite

Integrating thrust vectoring power to the control power substantially improves the potential of the aircraft for unlimited maneuverability. MTV is designed for both pitch, and yaw thrust vectoring for ICE aircraft and also allows the full utilization of the roll control power by integration with SSDs and by coordinating the yaw control. In Table 1, the list of the control suites as well as their position and rate limits are described.

Table 1 ICE aircraft control suite specification

Effector	Position limits [deg]	Rate limits [deg]	Positive direction
Elevon	[-30, 30]	150	Downwards
PF	[-30, 30]	150	Downwards
ILEF	[0, 40]	40	Downwards
OLEF	[-40, 40]	40	Downwards
AMT	[0,60]	150	Downwards
SSD	[0, 60]	150	Upwards
MTV	[-15,15]	150	[-]

C. Three Degree of Freedom Aerodynamic Model

A spline-based model developed by [27] for ICE aircraft is used for calculating the force and moment coefficients in the MATLAB environment. In the spline model the aerodynamic coefficients are identified with a spline approximation as a function of the $(\alpha, M, \beta, V, p, q, r, T, u)$, where $V(ft/s)$ is the true airspeed, p, q and r (rad/s) are the body rotation rates in x, y and z axes, $T(lbs)$ is the thrust and u is the input vector given by:

$$\begin{aligned}\mathbf{u} =&[\delta_{ILEF_L}, \delta_{OLEF_L}, \delta_{AMT_L}, \delta_{E_L}, \delta_{SSD_L}, \delta_{PF},\\ &\delta_{ILEF_R}, \delta_{OLEF_R}, \delta_{AMT_R}, \delta_{E_R}, \delta_{SSD_R}, \delta_{pTV}, \delta_{yTV}]\end{aligned} \tag{25}$$

where L and R denotes to the left and right effector and pTV and yTV corresponds to the thrust vectoring for pitch and yaw movements. As this study only considers longitudinal motions of the aircraft, only the equations correspond to the symmetrical moves are described here. Also, to simplify, the control surfaces deflections are assumed to be symmetric and equal. The yaw thrust vectoring value, too, is set to zero for the whole simulation. Including the thrust vector projection on the body frame $\mathcal{F}^b$ in the derivation of the aerodynamic forces and moments, the aerodynamic coefficients are obtained by the model and the longitudinal forces and moments are computed by:

$$F_x = Tcos(\delta_{pTV})cos(\delta_y TV) - \frac{1}{2}\rho V^2 S C_{F_x} \tag{26}$$

$$F_z = -Tsin(\delta_{pTV})cos(\delta_y TV) - \frac{1}{2}\rho V^2 S C_{F_z} \tag{27}$$

$$M_y = \frac{1}{2}\rho V^2 S\bar{c} C_{M_y} - Td_{TV} sin(\delta_{pTV})cos(\delta_{yTV}) \tag{28}$$

where d_{TV} is the thrust vectoring arm from aircraft center of gravity and the forces and moments are in reference body frame $\mathcal{F}^b$ in $[lbf]$ and $[lbf.ft]$, respectively. Euler equation is used for computing the changes in the system dynamics. The atmospheric changes and conditions are described based on the International Standard Atmosphere (ISA).

IV. The Adaptive Critic Design-Based Flight Control System

The ICE aircraft spline model is used as the environment. The ICE environment is implemented using MATLAB script to simulate the dynamics of the designed aircraft. The thrust is controlled by an internal Proportional and Integral (PI) speed controller in the simulation model. The lateral-directional movements of the aircraft are not considered, and only longitudinal changes in the dynamics are considered for the angle of attack track. Also, it is assumed that the perturbations in the longitudinal and lateral direction are independent, and their variables are not coupled, which is valid for an aircraft of symmetric shape under a steady equilibrium condition. The state vector for the longitudinal control of the ICE aircraft is defined as Eq. (29), including the position, the pitch rate, airspeed, aerodynamic and Euler angles.

$$x^{ICE} = [\dot{x} \;\; \dot{z} \;\; x \;\; z \;\; u \;\; w \;\; \alpha \;\; \theta \;\; q \;\; V \;\; M \;\; \rho \;\; \bar{q}]^T \tag{29}$$

A longitudinal nonlinear attitude control system based on the available spline aerodynamic model with a heuristic dynamic programming RL controller was designed for the ICE aircraft. Such an architecture for the control system is known as Adaptive Critic Designs (ACDs). The HDP framework discussed earlier is used to perform a reference tracking control problem for the ICE aircraft. The aim is to explore the tracking performance of the proposed HDP framework for the over-actuated system. First, the angle of attack tracking problem is investigated with multiple benchmarks starting with one effector (e.g. pitch flaps) as the control inputs available and adding more effectors till all the effectors (except the thrust vectoring) become available for the system. The performance of the controller in the tracking task for the inner loop attitude control and its exploitation of the available control inputs with its effect on the tracking performance will be explored both qualitatively and quantitatively. As only longitudinal control is considered, the control effectors' deflections are assumed to be symmetric to avoid the lateral-directional movements.

A. Angle of Attack Tracking Problem

The ACD controller is implemented for the inner control loop of longitudinal body-frame angle control, so the angle of attack of the aircraft. The RL controller uses the pitch flap control surfaces first to provide angular rate control in the longitudinal axis. There is no direct control for the angular rates. For the model identification, however, both the angular positions and velocity are fed to the network. In this way, the RL controller learns the direct effect of the actuators on the angular position and rates and the aircraft dynamics can be learned quickly. This RL controller can then be used to provide a higher level of control objective (e.g. altitude control) using other control techniques (e.g. PID controllers or nonlinear dynamic inversion techniques), as will be shown by V.

Fig.5 gives a general overview of the inner control framework, including the higher-level overview of the continuous RL controller. As shown, the architecture consists of the designed RL controller and the ICE plant. In the implementation of the HDP method for the ICE aircraft, the same framework, as described by section II, is used. The controller includes three main entities, namely the critic, actor and the model of the system. In the above framework, the actor learns the control policy and develop the control law while the critic updates the value function and gives the direction to the actor towards the optimal policy. To perform back-propagation of the actor network, online system identification is used to provide the derivatives and state predictions. Three neural networks were used for the critic, actor and model networks with gradient descent method used for propagation of the information as shown by dotted lines in Fig.5.

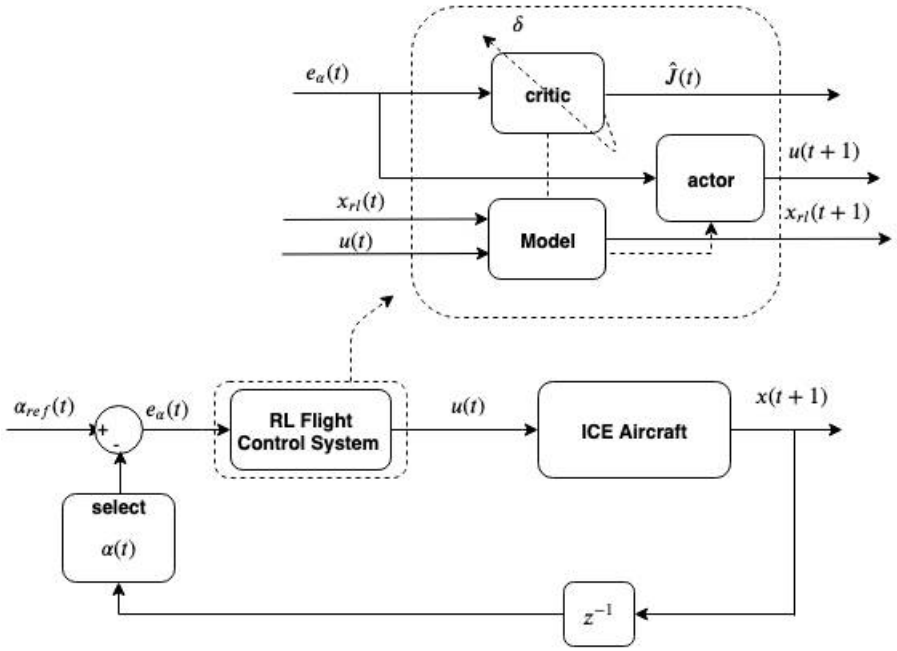


Fig. 5 Layout of the control loop including the reinforcement learning controller for attitude control of the ICE aircraft

The angle of attack response is selectively used to find the error from the response and the reference signal to generate the inputs to the critic and the actor. Therefore, the controller receives the angle of attack as the control state for each time step, $x(t)$, in addition to the reference signal, $x_{ref}(t)$. The reference vector includes only one component of the angle of attack, α_{ref}. The value of the angle of attack reference is provided by the designer and defined by Eq. (30) as a sinusoidal signal initializing at the angle of attack of 8.6 degrees.

$$\alpha_{ref}(t) = \alpha_{trim} + 5sin(t) \quad (deg) \tag{30}$$

To provide the RL controller with the information on the tracking task and since an accurate model of the system is not required for the HDP architecture, the reinforcement learning states $x_{rl}(t)$ are selected from the system states $x(t)$ as defined by Eq. (31). This information from the states will be used for updating the value function, the policy and, ultimately, the control inputs.

$$x_{rl}(t) = \begin{bmatrix} \alpha(t) \\ q(t) \end{bmatrix} \tag{31}$$

In order to get a faster learning performance, from the RL states, only the angle of attack is used to obtain the tracking error that will be used as the inputs for the actor and critic networks. In this way, the actor and the critic receive information on the current systems as well as the goal. The tracking error is given by Eq. (32).

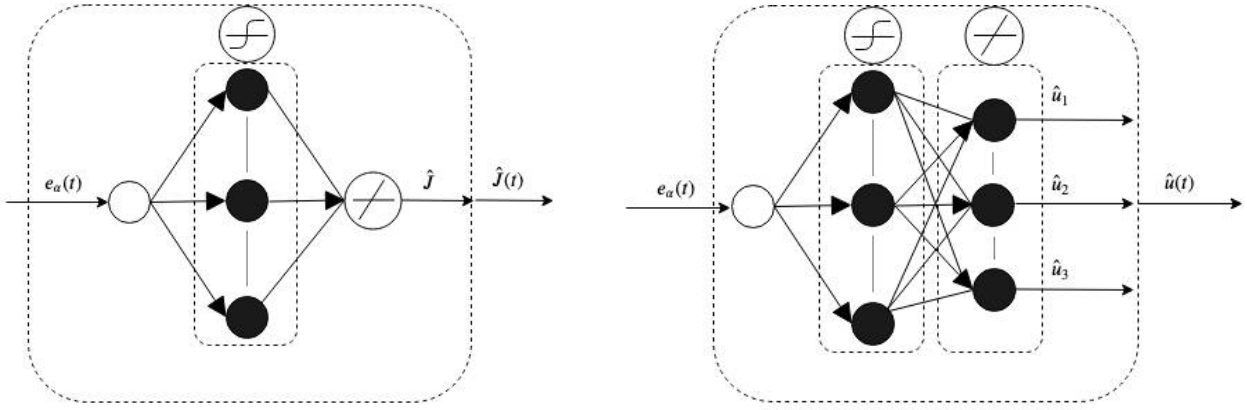
$$e_\alpha(t) = \alpha(t) - \alpha_{ref}(t); \tag{32}$$

To update the controller in the direction that the system moves towards the current system reference and maintain that, the controller requires the prediction of the successive reinforcement learning state $\hat{x}_{rl}$ that comes from the model and is used for the update of the value function.

B. The Neural Networks Design

In Fig. 6, a visualization of the actor and critic designs are shown. Both the critic and the actor networks consist of a single fully connected neural network. The critics map the state error to the value corresponds to that system state while

the actor assigns the state error to the control action. The number of outputs of the actor will be initially set based on the number of available control effectors.



(a) **(b)**

Fig. 6 Schematic of the proposed (a) critic and (b) actor networks

For the model of the system, again, a fully connected neural network is used with its structure shown by Fig.7. The model network inputs increase based on the available control inputs to the system. Therefore, an augmented vector will be used as the input to the model network.

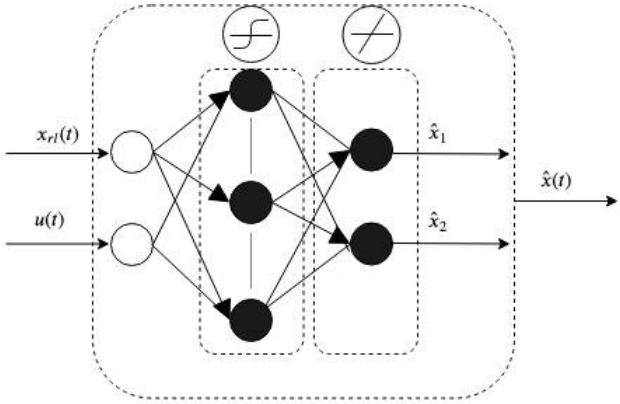


Fig. 7 Schematic of the proposed model network

C. The Learning Rates Scheduling

In an HDP framework, the actor and the critic are trained iteratively and simultaneously, which results in the online learning ability. In other words, the weights are updated incrementally at each time step based on the current error. The learning rates are very important in the convergence to the optimal network weights. Therefore, the proper choice for these parameters is crucial for convergence time and to avoid local optimum trapping. For the critic, actor and model networks, adaptive learning rates are used [14], with the update rule defined by Eq. (33). To avoid unwanted parameter variations in online learning, the learning rates scheduling is only performed in the first few seconds and are set constant afterwards.

$$\alpha(t+1) = c_\alpha \alpha(t) \tag{33}$$

where c_α is the learning rate coefficient.

D. The Networks Initialization

For any RL approach, including the HDP method, hyper-parameters are set for the controller. These parameters include learning rates, α and discounting factor γ. The number of neurons and the weight initialization are also crucial in the convergence of the controller. The learning parameters for the RL controller are shown by the table below. The weights of the networks are initialized using symmetric randomization. The initial learning rates are mentioned in the table below. During the online learning, these parameters, except for the learning rates, are kept constant to maintain the converged policy in a stationary condition.

Table 2 The RL controller learning parameters

Parameter	Value	Parameter	Value	Parameter	Value
$\boldsymbol{\alpha}_{a_0}$	500	γ_c	0.995	c_{α_a}	0.9
$\boldsymbol{\alpha}_{c_0}$	1	neurons	25	c_{α_c}	0.8
$\boldsymbol{\alpha}_{m_0}$	0.15	W_0	[-1, 1]	c_{α_m}	0.99

The model matrix which is adjusted by trial and error is defined as:

$$Q_m = \begin{bmatrix} 10.5050 & 0 \\ 0 & 3.2423 \end{bmatrix} \tag{34}$$

This weight matrix is used to adjust the focus of the model learning on the angle of attack rather than the pitch rate. Although both important, however, the RL controller task is set for tracking the angle of attack and more information for the controller is required for this state. Therefore, the accuracy of the model identification for this state becomes more crucial that can be implemented by giving more weights to the angle of attack updates.

E. Cost Function

The reward or cost definition highly depends on this environment. The configuration of the ICE model was described in section III. Since the goal is to track the reference signal for the angle of attack α_{ref}, a cost function is designed in a way to return the optimal value of zero for tracking the reference signal precisely, in any other case, the cost value will be a positive value. Therefore, the cost function is designed for this case as given by Eq. (35).

$$c(t) = Q_c e_\alpha^2(t) \tag{35}$$

where Q_c is the cost weight matrix and is defined as:

$$Q_c = \frac{200}{\alpha_{max}^2} \tag{36}$$

F. Research Cases for Attitude Control

To compare the performance of the controller for different effector configurations, a benchmarking set up is considered. The same initial conditions are used for the controller in each situation. The aerodynamic control surfaces considered for the four bench markings are shown by Table 3. These control surfaces are used as controllable inputs to the model, which includes the commanded pitch flaps in the beginning and add the elevons, the inboard and outboard leading edge flaps, and for the last benchmark, all the effectors eliminating the yaw and pitch thrust vectoring are considered. It is assumed that the effectors deflect symmetrically to eliminate their effect in lateral-directional movements. The performance of the controllers is compared in terms of their tracking performance, their computational effort, the controller stability and the effectors' control effort.

Table 3 Attitude control benchmarking cases

Case	Control Inputs
BC1	δ_{PF}
BC2	$\delta_{PF}, \delta_{Elevon}$
BC3	$\delta_{PF}, \delta_{Elevon}, \delta_{ILEF}, \delta_{OLEF}$
BC4	$\delta_{PF}, \delta_{Elevon}, \delta_{ILEF}, \delta_{OLEF}, \delta_{AMT}, \delta_{SSD}$

V. Outer Loop Altitude Control

To show the advantage of the RL controller in performing more difficult control objectives such as altitude tracking, an altitude control outer loop is implemented. The PID controller receives z_{ref} and selected $z(t)$ as inputs. The gains for the PID feedback controller are set manually and using trial and error. In contrast to the previous definition for the controller, the values of the angle of attack reference are provided from the PID controller. An overview of the structure of the complete control framework is shown by Fig.8. The framework of the RL controller is the same as was shown in the previous section.

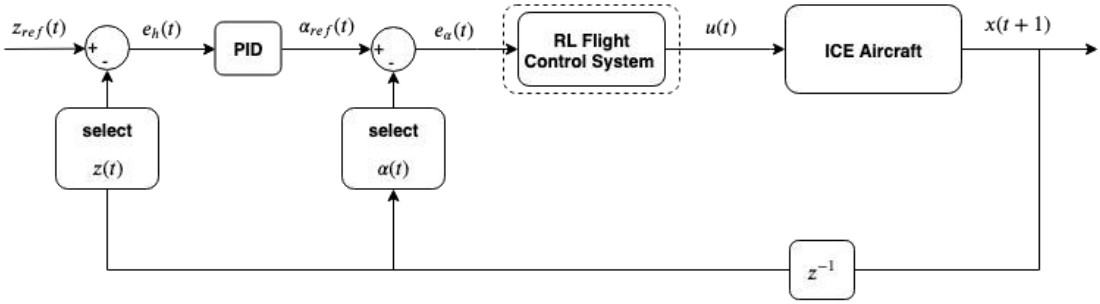


Fig. 8 Layout of the altitude control framework including the RL and the outer loop PID controllers

The controller starts with an offset from the desired altitude, and the goal is to reach the goal altitude and stabilize there (a regulation problem). The z_{ref} is defined by the designer as a constant value of:

$$z_{ref} = -10000(ft) \tag{37}$$

A. Research Cases for Altitude Control

The PID control values are presented for three different altitudes benchmarking conditions of BCH11, BCH12 and BCH22. In the first two cases, the two effectors of pitch flap and elevons are available for the controller, and only the proportional gain is changed. For the last case, all the effectors are used except for the thrust vectoring. Table 4 shows these conditions and the PID gains associated with each situation. The saturation limits are set according to the minimum, and maximum angle of attack values derived from the ICE aircraft flight envelop.

Table 4 Altitude control benchmarking cases

Case	Control Inputs	$K_P(10^{-4})$	$K_I(10^{-5})$	$K_D(10^{-3})$
BCH11	$\delta_{PF}, \delta_{Elevon}$	5.8	2	1
BCH12	$\delta_{PF}, \delta_{Elevon}$	2.6	2	1
BCH22	$\delta_{PF}, \delta_{Elevon}, \delta_{ILEF}, \delta_{OLEF}, \delta_{AMT}, \delta_{SSD}$	2.0	2	1.5

The framework for the control loop discussed in this section is used for online altitude control of the ICE aircraft.

VI. Simulations and Results

In this section, the simulation setup and online results for the ICE aircraft attitude control using the ACD controller and the altitude control with integration of the ACD and PID controller are given separately. The controller and the dynamics of the aircraft are simulated in the MATLAB environment. The dynamics of the aircraft is updated using the longitudinal force and moment coefficients that are derived from a spline model [27]. The forces and moments are then derived from these coefficients (subsection III.C). The states are updated using the Eulers method.

A. ICE Aircraft Environment

The simulation is operating with a frequency of $1000Hz$ and does not consider the turbulence effects. The measurement errors and sensor dynamics are not included in the output states. To add exploration to the system, persistent excitation in the form of white noise is added to the control inputs. The initialization of the ICE aircraft is performed in the flight condition of -1000 ft altitude and $432ft/s$ airspeed for the attitude control, with the initial values for the thirteen effectors set for their trim values for the specified altitude and airspeed. The heavyweight condition of the ICE aircraft is considered for the simulation. The ICE properties are shown by Table 5.

Table 5 The ICE aircraft properties

Property	Value	Unit
mass	1152.6	$[slug]$
S_{ref}	808.6	$[ft^2]$
$\bar{c}$	28.75	$[ft]$
I_{yy}	81903	$[slug.ft^2]$
g	32.174	$[ft/s^2]$

A Proportional-Integral (PI) thrust controller is used for airspeed control [10]. The reference airspeed is defined as the initial airspeed value, so 430 ft/s. The thrust controller proportional gain is set equal to the mass of the aircraft, while the integral gain is set to 10. The thrust values are saturated between 0 and 10, 000 lbf with anti-windup added for saturation of the integral control signal to the maximum thrust.

B. Initialization

For the attitude control part, the aircraft is initialized at the altitude and airspeed mentioned above, while the angle of attack, the pitch angle, and the pitch rate are set to values different from the trim. The aircraft is highly unstable, mainly if the trim inputs are not applied. Therefore, the control problem becomes very challenging for the HDP controller. Without a suitable controller, the states can easily deviate from initial values. The same state initialization is used for all the presented experiments in this paper for the sake of comparison except for the altitude control problem where the aircraft starts with an offset from the desired altitude with other states' initialization remain the same.

C. Attitude Control Results

The online attitude control lasts for 60 seconds. In this part, the online results are shown. The results displayed here are corresponding to one of the successful runs of the online controller. At each simulation step, the ICE model receives the reinforcement learning state vector x_{rl} and the control inputs vector $u(t)$ and predicts the response of the reinforcement learning states (state estimation). Since the initial visits in the state and action space are essential for the controller faster convergence, the persistent excitation is only applied for the first 10 seconds. Fig.9 shows the progress of the controller

as more experience is obtained during online learning. The figure shows one of the best recorded performance with the dashed line showing the angle of attack desired trajectory. The top part of the picture shows the error values for the critic (top left) and actor (top right). The plots illustrate the progress of the networks with the initial peaks corresponding to the initial exploration. The figure presents high critic error in the first 5 seconds. The same tracking error can be seen for the tracking performance by just looking at the angle of attack response, and the reference signal in the middle left part of the figure. The actor errors are small in the beginning and become very large between 1 to 3 seconds. However, the controller can adapt very fast in the very first few seconds. Fig. 10 shows the attitude time history of the controller, with the first seconds zoomed in. The figure confirms that the controller has learned to track the reference trajectory precisely in less than 5 seconds.

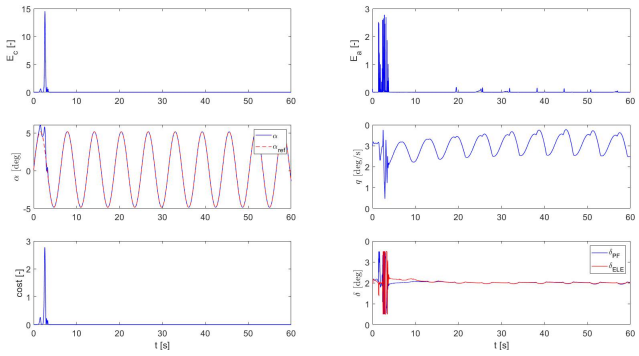


Fig. 9 The attitude control results with the critic network error in top left, the actor network error in the top right, the angle of attack response and the reference angle of attack in the middle left, the pitch rate response in the middle right, the cost value in bottom left, the effectors deflection in the bottom right (case BC2)

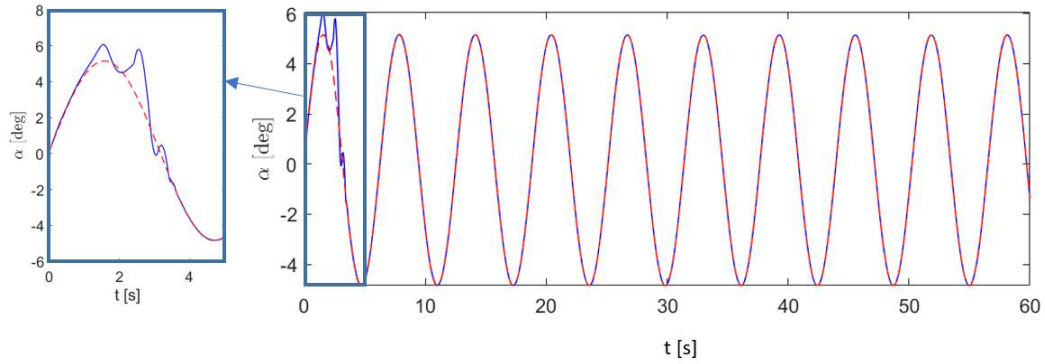


Fig. 10 The time history of the angle of attack response and reference signal with the first 5 seconds zoomed in (case BC2)

The pitch rate response is shown in the middle right part of the graph. The response shows variations with less frequency in the first few seconds, and these changes indicate a higher frequency after 2 seconds, while after 5 seconds, the pitch rate response becomes oscillatory with some small changes in the periodic behavior over time. The left bottom plot represents the cost values, which shows that the agent successfully progresses toward minimization of the cost, except for the first 3 seconds of exploration. As the cost is only a function of the states, the cost values show peaks with the same frequency as the tracking error. The cost values indicate that the controller can adapt to the reference trajectory within a few seconds. The bottom right shows the pitch flap and elevon deflections. The control inputs are saturated according to the limitations mentioned in the configuration. It can be seen that the controller first applies smaller contributions, starting from their initial values. The effectors are pushed to their limits with the increase in the actor error; however, after some exploration, the controller obtains less aggressive control signals. It shows that the actor networks are continually changing for a better policy over the 60 seconds. The significant error for the actor shows its deterioration effect on the pitch flap and elevons deflections that vary with high frequency in the same duration.

The same high frequency but high amplitude variations can be seen in the control input response, which reduces the performance. Since the control effector deflections are not considered in the cost function, there is no direct information obtained by the RL controller for the control effector deflections' influence on the cost function. However, the controller can converge quickly to a smoother response when the high tracking performance is achieved. While the controller can perform the tracking task accurately from the first few seconds, the policy still shows a few changes in the learning period.

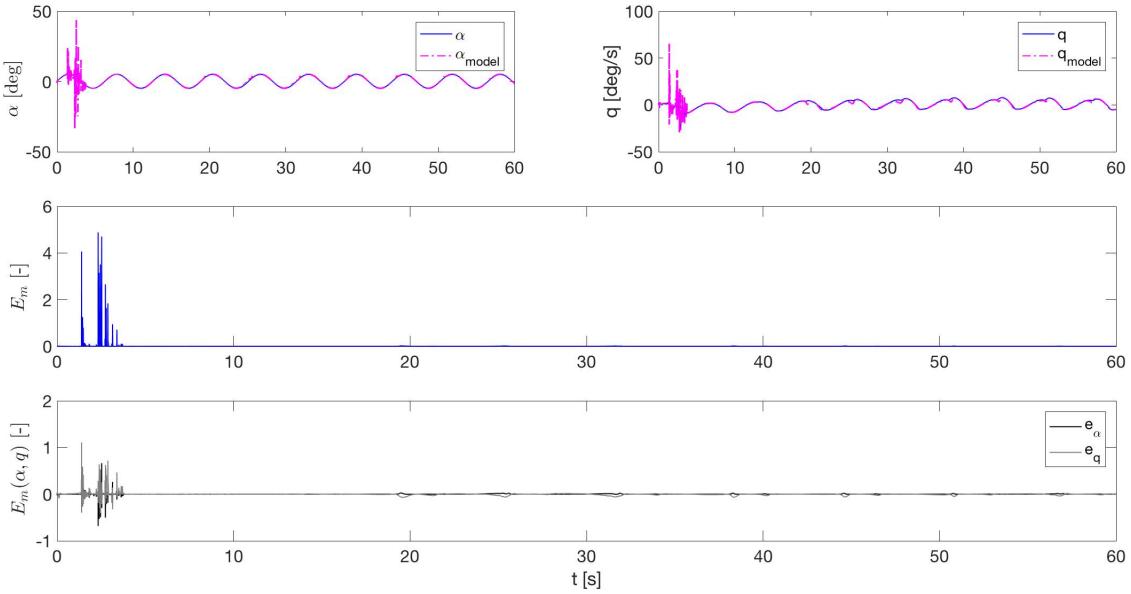


Fig. 11 The online model identification results with the angle of attack and pitch rate plant and model responses on top, the overall model error in middle and the angle of attack and pitch rate errors in the bottom (case BC2)

Fig.11 shows the online model identification results. The top left and right top parts of the plot shows the identification of the angle of attack and pitch rates, respectively. The dotted lines show the model identified states while the solid lines show the response of the ICE plant. The model is identified precisely for the angle of attack after 5 seconds, while for the pitch rate, there is some deterioration at some points in the identification process. The same observation can be made looking at the model error for the pitch rate in the bottom figure. The reason can be the low learning rate value for the model network and the weight matrix for model identification. However, the overall model error shows a great improvement after 5 seconds. The zoomed-in figure for the identified model in Fig.12 shows that the identification improves over time. The identification errors agree with continuous change in effectors deflections for the whole learning period as part of the actor training depends on the derivative of the identified model states to the effector inputs.

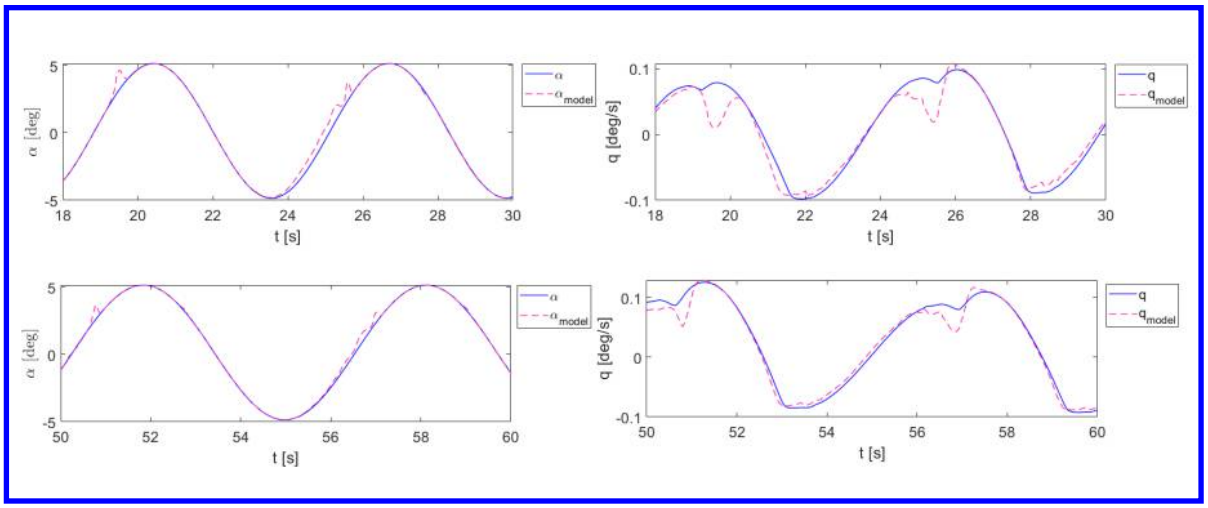


Fig. 12 The zoomed-in (case BC2) plot for model identification for the angle of attack and pitch rate between 18 to 30 seconds (top plots) and 50 to 60 seconds (bottom plots)

Fig.13 indicates the changes in the network weights for the controller. The convergence of the critic networks agrees with what already was shown by the tracking performance of the controller. However, the actor hidden layer networks show some changes within time, while the output layer weights show convergence. Therefore, the controller is still updating the policy during online learning, as was also observed from the figures above. The actor weights show more significant changes between the first 20 seconds, which was also seen in the policy obtained and presented with pitch flap and elevon deflections. The model weights show variations through online learning, which is expected as the model identification is performed locally. However, it is interesting to note that the PE added to the system initially is identified quickly by the controller and not influenced the model weights.

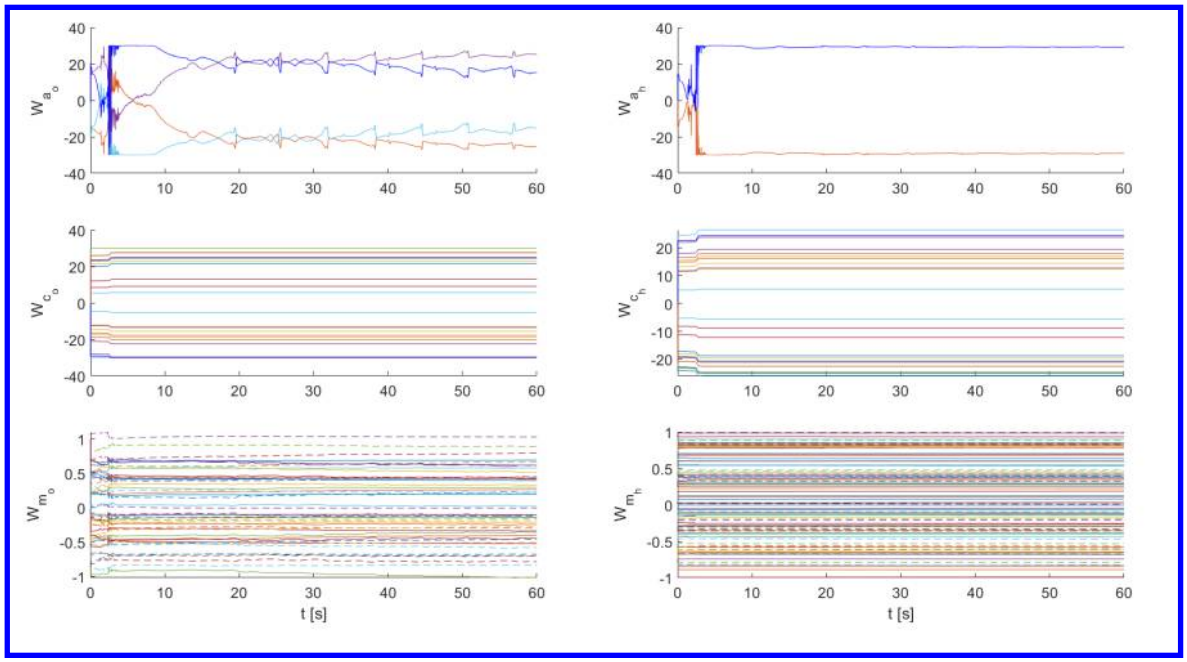


Fig. 13 The weights updates for the critic, actor and model networks (case BC2)

Qualitatively speaking, by looking at the graphs presented above, it was observed that the controller can adapt to the reference signal, and the controller is able to follow the reference angle of attack precisely in the first few seconds with online training. To see the control allocation performance of the HDP controller, in this case, the BC2 control

effectors response is presented by Fig.14. pitch flaps and elevons are the control effectors that have the most influence in longitudinal pitch control of the ICE aircraft (without thrust vectoring). Therefore, it is interesting to see how the control effort is distributed between these effectors.

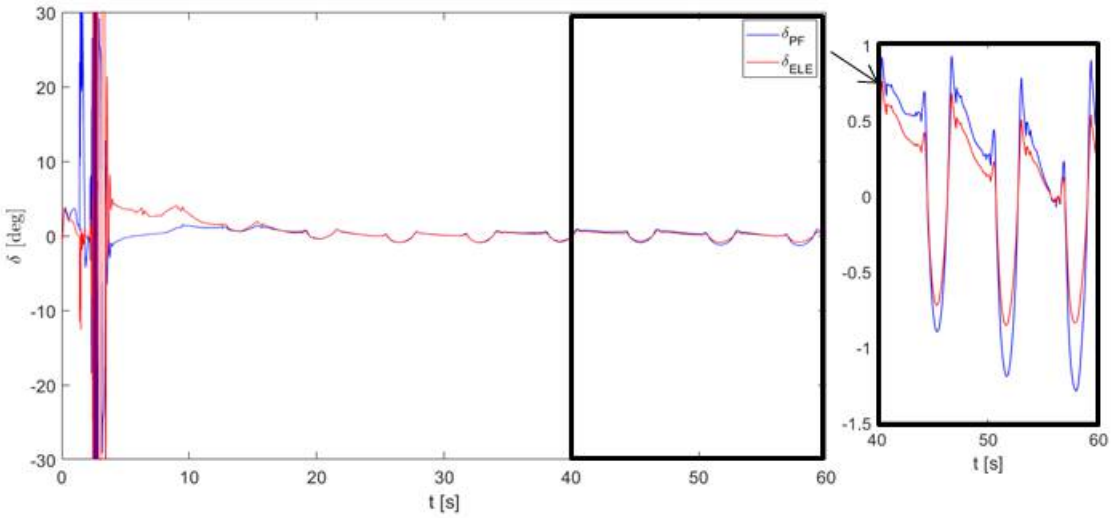


Fig. 14 The online progress of the control policy for 60 seconds of learning and the zoomed in response for the last 10 seconds (case BC2)

Fig.14 shows that the controller is capable of achieving a policy that uses the power for both effectors to achieve the angle of attack control task and the effectors' deflection are contributing in the same direction. The current performance is made while the cost function definition does not provide any feedback for the control effort, and the controller performs the angle of attack tracking objective based just on the tracking error information.

D. Comparison of the Tracking Performance

In the previous section, one of the benchmark conditions with the best performance was presented. The quantitative performance presentation for all the research cases is shown in this section to provide a basis for the comparison. The values for comparing criteria correspond to one of the successful runs. To compare the tracking performance for each condition, the Root Mean Square (RMS) of the tracking error is shown by Table 6. In addition, the simulation run time is shown to compare the computational cost of each condition. For the experiment runs, a PC with i5-6500 CPU and 8 GB of RAM is used. The control effort for each control surface is shown by Table 7, while Mean Control Effort (MCE) is defined by Eq. (38).

$$MCE = \frac{\sum |\delta_{eff}/\delta_{eff_{max}}|}{N} \tag{38}$$

where N is the number of samples during the online simulation. The success ratio is also provided for 100 runs for each condition with randomization in the initialization of the network weights to check for the stability of the controller.

Table 6 The learning performance for each benchmark condition

Case	RMS (α) [deg]	Run Time [seconds]	Success Ratio [%]
BC1	1.2843	28.9773	28
BC2	0.2251	29.7998	38
BC3	1.1277	42.4319	15
BC4	1.1589	124.5123	9

The above table results for the tracking error shows that the best tracking performance is obtained for the BC2 condition. Comparing the last two benchmarks, better tracking performance is achieved using all effectors in used when compared to the BC1. The tracking performance between the two last benchmarks is close to each other. The computational effort results are as expected, as the controller takes more time to learn the optimal policy when a higher number of effectors are used. The computational effort becomes much higher when all effectors are used, and both the model and actor networks become more complicated. The success ratios are low, which shows that the controller is quite unstable, and the success ratio becomes very low for the last benchmark. The instability of the controller can improve with some increase in the computational cost by adding more neurons to the actor and model networks. It was also found that the initialization of the network weights plays a vital role in the stability of the controller. As an example, a success ratio of more than 50 percent can be achieved if the controller weights are set to bigger initial values.

Table 7 The control effort for each benchmark condition

Case	PF MCE [-]	ELE MCE [-]	ILEF MCE [-]	OLEF MCE [-]	AMT MCE [-]	SSD MCE [-]
BC1	0.1690	0	0	0	0	0
BC2	0.0387	0.0462	0	0	0	0
BC3	0.0322	0.0407	0.0266	0.0203	0	0
BC4	0.0404	0.0468	0.0256	0.0185	0.0253	0.0122

Looking at the control efforts for pitch flaps, it can be seen that effort is reduced clearly when more effectors became available for the controller. This shows that the controller is successful in distributing the required control power between the available control effectors to follow the desired angle of attack trajectory. However, the control effort changes for pitch flaps become smaller for the cases of BC2 and BC3, while there is also a slight increase when all effectors are used when compared to the case where only four effectors are available to the controller. For elevons, the same trend can be seen as the control effort becomes smaller between the second and third benchmark condition but become higher for the last one. This can be due to the interaction between the effectors, especially for higher angles of attack that can cause more engagement of the control effectors. Also, by looking at the graphs, it was found that at the beginning of the learning, the controllers are pushed to their limits with higher frequency, while through the learning, they almost converge to a periodic response.

Another observation is the ILEF and OLEF control efforts for the last two benchmarks. The control effort for ILEF almost remains the same, while OLEF experiences lower control effort for the last benchmark compared to BC3. This can be due to the reason that the controller converges to another (near) optimal policy, and there is a different division of control power required between the control effectors. This can also include compensation of one effector for another. The control effort for the AMT and SSD effectors is quite small, which can be expected as these effectors have the most control power in lateral-directional movements.

E. Altitude Control Results

For the altitude control, the ICE aircraft starts at the same flight condition as for the attitude control, so airspeed of 430 ft/s with the same initial values for the effectors. However, in this case, the aircraft starts with -300 ft of offset from the desired altitude of -1000 ft. Therefore, the control problem is deemed more difficult if compared to a case where the

aircraft starts in the trim condition. The simulation runs at 1000 HZ with a maximum duration of 120 seconds. Different configurations for different available effectors, in addition to different PID controller gains, are tested as described by subsection V.A. The purpose is to see the ability of the outer loop controller to achieve the altitude tracking task while the RL controller is learning in parallel in an online framework. Fig.15 shows the altitude time history for the controller for three different research cases. The figure compares the approach behaviour for the regulation problem for three cases and shows how the controller behaves under different conditions. The results are shown for one of the best recorded performance. The dashed horizontal line in the figure shows the desired altitude. The two first cases only consider pitch flaps and elevons as the available control effectors, while the last case considers all the effectors except for the thrust vectoring. The altitude behaviour shows the success of the controller in moving toward the desired trajectory.

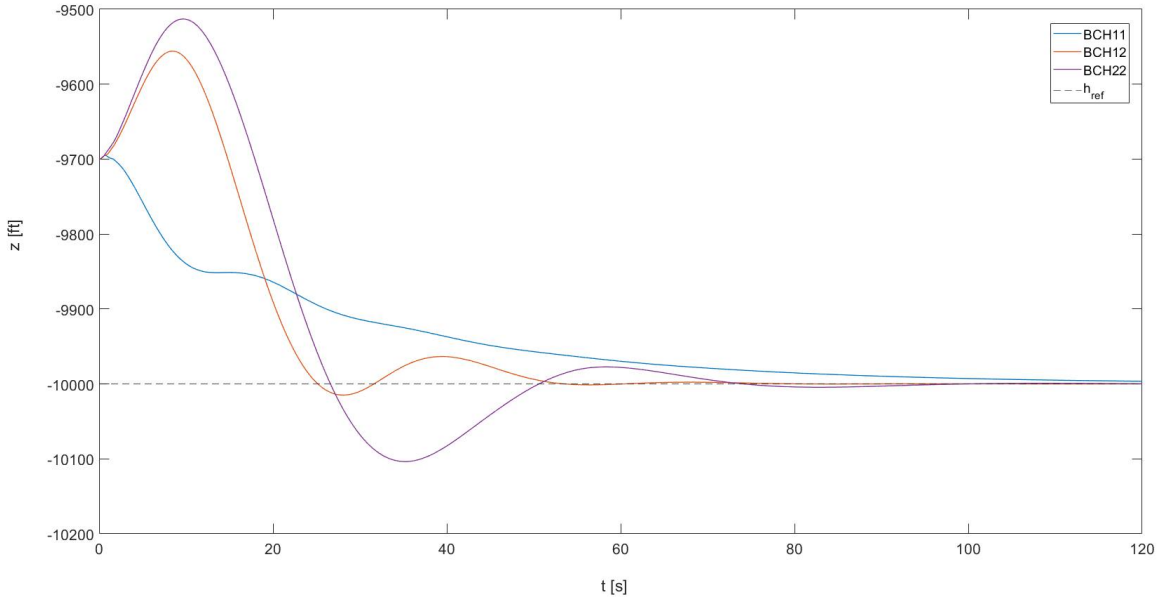


Fig. 15 The altitude tracking results for different benchmarking conditions with 300 ft offset

The altitude time history shows that the controller is capable of approaching the desired altitude and stabilizing the aircraft, especially for cases BCH12 and BCH22. Some differences can be seen for various conditions, as the chosen trajectory is expected to depend highly on the PID controller gains. The BCH11 case reflects a less aggressive manoeuvre and a smoother approach to the desired altitude. In other words, the variations in altitude are more gradual and with less frequency if any. While conditions BCH12 and BCH22 attempt to reach the reference altitude, there is an overshoot in the first few seconds, which become more prominent for the last case compared to BCH11. The 'rise time' is shorter for case BCH12 compared to BCH22, while there is an oscillation afterwards with a smaller amplitude around the desired trajectory. The overshoot can be explained due to the large initial angle of attack of the aircraft, which cause the behaviour of the aircraft first to start losing altitude and then recover from that and reaches the desired altitude and attempt to maintain that. However, it can be seen that by choosing a larger proportional gain, this overshoot can be very much damped while still happening. Comparing the first two cases, the more significant overshoot and more aggressive manoeuvre for the BCH12 compared to BCH11 result in stabilization of the aircraft around the desired altitude before 60 seconds. However, for the BCH11, the desired altitude is only reached after 150 seconds and slower. For the last case BCH22, the altitude stabilizes only after about 90 seconds, while the amplitude of the oscillations is larger. To get a better understanding of aircraft behaviour and to see if the aircraft is able to follow the obtained trajectory, the angle of attack of attack tracking and the effectors' policy should be analyzed.

Fig.16 shows the angle of attack response. The dashed lines in the figure show the reference angle of attack signals derived from the PID altitude controller. What the plot shows is that the angle of attack follows the reference signal precisely before 10 first seconds. While the zoomed-in figure also shows that the angle of attack is damped around the reference PID output after some initial oscillations. As could have expected for the BCH22 condition, the angle of attack tracking becomes accurate in a longer period as a more difficult task is defined for the controller. One interesting observation is that, as expected, the angle of attacks converge to a value close to the trim value at this altitude, where the

initial effectors were set based on. There are minimal deviations from the trim value, which is caused by the integration control signal. It was found from Fig.15 that the most significant result is obtained by case BCH11.

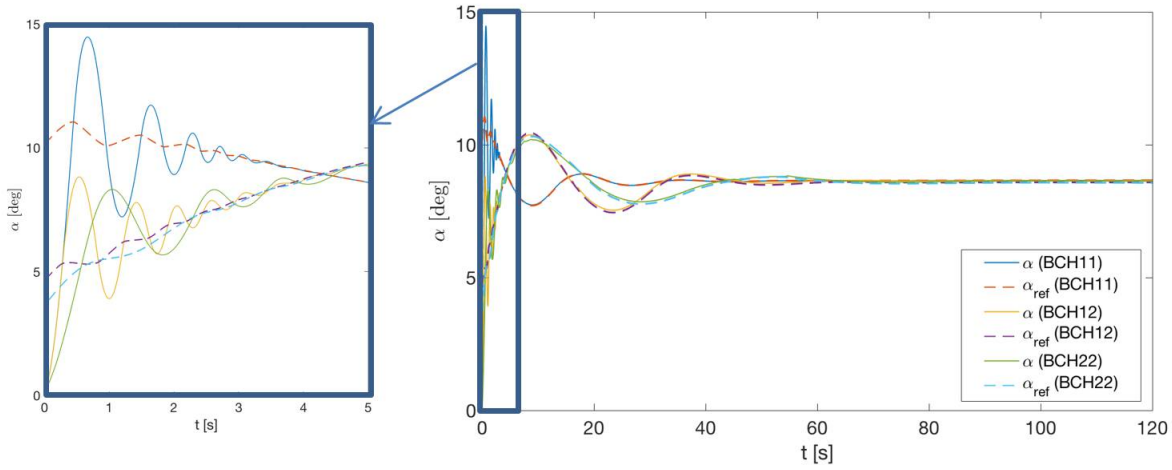


Fig. 16 The angle of attack reference signal and the response of the ICE plant for different benchmarking conditions with the outerloop altitude control

From Fig.16, it can be seen that for the BCH11 and BCH12, the controller performs more aggressive angle of attack tracking maneuver to increase the angle of attack at the start of the learning and adding to the thrust values to compensate for the decrease and drop in the airspeed as shown by Fig.17 and Fig.18. On the contrary, from the same figure, it can be seen that for the last case, the speed is initially increased very slightly, and therefore, the thrust decreased significantly while the angle of attack is expanding to reach the reference signal. After the angle of attack is damped around the reference signal, the velocity follows the same trend as the angle of attack behavior. As expected, since the integral gain for all the research cases of BCH11 and BCH12 are the same, the angle of attack converges around the equal value for these cases. The BCH11 and BCH12 also show better tracking performance by just looking at the graphs as for these conditions, the angle of attack response is able to follow the reference signal precisely before 5 seconds. This is while, for BCH22, the angle of attack is still oscillating around the reference trajectory. Obviously, for the BCH22 case, the controller needs more exploration within the more complicated action space to find the best combination of the effectors to achieve the control task.

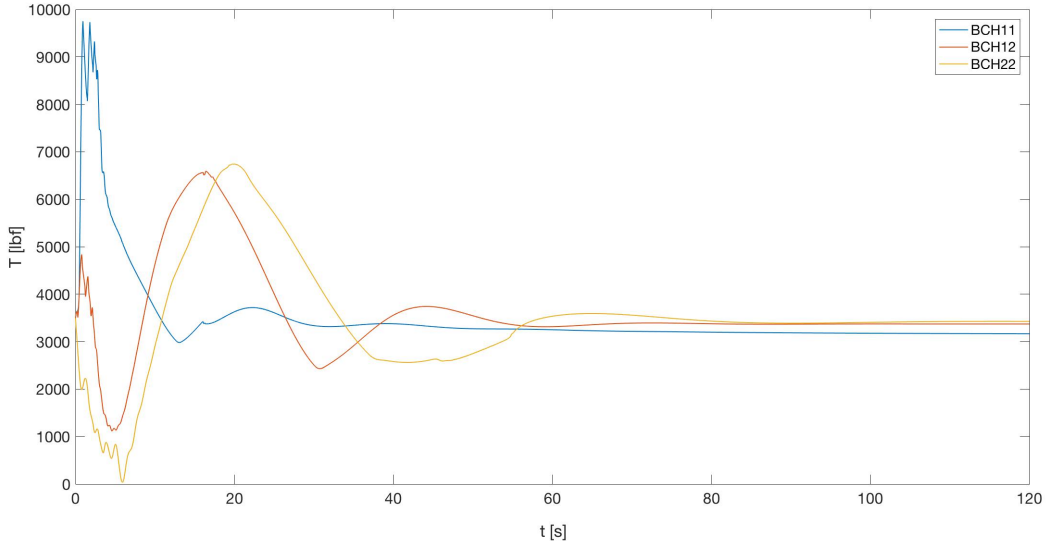


Fig. 17 The thrust values for different benchmarking conditions with the outerloop altitude control

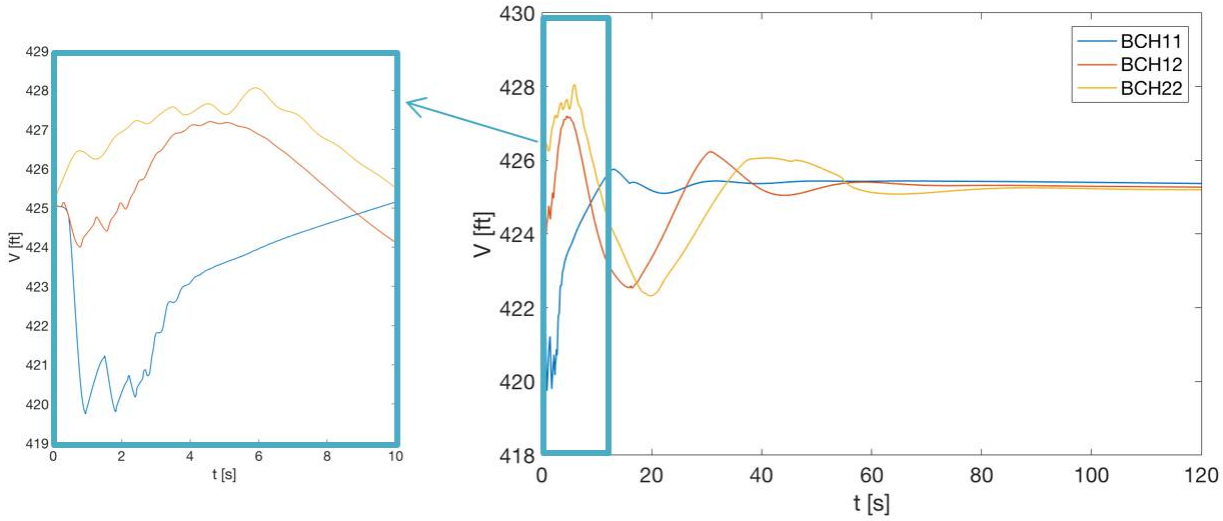


Fig. 18 The airspeeds for different benchmarking conditions with the outerloop altitude control

Although the same initial airspeed drop can be seen for cases BCH11 and BCH12, for the latter, the airspeed drop is smaller compared to the former. However, the airspeed for the rest of the learning shows some oscillations before stabilizing, which relates to the angle of attack changes. Apart from the different initial responses of the airspeed in the first few seconds, for the rest of the learning with BCH22, the airspeed tends to oscillate with the angle of attack stabilizing around its trim value. Therefore, the controller will need more than 1 minute to maintain the altitude. The gradual increase in airspeed can also be seen here for the first research case, which converges after almost 30 seconds while the altitude is still increasing towards the desired altitude.

Fig.19 shows the effectors' control inputs time histories for the controllers under two first cases. It can be seen that a different control policy is achieved for the pitch flap and elevon deflections in two cases, while there is a high-frequency usage of the control effectors at the beginning for both cases. The effectors are pushed to their limits initially, while the control policy starts becoming less aggressive sooner for the BCH12 case. For both cases, the control effector deflections utilization decrease before 5 seconds. It can be seen that the pitch flaps and elevons are used antagonistically in both cases and produce additional drag that is not desired. As already mentioned, the effort of the controllers is not included in the objective function, and therefore, there is no information provided about the effectors' deflections to the controller. Although in the attitude case, the controller is able to achieve a contributing effect from the controllers, in this case, a larger control effector utilization is achieved. However, no direct conclusion can be made by comparing the two, as there is no basis for comparison in this case. The excessive effector utilization also explains the increase in thrust for case BCH12 after 10 seconds and the decrease in the slope of the thrust behavior for case BCH11, while for both cases, the thrust values are converging to values close to their trim values.

The control effectors response for the BCH22 case are shown by Fig.20. The plot shows the same high-frequency changes in the control effectors deflections for the first few seconds. However, it can be seen that for this case, where six effectors become available for the controller, the control policy does not include any effector deflections to their saturation limits, while the deflections become smaller in frequency after 6 seconds of learning and change with a smaller slope. Another observation is that close to 18 seconds of learning, the deflections change signs and obtain larger values while the controller also starts using spoilers (SSDs). This is followed by the decrease in effectors utilization after 42 seconds and then increasing them again before 50 seconds of learning where the control policy start converging.

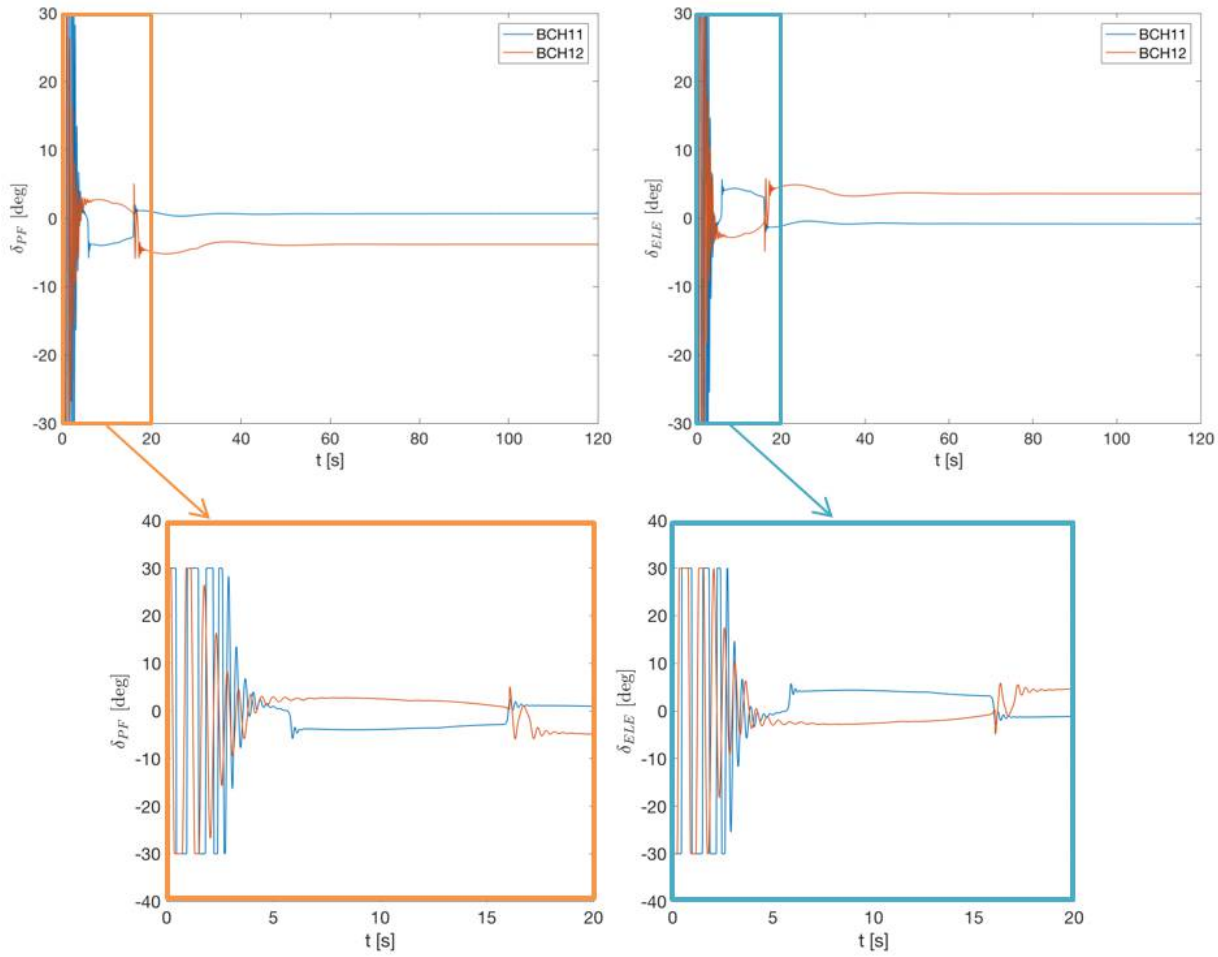


Fig. 19 The pitch flap and elevon responses for BCH11 and BCH12

It can be seen that after first high deflections of the ILEF effector, the controller decides not to use this effector for the rest of the learning after 1bout 19 seconds of online learning (due to overlapping of the AMT and ILEF graphs, it's not possible to distinguish the plots). The same can be seen for the AMT deflections, as these effectors deflections are reduced to zero by the change in the policy after 19 seconds. By just looking at the graph, one can see that the significant changes in the policy happen in 19, 45, and 58 seconds of learning before the controller convergence. It can be seen that for this case, also, the PF and ELE effectors are deflecting antagonistically, while the outboard leading edge flaps and spoilers deflections are contributing with the elevons. However, the overall effect would increase the altitude and therefore contribute to altitude regulation. This way of effector utilization, however, would result in excessive drag. Therefore, more thrust power is commanded by the aircraft after 10 seconds, as shown by Fig.17. However, the converged control policy requires almost the same thrust values that the controller converges to when compared to the two other research cases where only two effectors are used. The deflections of the rest of the control inputs remain unchanged while setting at small values.

The quantitative comparison of the different research conditions is presented by Table 8. The results presented in this table correspond to the successful runs plotted above. It can be observed that the altitude tracking error has the least value for the BCH11 case, and for the last case, the tracking error increase by almost 95 % from the first case while the angle of attack tracking error, in this case, is smaller than BCH11 and BCH12. Therefore, for the current solutions, the inclusion of more effectors resulted in a worse altitude tracking performance but a better angle of attack track. Therefore, the angle of attack tracking performance almost shows the opposite behavior than the altitude, by achieving the most tracking error for the case BCH11 and the least for BCH22. This also reflects in the Fig.16 where we see bigger amplitudes for the angle of attack oscillations around the reference signal.

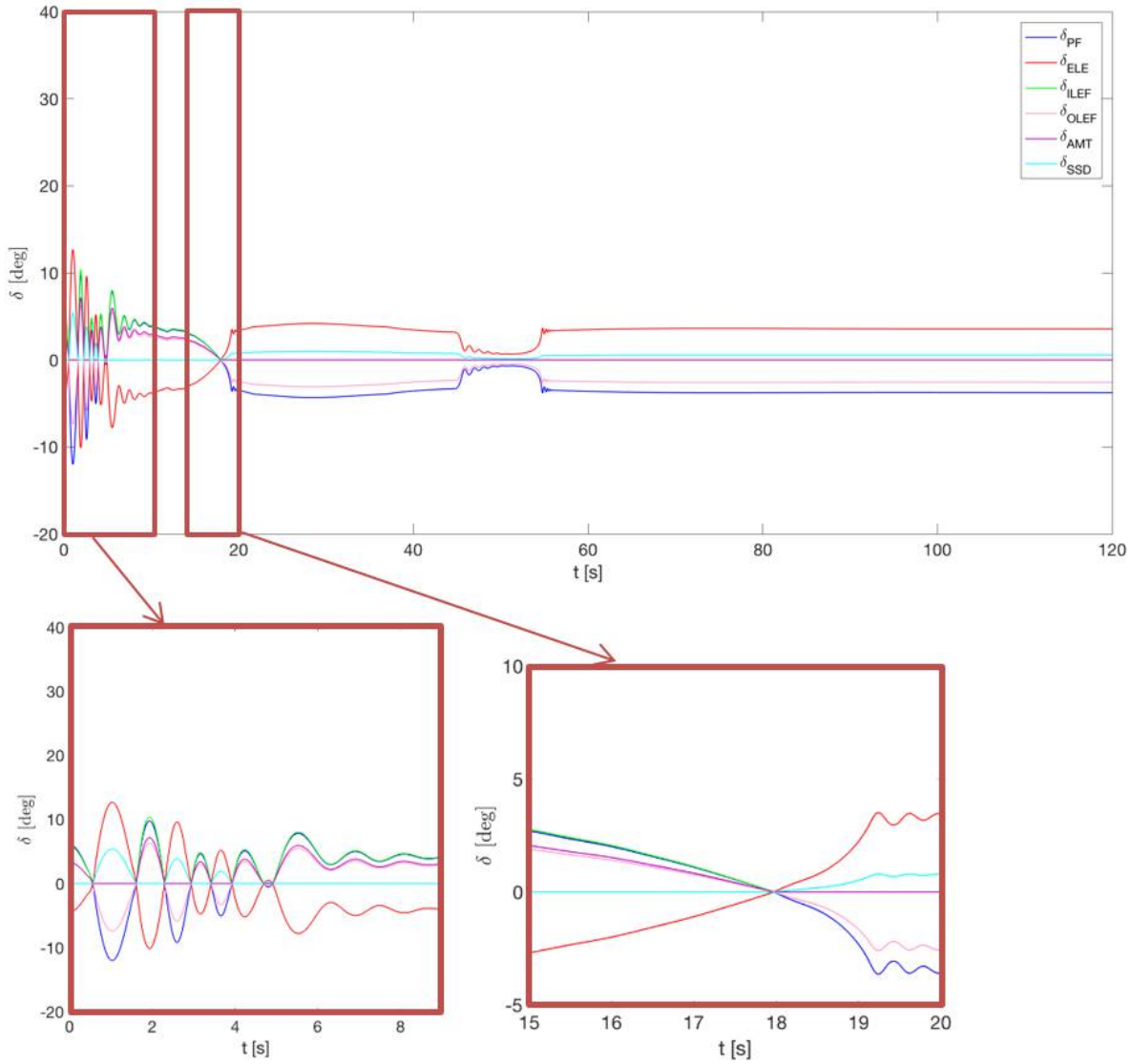


Fig. 20 The effectors responses for BCH22 case

So, the best angle of attack performance is achieved by BCH22. The results corresponding to the stability of the controller show that case BCH11 shows more than twice the success ratio when compared to the following two conditions for 100 runs of all the controllers with random initialization of the network weights. For the last two cases, the success ratio is almost the same for the two controllers.

Table 8 The altitude tracking performance for benchmark conditions

Case	RMS (z) [ft]	RMS (α) [deg]	Success Ratio [%]
BCH11	93.3719	0.4958	50
BCH12	143.2720	0.2695	21
BCH22	169.4833	0.2691	22

Table 9 shows the mean control effort for all the effectors in all the research conditions. All effector utilization (except for the MTV) is only possible for the last research case, while in the first two cases, only the pitch flap and elevons

can be used. In all instances, the cost function of the ACD controller only considers the angle of attack tracking error. In general, it can be seen that making more effectors available for the controller does not cause a lower control effort for the pitch flaps and elevons when research cases BCH11 and BCH22 are compared. Interestingly, for both effectors, the control effort is significantly smaller for the case BCH11. The control effort for the rest of the control surfaces is little, with the AMT and SSD control efforts substantially lower compared to the rest of the effectors. Therefore, the pitch flaps and elevons, which have the most direct effect on longitudinal controls are used considerably more than the effectors that are designed for lateral-directional movements.

Table 9 The altitude tracking control efforts for benchmark conditions

Case	PF	ELE	ILEF	OLEF	AMT	SSD
	MCE [-]	MCE [-]	MCE [-]	MCE [-]	MCE [-]	MCE [-]
BCH11	0.0544	0.0587	0	0	0	0
BCH12	0.1373	0.1325	0	0	0	0
BCH22	0.1178	0.1137	0.0116	0.0611	0.0056	0.0094

In the last case, the elevons control effort is almost the same as pitch flaps, and as was shown, they are moving antagonistically. This explains why to achieve the control objective; the policy contains using other effectors. As was expected, all moving tips control effectors control effort is close to zero, and they have been utilized the least. It can be seen that the OLEF has a more considerable control effort and can be seen by the Fig.20, the OLEF effectors are deflecting in contribution to pitch flaps and probably compensate for the elevons antagonistic behavior. This results directly in the decrease in the airspeed due to the increase in the drag in this duration that will be compensated by an increase in the thrust values. Introducing more effectors to the controller increase the ambiguity in the solutions, and with the effector utilization not being considered in the cost function, it can be seen that while the primary objective of altitude control is achieved, however, the effectors are not being used in an efficient way.

VII. Conclusions and Final Remarks

Adaptive control methods have been proposed for the control of the automated systems. Most of these approaches are model-based or cannot be applied online. The Innovative Control Effector (ICE) aircraft design comes with a large control suite that results in very complex dynamics for the aircraft. Therefore, a model-free control approach that is adaptive and can deal with the complicated system of the ICE aircraft is beneficial. A controller design based on continuous Reinforcement Learning (RL) algorithms can provide an adaptive controller for the ICE aircraft that is model-free and can be used for online control of the aircraft.

The results for the application of the approximate actor-critic-based reinforcement learning method of Heuristic Dynamic Programming (HDP) for control of the nonlinear innovative control effector aircraft model using HDP shows the advantage of this approach in two ways. First, the ability of the approximate RL controller is shown in achieving an attitude tracking task with online internal local model identification. Second, the controller is able to learn online and without prior knowledge of the system dynamics, to control the longitudinal dynamics of the aircraft in achieving the desired altitude by integrating the RL controller with an outer loop PID controller, both learning online and simultaneously. The results are shown for symmetric utilization of different configuration of the control effectors. In all cases, the tracking error is the only source of information for the controller. While the HDP controller is successful in both problems of attitude control and altitude control of the ICE aircraft, it is concluded that achieving the control objective, especially for the second problem, results in inefficient effector utilization, causing excessive drag and, therefore, misuse of thrust power. This means that for better and more efficient use of the effectors, the control effector utilization and possibly thrust usage should be directly included in the cost function. It is shown that inefficient control effector utilization is not entirely reflected in the attitude control problem or when small numbers of effectors are considered. However, for altitude control and with the introduction of the new effectors, this issue becomes more important for the controller.

The quantitative comparison for attitude control with a different number of effectors shows that one of the best tracking performances are obtained when the controller considers pitch flaps and elevons as the available control powers.

It is also concluded that the increase in the number of effectors results in longer run times and adding the effectors of leading-edge flaps, all moving tips and spoilers reduces the success ratio significantly. In general, it is found that increasing the number of effectors does not necessarily decrease the control effort in attitude control, and some cases increase the effort for effectors that are designed for longitudinal control, like pitch flaps and elevons). For the altitude control case, it is shown that while the solution depends on the gains of the outer-loop controller, the altitude error increase with adding more effectors while the angle of attack error decreases. Also, a more success ratio is achieved if the altitude control is performed with less overshoot. In this case, too, adding more effectors is shown not to decrease the control effort of the controllers necessarily.

Finally, for future studies, the RL controller implementation for higher order control problem of altitude control with no cooperation with the PID controller should be investigated. For the current proposed framework, the effectors' response to the inclusion of effector utilization in the cost function can be considered to solve the problem of inefficient effector use and as a result, to avoid the excessive drag. One downside of the HDP control with neural networks, which is inherited by policy gradient methods, is the probability that you might trap in a local optimum for some function approximation approaches. One solution can be to use Softmax functions or by using TD prediction methods (e.g., using target networks). Finally, one of the vital control power available for longitudinal control of the ICE aircraft is the thrust vectoring, or more precisely, the pitch thrust vectoring. In future studies, this effector should also be considered for the investigations with RL using the complete control suite.

.

References

[1] Airplanes, C., "Statistical summary of commercial jet airplane accidents," *Worldwide Operations*, Vol. 2008, 1959.

[2] Dorsett, K. M., and Mehl, D. R., "Innovative control effectors (ICE)," Tech. rep., Lockheed Martin Tactical Aircraft Systems Fort Worth Tx, 1996.

[3] Bowlus, J., Multhopp, D., Banda, S., Bowlus, J., Multhopp, D., and Banda, S., "Challenges and opportunities in tailless aircraft stability and control," *Guidance, Navigation, and Control Conference*, 1997, p. 3830.

[4] Niestroy, M. A., Dorsett, K. M., and Markstein, K., "A Tailless Fighter Aircraft Model for Control-Related Research and Development," *AIAA Modeling and Simulation Technologies Conference*, 2017, p. 1757.

[5] Whitford, R., "Design for air combat," *RUSI Journal*, Vol. 132, 1987, pp. 79–80.

[6] Thrun, S., and Littman, M. L., "Reinforcement learning: an introduction," *AI Magazine*, Vol. 21, No. 1, 2000, pp. 103–103.

[7] Zhou, Y., Van Kampen, E., and Chu, Q. P., "Incremental approximate dynamic programming for nonlinear flight control design," *Proceedings of the 3rd CEAS EuroGNC: Specialist Conference on Guidance, Navigation and Control, Toulouse, France, 13-15 April 2015*, 2015.

[8] Zhou, Y., van Kampen, E., and Chu, Q. P., "Launch vehicle adaptive flight control with incremental model based heuristic dynamic programming," *68th International Astronautical Congress (IAC)*, 2017.

[9] Zhou, Y., van Kampen, E.-J., and Chu, Q. P., "Incremental model based online dual heuristic programming for nonlinear adaptive control," *Control Engineering Practice*, Vol. 73, 2018, pp. 13–25.

[10] de Vries, P. S., and Van Kampen, E.-J., "Reinforcement Learning-based Control Allocation for the Innovative Control Effectors Aircraft," *AIAA Scitech 2019 Forum*, 2019, p. 0144.

[11] Ferrari, S., and Stengel, R. F., "Online adaptive critic flight control," *Journal of Guidance, Control, and Dynamics*, Vol. 27, No. 5, 2004, pp. 777–786.

[12] Ferrari, S., Steck, J. E., and Chandramohan, R., "Adaptive feedback control by constrained approximate dynamic programming," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, Vol. 38, No. 4, 2008, pp. 982–987.

[13] Lewis, F. L., Vrabie, D., and Syrmos, V. L., *Optimal control*, John Wiley & Sons, 2012.

[14] Zhou, Y., "Online reinforcement learning control for aerospace systems," Ph.D. thesis, Delft University of Technology, 2018.

[15] van Kampen, E.-J., Chu, Q., and Mulder, J., "Continuous adaptive critic flight control aided with approximated plant dynamics," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2006, p. 6429.

[16] Powell, W. B., "What you should know about approximate dynamic programming," *Naval Research Logistics (NRL)*, Vol. 56, No. 3, 2009, pp. 239–249.

[17] Bradtke, S. J., and Barto, A. G., "Linear least-squares algorithms for temporal difference learning," *Machine learning*, Vol. 22, No. 1-3, 1996, pp. 33–57.

[18] Eerland, W., de Visser, C., and van Kampen, E.-J., "On Approximate Dynamic Programming with Multivariate Splines for Adaptive Control," *arXiv preprint arXiv:1606.09383*, 2016.

[19] Sutton, R. S., and Barto, A. G., *Reinforcement learning: An introduction "Complete Draft"*, MIT press Cambridge, 2018.

[20] Modares, H., Sistani, M.-B. N., and Lewis, F. L., "A policy iteration approach to online optimal control of continuous-time constrained-input systems," *ISA transactions*, Vol. 52, No. 5, 2013, pp. 611–621.

[21] Si, J., Barto, A. G., Powell, W. B., and Wunsch, D., *Handbook of learning and approximate dynamic programming*, Vol. 2, John Wiley & Sons, 2004.

[22] Kiumarsi, B., Vamvoudakis, K. G., Modares, H., and Lewis, F. L., "Optimal and autonomous control using reinforcement learning: A survey," *IEEE transactions on neural networks and learning systems*, Vol. 29, No. 6, 2018, pp. 2042–2062.

[23] Modares, H., Lewis, F. L., and Naghibi-Sistani, M.-B., "Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems," *Automatica*, Vol. 50, No. 1, 2014, pp. 193–202.

[24] Dierks, T., and Jagannathan, S., "Online optimal control of nonlinear discrete-time systems using approximate dynamic programming," *Journal of Control Theory and Applications*, Vol. 9, No. 3, 2011, pp. 361–369.

[25] Luo, X., and Si, J., "Stability of direct heuristic dynamic programming for nonlinear tracking control using PID neural network," *Neural Networks (IJCNN), The 2013 International Joint Conference on*, IEEE, 2013, pp. 1–7.

[26] Buffington, J., "Tailless aircraft control allocation," *Guidance, Navigation, and Control Conference*, 1997, p. 3605.

[27] van der Peijl, I. V., "Physical Splines for Aerodynamic Modelling of Innovative Control Effectors," Master's thesis, Delft University of Technology, 2017.