CEAS-GNC-2022-075

CEAS EuroGNC 2022
*"Conference on Guidance, Navigation and Control"*
3-5 May 2022 @ Technische Universität Berlin, Germany

# Intelligent Aircraft Stabilization Control with Event-Triggered Scheme

**Bo Sun** PhD Candidate, Delft University of Technology, Department of Control and Operations, 2629 HS, Delft, The Netherlands. B.Sun-1@tudelft.nl

**Cheng Liu** PhD Candidate, Delft University of Technology, Department of Control and Operations, 2629 HS, Delft, The Netherlands. C.Liu-10@tudelft.nl

**Killian Dally** Control Engineer, DeepX Inc, 113-0034, Tokyo, Japan. killian.dally@deepx.co.jp

**Erik-Jan van Kampen** Assistant professor, Delft University of Technology, Department of Control and Operations, 2629 HS, Delft, The Netherlands. E.vanKampen@tudelft.nl

***ABSTRACT***

**This paper develops an intelligent optimal stabilization control approach that can be applied to general flight control systems. The self-learning controller is developed based on dual heuristic programming (DHP) in cooperation with the event-triggered scheme to save computational load. Besides, the control inputs can be handled by the combination of an integral cost function and a bounding actor network. A simulation study is carried out based on a nonlinear aerospace system to demonstrate the applicability of the constructed approach. The results show that the proposed event-triggered DHP control approach can maintain comparable performance with the time-based approach and meanwhile substantially decrease the amount of computation. The source code can be found at** `https://github.com/sunbojason/Event-triggered-DHP`**.**

**Keywords:** Flight Control System; Dual Heuristic Programming; Event-Triggered Scheme; Input Constraints

## 1 Introduction

As the epidemic eventually comes to an end, the world is supposed to be booming again and the aerospace industry will not be an exception [1, 2]. With the rapid development of technology and the pursuit of sustainable lifestyle, aerospace systems have become more complex for higher efficiency and environmental friendliness [3], which has given birth to a variety of promising projects such as Flying-V [4] and Smart-X [5]. Nevertheless, due to structural limitations, aerospace systems often suffer from the underactuation, which means that the system states are driven by fewer control inputs. This phenomenon turns to be more serious as flexibility and morphing capabilities are sought-after, which largely increases the system dimensionality [3, 5]. To deal with this issue, the trade-off between system states and control inputs is required, in which optimal control plays a significant role.

Optimal control strives to optimize the control policy by maximizing/minimizing a given performance function that captures desired objectives [6] and has always been one of the most mainstream control methods in aerospace systems [7]. Traditionally, the system is firstly linearized and LQR control is adopted by solving the Riccati equation [7]. However, this linear approach cannot handle more complicated limitations such as input saturation, and the linear model can result in the loss of details in

the system dynamics, which will degrade the control performance. When dealing with nonlinear optimal control problems, solving the Hamilton-Jacobi-Bellman (HJB) equation is usually involved [8, 9]. However, until now there is no effective approach to obtain the analytical solution of the HJB equation for general nonlinear systems [6]. Therefore, effective ways to acquire numerical solutions of general HJB equations are demanded.

In recent decades, the surge of Reinforcement Learning (RL) has brought another perspective on the traditional optimal control field. As a bio-inspired approach, RL can optimize the control policy in an online manner by interacting with the environment just like human's learning process [10]. Combined with approximation functions such as Artificial Neural Networks (ANNs), RL has become a powerful tool in solving sequential decision making and optimal control problems in various aerospace systems [11–14]. One branch of RL is Adaptive Dynamic Programming (ADP), which provides a promising tool for numerically solving general HJB equations by the general policy iteration [6, 10]. For the Discrete-Time (DT) implementation, ADP is often called Adaptive Critic Design (ACD) because an actor network and a critic network are constructed to approximate policy improvement and policy evaluation to handle the "bootstrap" property, respectively [6, 10, 11]. Among common ACD structures, Dual Heuristic Programming (DHP) has obtained much attention and been successfully applied to aerospace control [14]. One of the significant merits of DHP is that, instead of approximating the performance function, its critic network approximates the performance derivative, which can reduce the error introduced by the derivation backward through the critic network and therefore enable faster convergence and higher precision [14, 15].

Nevertheless, a drawback of conventional ACD methods is, to achieve satisfactory accuracy, multiple iterations are carried out, which increases the computational load. To tackle this issue, the event-triggered scheme is introduced. Originating from the networked control, it aims at decreasing the communication's cost, in that the control policy is not updated until a certain condition is satisfied [16, 17]. The event-triggered scheme has successfully been verified in aerospace systems, such as cooperative localization [18] and spacecraft attitude control [19]. Recently, some event-triggered DHP methods have been developed for controller design [15, 20], but their applications have not been spread to aerospace systems. Furthermore, input saturation is normally encountered in flight control due to structural limitations. The classic approach to deal with it is merely introducing a non-quadratic cost function from which a bounded control input can be derived. This method works well in continuous time control [21, 22] but is not adequate for DT systems [20] because the control signal is actually generated by the actor network. Inspired by the work of [8, 16, 23], this paper develops a bounding layer in the actor network, which can provide an absolute constraint guarantee. Overall, the main contributions of this paper are summarized as follows:

1) To the best of our knowledge, it is the first time that an event-triggered DHP control approach is applied to aerospace systems.
2) Control input constraints are taken into consideration and coped with by adopting both the non-quadratic cost function and the bounding layer.
3) The developed approach is verified through numerical simulations on a linear and a nonlinear aerospace system.

The remainder of this paper is structured as follows. Section 2 describes the optimal regulation problem in the event-triggered scheme. Section 3 introduces the DHP technique for approximately solving the DTHJB equation by constructing the critic and actor networks. Subsequently, Section 4 verifies the developed event-triggered DHP control approach by applying it to two aerospace systems, which are linear and nonlinear, respectively. Finally, Section 5 summarizes the paper and proposes possibilities for future research.

## 2 Optimal Event-Triggered Control

This paper aims at developing a general problem-solving framework that can be applied to a wide range of aerospace systems. Therefore, a general nonlinear DT system is taken into consideration as:

$$x_{t+1} = f(x_t, u_t),\ t \in \mathbb{N}, \tag{1}$$

where $t$ denotes the time instant, $x_t \in \Omega \subset \mathbb{R}^n$ is the state vector, and $u_t \in \Omega_u$ is the control input vector. $\Omega_u = \{u | u \in \mathbb{R}^m, |u_i| < u_\mathrm{b}, i = 1, \ldots, m\}$, with $u_\mathrm{b} > 0$ denoting the constraint on $u_i$. In this paper, the input constraints are assumed the identical and symmetric for each input element. For different and asymmetric input constraints, readers can refer to [8] and [16], respectively. Besides, the following assumption holds for system (1), such that a continuous state feedback control policy $u_t = \mu(x_t)$, $\mu : \Omega \to \Omega_u$ that can stabilize it to the equilibrium point exists:

**Assumption 1.** *System* (1) *is controllable and observable, and* $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ *is a Lipschitz continuous function. With the control input* $u_t$*, there exists a unique equilibrium point at the origin* $x_t = 0$*, i.e.,* $f(0,0) = 0$*.*

Subsequently, the event-triggered scheme is introduced, where the control input is not updated until a certain triggering condition is satisfied. By defining a sequence of triggering instants $\{s_k\}_{k=0}^{\infty}$, where $s_k < s_{k+1}$, $k \in \mathbb{N}$, a gap function can be defined using the event error:

$$e_t = x_{s_k} - x_t, \forall t \in [s_k, s_{k+1}), \tag{2}$$

where $x_t$ is the current state and $x_{s_k}$ is the nearest previous triggering state. Then, the feedback control policy can be represented as:

$$u_t = \mu(x_{s_k}) = \mu(e_t + x_t). \tag{3}$$

It is worth mentioning that $u_t$ remains constant by involving a Zero-Order Hold (ZOH) until the next triggering instant $s_{k+1}$ is reached [17]. Accordingly, system (1) takes the form:

$$x_{t+1} = f(x_t, \mu(e_t + x_t)). \tag{4}$$

The new system (4) inherits all properties of the original system (1). With the only origin equilibrium point, a non-discounted cost can be formulated as:

$$J(x_t) = \sum_{l=t}^{\infty} U(x_l, \mu(x_{s_k})), \tag{5}$$

where $U(x_t, \mu(x_{s_k}))$ is known as the utility function, which is supposed to satisfy $U(x, \mu) \geq 0$ and $U(0,0) = 0$. Therefore, we define $U(x_t, \mu(x_{s_k}))$ as follows:

$$U(x_t, \mu(x_{s_k})) = x_t^\mathsf{T} Q x_t + Y(\mu(x_{s_k})), \tag{6}$$

where $Q \in \mathbb{R}^{n \times n}$ is a symmetrical positive definite matrix, and $Y(\mu(x_{s_k}))$ is a positive semi-definite function that satisfies $Y(\mu(x_{s_k})) \geq 0$. Inspired by the work of [8, 21, 22], $Y(\mu(x_{s_k})))$ is designed as an integrand function:

$$Y = 2u_\mathrm{b} \int_0^{\mu(x_{s_k})} \varphi^{-\mathsf{T}}(\upsilon / u_\mathrm{b}) R d\upsilon, \tag{7}$$

where $\varphi^{-\mathsf{T}}(\cdot)$ stands for $(\varphi^{-1}(\cdot))^\mathsf{T}$, and $\varphi^{-1}(\cdot)$ is the inverse function of $\varphi(\cdot)$, which is an element-wise monotonic odd function and satisfies $\varphi(0) = 0$ and $|\varphi(\cdot)| \leq 1$. Without loss of generality, the hyperbolic tangent function $\tanh(\cdot)$ is adopted as a specific form of $\varphi(\cdot)$ in this paper. $R = \mathrm{diag}([r_1, \cdots, r_m]) \in \mathbb{R}^{m \times m}$ is a positive definite weight matrix, where $\mathrm{diag}(\cdot)$ reshapes the vector to a diagonal matrix.

The target of the optimal controller is to minimize the designed cost function (5) based on the state feedback. According to Bellman's principle of optimality, the optimal cost function $J^*(x_t)$ conforms to the DTHJB equation [8, 9]:

$$J^*(x_t) = \min_{\mu(x_{s_k})} \{U(x_t, \mu(x_{s_k})) + J^*(x_{t+1})\}. \tag{8}$$

Accordingly, the optimal control law $\mu^*(x_{s_k})$ at time instant $t$ is presented as:

$$\begin{aligned} \mu^*(x_{s_k}) &= \arg\min_{\mu(x_{s_k})} \{U(x_t, \mu(x_{s_k})) + J^*(x_{s_k+1})\} \\ &= u_b \varphi(D^*(x_{s_k})), \end{aligned} \tag{9}$$

where $D^*(x_{s_k})$ is described by:

$$D^*(x_{s_k}) = -\frac{1}{2u_b} R^{-1} \frac{\partial x_{s_k+1}}{\partial \mu(x_{s_k})} \lambda^*(x_{s_k+1}), \tag{10}$$

in which $\lambda^*(x_{s_k+1}) = \partial J^*(x_{s_k+1})/\partial x_{t+1}$ is the optimal costate function. It is noted that $\partial x_{s_k+1}/\partial \mu(x_{s_k})$ sometimes may not be directly available, and therefore system identification technique is utilized to approximate the dynamics and obtain the predicted state $\hat{x}_{t+1}$ by building an approximation of the plant, such as the model network [20] and the incremental model [11].

It is also noted that the optimal feedback control law $\mu^*(x_{s_k})$ employs the sampled state $x_{s_k}$ at the triggering instant $s_k$, rather than the current state $x_t$. With $\mu^*(x_{s_k})$, the optimal DTHJB equation is provided by:

$$J^*(x_t) = U(x_t, \mu^*(x_{s_k})) + J^*(x_{t+1}). \tag{11}$$

Obviously, the design of the triggering condition plays a significant role in determining appropriate triggering instants. Ahead of it, the following assumption is introduced [16]:

**Assumption 2.** *For system* (4)*, there exists a positive constant* $C \in (0, 0.5)$ *such that the following equation holds:*

$$||f(x_t, \mu(e_t + x_t))|| \leq C||x_t|| + C||e_t||, \tag{12}$$

*where* $||e_t|| \leq ||x_t||$.

Based on Assumption 2, a well-proved triggering condition in the literature [16] can therefore be introduced.

**Theorem 1.** *If Assumption 2 holds, the triggering condition can be defined as follows:*

$$||e_t|| > e_{Thr} = C\frac{1-(2C)^{t-s_k}}{1-2C}||x_{s_k}||, \tag{13}$$

*with which, the event-triggered system* (4) *is input-to-state stable and is asymptotically stable.*

*Proof.* Readers can refer to [16] for detailed proof and thus the complete proof is omitted here. □

It is worth noticing that the threshold value $e_{\mathrm{Thr}}$ is not unique because it changes as the triggered state $x_{s_k}$ and the designed triggering constant $C$ vary, and the triggering constant $C$ is usually chosen experimentally.

So far a general optimal event-triggered control scheme has been constructed, where an explicit triggering condition that can guarantee the system stability is provided to determine triggering instants.

Nevertheless, the optimal control problem has not been solved yet. The DTHJB equation (8) cannot be solved analytically in the generally nonlinear cases, and therefore the DHP algorithm with the aid of ANNs is introduced to obtain the numerical solution in the next section.

# 3 DHP Implementation with ANNs

In this section, an actor-critic scheme is designed with the facilitation of ANNs to iteratively approximate the solution of DTHJB. The structural diagram of DHP is illustrated in Fig. 1 [14], where a model network provides the state transition information in case of unknown dynamics. Specifically, the critic network approximates the derivative of the cost function that is denoted by $\lambda(x_t)$, which is the main character of the DHP technique. With the critic network, a near optimal control command is generated by substituting parameters into Eq. (9), which then is approximated by the actor network for controlling the real system. Both critic and actor networks are constructed with the full-connected feed-forward architecture, where a $\tanh(\cdot)$ function is adopted as the activation function for all hidden neurons. The whole control architecture incorporating the event-triggered scheme is introduced at the end of this section.

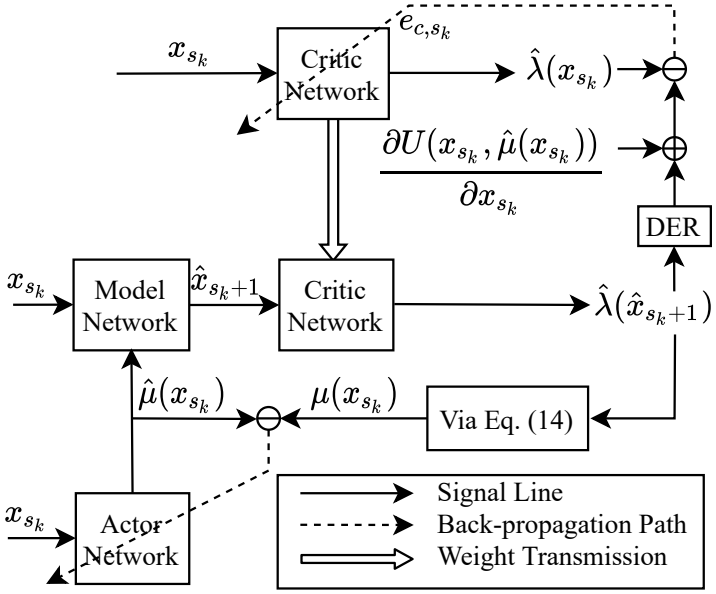


**Fig. 1 The DHP technique with an aided model network.**

## 3.1 The Critic Network

The architecture of the critic network is depicted in Fig. 2, in which the output of the critic network $\hat{\lambda}(\hat{w}_c, x_{s_k})$ is the approximation of the real $\lambda(x_{s_k})$, where $x_{s_k}$ is the input of the critic network and $\hat{w}_c$ is the weight of the critic network, which is the estimation of the ideal weight $w_c$.

Substituting $\hat{\lambda}(\hat{w}_c, x_{s_k})$ into Eqs. (9) and (10) can yield a near optimal control input as follows:

$$\mu(x_{s_k}) = u_b \varphi(-\frac{1}{2u_b} R^{-1} \frac{\partial \hat{x}_{s_k+1}}{\partial \mu(x_{s_k})} \hat{\lambda}(\hat{x}_{s_k+1})). \tag{14}$$

However, $\mu(x_{s_k})$ cannot directly be used for real system control, because of the so-called "bootstrap" property [8, 10]. Consequently, an actor network is introduced subsequently to generate an approximation of $\mu(x_{s_k})$, that is denoted by $\hat{\mu}(x_{s_k})$, and $\partial \hat{x}_{s_k+1}/\partial \mu(x_{s_k})$ is also replaced by $\partial \hat{x}_{s_k+1}/\partial \hat{\mu}(x_{s_k})$.

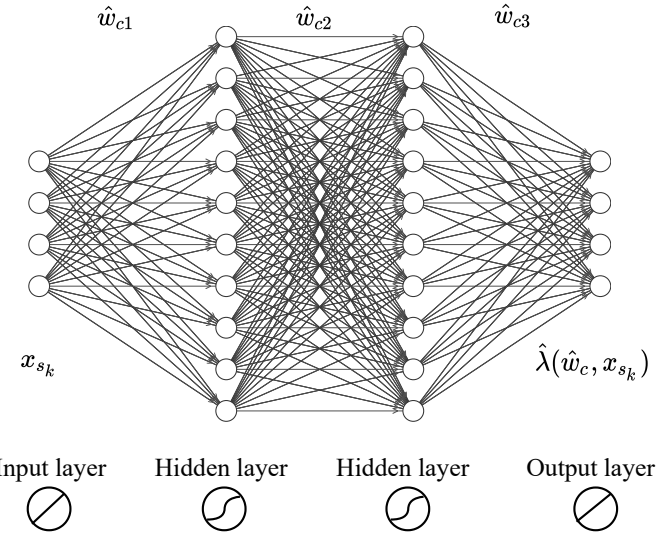


**Fig. 2 The structure of the actor network, where $\hat{w}_{c1}$, $\hat{w}_{c2}$ and $\hat{w}_{c3}$ are the weight matrices to be updated. It is assumed that there are 4 system states and the number of neurons in both hidden layers is 10. The presented structure is denoted by 4-10-10-4.**

The loss function of the critic network is formulated based on the time-differential error as follows:

$$E_{c,s_k} = \frac{1}{2} e_{c,s_k}^{\mathsf{T}} e_{c,s_k}, \tag{15}$$

$$e_{c,s_k} = \frac{\partial [\hat{J}(x_{s_k}) - U(x_{s_k}, \hat{\mu}(x_{s_k})) - \hat{J}(\hat{x}_{s_k+1})]}{\partial x_{s_k}}, \tag{16}$$

where $\hat{J}(x_{s_k})$ denotes the approximation of the real cost function $J(x_{s_k})$. According to the chain rule, Eq. (16) can be expanded as:

$$e_{c,s_k} = \hat{\lambda}(x_{s_k}) - 2Qx_{s_k} - \frac{\partial \hat{\mu}(x_{s_k})}{\partial x_{s_k}} \frac{\partial Y(\hat{\mu}(x_{s_k}))}{\partial \hat{\mu}(x_{s_k})} - \mathrm{DER} \cdot \hat{\lambda}(\hat{x}_{s_k+1}), \tag{17}$$

$$\mathrm{DER} = \frac{\partial \hat{x}_{s_k+1}}{\partial x_{s_k}} + \frac{\partial \hat{\mu}(x_{s_k})}{\partial x_{s_k}} \frac{\partial \hat{x}_{s_k+1}}{\partial \hat{\mu}(x_{s_k})}, \tag{18}$$

where $\partial \hat{\mu}(x_{s_k}) / \partial x_{s_k}$ is computed through the actor network that will be introduced in the next subsection, while $\partial \hat{x}_{s_k+1} / \partial x_{s_k}$ and $\partial \hat{x}_{s_k+1} / \partial \hat{\mu}(x_{s_k})$ are computed through the approximated model (or directly utilized if the system dynamics is known). As in [16], the partial derivative of $\hat{\mu}(x_{s_k})$ with respect to $x_{s_k}$, i.e., $\partial \hat{\mu}(x_{s_k}) / \partial x_{s_k}$, is taken into account in this paper, which theoretically outperforms [15, 20] in completeness and precision for this item being neglected in these papers.

Accordingly, the partial derivative of the critic loss $E_{c,s_k}$ with respect to the critic weight $\hat{w}_c$, i.e., $\partial E_{c,s_k} / \partial \hat{w}_c$, can be computed. Then, the critic weight $\hat{w}_c$ is updated by minimizing the critic loss $E_{c,s_k}$ through the gradient-decent algorithm with a learning rate $\eta_c > 0$.

## 3.2 The Actor Network

The actor network has two significant functions, namely generating a control input used for the closed loop and constructing a direct differentiable mapping from the state to the control input. Inspired by [16, 23], considering the input constraints, a bounding layer is involved to provide an absolute limitation, in which the aforementioned function $\varphi(\cdot)$ is adopted as the activation function. Fig. 3

demonstrates the architecture of the actor network, and the output of the actor network is represented by $\hat{\mu}(\hat{w}_a, x_{s_k}, \varphi, u_\mathrm{b})$, where the actor weight $\hat{w}_a$ is the estimation of the ideal weight $w_a$ and $x_{s_k}$ is the network input.

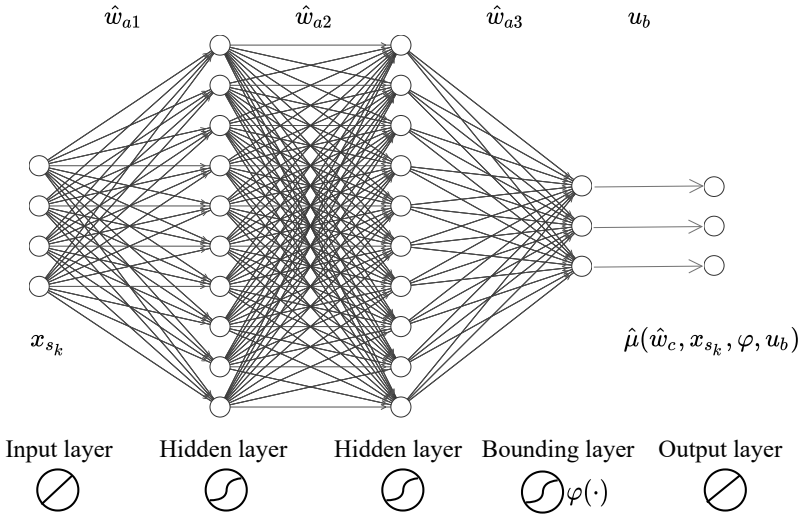


**Fig. 3 The structure of the actor network, where $\hat{w}_{a1}$, $\hat{w}_{a2}$ and $\hat{w}_{a3}$ are the weight matrices to be updated. It is assumed that there are 4 system states and 3 control inputs, and the number of neurons in both hidden layers is 10. The presented structure is denoted by 4-10-10-3.**

The output of the actor network $\hat{\mu}(\hat{w}_a, x_{s_k}, \varphi)$ should be as close to the target control $\mu(x_{s_k})$ that is defined in Eq. (14) as possible. Therefore the loss function to be minimized for the actor network can be presented by:

$$E_{a,s_k} = \frac{1}{2}[\hat{\mu}(x_{s_k}) - \mu(x_{s_k})]^\mathsf{T}[\hat{\mu}(x_{s_k}) - \mu(x_{s_k})]. \tag{19}$$

Similarly, with a learning rate $\eta_a > 0$, the actor weight $\hat{w}_a$ can be adjusted by the gradient-decent algorithm.

From the above analysis, it can be found that the updates of both networks rely on each other. To attain satisfying performance, multiple iterations are carried out at one time step and both networks are simultaneously updated once in an iterations. To implement this, the concept of "epoch" is thus introduced for policy iteration [10, 16]. In each epoch, the weight gradients of the critic and actor networks are computed based on the current system state $x_{s_k}$, and their weights are updated simultaneously after the computation. Then, the next epoch starts. Only the updated weights will be transmitted to the next epoch whereas the system state and the control input are kept unchanged. When a predesigned precision threshold is satisfied or the maximum epoch number is reached (this paper only considers the latter condition), the iteration stops, and the control input at the final epoch is used for updating the ZOH value.

The iteration improves the precision but increases the computational burden. Nevertheless, in the event-triggered scheme, only when a triggering instant is reached, will the DHP algorithm be activated and the control input be updated. Otherwise, the controller is not updated, which therefore can save computational load. In summary, the simple diagram of the event-triggered scheme incorporating the DHP technique is depicted in Fig. 4.

# 4 Simulation Study

In this section, a simulation study is carried out to illustrate the feasibility of the developed event-triggered DHP approach and compare its performance with the time-based approach that updates the control policy at each time instant. All simulations are conducted with Pytorch, an open source machine learning library for python [24]. The source code can be found at `https://github.com/sunbojason/`

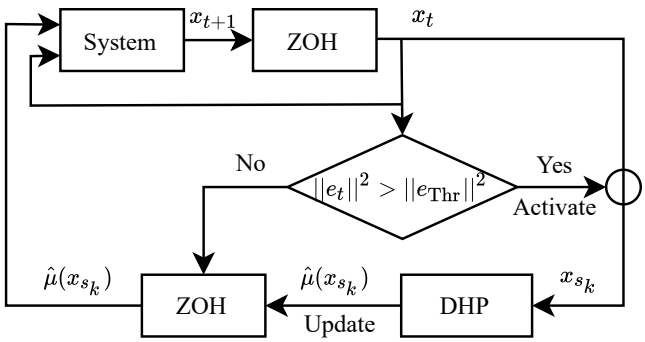


**Fig. 4 Simple diagram of the event-triggered DHP control approach (adapted from [16]).**

`Event-triggered-DHP`. The simulation is conducted in an online manner, which means that the control policy improves as it is applied to the real system [16].

**Remark 1.** *For a totally online learning, due to the random initialization of ANNs, the simulation results can differ at each run. This phenomenon can be mitigated or tackled by combining the offline training or involving a stable updating criterion [25].*

A second order nonlinear model of a launch vehicle [14, 26] is taken into account, which consists of the longitudinal force and moment equations, with angle of attack $\alpha$, pitch rate $q$ and elevator actuator $\delta e$ as system states, and the deflection command $\delta_e^c$ as the control input, i.e., $x = [x_1, x_2, x_3]^\mathsf{T} = [\alpha, q, \delta_e]^\mathsf{T}$ and $u = \delta_e^c$. In a valid flight envelope of $\alpha \in (-10^\circ, 10^\circ)$ and $M_a \in (1.8, 2.6)$, at an altitude of approximately 6000 meters, the nonlinear model of the launch vehicle around a steady wings-level flight condition can be formulated as:

$$\begin{aligned} \dot{\alpha} &= q + \frac{\bar{q}S}{mV_T} C_z(\alpha, q, M_a, \delta_e) \\ \dot{q} &= \frac{\bar{q}Sd_l}{I_{yy}} C_m(\alpha, q, M_a, \delta_e) \\ \dot{\delta}_e &= \frac{\delta_e^c - \delta_e}{\tau_e}, \end{aligned} \tag{20}$$

where $\bar{q}$, $S$, $m$, $V_T$, $d_l$, $I_{yy}$ are dynamic pressure, reference area, mass, speed, reference length and pitching moment of inertia respectively; $M_a$ is Mach number; $C_z$ and $C_m$ are the aerodynamic force and moment coefficients, respectively; $\tau_e$ is the time constant of the system. In this paper, we set $M_a = 2.0$ and $\tau_e = 0.1$ s, and set the sampling frequency is set to be 1 kHz, by which the system is discretized using Euler method [26]. It is assumed that the system is noisy on the state measurements, modeled as zero-mean white noises with standard derivation of $1.8 \times 10^{-3}$ deg, $3.0 \times 10^{-2}$ deg/s and $4 \times 10^{-2}$ deg, respectively [11].

Since the nonlinear dynamics cannot directly provide the derivative information, a model network is first of all trained offline to approximate the system dynamics. The model network is built with the structure of 4-10-10-3, with the current system state and control input as the network input and the predicted state as the network output [11]. The model network is trained using 2000 different data samples for 1000 epochs with a learning rate $\eta_m = 0.01$ and tested on another 500 samples. The data is collected by providing random states and control inputs within the feasible flight envelope to the nonlinear model. The identification errors of the model network on the testing set are illustrated in Fig. 5, where the mean sum of squares of the identification errors is below $1.0 \times 10^{-3}$ which demonstrates the high accuracy of approximation. After training, the model network is kept unchanged for controller design.

The actor and critic networks are constructed as 3-15-15-1 and 3-15-15-3, respectively. Their weights are randomly initialized within $[-0.05, 0.05]$ and updated 10 epochs at each time instant with $\eta_a = \eta_c = 0.001$ using the conventional stochastic gradient descent (SGD) method, because online control SGD can achieve safer behavior than Adam [27] that is utilized to update the model network. To satisfy the persistent excitation (PE) condition [11, 15] an exploration noise $u_e$ is introduced, which is a composition of decaying sinusoidal functions. It is noted that $u_e$ is added on the target control $\mu(x_{s_k})$ instead of the real control command $\hat{\mu}(x_{s_k})$ considering the saturation. The parameters in the utility function are selected as $Q = \mathrm{diag}([5, 2, 0])$ and $R = 2$. Through setting $C = 0.1$, we can accordingly obtain the triggering threshold $e_{\mathrm{Thr}}$ as:

$$e_{\mathrm{Thr}} = 0.1 \frac{1 - 0.2^{t - s_k}}{1 - 0.2} ||x_{s_k}||. \tag{21}$$

By initializing the system state as $x_0 = [10 \text{ deg}, 0, 0]^{\mathrm{T}}$ and setting $u_{\mathrm{b}} = 15$ deg, we carry out the online control simulation to verify the performance of the developed event-triggered DHP algorithm. The state and control command trajectories of the event-triggered approach and the time-based approach are displayed in Figs. 6 and 7. Comparing the event-triggered and time-based approaches, it can be observed that, for both approaches the system state eventually converges to to a small vicinity of the equilibrium point without obvious differences in accuracy and converge rate. Although the original control command outputted by event-triggered DHP is stepwise, the real elevator deflection is adequately smooth for the wing surface control after the signal going through the actuator. As depicted in Fig. 8, after a learning procedure where the ANN weight is updated online, the Frobenius norm [28] of weight matrices eventually converge to constant values for both networks, which demonstrates the convergence of the control policy.

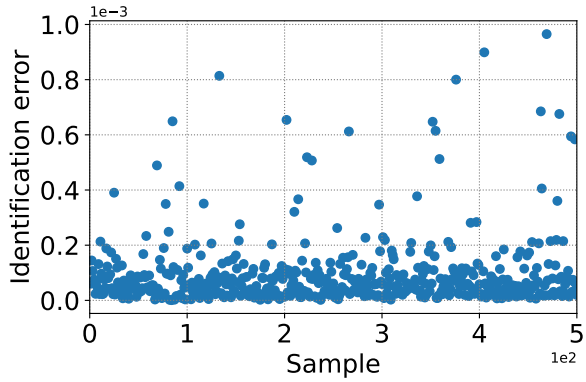


**Fig. 5 The mean sum of squares of the identification errors using the model network.**

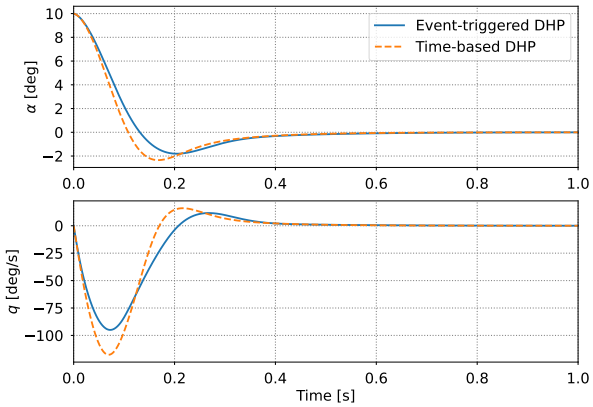


**Fig. 6 Launch vehicle longitudinal states.**

Furthermore, the triggering threshold trajectory is displayed in Fig. 9, which presents a trend to zero along with the event error. The inter-execution time is depicted in Fig. 10. One thousand samples are utilized by the time-based controller, whereas the proposed event-triggered approach only requires 420 samples. Therefore, the event-triggered method reduces the control updates in the learning process up to 58.0%, and thus improves the resource utilization. The simulation results collectively verify the feasibility and the effectiveness of the developed event-triggered DHP control approach.

# 5 Conclusion

In this paper, an intelligent optimal control algorithm called event-triggered dual heuristic programming (DHP) has been developed. Through the event-triggered scheme, the number of times the actor

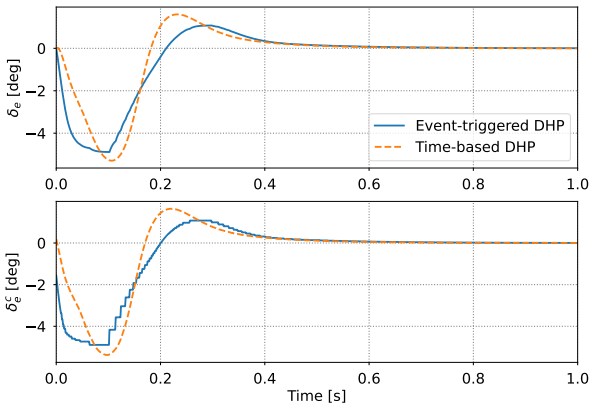


**Fig. 7 Elevator deflection and the control commands.**

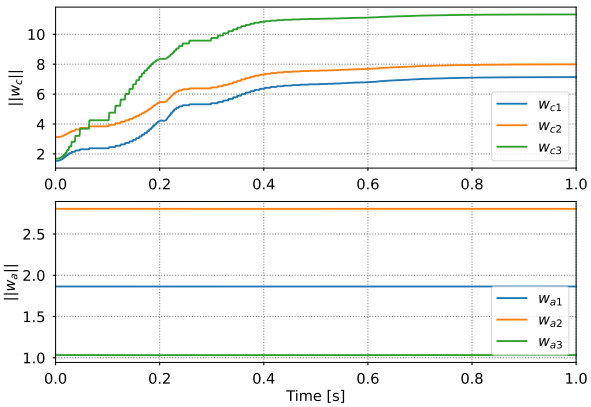


**Fig. 8 ANN weight norms.**

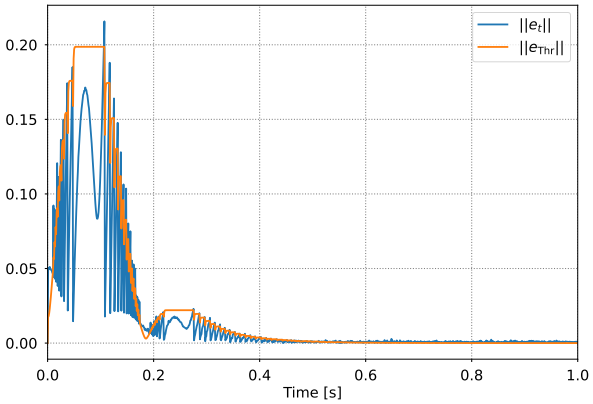


**Fig. 9 Evolution of the triggering condition.**

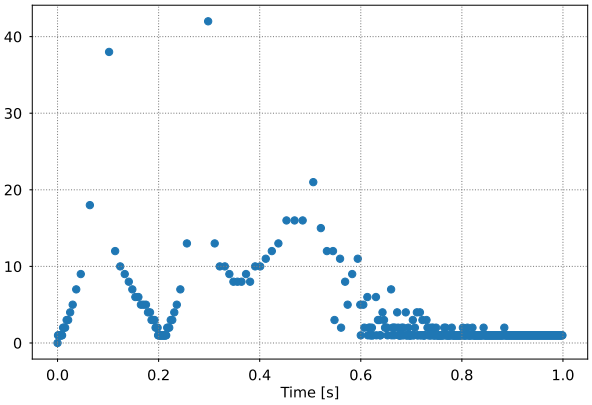


**Fig. 10 Trajectory portrait of ANN weight norms.**

and critic are updated can be largely reduced. Besides, the input constraints are coped with by the combination of an integral performance function and a bounding layer of the actor network. For illustrative verification, the proposed event-triggered DHP approach is applied to an online stabilization problem of a nonlinear second-order aerospace system. Compared with the time-based DHP approach, the experimental results indicate that the nonlinear system can successfully be stabilized with comparable performance. Furthermore, compared to the conventional time-based approach, the developed event-triggered approach can not only significantly reduce the computational burden up to 58%, but also save the communication's load between the controller and the system. Theses results collectively demonstrate the advantage of the proposed approach.

Nevertheless, the developed control scheme can be further improved before realistic applications. First of all, this paper concentrates on the stabilization problem while in numerous situations tracking control is required. Secondly, in this paper when the system dynamics are unknown, an offline trained model network is involved, which demands for collecting offline data. Therefore, the combination with online identification technique such as the incremental model for stronger adaptiveness is highly recommended for the next step.

# References

[1] Amalio Telenti, Ann Arvin, Lawrence Corey, Davide Corti, Michael S Diamond, Adolfo García-Sastre, Robert F Garry, Edward C Holmes, Phillip S Pang, and Herbert W Virgin. After the pandemic: perspectives on the future trajectory of covid-19. *Nature*, 596(7873):495–504, 2021. DOI: 10.1038/s41586-021-03792-w.

[2] D Liptakova, J Kolesar, and M Keselova. Challenges to the global aerospace industry due to the pandemic epidemy of covid-19. In *2020 New Trends in Aviation Development (NTAD)*, pages 155–158. IEEE, 2020. DOI: 10.1109/NTAD51447.2020.9379076.

[3] Xuerui Wang, Tigran Mkhoyan, and Roeland De Breuker. Nonlinear incremental control for flexible aircraft trajectory tracking and load alleviation. *Journal of Guidance, Control, and Dynamics*, 45(1):39–57, 2022. DOI: 10.2514/1.G005921.

[4] Julia Gebauer and Justus Benad. Flying V and reference aircraft evacuation simulation and comparison. *arXiv preprint arXiv:2102.06502*, 2021. DOI: /10.48550/arXiv.2102.06502.

[5] Tigran Mkhoyan, Nisarg Rashmin Thakrar, Roeland De Breuker, and Jurij Sodja. Design of a smart morphing wing using integrated and distributed trailing edge camber morphing. In *Smart Materials, Adaptive Structures and Intelligent Systems*, volume 84027, page V001T04A023. American Society of Mechanical Engineers, 2020. DOI: 10.1115/SMASIS2020-2370.

[6] Bahare Kiumarsi, Kyriakos G Vamvoudakis, Hamidreza Modares, and Frank L Lewis. Optimal and autonomous control using reinforcement learning: A survey. *IEEE transactions on neural networks and learning systems*, 29(6):2042–2062, 2018. DOI: 10.1109/TNNLS.2017.2773458.

[7] Brian L Stevens, Frank L Lewis, and Eric N Johnson. *Aircraft control and simulation: dynamics, controls design, and autonomous systems*. John Wiley & Sons, 2015.

[8] Bo Sun and Erik-Jan van Kampen. Reinforcement-learning-based adaptive optimal flight control with output feedback and input constraints. *Journal of Guidance, Control, and Dynamics*, 44(9):1685–1691, 2021. DOI: 10.2514/1.G005715.

[9] Asma Al-Tamimi, Frank L Lewis, and Murad Abu-Khalaf. Discrete-time nonlinear HJB solution using approximate dynamic programming: Convergence proof. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 38(4):943–949, 2008. DOI: 10.1109/ADPRL.2007.368167.

[10] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, second edition, 2018.

[11] Bo Sun and Erik-Jan van Kampen. Incremental model-based global dual heuristic programming with explicit analytical calculations applied to flight control. *Engineering Applications of Artificial Intelligence*, 89:103425, 2020. DOI: 10.1016/j.engappai.2019.103425.

[12] Jemin Hwangbo, Inkyu Sa, Roland Siegwart, and Marco Hutter. Control of a quadrotor with reinforcement learning. *IEEE Robotics and Automation Letters*, 2(4):2096–2103, 2017. DOI: 10.1109/LRA.2017.2720851.

[13] Tianhao Zhang, Gregory Kahn, Sergey Levine, and Pieter Abbeel. Learning deep control policies for autonomous aerial vehicles with mpc-guided policy search. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 528–535. IEEE, 2016. DOI: 10.1109/ICRA.2016.7487175.

[14] Ye Zhou, Erik-Jan van Kampen, and Qi Ping Chu. Incremental model based online dual heuristic programming for nonlinear adaptive control. *Control Engineering Practice*, 73:13–25, 2018. DOI: 10.1016/j.conengprac.2017.12.011.

[15] Ding Wang, Mingming Ha, and Junfei Qiao. Self-learning optimal regulation for discrete-time nonlinear systems under event-driven formulation. *IEEE Transactions on Automatic Control*, 65(3):1272–1279, 2020. DOI: 10.1109/TAC.2019.2926167.

[16] Bo Sun and Erik-Jan van Kampen. Event-triggered constrained control using explainable global dual heuristic programming for nonlinear discrete-time systems. *Neurocomputing*, 468(1):452–463, 2022. DOI: 10.1016/j.neucom.2021.10.046.

[17] Avimanyu Sahoo, Hao Xu, and Sarangapani Jagannathan. Near optimal event-triggered control of nonlinear discrete-time systems using neurodynamic programming. *IEEE transactions on neural networks and learning systems*, 27(9):1801–1815, 2016. DOI: 10.1109/TNNLS.2015.2453320.

[18] Michael Ouimet, David Iglesias, Nisar Ahmed, and Sonia Martínez. Cooperative robot localization using event-triggered estimation. *Journal of Aerospace Information Systems*, 15(7):427–449, 2018. DOI: 10.2514/1.I010600.

[19] Baolin Wu, Qiang Shen, and Xibin Cao. Event-triggered attitude control of spacecraft. *Advances in Space Research*, 61(3):927–934, 2018. DOI: 10.1016/j.asr.2017.11.013.

[20] Mingming Ha, Ding Wang, and Derong Liu. Event-triggered constrained control with DHP implementation for nonaffine discrete-time systems. *Information Sciences*, 519:110–123, 2020. DOI: 10.1016/j.ins.2020.01.020.

[21] Hamidreza Modares, Frank L Lewis, and Mohammad-Bagher Naghibi-Sistani. Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems. *Automatica*, 50(1):193–202, 2014. DOI: 10.1016/j.automatica.2013.09.043.

[22] Ali Heydari and Sivasubramanya N Balakrishnan. Finite-horizon control-constrained nonlinear optimal control using single network adaptive critics. *IEEE Transactions on Neural Networks and Learning Systems*, 24(1):145–157, 2013. DOI: 10.1109/TNNLS.2012.2227339.

[23] Bo Sun and Erik-Jan van Kampen. Intelligent adaptive optimal control using incrementalmodel-based global dual heuristic programming subject topartial observability. *Applied Soft Computing*, 103:107153, 2021. DOI: 10.1016/j.asoc.2021.107153.

[24] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019.

[25] Bo Sun, Xuerui Wang, and Erik-Jan van Kampen. Event-triggered intelligent critic control with input constraints applied to a nonlinear aeroelastic system. *Aerospace Science and Technology*, 120:107279, 2022. DOI: 10.1016/j.ast.2021.107279.

[26] Bo Sun and Erik-Jan van Kampen. Incremental model-based global dual heuristic programming for flight control. *IFAC-PapersOnLine*, 52(29):7–12, 2019. DOI: 10.1016/j.ifacol.2019.12.613.

[27] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. DOI: 10.48550/arXiv.1412.6980.

[28] Gene H Golub and Charles F Van Loan. *Matrix computations. Johns Hopkins studies in the mathematical sciences*. Johns Hopkins University Press, Baltimore, MD, 2nd edition, 1985.