Design and Evaluation of Advanced Intelligent Flight Controllers

Daniel Milz and Dr. Gertjan Looye

Reinforcement learning based methods could be feasible of solving adaptive optimal control problems for nonlinear dynamical systems. This work presents a proof of concept for applying reinforcement learning based methods to robust and adaptive flight control tasks. A framework for designing and examining these methods is introduced by means of the open research civil aircraft model (RCAM) and optimality criteria. A state-of-the-art robust flight controller - the incremental nonlinear dynamic inversion (INDI) controller - serves as a reference controller. Two intelligent control methods are introduced and examined. The deep deterministic policy gradient (DDPG) controller is selected as a promising actor critic reinforcement learning method that currently gains much attraction in the field of robotics. In addition, an adaptive version of a proportional-integral-derivative (PID) controller, the PID neural network (PIDNN) controller, is selected as the second method. The results show that all controllers are able to control the aircraft model. Moreover, the PIDNN controller exhibits improved reference tracking if a good initial guess of its weights is available. In turn, the DDPG algorithm is able to control the nonlinear aircraft model while minimizing a multi-objective value function. This work provides insight into the usability of selected intelligent controllers as flight control functions as well as a comparison to state-of-the-art flight control functions.

Design and Evaluation of Advanced Intelligent Flight Controllers

Daniel Milz* and Dr. Gertjan Looye†
DLR German Aerospace Center, Institute of System Dynamics and Control, 82234 Wessling, Germany

Reinforcement learning based methods could be feasible of solving adaptive optimal control problems for nonlinear dynamical systems. This work presents a proof of concept for applying reinforcement learning based methods to robust and adaptive flight control tasks. A framework for designing and examining these methods is introduced by means of the open research civil aircraft model (RCAM) and optimality criteria. A state-of-the-art robust flight controller - the incremental nonlinear dynamic inversion (INDI) controller - serves as a reference controller. Two intelligent control methods are introduced and examined. The deep deterministic policy gradient (DDPG) controller is selected as a promising actor critic reinforcement learning method that currently gains much attraction in the field of robotics. In addition, an adaptive version of a proportional-integral-derivative (PID) controller, the PID neural network (PIDNN) controller, is selected as the second method. The results show that all controllers are able to control the aircraft model. Moreover, the PIDNN controller exhibits improved reference tracking if a good initial guess of its weights is available. In turn, the DDPG algorithm is able to control the nonlinear aircraft model while minimizing a multi-objective value function. This work provides insight into the usability of selected intelligent controllers as flight control functions as well as a comparison to state-of-the-art flight control functions.

I. Introduction

Modern flight control strives for resilience, where control functions are retained despite occurring errors [1–3]. Major steps in this direction are robust and adaptive control. Together with optimal control, these form the basis of modern control theory where the ultimate goal seems to be an optimal and resilient control law.

Current flight control systems still revert back to basic modes in case of system malfunction or failures, or changes in aircraft behavior [4, 5]. In these cases, the controller reverts and gives back control authority to the pilot [6]. This implies increased dependencies on the pilot and workload for him. Ongoing research tries to overcome this issue by means of control reconfiguration, i.e. the adaptation of flight controllers to new circumstances [3, 7].

Contrary to the concept of decoupled control by means of linear, scheduled controllers, more and more efforts are being made to develop multiple-input-multiple-output (MIMO) and nonlinear controllers [2, 8–11]. Adaptive flight control, which can safely handle unexpected events, is a growing field of research [2, 12]. Previous approaches there are mostly based on the concept of indirect adaptive control, where a continuous system identification updates a dynamics model that us used inside a continuously updated optimal controller. A lot of research in this context has been done on nonlinear dynamic inversion (NDI) in combination with an online system identification [2, 13, 14]. This approach allows to handle changing system dynamics by updating an inverse model. With the introduction of incremental NDI (INDI), another step towards resilient control has been done [15–17], by further reducing the dependency on the system dynamics as shown in flight tests [7, 18]. Alternative concepts include direct adaptive control, where attempts are made to find an optimal control for the current conditions without a dedicated system identification [19, 20]. However, for nonlinear systems these adaptive control approaches may become very complex. One promising method to tackle such problems is reinforcement learning [21–24]. There, an attempt is made to iteratively find a control concept that meets certain optimality criteria.

Reinforcement learning and artificial intelligence is quickly evolving into a key instrument in control engineering. Although the idea of using intelligent control functions in terms of flight control function sounds promising, it was previously mostly applied in robotics or on high level functionalities like vision based object detection [25]. Until now, research regarding machine learning based control algorithms in terms of flight control functions [19, 20, 26] is scarce.

*Graduate Student, Department of Aircraft System Dynamics, Institute of System Dynamics and Control, German Aerospace Center.
†Head, Department of Aircraft System Dynamics, Institute of System Dynamics and Control, German Aerospace Center, AIAA member.

The basic idea behind reinforcement learning algorithms is inspired by behaviorist psychology: An agent interacts with its environment and iteratively learns the best or appropriate behavior through a reward and penalty optimization [27]. The fundamental procedure of reinforcement learning is illustrated in Fig. 1. In other words, for a given set of

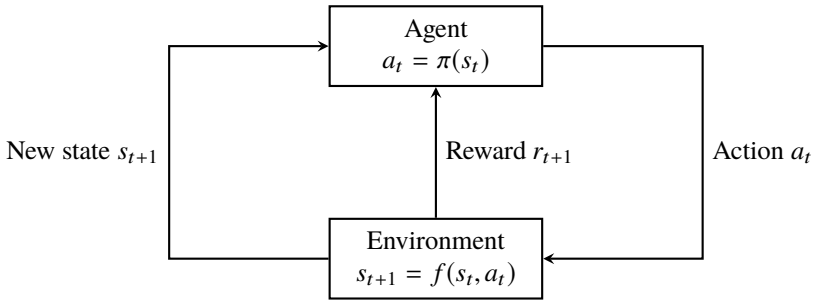


Fig. 1 Basic theory of reinforcement learning represented by an agent that interacts with a given environment by commanding actions based on the received reward and state.

possible states $\mathcal{S}$ and actions $\mathcal{A}$, an agent tries to optimize its policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$. A policy π is the strategy (or pattern) according to which the agent acts and chooses its next action. In general, the policy is iteratively updated (learned) by the agent. The policy is either defined in a stochastic manner as the probability of a state-action-pair (Eq. (1)), or as a deterministic policy that maps the current state to the preferred action (Eq. (2)). The respective definitions are given as:

$$\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1] \subset \mathbb{R}, \quad \pi(a|s) = P(a_t = a|s_t = s) \tag{1}$$

$$\pi : s_t \in \mathcal{S} \rightarrow a_t \in \mathcal{A} \tag{2}$$

with respect to an action a_t at time t concerning the current state s_t. In the scope of this work, deterministic policies are used. The learning relies on the maximization of a reward that is returned from the environment and depends on the current state and applied action. This can also be seen vice versa as the minimization of a loss or cost function as it is mostly done in optimal control. When experiencing a certain reward, the state-action-reward-tuple is used to improve the policy regarding the reward.

This article shows the design and validation of intelligent flight controllers on a nonlinear flight dynamics model. More precisely, the PIDNN and DDPG controller are designed and compared by means of the RCAM benchmark model. An INDI controller serves as a reference for comparison. This is intended to provide new insights on the practicability of reinforcement learning based methods for flight control

Therefore, design and validation criteria for intelligent (flight) control systems are introduced in Sect. II. The RCAM benchmark on robust flight control is described in Sect. III. Based on this, two intelligent flight controllers are designed in Sect. IV. We clarify by means of the results (Sect. V) in what way machine learning controllers may be advantageously applied in flight control tasks. For this, we compare the intelligent controllers with a state-of-the-art flight controller by means of the benchmark in Sect. V.

II. Design and validation criteria for intelligent control systems

For classical control methods, numerous design and validation criteria exist [28]. Classical flight control functions are designed to control aircraft dynamics by means of stabilization, damping poles, improving reference tracking et cetera. In contrast, there is no such thing for reinforcement learning concepts, especially neural-network based ones. This motivates the definition of new design and validation criteria for intelligent control methods. Ideally, a direct equivalent to the known criteria is desirable. However, for the moment no elaborated equivalent to classical criteria exists. In order to formulate these criteria, the characteristics of reinforcement learning are taken into account, i.e. the iterative optimization of a policy to satisfy the objective and minimize the loss. A loss or cost function is hence needed in order to form a validation and a design criterion for reinforcement learning control methods.

A. General design and validation criteria for intelligent control systems

An essential method for rating the control performance is a value function [29]. This value function $J : \mathbb{R}^n \rightarrow \mathbb{R}$ is introduced to rate the control performance quantitatively. For the assessment, a well-grounded selection of the rated

variables and the corresponding parameters is essential. An established method for control systems is to look at the tracking error $\mathbf{e} \in \mathbb{R}^m$ as well as the actuation energy that is directly related to the control input $\mathbf{u} \in \mathbb{R}^{n-m}$ [29]. The tracking error $\mathbf{e}$ is calculated as follows:

$$\mathbf{e} = \mathbf{r} - \mathbf{y_r} \tag{3}$$

with the reference signal $\mathbf{r} \in \mathbb{R}^m$ and the corresponding model outputs $\mathbf{y_r} \in \mathbb{R}^m$. These criteria are assessed over a period of time, i.e. from t_0 until t_{end}, via integration or summation. Therefore, parameters $a_1, ..., a_n \in \mathbb{R}$ are required in order to normalize, scale or weight each variable. The aim of a controller is then to minimize this value function J. A common value function for reinforcement learning is to use the weighted sum of the error distances and the control effort:

$$J = \int_{t_0}^{t_{\text{end}}} \|\mathbf{e}\|_{\mathbf{Q}}^2 + \|\mathbf{u}\|_{\mathbf{R}}^2 \; dt = \int_{t_0}^{t_{\text{end}}} \mathbf{e}^T \mathbf{Q} \mathbf{e} + \mathbf{u}^T \mathbf{R} \mathbf{u} \; dt \tag{4}$$

where $\mathbf{Q} \in \mathbb{R}^{m \times m}$ and $\mathbf{R} \in \mathbb{R}^{(m-n)\times(m-n)}$ are the weighting matrices w.r.t. the tracking error or the actuation. Eq. (4) is similar to the linear quadratic regulator (LQR) design objective described in [30]. In many cases, the weighting of couplings, i.e. non-diagonal entries in the weighting matrices, is not needed or unclear. Therefore, $\mathbf{Q}$ and $\mathbf{R}$ often reduce to diagonal matrices $\text{diag}(a_1, ..., a_m)$ and $\text{diag}(a_{m+1}, ..., a_n)$. In this case, J can also be written as follows:

$$J = \int_{t_0}^{t_{\text{end}}} \sum_{i=1}^{m} a_i \cdot e_i^2 + \sum_{i=1}^{n-m} a_{m+i} \cdot u_i^2 \; dt \tag{5}$$

B. Aircraft-specific design and validation criteria

Depending on the desired control goals plenty of possible value functions exist. For standard fixed wing aircraft configurations, aircraft dynamics can be assumed as decoupled in the longitudinal and lateral movement plane. For longitudinal dynamics, the control objective for example could be to track a specific flight path angle γ or pitch angle θ and velocity V, while reducing thrust δ_T and elevator deflection δ_E. Due to the cascading of various controllers inside a flight control system, another common objective is to track the pitch rate q. Here, the commanded signal is the output of a previous controller that transfers e.g. the flight path tracking error to a desired pitch rate command. In the scope of this work, a tracking of θ will be the aim. Lateral dynamics often aim at tracking the roll angle ϕ and minimizing the side slip angle β. This can mostly be realized by using ailerons δ_A and the rudder δ_R. In the inner loop, the roll rate p is commonly tracked. A tracking of ϕ will be done in this work.

III. Benchmark

For reinforcement learning, the OpenAI Gym [31] is a common benchmark. However, these models often lack physical near control tasks and flight control specific environments. This motivates the definition of a custom benchmark model and scenario.

A. Research Civil Aircraft Model

The GARTEUR Action Group on Robust Flight Control drafted a design challenge on a benchmark problem from 1995 to 1996. This RCAM problem is based on the automatic landing of a large, modern cargo aircraft and aims on testing and benchmarking novel methods on robust flight control. The contributions to these design challenges are collected in [32]. The RCAM model is based on the non-linear rigid-body aircraft equations of motion and is therefore chosen as the model for the reinforcement learning control benchmark.

B. Nonlinear Rigid-Body Aircraft Equations of Motion

The RCAM model is a 6 degrees of freedom model governing the following state vector:

$$\mathbf{x} = [p, q, r, \phi, \theta, \psi, u, v, w, x, y, z]^T \tag{6}$$

with the yaw rate r, pitch angle θ, yaw angle ψ. The velocities in north-east-down (NED) coordinated are denoted as u, v and w respectively. Furthermore, the current position resolved in NED axis are denoted as x, y and z. The input

vector is given as:

$$\mathbf{u} = [\delta_A, \delta_E, \delta_R, \delta_{T1}, \delta_{T2}]^T \tag{7}$$

with the rudder deflection δ_R and the throttle of the left and right engine, δ_{T1} and δ_{T2} respectively.

The aircraft has a total mass of 120 000 kg and a trimmed airspeed of 80 m s^{-1}, see [33] for the corresponding derivation. A more detailed description and explanation of the nonlinear rigid-body equations of motion for the RCAM model can be found in [34].

C. Incremental Nonlinear Dynamic Inversion

In order to compare the intelligent flight control functions, a non-learning flight controller is designed as a reference based on the incremental nonlinear dynamic inversion (INDI) approach. An INDI controller is chosen. The derivation of the according control laws can be found in [15], and the controller is implemented according to [7]. For a given over-actuated system

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x}) \cdot \mathbf{u} \tag{8}$$

$$\mathbf{y} = \mathbf{h}(\mathbf{x}) \tag{9}$$

the INDI control law is given as:

$$\Delta\mathbf{u} \approx (\nabla\mathbf{G}(\mathbf{x_0}))^{-1} (\nu - \dot{\mathbf{x}}_0) \tag{10}$$

$$\mathbf{u} = \mathbf{u_0} + \Delta\mathbf{u} \tag{11}$$

where $\nabla\mathbf{G}(x_0)$ denotes the Jacobian of the the function $\mathbf{G}$ at the current state $\mathbf{x_0}$, ν the pseudo control command representing the desired state derivative, $\mathbf{u}_0$ the current control command applied to the system and $\mathbf{u}$ the new control command for the system.

D. Benchmark Scenario

The benchmark consists of four scenarios. For the first one ideal circumstances apply, i.e. no disturbances and noise like wind and turbulence are present. In the second one, a Gaussian distributed noise with a mean $\mu = 1$ and variance $\sigma^2 = 0.05$ is multiplied to the plant's output signal. The third scenario introduces uncertainties to the model. In this case, the lift coefficient w.r.t. the angle of attack α, $C_{L,\alpha}$, and the pitch moment w.r.t. α, $C_{M,\alpha}$, are changed to $1.3 \cdot C_{L,\alpha}$ and $0.7 \cdot C_{M,\alpha}$. The choice of those two parameters is based on the circumstance that these are in general hard to estimate. The last scenario is a combination of both non-ideal cases. With these scenarios a basic rating of the controllers robustness can be done.

The benchmark maneuver was defined as two overlapping doublets for ϕ and θ as shown in Fig. 2. The magnitude of the different steps was set to 0.3 rad ≈ 17.2 deg for the lateral movement and 0.15 rad ≈ 8.6 deg for the longitudinal one. This was done in order to identify the controllers step response and corresponding characteristics. Furthermore, the coupling of longitudinal and lateral dynamics can be investigated, e.g. at time 5s, 15s or 30s.

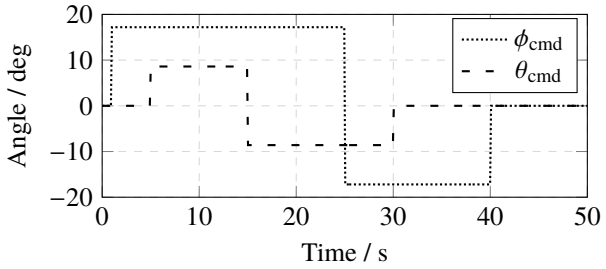


Fig. 2 The benchmark maneuver consists of two overlapping doublets for the roll angle ϕ and pitch angle θ.

IV. Advanced intelligent control methods

In the field of reinforcement learning, Q-Learning represents one basic concepts [35]. Q-Learning algorithms are able to learn an optimal control policy in a discrete action and state space. Discrete Q-learning is incapable of handling continuous control problems. Deep Q-Network (DQN) [36], promise to overcome this issue by using neural networks. There have already been great achievements with DQN [36–38]. In 2015, Lillicrap et al. published the DDPG algorithm [39] that is inspired by DQN. This reinforcement learning algorithm is capable of handling continuous state and action spaces, as well as high dimensional and non-linear dynamics. The DDPG further is a deterministic reinforcement learning method [39]. Due to those properties, the DDPG is well suited for the application as a flight control algorithm.

Another approach of bringing intelligent control methods into flight control is using artificial intelligence based versions of classical controllers. For instance, there is the PIDNN controller [40–42]. This controller is inspired by and based on the PID controller but consists of learning neurons that imitate the PID behaviour. The adaptation to new circumstances allows the usage on nonlinear and time-varying systems.

Representatively for intelligent control methods, we examine the PIDNN and DDPG. Both aim at the minimization of a given value function. While the PIDNN minimizes the tracking error, the DDPG algorithm aims at minimizing an arbitrary value function. The DDPG can be considered as an optimal control method. In the scope of this section, both algorithms are introduced and implemented for the benchmark.

A. PID Neural Network

PIDNN is an approach of combing PID control with biology-inspired neuronal networks to handle time-delayed linear systems, nonlinear systems, complex and vague systems [42]. A PIDNN controller consists of three layers filled with P, I and D neurons in a predefined structure shown in Fig. 3. The neurons behavior is inspired by neurological insights of various neuron models [41]. The neuronal weights are adjusted online by means of the backpropagation algorithm, which leads to the self-learning characteristic of the PIDNN [41]. The learning objective is to minimize the tracking error. In general, a PIDNN is a single-input-single-output (SISO) controller that can be extended to a MIMO controller, see [40, 43].

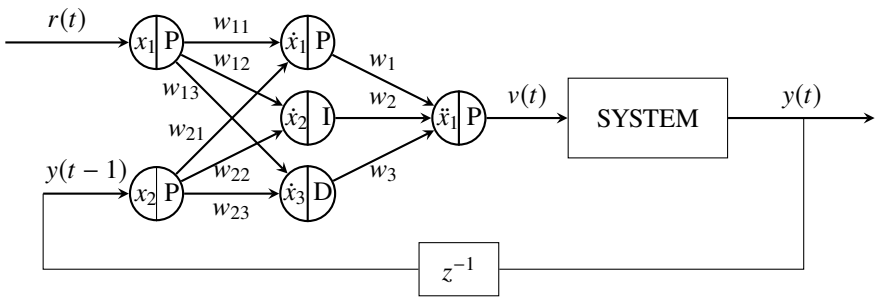


Fig. 3 2-3-1 structure of a PIDNN with the reference signal $r(t)$ at time t, the output $y(t)$, the control command $v(t)$ and the internal weights w.

Each neuron is defined by its input $u(t) \in \mathbb{R}$ and output $x(t) \in \mathbb{R}$ at time t, whereas $u(t)$ equals the weighted sum of all neurons inputs for neurons with multiple inputs. Thus, for a neuron in layer l the following formula for its input $u^l(t) \in \mathbb{R}$ with the weights matrix $W^{l-1,l} \in \mathbb{R}^{1\times n_{l-1}}$ holds:

$$u^l(t) = W^{l,l-1} \cdot x^{l-1}(t) \tag{12}$$

The different neuron types have different activation functions. For the sake of clarity, the input and outputs are denoted as $u_i(t)$ and $x_i(t)$ with $i \in \{P, I, D\}$ according to their neuron type. The activation functions are defined as:

$$
x_P(t) = \begin{cases} -1, & u_P(t) < -1 \\ u_P(t), & |u_P(t)| \leq 1 \\ 1, & u_P(t) > 1 \end{cases}, \qquad x_I(t) = \begin{cases} -1, & x_I(t) < -1 \\ x_I(t-1) + u_I(t), & |x_I(t)| \leq 1 \\ 1, & x_I(t) > 1 \end{cases},
$$
$$
x_D(t) = \begin{cases} -1, & x_D(t) < -1 \\ u_D(t) - u_D(t-1), & |x_D(t)| \leq 1 \\ 1, & x_D(t) > 1 \end{cases} \tag{13}
$$

with $x_i(t-1)$ being the neuron output at the previous time step and $u_i(t-1)$ as neuron input at the previous time step ($i \in \{P, I, D\}$). Aim of PIDNN is to minimize the value function by applying weight update terms. The value function J is defined according to Sect. II as:

$$
J = \sum_{t=1}^{N} \|e_t\|_2^2 = \frac{1}{N} \sum_{t=1}^{N} (r(t) - y(t))^2 \tag{14}
$$

with the error term $e_t = r(t) - y(t)$ at sample t, the desired system output (reference) $r(t)$, the real system output (feedback) $y(t)$ and N as the number of samples. Let η denote the learning rate, then minimization can be achieved by applying the following weight update in the last layer ($\dot{x}_i \rightarrow \ddot{x}_1, \forall i$):

$$
w_i(n+1) = w_i(n) - \eta \cdot \frac{\partial J}{\partial w_i} \approx w_i(n) + \eta \cdot \frac{2}{N} \cdot \sum_{t=1}^{N} \left[e_t \cdot \frac{y(t) - y(t-1)}{v(t) - v(t-1)} \cdot \dot{x}_i(t) \right] \tag{15}
$$

And the corresponding weight update in the former layer ($x_i \rightarrow \dot{x}_j, \forall i, j$):

$$
w_{ij}(n+1) = w_{ij}(n) - \eta \cdot \frac{\partial J}{\partial w_{ij}} \approx w_{ij}(n) + \eta \cdot \sum_{t=1}^{N} \left[\delta_j(t) \cdot \frac{\dot{x}_j(t) - \dot{x}_j(t-1)}{\dot{u}_j(t) - \dot{u}_j(t-1)} \cdot w_j \cdot x_i(t) \right] \tag{16}
$$

with $\delta_i(t) = \frac{\partial J}{\partial w_i}$.

Implementation of the PIDNN Controller. The PIDNN algorithm is implemented according to the previous description with a slight modification to fit the desired use of online tuning. This is done by not updating the weights w.r.t. to the last N samples or more general to N random samples but by updating the weights iteratively with the current state and action tuple. Fig. 4 shows the block diagram structure of the forward and backpropagation path of the PIDNN. The learning rate of the weights η is set to 10^{-8}. A single PIDNN controller is used as a transfer function of the current and desired rate p or q to the commanded actuator deflection. The rate values are a direct output of the model. The desired value is created by an outer loop controller transferring the error of the current attitude through a proportional element to the desired rate value. Furthermore, a reference model filtering the attitude command is used. The reference model is implemented as a PT_2 element and parametrized according to the model bandwidths.

B. Deep Deterministic Policy Gradient

The DDPG has proven to be suited for a wide range of control problems [39]. DDPG is inspired by DQN [36], but allows the use of continuous action and state spaces. DDPG is an actor-critic method [44] with two neural networks representing the policy (Actor) and value function (*Critic*) and two additional neural networks as target networks. Fig 5 shows the layout and interaction of an actor-critic agent with its environment. The target network weights are updated in the same direction as the primary network but slower. This is done to prevent the system from becoming unstable due to oscillatory changes in the value prediction [39]. The actor predicts an action based on the state given as input, while the critic predicts the value of the current state-action tuple. Since just the actual (desired) reward is available from the environment, and not the best possible action, the actor is updated using the gradient of the critics value prediction with respect to the action.

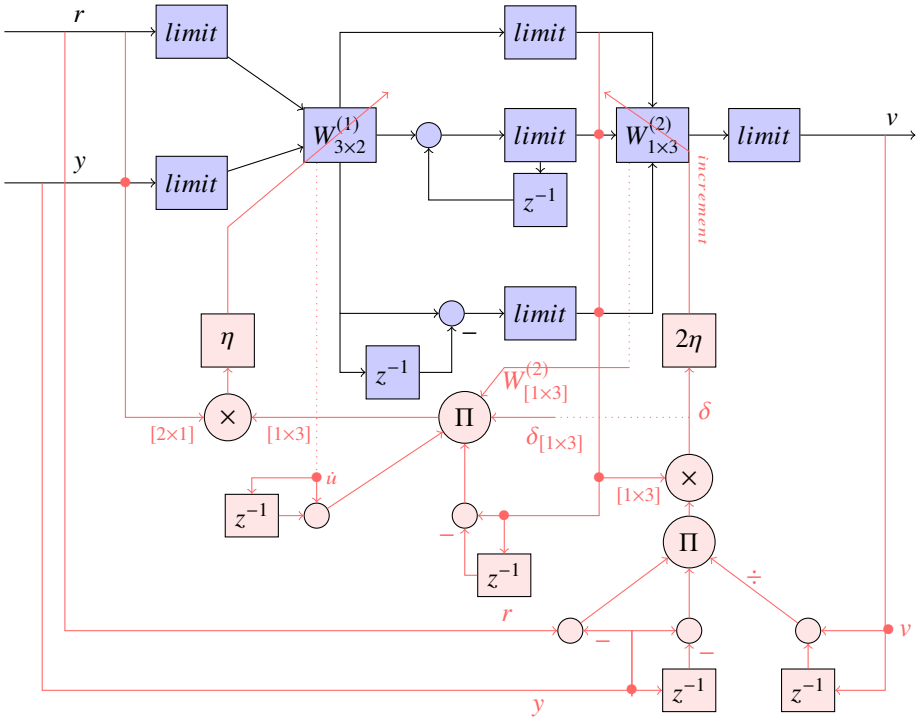


Fig. 4 **Control structure of an PIDNN controller with forward and backpropagation path.**

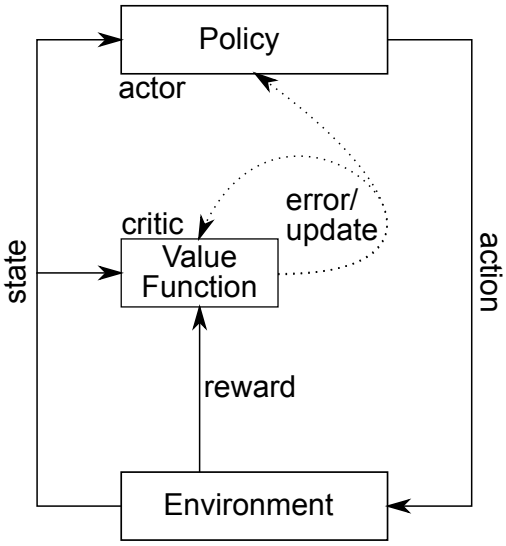


Fig. 5 **Actor-Critic architecture as described in [44].**

Implementation of the DDPG Algorithm. The DDPG algorithm is implemented using the open source machine learning library *TensorFlow* [45]. The DDPG is made up of an actor and a critic network. Both networks are implemented according to [39], with slight modifications to fit the aimed use. The most notable modification is the exchange of the first dense, fully connected layer with a recurrent layer in order to include dynamical behavior into the network. Furthermore, the output saturation is moved from the network to the model in order to prevent possible vanishing gradients. The remaining control structure is similar to the PIDNN one. The value function being iteratively optimized by the DDPG controller is an essential tuning parameter. Thus a lot of effort can be put into the selection of this function - here denoted as $dJ(t)$. For this benchmark dJ is defined as:

$$dJ(t) = \left\| \begin{bmatrix} q - q_{\text{ref}} \\ \delta_E \\ \delta_E - \delta_{E,\text{cmd}} \end{bmatrix} \right\|_Q^2, \quad Q = \begin{bmatrix} 1.0 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.5 \end{bmatrix} \tag{17}$$

This can be flattened to:

$$dJ(t) = 1.0 \cdot (q - q_{\text{ref}})^2 + 0.1 \cdot (\delta_E)^2 + 0.5 \cdot \left(\delta_E - \delta_{E,\text{cmd}}\right)^2 \tag{18}$$

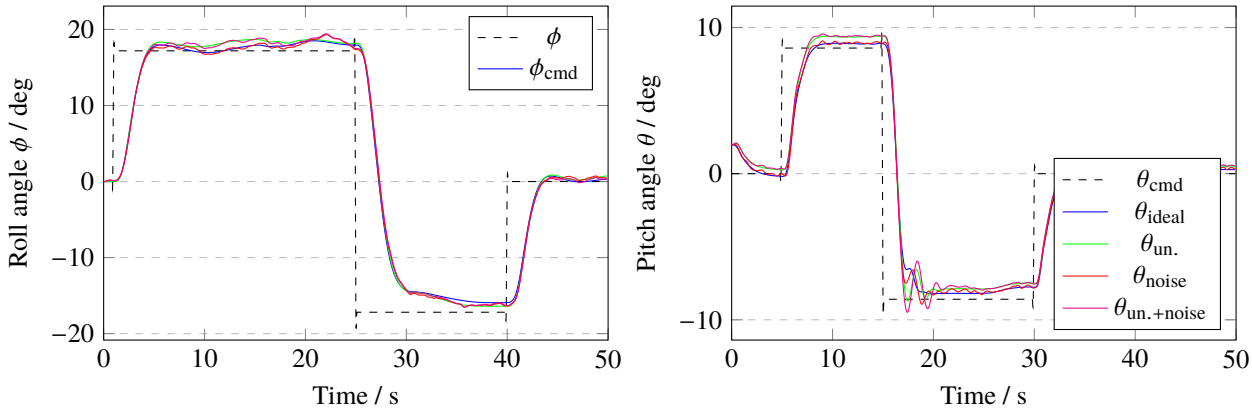
The first term in Eq. (18) is the main term forcing the controller to track q. The second term minimize the applied actuation energy. By choosing the weight smaller than the tracking error weight, the priority is set to track the reference. The last term penalizes unrealistic command signals, for example commands outside the actuator limit or its bandwidth. The latter terms are also crucial to avoid oscillating or bang-bang control signals.

V. Evaluation Results

The designed controllers are examined by means of the benchmark scenarios defined in Sect. III.

A. PID Neural Network Controller

Figure 6a and Figure 6b show the performance of the PIDNN controller when initializing its weights with the final weights of a randomly-initialized training run.



(a) Lateral **(b) Longitudinal**

Fig. 6 Combined simulation results of the PIDNN controller on the different scenarios (ideal, with model uncertainties, with noise, with model uncertainties and noise)

The lateral movement plotted in Fig. 6a is smooth and has figuratively no overshoots. However, the roll angle did not settle to the desired value at time 30 s to 40 s. This is owed to a previous wrong weight update where the update process went into a local minimum. The observed behavior comes most likely from a too weak integrative or too strong derivative behavior. When looking at the actuator signal of the PIDNN shown in Fig. 7 a smooth trend can be seen. The utilized actuator signal is small and suggests minimal actuation energy. This shows that the controller did not tend to oscillatory or bang-bang control behavior.

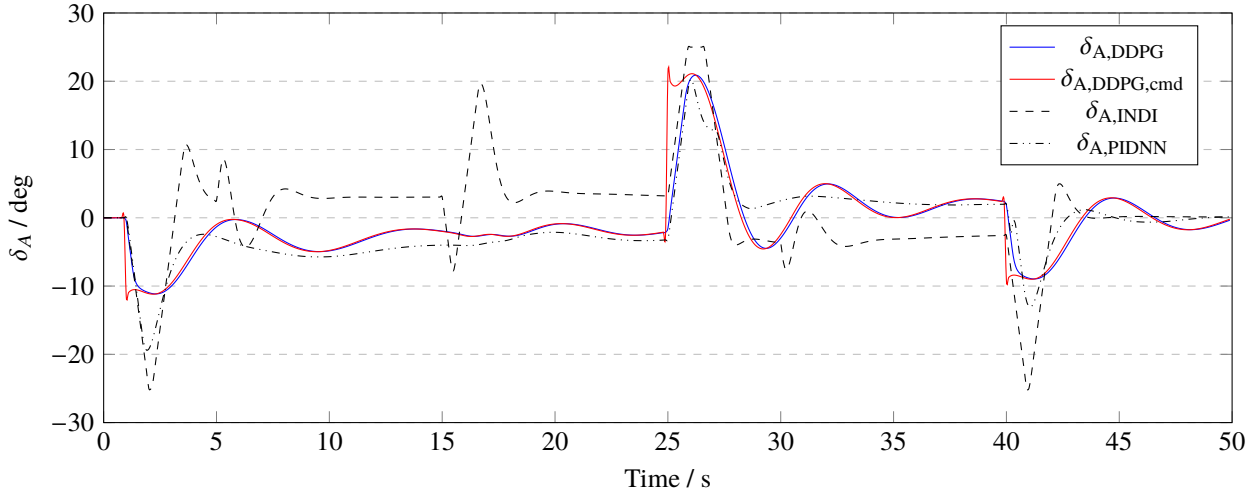


Fig. 7 Aileron actuator deflections for the different controllers including the commanded and measured aileron deflection of the DDPG.

The longitudinal movement shown in Fig. 6b does show a slight constant offset. This is most likely owed to the same reason as previously described. Thus local optimization minima seem to be an issue of the PIDNN controller. The

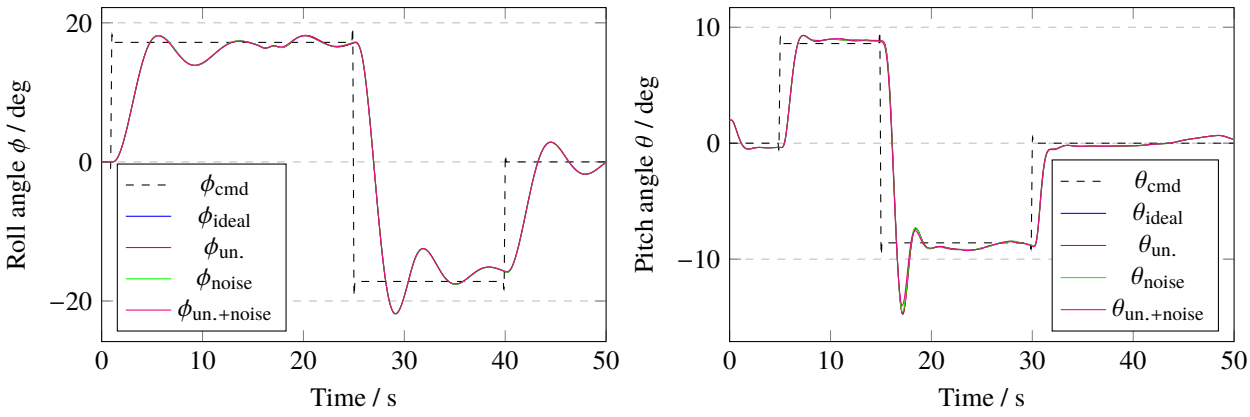
settling takes place in a few seconds (circa 4 s each) with an overshoot of around 1° to 2° depending on the scenario. This shows that the PIDNN controller is able to handle uncertain dynamics and noise, but with a slight performance impact.

The major insight was that a good initial value for the weights is needed. Otherwise, the controller was not constantly stable.

Furthermore, the result suggests to consider the PIDNN controller as a worthy alternative to the classical linear PID controller when applied to non-linear models. More information on this is stated in [46].

B. DDPG Controller

Fig. 8a and 8b show the lateral and longitudinal movement of the benchmark aircraft when using the DDPG controller according to the previous section. In Fig. 8 can be seen that the responses to the different scenarios hardly differ. This finding suggests that the controller is hardly affected by model uncertainties and noise.



(a) Lateral **(b) Longitudinal**

Fig. 8 Combined simulation results of the DDPG controller on the different scenarios (ideal, with model uncertainties, with noise, with model uncertainties and noise)

The lateral case shows that the DDPG is capable of stably tracking the doublet. Thus, a proof of function can be made. However, the results show oscillatory transient behavior. This trend is present in all scenarios suggesting a trained behavior. This behavior can occur if the optimization get stuck in a local minima and no longer improves. Further research on this is still needed in order to reliably bypass this error. When considering Fig. 7 and the minimized actuator energy applied during this benchmark, one possible cause could be the trade-off between minimizing the tracking error and the actuation. The DDPG controller does not overutilize the actuator but is trying to minimize the actuator load. Synoptically, the DDPG controller showed a stable reference tracking with minimal actuation energy in the lateral movement.

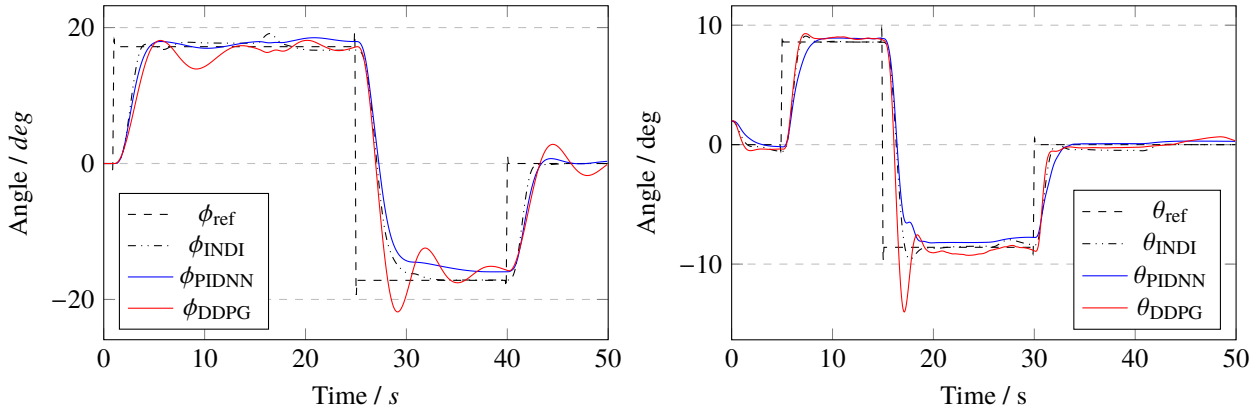
Figure 8b shows the longitudinal movement with an overshoot of around 5° in all scenarios around time 15 s. This finding indicates that this error is again owed to a faulty training. The actuator command there exceeded and stopped too late. This is probably due to the steep flak in the commanded signal. Besides this overshoot, no notable outliers can be identified. In the further trend, the θ signal tracks the command well.

The DDPG algorithm especially convinced through capability of minimizing a multi-objective value function. Although the PIDNN controller is able to handle non-linear systems as well, it is designed to minimize the tracking error, whereas the objective of the DDPG controller has to be specified by the designer. This may lead to a better overall performance.

C. Comparison

An overview on how each controller performed at the benchmark test is given in Fig. 9. The results are again separated into lateral and longitudinal movements. The scenario that is shown in this figure is the ideal one, since the fundamental trend is similar for all scenarios within one controller. Additionally, the results of using INDI control are plotted for comparison and as a non-learning reference controller. The reference controller shows best performance,

except the coupling of the lateral and longitudinal movement at time 15s.



(a) Lateral **(b) Longitudinal**

Fig. 9 Comparison of all applied controllers at the RCAM model.

In the lateral case (Fig. 9a) all controllers exhibit a working proof of concept. The lateral command could be tracked in a stable manner in all cases. The INDI and PIDNN both show an better overall performance than the DDPG controller which exhibits oscillatory behavior throughout the simulation. The DDPG controller had constantly oscillating movement which makes the controller worse. Nevertheless, these issues could possibly be overcome by improving the training and the value function and adapting the neural networks and hyperparameters. The proof of function could be provided by all controllers.

For the longitudinal case (Fig. 9b), the results were comparable to the lateral case. Both, INDI and PIDNN, showed mostly similar behavior. The DDPG showed a good performance without oscillatory behavior, but with an overshoot.

When looking at the actuator signal represented by the aileron deflection in Fig. 7 it can be seen, that the designed INDI controller utilizes the actuator the most. The DDPG and PIDNN show similar results regarding the actuation. They also issue relatively smooth actuator commands since the actuator dynamics are taken into account.

VI. Conclusion and Outlook

The results show that all compared control methods are able to control the benchmark model. A proof of function could thus be completed for both intelligent flight control algorithms.

The PIDNN turned out as a learning alternative to the well-established PID controller. Its main advantage lies in the fact that the PIDNN controller is adapting online to changing dynamics. However, these changes have to be slow in comparison to the PIDNN update rate to be handled well. Furthermore, it is important to initialize the weights with a good initial value in order to have stable behavior. This can either be realized with a preceding run or by translating PID parameters into PIDNN weights. These prerequisite are fulfilled. The results show that the PIDNN controller initialized with a well-grounded initial parameter setting quickly adapts to a reference signal with less errors. Finally, the learning process tends to run into local minima. This issue can be seen in the training runs and should be investigated in more detail.

The DDPG controller showed primarily a good control performance. Although the achieved results are worse compared to the other approaches, a proof of function was successfully performed. The controller handles uncertainties and noise with hardly no performance loss. However, Beyond this first study, this approach seems to be promising for further research, yielding improved performance.

The two intelligent controllers show that artificial intelligence methods may influence future flight control functions. Furthermore, it has to be noted that those two control functions do not represent the whole field of intelligent controllers. There are many more alternatives that have to be tested in order to find a well-fitting future control method. Especially methods from the field of dynamic programming seem to be promising as well. Nevertheless, the results show that the selected controllers are feasible of stably controlling a nonlinear aircraft model.

Future research could continue to explore alternative intelligent control approaches for the use as flight controllers. Furthermore, more research on promising reinforcement learning controllers can be conducted including well-grounded

robustness analysis, flight tests, and should also investigate the performance gain after optimizing the selected control functions. It is also worth to put more effort in researching the PIDNN controller as a PID alternative that adapts to new environmental conditions and improves uncertainty and failure handling.

Reinforcement learning methods will be a future focus of research in the institute, particularly w.r.t. fault tolerant and robust control, with the ultimate goal of flight testing.

References

[1] Looye, G., and Joos, H.-D., "Design of Robust Dynamic Inversion Control Laws using Multi-Objective Optimization," *Proc. of Guidance, Navigation, and Control Conference and Exhibit*, 2001. URL https://elib.dlr.de/12214/, lIDO-Berichtsjahr=2001,.

[2] Lombaerts, T., Huisman, H., Chu, P., Mulder, J. A., and Joosten, D., "Nonlinear Reconfiguring Flight Control Based on Online Physical Model Identification," *Journal of Guidance, Control, and Dynamics*, Vol. 32, No. 3, 2009, pp. 727–748. doi:10.2514/1.40788.

[3] Lombaerts, T., Looye, G., Seefried, A., Neves, M., and Bellmann, T., "Proof of concept simulator demonstration of a physics based self-preserving flight envelope protection algorithm," *Engineering Applications of Artificial Intelligence*, Vol. 67, 2018, pp. 368–380. doi:10.1016/j.engappai.2017.08.014, URL https://linkinghub.elsevier.com/retrieve/pii/S0952197617301987.

[4] Goupil, P., Boada-Bauxell, J., Marcos, A., Cortet, E., Kerr, M., and Costa, H., "AIRBUS efforts towards advanced real-time Fault Diagnosis and Fault Tolerant Control," *IFAC Proceedings Volumes*, Vol. 47, No. 3, 2014, pp. 3471–3476. doi:10.3182/20140824-6-za-1003.01945.

[5] Balas, G. J., "Flight Control Law Design: An Industry Perspective," *European Journal of Control*, Vol. 9, No. 2, 2003, pp. 207–226. doi:10.3166/ejc.9.207-226.

[6] Goupil, P., and Marcos, A., "Advanced Diagnosis for Sustainable Flight Guidance and Control: The European ADDSAFE Project," *SAE Technical Paper Series*, SAE International, 2011. doi:10.4271/2011-01-2804.

[7] Grondman, F., Looye, G., Kuchar, R. O., Chu, Q. P., and Van Kampen, E.-J., "Design and Flight Testing of Incremental Nonlinear Dynamic Inversion based Control Laws for a Passenger Aircraft," *2018 AIAA Guidance, Navigation, and Control Conference*, American Institute of Aeronautics and Astronautics, Kissimmee, Florida, 2018, p. 0385. doi:10.2514/6.2018-0385, URL https://elib.dlr.de/124710/, aIAA SciTech Forum.

[8] Weiser, C., Ossmann, D., Kuchar, R., and Looye, G., "Design and Verification of a Linear Parameter Varying Control Law for a Transport Aircraft," *5th CEAS Specialist Conference on Guidance, Navigation & Control*, 2019. URL https://elib.dlr.de/127005/.

[9] Ossmann, D., Luspay, T., and Vanek, B., "Baseline Flight Control System Design for an Unmanned Flutter Demonstrator," *2019 IEEE Aerospace Conference*, IEEE, 2019, pp. 1–10.

[10] Lombaerts, T., and Looye, G., "Design and flight testing of nonlinear autoflight control laws," *AIAA Guidance, Navigation, and Control Conference*, American Institute of Aeronautics and Astronautics, Minneapolis, Minnesota, 2012, pp. 315–24. doi:10.2514/6.2012-4982.

[11] Keijzer, T., Looye, G., Chu, Q. P., and Van Kampen, E.-J., "Design and Flight Testing of Incremental Backstepping based Control Laws with Angular Accelerometer Feedback," *AIAA Scitech 2019 Forum*, American Institute of Aeronautics and Astronautics, Reston, Virginia, 2019, p. 0129. URL https://elib.dlr.de/128283/.

[12] Lombaerts, T. J. J., Van Oort, E. R., Chu, Q. P., Mulder, J. A., and Joosten, D. A., "Online Aerodynamic Model Structure Selection and Parameter Estimation for Fault Tolerant Control," *Journal of Guidance, Control, and Dynamics*, Vol. 33, No. 3, 2010, pp. 707–723. doi:10.2514/1.47256.

[13] Lombaerts, T., Looye, G., Chu, Q., and Mulder, J., "Design and simulation of fault tolerant flight control based on a physical approach," *Aerospace Science and Technology*, Vol. 23, No. 1, 2012, pp. 151–171. doi:10.1016/j.ast.2011.07.004, URL https://linkinghub.elsevier.com/retrieve/pii/S127096381100112X.

[14] Sonneveldt, L., Van Oort, E. R., Chu, Q. P., and Mulder, J. A., "Nonlinear Adaptive Trajectory Control Applied to an F-16 Model," *Journal of Guidance, Control, and Dynamics*, Vol. 32, No. 1, 2009, pp. 25–39. doi:10.2514/1.38785.

[15] Sieberling, S., Chu, Q. P., and Mulder, J. A., "Robust Flight Control Using Incremental Nonlinear Dynamic Inversion and Angular Acceleration Prediction," *Journal of Guidance, Control, and Dynamics*, Vol. 33, No. 6, 2010, pp. 1732–1742. doi:10.2514/1.49978.

[16] Bacon, B., and Ostroff, A., "Reconfigurable flight control using nonlinear dynamic inversion with a special accelerometer implementation," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, American Institute of Aeronautics and Astronautics, 2000. doi:10.2514/6.2000-4565.

[17] Simplício, P., Pavel, M., van Kampen, E., and Chu, Q., "An acceleration measurements-based approach for helicopter nonlinear flight control using Incremental Nonlinear Dynamic Inversion," *Control Engineering Practice*, Vol. 21, No. 8, 2013, pp. 1065–1077. doi:10.1016/j.conengprac.2013.03.009.

[18] Keijzer, T., "Design and Flight Testing of Incremental Control Laws using Angular Accelerometer Feedback on a CS-25 Aircraft," Master's thesis, Delft University of Technology, Sep. 2018. URL https://elib.dlr.de/125860/.

[19] Kim, B. S., and Calise, A. J., "Nonlinear Flight Control Using Neural Networks," *Journal of Guidance, Control, and Dynamics*, Vol. 20, No. 1, 1997, pp. 26–33. doi:10.2514/2.4029.

[20] 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.

[21] Song, R., Lewis, F. L., Wei, Q., and Zhang, H., "Off-Policy Actor-Critic Structure for Optimal Control of Unknown Systems With Disturbances," *IEEE Transactions on Cybernetics*, Vol. 46, No. 5, 2016, pp. 1041–1050. doi:10.1109/TCYB.2015.2421338.

[22] Sutton, R. S., Barto, A. G., and Williams, R. J., "Reinforcement learning is direct adaptive optimal control," *IEEE Control Systems Magazine*, Vol. 12, No. 2, 1992, pp. 19–22. doi:10.1109/37.126844.

[23] Lewis, F. L., Vrabie, D., and Vamvoudakis, K. G., "Reinforcement Learning and Feedback Control: Using Natural Decision Methods to Design Optimal Adaptive Controllers," *IEEE Control Systems Magazine*, Vol. 32, No. 6, 2012, pp. 76–105. doi:10.1109/MCS.2012.2214134.

[24] Vamvoudakis, K. G., and Lewis, F. L., "Online actor–critic algorithm to solve the continuous-time infinite horizon optimal control problem," *Automatica*, Vol. 46, No. 5, 2010, pp. 878–888. doi:10.1016/j.automatica.2010.02.018, URL http://www.sciencedirect.com/science/article/pii/S0005109810000907.

[25] Carrio, A., Sampedro, C., Rodriguez-Ramos, A., and Campoy, P., "A Review of Deep Learning Methods and Applications for Unmanned Aerial Vehicles," *Journal of Sensors*, Vol. 2017, 2017, pp. 1–13. doi:10.1155/2017/3296874.

[26] Zhou, Y., "Online reinforcement learning control for aerospace systems," Ph.D. thesis, Technische Universiteit Delft, Apr. 2018. doi:10.4233/uuid:5b875915-2518-4ec8-a1a0-07ad057edab4.

[27] Tokic, M., "Reinforcement Learning mit adaptiver Steuerung von Exploration und Exploitation," Ph.D. thesis, Universität Ulm, 2013. doi:10.18725/oparu-2517.

[28] Dorf, R. C., and Bishop, R. H., *Modern control systems*, 12th ed., Pearson, 2011.

[29] Schultz, W. C., and Rideout, V. C., "Control system performance measures: Past, present, and future," *IRE Transactions on Automatic Control*, Vol. AC-6, No. 1, 1961, pp. 22–35. doi:10.1109/TAC.1961.6429306.

[30] Kalman, R. E., et al., "Contributions to the theory of optimal control," *Bol. soc. mat. mexicana*, Vol. 5, No. 2, 1960, pp. 102–119.

[31] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W., "OpenAI Gym," *ArXiv*, 2016.

[32] Magni, J.-F., Bennani, S., and Terlouw, J. (eds.), *Robust Flight Control*, Springer Berlin Heidelberg, 1997. doi:10.1007/bfb0113842.

[33] Looye, G., Varga, A., Bennani, S., Moormann, D., and Grübel, G., "Robustness Analysis Applied to Autopilot Design, Part 1: mu-Analysis of Design Entries to a Robust Flight Control Benchmark," *21st Congress of the International Council of Aeronautical Sciences*, 1998. LIDO-Berichtsjahr=1999,.

[34] GARTEUR, F. M. G. ., "Robust Flight Control Design Challenge Problem Formulation and Manual: the Research Civil Aircraft Model (RCAM)," Tech. rep., GARTEUR, 1995. URL http://www.garteur.org/Technical%20Reports/FM_AG-08_TP-088-3.pdf.

[35] Watkins, C. J. C. H., "Learning from Delayed Rewards," Ph.D. thesis, King's College, Cambridge, UK, May 1989.

[36] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M., "Playing Atari with Deep Reinforcement Learning," *arXiv*, 2013. Cite arxiv:1312.5602Comment: NIPS Deep Learning Workshop 2013.

[37] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al., "Human-level control through deep reinforcement learning," *Nature*, Vol. 518, No. 7540, 2015, p. 529.

[38] Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., Silver, D., and Kavukcuoglu, K., "Asynchronous Methods for Deep Reinforcement Learning," *arXiv:1602.01783 [cs]*, 2016. URL http://arxiv.org/abs/1602.01783, arXiv: 1602.01783.

[39] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D., "Continuous control with deep reinforcement learning," *CoRR*, Vol. abs/1509.02971, 2015. URL http://arxiv.org/abs/1509.02971, arXiv: 1509.02971.

[40] Zribi, A., Chtourou, M., and Djemel, M., "A New PID Neural Network Controller Design for Nonlinear Processes," *CoRR*, Vol. abs/1512.07529, 2015.

[41] Lin, S. H., "PID Newral Network Control for Complex Systems," *Computational Intelligence for Modelling, Control and Automation*, 1999.

[42] Yongquan, Y., Ying, H., and Bi, Z., "A PID neural network controller," *Proceedings of the International Joint Conference on Neural Networks, 2003.*, Vol. 3, 2003, pp. 1933–1938 vol.3. doi:10.1109/IJCNN.2003.1223703.

[43] Kang, J., Meng, W., Abraham, A., and Liu, H., "An adaptive PID neural network for complex nonlinear system control," *Neurocomputing*, Vol. 135, 2014, pp. 79–85. doi:10.1016/j.neucom.2013.03.065.

[44] Sutton, R. S., and Barto, A. G., *Reinforcement learning: An introduction*, Vol. 1, MIT press Cambridge, 1998.

[45] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., et al., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," , 2015. Software available from tensorflow.org.

[46] Zhang, M., and Qiang, M., "Study of PID Neural Network Control for Nonlinear System," *2006 8th international Conference on Signal Processing*, Vol. 3, 2006. doi:10.1109/ICOSP.2006.345787.