Safe Curriculum Learning for Linear Systems with Parametric Unknowns in Primary Flight Control

De Buysscher, D.D.C.; Pollack, T.S.C.; van Kampen, E.

DOI
10.2514/6.2022-0790

Publication date
2022

Document Version
Final published version

Published in
AIAA SCITECH 2022 Forum

AIAA SciTech Forum
January 3-7, 2022, San Diego, CA & Virtual
AIAA SCITECH 2022 Forum

10.2514/6.2022-0790

Safe Curriculum Learning for Linear Systems with Parametric Unknowns in Primary Flight Control

D.D.C De Buysscher*, T.S.C. Pollack† and E. van Kampen‡
Delft University of Technology, 2629HS Delft, The Netherlands

Safe Curriculum Learning aims at improving safety and efficiency aspects of Reinforcement Learning (RL). Curricular RL approaches divide a task into stages of increasing complexity in order to increase efficiency. This paper proposes a black box safe curriculum learning architecture applicable to systems with parametric unknowns. The agent domain solely requires knowledge of the state and action spaces' dimensions for a given task and system. By adding system identification capabilities to existing safe curriculum learning paradigms, the proposed architecture ensures safe learning of tracking tasks without requiring initial knowledge of the system dynamics. A model estimate is generated online to complement safety filters that rely on uncertain models for their safety guarantees. This research explicitly targets linearised systems with decoupled dynamics. The paradigm is initially verified on a mass-spring-damper system, after which it is applied to a quadrotor altitude and attitude tracking task. The RL agent is able to safely learn an optimal policy that can track an independent reference on each degree of freedom.

Nomenclature

$\mathcal{A}$	=	Set of all possible actions	u_t	=	Action vector at time t
c	=	Damper constant $[N/ms^{-1}]$	W()	=	Probabilistic weight function
f_t	=	Thrust force $[N]$	X_k	=	Augmented state at time step k
H()	=	Statistical entropy function	x_t^r	=	Reference state vector at time t
H	=	Kernel matrix	x_t	=	State vector at time t
k	=	Spring constant$[N/m]$	γ	=	Discount factor
K	=	Gain	$\hat{\theta}$	=	Model parameter estimates
m	=	Mass $[kg]$	λ	=	Curricular step index
P	=	Regression matrix	μ	=	Distribution mean
q()	=	q value function	v_π	=	Value function conditioned on π
Q	=	State weight matrix	ξ	=	Policy mapping matrix
R	=	Input weight matrix	π	=	Agent policy
$\mathcal{R}$	=	Reward function	σ	=	Distribution standard deviation
r or c	=	Reward or cost	$\tau_\bullet$	=	Torque around $\bullet$ axis $[Nm]$
$\mathcal{S}$	=	Set of all possible states	Ω_i	=	Rotor RPM of i^{th} rotor$[RPM]$
$\mathcal{T}$	=	Transition function	$\epsilon_\bullet$	=	Error between a state ($\bullet$) and its reference state ($\bullet^r$)

I. Introduction

The field of control has become increasingly important in numerous domains around the world ranging from transportation to appliances or even the agricultural sector. A common desire in these industries is to have a system that is capable of tracking a pre-defined trajectory. For example, in the airborne transport industry where personal UAVs follow a defined flight path. The complexity of these systems along with their ever increasing variety, demands for

*Graduate student, Control & Simulation Division, Faculty of Aerospace Engineering, Kluyverweg 1, 2629HS Delft, The Netherlands
†Ph.D. student, Control & Simulation Division, Faculty of Aerospace Engineering, Kluyverweg 1, 2629HS Delft, The Netherlands
‡Assistant professor, Control & Simulation Division, Faculty of Aerospace Engineering, Kluyverweg 1, 2629HS Delft, The Netherlands

controllers that are more flexible in their design process. Additionally, complex systems are often difficult to model and validate. Consequently, controllers that can be derived using data-driven methods are preferred in such cases. Advanced sensor-based controllers exist, such as incremental dynamic inversion controllers [1], to alleviate the burden of modelling complex systems. However, the aforementioned methods suffer from state reconstruction dependencies and synchronisation issues with the sensor data. Another approach to data driven controller design resides in machine learning paradigms such as Reinforcement Learning (RL). The fundamental principle used in RL is the representation of the world as an agent being confronted with a choice of action. The agent learns a control policy by interacting with the environment and gaining experience of the dynamics over time. Its basic principle is simple yet effective. However, with increasing task complexity (i.e., growing state and action spaces) increases, RL agents have a tendency to struggle learning a policy reliably [2]. Curriculum Learning (CurL) indroduced in [2], provides a structured approach to allow learning on more complex applications by dividing the initial task into sub-tasks [3, 4]. This facilitates the agent's learning process and increases the likelihood of successfully finding a control policy [5]. Given the examples cited previously, certainly in transport applications where stringent (safety) requirements apply, the safety aspect of the learning process and the correct operation of the controller is of crucial importance. Unlike RL methods which in their simplest forms generally lack consideration of the safety aspect [6], Safe Learning (SL) does provide a framework to this end [7].

The research outlined in this paper proposes a safe curriculum learning architecture that builds on the research presented in [8]. Here, the dependency on knowledge about an uncertain model for the safety algorithm is removed by complementing the paradigm in [8] with a system identification capability.

First a brief introduction to the fields of RL, Curriculum Learning, Safe Learning, and system identification is provided in sections II.A, II.B, II.C and II.D, respectively. This is followed by a detailed presentation of the approach chosen in this research outlined in Section III. Finally, the proposed paradigm is tested through two experiments. Initially, a Mass-Spring-Damper (MSD) system is used to verify the architecture for which the results are presented in Section IV.A. In Section IV.B, the results of a the safe curriculum architecture applied on a quadrotor are outlined. The paper is concluded with a discussion of the results of the experiments, as well as a conclusion and recommendations for further research.

II. Safe Curriculum Learning Framework

The core principles in safe curriculum learning are derived from three research fields: reinforcement learning, curriculum learning and safe learning. Inherently, the fundamentals originate from the more general machine learning field of study that is reinforcement learning. This learning process is then altered in curriculum and safe learning approaches.

A. Reinforcement Learning

Fundamentally, the concept of reinforcement learning aims at translating decision-making tasks into a logical space that is more accessible to computational implementations. It does so by using the high-level and more general formulation contained in a Markov Decision Process (MDP). All the possible actions available to the agent are contained in the action space, $\mathcal{A}$, and all possible states, x_k are defined as part of the state space, $\mathcal{S}$. Mathematically, the decision is characterised by the policy function, $\pi(x_t) = u_t$ [6]. It should be noted that all MDP's rest on the fundamental Markov property, stating informally that the future is independent of the past given the present[*]. In essence, this property describes that all information from the past is contained in the present observation, which is used by the policy to define the action to be taken by the agent. Once the agent has made a decision on the action to take, the action is executed and the agent is then transitioned to a new state. All the transitions between states dependent on the chosen action are defined in the transition function $\mathcal{T} : \mathcal{A} \times \mathcal{S} \to \mathcal{S}$. Finally, each transition is rated by a reward function, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathcal{R}$, assessing the optimality of the transition and as a consequence the state-action pair, (x, u). An MDP is uniquely defined by a tuple, $\langle \mathcal{A}, \mathcal{S}, \mathcal{T}, \mathcal{R} \rangle$ containing the aforementioned spaces and functions.

Canonical RL problems aim at optimising the path between an initial state, x_{init}, to the goal state, x_{goal}, by finding the optimal policy,π^*, for the MDP. The optimality of the policy is defined according to a reward function $\mathcal{R}$. To estimate this policy, RL agents perform a non-linear optimisation on the MDP at hand.

Reinforcement learning embodies two main optimisation strategies for MDP solvers, namely, Policy Iteration (PI) and its truncated version called Value Iteration (VI), which can be generalised to Generalized Policy Iteration (GPI) [6]. Computational complexity is reduced when using VI compared to PI, since the policy evaluation step does not require

[*]B. Langmead. Markov chains. University Lecture 22, John Hopkins Whiting School of Engineering, 2013

the value function to be the true one. The true value function is reached only in the limit. The policy evaluation step is outlined in Eq. (1), where $p(x', r|x, u)$ defines the probability of a subsequent state, x', occurring with the respective reward, r. Furthermore, $\pi(u|x)$ denotes the probability of an action to be taken at a given state, x. Finally, a discount factor, γ, is used to reduce the importance of the previous value function. The other iterative step in GPI, is policy improvement (Eq. (2)), which based on the current estimate of the value function will adapt the policy.

$$v_\pi^{k+1}(x) = \sum_u \pi(u|x) \sum_{x'} \sum_r p(x', r|x, u) \left[r + \gamma v_\pi^k(x')\right] \tag{1}$$

$$\pi^{k+1} = \begin{cases} \pi' & if \quad v_{\pi'}(x) \geq v_{\pi^k}(x) \\ \pi^k & if \quad v_{\pi'}(x) < v_{\pi^k}(x) \end{cases} \tag{2}$$

Solving the MDP can be done using multiple methods, where the method of choice is dependent on the knowledge of the MDP available to the operator. In [6], three methods are discussed, namely Dynamic Programming, Temporal Difference and Monte Carlo. Dynamic Programming requires full knowledge of the MDP, full state space, action space, transition function and reward function. The other two methods are used when a perfect model of the MDP is not available. Temporal difference and Monte Carlo focus on experience driven optimisation, through sampling of visited state and actions as well as the transitions that occurred with their respective rewards. The Monte Carlo approach is derived from the statistical theorem of large numbers where the more a state is visited the more accurate the MDP approximation of that state is. All make use of the value and policy iteration principles provided earlier in this section.

Applicability of these iterators is reflected in two common implementation of RL, namely SARSA and Q-learning [6]. These take the concept of a value function further. Instead of assessing the value of states, both implementations make use of a q-function, $q(x, u)$, which considers the value of a state-action pair. Detailed algorithmic formulations are provided in [6].

Besides the VI and PI strategies, other approaches do exist, such as Deep Deterministic Policy Gradient (DDPG) [9], Proximal Policy Optimisation (PPO) [10] and Trust Region Policy Optimisation (TRPO) [11]. The core principle behind each of these, is the use of the policy gradient to find the optimal policy of the topical MDP. Due to their more complex implementation requirements, they are not considered in this research.

B. Curriculum Learning

Mimicking the learning process of humans and animals on complex tasks, Curriculum Learning (CurL) provides a step-wise learning approach. Specifically, by decomposing the final task into sub-tasks that have relatively lower complexity. Each subsequent curricular step uses knowledge gained in the previous one resulting in the agent learning the ultimately complex task in a more gradual manner. Specifically, CurL has been proven successful in numerous fields, such as system control and complex games [5, 12]. Ng et al [5] have used curriculum learning to gradually train an RL agent to perform non-linear helicopter maneuvers. In 2016, Alpha Go defeated the – at the time – world champion in the game Go. It was trained using curriculum learning enhancements to facilitating the learning process of the complex game [4, 12, 13]. On a high-level and abstract basis, a curriculum obeys two statistical properties [2] which are detailed in a subsequent paragraph.

The design of a curriculum can have substantial impact on the overall stability and convergence of the RL agent's learning process. As defined by Bengio et al [2], curricula are subject to two rules, summarised in Eq. (3), in order to ensure a progressive learning strategy. One relates to a monotonically increasing statistical entropy, $H(Q_\lambda(x))$, throughout the curriculum, increasing the uncertainty of which states will be encountered by the agent. The other requires a monotonically increasing weight function, $W_\lambda(x)$, to add states to the state space made available to the RL agent. In the equation below, the subscript λ denotes the curricular step index and ϵ is a positive integer added to the former to denote a subsequent curricular step.

$$\begin{array}{lll} 1) & H(Q_\lambda(x)) < H(Q_{\lambda+\epsilon}(x)) & \forall\epsilon > 0, \forall x \in \mathcal{S} \\ 2) & W_\lambda(x) \leq W_{\lambda+\epsilon}(x) & \forall\epsilon > 0, \forall x \in \mathcal{S} \end{array} \tag{3}$$

Statistical entropy is a mathematical expression for designating the amount of uncertainty in a statistical process. It can be seen as a metric indicating how predictable a process is. In the case of the first condition given in Eq. (3), the entropy is tied to the probability of occurrence of a certain state or observation, $Q(x)$. In essence, given a state space, the function $Q(x)$ is the probability density function of that state space. If only a few states are likely to occur, the predictability is high. By adding entropy, the uncertainty of that distribution is increased and the occurrence predictability of a state or observation is reduced.

The second condition proposed in [2] and repeated in Eq. (3), relates to the weighting function allocated to an observation. By increasing $W_\lambda(x)$ throughout the curriculum, the state space available in the environment is enlarged. Furthermore, using knowledge about the topical RL problem, more *difficult* situations can be added in a later stage of the curriculum (with larger state and action spaces). This difficulty is problem dependent.

Inherently the two conditions are closely related. By increasing both the entropy and the weighting function throughout the curriculum, the diversity in the state space in increased.

Recent research [4] has found that curriculum learning approaches can be categorised in three main branches: reward shaping [14], variable task complexity and reversed problem space. All of which adapt one or multiple parts of the MDP's tuple in the design of the curriculum.

As the name suggests, reward shaping introduces variations in the reward function, $\mathcal{R}$, of the MDP. The agent is given higher rewards for good actions taken at a given state. Over the course of the curriculum, the bias introduced in the reward function is gradually removed, to converge towards the true reward function of the MDP. One example of the benefits of such approach is the non-linear helicopter maneuvering capabilities learned by the agent in Ng et al [5].

Gradually increasing task complexity is another approach to construct a curriculum which is more closely related to the way the academic systems are designed. The benefit of a curriculum becomes clear when using the example of an inexperienced agent being taught highly complex tasks (large state and action spaces). The learning process would be inefficient in two aspects: the amount of knowledge gained by the student and the time required for that knowledge to be assimilated. Instead, with a structured teaching approach the learning process's efficiency is increased [15].

Finally, there is the reversed problem space approach where the agent is initially placed close to the goal state. Throughout the curriculum, the agent is removed further away from the goal state. As such, it has to find a path to its comfort space, which it learned in previous curricular steps. Intrinsically these approaches increase the size of the state space and even the action space in each curricular step. As such, they allow the conditions presented in Eq. (3) to be satisfied.

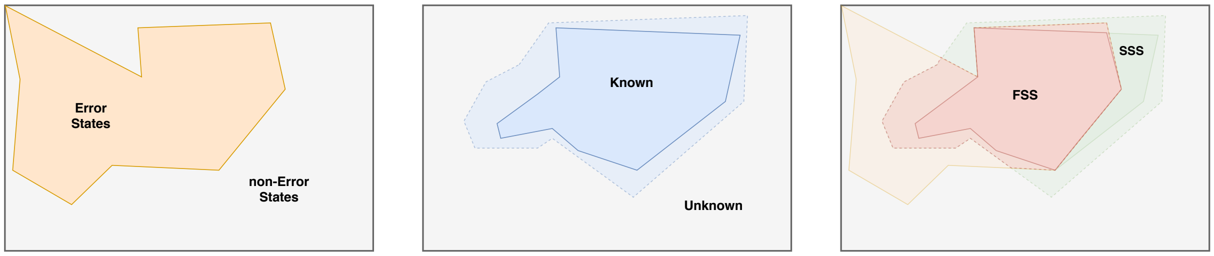
When moving from one curricular step to another, the purpose is to retain the knowledge and experience acquired thus far in the curriculum. Transferring knowledge between two sub-tasks is achieved using methods contained in Transfer Learning (TL) [16]. In RL, the gained experience and knowledge are generally contained in an agent's internal representation of the MDP. Based on the state and action space representations of the MDP by the agent, the knowledge transfer, or mapping, has to be adapted. When state and action spaces are represented equally in both curricular steps between which the knowledge transfer is to be performed, the mapping is straight forward. Indeed scaling is possible and no intricate mapping strategy is required. However, when either one of these spaces is different, mapping strategies become more strenuous. Two ideologies arise in this field [16]. The first focuses on learning the value function, v_π, which is transferred between source and target task [17]. The other proposes to transfer the policy learned in the previous curricular step to the target task, whilst re-iterating the value function during the new curricular step [18].

C. Safe learning

Reinforcement learning comes with the benefit of not being model-based. Indeed, methods such as temporal difference and Monte Carlo are able to find optimal policies without knowledge of part of the MDP. However, the safety of a system is not considered during the learning process. The safety aspect can be a crucial subject when applying reinforcement learning on real-life experiments, particularly on systems that are expensive to build or set up, such as aircraft, cars, boats or even robotic arms used in the production lines of factories [19, 20].

According to [7], safety in learning processes is attributed to either the optimisation criterion, or the exploration process. The optimisation criterion relates to the reward function of the MDP, discussed in previous section. Ensuring safe learning can indeed be addressed by adapting this criterion to be risk sensitive [7, 21]. As such, the reward function can be adapted to give lower rewards when a transition leads to an unsafe state (elaborated upon in the next paragraph). The exploration process can thus be interfered with by either providing external knowledge to the agent and guiding it away from dangerous state spaces [22], or by adding an element of risk into the exploration process. This provides the agent with a risk assessment tool enhancing its ability to define the optimal safe action.

During the exploration process of the state and action spaces, certain state-action pairs can pose danger to systems. In safe learning research, safety is assessed by associating a given state to a state space with certain properties. In Fig. 1, an overview is given of three approaches used to divide the state space into groups based on certain characteristics.



(a) ***Error*** **and** ***Non-Error*** **states within the set of states $\mathcal{S}$**

(b) ***Known*** **and** ***Unknown*** **states within the set of states $\mathcal{S}$. Dashed line represents the exploration of $\mathcal{S}$ over time.**

(c) ***Safe State Space (SSS)*** **and** ***Fatal State Space (FSS)*** **within the set of states $\mathcal{S}$. Dashed line represents the exploration of $\mathcal{S}$ by the agent over time. Note that this figure is an overlap of figures 1a and 1b**

Fig. 1 State space segregation based on different concepts: (left) error-states, (middle) known states, (right) safe states. The grey rectangle represents the complete state space of the task at hand, $\mathcal{S}$.

In a first instance, the state space system can be divided in error and non-error spaces which define the consequence of being in their respective states (see Fig. 1a) [23]. Error states are considered as harmful to the system as they either lead to system termination or to serious damage. Non-error states are considered innocuous to the system. In a second instance, the state space can be segregated based on the experience gained by the agent. Shown in Fig. 1b, states are labeled as known or unknown, depending on the agent having visited the state previously [24]. Finally, combining these concepts, the definition of a Safe State Space (SSS) and a Fatal State Space (FSS) can be given [25–27]. In a SSS, only states that are labeled non-error and known can be included. The FSS includes error states, regardless of whether the agent has visited the state previously. This is shown in Fig. 1c.

In the scope of exploratory safety, the Safety Handling Exploration with Risk Algorithm (SHERPA) and optiSHERPA introduced in [25–27] provide a risk-sensitive and model-predictive exploration method. SHERPA makes use of the policy reuse concept outlined in [28], in which the probability of using a safe policy is related to the risk sensed at a given state. In SHERPA's terminology, this safe policy is defined as a backup policy, which relies on the ergodicity condition. Discussed in [29], an exploration method based on ergodicity goes into more detail on this regard. When an agent follows its policy, that policy is deemed safe if and only if there is the possibility to safely return to a previously visited state. The way to return to a state, $x \in SSS$, is SHERPA's backup policy. This backup policy has to satisfy a complementary condition that the trajectory to this state is within the SSS at all times. A considerable drawback of the ergodicity condition and algorithms relying on its application is its dependence on knowledge of system dynamics, albeit in the form of of a bounding model defined as $[x_{k+1}] = \Delta([x_k], u_k)$ (where [x] defines the interval notation of a state) [27].

D. Safe Learning for Systems with Parametric Unknowns

Safe learning algorithms rely mainly on external knowledge, in the form of guidance during exploration or sensible adaptions of optimisation criteria [7]. Lacking the availability of system dynamics, albeit certain or uncertain, has a more significant impact on safety methods based in the exploration process of the state and action spaces by the agent. More specifically, SHERPA and optiSHERPA that are methods founded on the ergodicity condition [25–27, 29]. By consequence, these paradigms inherently rely on system and input dynamics representations to project trajectories internally. In order to solve the problem in such methods when working with a system where only the state and action orders are known, system identification techniques can be used.

Linear systems mostly make use of a state space model representation for their system and input dynamics. Moreover, their non-linear counterparts can be approximated by time-variant linear state space systems, with the exception of highly non-linear systems due to inaccuracies arising with such models. Consequently, for highly non-linear systems multivariate splines or Artificial Neural Networks (ANN) are preferred for a more accurate model representation. Multivariate splines are model representations that are constructed by a set of piecewise continuous functions, locally approximating the dynamics. Data points are collected to optimise simplex coordinates, which in turn provide a baseline for the piecewise functions. On the other hand ANN's provide a non-linear model approximation by virtue of their

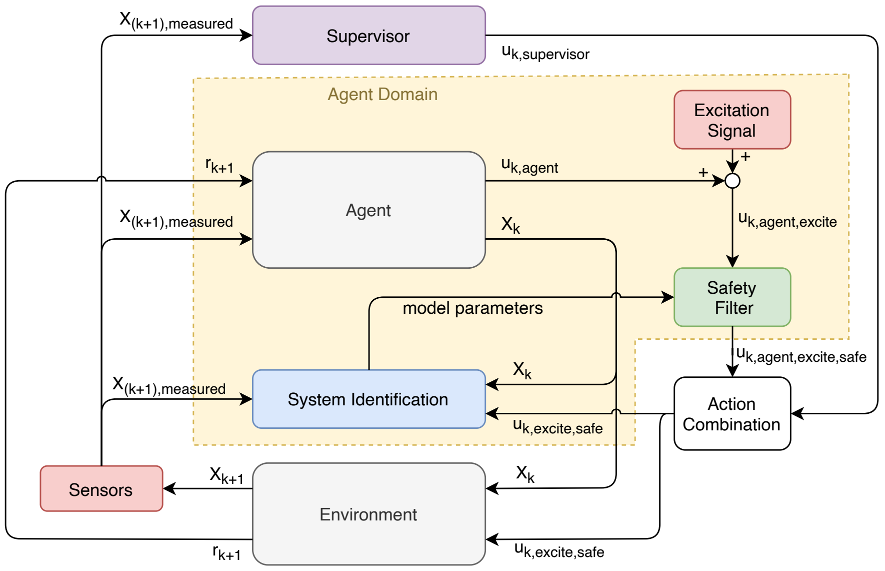


Fig. 2 Schematic overview of the online learning architecture illustrated for a single curricular step

activation functions defining their nodes [30].

E. Safe Curriculum Learning for Systems with Parametric Unknowns

The basis of the Safe Curriculum Learning paradigm was first introduced in [8], where it was used to regulate a 3-mass-spring-damper system as well as a quadcopter on its six degrees of freedom. Combining the knowledge found in research discussed in previous sections, the paradigm aims at finding a control policy of complex systems whilst considering safety of the learning process. The research provided successful results for a regulation task by using the SHERPA algorithm as a safety filter.

Furthering the paradigm by providing independence of system dynamics knowledge, an overview of the learning architecture is given in Fig. 2. Here, a system identification module is added to the process to provide the safety filter with an estimate of the dynamics during the online learning process.

Each curricular step uses the logic demonstrated by this diagram to find an optimal safe policy for the topical task at that stage of the curriculum and its respective MDP. Mapping the agent's representation of the value function and policy is then used to transfer knowledge between curricular steps. A curriculum can be constructed by repeating the schematic provided in Fig. 2 in a sequential or parallel manner, depending on the curricular requirements of the task at hand.

III. Methodology

The research in this article proposes an online safe curriculum learning paradigm with the purpose of controlling systems with parametric unknowns to track a reference signal. The black box approach assumes knowledge about the order of the state and action spaces to be available. However, system dynamics and input dynamics are not provided in an exogenous manner throughout the learning process. The decisions made during the research regarding the specific methods used to construct the multiple building blocks of the safe curriculum learning paradigm are presented in this section. The section is started with the task to be performed by the agent along the RL method of choice, after which the curricular construct is elaborated, followed by the safety filter.

A. Learning Framework

Provided that a reference tracking controller is sought, as outlined in the introduction, a tracking task was devised to obtain a high-performance tracking policy for a system. As mentioned in Section II.A, reinforcement learning processes are guided by their underlying MDP's reward function, $\mathcal{R}$, for optimisation of the policy and value function. In this research the Linear Quadratic (LQ) cost function (shown in Eq. (4)) is selected as it can be used in optimal control strategies in concordance with RL methods such as Q-Learning [31, 32]. The Q and R matrices found in the LQ cost function are diagonal weighting matrices. Their respective diagonals contain a weight expressing the importance of the state and input vector elements' contribution to the cost [32].

$$\mathcal{R} : c = x^T Q x + u^T R u \tag{4}$$

By limiting the scope of the research to discrete-time linear systems, the Linear Quadratic Tracking (LQT) paradigm proposed by [31] was found to be promising. That research uses Q-learning with PI to learn a tracking task. Fundamentally based on Linear Quadratic (LQ) optimal control theory, the authors provide analytical proof of a RL agent's ability to learn a tracking stabilising policy. Instead of finding a controller that regulates the states to an equilibrium, the LQT task aims to regulate the error between the system's state and the reference signal. The scope being limited to linear systems, it was chosen to use the linear discrete-time state space (A, B, C, D) representation to model the system dynamics in the environment and consequently as the transition function, $\mathcal{T}$. By means of the C matrix of this representation and the assumption that feedforward dynamics are omitted, the error is defined as shown in Eq. (5). This leads to the use of an augmented state, X_k containing both the system's state and the reference's state, x_k^r (generated by the command module). Due to the introduction of an augmented state, X_k, the Q matrix of the LQ cost function in Eq. (4) needs to be altered to accommodate for the new dimension of this augmented vector. In [31], a substitute matrix, $Q_1 = C_1^T Q C_1$, is defined to ensure conforming dimensions with the augmented state.

$$e_k = y_k - x_k^r = C x_k - x_k^r = \begin{bmatrix} C & -I \end{bmatrix} \begin{bmatrix} x_k \\ x_k^r \end{bmatrix} = C_1 X_k \tag{5}$$

The Q-Learning approach that was chosen is not of tabular form. Although the state and action spaces are discrete in time, they are not discretised to fit within a tabular formulation where actions are related to ranges of states. Instead, the possibilities of state-action pairs is infinite. In order to accommodate such spaces, a continuous Q-function is required. For this purpose, Kiumarsi et al. [31] propose to use a kernel matrix, H, to determine the Q-value as outlined in Eq. (6). The focus of the RL process is to find the optimal kernel matrix for the topical task. In the equations below, the vector, $Z = \begin{bmatrix} X_k & u_k \end{bmatrix}^T$, contains both the augmented state as well as the action at a given time step. Additionally, the constant γ represents the discount factor.

$$q_k = \frac{1}{2} Z_k^T H Z_k \tag{6}$$

In more general terms, the Q-function satisfying the the Bellman equation [31, 33] for an LQT task is formulated in Eq. (7). This equation holds for discrete-time linear systems. Obtaining the values of the symmetric kernel matrix, H, is done by using sample based VI. By sampling the states and inputs along the system trajectory, an estimate of the kernel matrix can be computed using a simple Ordinary Least Squares (OLS) regression. The estimator is discussed in more detail in Section III.C. Mathematically, the estimate of the subsequent kernel matrix, H^{j+1}, is obtained by solving Eq. (8), in which Z_k is sampled over time. This covers the policy evaluation step of the VI strategy.

$$Z_k^T H Z_k = X_k^T Q_1 X_k + u_k^T R u_k + \gamma Z_{k+1}^T H Z_{k+1} \tag{7}$$

$$(Z_k \otimes Z_k)^T \; vec(H^{j+1}) = X_k^T Q_1 X_k + u_k^T R u_k + \gamma Z_{k+1}^T H^j Z_{k+1} \tag{8}$$

In order to complete the iterative loop of VI, the policy improvement step has to be performed. Here, the policy takes the form of a control gain, K_1, in control theoretical terms. Based on the findings in [31], the logic within the policy is given by Eq. (9), where H_{uu} and H_{uX} are sub-matrices of the kernel matrix H (see Eq. (10)).

$$u_k = -\left(H_{uu}^{-1} H_{uX}\right) X_k = -K_1 X_k \tag{9}$$

Successful learning convergence is achieved by adding a Persistence of Excitation (PE) signal to the input vector, in order to avoid non-invertible precision matrices in the OLS estimator. Random noise has been selected as the excitation signal of choice. In accordance with Narendra and Annaswamy [34], the PE signal shall not integrate to zero over the course of the sampling period if the PE condition is to be satisfied. White noise is not likely to fail this condition. However, such signal comes with the drawback that it is fundamentally not band limited and the signal power is spread equally over all frequencies.

B. Curriculum Setup

The learning task presented in the previous section, although proven to have convergent learning behaviour, suffers from the curse of dimensionality. Increasing the amount of states causes an exponential increase in the dimensions of the kernel matrix. For complex systems that adhere to state and action spaces with numerous dimensions, such as aircraft, this can result in relatively large kernel matrices. By designing a curriculum, the strain can be relieved from the learning process and positively impact the learning convergence, performance and safety on the aforementioned state and action spaces.

The curriculum learning strategy chosen for this research is a curriculum based on gradually increasing the task complexity. Two staging approaches were considered for this research, namely intra-task and inter-task [8]. For intra-task stages, the agent's representation of the state and action spaces was set to be the same between two consecutive curricular steps. On the other hand, inter-task staging assumes a different representation between the source and target tasks be it in terms of dimensions or in terms of internal dynamics representation. Besides increasing the task complexity, reward shaping was implemented on more complex systems as an additional guide for the agent. By changing the values on the diagonals of Q and R from Eq. (4), the contribution of each state and action element towards the total cost can be adjusted. Its use was primarily prominent when a state and its derivative were present in the state vector. For example, when a position state element is tasked with tracking a reference, the respective rate component of that state is given a lower value in Q to focus the agent's attention on the position tracking before the velocity tracking. In the consecutive curricular step, the tracking of the rate is removed entirely.

A second point of attention when designing a curriculum is the transfer of knowledge between curricular steps. The knowledge gained during a particular curricular step is stored in the kernel matrix H. Since this matrix is used to define the Q-value space, it contains the necessary information to dictate the agent's value function and policy. Here, it was opted for a semantic mapping strategy of the kernel matrix. In essence, the environment and agent of the subsequent curricular step dictate the new dimensions of the kernel matrix for that step as they set the dimensions of the state and action spaces. The basis for the new kernel matrix is an identity matrix onto which the kernel values of the source task are mapped based on corresponding state, reference and action elements. For the sake of clarity, Eq. (10) outlines an overview of the different blocks of the kernel matrix relating to the state, reference and action elements denoted by the subscripts x, x^r and u respectively.

$$H = \begin{bmatrix} H_{xx} & H_{xx^r} & H_{xu} \\ H_{x^r x} & H_{x^r x^r} & H_{x^r u} \\ H_{ux} & H_{ux^r} & H_{uu} \end{bmatrix} = \begin{bmatrix} H_{XX} & H_{Xu} \\ H_{uX} & H_{uu} \end{bmatrix} \tag{10}$$

Mapping semantically requires an operator with a broad understanding of the system and input dynamics, which comes as a disadvantage of this specific strategy. In this work, the operator is responsible for defining the mapping strategy by allocating the state and and action positions correctly throughout the mapped kernel matrix. Additionally, it means that knowledge of the specific internal dynamics (i.e., coefficients in the kernel matrix) is not a requirement for the mapping process. An accurate mapping of the states and actions provides a more reliable starting point for the RL agent in the subsequent curricular step.

C. Safety Filter for Systems with Parametric Unknowns

Completing the proposed paradigm in this research is the safety aspect. It is true that safety can be a consequence of using curriculum learning. However, no guarantees can be asserted by these means alone. Consequently, an additional safety module was used to ensure a safe learning process, as is shown in Fig. 2. The safety mechanism of choice is a safety filter fundamentally relying on ergodicity named SHERPA [25–27].

Essential to the SHERPA safety filter are the concepts of the SSS and FSS discussed in Section II.C. In essence, the SSS is an evolving part of the state space that is enlarged through experience. States that have been visited by the agent and are considered safe are added to the space. To push the safety aspect further, a sensor reach is added to the state to act as the risk function required for the SHERPA algorithm as it is defined by Mannucci et al [27]. For example, a radar sensor can be equipped to a quadcopter, allowing it to sense for obstacles up to a certain distance thus giving it a better idea of potential threats in its surroundings. The binary risk function sets risk at 0% when the sum of the state and the sensor reach are outside of the FSS (and conversely sets it to 100% otherwise). Besides the expansion of the SSS over time, the safety of the system is retained by finding backup policies. These policies aim at verifying that the agent is not in Lead to Fatal States (LFS), which are states excluded from the FSS but lead to it with certainty [27]. Additionally, backup policies are validated by satisfying the ergodicity condition [27, 29].

The search for such policies lies at the core of SHERPA's paradigm. Using a form of model representation, the algorithm projects a number of trajectories for a pre-defined number of time steps based on a set of policies. The trajectory generation proposed by Mannucci et al [27] relies on a series of random actions. Instead, this research considers the SHERPA safety filter to be part of the agent domain (see Fig. 2). Consequently, SHERPA has access to the agent's current policy, from which it generates a set of policies used to define the series of inputs taken for the projected trajectories. More specifically, the policy generation is based on the ϵ-greedy [6] approach used to promote exploration in RL. In most cases the agent's current policy parameters are altered by a certain random factor. In the remaining cases, a completely random policy is generated. The projection of these trajectories is performed using an uncertain model of the system and input dynamics that is identified online.

The proposed contribution of this research finds itself in the addition of a system identification module to estimate a model of the system and input dynamics. As model representation, the linear state space formulation was chosen by virtue of its efficiency and fitness for the scope of this research. The model is estimated using sample based Ordinary Least Squares (OLS), with an estimate update frequency that is different and higher than the VI iteration process. Samples of the state and input at time step k as well as the subsequent state are collected to construct the regression matrix, P, and regression vector, b, which are then used to find the model estimate as shown in Eq. (11).

$$\hat{\theta} = \left(P^T P\right)^{-1} P^T b \tag{11}$$

The model uncertainty of the estimate is derived from the residuals between the estimated model and the sampled data. Thereby resulting in the absolute parametric uncertainty of the system dynamics estimate, $\hat{A}$, the input dynamics estimate, $\hat{B}$, and the reference dynamics estimate, $\hat{F}$. These uncertainties are crucial for the uncertainty region's propagation in SHERPA's backup policy projection process.

IV. Simulation Results

The safe curriculum learning architecture for systems with parametric unknowns proposed by this research was first verified on a cascaded Mass-Spring-Damper (MSD) system supported at one end. This is followed by an example of the application on primary flight control on a linearised quadrotor model, which has been limited to attitude and altitude control.[†]

A. Model Verification on Mass-Spring-Damper System

The architecture presented in Fig. 2, was verified on a N-MSD system, where N denotes the number of masses present in the system. This system is used because of its modularity and scalability. Additionally, making a N-MSD system unstable can be achieved by making either the spring constant, k, or the damping constant, c, negative. The curricular construction is centred around inter-task staging steps where the agent's representation of the system dynamics varies throughout the curriculum. Initially, a single mass attached to fixed bound (1-MSD) is used as an environment for the agent, with stable open-loop characteristics. The agent's task is to have the mass position track a sinusoidal reference signal. Throughout the curriculum, more masses are added between the wall and the previously existing system. Moreover, the system is gradually made unstable to conform with the statistical conditions of entropy and diversity that are needed for a curriculum. The ultimate goal is to have the agent track a sinusoidal reference signal on a

[†]Full code of the implementation used in theses simulations can be found at `https://github.com/DBdiego/SafeCurriculumLearning.git`

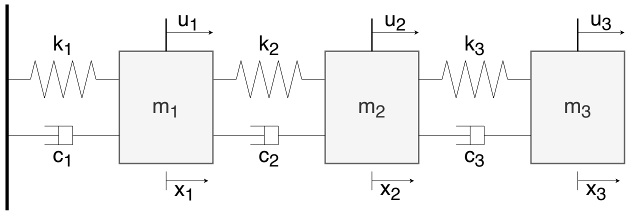


Fig. 3 Figure showing a cascaded 3-Mass-Spring-Damper system

Table 1 Characteristic parameters of the unstable 3-MSD system shown in Fig. 3

Mass	m $[kg]$	k $[N/m]$	c $[N/(ms^{-1})]$
1	0.3	-1	6
2	0.8	3	-1
3	0.4	4	3

Table 2 Characteristic parameters of the persistence of excitation signal and measurement noise

PE signal		Measurement Noise	
Parameter	Value	Parameter	Value
μ_{PE}	0.0	μ_{MN}	0.0
σ_{PE}	1.5	σ_{MN}	$2.0 \cdot 10^{-3}$
f-band	$[-\infty, \infty]$		

$3 - MSD$ system with the last mass's position, x_3, whilst staying within provided safety bounds. Additionally, the other masses in the system are regulated. Finally, the actions taken by the agent are bounded by saturation limits set to 5N in either application direction.

The focus of this verification is not only to prove that the agent can safely learn but also to show that the learning efficiency is improved when complementing the agent with a safety filter and pacing it through a curriculum. Therefore, the results of the proposed architecture are compared to flat learning of the aforementioned tracking task on the full 3-MSD system. With the characteristic parameters of the 3-MSD system and excitation signal presented in Table 1 and Table 2, respectively, the agent converged to a stabilising policy without leading the system in the FSS in a curriculum learning framework. Consequently, completing the input's excitation signal, a measurement noise, with characteristic hyperparameters outlined in Table 2, was added to the state time series to provoke unstable kernel and policy updates by the agent. Without the addition of measurement noise, the learning process stays safe even without a safety filter. By adding measurement noise, the benefits of SHERPA can be showcased more clearly. Consequently, to make the results comparable, both the flat learning benchmark and the safe curriculum implementation were complemented by measurement noise.

When neither the curriculum and the safety filter are activated, the agent's learning behaviour is volatile and divergent. This is reflected in Fig. 4. It takes two kernel updates ($H^j \leftarrow H^{j+1}$) for the agent's policy to dictate actions that are continuously saturated. Furthermore, once the position of the three are well beyond the SSS (green area in Fig. 4), the tracking task becomes more arduous. Conversely, when using the safe curriculum architecture proposed in this research, the resulting learning process is stable and convergent. The convergence rate, however, varies with the respective amplitudes of the PE and measurement noise signals. In Fig. 5, the position state and action force propagation of all masses is given, showing an adequate tracking of the reference by x_3. The total learning time for

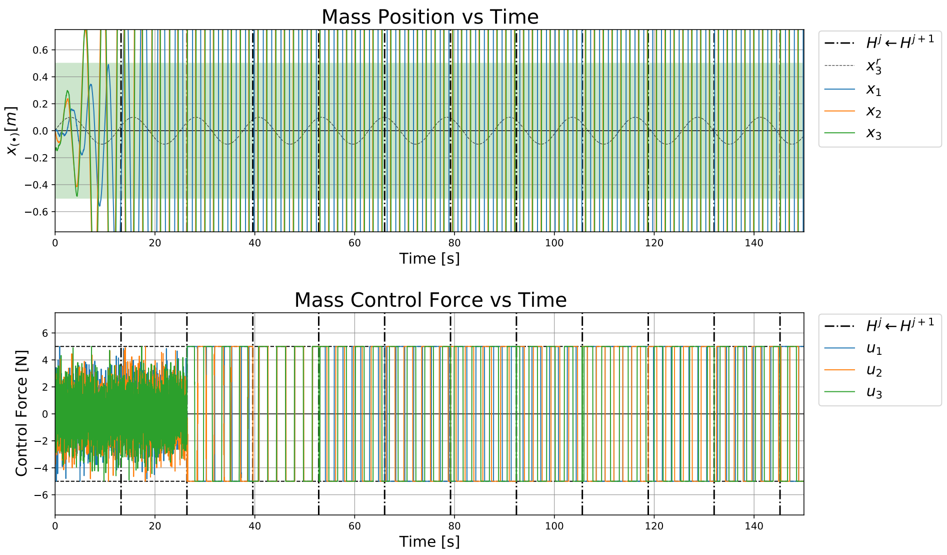


Fig. 4 Evolution of mass positions and control forces over time; Online VI; Flat Learning; No SHERPA; Gaussian sensor noise, $\sigma = 0.0012$; $x_{init} \in [-0.2, 0.2]$; $\dot{x}_{init} \in [-0.25, 0.25]$; Green is the SSS for x_3

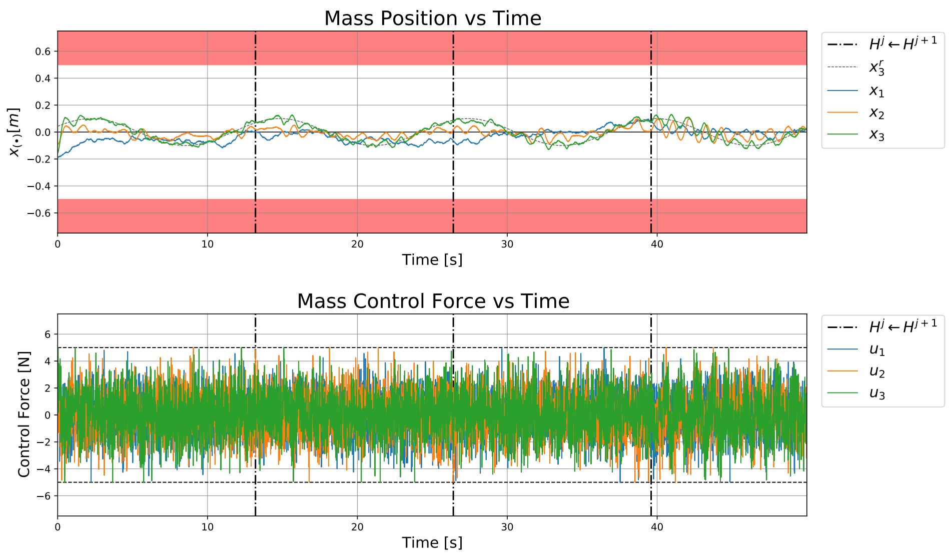


Fig. 5 Evolution of mass positions and control forces over time; Online VI; Safe Curriculum Learning (step 3); SHERPA activated; Gaussian sensor noise, $\sigma = 0.0012$; $x_{init} \in [-0.2, 0.2]$; $\dot{x}_{init} \in [-0.25, 0.25]$; Red is the FSS for x_3

the safe curriculum learning implementation was set to equal the flat learning benchmark's total learning time. Each curricular step provides 50 seconds (arbitrarily defined) to the agent to learn a stabilising tracking policy.

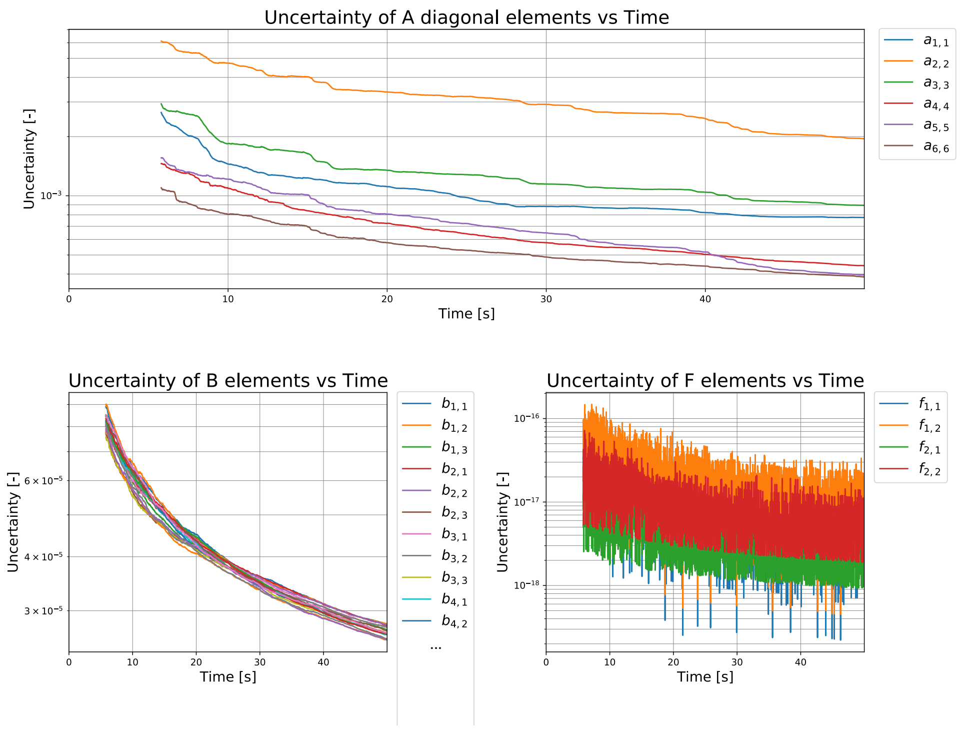


Fig. 6 Evolution of the absolute parametric uncertainty of all matrices forming the uncertain model estimate used in SHERPA, $\hat{A}$, $\hat{B}$ and $\hat{F}$; Online VI; Safe Curriculum Learning (step 3); SHERPA activated; Gaussian sensor noise, $\sigma = 0.0012$.

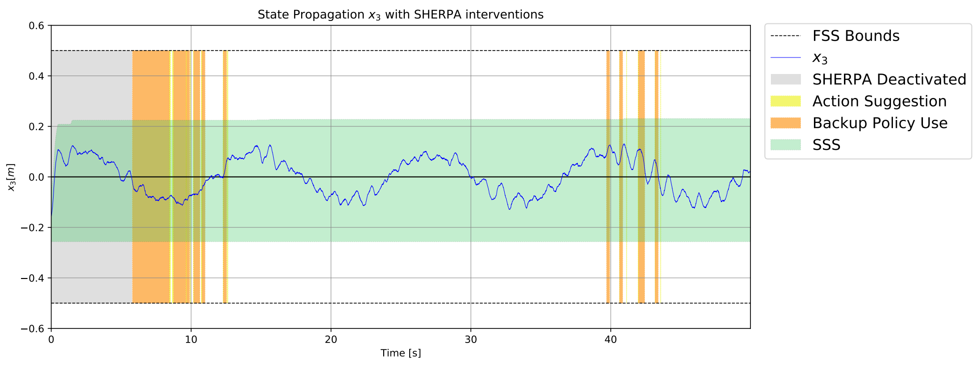


Fig. 7 Evolution of the SSS over time along with SHERPA interventions

Furthermore, the system identification module (blue box in Fig. 2) gradually lowers the parametric absolute uncertainty during the simulation. By assuming that the system to be modelled is linear, the estimated model parameters are time-invariant. As such, the sliding window principle can be used for system identification. The initial estimate is the only one that is delayed as it requires sufficient samples to be collected. During this sampling, SHERPA does not need to be activated since no uncertain model estimate is available for the backup policy projection. The purpose is to reduce the system identification module's initial sampling time as much as possible whilst keeping the estimate's uncertainty relatively low. The evolution of these absolute parametric uncertainties are shown in Fig. 6. It can be observed that the estimate becomes available at $t = 5.8$ seconds. Finally, SHERPA's interventions are shown in Fig. 7. Here, the grey area provides a better understanding of this architecture's major drawback, namely, that the system is prone to go into the FSS during this time interval.

This concludes the verification of the proposed architecture on a simple system such as a Mass-Spring-Damper system. The results show that an agent can indeed safely learn a complex task by increasing the task complexity throughout a curriculum with inter-task staging. Consequently, the paradigm can be applied on more complex systems such as quadrotors.

B. Quadrotor Primary Flight Control

Having verified the proposed paradigm, it is possible to apply it to more complex systems such as a quadrotor for the proof-of-concept. More specifically, the goal is to find a stabilising policy that can track references in all four Degrees Of Freedom (DOF) available in this model. These are constituted of the pitch, roll and yaw attitudes, and the altitude.

A linearised continuous-time quadrotor model was derived based on the state and action vector formulations of Eq. (12) and is shown in Eq. (13). Due to the paradigm's limitation to discrete-time dynamics, the state space system shown in Eq. (13) was discretised using the zero-order hold method. By assuming small angles for ϕ, θ and ψ, the rates p, q and r can be assumed to be the same as $\dot{\phi}$, $\dot{\theta}$ and $\dot{\psi}$. Furthermore, the equilibrium point used for linearisation is hovering flight. The quadrotor characteristics can be found in Table 4.

$$\begin{aligned} x &= \begin{bmatrix} \phi & \theta & \psi & p & q & r & w & z \end{bmatrix}^T \\ u &= \begin{bmatrix} f_t & \tau_x & \tau_y & \tau_z \end{bmatrix}^T \end{aligned} \tag{12}$$

$$A = \begin{bmatrix} 0&0&0&1&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&0 \end{bmatrix} \quad B = \begin{bmatrix} 0&0&0&0 \\ 0&0&0&0 \\ 0&0&0&0 \\ 0&\frac{1}{I_{xx}}&0&0 \\ 0&0&\frac{1}{I_{yy}}&0 \\ 0&0&0&\frac{1}{I_{zz}} \\ -\frac{1}{m}&0&0&0 \\ 0&0&0&0 \end{bmatrix} \tag{13}$$

Table 3 Curriculum setup. Red are the states and inputs controlled by an external *supervisor* LQR-controller. Green are the states and inputs controlled by the agent. Underlined states track a reference.

Curricular Steps		1	2	3	4	5
Inputs	Thrust force	🟥 f_t	🟥 f_t	🟥 f_t	🟩 f_t	🟩 Ω_1
	Torque around x	🟥 τ_x	🟩 τ_x	🟩 τ_x	🟩 τ_x	🟩 Ω_2
	Torque around y	🟥 τ_y	🟥 τ_y	🟩 τ_y	🟩 τ_y	🟩 Ω_3
	Torque around z	🟩 τ_z	🟩 τ_z	🟩 τ_z	🟩 τ_z	🟩 Ω_4
States	roll	🟥 ϕ	🟩 $\underline{\phi}$	🟩 $\underline{\phi}$	🟩 $\underline{\phi}$	🟩 $\underline{\phi}$
	pitch	🟥 θ	🟥 θ	🟩 $\underline{\theta}$	🟩 $\underline{\theta}$	🟩 $\underline{\theta}$
	yaw	🟩 $\underline{\psi}$	🟩 $\underline{\psi}$	🟩 $\underline{\psi}$	🟩 $\underline{\psi}$	🟩 $\underline{\psi}$
	roll rate	🟥 p	🟩 $\underline{p}$	🟩 p	🟩 p	🟩 p
	pitch rate	🟥 q	🟥 q	🟩 $\underline{q}$	🟩 q	🟩 q
	yaw rate	🟩 $\underline{r}$	🟩 r	🟩 r	🟩 r	🟩 r
	vertical velocity	🟥 w	🟥 w	🟥 w	🟩 $\underline{w}$	🟩 w
	z-position	🟥 z	🟥 z	🟥 z	🟩 $\underline{z}$	🟩 $\underline{z}$

Table 4 Characteristic parameters for the quadrotor system

Parameter	Value	Unit
I_{xx}	0.045667	$[m^4]$
I_{yy}	0.045667	$[m^4]$
I_{zz}	0.090667	$[m^4]$
h_{body}	0.1	$[m]$
w_{body}	0.1	$[m]$
l_{body}	0.1	$[m]$
l_{arm}	0.3	$[m]$
m_{body}	0.2	$[kg]$
m_{arm}	0.1	$[kg]$
m_{motor}	0.2	$[kg]$
m_{quad}	1.60	$[kg]$
α_{arms}	45	$[deg]$
b	2.04366e-3	$[N/RPM]$
d	2.78893e-4	$[Nm/RPM]$

In this implementation, a combination of intra-task and inter-task staging strategies was used, where the agent keeps the system dynamics representation constant throughout the curriculum. However, the representation of the input and reference dynamics varies. This was chosen based on the task complexity being relatively high in the curricular steps presented in Table 3. The agent gains control of an additional input at every step. The other inputs are controlled by an LQR supervisor supervisor. The gains of the supervisor were deduced using an LQR theory with the Q and R matrices given in Table 7. Along with a new input dimension, comes one or multiple new degrees of freedom to be controlled by the agent resulting in an additional reference signal for each step. The curricular setup has been summarised in Table 3 to provide a better overview. The order in which the inputs are switched from the supervisor to the agent has no specific importance other than the hyperparameters shown in Table 5 should be altered. Indeed, the attitude angles that are not learned have to remain small in order to satisfy the small angle assumption made when linearising the model.

In essence, each input has respective states that it controls. Not only does this provide better tractability during the learning phases, but it also allows for an easier separation between the LQR supervisor and the agent's control actions. Indeed, this approach avoids the contribution of the supervisor actions and controlled states to the cost function (Eq. (4)) during the learning process. Consequently, the agent is able to better understand the influence of its own actions.

The fifth curricular step focuses on mapping the tracking policy found in the first four steps, into a policy that uses the rotor RPMs as input vector instead of torques and forces along the respective degrees of freedom. In this last step, no learning occurs; only state propagation checks to ensure the learned policy can track along all degrees of freedom with minimal error ($RMSE < 0.05$).

The policy learned by the agent in the first four curricular steps applies to any quadcopter configuration as long as there is a mapping, ξ, to a rotor RPM input vector, Ω. Note that the experiment conducted in this research has assumed a linear relationship between a rotor RPM and the thrust force, $f_{t,i} = b \cdot \Omega_i$, and torque, $\tau_{z,i} = d \cdot \Omega_i$, it generates. This assumption facilitates the mapping process and proof of concept. Notably, however, effects such as blade flapping and aerodynamic disturbances are not considered in this experiment. With this assumption, the mapping matrix, ξ, for an X-configuration quadrotor can be defined as outlined in Eq. (15). This map is derived from a basic understanding of forces on a quadrotor shown in the form of three free body diagrams in Fig. 8.

$$\pi_\Omega = \xi^{-1} \cdot \pi_{f,\tau} \tag{14}$$

$$\xi = \begin{bmatrix} b & b & b & b \\ -bl_{1,y} & -bl_{2,y} & bl_{3,y} & bl_{4,y} \\ bl_{1,x} & -bl_{2,x} & -bl_{3,x} & bl_{4,x} \\ d & -d & d & -d \end{bmatrix} \tag{15}$$

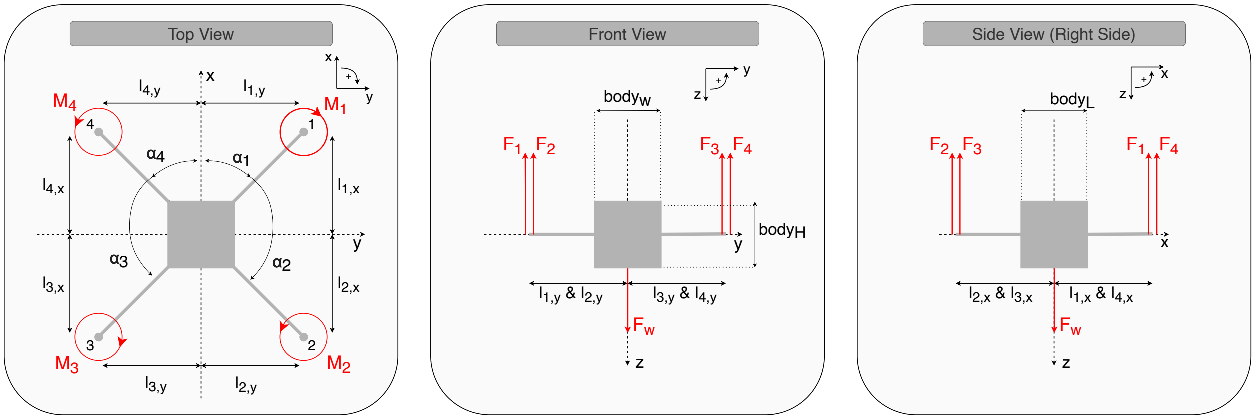


Fig. 8 Views of the quadrotor forces and other important dimension.

For a quadrotor, it was found empirically that the amplitude and the frequency of the PE signal both have an essential impact on the learning behaviour; which was not apparent for the N-MSD system used to verify the architecture. The agent was more specifically found to experience more difficulties learning in a convergent manner when the PE signal has a low frequency bandwidth. Therefore, a band-limited excitation signal was used in this experiment. An arbitrary frequency range was defined to be between 50Hz and 100Hz (without consideration to actuator dynamics). By increasing the frequency, it was found that the lower derivatives of each DOF (i.e. attitude angles) are more centred around their respective regulated values, specifically zero in this experiment.

Additionally, the frequency and amplitude of the reference signal play an essential role in the convergence and stability of the learning process. They fundamentally define the task that the agent has to learn. Moreover, when their frequency range is similar to the eigenfrequencies of the system, an additional complexity is added to the learning process as the system starts to resonate. The hyperparameters for both signals throughout the curriculum are summarised in Table 5. In this table, the amplitude of the attitudes' reference signals diminish throughout the curriculum. Solely the dimension on which the agent's learning is focused receives a reference signal with a larger amplitude[‡]. Due to the linearisation of the system, small angle approximation and the assumption that $\{p, q, r\} = \{\dot{\phi}, \dot{\theta}, \dot{\psi}\}$, the angles that are not learned have to remain small in order for the modelled dynamics to be representative of a quadrotor.

The improvements provided by both a curriculum on the learning stability and a safety filter on the learning process's safety are shown by comparing their simulation results to a benchmark flat learning case. The cumulative learning time within the simulation is limited to 170 seconds. Outlined in Fig. 9 is the evolution of all the attitude state angles and the altitude position for flat learning. It is apparent that the agent is not able to find a stabilising tracking policy in the given time frame. Furthermore, the safety-critical roll and pitch attitudes are constantly in the FSS. Consequently, this task may benefit from a curriculum to more gradually increase the task complexity and use a safety filter to ensure a safe learning process.

Following the curricular strategy presented in Table 3, the first sub-task was learning to track a reference on the yaw angle, ψ, whilst all the other DOF are under the authority of a regulating supervisor. To guide the agent in the learning process, an additional reference is provided for the yaw rate, r[§]. The learning behaviour is shown in Fig. 10a. In a second instance, control of the roll attitude comes under the agent's authority, and the agent learns a policy that tracks both roll and yaw angle. This is presented in Fig. 10b. This is then followed by the addition of the pitch angle, which is shown in Fig. 11a. Finally, the altitude DOF is controlled by the agent in the fourth curricular step giving the agent full authority on all the system's degrees of freedom. The final tracking time series is shown in Fig. 11b.

In Figs. 12 and 13, the evolution of the policy through the curriculum can be seen. The rows annotated with a * contain the LQR supervisor gains, whereas the others contain ones that have been learned during the curricular step by the agent. The policies reflect the curricular setup presented in Table 3, where the states and their respective references that are learned and tracked during a curricular step are highlighted. This means that they receive a gain that is non-zero (zero gains are white).

In the experiment proposed by this research only two states define the safety of the system, namely roll and pitch, for

[‡]Note that the agent has full observation of the system when learning states in each curricular step.

[§]A similar strategy is used on the other axes: (ϕ, p), (θ, q), (z, w)

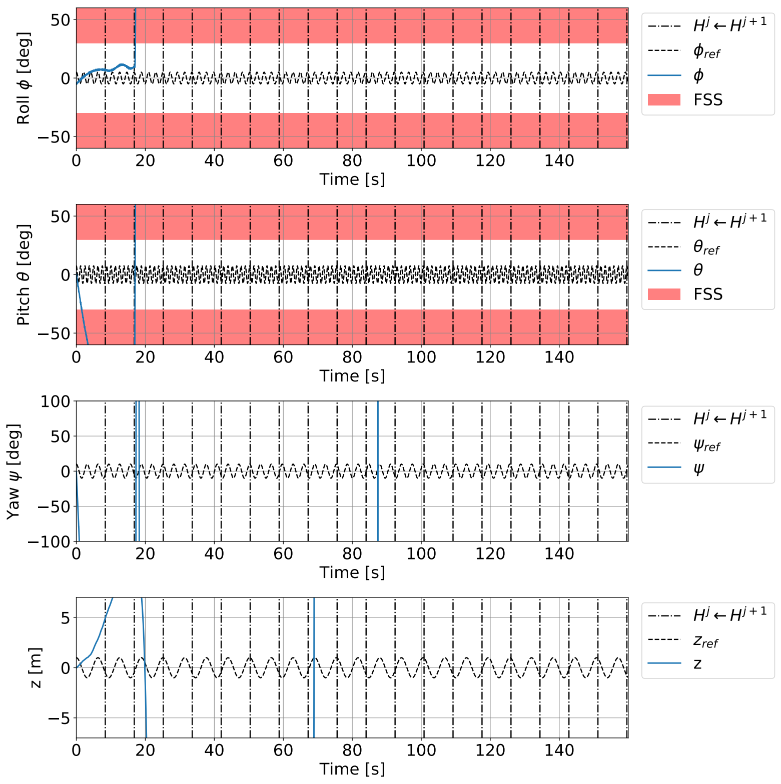


Fig. 9 Evolution of attitude and altitude over time; Flat Learning; Online VI; No Safety Filter (SHERPA)

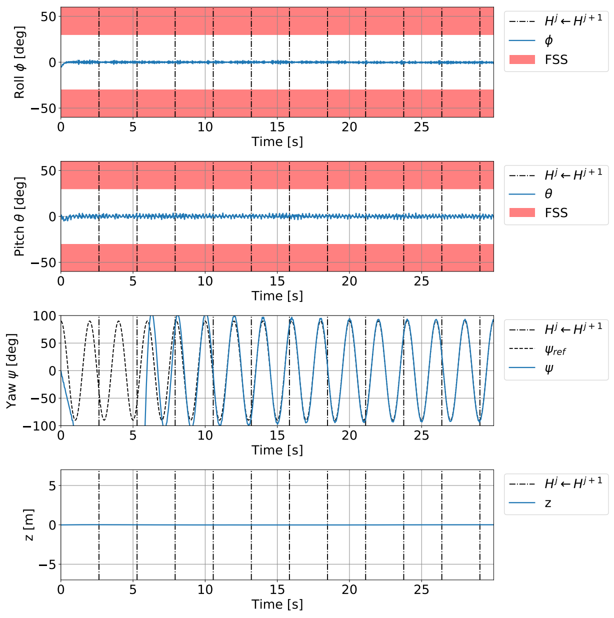
which the FSS limits have been set at ±30°. From figures 10b and 11a, it can be observed that the trajectory in the respective states adventures into the FSS, which is considered to be equivalent to a crash or resulting in an undesired situation. The final module required to complete the paradigm proposed in Fig. 2, is the safety filter SHERPA.

The limitations on roll and pitch angles serve as the principal safety constraints on the system, whereas the position of the quadrotor has been left unbounded. As such, the safety filter is only applied to curricular steps 2 and 3. As seen in figures 10b and 11a, the system goes into the FSS for both the roll and pitch states when the agent learns to control these dimensions. As mentioned in Section II.C, a backup policy has to satisfy two conditions. One is the ergodicity condition where the system has to come back to a previously visited state (or close to the said state). The other is a safe trajectory to reach this previously visited state. As the model used in this experiment consists of decoupled dynamics, the ergodicity can be limited to the set of states directly related to the input learned by the agent at a given curricular step.

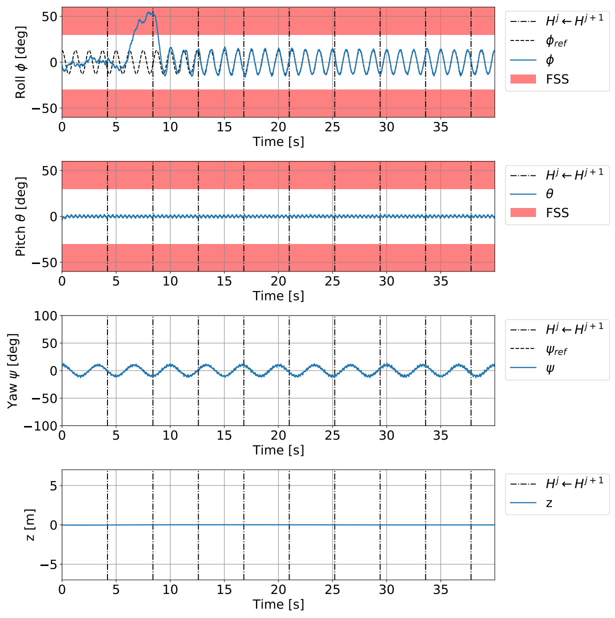
In the second curricular step, the ergodicity and safe trajectory conditions apply solely to state elements related to the input torque around the x-axis, τ_x. More specifically, only the roll angle, ϕ, is checked for inclusion in the SSS. For the ergodicity condition, both roll angle and roll rate, $\{\phi, \dot{\phi}\}$, as well as their respective reference signals, $\{\phi^r, \dot{\phi}^r\}$, are used. Due to the use of augmented states (Section III.A), the reference plays an important role in state propagation and, consequently, the system's safety. The closeness interval is set to ±5° and ±7.5° for roll angle and roll rate states, respectively. Furthermore, the reference states have a closeness interval of ±7.5°. As an initial backup policy, SHERPA uses the first policy given to the agent after the knowledge transfer between curricular steps.

In the subsequent curricular step, SHERPA ensures the safety of both the roll angle and the pitch angle. The ergodicity condition is applied to all states related to the input torques in both directions, τ_x and τ_y. As such, it drastically complicates the process of finding a backup policy. Due to the addition of six dimensions, the space in which to look for these policies becomes relatively large, making the search more computationally expensive [¶].
The safe curriculum learning process is shown in Figs. 14a and 14b for curricular step 2 and 3 respectively. By comparing the propagation of these states with the ones in Figs. 10b and 11a, it can be seen that safety is indeed

[¶]For the 40 second simulation time shown in Fig. 14b the run time was 3h+ on a MacBook Pro Intel i7 2.3 GHz (late 2013)

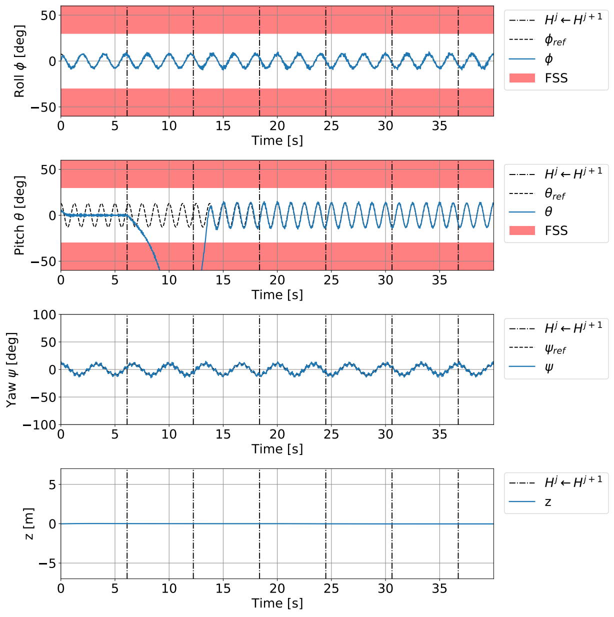


(a) Attitude evolution during curricular step 1; Agent authority = $\{\psi\}$; Learning yaw angle, ψ, tracking.

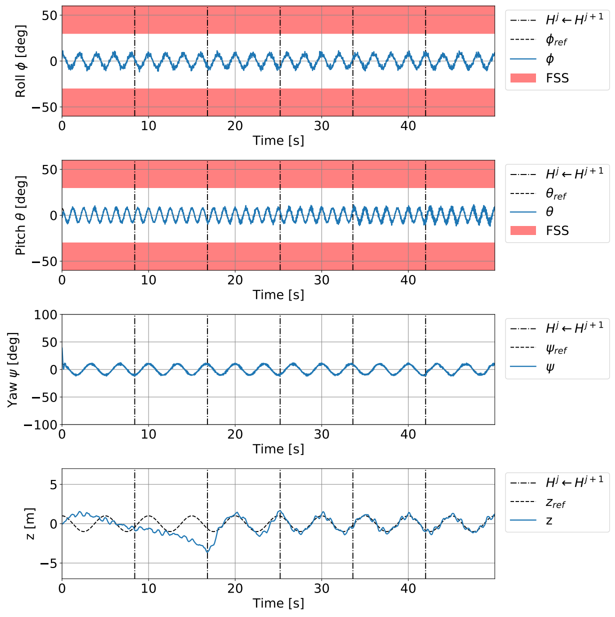


(b) Attitude evolution during curricular step 2; Agent authority = $\{\phi, \psi\}$; Learning roll angle, ϕ, tracking.

Fig. 10 Evolution of attitude angles over time for curricular steps 1 and 2



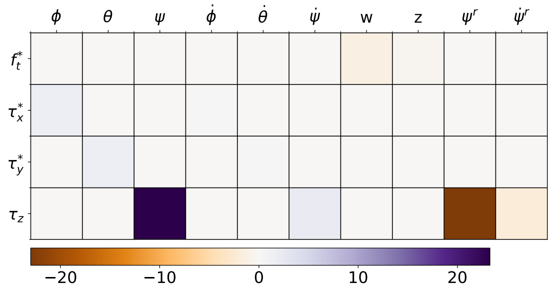
(a) Attitude evolution during curricular step 3; Agent authority = $\{\phi, \theta, \psi\}$; Learning pitch angle, θ, tracking.



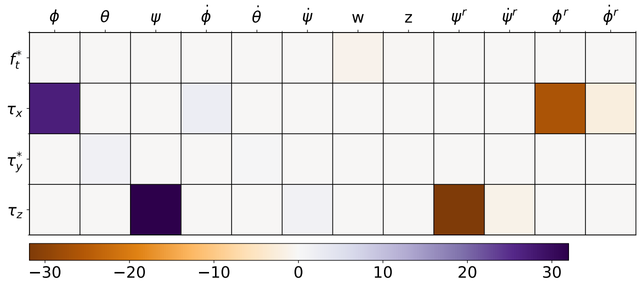
(b) Attitude evolution during curricular step 4; Agent authority = $\{\phi, \theta, \psi, z\}$; Learning altitude position, z, tracking.

Fig. 11 Evolution of attitude angles over time for curricular steps 3 and 4

preserved at all times during the learning process of each respective curricular step. It should be noted, however, that the presence of a safety filter impedes the learning efficiency. This effect can be attributed to the addition of PE before

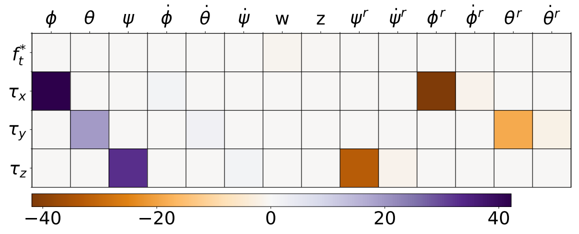


(a) The resultant policy gains after the first curricular step; Online VI; No Safety filter present; * denotes the policy rows that are controlled by the LQR supervisor.

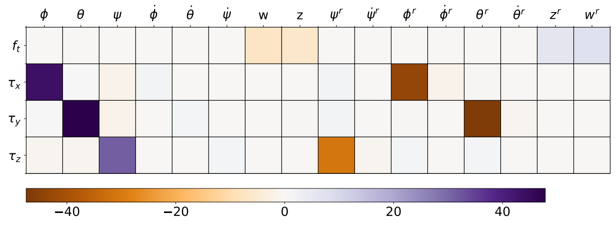


(b) The resultant policy gains after the second curricular step; Online VI; No Safety filter present; * denotes the policy rows that are controlled by the LQR supervisor.

Fig. 12 Resultant policy parameters after the first and second curricular step, left and right, respectively.

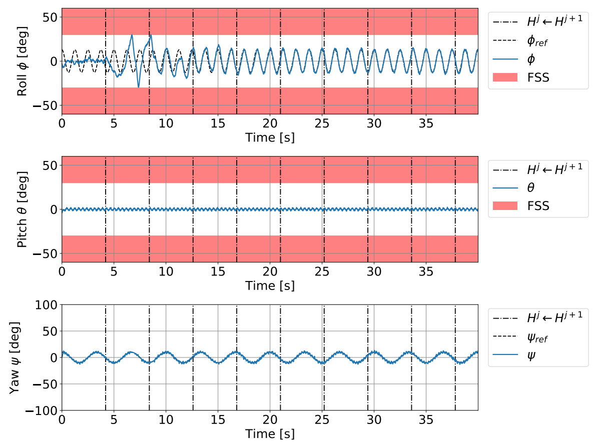


(a) The resultant policy gains after the third curricular step; Online VI; No Safety filter present; * denotes the policy rows that are controlled by the LQR supervisor.

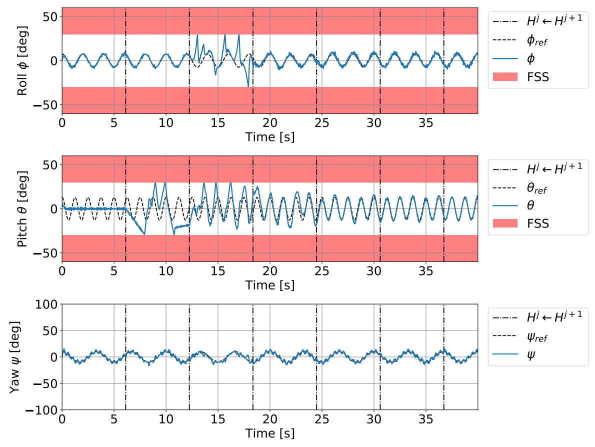


(b) The resultant policy gains after the fourth curricular step; Online VI; No Safety filter present; *denotes the policy rows that are controlled by the LQR supervisor.

Fig. 13 Resultant policy parameters after the third and fourth curricular step, left and right, respectively.



(a) Evolution of attitude angle states during curricular step 2; Agent authority = $\{\phi, \psi\}$; Safe Curriculum Learning; Online VI; SHERPA active on ϕ.



(b) Evolution of attitude angle states during curricular step 3; Agent authority = $\{\phi, \theta, \psi\}$; Safe Curriculum Learning; Online VI; Red is FSS for $\{\phi, \theta\}$; SHERPA active on $\{\phi, \theta\}$.

Fig. 14 Evolution of attitude and altitude over time for Safe Curriculum Learning steps 2 and 3.

the action is passed into the safety filter (as shown in Fig. 2). Without an appropriate PE signal, the agent cannot learn correctly, and policy updates are more likely to be unstable.

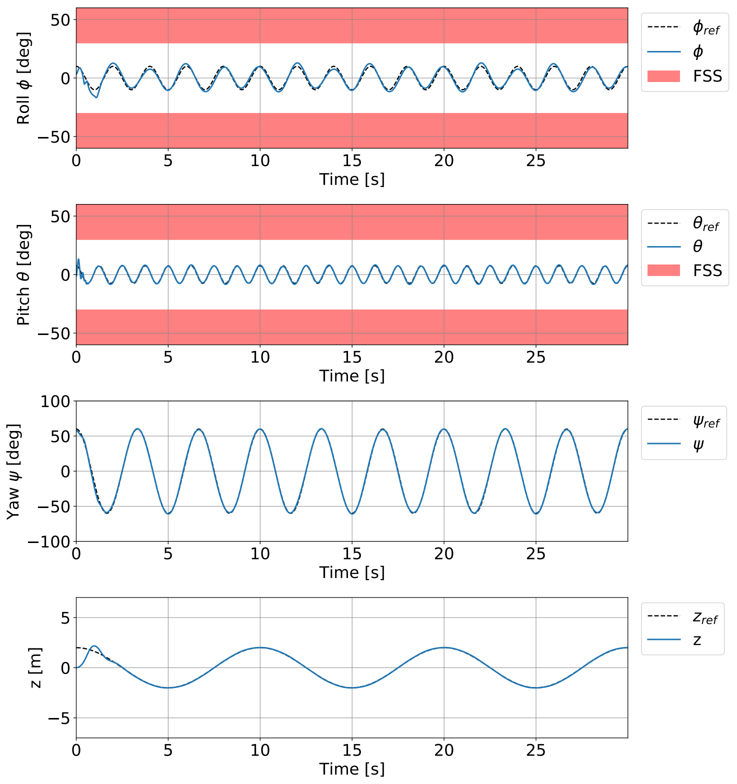


Fig. 15 Evolution of attitude and altitude states using π_Ω (step 5); No learning.

By mapping the policy using the mapping matrix ξ given in Eq. (15), the tracking task could be defined in terms of rotor RPMs. Ultimately, the goal is to obtain a policy for optimal tracking control of a given quadcopter configuration. As such, no further learning is performed during the last curricular step. Instead, the mapped policy is simulated within the environment and the final tracking performance is presented in Fig. 15.

The amplitudes and frequencies of the reference signal were changed in the last step to confirm that no overfitting occurs during the curriculum. Furthermore, it allows for showcasing the range in which the obtained policy can be applied. In Fig. 16a, the rotational speed of each motor blade is outlined. It can be seen that the rotors move in pairs depending on the reference of each state. As shown in Fig. 16b, the tracking performance is similar to a standard LQT controller for which the policy has been generated by solving the augmented LQT algebraic Ricatti equation (ARE) directly [31]:

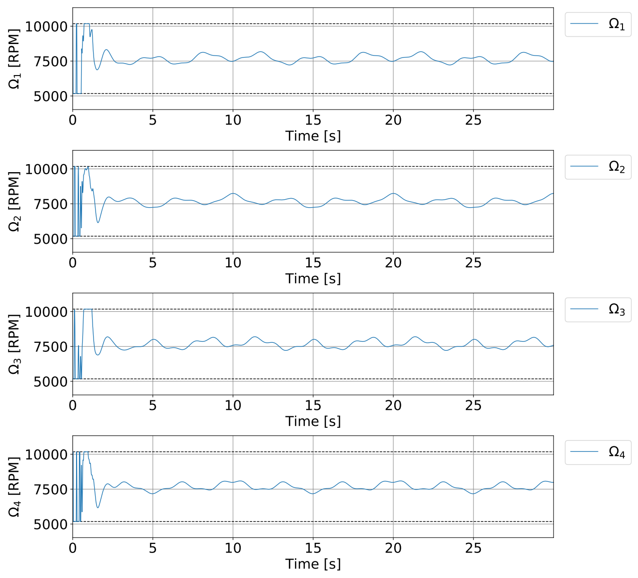
$$Q_1 - P + \gamma T^T PT - \gamma^2 T^T PB_1(R + \gamma B_1^T PB_1)^{-1} B_1^T PT = 0 \tag{16}$$

Note that this direct synthesis requires full knowledge of system dynamics. Even at extreme reference frequencies and amplitudes, these tracking errors remain low, which is determined based on the action saturation imposed on the rotor RPMs.

This section has shown the application case for the proposed paradigm in Fig. 2 on a quadrotor. Due to its larger state and action spaces, this system complicates the agent's task of learning a stabilising and tracking policy. Therefore, a more elaborate curricular strategy was devised to provide the agent with a more gradual learning complexity. Furthermore, modified policy search strategies were put in place within SHERPA to accommodate larger and sparse (safe backup) policy spaces.

C. Discussion

The safe curriculum learning architecture presented in Fig. 2 has proven to be effective in learning an optimal tracking control policy on linear systems with parametric unknowns. However, when analysing the results more closely, it is found that the complexity of the system on which the paradigm is applied has an essential impact on the learning


(a) Evolution of rotor RPMs after mapping the policy (step 5); No learning.

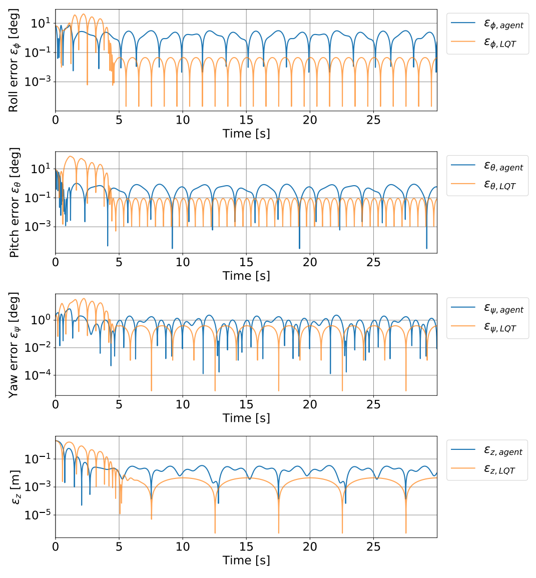

(b) A comparison of error between the system's attitude and their respective reference signals for both the agent and a standard LQT controller.

Fig. 16 Evolution of rotor RPMs over time (left) and the evolution of these states' errors with respect to their reference signal for both the agent and a standard LQT controller.

stability and the learning convergence. When the comparison is made between a more simplistic N-MSD system and a linearised quadrotor model, it is noticeable that the curricular strategy and the safe learning one are more elaborated on the more complex quadrotor system. The additional complexity is also reflected in the wall clock times of each learning process, where the quadrotor simulation requires additional computational power.
Additionally, the complexity of the system on which the paradigm is applied defines the hyperparameter sensitivity similar to the effect of the PE signal on the quadrotor. In contrast with the N-MSD system, the PE signal had to be band-limited for the UAV to achieve conceivably positive results. Moreover, a more advanced backup policy search strategy had to be implemented within SHERPA for the quadrotor to deal with the more sparse safe policy space. In this strategy, a log is kept of previously used backup policies tested during the backup search at each time step in the simulation.

An inherent deficiency was discovered during the experiments regarding the paradigm presented in Fig. 2, namely the position of the PE module block in this figure. An adequate excitation signal is required for the agent to learn stably. However, the safety filter receives the action combined with the PE signal. Should this combination lead to the FSS, the safety filter overwrites the action and provides a new one. The latter lacks an excitation signal. The process is worsened when a backup policy is followed for a relatively large number of time steps, or if multiple backup policies are followed back-to-back. During these interventions, the agent is deprived of the PE signal essential for its learning stability. Besides the agent, the system identification module in the experiments shown in this paper relies on the PE signal for its model estimate. When that signal is removed, the system identification model estimate is more likely to have increased parametric uncertainty. Higher uncertainty in the model has a consequence in the internal projection algorithm of SHERPA, which it uses to find backup policies. It becomes harder to find backup policies, thus the safety filter tends to follow more backup policies in its repertoire. It fits the description of an unstable loop affecting both the learning stability and the system's safety, with a snowball effect. It is, therefore, important to accurately tune the hyperparameters for more complex systems since the paradigm is particularly sensitive to the persistence of excitation and reference signals for the learning efficiency. Additionally, depending on the dynamic stability of the system, the severity of SHERPA's hyperparameters regarding the ergodicity condition has to be adapted to promote the quality of the backup policies. Indeed, with backup policies of high quality, safety can be ensured using SHERPA [27, 29].

Finally, the design of the curriculum and the respective staging strategy still relies on the operator's knowledge of the system's approximate dynamics. In order to successfully transfer knowledge between curricular steps, a sensible mapping is required. However, the agent inherently is not aware of the dynamics when simulated in the environment.

V. Conclusion

The insurgent needs for system controllers that can optimally track a reference signal have become apparent in recent years. Not only does the range of applications for such controllers expand over time, but the complexity of the systems in these applications is increasing as well. This paper proposes a safe curriculum learning architecture where the the fundamental principles of reinforcement learning, curriculum learning, safe learning and system identification are combined to provide a methodological approach for finding optimal stabilising tracking policies for such applications. Through two different experiments, the paradigm has been proven effective in stable learning of such policies for systems with parametric unknowns in their dynamics. Although computationally expensive, the results show it is possible to learn a complex tracking task safely. The complexity of the system on which the learning problem is applied was found to have an important impact on the level of strategies to be used and the sensitivity on hyperparameters. Even though the paradigm proposed in this paper shows great potential for a variety of applications, it has only been demonstrated for linear systems with decoupled dynamics thus limiting the conclusions of this research to linearised systems. However, the results in this paper can serve as a foundation for further research in this direction.

A. Recommendations & Future Work

The research presented in this paper is focused on the analysis and experimentation on time-invariant linear systems. Further research should detail the applicability of the architecture proposed in this paper to non-linear systems and time-variant linear systems. For such implementations, more advanced reinforcement learning approaches are required such as an actor-critic architecture [6]. A system identification module could be added to [8], for example. Furthermore, the system identification module requires adaptation as well to allow accurate model representations of non-linear systems.

Additionally, another limitation that became apparent in the experiments is the dependence on the persistence of excitation signals for stable and convergent learning. In the paradigm presented in this paper the placement of a safety filter between the PE signal and the state propagation in the environment poses a risk to the inclusion of such a signal. One way this could be solved, would be to define an uncertainty that is representative of the PE signal and include it into the model estimate uncertainties coming from the system identification module. Adding PE uncertainty would allow SHERPA to perform its policy projections with the inclusion of PE and return an action with excitation.

Furthermore, the safety filter used in the experiments proposed in this paper, namely SHERPA is relatively computationally expensive. Consequently, the learning process can not happen in real-time on conventional computers, and certainly not onboard. However, the Mannucci et al [27] proposes another method, optiSHERPA, which has lower computational complexities.

Finally, it should be investigated if this paradigm shows equivalent potential when applied on systems with coupled dynamics. Coupled dynamics require a different approach in curricular and safe learning strategies when a supervisor is present. This is due to the actions computed through the policy gain being partially defined by the agent and partially by the supervisor. Consequently, the supervisor has an impact on the cost function in the agent's learning process.

Appendix

The table below contains the hyperparameters for the persistence of excitation signal and the reference signal throughout the curriculum of the quadrotor. Note that the agent does not learn in the 5^{th} curricular step, which justifies the absence of excitation signals in this step.

Table 5 Hyperparameters of persistence of excitation signal and reference signal for each curricular step.

Curricular Steps			1	2	3	4	5
PE	f_t [N]	Amplitude [N]	10.5	13.0	13.0	21.0	-
		Frequency band [Hz]	[50, 100]	[50, 100]	[50, 100]	[50, 100]	-
	τ_x [Nm]	Amplitude [N]	10.5	5.0	5.0	21.0	-
		Frequency band [Hz]	[50, 100]	[50, 200]	[50, 200]	[50, 100]	-
	τ_y [Nm]	Amplitude [N]	10.5	5.0	5.0	21.0	-
		Frequency band [Hz]	[50, 100]	[50, 100]	[50, 100]	[50, 100]	-
	τ_z [Nm]	Amplitude [N]	90.0	15.0	15.0	21.0	-
		Frequency band [Hz]	[50, 100]	[50, 100]	[50, 100]	[50, 100]	-
Reference	ψ [rad]	Amplitude [rad]	$\frac{\pi}{2}$	$\frac{\pi}{18}$	$\frac{\pi}{18}$	$\frac{\pi}{18}$	$\frac{\pi}{4}$
		Frequency [Hz]	0.5	0.3	0.3	0.3	0.3
		Phase [rad]	$[-\frac{\pi}{2}, \frac{\pi}{2}]$	$[-\frac{\pi}{2}, \frac{\pi}{2}]$	$[-\frac{\pi}{2}, \frac{\pi}{2}]$	$[-\frac{\pi}{2}, \frac{\pi}{2}]$	$[-\frac{\pi}{2}, \frac{\pi}{2}]$
	ϕ [rad]	Amplitude [rad]	-	$\frac{\pi}{14}$	$\frac{\pi}{24}$	$\frac{\pi}{24}$	$\frac{\pi}{24}$
		Frequency [Hz]	-	0.8	0.5	0.5	0.5
		Phase [rad]	-	$[-\frac{\pi}{2}, \frac{\pi}{2}]$	$[-\frac{\pi}{2}, \frac{\pi}{2}]$	$[-\frac{\pi}{2}, \frac{\pi}{2}]$	$[-\frac{\pi}{2}, \frac{\pi}{2}]$
	θ [rad]	Amplitude [rad]	-	-	$\frac{\pi}{14}$	$\frac{\pi}{24}$	$\frac{\pi}{30}$
		Frequency [Hz]	-	-	0.8	0.8	0.8
		Phase [rad]	-	-	$[-\frac{\pi}{2}, \frac{\pi}{2}]$	$[-\frac{\pi}{2}, \frac{\pi}{2}]$	$[-\frac{\pi}{2}, \frac{\pi}{2}]$
	z [m]	Amplitude [m]	-	-	-	1	2
		Frequency [Hz]	-	-	-	0.2	0.1
		Phase [rad]	-	-	-	$[-\frac{\pi}{2}, \frac{\pi}{2}]$	$[-\frac{\pi}{2}, \frac{\pi}{2}]$

Additionally, this appendix holds the Q and R matrices used while training the agent throughout the curriculum in Table 6 and the matrices used to determine the gains for the LQR supervisor and the LQT comparative controller in Table 7.

Table 6 Q and R diagonal values of agent throughout the curriculum

Step	Q_{diag}								R_{diag}			
1	-	-	1e5	-	-	1e2	-	-	-	-	1e1	-
2	1e5	-	1e5	1e2	-	0	-	-	1e1	-	1e1	-
3	1e5	1e5	1e5	0	1e2	0	-	-	1e1	1e1	1e1	-
4	1e5	1e5	1e5	0	0	0	1e7	1e4	1e1	1e1	1e1	1e1
5	-	-	-	-	-	-	-	-	-	-	-	-

Table 7 Q and R diagonal values of LQR supervisor and Comparative LQT controller

Controller	Q_{diag}								R_{diag}			
LQR Supervisor	1e5	1e5	1e5	3e0	3e0	3e0	1e1	1e3	1e0	1e0	1e0	1e0
LQT	1e5	1e5	1e5	0	0	0	1e7	0	1e1	1e1	1e1	1e1

References

[1] Ludeña Cervantes, T., Choi, S., and Kim, B., "Flight Control Design using Incremental Nonlinear Dynamic Inversion with Fixed-lag Smoothing Estimation," *International Journal of Aeronautical and Space Sciences*, Vol. 21, No. 4, 2020, pp. 1047–1058. https://doi.org/10.1007/s42405-020-00273-8.

[2] Bengio, Y., Louradour, J., Collobert, R., and Weston, J., "Curriculum learning," *Proceedings of the 26th International Conference On Machine Learning*, Association for Computing Machinery, 2009, pp. 41–48. https://doi.org/10.1145/1553374.1553380, URL https://doi.org/10.1145/1553374.1553380.

[3] Helmer, A., De Visser, C., and Van Kampen, E., "Flexible Heuristic Dynamic Programming for Reinforcement Learning in Quad-Rotors," *2018 AIAA Information Systems-AIAA Infotech @ Aerospace*, Kissimmee, FL, USA, 2018. https://doi.org/10.2514/6.2018-2134, AIAA 2018-2134.

[4] West, J., Maire, F., Browne, C., and Denman, S., "Improved reinforcement learning with curriculum," *Expert Systems with Applications*, Vol. 158, 2020. https://doi.org/10.1016/j.eswa.2020.113515.

[5] Ng, A. Y., Coates, A., Diel, M., Ganapathi, V., Schulte, J., Tse, B., Berger, E., and Liang, E., "Autonomous Inverted Helicopter Flight via Reinforcement Learning," *Experimental Robotics IX*, edited by M. H. Ang and O. Khatib, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006, pp. 363–372. https://doi.org/10.1007/11552246_35.

[6] Sutton, R. S., and Barto, A. G., *Reinforcement Learning: an Introduction*, 2nd ed., MIT Press, Cambridge, MA, 2019.

[7] Garcıa, J., and Fernández, F., "A comprehensive survey on safe reinforcement learning," *Journal of Machine Learning Research*, Vol. 16, No. 1, 2015, pp. 1437–1480.

[8] Pollack, T., and Van Kampen, E., "Safe Curriculum Learning for Optimal Flight Control of Unmanned Aerial Vehicles with Uncertain System Dynamics," *AIAA Scitech 2020 Forum*, San Diego, CA, USA, 2020. https://doi.org/10.2514/6.2020-2100, AIAA 2020-2100.

[9] Lillicrap, T., Hunt, J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D., "Continuous control with deep reinforcement learning," *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, San Juan, Puerto Rico, 2016.

[10] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O., "Proximal policy optimization algorithms," *arXiv:1707.06347*, 2017.

[11] Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P., "Trust Region Policy Optimization," *Proceedings of the 32nd International Conference on Machine Learning*, Vol. 37, Lille, France, 2015, pp. 1889–1897.

[12] Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., Graepel, T., and Hassabis, D., "Mastering the game of Go without human knowledge," *Nature*, Vol. 550, No. 7676, 2017, pp. 354–359. https://doi.org/10.1038/nature24270.

[13] Silver, D., Huang, A., Maddison, C., Guez, A., Sifre, L., Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D., "Mastering the game of Go with deep neural networks and tree search," *Nature*, Vol. 529, 2016, pp. 484–489. https://doi.org/10.1038/nature16961.

[14] Burden, J., and Kudenko, D., "Using uniform state abstractions for reward shaping with reinforcement learning," *Workshop on Adaptive Learning Agents (ALA) at the Federated AI Meeting*, 2018.

[15] Lee, Y., and Grauman, K., "Learning the easy things first: Self-paced visual category discovery," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, USA, 2011, pp. 1721–1728. https://doi.org/10.1109/CVPR.2011.5995523.

[16] Taylor, M., and Stone, P., "Transfer learning for reinforcement learning domains: A survey," *Journal of Machine Learning Research*, Vol. 10, 2009, pp. 1633–1685.

[17] Taylor, M., and Stone, P., "Representation transfer for reinforcement learning," *AAAI Fall Symposium: Computational Approaches to Representation Change during Learning and Development*, 2007, pp. 78–85.

[18] Taylor, M., Stone, P., and Liu, Y., "Transfer learning via inter-task mappings for temporal difference learning," *Journal of Machine Learning Research*, Vol. 8, 2007, pp. 2125–2167.

[19] Wang, L., Theodorou, E., and Egerstedt, M., “Safe Learning of Quadrotor Dynamics Using Barrier Certificates,” *2018 IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, QLD, Australia, 2018, pp. 2460–2465. https://doi.org/10.1109/ICRA.2018.8460471.

[20] Huh, S., and Yang, I., “Safe reinforcement learning for probabilistic reachability and safety specifications: A Lyapunov-based approach,” *arXiv:2002.10126*, 2020.

[21] Basu, A., Bhattacharyya, T., and Borkar, V., “A learning algorithm for risk-sensitive cost,” *Mathematics of Operations Research*, Vol. 33, No. 4, 2008, pp. 880–898. https://doi.org/10.1287/moor.1080.0324.

[22] Torrey, L., and Taylor, M., “Help an agent out: Student/teacher learning in sequential decision tasks,” *Proceedings of the Adaptive and Learning Agents workshop (at AAMAS-12)*, 2012, pp. 41–48.

[23] Geibel, P., and Wysotzki, F., “Risk-Sensitive Reinforcement Learning Applied to Control under Constraints,” *Journal of Artificial Intelligence Research*, Vol. 24, No. 1, 2005, p. 81–108.

[24] García, J., and Fernández, F., “Probabilistic Policy Reuse for Safe Reinforcement Learning,” *ACM Transactions on Autonomous and Adaptive Systems*, Vol. 13, No. 3, 2019. https://doi.org/10.1145/3310090.

[25] Mannucci, T., Van Kampen, E., de Visser, C., and Chu, Q., “SHERPA: A safe exploration algorithm for reinforcement learning controllers,” *AIAA Guidance, Navigation, and Control Conference*, Kissimmee, FL, USA, 2015. https://doi.org/10.2514/6.2015-1757, AIAA 2015-1757.

[26] Mannucci, T., “Safe Online Robust Exploration for Reinforcement Learning Control of Unmanned Aerial Vehicles,” Ph.D. thesis, Delft University of Technology, 2017. https://doi.org/10.4233/uuid:dbaf67cc-598c-4b26-b07f-5d781722ebfd.

[27] Mannucci, T., Van Kampen, E., De Visser, C., and Chu, Q., “Safe Exploration Algorithms for Reinforcement Learning Controllers,” *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 29, No. 4, 2018, pp. 1069–1081. https://doi.org/10.1109/TNNLS.2017.2654539.

[28] Fernández, F., and Veloso, M., “Probabilistic policy reuse in a reinforcement learning agent,” *Proceedings of the International Conference on Autonomous Agents*, 2006, pp. 720–727. https://doi.org/10.1145/1160633.1160762.

[29] Moldovan, T., and Abbeel, P., “Safe exploration in Markov decision processes,” *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, Vol. 2, 2012, pp. 1711–1718.

[30] White, L., Togneri, R., Liu, W., and Bennamoun, M., *Neural Representations of Natural Language*, Springer Singapore, Singapore, 2019, Studies in Computational Intelligence, Vol. 783, Chap. Introduction to Neural Networks for Machine Learning. https://doi.org/10.1007/978-981-13-0062-2_1.

[31] Kiumarsi, B., Lewis, F. L., Modares, H., Karimpour, A., and Naghibi-Sistani, M. B., “Reinforcement Q-learning for optimal tracking control of linear discrete-time systems with unknown dynamics,” *Automatica*, Vol. 50, No. 4, 2014, pp. 1167–1175. https://doi.org/10.1016/j.automatica.2014.02.015.

[32] Lewis, F. L., Vrabie, D., and Syrmos, V. L., *Optimal Control*, 3$^{\text{rd}}$ ed., John Wiley & Sons, 2012.

[33] Kiumarsi, B., Lewis, F., Naghibi-Sistani, M.-B., and Karimpour, A., “Optimal tracking control of unknown discrete-time linear systems using input-output measured data,” *IEEE Transactions on Cybernetics*, Vol. 45, No. 12, 2015, pp. 2770–2779. https://doi.org/10.1109/TCYB.2014.2384016.

[34] Narendra, K., and Annaswamy, A., “Persistent excitation in adaptive systems,” *International Journal of Control*, Vol. 45, No. 1, 1987, pp. 127–160. https://doi.org/10.1080/00207178708933715.