doi: 10.82124/CEAS-GNC-2026-033 CEAS-GNC-2026-033

EuroGNC Navigation Guidance Control CEAS Madrid, Spain May 5th-7th 2026
uc3m Universidad Carlos III de Madrid AIAA

Safe, Hybrid Control Allocation for the Innovative Control Effectors (ICE) Aircraft

Hasan Isci Ph.D. Student, Istanbul Technical University, Munich, Germany. iscih@itu.edu.tr

Emre Koyuncu Aerospace Research Center / Istanbul Technical University, Istanbul, 34469 . emre.koyuncu@itu.edu.tr

ABSTRACT

This study presents a safe, frequency-separated hybrid control-allocation approach that combines a fast convex quadratic program (QP) with a slow reinforcement-learning (RL) guidance-layer. The fast allocation runs frame-wise at high rates, solving a strictly convex control allocation optimization problem via quadratic programming with equality slack, position and rate constraints. The slow layer is moment invariant, dynamically manipulating weights to redistribute control effort within the null space of the local control effectiveness. The Innovative Control Effectors (ICE) aircraft is used as the testing framework for the proposed approach. Results show that this allocation architecture can reduce cumulative control allocation error over a period and better distribute commands across available control effectors without injecting unintended moments into system.

Keywords: Dynamic control allocation, Reinforcement learning, Quadratic programming, Over-actuated aircraft, Null space, Hybrid allocation, Innovative Control Effectors (ICE)

1 Control allocation in incremental form

Modern high-performance aircraft designs especially those with redundant unconventional surfaces such as tailless or blended-wing-body ones, rely on over-actuated control architectures. In these designs, multiple effectors have similar control effectiveness, which requires that the distribution of commands be done intelligently. Control allocation methods provide a systematic way to translate pseudo control demands—usually generated by inner-loop flight control laws—into surface deflections. This approach not only decouples the control law design from actuator-specific limitations but also allows for optimal resource usage, constraint handling, and fault tolerance.

For the angular motion of an aircraft, the system dynamics can be given as

$$\dot{x} = f(x, u) \tag{1}$$

Where x is the aircraft's angular rate state vector, and $u \in \mathbb{R}^m$ is the vector of control surface positions. In real applications, the surface movement is limited in terms of position and rate, which translates to the following constraints:

$$u_{min} < u < u_{max} \quad , \quad |\dot{u}| < \dot{u}_{max} \tag{2}$$

Where u_{min} and u_{max} are lower and upper bounds for the surface position, $\dot{u}_{max}$ is the maximum surface deflection rate in each direction. A successful control allocation algorithm should respect the position and rate limits to avoid over-allocating the surfaces. Equation (1) could be approximated as input affine form in equation (3).

$$\dot{x} = f(x) + B(x,u)\,u \tag{3}$$

Where $B(x,u)$ is the control effectiveness matrix mapping surface deflections to angular acceleration. Following the incremental nonlinear dynamic inversion technique [1] the desired control torques can be generated as $\Delta\tau_d$ as to be satisfied via the surface deflections.

$$\Delta\tau_d \approx I\,(\dot{x}_d - \dot{x}_{act}) \tag{4}$$

$\dot{x}_d$ are the desired state derivatives while $\dot{x}_{act}$ is the actual approximated derivatives for the body fixed angular rate states. I is the moment of inertia of the aircraft in three dimensions. Then, the control surface command can be found as given in equation (**5**) when the incremental control allocation approach is applied from [2]:

$$B_{loc}\,\Delta u \approx \Delta\tau_d \quad , \quad u = u_{act} + \Delta u \tag{5}$$

Where $B_{loc} \in \mathbb{R}^{3\times m}$ is the local control effectiveness model calculated as a local jacobian of the control effectiveness matrix at a given aircraft state and the surface position. [2] also shows that using local Jacobian of the control effectiveness matrix reduces the model dependency and increases robustness. u is the total control surface command, u_{act} is the current surface deflections and Δu is incrementally allocated control surface commands per allocation cycle.

2 Innovative Control Effectors aircraft

The Innovative Control Effectors (ICE) aircraft is a tailless, delta-wing research configuration introduced in 1990s [3] to study aircraft control related topics in a tailless low observable configuration. Due to its aerodynamic design, the aircraft has multiple redundant control surfaces that generate highly nonlinear characteristics. In 2016, Niestroy et al. published a nonlinear simulation model of the aircraft for public research [4].

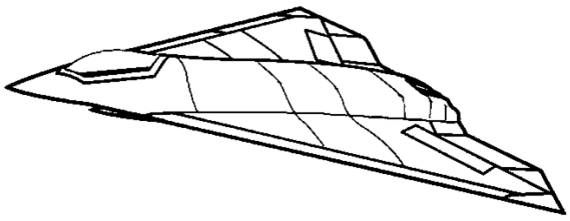

Figure 1: Innovative Control Effectors aircraft

The aircraft includes 13 individual control effectors such as two inboard/outboard leading-edge flaps (LEFi/o), two all moving tips (AMT), two elevons (ELE), two spoilers (SSD), one flap (PF) and 2-axis thrust vectoring (TV) for providing desired moments in 3 axes. These effectors have overlapping authority and strong cross-coupling behavior which leads to rich null space properties and often actuator saturation. The 6DoF nonlinear simulation model of the aircraft in Simulink has been used in this work as the aircraft platform for control allocation study. For simplicity, thrust vectoring is not used in this study, and its pitch and yaw components are kept fixed at their neutral point.

3 Nonlinear Quadratic Programming Control Allocation Approach

Recalling the equation (**5**) as,

$$B_{loc}\,\Delta u \approx \Delta\tau_d \tag{6}$$

Where $B_{loc}$is the local effectiveness matrix, $\Delta\tau_d \in \mathbb{R}^3$ is the delta desired torque and $\Delta u \in \mathbb{R}^m$ is the small change in the effector deflection. The use of nonlinear programming for finding Δu has been studied in various studies [5], [6], [7]. The nonlinear optimization approach is applied in equation (**7**) for finding control surface commands to satisfy a given desired torque.

$$\begin{aligned} \min_{\Delta u,\ \xi}\ & \frac{1}{2}\,\Delta u^T\, W_r\,\Delta u \ + \frac{1}{2}\xi^T\, Q_s\,\xi \\ s.t.\quad & B_{loc}\Delta u \ + \xi = \Delta\tau_{\mathrm{d}} \\ & \Delta u_{min} < \Delta u < \Delta u_{max} \\ & u_{min} < u_{act} + \Delta u < u_{max} \end{aligned} \tag{7}$$

W_r is the weight matrix for each effector command which can be selected arbitrarily and influences the selection of the allocated surfaces while respecting the constraints. u_{act} are the current effector positions. u_{min} and u_{max} are the position limits of each effector while Δu_{min} and Δu_{max} are the maximum rate that the effectors can achieve at one step.

The minimization problem defined in equation (**7**) is following a similar structure given in dynamic control allocation ([8], [9]). However, the dynamic control allocation problem formulation has been enhanced slightly with an additional term slack $\xi \in \mathbb{R}^3$. Slack is the residual between the desired torque and the torque allocated by the selected control effectors Δu per frame. Ideally, the slack term should always converge to zero whenever the desired torque is attainable. However, it is included to have a convex problem in cases where the desired torque is not attainable. The slack weight Q_s is chosen significantly larger than terms associated with Δu in the cost function, ensuring that the optimizer prioritizes reducing the slack first. Specifically, the magnitude of Q_s is scaled to be at least twice the maximum attainable moment to guarantee tracking priority without causing numerical ill-conditioning in the solver. Furthermore, Q_s incorporates simple axis prioritization (pitch > roll > yaw). Because the tailless aircraft possesses highly limited yaw authority, strictly enforcing yaw tracking can quickly drive effectors to premature saturation, thereby degrading the critical pitch and roll control.

While an unbalanced slack weight may lead to instabilities in optimization, with a proper selection of the slack weight, the problem remains valid for control allocation purposes [10]. As long as $W_r > 0$ and $Q_s > 0$ conditions are satisfied, the optimization problem is unique and convex. There is no explicit definition of the long-term (or slow frequency) behavior of the allocator as opposed to dynamic allocation. Steady-state effector recentering has been removed from the allocation formula in favor of the hybrid allocation strategy, which will be presented in the following section.

W_r determines per effector rate effort. Larger weight means less movement when achieving the desired moment. Therefore, it leads to prioritization between the effectors. This prioritization turns into something that could be selected arbitrarily depending on current flight and effector states, however selection of the prioritization matters for many aspects such as effector wear, long term allocation performance, control surface related drag force, etc. While the classical solution methods are optimal for the problem as posed, their performance is heavily influenced by the choice of weights W. Traditionally,

W is fixed offline based on performance, efficiency, or failure resilience. However, a static choice may underperform under varying flight regimes or actuator stress.

4 Hybrid Control Allocation Approach

There may be various alternatives to the selection of the weights in nonlinear quadratic programming approach. They could be fixed weights that have been chosen at the design phase or alternatively, various fixed weight sets could be chosen depending on the flight phases. In addition to the fixed weights selection, they could be assigned/updated dynamically by higher level logics as it is in [11]. There have been studies on using the global optimization capabilities of reinforcement learning techniques to solve the control allocation problem directly in [12], [13], [14]. There are also interesting studies that combine robust control design techniques and reinforcement learning for determining solutions to dynamic optimization problems [11].

This study focuses on selecting the weights in the incremental dynamic control allocation problem via a slowly changing guidance layer based on a trained intelligent agent. A reinforcement learning agent, which uses feedback from the aircraft states and the dynamic control allocation, is utilized as the guidance layer. This is called a hybrid dynamic control allocation approach that leverages the global optimization capability of reinforcement learning to guide the local QP optimization slowly to get better performance over a period. To accurately reflect that the core allocation is still solved mathematically by a quadratic programming solver under the supervision of the agent, this hybrid architecture will hereafter be referred to as the RL-guided QP. The overall scheme of the RL-guided QP is given in Figure 2.

The main highlights of the RL-guided QP are listed as follows:

- The proposed approach can be applied to the safety critical designs as the RL agent cannot override the effector commands directly. The agent acts only through the null space of the local B matrix as the resulting behavior. What the agent can change is the instantaneous work across the effectors.
- The agents' outputs are filtered appropriately to slowly adjust the effector's workload depending on higher level objectives. The actions of the agent are therefore aligned with the purpose of the meta-layer.
- There is only one quadratic programming call. The execution time is bounded, and worst-case execution is deterministic due to the convex formulation of the optimization cost.
- The RL agent learns how to efficiently spread the control surfaces depending on the current effectors' position and rate availability.

In simpler words, the reinforcement-learning meta control-allocation layer dynamically guides the safety-critical deterministic QP control allocator. The meta layer uses aircraft and actuator state information when shaping that guidance. Over a time horizon, this allows the hybrid allocator to achieve lower actuator usage, more rate and position availability, and a faster slack decay rate by gradually repositioning the initial actuator operating points.

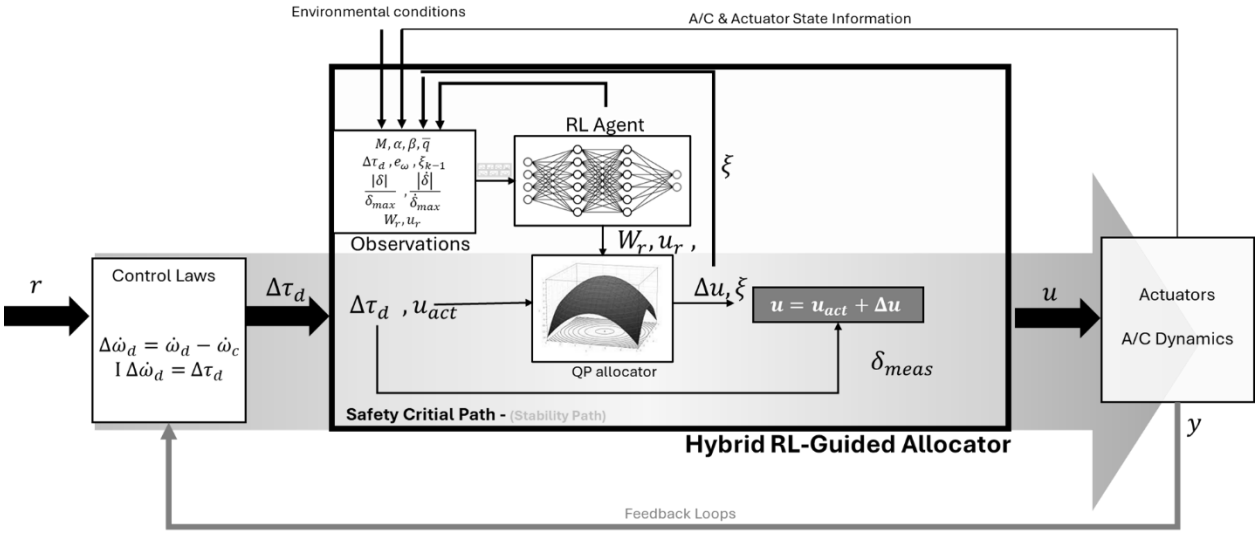

Figure 2: Hybrid Control Allocation (RL-guided QP) Scheme

It is important to discuss the conditions under which dynamic alterations to the weighting matrix W_r guarantee this moment invariance. When the desired torque $\Delta\tau_d$ is fully attainable and the actuator position and rate constraints are inactive (i.e., operating in the attainable moment set), the slack variable vanishes ($\xi = 0$). Under these unconstrained feasible conditions, the optimization solution analytically reduces to the W_r-weighted pseudoinverse. Consequently, any adjustment to W_r by the guidance layer strictly redistributes the control effort Δu within the null space of the local effectiveness matrix B_{loc}, leaving the equality constraint $B_{loc}\,\Delta u = \Delta\tau_d$ unchanged. If the constraints become active, W_r modifications may influence the active set, but the heavily penalized slack weight Q_s ensures that moment tracking remains the primary objective.

Because the meta-layer is designed such that its guidance cannot change the allocated torque of the underlying QP solver, it cannot directly improve the single-frame control allocation performance. Instead, the main benefit of using a reinforcement learning agent in this role is the ability to optimize higher-level objectives over time. This leads to better actuator usage and lower cumulative slack. Introducing a learning-based agent in this structure does not create a safety risk in the hybrid allocation scheme.

The observation array of the agent includes information from various aspects:

- Aircraft state information: Mach number, angle of attack, angle of sideslip, dynamic pressure
- Actuator state information: Rate and position availability of each effector, total rate and position availability
- Demand information: desired torque, current slack information,
- Inner loop tracking information: errors of angular rate tracking
- Previous actions of the agent

Selection of the observations is important to ensure the Markov process properties in the RL problem formulation. As the nonlinearity of the control allocation problem is very high, the selection of the observations has major impact on supporting the agents learning and the sample efficiency in training.

Table 1: List of observation elements

	Group	Size	Description
Mach	A/C State information	1	
Angle of attack		1	
Angle of sideslip		1	
Dynamic pressure		1	
Angular rate command	Control Demand	3	The angular rate command value sent to the inner loop controllers
Tracking error		3	The error signal of angular rate tracking loops
Norm of Desired torque		1	$\|\Delta\tau_d\|_2$
Slack	Allocation Status	3	ξ allocation error of the previous step
Effector rates		m	Current rate value of each effector
Mean of effector rates		1	Mean value of effector rates
Effector positions		m	Current position of each effector
Mean effector positions		1	Mean value of effector positions
Reserve effector positions		m	$1-\frac{\lvert u_{act}\rvert}{u_{max}}$ How much deflection is left for each effector
Reserve effector rates		m	$1-\frac{\lvert \Delta u\rvert}{\Delta u_{max}}$ How much rate is left for each effector
Effector Effectiveness		m	$\left\|\mathrm{B}_{\mathrm{loc_j}}\right\|_2$ where $j=1..m$ 2-norms of local effectiveness matrix' columns
Effector Alignment		m	Cosine alignment between desired and effector torque directions for each effector
Previous weights	Command History	n	ρ_{norm} in previous step

All the elements in the observation vector are normalized by their maximum values. Therefore, the input to the agent is in range of $[-1, 1]$. m in the table shows the effector number which is 11 for the ICE aircraft when the thrust vectoring effectors are excluded. The total number of elements in the observation array is 87. n in the table shows the number of actions for the agent.

The actions of the agent are as follows:

- Grouped multipliers $a \in [-1, 1]^n$ where $n = 5$ for the PF, LEF, ELE, AMT and SSD.

Then, per group RL agent actions are mapped to individual effector weights as the diagonal elements of W_r. However, there are several intermediate steps that are applied to safely map the actions to the underlying quadratic programming allocator.

$$\rho_{raw,i} = scale(a_i; w_{min}, w_{max}) \quad i = 1,\ldots,5\ (PF, LEF, ELE, AMT, SSD) \tag{8}$$

Where the function $scale(.)$ is just an affine map from the RL agent's action range to a proper range for the weight elements.

$$scale(a_i; w_{min}, w_{max}) = \frac{1}{2}(a_i + 1)(w_{max} - w_{min}) + w_{min} \tag{9}$$

The function given in equation (**9**) simply translates the action inputs to $[w_{min}, w_{max}]$ range linearly. To satisfy the conditions required for W_r, w_{min} and w_{max} are selected as positive numbers around normal value 1. For the convex optimization problem, what is important for the optimizer is the ratio between the effectors. In other words, "*how expensive to choose one effector to the another comparably*" is the only quantity that matters to the optimizer. Therefore, to keep the actions consistent in all frames, the raw weight values are geometrically normalized (ρ_{norm}) before used in weighting matrix W_r. This ensures that the average values of weights stay around 1. This also means to the underlying QP optimizer (or to the RL agent on a slower scale) that there is no way of making all effectors cheap/expensive. Finally, the corresponding diagonal element of W_r is selected from one of the normalized group weights ρ_{norm}. The relative ratio between the effectors is selected as 3 in the study. Thus, an appropriate range is selected for min and max values of the weights.

The meta-RL guidance layer runs at 5Hz while the underlying quadratic programming solver runs at 100Hz in the RL-guided QP allocator. However, the outputs of the RL agent are filtered by 2Hz low-pass filters to avoid chattering and only introduce changes to the allocator at an appropriate rate.

The RL agent is trained for a piecewise reward function that focuses on two complementary objectives:

- When an allocation slack exists, reduce it as quickly as possible
- Minimize sustained effector stress when the moment is attainable.

The reward computed over a short sliding window of simulation time for 0.2 seconds to match the control frequency of the RL-guidance layer. The reward should also include the temporal information about the environment in between its samples to enhance the training.

Let the window length be H frames, the windowed sequences are then defined: ($k = 1, \ldots . H$ showing underlying fast QP calculation steps)

- $E(k)$: Normalized allocation slack $\left\| \frac{\xi}{\tau_{ref}} \right\|_2$ at frame k, where τ_{ref} is constant reference value for normalization.
- $P_{RL}(k)$ and $\mathrm{R_{RL}}(k)$: Mean position and rate reserve values at frame k for RL-guided QP allocator.
- $P_{QP}(k)$ and $R_{QP}(k)$:Mean position and rate reserve values at frame k for a shadow fixed weight QP allocator.

The shadow fixed-weight-QP allocator is the same quadratic programming solution at the calculation instance but without the weights resulted from the RL agent. The shadow fixed-weight-QP allocator is only used during the training phase and the weight values of this allocator are constant and selected before the training. The purpose of the shadow QP is to provide delta feedback to the RL agent about how the weight selections affected the baseline solver in RL-guided QP. In other words, a shadow QP allocator is used to derive the reward signal during the training phase.

$$r = \begin{cases} r_{slack}, & if\ slack\ is\ present \\ r_{duty}, & otherwise \end{cases} \tag{10}$$

The reward value r is selected based on allocation error. If there is an allocation error (slack) exist, the agent is rewarded by r_{slack} which focuses on how much the slack has been reduced in the time window. If there is no allocation error during the time window, then the agent is rewarded by r_{duty} which focuses on how long the effectors remain at their position and rate limits.

The average slack drop is estimated from the average of discrete slope which is defined in equation (**11**).

$$\Delta E_{avg} = \frac{1}{H-1}\sum_{k=1}^{H-1} E(k) - E(k+1) \tag{11}$$

Then the reward value is defined as:

$$\mathrm{r_{slack}} = -\beta\ \min\ (0, \Delta E_{avg}) \tag{12}$$

r_{slack} simply means that the agent is rewarded for the average slack drop. The opposite is not true: if the command is not attainable and there is nothing the allocator can do, then there is also nothing meaningful the RL guidance can do. To improve training stability, the parts where the slack is increasing are treated as neutral instead of penalized. Only the parts where the RL agent can actively help reduce the slack faster are considered, and those successful actions are rewarded positively.

When the desired torque is achievable, the goal of the RL guidance shifts; track the desired torque efficiently. The intention of being efficient is to minimize the effector use and reserve as much as possible for future demands. As the control allocator cannot predict the future demand, the best option that an allocator can have is to minimize the limit effector position and rate usage. Mean position and rate limit usage are defined in equations 13. The same approach has been taken for both RL-guided QP allocator ($\bar{U}_{RL}$ and $\bar{R}_{RL}$) and the fixed-weight shadow QP allocator ($\bar{U}_{QP}$ and $\bar{R}_{RL}$).

$$\bar{\mathrm{U}} = \frac{1}{H}\sum_{k=1}^{H} U(k) \qquad \bar{\mathrm{R}} = \frac{1}{H}\sum_{k=1}^{H} R(k) \tag{13}$$

$U(k)$ and $R(k)$ are the fraction of actuators at a position and rate limit. Being at a position limit is defined as a hitting either $u \le u_{min}$ or $u \ge u_{max}$ for the position whereas, $|\Delta u| \ge \Delta u_{max}$. The calculation of the fractions U and R for every step k are given in equation (14).

$$\mathrm{U(k)} = \frac{1}{m}\sum_{i=1}^{m} 1\left\{ u_{act_i}(k) \le u_{\mathrm{i,\ min}} \ \ or\ \ u_{act_i}(k) \ge u_{\mathrm{i,\ max}} \right\}$$

$$\mathrm{R(k)} = \frac{1}{m}\sum_{i=1}^{m} 1\,\{\,|\Delta u(k)| \geq \Delta u_{max}\,\} \tag{14}$$

After that the current limit effector usage for each allocator becomes $D = \bar{U} + \bar{R}$ meaning that how much of the nominal effector availability has been consumed for each calculation step. Then, the agent can be simply rewarded as given in equation (15).

$$\mathrm{r_{duty}} = D_{QP} - D_{RL} = \left(\bar{U}_{QP} + \bar{R}_{QP}\right) - \left(\bar{U}_{RL} + \bar{R}_{RL}\right) \tag{15}$$

The reward defined in equation (15) simply focuses on reducing effector usage relative to the shadow QP allocator. The agent is not rewarded for reproducing the baseline solution (which is given by the shadow QP); instead, the agent is rewarded in situations where the overall duty of the effectors is lower than the shadow fixed-weight QP. This directly tells the agent that it must adjust the allocation weights in a way that it performs better effector reserve while still achieving the current desired torque.

Under this reward scheme, an RL agent can learn how to efficiently distribute the weights in alignment with the aircraft's aerodynamic characteristics and the control surface couplings. A deterministic proximal policy optimization (PPO) agent [15] has been selected as the reinforcement learning policy.

5 Training and Simulation Results

During training, randomly generated angular rate commands are used to generate the desired torque inputs to the allocators, and these serve as the driving inputs for each fixed length episode. Rather than sampling these commands uniformly, a weighted distribution is applied to intentionally expose the agent to critical edge cases. The sampling is heavily biased toward generating very high torque demands, deliberately pushing the aircraft toward actuator saturation and severe allocation error to force the agent to learn aggressive recovery tactics. To ensure the learned policy remains smooth and efficient during standard operations, the distribution is gradually tapered to include low-magnitude commands representative of normal, steady-state flight.

A Gymnasium environment [16] has been created that uses the code generated revision of the ICE nonlinear aircraft model together with the nonlinear rate and position limited actuator models. The observation and reward calculations are included in the MATLAB Simulink environment. The source code of the Simulink model has been wrapped by a custom Gymnasium environment class. The PPO agent then trained by using RL_zoo library [17] which uses SB3 [18] reinforcement library to instantiate an agent.

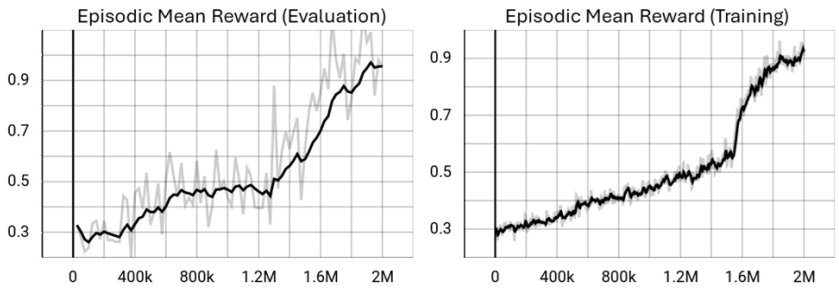

Figure 3: Training Results

Figure 3 shows the training results for the PPO agent. 2M training steps have been iterated and the agent approaches a satisfactory level for the intended action by the reward function. The training has been done on a standard laptop MacBook Air M1 Apple Silicon and Pytorch libraries.

Figure 4 shows the allocation results of the RL-guided QP allocator and a fixed-weight standard QP allocator under the same angular rate command inputs. Each subplot title in Figure 4 reports assessment metrics for that allocation scenario. Consistent with the metrics used in the observations and the reward, these include: the 2-norm of the current allocation error, the forward difference of the allocation error norm, the 2-norm of the current desired moment, the relative ratio of the norm of the local effectiveness matrix, and the U and R reserve metrics that represent available effector position and rate.

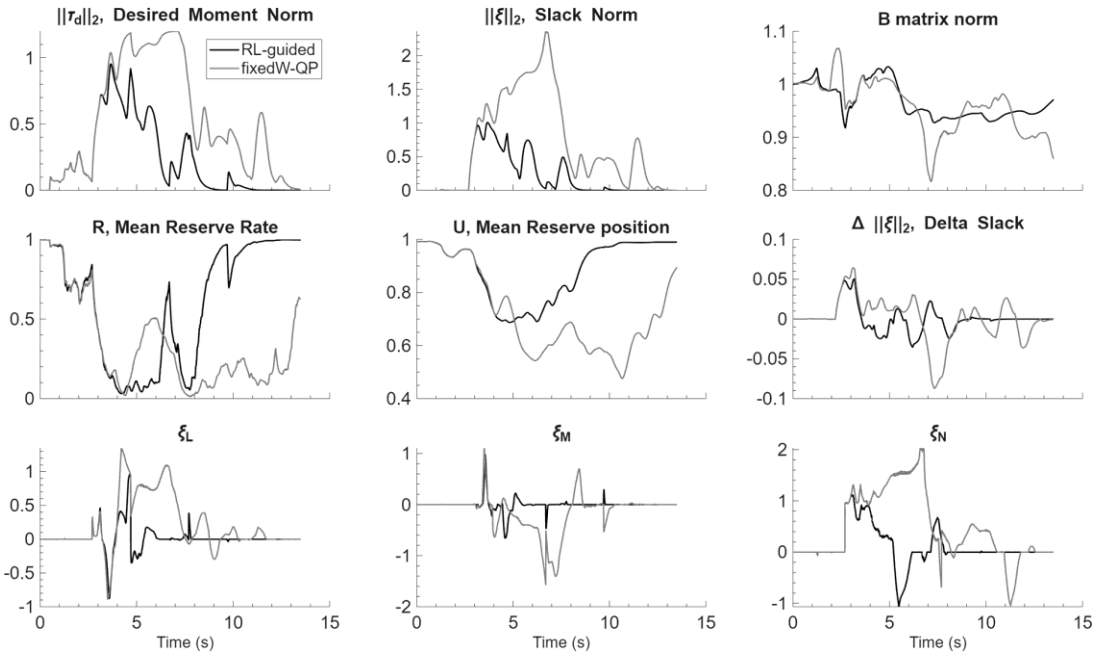

Figure 4: Allocation result of an evaluation episode (extreme flight conditions)

The RL-guided QP outperforms the standard QP allocator in the given episode. It should be noted that the desired torque is quite high for most of the simulation, so both allocators are effectively forced to drive their local effectiveness matrices B toward the commanded moment. The RL-guided allocator is generally better at preserving actuator rate margin. Although there are intervals where the RL-guided allocator uses more rate than the standard QP allocator, this is most likely because it is actively repositioning effectors to create better reserve for the following steps. The RL agent unloads highly loaded effectors over time to restore both position and rate margin. Since the torque demand persists for a long duration, these actions to rebuild reserve pay off in the later stages.

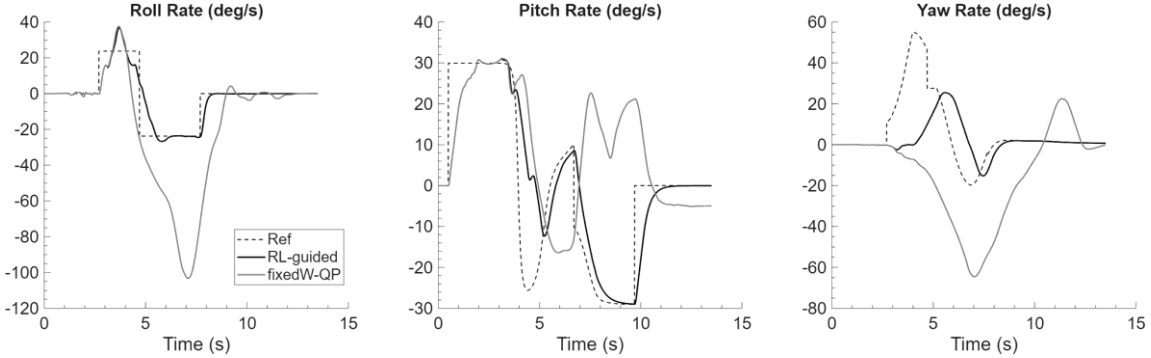

Figure 5: Angular rate command tracking (extreme flight conditions)

Figure 5 shows the angular rate command tracking in two simulations. The RL agent distributes the desired moment across the effectors more effectively and unloads them as they approach their limits. This is done by selecting corresponding weights by the agent and this behavior helps the controller to maintain better tracking over longer time horizons. Although the RL-guided allocator tends to use higher rates in several intervals of simulation, this translates into improved slack behavior over time, which in turn yields better tracking performance in the inner-loop control laws (CLAWs).

The effector positions and the weights of W_r matrix in the RL-guided allocator are given in Figure 6, Figure 7, and Figure 8.

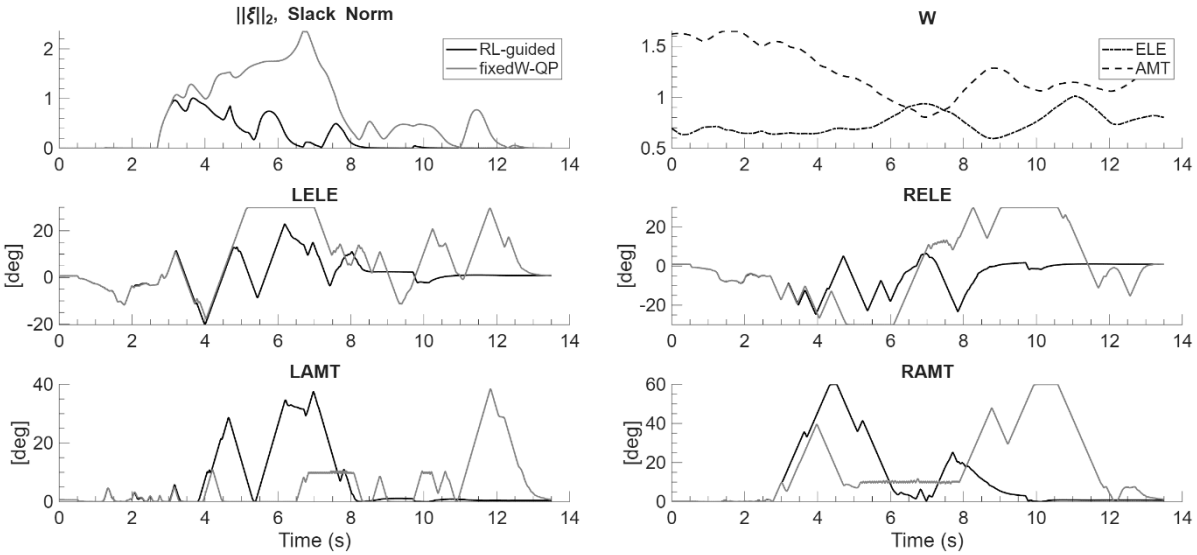

Figure 6: Effector positions: ELE, AMT (extreme flight conditions)

Figure 6 shows the left and right ELE and AMT positions for both allocators. The corresponding elements of the W_r matrix for the same episode are also given. At the beginning of the episode, where the pitch rate command is dominant, the AMTs are weighted as more expensive than the ELEs. However, around 4–7 seconds, the agent gradually shifts the load towards the AMTs by making the ELEs more expensive. This change in weighting causes the RL-guided allocator to use the AMTs more than the standard QP allocator. As a result, the ELEs maintain more position reserve and spend less time at saturation when the desired torque is beyond the attainable set.

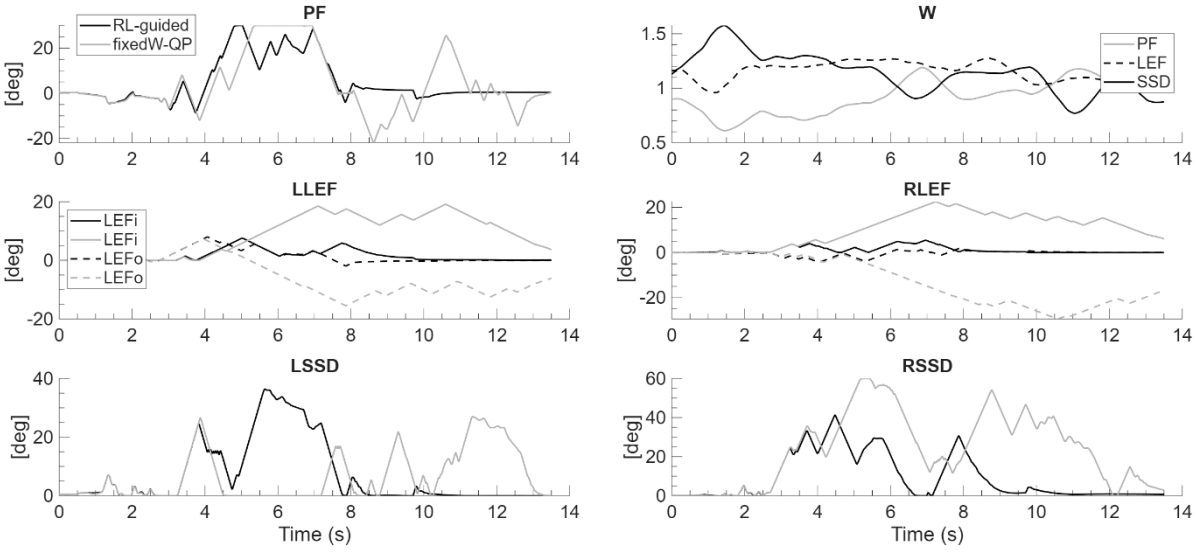

Figure 7: Effector positions: PF, LEF, SSD (extreme flight conditions)

In Figure 7, effector positions and weights are given. It has been shown that RL agent does a similar tradeoff at these effectors as well. The main difference to the standard QP allocation is that using SSDs when PF is saturated. Also, a general penalty is applied by the RL agent to the LEF effectors to avoid using them for high roll and pitch demand cases. It could be most likely that the agent has discovered the cross effects of LEFs over other surfaces during training.

To understand better how the posture of the effectors changes the local control effectiveness, the maximum attainable moment set in three dimensions is given in Figure 8. The maximum attainable moment set is calculated via Durham's algorithm over the local control effectiveness matrix at simulation time 7s. The current effector positions are considered when calculating maximum allowed deflections on either side. All the effector positions of the RL-guided and QP allocators are given again in the same figure to understand better where they are positioned. Also, the angular rate values given as subplots to represent the aircraft state. The dark volume in Figure 8 shows the maximum attainable moment for the B_{loc} matrix from the simulation with the standard QP agent, while the lighter one shows the same from the simulation with RL-guided allocator.

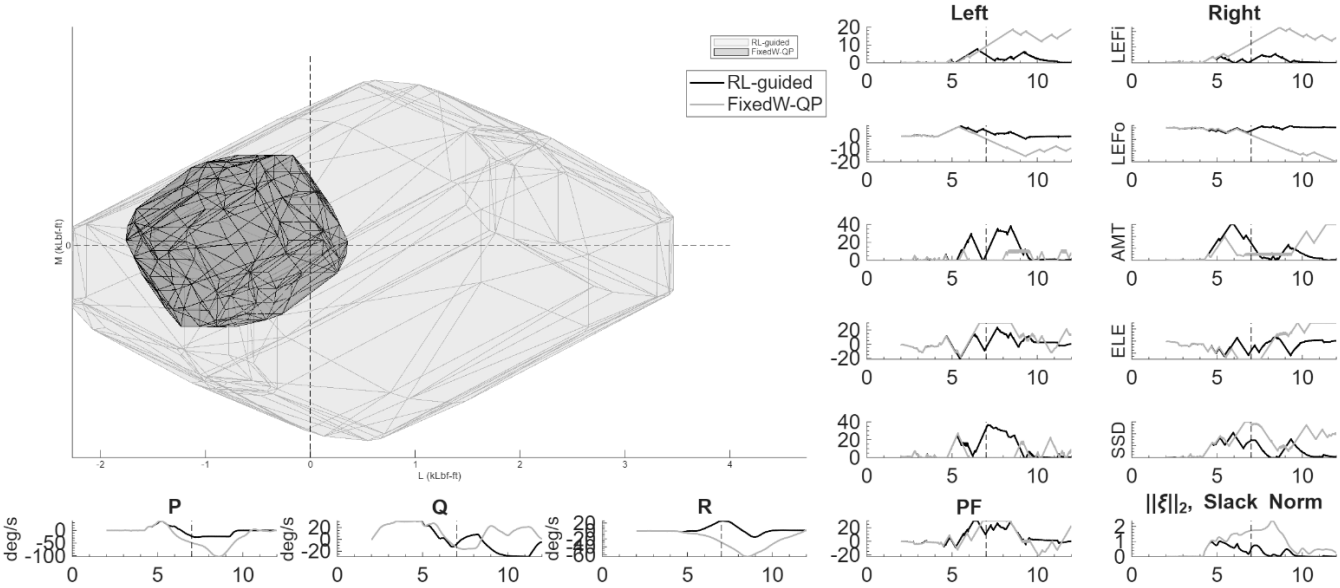

Figure 8: Maximum attainable moment set difference at t=7s (extreme flight conditions)

It has been realized that effector posture has a significant impact on maximum attainable moments in all axes. And preserving limits yields much better roll and pitch moment potentials while providing approximately similar moments from the beginning of the episode.

While the RL agent was trained using aggressive, randomized angular rate doublets designed to push the allocator into physically infeasible regions to learn slack reduction, low-demanding angular rate commands have also been injected into the training scenarios. Additionally, due to the proposed safe-manipulating control allocation architecture, it is not expected that the RL agent will break the smoothness of the allocated commands in nominal flight conditions. To verify this, the trained hybrid allocator was subjected to smooth, continuous guidance profiles representative of standard autopilot waypoint navigation.

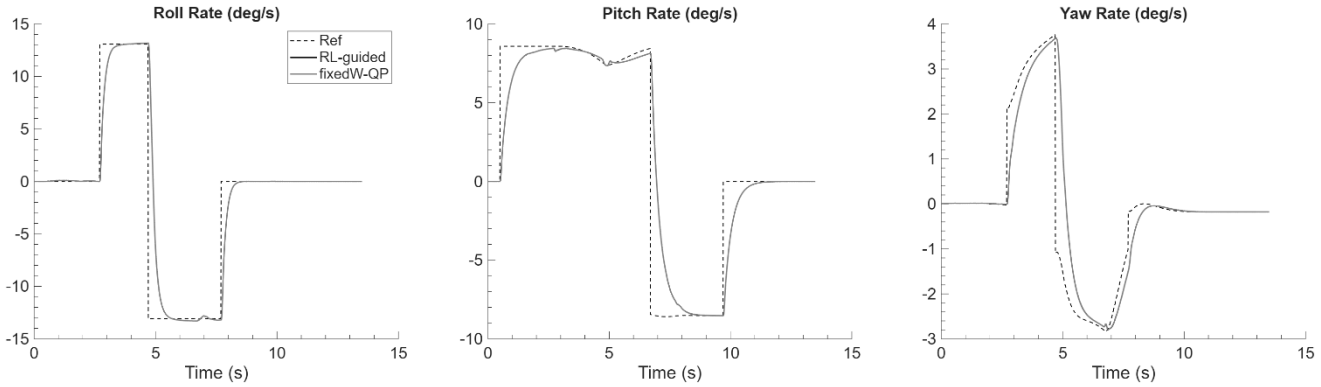

Figure 9: Angular rate command tracking (normal flight conditions)

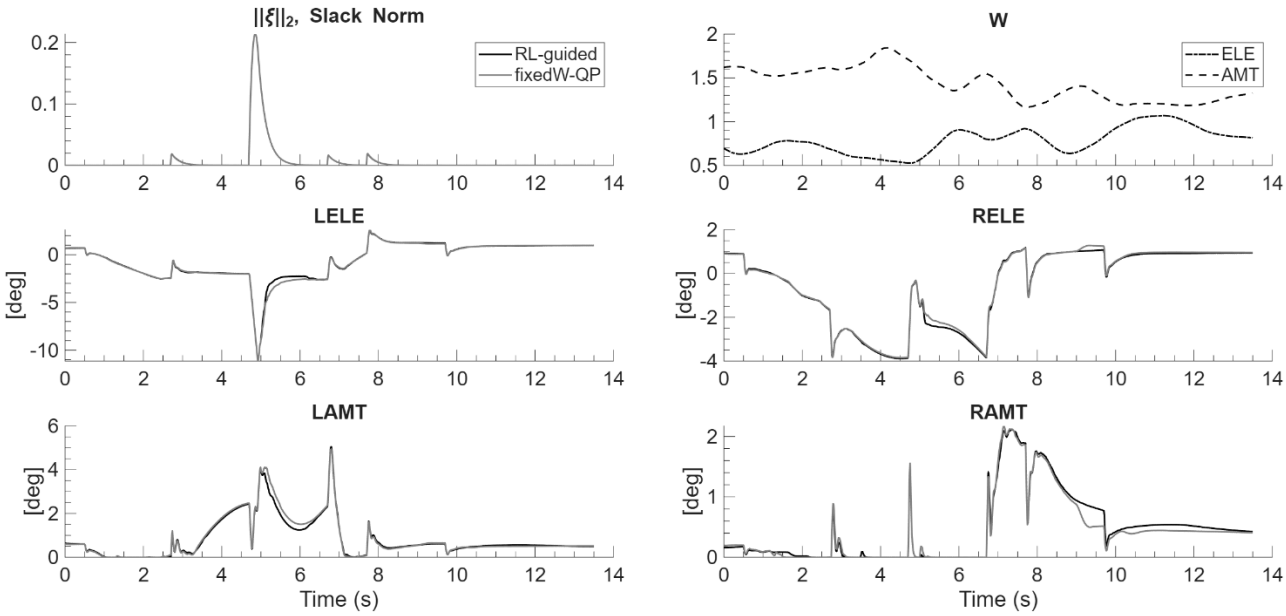

Figure 10: Effector positions: ELE, AMT (normal flight conditions)

The angular rate tracking of Figure 9 and effector positions (only ELE and AMT positions are given for the sake of simplicity, others effectors show similar trends) in Figure 10 confirm that when feasible moment demands are present, the RL-guided allocator maintains perfect inner-loop tracking without introducing allocation error or high-frequency effector chattering. Because the reward function shifts its focus to minimizing absolute effector duty (r_{duty}) when slack is absent, the agent refrains from making aggressive weight changes. This demonstrates that training for worst-case, slack-heavy recovery does not compromise the stability, smoothness, or performance of the aircraft during nominal flight conditions.

A deeper analysis of the allocation logs reveals the complex aerodynamic trade-offs made to achieve these improvements. The RL guidance layer demonstrates an acquired understanding of the cross-coupling effects inherent in the local control effectiveness matrix (B_{loc}). For instance, the policy applies higher penalty weights to the All-Moving Tips (AMTs) and Spoilers (SSDs), as these surfaces generate significant adverse yaw and asymmetric drag. By shifting the baseline workload to aerodynamically 'cleaner' surfaces like elevons and pitch flaps, the framework preserves the rate headroom of the powerful AMTs for critical moments and reduces the total physical effort caused by surfaces fighting each other's parasitic moments. However, relying on this rigidly learned hierarchy may introduce specific vulnerabilities. In rare edge cases where torque demand is aggressive (e.g. simultaneous multi-axis

maneuvers such as concurrent hard pitch and roll), the heavy reliance on these primary surfaces can cause them to reach saturation prematurely. The hesitation to utilize the heavily penalized, cross-coupled effectors in these exact moments can occasionally lead to temporary tracking delays or slight performance degradation, highlighting a fundamental trade-off.

6 Conclusions

A reinforcement learning based hybrid control allocation approach is proposed in this study. A safety-critical baseline quadratic programming (QP) allocator is guided by a higher-level RL layer that adjusts allocation weights dynamically over time. The key idea is that, although a convex QP allocator is already optimal for its defined cost and constraints, it does not reason for longer time horizons. The RL guidance layer addresses that gap by slowly adapting the allocator's weighting and posture logic. Unlike heuristic weight-switching methods that introduce discrete penalties, the continuous RL guidance proactively shifts the workload away from easily saturated or highly cross-coupled effectors. It can reduce allocation error cumulatively, preserve actuator rate and position margins for critical maneuvers, and finally improve tracking performance over time.

As the design is hybrid, safety can be preserved. The RL guidance layer does not inject additional commanded moments; instead, it operates in directions that lie in the null space of the local control effectiveness matrix. This means the framework can redistribute the effort across effectors and manage actuator usage without violating the desired torque. Finally, the RL guidance slowly influences how the moment is shared, not what moment is commanded through the effector commands in the fast allocation, which may open the doors for safe integration of a learning-based element into a flight-critical allocation loop.

This study serves as a foundational step in the research of RL-guided control allocation. By utilizing an RL-guided architecture, long-horizon guidance is introduced to classical control allocation methods, significantly enhancing the allocation performance and potentially the overall control allocation robustness. In this regard, the findings are highly promising and warrant deeper investigation.

Future work will focus on addressing the limitations of rigidly learned effector hierarchies, which can occasionally lead to premature saturation under simultaneous multi-axis demands, and expanding the framework to dynamically relax these policies across a broader flight envelope.

Declaration of Use of Artificial Intelligence

In this study, ChatGPT (GPT-5) was used only for early-stage brainstorming of developing/assessing design options; GitHub Copilot was used slightly for basic Python syntax while developing the Gymnasium environment and setting up Python training environment. No AI/chatbot/writing bot was used in writing manuscript, or to generate figures/plots. All code and results are authored, reviewed, and validated by the authors; no proprietary data was shared with AI tools.

References

[1] S. Sieberling, Q. P. Chu, and J. A. Mulder, "Robust flight control using incremental nonlinear dynamic inversion and angular acceleration prediction," *Journal of Guidance, Control, and Dynamics*, vol. 33, no. 6, pp. 1732–1742, 2010, doi: 10.2514/1.49978.

[2] I. Matamoros and C. C. de Vissery, "Incremental nonlinear control allocation for a tailless aircraft with innovative control effectors," *AIAA Guidance, Navigation, and Control Conference, 2018*, no. 210039, pp. 1–25, 2018, doi: 10.2514/6.2018-1116.

[3] K. M. Dorsett and D. R. Mehl, "INNOVATIVE CONTROL EFFECTORS (ICE)," Jan. 1996.

[4] M. A. Niestroy, K. M. Dorsett, and Markstein, "A Tailless Fighter Aircraft Model for Control-Related Research and Development," 2016.

[5] J. M. Buffington, "Modular Control Law Design for the Innovative Control Effectors (ICE) Tailless Fighter Aircraft Configuration 101-3," 1999.

[6] I. Matamoros, "Nonlinear Control Allocation for a High-Performance Tailless Aircraft with Innovative Control Effectors An Incremental Robust Approach," Delft University of Technology, 2017.

[7] R. Stolk and C. De Visser, "Minimum drag control allocation for the Innovative Control Effector aircraft."

[8] O. Härkegård, "Dynamic Control Allocation Using Constrained Quadratic Programming," *Journal of Guidance, Control, and Dynamics*, vol. 27, no. 6, pp. 1028–1034, Nov. 2004, doi: 10.2514/1.11607.

[9] L. Zaccarian, "Dynamic allocation for input-redundant control systems *," 2008.

[10] W. Durham, K. A. Bordignon, and R. Beck, "Aircraft control allocation," p. 281.

[11] P. Kolaric, V. G. Lopez, and F. L. Lewis, "Optimal Dynamic Control Allocation with Guaranteed Constraints and Online Reinforcement Learning," 2020. [Online]. Available: https://www.elsevier.com/open-access/userlicense/1.0/

[12] P. S. de Vries and E. J. van Kampen, "Reinforcement learning-based control allocation for the innovative control effectors aircraft," in *AIAA Scitech 2019 Forum*, American Institute of Aeronautics and Astronautics Inc, AIAA, 2019. doi: 10.2514/6.2019-0144.

[13] K. C. Wu and J. S. Litt, "Reinforcement Learning Approach to Flight Control Allocation With Distributed Electric Propulsion," 2023. [Online]. Available: http://www.sti.nasa.gov

[14] K. Dally and E.-J. van Kampen, "Soft Actor-Critic Deep Reinforcement Learning for Fault Tolerant Flight Control," Feb. 2022, doi: 10.2514/6.2022-2078.

[15] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *ArXiv*, pp. 1–12, 2017.

[16] M. Towers *et al.*, "Gymnasium: A Standard Interface for Reinforcement Learning Environments," 2024. [Online]. Available: https://arxiv.org/abs/2407.17032

[17] A. Raffin, “RL Baselines3 Zoo,” 2020, *GitHub*. Accessed: Sep. 30, 2025. [Online]. Available: https://github.com/DLR-RM/rl-baselines3-zoo

[18] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, “Stable-Baselines3: Reliable Reinforcement Learning Implementations,” *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021, [Online]. Available: http://jmlr.org/papers/v22/20-1364.html