NASA/TM-20230014863

# Reinforcement Learning Approach to Flight Control Allocation With Distributed Electric Propulsion

*Kristin C. Wu and Jonathan S. Litt*
*Glenn Research Center, Cleveland, Ohio*

November 2023

# NASA STI Program . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) Program plays a key part in helping NASA maintain this important role.

The NASA STI Program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI Program provides access to the NASA Technical Report Server—Registered (NTRS Reg) and NASA Technical Report Server—Public (NTRS) thus providing one of the largest collections of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counter-part of peer-reviewed formal professional papers, but has less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., "quick-release" reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

For more information about the NASA STI program, see the following:

- Access the NASA STI program home page at http://www.sti.nasa.gov

- E-mail your question to help@sti.nasa.gov

- Fax your question to the NASA STI Information Desk at 757-864-6500

- Telephone the NASA STI Information Desk at 757-864-9658

- Write to:
  NASA STI Program
  Mail Stop 148
  NASA Langley Research Center
  Hampton, VA 23681-2199

NASA/TM-20230014863

# Reinforcement Learning Approach to Flight Control Allocation With Distributed Electric Propulsion

*Kristin C. Wu and Jonathan S. Litt*
*Glenn Research Center, Cleveland, Ohio*

National Aeronautics and
Space Administration

Glenn Research Center
Cleveland, Ohio 44135

November 2023

## Acknowledgments

This work was funded by the Convergent Aeronautics Solutions project, under the Transformative Aeronautics Concepts Program within the NASA Aeronautics Research Mission Directorate. This work could not have been done without the help of various people. The authors would like to thank Shane Sowers for all his help with the software of the SUSAN model, Jonah Sachs-Wetstone and Halle Buescher for answering questions about the aircraft system, and Felipe Valdez and the rest of the SUSAN flight controls team for sharing their expertise.

This work was sponsored by the
Transformative Aeronautics Concepts Program.



*Level of Review*: This material has been technically reviewed by technical management.

This report is available in electronic form at https://www.sti.nasa.gov/ and https://ntrs.nasa.gov/

# Reinforcement Learning Approach to Flight Control Allocation With Distributed Electric Propulsion

Kristin C. Wu* and Jonathan S. Litt
National Aeronautics and Space Administration
Glenn Research Center
Cleveland, Ohio 44135

## Abstract

The flight control system of the SUSAN Electrofan concept aircraft achieves attitude control using both conventional flight control surfaces and differential thrust through distributed electric propulsion (DEP) from sixteen wing-mounted electric engines. The introduction of eight pairs of wing fans for attitude control creates a highly actuated system. Such a system requires more sophisticated control to operate, especially in the presence of wingfan failures where the loss of a single wingfan can result in a thrust imbalance. This paper investigates the use of deep reinforcement learning (RL) using proximal policy optimization (PPO) to achieve attitude control through a combination of DEP and control surface deflections. First, the paper examines the aircraft undergoing a coordinated turn. Then, it examines the aircraft experiencing a wingfan failure during cruise conditions. It is shown that deep reinforcement learning can be a potential avenue for nonlinear flight control design.

## Nomenclature

### Aerospace Nomenclature

| | |
|---|---|
| SUSAN | SUbsonic Single Aft eNgine |
| DEP | Distributed Electric Propulsion |
| PID | Proportional-Integral-Derivative |
| cmd | command |
| $\delta$ | delta |
| T | thrust [*lbf*] |
| dT | differential Thrust [*lbf*] |
| $(\phi, \theta, \psi)$ | roll, pitch, and yaw, respectively [*deg*] |
| $(\dot{\phi}, \dot{\theta}, \dot{\psi})$ | roll rate, pitch rate, and yaw rate, respectively [*deg/s*] |
| $(\ddot{\phi}, \ddot{\theta}, \ddot{\psi})$ | roll acceleration, pitch acceleration, and yaw acceleration, respectively [$deg/s^2$] |
| $\beta$ | sideslip [*deg*] |

### Reinforcement Learning Nomenclature

| | |
|---|---|
| RL | Reinforcement Learning |
| PPO | Proximal Policy Optimization |
| SAC | Soft Actor Critic |
| TD3 | Twin Delayed Deep Deterministic Policy Gradient |
| t | time [*s*] |

*Spring 2023 OSTEM internship program at Glenn Research Center.

| | |
|---|---|
| $a \in A$ | action vector ∈ action space |
| $s \in S$ | state vector ∈ state space |
| $r$ | reward |
| $\gamma$ | discount factor |
| $\pi$ | policy function |
| $\theta$ | parameters of policy function |
| $L$ | clipped surrogate objective function |

# 1.0 Introduction

The Subsonic Single Aft eNgine (SUSAN) Electrofan is a NASA concept aircraft consisting of a single gas turbine engine and sixteen wing-mounted electric engines (or wingfans) for distributed electric propulsion (DEP) as shown in Figure 1. The wingfans are driven by power extracted from the gas turbine engine. The flight control system consists of a three-loop baseline controller with cascaded Proportional-Integral-Derivative (PID) feedback controllers that generate desired body frame rotational rate targets. These targets are then passed to the control allocation subsystem, which converts them into desired control surface deflections (Ref. 1). The introduction of eight pairs of wing fans for attitude control creates a highly actuated system, which requires more sophisticated control to operate. This is especially true in the presence of a wingfan failure, where the loss of a single wingfan can result in a thrust imbalance if no thrust reallocation system is in place.

To address this challenge, this paper investigates the use of machine learning techniques to enhance existing control allocation schemes. This report focuses on two separate scenarios where distributed electric propulsion plays an important role: (1) a coordinated turn; and (2) straight and level flight in the presence of a wingfan failure. Machine learning offers the potential for automatic discovery of patterns from data, avoiding the need for manual system design and expert knowledge. Specifically, the paper explores the use of reinforcement learning (RL), which works by determining the best output for some input based on the response of the environment to that output. Reinforcement learning differs from supervised learning in that it can go beyond expert data to determine the optimal behavior, but its sample efficiency is low, meaning that it requires a significant amount of data before becoming proficient.

The current implementation of control allocation for the SUSAN aircraft developed by Ogden and Patterson relies on a pseudoinverse weighted dynamically based on the aircraft's distance from state and control effector limits (Ref. 1). Previous work on reinforcement learning for attitude control applied to aircraft has yielded promising results. Bohn et al. (Ref. 2) and Koch et al. (Ref. 3) demonstrated a reinforcement-learning-based controller for an unmanned aerial vehicle that can outperform a PID controller, particularly in unpredictable environments. De Vries and Van Kampen also achieved successful control of an aircraft model's altitude using reinforcement learning for a highly actuated system (Ref. 4).


Figure 1.—Artist rendering of SUSAN.

The remainder of the paper is structured as follows. Section 2.0 provides an overview of the SUSAN flight control system and the reinforcement learning algorithm used. Section 3.0 outlines the configuration of the RL controller and the methodology. Section 4.0 presents an analysis and comparison of results to existing implementations. Finally, Section 5.0 concludes the paper and highlights potential avenues for future research.

# 2.0 Background

## 2.1 Flight Control

The flight control system consists of a multiloop architecture. To achieve a desired altitude, heading, and speed, the autopilots control position and course while the outer loops control attitude to produce desired attitude rates. Additionally, there is an inner loop that controls the aircraft's velocities, but that step is ignored in this RL implementation. Control allocation is the final step of the flight control system, displayed in Figure 2, which receives attitude rate commands and converts them into effector commands.

The flight control surfaces consist of $\delta_{aileron}$, $\delta_{elevator}$, and $\delta_{rudder}$, which represent commanded angle displacements from the trim condition of the aileron, elevator, and rudder, respectively. $\delta_T$ represents a change in commanded total thrust from the trim condition, a percentage of which is allocated to the gas turbine engine based on the power extraction ratio while the remaining $\delta_T$ is split evenly between the sixteen electric engines. $dT_1 \ldots dT_8$ represent differential thrust commands for the eight pairs of wing fans. These differential thrust commands are symmetric for each pair in order to maintain the same power extraction from the gas turbine engine.

For the most part, ailerons affect roll, elevators affect pitch, and the rudder and wingfans affect yaw, but the reality is a more complex and highly coupled system. The two cases examined in this paper demonstrate situations where an adverse yaw, i.e., yaw in the direction opposite of the turn, would be generated. During a coordinated turn, when the ailerons are used to roll into the turn, an adverse yaw is generated, which must be counteracted by the rudder and distributed electric propulsors. Similarly, if a wingfan ceases to produce thrust, then the result is adverse yaw generated by asymmetrical thrust from the electric wingfans.

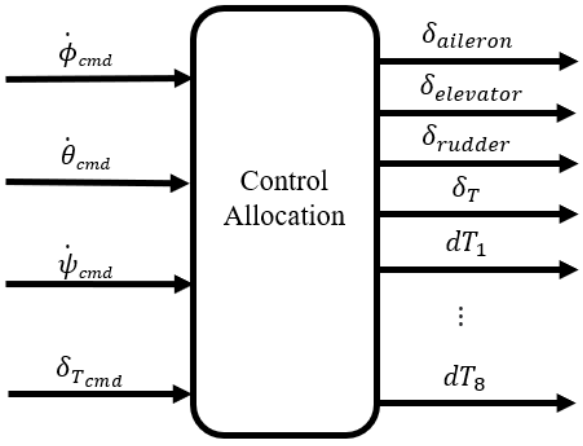


Figure 2.—Control allocation block diagram. This block takes in attitude rates and outputs effector setpoints.

## 2.2 Reinforcement Learning

This work considers a standard reinforcement learning setup consisting of an agent interacting with an environment in discrete timesteps. At each timestep, $t$, the agent receives a state $s \in S$ (i.e., the observation vector), takes an action $a \in A$ (i.e., the effector setpoints vector), and receives a scalar reward $r$ defined by a reward function $r(s, a)$. Upon taking action $a$, a new state $s'$ is returned and the cycle is repeated.

An agent's behavior is defined by a policy $\pi: S \rightarrow P(A)$, which maps states to a probability distribution over the actions. In other words, the policy $\pi$ takes the state $s$ as its input and outputs an action $a$, or more specifically in this case, the means and variances of a multivariate Gaussian from which actions are drawn. During training, actions are randomly sampled from this distribution to increase exploration, while the mean is taken as the action when training is completed (Ref. 2). In this implementation, the policy $\pi$ is defined by a neural network with parameters $\theta$, which represent the weights and biases of units in the neural network. Neural networks are universal function approximators and can be used to approximate the policy function $\pi$.

The goal in reinforcement learning is to create a policy that maximizes the total sum of future discounted rewards, also known as the return, defined as $\sum_{k=0}^{\infty} \gamma^k r_{t+k}$ for discount factor $\gamma$. Policy gradient algorithms work by updating the parameters $\theta$ of the policy $\pi$ in the direction of gradient ascent in order to maximize this value. This is the general idea behind policy gradient algorithms, but many variations exist that seek to improve efficiency and stability. The structure of most state-of-the-art reinforcement learning algorithms for continuous control is displayed in Figure 3.

There are several reinforcement learning algorithms that would be applicable to this problem such as Soft Actor-Critic (SAC) (Ref. 5) and Twin Delayed Deep Deterministic Policy Gradient (TD3) (Ref. 6). For this implementation, we use Proximal Policy Optimization (PPO), which is a state-of-the-art policy gradient actor-critic RL algorithm developed by OpenAI (Ref. 7). PPO was chosen due to its previous success with attitude control in aircraft (Refs. 2 and 3) and its faster performance for environments where sampling takes little time. The motivation behind using PPO is to avoid making too large policy updates in order to improve training stability. This is done by constraining the policy update to a small range:

$$L(\theta) = E[\, min(\, R(\theta) * A_t, clip(R(\theta), 1 - \epsilon, 1 + \epsilon) * A_t \,)\,]$$

where $E$ denotes the expectation and $\epsilon$ is some scalar clip range.[1] $A_t$ is the advantage (i.e., improvement) from taking action $a$ at state $s$ and uses the value function $V(\boldsymbol{s})$ in its calculation. $R(\theta) = \pi(a|s)/\pi_{old}(a|s)$ defines how likely action $a$ at state $s$ is in the current policy over the old policy. In summary, by taking a gradient ascent step on $L(\theta)$, known as the clipped surrogate objective function, the agent is pushed to take actions that lead to higher rewards and avoid harmful actions. The algorithm for PPO is outlined in Algorithm 1.

**Algorithm 1: PPO**

**for** iteration = 1, 2, … **do**
  Run policy $\pi_{old}$ in environment for T time steps
  Compute advantage estimates $A_t$ for t = 1, 2, …, T
  Optimize $L$ with respect to $\theta$
  $\theta_{old} \leftarrow \theta$
**end**

[1] *Clip* is a python function that limits values to an interval; values outside the interval are clipped to the interval limits.

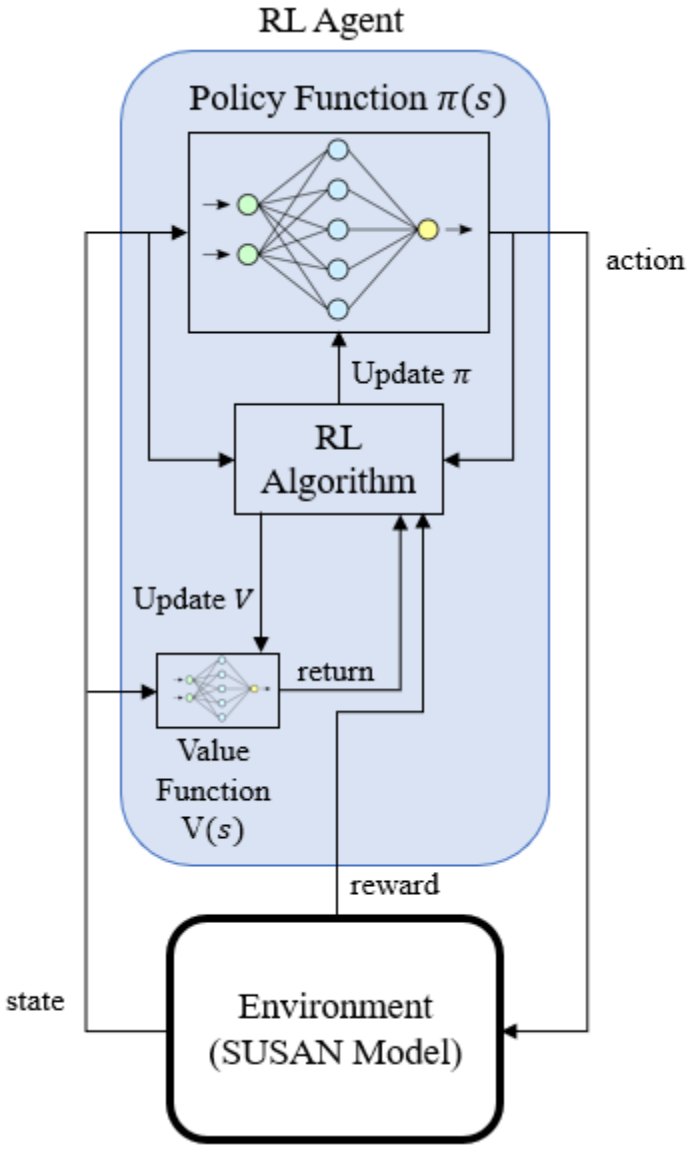


Figure 3.—General configuration of reinforcement learning algorithms. The Policy Function is a neural network that maps states to a probability distribution over the actions. The Reinforcement Learning (RL) Algorithm updates the policy based on the result of the action on the environment. The value function $\boldsymbol{V}: \boldsymbol{S} \rightarrow \mathbb{R}$ is a separate neural network that is trained concurrently and outputs the estimated return from a state $\boldsymbol{s}$.

## 3.0 Method

The environment is a multilayered architecture composed of a Simulink® aircraft model, a communication layer, and an agent-environment interface layer, as displayed in Figure 4. At the core of the environment is a full nonlinear model of the version of the SUSAN aircraft described in Reference 1, built in Simulink®. In order to facilitate the training of the RL controller, the Simulink® model is compiled into a standalone executable, speeding up execution time by ~20 times. Next, the communication layer is a TCP/IP interface positioned between the Simulink® model and the Python agent-environment layer. This channel uses socket communication to handle low-level data transfer between the two layers. Finally, the agent-environment interface is created in Python using the OpenAI Gym environment. This layer facilitates tasks such as data pre-processing and reward calculation for communication to the RL agent.

The PPO algorithm is implemented using Stable Baselines3 (Ref. 8), a set of reinforcement learning algorithm implementations in PyTorch. In order to minimize the effect of initial transients, the environment is run for 215 s in flight-time before training to allow the model to trim. The controller is trained in fixed-length episodes at 50 Hertz on an Intel® Core™ i7-10850H CPU and NVIDIA® Quadro T1000 GPU. Although the task at hand is not truly episodic in the sense of having natural terminal states, it allows for greater control of the agent's exploration of the state space and typically generalizes well.

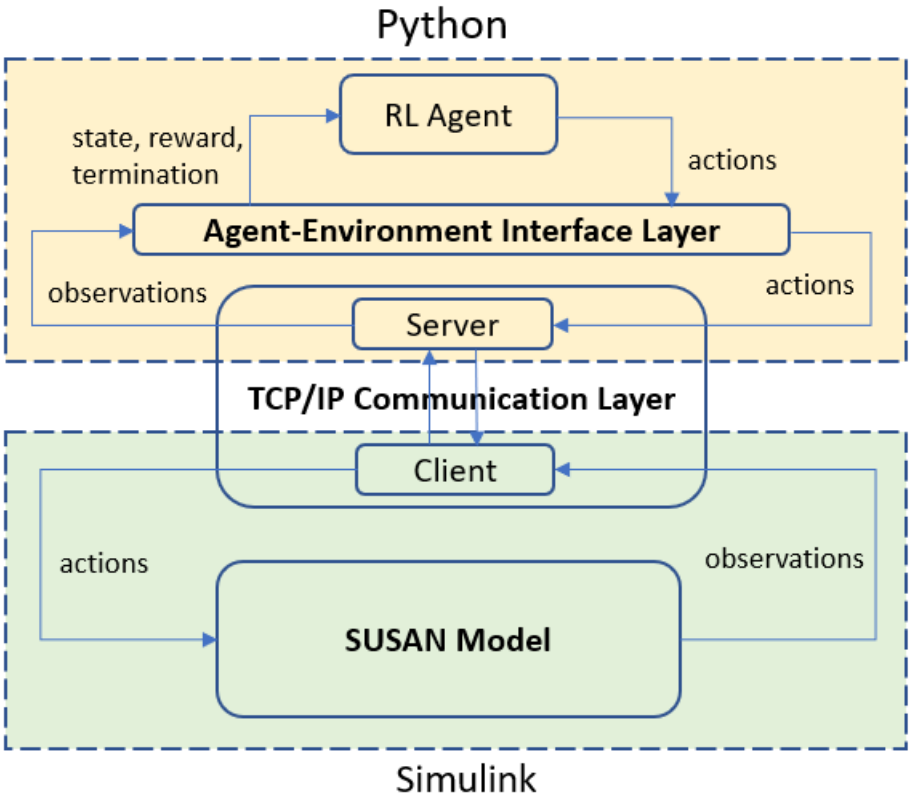


Figure 4.—Environment architecture consisting of the Simulink® aircraft model, a TCP/IP communication layer, and an agent-environment interface layer.

Neural networks are known to converge faster when input features share a common scale because the network does not need to learn the scaling values. The state vector is standardized beforehand in such a way that the mean corresponds to trim conditions. Similarly, the action vector is normalized to be between –1 and 1 based on effector limits.

The flight condition is cruise at 35,000 ft altitude and 0.785 Mach. The 100 percent vertical tail configuration, i.e., baseline, was used (Ref. 1). (Note that the full nonlinear SUSAN aircraft model used in this work was developed to enable trade studies that evaluate reducing the size of the vertical tail, thus reducing drag and therefore fuel consumption.)

## 3.1 Case 1: Coordinated Turn

For the purposes of ignoring the lateral axis, assume that the elevator and throttle are decoupled from the rudder, ailerons, and distributed electric propulsion system. Thus, the actuator setpoints and action space are reduced to $\delta_{aileron}$, $\delta_{rudder}$, and $dT_1 \ldots dT_8$. The state space consists of sideslip ($\beta$) in addition to roll and yaw angles ($\phi$, $\psi$), roll and yaw rates ($\dot{\phi}$, $\dot{\psi}$), roll and yaw accelerations ($\ddot{\phi}$, $\ddot{\psi}$), and roll and yaw rate commands ($\dot{\phi}_{cmd}$, $\dot{\psi}_{cmd}$).

In accordance with traditional control theory, where typically costs are to be minimized rather than rewards are to be maximized, the total reward returned is a negative value normalized to be in the range of –1 to 0:

$$
\begin{aligned}
r_{\dot{\phi}} &= \text{clip}\left(\zeta_{\dot{\phi}}\left|\dot{\phi} - \dot{\phi}_{cmd}\right|, 0, \zeta_{\dot{\phi}}\right) \\
r_{\dot{\psi}} &= \text{clip}\left(\zeta_{\dot{\psi}}\left|\dot{\psi} - \dot{\psi}_{cmd}\right|, 0, \zeta_{\dot{\psi}}\right) \\
r_{aileron} &= \text{clip}\left(\zeta_{aileron}\left|\delta_{\text{aileron}}\right|, 0, \zeta_{aileron}\right) \\
r_{rudder} &= \text{clip}\left(\zeta_{rudder}\left|\delta_{rudder}\right|, 0, \zeta_{rudder}\right) \\
r_{dT} &= \text{clip}\left(\frac{\zeta_{dT}}{8}\sum_{i=1}^{8}\left|dT_i\right|, 0, \zeta_{dT}\right)
\end{aligned}
$$

$r_{\dot{\phi}}$ and $r_{\dot{\psi}}$ represent costs associated with errors between the commanded and measured roll and yaw rates, respectively. $r_{aileron}$, $r_{rudder}$, and $r_{dT}$ are costs to minimize effector usage for the ailerons, rudder, and wingfans, respectively. Compared to other implementations of RL aircraft attitude control (Ref. 2), the error costs are weighted relatively less since the goal of this case is to minimize effector usage and not necessarily overshoot or settling time.

When the roll $\phi$ becomes too large, the aircraft enters an unhealthy state and the episode terminates with a penalty incurred, denoted by $r_{unhealthy}$. Since the total reward is nonpositive, the penalty chosen is large enough such that the agent doesn't choose to terminate early in favor of avoiding future costs.

$$
\begin{aligned}
r_{unhealthy} &= 1000 \; if \; \phi \geq 45° \text{ else } 0 \\
r &= -\left(r_{\dot{\phi}} + r_{\dot{\psi}} + r_{aileron} + r_{rudder} + r_{dT} + r_{unhealthy}\right)
\end{aligned}
$$

The controller was trained in episodes of 6250 time steps, corresponding to 125 s of flight time, for 10 million time steps. For this setup, training took approximately 8 hr to complete. Each episode began in steady-state and was initiated with a commanded turn to a randomly generated target heading.

### 3.2 Case 2: Wingfan Failure

Since a formal failure injection framework has not been implemented in the SUSAN Simulink® model at the time of this report, a wingfan failure is mimicked by outputting a commanded thrust of zero for the failed wingfan. The reallocation of thrust is performed separately from control allocation. This examines the case of a failure of the leftmost wingfan, representing the worst-case scenario for a single wingfan failure.

The action space consists of a weight, $-1 \leq w_i \leq 1$, for each nonfailed wingfan that represents the proportion of the lost thrust that should be redistributed to that wingfan relative to the other weights (i.e., $w_1/\sum_j w_j$). This is to ensure that the total power extraction from the gas turbine engine is maintained. The state space consists of the wingfan thrust, $T$, and the error between the commanded and measured yaw rates, $\dot{\psi}_{cmd}$ and $\dot{\psi}$. The reward consists of this error value, $r_{\dot{\psi}}$, and the change in wingfan usage, $r_T$.

$$
\begin{aligned}
r_{\dot{\psi}} &= \text{clip}\left(\zeta_{\dot{\psi}}(\dot{\psi} - \dot{\psi}_{cmd})^2, 0, \zeta_{\dot{\psi}}\right) \\
r_T &= \text{clip}\left(\frac{\zeta_T}{15}\sum_{i=1}^{15}\left(\frac{w_i}{\sum_j w_j}T\right)^2, 0, \zeta_T\right) \\
r &= -(r_T + r_{\dot{\psi}})
\end{aligned}
$$

The controller was trained in episodes of 4200 time steps, corresponding to 84 s of flight time, for 1 million time steps. Each episode begins in steady-state and is initiated with a failure of the leftmost wingfan.

## 4.0 Results

### 4.1 Case 1: Coordinated Turn

The model does not include the effect of the flight control surfaces on drag. Thus, in lieu of minimizing drag, the goal of this case is to directly reduce the effector usage. A key challenge in reinforcement learning is engineering the reward function. In this case, the penalty of the attitude error compared to the effector usage can make a significant impact. If the attitude error is penalized too much, then there may not be any improvement in effector usage over the current implementation. If the attitude error is penalized too little, then aircraft attitude may deviate too much from the desired attitude.

Figure 5 shows the effector usage during a coordinated turn for a 30° heading change using the original flight control (Ref. 1) and the RL controller. We can see that the rudder and aileron usage has been minimized in the RL implementation. The change in wingfan thrust from trim conditions has also been reduced. The thrust of the wingfan pairs are not clear multiples of one another and instead each act independently from the other wingfan pairs, which may be worth investigating further. Although engine thrust is not directly controllable in this RL implementation, it is displayed here as well.

The lateral, vertical, and longitudinal response during the coordinated turn is shown in Figure 6 and Figure 7 (including measurements for altitude, Mach, roll, and yaw). We can see that there is minimal change in the values, which suggests that an RL controller is able to produce a similar attitude response with less effector usage compared to the current implementation.

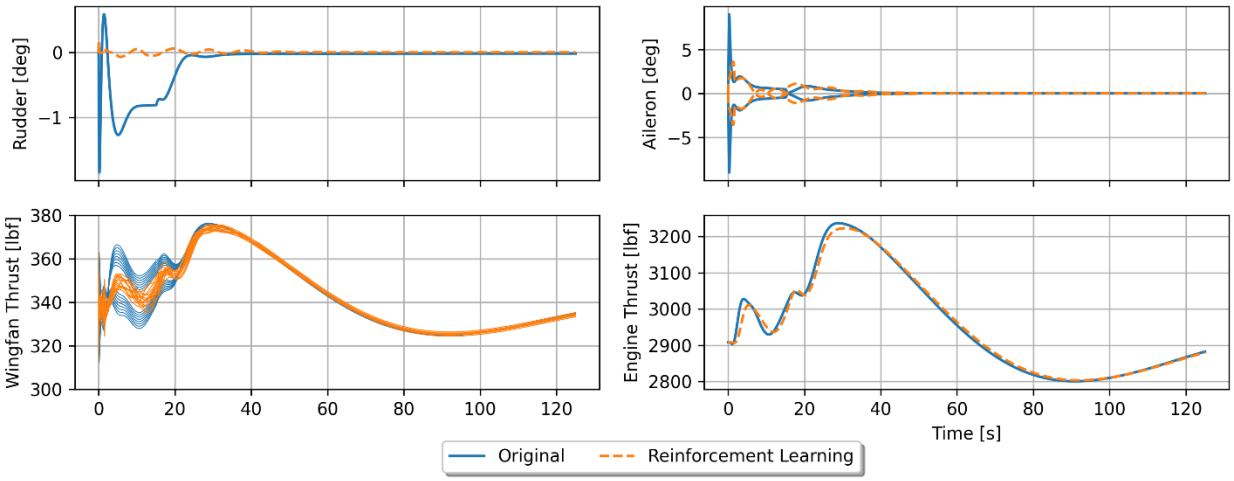


Figure 5.—Effector usage during a coordinated turn of 30°.

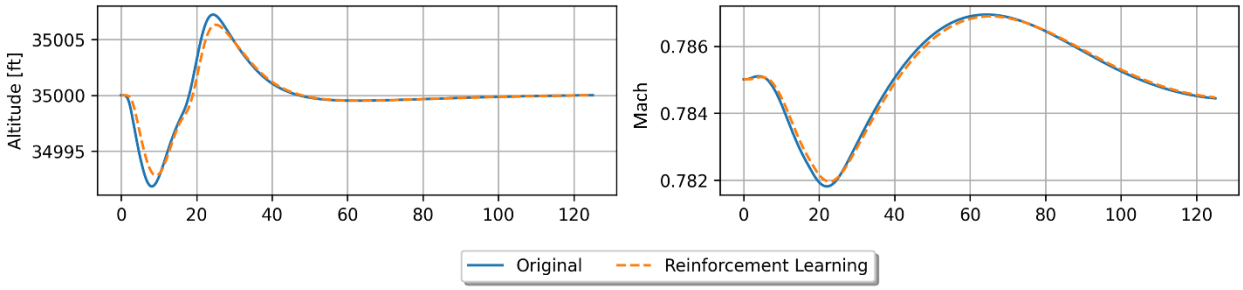


Figure 6.—Altitude and Mach speed response during a coordinated turn of 30°.

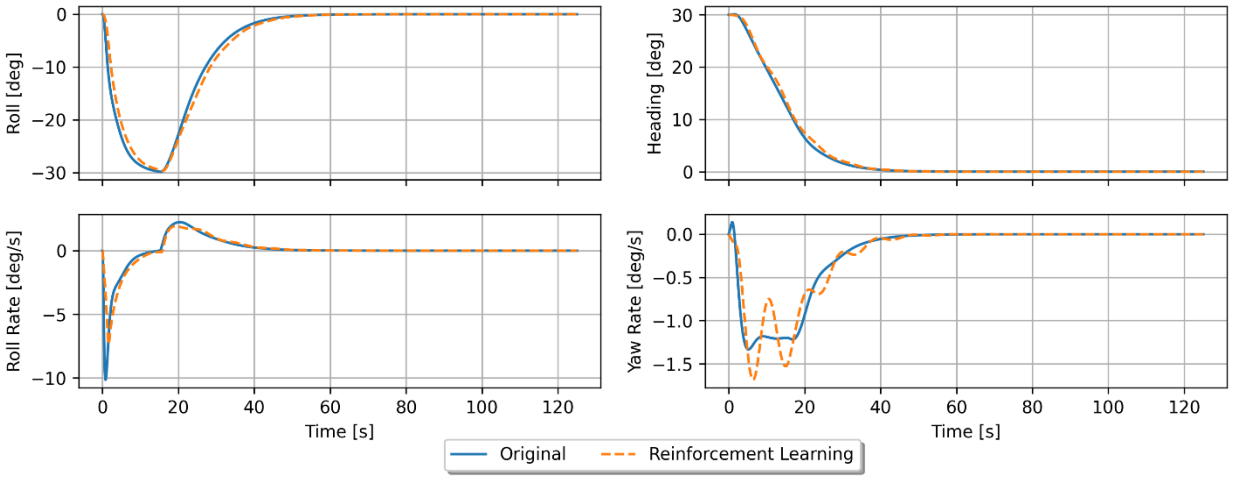


Figure 7.—Vertical and lateral response during a coordinated turn of 30°.

## 4.2 Case 2: Wingfan Failure

A failure of the leftmost wingfan is injected, representing a worst-case scenario for lateral controllability with a single wingfan failure. Since no formal failure injection framework has been developed, a wingfan failure is mimicked by outputting a commanded thrust of zero for the failed wingfan.

Without a thrust reallocation system, the thrust allocated to the failed wingfan is lost and the aircraft must rely on the flight control system for stabilization. Figure 8 displays the vertical and lateral response to the wingfan failure with and without the thrust reallocation system. The results demonstrate that the system converges back toward equilibrium quicker with the reallocation system than without. The time it takes for the roll and yaw rate to settle to zero is faster and the magnitude of these deviations is also smaller. Precise values for the settling time and overshoot can be found in Table I.

The effector usage in response to the wingfan failure is displayed in Figure 9. The plots show that the control surfaces must move significantly more under the original flight control to compensate for the thrust imbalance resulting from the failed wingfan than with thrust reallocation. As expected, without the thrust reallocation system, the gas turbine engine must produce more thrust to maintain cruise conditions.

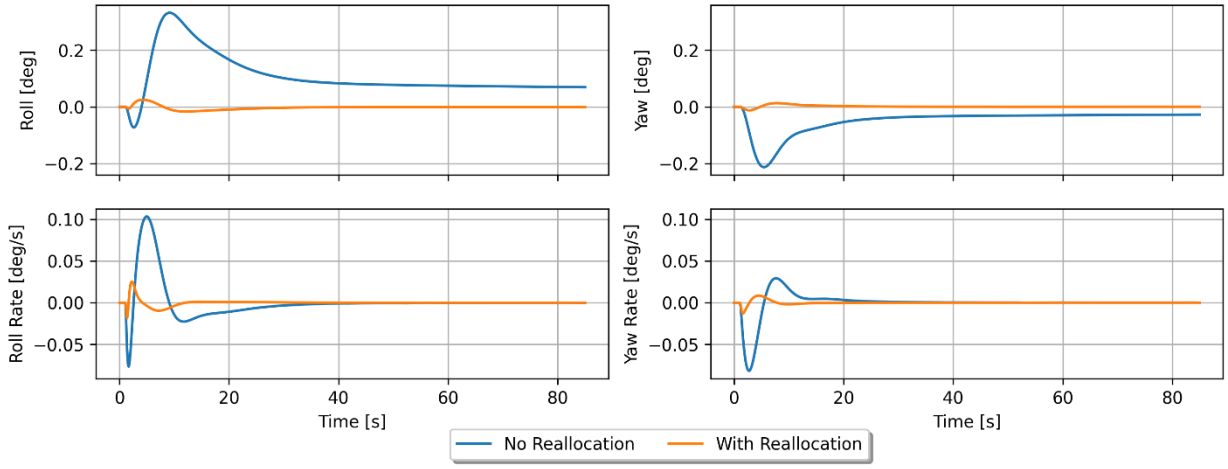


Figure 8.—Vertical and lateral response to a failure of the leftmost wingfan. The original flight control does not account for wingfan failures (No Reallocation). The Reinforcement Learning approach does account for wingfan failures (With Reallocation).

TABLE I.—SETTLING TIME FOR ROLL RATE AND YAW RATE TO REACH 0.01 DEG/S AND OVERSHOOT IN RESPONSE TO A FAILURE OF THE LEFTMOST WINGFAN

| Reallocation | Settling time, *s* | | Overshoot, *deg/s* | |
|---|---|---|---|---|
| | Roll rate | Yaw rate | Roll rate | Yaw rate |
| No Reallocation | 21.0 | 11.1 | 0.103 | 0.0292 |
| With Reallocation | 3.1 | 1.9 | 0.0251 | 0.00852 |

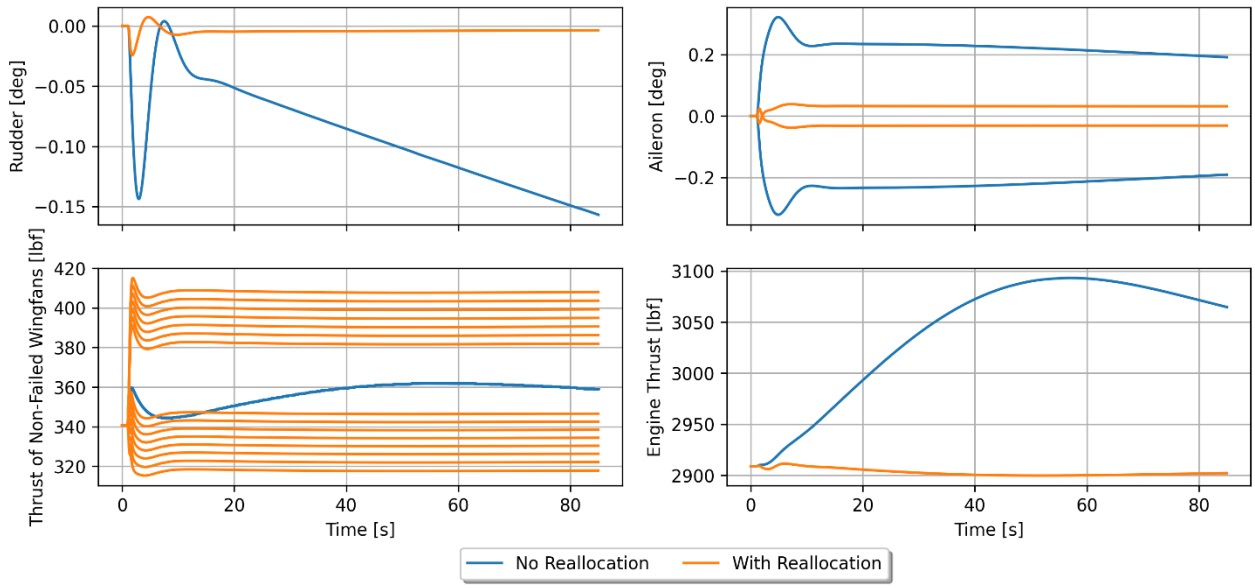


Figure 9.—Effector usage during a failure of the leftmost wingfan. The original flight control does not account for wingfan failures (No Reallocation), so the remaining wingfans continue to move together; the Reinforcement Learning approach does account for wingfan failures (With Reallocation), so the remaining wingfan thrust levels move individually. In both cases, the ailerons move symmetrically in opposite directions, which is inherent in the flight control, however they move much more in the case with the original flight control to counteract the induced roll caused by the thrust asymmetry.

## 5.0 Conclusion

An RL controller for control allocation during a coordinated turn and for thrust reallocation during a wingfan failure was presented in this paper. The results suggest that deep reinforcement learning can be a potential avenue for nonlinear flight control design. Still, there is plenty of future work that can be done. The training methodology can be expanded to increase generality to multiple flight conditions and more effectors can be added to the control allocation, including the throttle, elevators, and individual wingfans, instead of asymmetric thrust pairs. Network structure, the reward function and experiment design can be tuned to facilitate faster and improved learning. Furthermore, at the time of this report, the SUSAN concept and its Simulink® model remain under development, so future iterations of the concept may necessitate updates to the results presented here.

Finally, PPO was applied in this implementation, but other algorithms such as SAC and TD3 should be explored, especially considering how transferable the strategies used in simulations are to real world design. Since SAC and TD3 can learn using offline data and are more sample efficient, these algorithms have the potential to perform better outside of a simulation environment.

## References

1. N.C. Ogden and A. Patterson, “A Framework for Evaluating Differential Electric Propulsion on the SUSAN Electrofan Aircraft,” NASA/TM–20230009523, July 2023.
2. E. Bohn, E.M. Coates, S. Moe, and T.A. Johansen, “Deep reinforcement learning attitude control of fixed-wing UAVs using proximal policy optimization,” *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2019.
3. W. Koch, R. Mancuso, R. West, and A. Bestavros, “Reinforcement learning for UAV attitude control,” CoRR, vol. abs/1804.04154, 2018. [Online]. Available: http://arxiv.org/abs/1804.04154.

4. P.S. de Vries and E. Van Kampen. “Reinforcement Learning-based Control Allocation for the Innovative Control Effectors Aircraft,” AIAA 2019-0144. *AIAA Scitech 2019 Forum*. January 2019.
5. T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, “Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor,” arXiv:1801.01290 [cs, stat], Jan. 2018, arXiv: 1801.01290. [Online]. Available: http://arxiv.org/abs/ 1801.01290.
6. T.P. Lillicrap, J.J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, “Continuous control with deep reinforcement learning,” arXiv:1509.02971 [cs, stat], Sep. 2015, arXiv: 1509.02971.
7. J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal Policy Optimization Algorithms,” arXiv:1707.06347 [cs], Jul. 2017, arXiv: 1707.06347.
8. A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, “Stable-Baselines3: Reliable Reinforcement Learning Implementations,” *Journal of Machine Learning Research*, 2021.