Automatic Design of Controller Parameters based on Reinforcement Learning

Beatriz Ventura dos Santos Pereira
beatriz.ventura.s.pereira@tecnico.ulisboa.pt
Instituto Superior Técnico, Lisboa, Portugal

***Abstract*—Due to the increasing demand for Reusable Launch Vehicles (RLVs) to improve the sustainability of rocket launching, research in vertical landing and recovery of rockets has gained significance in recent years. Therefore, the objective is to investigate novel algorithms, implemented as a software package, for the automated design of linear controllers using approaches based on the Youla Parameterization (YP) and Reinforcement Learning (RL) to tackle the attitude control problem of a landing RLV.**

The YP generates the set of all the linear controllers that stabilize a given linear plant by varying a stable transfer function known as the Youla parameter. We propose the adoption of a novel approach to adjust the Youla parameter, employing an RL algorithm, known as Episodic REINFORCE, for unstable systems. Limited research has been conducted on RL for feedback design in circumstances where instability can interfere with the learning or the hardware. The commonly used parameterizations tend to perform poorly in such cases. The combination of RL and YP offers stability, robustness, and performance advantages. The results obtained in this work demonstrate that the problem is successfully addressed, with the algorithm being validated and compared against a state-of-the-art optimization algorithm. An approach to accelerate the convergence of the learning algorithm was proposed.

The integration of RL with YP provides new possibilities for designing robust and efficient control systems for RLVs. This work contributes to the advancement of reusable launch technology by providing an approach to tackle the challenges associated with automatic attitude control.

***Index Terms*—Automatic controller design, Youla-Kucera parameterization, reinforcement learning, REINFORCE, rocket landing**

I. Introduction

A. Motivation and Problem Definition

The design of a controller that complies with prescribed specifications and constraints is a complex task, even for linear systems. To increase the efficiency of the design process, it is convenient to have an automatic design tool that allows an efficient exploration of the effects of different specifications, as well as of the trade-offs between performance and robustness.

This design tool, to be embedded in a software package, must rely on a suitable controller parameterization, that ensures that the search for the optimal controller is made only among stabilizing controllers, combined with an optimization algorithm that can tackle high-dimension problems with complex objective functions, possibly non-convex.

The main goal of this work is to develop a software package using MATLAB for the automatic design of controllers for linear, time-invariant, continuous-time systems, with application in aerospace vehicles' motion control. The algorithm will be exemplified using an attitude control problem for a space vehicle, in this case, a vertical landing rocket.

For such a system, the YP provides a way to parameterize all the controllers that stabilize it ([1], [2]). By varying the Youla parameter Q to optimize the cost functional that defines the goal, a technique called Q-Design, it is possible to find the controller that can satisfy the problem restrictions and guarantee stability. The automation of this process will be done with RL, with focus on the algorithm Episodic REINFORCE ([1], [3], [4]). The use of RL for feedback design systems when instability can disrupt the learning process is not yet heavily studied. Through the Q-parameterization, the YP offers a set of robust stabilizing controllers. This approach will provide the tool used to develop the algorithm to tune controllers, given a linearized model of a rocket landing.

The possibility of extending the YP, present in the control literature, to more complex systems aligned with the use of RL represents a door to apply learning to real-world problems, such as the vertical landing of reusable rockets, a case study of importance to the field of RLVs.

B. Scientific Framework and State of the art

The Youla-Kucera (YK) parameterization is a popular control literature parameterization that had its origin in the 70s when Youla and others developed an analytical feedback design technique for the Single-Input Single-Output (SISO) case and the multivariable case (see [5], [6]) in continuous-time and Kucera added the extension to the discrete-time case [7]. These works proposed the YK parameterization, also commonly referred to as YP, capable of providing all linear stabilizing controllers for a given Linear Time-Invariant (LTI) plant in a feedback control loop, based on the transfer function Q (Youla parameter), that guarantees stability.

The Q-parameterization provides the set of all stabilizing controllers for a given plant that can be characterized by knowing a controller stabilizing the given plant. Many important cost functions are convex in the $Q(s)$ because the closed-loop system is affine in $Q(s)$. Hespanha [2], in lectures 24 and 25, shows how a given Linear Quadratic Gaussian (LQG)/Linear Quadratic Regulator (LQR) controller can be used to parameterize all feedback controllers capable of stabilizing a given LTI system, that is later used as a control design method based on Q parameterization and numerical

optimization. Furthermore, this reference mentions a finite-dimensional optimization technique known as the *Ritz Approximation*.

Ronald J. Williams [4] presents a general class of associative RL algorithms, called REINFORCE algorithms. Also, in [3], chapter 13.4, it is explained how to obtain the REINFORCE from a gradient algorithm.

However, the performance of RL in learning controllers is highly dependent on the controller parameterization used. A poorly chosen parameterization can result in an unstable controller, a higher cost function, and weak learning performance.

Roberts *et al.*, [1] explore four different parameterizations of linear feedback control in the context of REINFORCE with application in the control of a reaching task with a linearized flexible manipulator, where closed-loop instability can be an issue. Here, the manipulator is modeled as an open-loop stable linear system that is underactuated, does not have full-state information, and, with the wrong controller, can quickly become unstable. This paper states that the two most natural-seeming parameterizations *i.e.*, state feedback gains with a fixed observer and a feedback controller transfer function do not guarantee stability, the set of stabilizing controllers are non-convex and even a small change in the parameters can worsen performance to the point of instability. On the other hand, LQR cost matrices with a fixed observer and YP do guarantee stability.

The YP provides a rich set of controllers and presents better performance when the cost function is non-quadratic and the noise is structured but non-Gaussian. Regarding the experimental results, it was concluded that the YP provided the best overall performance. It had a quick predictable convergence, which showed good learning performance and good ultimate controller performance. These results validate the premise that the representation for $Q(s)$ has a great influence on the performance of the parameterization and that with the appropriate choice of $Q(s)$ any stabilizing linear controller can be represented, even the LQR controller.

In conclusion, it is possible to affirm that the application to nonlinear and uncertain systems, such as motion control of aerospace vehicles, is of real-world importance, and applying learning to hardware could benefit from the flexibility and guaranteed stability of the YP.

In the future, there will be an increasing search for reusable rocket development. Research in vertical landing and recovery of rockets has gained prominence in recent years, with the increasing need for RLVs to minimize cost, time, and impact associated with rocket launching. The upswing is related to the successful breakthroughs made by SpaceX and Blue Origin in this field.

C. Objective and Contributions

The objective of this work is to develop a linear controller design algorithm, based on the YP and RL to adjust the Youla parameter Q, to solve an attitude control problem of a landing RLV.

An important contribution of this study is the extension of the approach proposed in [1] to account for open-loop unstable plants. By combining the learning algorithm Episodic REINFORCE and YP, this study addresses not only the attitude control of an RLV but also the challenges associated with more complex systems.

II. Problem Definition

Let us define the general idea of the problem. The objective is to find the optimal vector of parameters that defines the controller as a finite dimension linear combination of basis transfer functions, using an RL algorithm called Episodic REINFORCE, to be defined in section IV, as an optimization algorithm to adjust the stabilizing controller obtained, to minimize a defined cost function.

The optimal parameter ϕ of the Youla Parameter $Q(s)$, is in an infinite-dimensional space. Therefore, it is necessary to approximate it by a search over a finite-dimensional space, as to be detailed in the sections below. For that purpose, one takes

$$\hat{Q}(s) = \sum_{i=1}^{K} \phi_i \left(\frac{\alpha}{s+\alpha} \right)^{i-1}, \quad \phi := \begin{bmatrix} \phi_1 \\ . \\ . \\ . \\ \phi_K \end{bmatrix}, \tag{1}$$

where α is a fixed positive constant closed-loop pole that is selected a priori.

The attitude control cost function

$$J = \sum_{j=1}^{t} (\theta_j - \theta_{goal_j})^2, \tag{2}$$

punishes deviation from the goal attitude angle of the vehicle, given that the error is the sum of the squared difference between the simulation output angle, θ, and the reference angle θ_{goal}.

Next, as to be detailed in section IV, the optimization can be conducted using an RL algorithm, and the results validated using a MATLAB unconstrained optimization problem solver such as *Fminunc*. Finally, it will be possible to obtain the optimal stabilizing controller for the problem.

III. Youla Parameterization

This section defines a parameterization that provides all linear stabilizing controllers for a given LTI plant, the YK parameterization. It is parameterized by a stable transfer function called Youla parameter, Q.

A. Control System Structure

For the SISO case consider, in a general form, a linear system described by the continuous transfer function

$$\frac{B(s)}{A(s)}, \quad \partial B < \partial A.$$

Now, let $\frac{S^0(s)}{R^0(s)}$ be a stabilizing controller, interconnected to the plant.

All rational stabilizing controllers can be defined as

$$\frac{S(s)}{R(s)} = \frac{S^0(s) + Q(s)A(s)}{R^0(s) - Q(s)B(s)}, \tag{3}$$

where $Q(s)$ is stable, and can be expressed as

$$Q(s) = \frac{Y(s)}{X(s)}, \tag{4}$$

in which $X(s)$ and $Y(s)$ are polynomials. Thus, the stabilizing controller (3) can be rewritten as

$$\frac{S(s)}{R(s)} = \frac{X(s)S^0(s) + Y(s)A(s)}{X(s)R^0(s) - Y(s)B(s)}. \tag{5}$$

The closed-loop transfer function is given by

$$H(s) = \frac{\frac{B(s)}{A(s)}\frac{S(s)}{R(s)}}{1 + \frac{B(s)S(s)}{A(s)R(s)}} = \frac{B(s)S(s)}{A(s)R(s) + B(s)S(s)}. \tag{6}$$

Since $Q(s)$ is stable, $X(s)$ must have all its roots located in the left half of the complex plane $Re(s) < 0$. Therefore, the closed-loop is stable.

All the stabilizing controllers can be written in the form (3), with stable Youla parameter $Q(s)$. Let $\frac{S(s)}{R(s)}$ be a stabilizing controller that yields the closed-loop characteristic polynomial

$$AR + BS = C, \tag{7}$$

with all roots of $C(s)$ located strictly in the left half of the complex plane.

Thus, from (7) the Youla Parameter $Q(s)$ can be written as

$$Q(s) = \frac{S(s)R^0(s) - R(s)S^0(s)}{A(s)R(s) + B(s)S(s)}, \tag{8}$$

which is stable, since $C(s)$ has all its roots located strictly in the left half of the complex plane.

B. Special Case: Open-loop Stable System

In [1] is presented a special case example of the YK parameterization for a finite-dimensional SISO stable plant $P(s)$ and feedback controller $K(s)$.

Here, the Youla parameter associated with the controller $K(s)$ is given by

$$Q(s) := \frac{K(s)}{1 - K(s)P(s)}. \tag{9}$$

The closed-loop system can be presented as

$$H(s) = P(s)[1 + Q(s)P(s)], \tag{10}$$

that is affine in $Q(s)$. It is important to note that the relationship between the parameters of the feedback controller $K(s)$ and the closed-loop system $H(s)$ is nonlinear.

Moreover, it is clear that for any $K(s)$, the parameter (9) exists, and if the plant $P(s)$ is stable, then $H(s)$ is also stable, if and only if the Youla parameter $Q(s)$ is stable.

This representation of the YK parameterization presented by Roberts *et al.* [1] is a particular case of the one shown in subsection III-A.

It is possible to obtain the special-case controller in the form of (3), if the system is open-loop stable. For that purpose, consider the stabilizing controller $\frac{S^0}{R^0}$ to be $S^0(s) = 0$ and $R^0(s) = A(s)$, with positive feedback.

In conclusion, every feedback controller $K(s)$ can be represented by $Q(s)$. If and only if $Q(s)$ is stable and affine, the closed-loop system will also be stable and affine. Moreover, because the closed-loop system $H(s)$ is affine in $Q(s)$, many cost functions, including LQG, are convex in the Q(s).

As a result, we may state that the set of all stable $Q(s)$ is an affine parameterization of all stabilizing linear controllers for a plant $P(s)$.

C. General Case: Open-loop Unstable System

As previously expressed in (3) which defines all rational stabilizing controllers, there is an initial stabilizing controller. This section introduces an LQG as the $\frac{S^0(s)}{R^0(s)}$ controller.

The goal is to achieve a more robust controller and generalize the parameterization to a wider set of cases. The LQG controller, implemented in a state-space framework, combines a state estimator and a controller to stabilize the system.

The separation theorem allows for independent design of observer gains and controller gains. The control gain vector is determined by minimizing a quadratic cost function, and the positive definite matrix P is obtained by verifying the Algebraic Riccati Equation (ARE). The controllability and observability of the system are important criteria to ensure the design's feasibility. The matrix Q_{LQ} determines the weights of the states, and R the weights of the control input in the cost function. These weight matrices are used to adjust the trade-off between control effort and response speed. A Kalman filter can be employed for state estimation, and the integral effect was included to eliminate steady-state errors. Adding the LQG controller as a stabilizing controller to the Youla controller in (3), will allow the controller to handle complex cases such as unstable plants with non-minimum phase roots.

D. Finite Dimension Approximation

One of the biggest obstacles in Q Design is that it is necessary to search over an infinite-dimensional set of all stable transfer functions Q. However, as proven in [8], the *Ritz approximation* can be used to solve infinite-dimensional optimization problems by converting them into finite-dimensional subsets.

The steps to achieve the Ritz approximation are:

1) Select a sequence of $k \times m$ Bounded Input, Bounded Output (BIBO) stable transfer functions that is *complete* such as

   $$\hat{Q}_1(s), \hat{Q}_2(s), ..., \hat{Q}_i(s), ...,$$

   A *complete* Q sequence [2] is a set of all BIBO stable transfer functions $\hat{Q}(s)$, for that there is a *finite* linear combination of $\hat{Q}_i(s)$ that is arbitrarily close to the original $\hat{Q}(s)$. The linear combination is obtained by defining $\hat{Q}_i(s)$ as

   $$\left(\frac{\alpha}{s+\alpha}\right)^{\ell}, \quad \alpha_i \in \mathbb{R}, \tag{11}$$

   for $\ell \geq 0$ and a chosen fixed constant $\alpha > 0$.
2) Restrict $\hat{Q}(s)$ to be

   $$\hat{Q}(s) := \sum_{i=1}^{K} \alpha_i \hat{Q}_i(s), \qquad \alpha_i \in \mathbb{R}. \tag{12}$$

3) The closed-loop transfer function matrix from any exogenous input signal to a given controlled output, $\hat{H}(s)$ can be written as

$$\begin{aligned} \hat{H}(s) &= \hat{H}_0(s) + \sum_{i=1}^{K} \alpha_i \hat{H}_i(s), \\ \hat{H}_i(s) &:= \hat{L}(s)\hat{Q}_i(s)\hat{R}(s). \end{aligned} \tag{13}$$

The transfer function $\hat{Q}_i(s)$ and transfer matrices $\hat{H}_0(s)$, $\hat{L}(s)$, $\hat{R}(s)$ are BIBO stable. Different inputs will lead to different transfer matrices, but will still be affine in $\hat{Q}(s)$.

4) Define the feasibility problem as

$$\begin{aligned} \text{find} \quad & \alpha_1, \alpha_2, ..., \alpha_N \quad \in \mathbb{R} \\ \text{such that} \quad & \hat{H}_0(s) + \sum_{i=1}^{K} \alpha_i \hat{H}_i(s) \\ \text{satisfies} \quad & \mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_k, \end{aligned} \tag{14}$$

for a family $\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_k$ of time domain and input-output closed-loop specifications. To solve the feasibility problem all specifications must be convex.

5) Use the obtained Q system from the feasible problem (14) to compute the Youla controller. However, if the problem is not feasible one can try to make it feasible by increasing the number of iterations N until it becomes inadmissible to compute the numerical optimization or the order of the Q-augmented controller becomes unreasonable. In that case, the problem might not have a viable solution for the determined specifications.

IV. Learning Algorithm: Episodic REINFORCE

The algorithm used for learning in this work is an extension of the class of REINFORCE algorithms called Episodic REINFORCE, mainly used for learning tasks that have a temporal credit-assignment component, known as a (Temporal) Credit Assignment Problem (CAP).

In Episodic REINFORCE the learning is performed episode-by-episode *i.e.* the system states are reset at the end of each policy evaluation and the stochasticity of the policy π(y,ϕ) is on the parameters ϕ, not the outputs of the system y. For this work, it is used a specific update derived in [1] - from the update that appears in [4] for learning the mean of a Gaussian element - that learns a vector of parameters with identical noise and learning rate.

From this point on, the notation is as follows: $\phi^{i'}$ are the actual parameters used on trial i, b is a cost baseline, J($\phi^{i'}$) is the cost associated with the policy parameters $\phi^{i'}$, and g($\phi^{i'}$) is the probability of using parameters $\phi^{i'}$ in trial i. The REINFORCE update

$$\phi^{i+1} = \phi^i - \eta(J(\phi^{i'}) - b))\frac{\partial}{\partial \phi^{i'}} ln(g(\phi^{i'})), \tag{15}$$

is formulated for cost instead of reward and the learning rate η is the same for the vector of parameters ϕ. In this case, the eligibility $\frac{\partial}{\partial \phi^{i'}} ln(g(\phi^{i'}))$ can be defined as

$$\frac{\partial}{\partial \phi^{i'}} ln(g(\phi^{i'})) \propto (\phi^{i'} - \phi^i), \tag{16}$$

in which $\phi^{i'} = \phi^i + \phi^{p_i}$, and considering that g($\phi^{i'}$) is a multivariate Gaussian distribution with mean ϕ^i, that has independent noise on each element with covariance σ^2.

Thus, it is possible to formulate the update

$$\phi^{i+1} = \phi^i - \eta(J(\phi^i + \phi^{p_i}) - J(\phi^i))\phi^{p_i}. \tag{17}$$

While in literature it is common to use an average baseline [1], we employed a second policy evaluation in order to learn in fewer iterations at the expense of having to carry out two evaluations per update. First, the policy given by ϕ^i is executed, in iteration i, to obtain the baseline

$$b = J(\hat{Q}(\phi^0)).$$

Then, the policy is perturbed by a small value ϕ^{p_i}, and that perturbed policy is executed. The generated white noise follows the standard normal distribution, with Probability Density Function (PDF) of the form

$$\frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}}, \tag{18}$$

in which x is a generated random real variable, with mean $\mu = 0$ and constant variance $\sigma = 1$. The introduction of small white noise in the algorithm adds variability and randomness. In addition, stochastic strategies encourage exploration of different courses of action and possibly find better strategies than a deterministic strategy.

Finally, the difference in performance is calculated, and the policy update (17) is computed. The system is run for N iterations until it converges to an optimal value and the cost function is minimized.

A. Algorithm Convergence

Even when REINFORCE is successful, the convergence to a local minimum is very slow. It has been discovered that the main algorithm in this study, Episodic REINFORCE, is particularly slow, but this is also not unexpected given that it carries out temporal credit-assignment by effectively dispersing credit or punishment over all previous times [4].

Analytical calculation of the convergence rate of stochastic algorithms such as the Episodic REINFORCE can be difficult, since it relies on a series of random factors, and the probability that the algorithm converges to a local maxima or minima depends on the decision made for the reinforcement baseline.

The use of a variation of the conjugate gradient method was one of the possible solutions implemented to make the convergence faster. The gradient conjugate method is used for solving linear systems and function optimization, *i.e.* finding the minimum of a quadratic function. The literature proposes several definitions of the conjugate gradient method for minimizing quadratic functions, *e.g.* [9].

The method consists of a sequence of conjugate directions that are designed to search the solution space more efficiently. Here, we perform the Episodic REINFORCE update (17) every two iterations, and at the third, we use the conjugate gradient method to compute the next search direction using the linear combination of the previous search direction, as illustrated in figure 1. This helps minimize oscillations and

speeds convergence by ensuring the new search direction is conjugate with the prior direction. Up until the defined stopping criterion is met, this process is repeated iteratively.

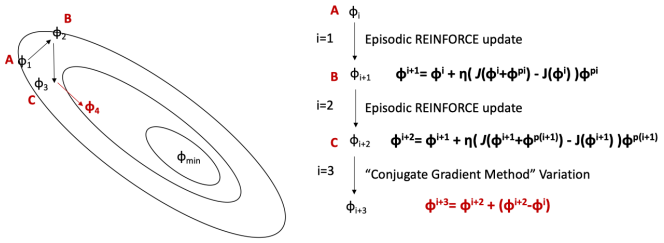


Fig. 1: Diagram of a variation of the conjugate gradient method implemented for convergence of the RL algorithm.

In figure 2 is depicted the convergence comparison between the algorithm with the conjugate gradient technique variation, and the vanilla method. The initial conditions of the optimization problem are the same in this example; the only distinction is in the approaches of convergence. Although both algorithms converge to the same value, the minimum cost is found at iteration 907 using the conjugate gradient technique, but only at iteration 1418 when utilizing the regular update. As a result, it has been proven to boost speed by approximately 64%.

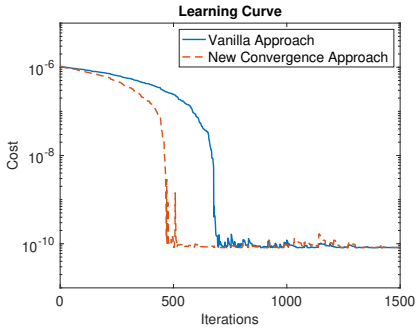


Fig. 2: Comparison of the convergence of the algorithm with and without the conjugate gradient method modification.

The learning rate is the hyperparameter that controls the size of the update on each iteration of the algorithm.

Finding the proper step size for a given optimization problem is crucial for ensuring convergence of the algorithm, given that if the step size is too large, the process is faster but the algorithm may pass the minimum of the loss function and fail to converge. Instead, if the step size is too small, the algorithm might take a long time to converge or get stuck in a local minimum. This process can be done through trial and error or with known techniques such as adaptive learning rate methods. A variety of step size adaptation methods for stochastic learning have been presented in the literature (see [10], [11]). In this work we concentrate on adapting the learning rate η based on percentual variation of cost, starting from an arbitrarily large value of step size and adapting it based on optimization performance.

For that, when the cost increases a defined percentual value instead of decreasing, the learning rate is altered and the algorithm returns to the previous value of the optimization parameter and cost. In the last iterations, usually the final 5%, it is important to reduce the learning rate significantly to avoid surpassing the minimum value. Thus, during these last iterations, the learning rate progressively decreases proportionally to the number of iterations left.

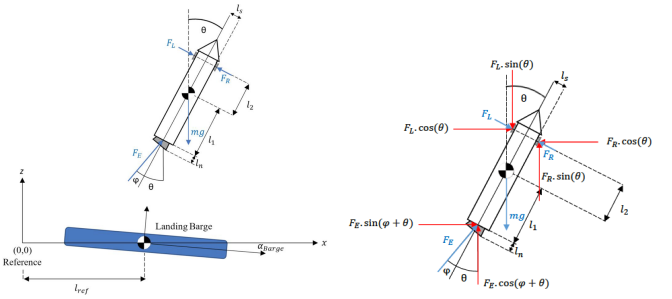
The step gain is also normalized based on the value of cost before it is multiplied by the search direction, *i.e.* the learning rate in the Episodic REINFORCE policy update (17) is divided by that iteration's cost $J(\phi^i)$, making the step gain $\eta/J(\phi^i)$. Hence, in an ideal example, if the random noise constant ϕ^{pi} in that iteration makes the perturbed policy zero, the search direction is $J(\phi^i)\phi^{pi}$. In these circumstances, the next iteration's value of ϕ will represent cost zero as $J(\phi^i + \phi^{pi})$.

V. Rocket Model

A. Nonlinear Model

The mathematical derivation of the rocket will be based on Newton's 3rd law of motion.

Figure 3(a) illustrates a simplified representation of a rocket landing and 3(b) depicts all the considered forces.



(a) Diagram of a rocket landing in a barge.

(b) Diagram of a rocket with all forces represented.

Fig. 3: Rocket simplified 2D representation [12].

The notation [12] used moving forward is

- F_E: main thruster force;
- F_R: right thruster force;
- F_L: left thruster force;
- F_S = F_L - F_R : right and left thruster forces as a single input;
- θ: angle between the z axis of the plan and the longitudinal axis of the rocket;
- φ: angle between the x axis of the plan and the longitudinal axis of the rocket;
- l_1: longitudinal length between the Center of Gravity (COG) of the rocket and F_E;
- l_2: longitudinal length between the COG of the rocket and F_R, F_L;
- l_n: length of the nozzle;
- m: sum of the rocket's dry mass and fuel mass;
- x: horizontal position of the rocket;
- z: vertical position of the rocket;
- α, β: real constants.

The state of the rocket dynamics depends on the horizontal and vertical positions of the rocket (x, z) and its velocity ($\dot{x}$,

$\dot{z}$), as well as the angle between the z axis and the longitudinal axis of the rocket (θ) and the angular velocity $\dot{\theta}$. This can be defined by the vector of states

$$X = \begin{bmatrix} x & \dot{x} & z & \dot{z} & \theta & \dot{\theta} \end{bmatrix}'. \tag{19}$$

The inputs, or main control variables of the rocket, are the main engine thrust F_E, the right and left side Nitrogen gas thrusters, that can be simplified into one input $F_S = F_R - F_L$, and the thrust angle φ. Nitrogen gas thrusters can be set to zero for simplicity even though they provide a more stable control.

The control outputs are x, z and θ.

We simplify the next equations for small angles, in which $cos(\theta) = cos(\varphi) \approx 1$, $sen(\theta) \approx \theta$, and $sen(\varphi) \approx \varphi$. Now, it is possible to infer the expressions of the translational forces with respect to the COG:

$$\begin{aligned} \ddot{x} &= \frac{F_E \cdot sen(\theta + \varphi) + F_S \cdot cos(\theta)}{m} \\ &\approx \frac{F_E \cdot \theta + F_E \cdot \varphi + F_S}{m}, \end{aligned} \tag{20}$$

$$\begin{aligned} \ddot{z} &= \frac{F_E \cdot cos(\theta + \varphi) - F_S \cdot sen(\theta) - mg}{m} \\ &\approx \frac{F_E - F_E \cdot \varphi \cdot \theta - F_S \cdot \theta - mg}{m}. \end{aligned} \tag{21}$$

Since the shape of the rocket does not change, the moment of inertia equation can be described by

$$J_T \cdot \ddot{\varphi} = \tau, \tag{22}$$

in which τ represents an applied torque on a rocket to the angular acceleration, $\ddot{\varphi}$.

When the nozzle angle, φ, which typically moves in three dimensions, is different from zero, a torque is generated. However, in this work, the problem will be reduced to a two-dimensional system, which results in a unidimensional rotation. The rotation torque with respect to the COG can be computed by

$$\begin{aligned} \ddot{\theta} &= \frac{-F_E \cdot sen(\varphi)(l_1 + l_n \cdot cos(\varphi)) + F_S \cdot l_2}{J_T} \\ &\approx \frac{-F_E \cdot \varphi(l_1 + l_n) + F_S \cdot l_2}{J_T}, \end{aligned} \tag{23}$$

where J_T is the moment of inertia, and $\ddot{\theta}$ is the angular acceleration.

The fuel burn of the rocket is modeled by

$$\dot{m} = -\alpha(\beta \cdot F_E - F_S), \tag{24}$$

which is directly proportional to the thrust.

B. Analytical Linearization

If the problem is LTI, it can be written in the state-space form

$$\begin{aligned} \dot{x} &= Ax + Bu, \\ y &= Cx + Du. \end{aligned} \tag{25}$$

However, the aforementioned model is not linear. Therefore, it must be linearized around an equilibrium point in order to design a controller.

Consider the general nonlinear differential equation written as

$$\dot{x}(t) = f(x(t), u(t)), \tag{26}$$

and take the function f, that maps $\mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n$, an equilibrium point $\overline{x} \in \mathbb{R}^n$ and an equilibrium input $\overline{u} \in \mathbb{R}^m$, so that

$$f(\overline{x}, \overline{u}) = 0. \tag{27}$$

A point is considered to be in equilibrium if, given a certain equilibrium input, all changing states go to zero. We set $\ddot{x}$, $\ddot{z}$, and $\ddot{\theta}$ to zero. Hence, by solving equations (20)-(23) it is possible to obtain the equilibrium input given by

$$\overline{u} = [mg, 0, 0]. \tag{28}$$

Now, it is possible to perform the Jacobian linearization [13] on a nonlinear differential equation about the point $(\overline{x}, \overline{u})$.

The theory states that if one starts the simulation in the equilibrium point, $x(t_0) = \overline{x}$, and applies the equilibrium input $u(t) = \overline{u}$, the system will remain in equilibrium for all t. Nevertheless, if one starts near $\overline{x}$, there will be deviations associated, such as $\Delta_x(t)$ and $\Delta_u(t)$. Given that, it is possible to rewrite (26) as

$$\dot{\Delta}(t) = f(\overline{x} + \Delta_x(t), \overline{u} + \Delta_u(t)). \tag{29}$$

After applying the Taylor Expansion to (29), ignoring the high-order terms, and considering (27), one obtains

$$\dot{\Delta}_x(t) \approx \left.\frac{\partial f}{\partial x}\right|_{x=\overline{x}, u=\overline{u}} \Delta_x(t) + \left.\frac{\partial f}{\partial u}\right|_{x=\overline{x}, u=\overline{u}} \Delta_u(t), \tag{30}$$

that can be of the form

$$\dot{\Delta}_x(t) \approx A\Delta_x(t) + B\Delta_u(t), \tag{31}$$

with A and B being the (25) system matrices.

To obtain the matrices A and B analytically, the problem (31) is equivalent to the computation of the partial differentiation on the state equations (19) and the inputs. The resulting matrices are

$$\begin{aligned} A &= \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\overline{F_E}}{m} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{(-\overline{F_E} \cdot \overline{\varphi} - \overline{F_S})}{m} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \\ B &= \begin{bmatrix} 0 & 0 & 0 \\ \frac{(\overline{\theta} + \overline{\varphi})}{m} & \frac{1}{m} & \frac{\overline{F_E}}{m} \\ 0 & 0 & 0 \\ \frac{(1 - \overline{\theta} \cdot \overline{\varphi})}{m} & -\frac{\overline{\theta}}{m} & \frac{(-\overline{F_E} \cdot \overline{\theta})}{m} \\ 0 & 0 & 0 \\ \frac{(-\overline{\varphi}(l_1 + l_n))}{J_T} & \frac{l_2}{J_T} & \frac{(-\overline{F_E}(l_1 + l_n))}{J_T} \end{bmatrix}. \end{aligned} \tag{32}$$

C. Nominal Trajectory

Next, we will define the rocket's nominal trajectory, which will serve as a reference throughout this work.

The goal is to find a nominal trajectory for the rocket, in order to reach the ground with a final velocity equal to zero. For that, we consider a *bang-bang* control trajectory, or "on-off", where the main thrust force signal switches abruptly between two states *i.e.* completely OFF ($F_E \approx 0$N) and fully ON ($F_E = F_{E_{(max)}} = 7 \times 10^6$N). Here, the thrust force will have a minimum value of 5% of the maximum force value and, therefore, will not be completely OFF so it is still possible to control the gimbal angle, φ. As illustrated in figure 4, the moment of the switch is represented as t_{ON}. Since the desired trajectory is vertical, the gimbal angle is assumed to be zero.

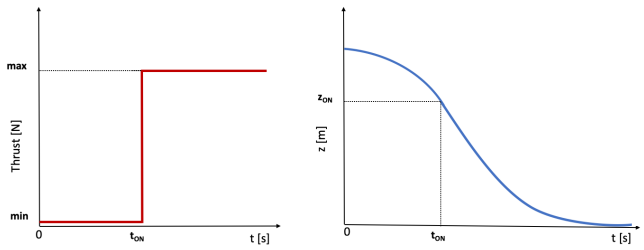


Fig. 4: *Bang-bang* profile.

The resulting vertical position, z, of the rocket when the thrust force depicted in figure 4 is applied, can be computed analytically using the equation of motion. For the first $[0, t_{ON}]$ seconds, the acceleration that acts on the body is the sum of the gravity with the quotient of the minimum thrust force applied and the mass. Therefore, considering the initial velocity to be zero, the position z_{ON} is of the form

$$z_{ON} = z_0 + \frac{1}{2}(-g + \frac{F_E}{m})(t_{ON})^2. \tag{33}$$

We ran several open-loop simulations to observe how different "switch" time instants, t_{ON}, that occur when the state changes from zero to the maximum force value, affect trajectory. For a chosen time instant of $t_{ON} = 14.6$ seconds, the vertical position z_{ON} on time instant t_{ON} is, approximately, 3023m. When analyzing the simulation results, it is proven that the rocket is descending fast for the first t_{0N} seconds until the thrust value switches and the trajectory changes, slowly reaching zero. When performing open-loop simulations on the rocket model, it is possible to verify that the rocket reaches the ground with a velocity of approximately zero.

D. System Identification

The F_L and F_R forces were adjusted to zero, and the 6th-order multivariate system was reduced to a direct input-output relationship to make the identification process simpler. Therefore, two different identification processes must be carried out to build the model.

Using data from the open-loop simulation, the MATLAB function *ssest* is used to estimate a continuous-time state-space model of a specific order. In order to build the attitude and altitude models, one should: **1.** Remove the double integral effect of $\ddot{\theta}$ and $\ddot{z}$, deriving it twice. **2.** Identify the model that relates: the input gimbal angle φ with the angle θ, and the input thrust force F_E with the vertical position of the rocket z. **3.** Add a double integrator to the model.

1) ***Attitude System****:* The input u of the attitude model can be determined by

$$\Delta\varphi = \varphi - \overline{\varphi}, \tag{34}$$

in which the gimbal angle φ is an input of the system, and the equilibrium value $\overline{\varphi}$ is zero radians. The input is a square wave with 25% duty cycle, representing the input gimbal angle with an amplitude of 1×10^{-6} radians, oscillating around the equilibrium value. The output y is the second derivative of the angle θ, from the *Simulink* simulation. The rocket is "hovering" since only the gimbal angle oscillates. The thrust force input is considered constant, equal to the mass times gravity of the system.

It is possible to relate the output ($\ddot{\theta}$) and the input (gimbal angle φ) by a gain $\alpha = -8146$, with the MATLAB function *ssest*. The fit to estimation data, of the form 100(1-Normalized Root Mean Squared Error (NRMSE)), is 96.51% and has a Mean Squared Error (MSE) of 6.44×10^{-08}.

Next, a double integrator needs to be added to design the linear controller. The LTI problem can be written in the classic state-space form defined in (25) as

$$\begin{cases} \dot{x} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ \theta \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \ddot{\theta}, \\ y = \begin{bmatrix} 0 & -8.146 \times 10^3 \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ \theta \end{bmatrix}. \end{cases} \tag{35}$$

2) ***Altitude System****:* For the altitude system identification, the input u is given by

$$\Delta F_E = F_E - \overline{F_E}, \tag{36}$$

where the thrust force F_E is a square wave and the equilibrium value is $\overline{F_E} = mg$ N. The input oscillates around the equilibrium value.

In this case, the output y is the second derivative of the position z from the *Simulink* simulation. The gimbal angle is forced to zero radians, so that the trajectory remains vertical.

As previously explained, given the input signal, it is possible to relate the output $\ddot{z}$ and the thrust force F_E by a gain $\beta = 1.820 \times 10^{-6}$ obtained using the MATLAB function *ssest*. Here, the fit to estimation data is of 96.52% and the MSE is 0.128. As stated for the attitude system, it is necessary to add the double integrator in order to design the linear controller.

From that, it is possible to express the LTI system in the classic state-form (25) as

$$\begin{cases} \begin{bmatrix} \dot{\dot{z}} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{z} \\ z \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \ddot{z}, \\ y = \begin{bmatrix} 0 & 0.1820 \times 10^{-5} \end{bmatrix} \begin{bmatrix} \dot{z} \\ z \end{bmatrix}. \end{cases} \tag{37}$$

E. Baseline Controller Design

It is necessary to design a baseline state feedback LQG, to control the attitude and altitude of the rocket. We will use it as an initial stabilizing controller in the YK parameterization.

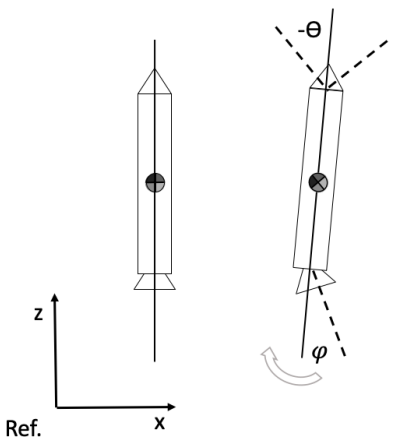


Fig. 5: Orientation of the rocket.

From the diagram shown in figure 5, it is possible to infer that if the input gimbal angle φ is positive, a negative torque will be generated. In order to balance the angle of the rocket, the controlled angle θ must be negative.

To design the attitude LQR controller for the angle between the z axis of the plan and the longitudinal axis of the rocket, θ, the matrix B identified for the LTI system in (35) must be multiplied by the output matrix C, to obtain a negative controller gain. Hence, after properly adjusting the weight matrices R and Q_{LQ}, the controller gain vector K can be obtained with MATLAB function *lqr*.

The eigenvalues determine the evolution of the system, by analyzing whether a fixed point is stable or unstable. A fixed point, or equilibrium point, is stable if when disturbed returns to its initial value or remains in the same location. The poles of the closed-loop system $A-BK$ can be obtained by computing the eigenvalues of that system. It is known that $A-BK$ is a stability matrix if all the poles have a negative real part. The separation theorem allows the regulator and estimator gains of the LQG controller to be designed independently from each other and guarantees that the controller makes the closed-loop system asymptotically stable, given that both $A-BK$ and $A-LC$ are stability matrices. Next, it is necessary to adjust the weight matrices Q_E and R_E to obtain the optimal LQG estimator gain L using the MATLAB function *lqe*.

It is now possible to compute the state-space model for a negative output feedback LQG controller for the process system with regulator gain K_θ and estimator gain L_θ. The attitude baseline controller is used as an initial stabilizing LQG controller, $\frac{S^0(s)}{R^0(s)}$, detailed in section III-C.

Using the same method, it is possible to design the LQG controller for the vertical position of the rocket, z. The goal is to follow the nominal trajectory detailed in section V-C. Here, the equilibrium point for the altitude controller is different than zero. Hence, it is necessary to subtract the z_{ON} position obtained for the nominal trajectory from the input altitude.

VI. Experimental Results and Discussion

For the simulations, the mass of the rocket was assumed to be constant and the Nitrogen gas thrusters F_S were set to zero for simplification.

The constant values considered for the simulations are presented in Table I. The dimensions of the rocket were chosen accordingly to the first-stage rocket data of Falcon 9 [14], [15].

TABLE I: Simulation constants.

m [Kg]	g [m/s^2]	l_1 [m]	l_2 [m]	l_n [m]	J_T [Kg m^2]
549054	9.8	60	10	1	40267

A. Implementation Details

The choice of the initial parameters ϕ^0, α, and η is important since it influences the evolution of the system. The definition of the appropriate base function that computes the parameter $Q(s)$ and represents the stabilizing controllers, shown in (1), as well as the type of input signal to which the system will respond, affect the optimization process and the speed of convergence to a minimum.

The learning rate must be dynamic. The technique used is expressed in section IV-A.

The choice of the initial vector ϕ^0 can determine whether the algorithm becomes "stuck" in local minima. Also, the pole α should be chosen within the range of frequencies for which the closed-loop response is expected to behave interestingly [2].

The number of iterations N is important because increasing the number of iterations until the computation is one of the only methods to ensure that a solution that minimizes the cost can be found.

In stochastic algorithms such as Episodic REINFORCE, the seed is important since it initializes the random number generator, ensuring the reproducibility of results. Therefore, by altering the seed value, it is possible to provide information about the algorithm's sensitivity to the chosen initial conditions and help in the identification of robust approaches that function well under different randomization circumstances.

B. Youla-Kucera Parameterization Applied to the Rocket Model

This section presents the experimental results of the attitude control problem of the rocket model derived in section V, using the Episodic REINFORCE algorithm. The problem is formulated in section II, to find the optimal Youla parameter that provides all the stabilizing controllers for the linear plant $P(s)$.

The objective of the attitude controller is to guarantee that by changing the input gimbal angle φ, the trajectory remains vertical, which entails maintaining the angle θ close to zero regardless of the disturbance applied. The gimbal angle limit of oscillation should range between $[-15, +15]^\circ$.

The altitude controller is the LQG baseline controller designed in section V-E, which assures that the rocket follows the defined nominal trajectory in section V-C. For that, the thrust force applied in all simulations is a *bang-bang* profile force, as detailed in section V-C.

The metric used to evaluate the deviation of the controlled systems from the goal angle θ and the nominal trajectory

is the Sum of Squares Error (SSE). The cost function used is expressed in (2), and all results will be compared to the ones obtained using the baseline controllers to validate the optimization.

*1) **Results and Discussion**:* In this test, the input gimbal angle φ is a square wave with 25% duty cycle, with an amplitude of 1×10^{-6} radians, used to disturb the rocket on its descent trajectory. Using a *Simulink* block, a band-limited white noise was added to the horizontal acceleration of the rocket within the nonlinear model to imitate real-world circumstances. It had a seed value of [23341], a sample time (correlation time of noise) of 1, and a power setting of 1×10^{-4}. The wind factor was used in the study to take external factors' effects on rocket attitude control into consideration.

The initial parameter ϕ^0 is set to zero, to ensure the initial condition of the parameter is equal to the baseline LQG attitude controller, since that if the Youla parameter $Q(s)$ is zero the controller is the initial stabilizing controller S^0/R^0, as proven in (3).

Initially, a test of the parameter ϕ was done, in which the cost was manually calculated for a range of values of this parameter to speculate the one that would give the minimum cost, in which is possible to conclude that the goal final value of ϕ is around -1012.5.

From the pole-zero map of the Youla Controller (3), it was possible to conclude that $\phi = -1012.5$ puts, simultaneously, the farthest and the closest pole to the origin before it gets unstable. It is known that for the system to be able to reject input disturbances the controller has to have a pole at the origin. The cost function used in this work is "blind" to elements such as the effect of fuel burn on the rocket's mass and other factors that influence performance. Thus, the controller prioritizes speed over robustness. The location of the poles of the controller has a great impact on the system's closed-loop behavior.

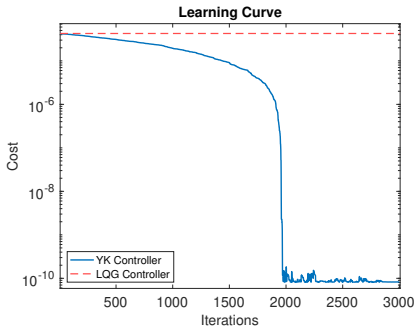


Fig. 6: Learning curve of the Episodic REINFORCE algorithm compared to the cost of the attitude LQG controller.

The comparison between the learning curve of the Episodic REINFORCE algorithm and the constant cost of the LQG controller is found in figure 6. The optimization began with a learning rate of 100, that was gradually adapted using the techniques detailed in section IV-A. The minimum was found at $\phi^{2733} = -1012.5$, corresponding to an SEE of 8.16×10^{-11}. The solver *Fminunc* stopped at the parameter $\phi = -1011.99$ corresponding to an SSE of 9.57×10^{-11}. The baseline controller error is $SSE_{LQG} = 4.26 \times 10^{-5}$, which corresponds to an, approximately, 99% optimization.

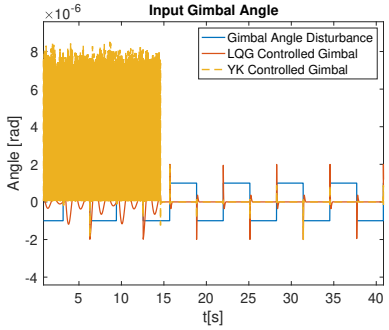


Fig. 7: Sum of input signals using the YK controller compared to the LQG controller.

The gimbal angle stabilizes around the approximate value of zero. The gimbal angle φ is the difference between the sum of the disturbance signal with the control signal shown in figure 7. The heavy oscillations for the first 14.6 seconds present in figure 7 correspond to the *bang-bang* control trajectory "switch" time instance t_{ON}, in which the thrust is considered OFF. The peaks observed in the controlled gimbal angle for each square in the square wave input do not represent overshoot, but rather the fact that the signal experiences a slight delay in its response to a quick change in the input signal, and are less pronounced with the Youla controller.

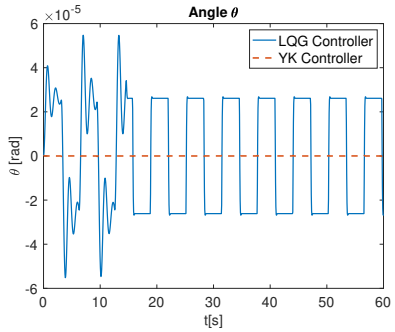


Fig. 8: Comparison of the controlled angle between the z axis of the plan and the longitudinal axis of the rocket, θ, using the YK or the LQG controller.

In figure 8 are depicted the two output angles from the two different controllers. As shown, the Youla controller lead to an attitude angle much closer to the reference value of zero.

With the addition of horizontal wind on the second derivative of the position x on the nonlinear model of the rocket, it is expected to have more horizontal deviation than the ideal vertical trajectory, even with an optimized controller for the attitude angle. It is possible that the cost function (2) mainly highlights the error between the reference and output attitude angle without explicitly considering other trajectory-related elements. Hence, the cost is optimized. However, since the cost function does not include the tracking error of the position x, the trajectory is still deviated.

Regardless of the seed value used in the experiments, the algorithm converged to the same minimum cost value. This

indicates that the performance and convergence behavior of the algorithm is robust and stable, showing some degree of independence from the initial seed value. These results confirm the reliability and consistent performance of the algorithm, even with random variations in the seed.

VII. Conclusions and Further Investigation

This thesis extends the results in [1] by applying the YP to complex and unstable systems, while also using RL for feedback design in situations where instability can disrupt the learning process, such as real-world systems. The YP ensures stability and convexity, making it suitable for the unstable rocket model.

The decision to add an initial stabilizing LQG controller to the Youla controller (3), ensured stability and robustness during the initial stages of control before the Youla controller.

The YP can introduce complexity and increase the order of the control problem, posing challenges for RL algorithms due to high-dimensional action spaces. The RL algorithm used represents a general class of stochastic algorithms called Episodic REINFORCE. The lack of a general theory of convergence that applies to this class of algorithms, as well as their apparent propensity for convergence to local minima, are the main disadvantages of REINFORCE algorithms.

The reliability and validity of the RL algorithm are confirmed by comparing its results with the MATLAB unconstrained optimization problem solver *Fminunc*. When the REINFORCE algorithm fails to converge to the global minimum, it is important to improve the random initialization of parameters ϕ or use a richer input signal that provides both systems with more data for the optimization. Increasing the number of algorithm iterations or expanding the dimensionality of the Youla parameter can also make the optimization problem feasible in certain cases.

The smooth relationship between policies and their cost function values facilitates convergence for the REINFORCE algorithm. Smaller optimization problem dimensions lead to smaller action spaces, enabling more efficient search for desired optima. However, the presence of high variance or noisy rewards obtained from the environment makes it harder for the algorithm to estimate the quality of the actions and obtain reliable updates, making the convergence slower.

A method discovered through research to speed the convergence and avoid local minima involves finding a suitable learning rate and alternating between the REINFORCE update and a traditional method like the conjugate gradient method. This variation of the conjugate gradient method, typically used to minimize quadratic functions, significantly reduces convergence time by enabling more efficient search within the optimization environment.

Overall, the successful combination of the YP with a learning algorithm solves the attitude control problem of landing an RLV by utilizing a well-defined model, cost function, and an initial stabilizing controller for robustness. It is encouraging to apply the research of the YK parameterization presented in the literature to real-world circumstances because of its reliability. However, the REINFORCE algorithm has to be strengthened to allow for confident learning applications to hardware.

Some suggestions for future research are:

- Account for fuel burn in the rocket model control system to improve performance, stability, and real-world applicability.
- Include more complex mathematical rocket models, considering aerodynamic forces like drag.
- Implement multiple controllers simultaneously to enhance performance, specifically addressing the requirements before and after the "switch" time in the *bang-bang* thrust profile.
- Expand the research focus from SISO control systems to Multiple-Input Single-Output (MISO) control systems.
- Define a more explicit cost function with specific penalties and rewards and selecting appropriate weights to balance their importance to improve the control performance.
- Further investigate the Episodic REINFORCE learning algorithm to improve reliability for complex, high-order systems, make it less dependent on initial conditions, and explore techniques to improve convergence.

References

[1] J. Roberts, I. Manchester, and R. Tedrake, "Feedback controller parameterizations for reinforcement learning," in *2011 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, 05 2011, pp. 310 – 317.

[2] J. P. Hespanha, "Linear systems theory," Princeton Press, 2009.

[3] R. Sutton and A. Barto, "Reinforcement learning, vol.2," 2018.

[4] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning. machine learning, vol. 8, no. 3–4," 05 1992, p. 229–256.

[5] J. B. D. Youla and H. Jabr, "Modern wiener–hopf design of optimal controllers part i: The single-input-output case," 02 1976, pp. 3–13.

[6] D. Youla, H. Jabr, and J. Bongiorno, "Modern wiener-hopf design of optimal controllers–part ii: The multivariable case," *IEEE Transactions on Automatic Control*, vol. 21, no. 3, pp. 319–338, 1976.

[7] V. Kučera, "Stability of discrete linear feedback systems," 1975, p. 573–578.

[8] S. P. Boyd and C. H. Barratt, "Linear controller design : limits of performance," in *Englewood Cliffs, N.J.: Prentice Hall*, 1991.

[9] "An introduction to the conjugate gradient method without the agonizing pain." [Online]. Available: https://www.cs.cmu.edu/~quake-papers/painless-conjugate-gradient.pdf

[10] W. B. Powell, "Reinforcement learning and stochastic optimization: A unified framework for sequential decisions," March 15, 2022, chapter 6.

[11] M. Liu, W. Zhang, F. Orabona, and T. Yang, "Adam$^+$: A stochastic method with adaptive variance reduction," 11 2020.

[12] R. Ferrante, "A robust control approach for rocket landing," in *M.S. thesis, School of Informatics University of Edinburgh*, 2017, p. 78. [Online]. Available: https://project-archive.inf.ed.ac.uk/msc/20172139/msc_proj.pdf

[13] K. P. An. Packard and R. Horowitz, "Jacobian linearizations, equilibrium points," in *Dynamic Systems and Feedback*, 2002, p. 169–200.

[14] SpaceX, "Falcon 9," 2021. [Online]. Available: https://www.spacex.com/vehicles/falcon-9/

[15] SpaceX, "Falcon 9 user's guide," September 2021, p. 9.