Maneuvering Characteristics Control Systems using Discrete-Time MDPs

Madison Coots
Department of Computer Science
Stanford University
Stanford, CA USA
mcoots@stanford.edu

Tumisang Ramarea
Department of Management Science and Engineering
Stanford University
Stanford, CA USA
ramarea@stanford.edu

***Abstract*—An effective Maneuvering Characteristics Augmentation System (MCAS) will mitigate the risk of stalls during takeoff for Boeing 737 MAX 8 aircraft. In this paper, we develop a flight simulator for modeling the flight trajectory of an aircraft throughout takeoff. We then proceed to model the MCAS system aboard these aircraft as a discrete-time Markov Decision Process and subsequently compute the MDP's optimal policy. Using this optimal policy, the MCAS system is able to prevent 100% of the stalls that would have otherwise occurred in absence of the MCAS system's intervention.**

***Index Terms*—MCAS, Markov Decision Process, Value Iteration, Reinforcement Learning**

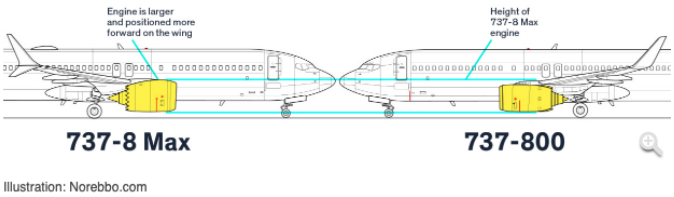

Fig. 1. Illustration of the difference in size of engine between the 737 MAX 8 and 737-800. Source: IEEE Spectrum

I. Introduction

A. Project Motivation

With over 10,000 Boeing 737 aircraft delivered and over 5,000 more on order since its birth in 1964 [14], this plane has earned its wings as a "faithful workhorse" for airlines around the world. Boeing has continued to work on improvements to the Boeing 737, partly inspired by competition from rival Airbus with their A320 aircraft family. These improvements are driven by the needs of airlines. As such the variants of the 737 have been centered around the needs of airlines for capacity, range, and fuel efficiency [17]. In 2010 Airbus launched the A320neo family of aircraft, which featured a new engine option on the original A319/A320/A321 frame with a 15-20% increase in fuel efficiency [14]. Boeing had to catch up or be left behind.

Boeing announced the 737 MAX, a re-engined variant of the 737, in 2011 [14] [15] [5]. The MAX variants used a new CFM LEAP-1B engine, which offers a 10-12% increase in fuel efficiency [15] over the classic CFM 56 engine used in the previous Boeing 737 series. Both engines are manufactured by CFM International, a leading supplier of jet engines for single-aisle aircraft. The launch of the Boeing 737 MAX and the Airbus A320neo reflect the advantage of improving an existing plane instead of introducing a brand new one: it is easier to launch [17]. A new aircraft requires a new type certification and for pilots to be completely retrained. The new CFM LEAP-1B engine is heavier than the CFM 56, and requires stronger engine struts, wings, fuselage, and landing gear [5]. Furthermore, the engine had to be moved forward and up to accommodate its larger fan diameter[1] [17].

For the Boeing 737 MAX to receive the same type certificate as the previous 737 aircraft, they had to have the same handling characteristics [17]. However, because of the change in size and position of the engine, the 737 MAX had different aerodynamic properties than its predecessors. Boeing tuned out any handling differences that resulted from the modifications to the design of the aircraft in the flight control systems so the aircraft types felt the same to crew. One of the additions to the flight control systems was the Maneuvering Characteristics Augmentation System (MCAS). This was added to counter the tendency of the new aircraft to pitch up when power was applied to the engine. MCAS is a flight control law that was designed and certified for the 737 MAX to enhance the pitch stability of the plane, so it could feel and fly like the other Boeing 737 aircraft [4]. The MCAS was implicated in two fatal crashes of the Boeing 737 MAX between October 2018 and March 2019, leading to the (ongoing) worldwide grounding of the aircraft type.

According to preliminary reports from the ongoing investigations into both crashes, it is believed that MCAS was activated due to erroneous angle-of-attack (AOA) data [11]. While this project does not model uncertainty surrounding the accuracy of readings from the angle of attack sensors, it proposes to model the MCAS system as a Markov Decision Process and compute an optimal policy for use during simulated flight.

[1] See Figure 7

B. Overview of MCAS

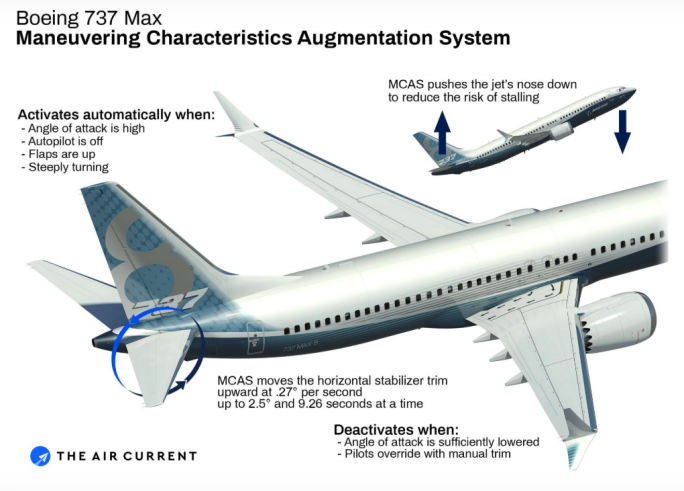

Fig. 2. Diagram showing how the MCAS works on the Boeing 737 MAX. Source: The Air Current

Given the new position of the engines in the Boeing 737 MAX, forward and up in comparison to previous models, the aircraft has a tendency to want to pitch upward [17]. Boeing introduced the MCAS as a fix to this. As shown in Figure 2, the MCAS system works by pitching the nose of the aircraft down when data from the AOA sensor warns that the aircraft is approaching an aerodynamic stall [5]. While stalls can occur at any airspeed, a low airspeed is often a sign of a developing stall [12]. The other sign is a high nose position, since stalls occur when the wing of an aircraft exceeds its AOA [12].

For most planes with stall warning devices, stalls are usually accompanied by a continuous stall warning to alert the pilots that they are approaching the critical AOA [2]. MCAS, on the contrary, is designed to reduce the AOA in the background without pilot knowledge and input [17]. It is designed to activate in manual flight, i.e. when autopilot is off [4]. It is activated when the AOA exceeds a threshold based on airspeed and altitude [5]. The stabilizer incremental commands are limited to 2.5 degrees and are provided at a rate of 0.27 degrees per second [4]. The function terminates either when the AOA falls below the critical AOA or if the flight crew overrides it. If the elevated AOA condition persists, MCAS commands another incremental stabilizer nose down command.

In the first accident in which MCAS is implicated, Lion Air Flight 610 (JT610), the preliminary report indicates that the MCAS was activated more than 24 times during the 11 minute flight [11]. According to the Federal Aviation Administration (FAA) guidelines on stall recoveries, some loss of altitude is expected when pitching the nose down as part of a stall recovery [2]. The preliminary report from the second accident, Ethiopian Airlines flight 302 (ET302), suggests pilots may fail to counter the MCAS [11]. This projects will need to take into account these insights in the design of the reward function for the Reinforcement Learning Algorithm it will use.

II. Related Work

Frank L. Lewis and Draguna Vrabie write an article, entitled "Reinforcement Learning and Adaptive Dynamic Programming for Feedback Control", in the IEEE Circuits and Systems Magazine with the aim to show the usefulness of RL techniques for the feedback control of human engineered systems [10]. Their article presents the main ideas and algorithms of RL, therefore it serves as a theoretical basis for our approach. Traditionally, flight control optimization has been solved using Calculus based methods such as Ordinary Differential Equations, H_∞ Control Method and Tuning Methods [1] [16]. However, the Mathematical complexity of these methods, especially without the linearity assumptions in the flight dynamics model, renders them less feasible in practice [6] [7].

The use of reinforcement learning in flight control optimization is gaining traction [10]. Recent examples include both the 2 degrees-of-freedom (DOF) and 3 DOF Helicopter problem, and a 6 DOF planetary powered descent and landing [7] [18] [8]. The Calculus based methods tend to be sensitive to the initial conditions, but RL sets up the problem as a Markov Decision Process (MDP) [10]. This means the concept of optimality from a given state, is independent of how that state is reached. The Markov assumption simplifies the complexity of the Mathematics and makes it easier to implement in a flight situation. Despite this simplification, a flight control problem still involves complex Mathematics as the states and time are both continuous. However, the complexity of the flight dynamics do not complicate the problem formulation since RL can be model-free [8].

Gaudet, Linares, and Furfaro, (hereafter referred to collectively as Gaudet), used deep reinforcement learning for the simultaneous optimization of both the guidance and control systems of a planetary lander (designed for Mars) [8]. They argue that the separate optimization of the two systems may reduce efficiency in one of the inputs, citing fuel as an example. Their aircraft model is defined in a 3 dimensional space. A 3 dimensional aircraft is the simplest they choose since their objective is to optimize for pin-point landing inside an oval plane with variable depth. They use their reward function to capture the relevant constraints. Specifically, they account for soft constraints by penalizing their violation with modest negative rewards. For hard constraints, they penalize their violation by terminating the episode and with a substantial negative reward. They fix the reference frame of their environment on the target landing spot. This makes this fixed position of the target a terminating state with positive reward. Hence an optimal action is to head directly towards the target.

Fandel, Birge, and Miah, (hereafter referred to collectively as Fandel), examine the use of Approximate Dynamic Programming to control a two degree-of-freedom helicopter [7]. Their approach is model based, and is proposed as an alternative to the Quanser AERO platform. The Quanser AERO platform is often used by researchers to develop control techniques for the helicopter problem. According to Fandel,

control techniques built through the platform involve linearization but most of them are not adaptive. They define their reward function to penalize the helicopter's error deviation from its desired state and also to penalize its control effort. Their study shows that ADP performs just as good as the LQR, one of the traditional methods for optimal control, with constant desired states. Moreover, ADP performs better for trajectories such as sinusoidal functions because it necessary to be more adaptive.

Xue, Li, and Yang, (hereafter referred to collectively as Xue), opt to use a model-free controller for the 3 DOF helicopter problem as it is trained only according to the control inputs and system states [18]. Specifically, they use the policy gradient method to train and optimize the controllers. They rely on a simplified helicopter model, from a study by Zheng and Zhong, that has led to "a large amount of research papers" [19]. Xue define their environment to include the elevation of the helicopter, angle position, and thrust vectors from the voltage input into both the front and back motor [18]. With these and other relevant variables, they define functions for the elevation and pitch motion of the helicopter. The objective is to use these as transition dynamics towards set target signals of elevation and pitch. Their reward function captures performance constraints and encourage moving towards the target elevation and pitch. Given that the Q-function used in Policy Iteration have shortcomings when applied to the continuous system problem, they use stochastic gradient descennt in the PG method[2] to maximize the sum of the reward. The controllers in their model are able to update by themselves using real-time data.

These 3 studies we have selected to discuss in depth in this section highlight the key considerations we would have to make in our modeling approach. An important consideration is on whether to model our MDP with discrete or continuous states and to use continuous or discrete time. For Mathematical simplicity, we choose to work in discrete time [10]. The paper by Kochenderfer, Holland, and Chryssanthacopoulus on the "Next-Generation Airborne Collision Avoidance System" presents a good example on how to discretize a flight state space [9]. The next consideration is to decide on the simplicity of our flight dynamics model. We draw inspiration from the work of Gaudet to model our flight dynamics model in a 2 dimensional space [8]. While Gaudet uses a 3 dimensional space, the nature of our problem allows us to simplify our model down by one dimension. The third consideration is whether to use model-free or model-based approaches. As we have seen from Fandel and Xue, both approaches make sense. Therefore, we opt to use ease of implementation in deciding our approach. We explain our modeling approach in detail in the next section.

[2]A method they get from a study Peters and Schaal on Reinforcement learning of motor skills with policy gradients.

III. Modeling Approach

A. Overview

A Markov Decision Process (MDP) models a system taking a sequence of actions under uncertainty to maximize its total reward. MDP's are comprised of a state space S, an action space A, a conditional probability function $T(s'|s,a)$ that models state transition dynamics, and a reward function that applies rewards to taking a certain action and transitioning into a certain successor state $R(s,a)$. The goal of the system is to determine the optimal sequence of actions that maximize the expected total reward:

$$\mathbb{E}\Big[\sum_{t=0}^{\infty}\gamma R(s_t,a_t)\Big]$$

where s_t and a_t denote the state and action taken at time t, respectively. $\gamma \in (0,1)$ is a discount factor used to encode preference for immediate rewards or past rewards. Applying a reinforcement learning algorithm to an MDP generates an optimal policy π that maps an optimal action $a \in A$ to every state $s \in S$.

Our model is concerned with the policy employed by the MCAS system aboard the 737 MAX 8 aircraft to control the aircraft's angle of attack. Given noiseless sensor input, the system attempts to the system's angle of attack from increasing to the point where the aircraft stalls during takeoff.

In our models, that state space and action spaces are discretized. We model the aircraft's position in space as continuous, but the state variables that play a role in the aircraft's transitions, angle of attack and vertical velocity, are discretized. The details on the flight dynamics model, state transitions, and rewards modeling are described below.

B. Flight Dynamics Modeling

We use a simplified flight dynamics model to model the aircraft's trajectory during the first phase of take-off. We focus our dynamics model on the take-off phase of the aircraft's flight because this is the phase wherein the MCAS system is most likely to activate due the aircraft's tendency to pitch up more than intended.

The flight dynamics model treats the aircraft as a point mass in 2-D space. Let the coordinates (x,h) denote the plane's horizontal position and altitude (vertical position) on a two-dimensional plane. We elect not to model the aircraft in three dimensions because the MCAS system is only concerned with the aircraft's angle of attack, which predominantly impacts the vertical lift experienced by the aircraft. We incorporate the aircraft's horizontal position into the model to allow us to visualize the aircraft's initial take-off trajectory. These modeling simplifications ultimately allow us to reduce the model of the aircraft's flight trajectory to two dimensions. Let u and v represent the aircraft's horizontal and vertical speed, respectively. Let α denote the aircraft's angle of attack. The flight state of the aircraft is specified as: (x,h,α,u,v). Using this flight state, we develop a basic flight simulator that uses

two-dimensional physics kinematics to simulate the aircraft's trajectory during takeoff.

The aircraft control consists of the aircraft's rate of change of the AOA, $\omega \in \{0, -\lambda\}$, where λ is the rate at which the MCAS system can change the angle of attack of the aircraft: $0.27°/s$ [5].

As discussed previously, the motivation for including the MCAS system on the aircraft was primarily to serve as a countermeasure against the aircraft's tendency to pitch upwards more than the regular 737 model. To model this tendency of the aircraft, we add a small uniformly distributed noise term $\epsilon_\alpha \sim U(0, 0.5)$ that is applied with probability p at every transition to make the aircraft's AOA at time $t+1$ non-deterministic. Let $\tau \in \{0, 1\}$ denote the outcome of whether or not this noise term is applied to the angle of attack, where $\tau \sim Bern(p)$.

Given (x, h, α, u, v) and choice of action $\omega \in \{0, -\lambda\}$, the new state of the aircraft in its takeoff trajectory after a small time duration Δt is given by:

$$
\begin{aligned}
h_{t+1} &= h_t + v_t \Delta t \\
\alpha_{t+1} &= \alpha_t + \omega \Delta t + \tau \epsilon_\alpha \\
u_{t+1} &= u_t \\
v_{t+1} &= v_t + f_v(\alpha) \Delta t
\end{aligned}
$$

$f_v(\alpha)$ is the vertical acceleration experienced by the aircraft as a function of α and is discussed in greater depth in the next section.

C. Stall Modeling

During flight, an aircraft will stall if the angle of attack is such that there is not sufficient airflow across the top of the wing. This relationship between α and the lift L experienced by the aircraft is defined by the following equation:

$$
L = \frac{1}{2} C_L \rho v^2 A
$$

where C_L is the lift coefficient of the aircraft, ρ is the density of the air, v is the aircraft's airspeed, and A is the area of the aircraft's wing. C_L is a function of alpha that is decreasing for values of α greater than the particular critical value for that aircraft. The function defining C_L as a function of α as well as the value of A for the 737 MAX 8 aircraft are not currently publicly available, making it challenging to create a high-fidelity model of how the airplane's lift decreases as α surpasses its critical value. Therefore, we developed an alternative representation of how the airplane's vertical velocity is affected as the aircraft's angle of attack increases beyond a critical threshold for α.

We define a critical AOA value, α_{crit} such that, for values of $\alpha \geq \alpha_{crit}$, the aircraft experiences an increase in acceleration in the downward direction. Therefore, the larger the difference between the α and α_{crit} when $\alpha \geq \alpha_{crit}$, the larger the downward acceleration. The function of α that defines the net acceleration the aircraft experiences is defined as follows:

$$
f_v(\alpha) = \begin{cases} 0 & 0 \leq \alpha \leq \alpha_{crit} \\ g \cdot (90 - \alpha_{crit})^{-\frac{1}{2}} \cdot (\alpha - \alpha_{crit})^{\frac{1}{2}} & \alpha > \alpha_{crit} \end{cases}
$$

where g is the acceleration due to gravity: -9.81 $\frac{m}{s^2}$. Therefore, when the nose of the aircraft is pointing straight up, because there is no airflow across the wings, it is modeled as being in free fall. In the model presented in this paper, a stall is defined as when the aircraft's vertical velocity drops below some threshold v_{crit} as a result of the negative acceleration it experiences due to the AOA being greater than α_{crit} for a sufficiently long period. We do not consider cases when the angle of attack is less then 0 degrees because AOA should not be less than 0 degrees during takeoff.

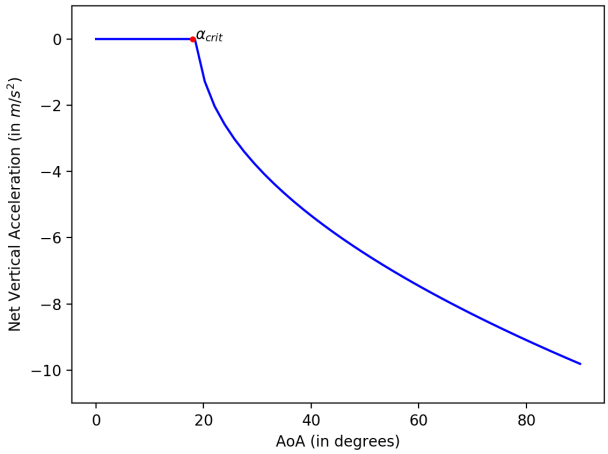

Fig. 3. Plot of the net vertical acceleration against AOA during cruise flight

D. Rewards Modeling

The reward function for this model, $R(s', a)$ assigns costs based on the value of the new state and the action taken to get there. The main considerations for assigning costs to states in the model are the value of α and whether or not the aircraft is in a stall. If the aircraft transitions into a state where it is in a stall, a very large negative reward is applied because it is now considered to be in failure state. Additionally, if α drops below a target value for take-off α_{target}, then a smaller negative reward is applied.

For actions, a small negative reward is applied to the action of decreasing α at a rate of $-\lambda$. This cost is intended to disincentive the MCAS system from too frequently lowering the nose of the aircraft.

E. Model Assumptions

This flight dynamics model makes some simplifying assumptions about the aircraft's movement through the air during take-off. In particular, the model assumes that the aircraft's horizontal velocity is constant throughout the flight. There are

no head winds experienced by the aircraft, and consequently the aircraft's horizontal ground speed are equal to its airspeed along the horizontal axis. Free of any negative acceleration experienced as a result of α exceeding its critical value, we assume that the aircraft has a constant vertical velocity or climb rate. According to estimates of the rates of climb and ground speeds of large commercial aircraft like the 737 MAX 8, we assume that the horizontal speed of the aircraft is roughly ten times of that of the rate of climb. Additionally, model input values for starting velocities were scaled down for simplicity and this scale is represented in the visuals in this paper.

This model also does not take into account uncertainty around the AOA sensors. While there is uncertainty around what the value of α will be in the next time step, the model does not represent uncertainty on whether or not the MCAS system knows the true value of α.

IV. Methods

We now briefly describe the methods used to solve the MCAS MDP.

A. State Space Reduction

The flight simulator developed to model the aircraft trajectory through space uses continuous values to represent the aircraft's current flight state. However, in order to simplify the computation of the optimal policy of the MCAS MDP, we discretized the values of α and v. Because the angle of attack α is ultimately the only factor in our model that affects vertical velocity v (which determines whether or not we are in a stall state), we are able to simplify the states used in learning the optimal policy to simply (α, v). We finely discretize the possible range of each state variable to creat a finite, discrete state space.

The optimal policy π computed through reinforcement learning thereby maps action $a \in A$ to states $s \in S$, where each state is a (α, v) tuple. Using the optimal policy π, we can simulate the aircraft's flight through the air with the continuous flight simulator by intermittently discretizing the continuous state in the simulator at each time step to get the corresponding discrete state s for which there is an optimal action in π.

B. Value Iteration

Value iteration is a common reinforcement learning method used to compute the value of an optimal MDP policy, as well as the optimal policy itself. Using a discount factor value of $\gamma = 0.95$, we used value iteration to compute the optimal policy for the MCAS MDP.

V. Results

A. Baseline

In order to effectively evaluate the performance of the optimal policy found through value iteration, we establish a baseline policy with which to compare the optimal policy. The baseline policy is $\pi(s) = 0 \ \forall s$; take no action to correct the aircraft's angle of attack at each time step. Simulating the use of this baseline policy over 10000 trials, we find that the success rate of the policy is about 30%, where a success is defined as completing take-off without stalling.

Algorithm 1: Value Iteration

$k \leftarrow 0$
$U_0(s) \leftarrow 0$ for all states s
repeat
  $U_{k+1}(s) \leftarrow \max_a[R(s,a) + \gamma \sum_{s'} T(s'|s,a)U_k(s')]$ for all states s
  $k \leftarrow k+1$
until; convergence
return U_k

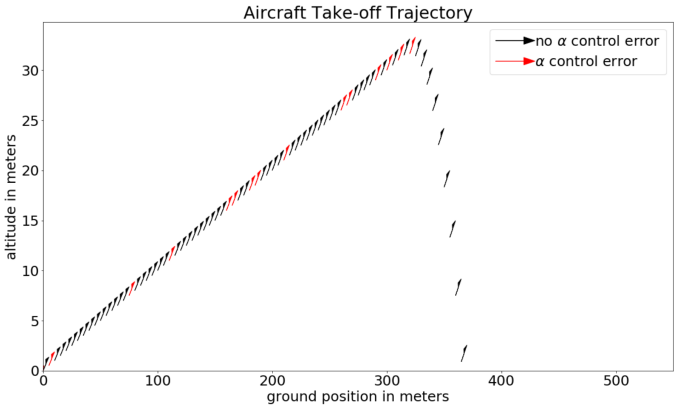

Fig. 4. Aircraft trajectory depicting stall occurrence using baseline policy. α control error refers to transitions wherein the aircraft unintentionally pitched upwards.

B. Optimal Policy

The optimal policy computed using value iteration had a success rate of 95%. This is indicative that the policy learned how to adequately correct for the stochastic tendency of the aircraft to pitch upwards during takeoff.

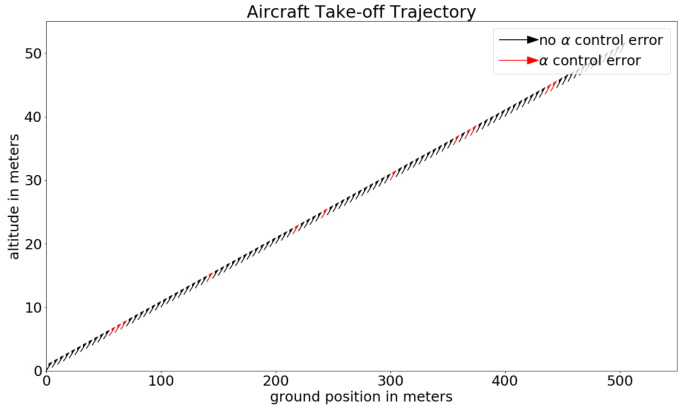

Fig. 5. Aircraft trajectory depicting successful takeoff using optimal policy. α control error refers to transitions wherein the aircraft unintentionally pitched upwards.

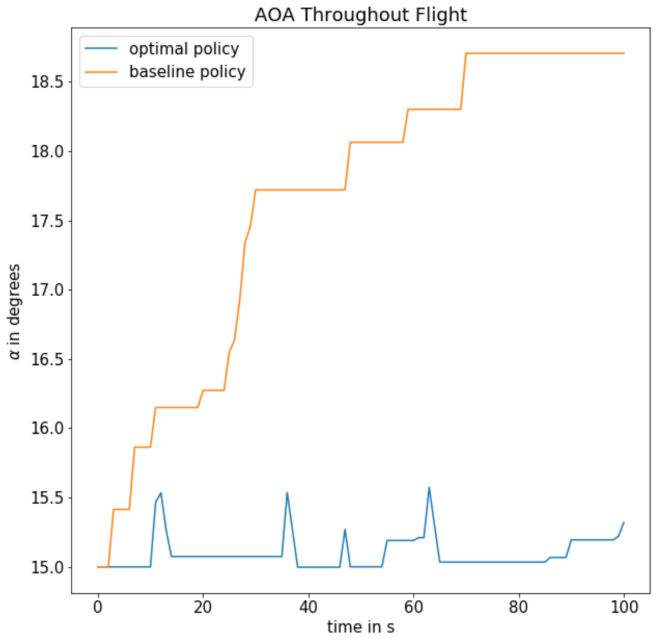

Fig. 6. Graph depicting how each policy controlled the angle of attack throughout the a given flight

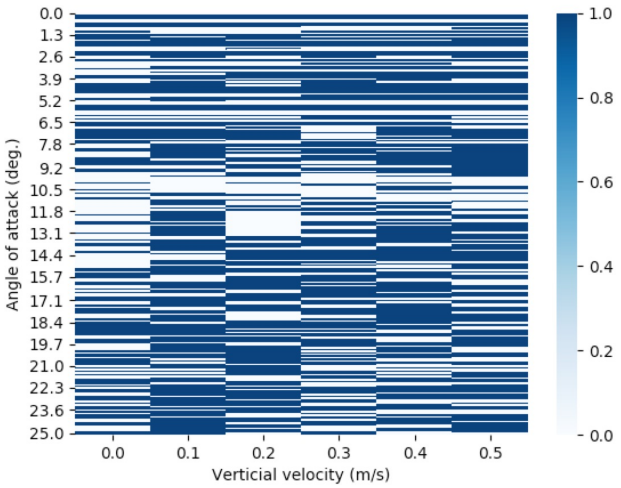

Fig. 7. Heat map depicting optimal policy. Dark squares correspond to states wherein the policy was to correct α at rate $-\lambda$

VI. Discussion

As depicted in Figure 6, it is evident that the optimal policy achieved better control around the angle of attack. The simulation had the the aircraft begin its takeoff using $\alpha = 15°$, and α_{crit} was set to equal $18°$. (These approximate values used in the simulation were recommended by industry experts interviewed for this project). While the baseline policy of taking no action allowed α to gradually increase and ultimately surpass the critical value used in the simulation, the optimal policy achieved greater control of the angle of attack, maintaining it at its target value of around $15°$.

Figure 7 is a heat map representation of the optimal policy learned through value iteration. Each square in the grid corresponds to a given state in the total state space. Intuitively, the policy takes more corrective actions as the velocity approaches the value of v_{crit} in the simulation, which was set to 0 m/s. The column corresponding to states wherein the $v = 0.1$ m/s is slightly more dense, which corresponds to their being more corrective actions for these states. Similarly, the lower half of the grid is also slightly more dense than the upper half, which corresponds to their being more corrective action when values of α are significantly above α_{crit}.

VII. Conclusion

A. Limitations

We acknowledge that numerous simplifying assumptions surrounding the physics and mechanics of flight that were made in order to make this project feasible in the time allowed. Inevitably, these assumptions will impact how reliably these results would translate into a real-world simulation.

B. Future Work

Future work can improve the fidelity of our flight dynamics model by relaxing the assumptions of this work. The assumptions we made in our modeling were to simplify the problem to fit the scope of the class project. Specifically, the assumptions we make about the different velocities of the aircraft being constant do not always hold. This is especially important since in the Ethiopian Airlines Boeing 737 MAX 8 crash, preliminary reports indicate that the velocity of the aircraft was out control right before the crash [11]. Therefore, velocity is an important variable to account for in the model. We can also improve the fidelity of our flight dynamics model by moving away from modeling the aircraft as a point mass, getting precise values for the aircraft's design parameters like its lift coefficient, and also better estimates of its target flight parameters like optimal angle of attack, airspeed, and so on.

We also recommend modeling the problem as a Partially Observable Markov Decision Process (POMDP) to account for the uncertainty in the values of the inputs of our model. In particular, given the controversy around the decision of Boeing to sell the capability for MCAS to rely on two AOA sensors instead of one as an add-on feature, it is essential to optimize the model for both configurations [13]. Furthermore, we recommend modeling the problem as a Continuous State POMDP using a similar approach to the work of Mykel J. Kochenderfer and colleagues entitled "Unmanned Aircraft Collision Avoidance using Continuous-State POMDPs" [3].

VIII. Special Thanks

The authors of this paper would like to extend special thanks to Ali Malik for his advice throughout the project. Additionally, the authors would also like to extend special thanks to Wayne Coots, Jay Juman, and Frank Bussey for sharing their expertise on the systems, avionics, and flight characteristics of Boeing aircraft.

IX. Contributions

Coming from outside the field of Aeronautics and Astronautics, we both started first by educating ourselves on the basics of flight dynamics. We also shared the responsibility of evaluating related work and formulating our problem as an MDP. Ramarea implemented the Flight Dynamics Model, while Madison implemented the Rewards Modeling, Stall Modeling, and the Value Iteration algorithm to solve the MDP. Since Madison is taking the class for 4 units, compared to Ramarea who is taking it for 3, Madison used her 30 hours of additional research to consult with Boeing aircraft experts throughout the project [3] at United Airlines, refine the MDP model, as well as the in-air flight kinematics.

References

[1] Richard J. Adams, James M. Buffington, and Siva S. Banda. Design of nonlinear control laws for high-angle-of-attack flight. *Journal of Guidance Control and Dynamics*, July 1994.

[2] Federal Aviation Administration. Maintaining aircraft control: Upset prevention and recovery training. In *Airplane Flying Handbook*, chapter 4, pages 4:1 – 4:24. Federal Aviation Administration, 2016.

[3] Haoyu Ba, David Hsu, Mykel J. Kochenderfer, and Wee Sun Lee. Next-generation airborne collision avoidance system. In *Decision Making Under Uncertainty: Theory and Application*. MIT Press, 2012.

[4] Boeing. 737 max updates.

[5] Chris Brady. Boeing 737 max - differences.

[6] Ximena Celia Mendez Cubillos and Luiz Carlos Gadelha de Souza. Using of h-infinity control method in attitude control system of rigid-flexible satellite. *Mathematical Problems in Engineering*, 2009, 2009.

[7] Andrew Fandel, Anthony Birge, and Suruz Miah. Development of reinforcement learning algorithm for 2-dof helicopter model. In *27th IEEE International Symposium on Industrial Electronics, ISIE 2018, Cairns, Australia, June 13-15, 2018*, pages 553–558. IEEE, 2018.

[8] Brian Gaudet, Richard Linares, and Roberto Furfaro. Deep reinforcement learning for six degree-of-freedom planetary powered descent and landing. *CoRR*, abs/1810.08719, 2018.

[9] Mykel J. Kochenderfer, Jessica E. Holland, and James P. Chryssanthacopoulos. Next-generation airborne collision avoidance system. *Lincoln Laboratory Journal*, 19, 2012.

[10] Frank L. Lewis and Draguna Vrabie. Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE Circuits and Systems Magazine*, 2009.

[11] House of Representatives. Status of the boeing 737 max: Stakeholder-perspectives, June 2019.

[12] Christopher L. Parker. Stall recovery: What you do and why you do it. *Flight Magazine*, March 2005.

[13] Rosie Perper. Boeing admits that it made a key alert system linked to faulty sensors optional on 737 max planes. *Business Insider*, May 2019.

[14] David Slotnick. Boeing 737 timeline: From the early days to the grounding of the 737 max after 2 fatal crashes that killed 346 people 5 months apart. *Business Insider*, Oct 2019.

[15] Ethan Steinberg. How to differentiate the various types of boeing 737s. *The Points Guy*, March 2019.

[16] G. Sudha and S. N. Deepa. Optimization for pid control parameters on pitch control of aircraft dynamics based on tuning methods. *Applied Mathematics & Information Sciences*, January 2016.

[17] Gregory Travis. How the boeing 737 max disaster looks to a software developer. *IEEE Spectrum*, April 2019.

[18] Shengri Xue, Zhan Li, and Liu Yang. Training a model-free reinforcement learning controller for a 3-degree-offreedom helicopter under multiple constraints. *Measurement and Control*, 52(7-8), 2019.

[19] B. Zheng and Y. Zhong. Robust attitude regulation of a 3-dof helicopter benchmark: Theory and experiments. *IEEE T Ind Electron*, 58(2), 2011.

[3] Acknowledged under the Special Thanks Section