Improving Single-Engine Aircraft Safety Through Evolutionary Neural Networks

Arkar Tan

Abstract

Engine failure in single-engine aircraft is an uncommon yet critically dangerous occurrence that forces pilots to maximize glide distance before finding an optimal landing zone. While modern avionics sometimes provide emergency features which offer guidance to pilots, they often rely on static values calculated for the plane found in handbooks that fail to account for differences in air density and other external factors. Using a Python-based simulator, I investigate two approaches, a PID controller and an Evolutionary Neural Network based AI controller and compare them to human trials in maximizing glide distance. Results indicate that, compared to humans, both AI and PID controllers offer high-efficiency solutions to increase glide distance by up to 63.8% and, hence, survivability in such emergency aviation scenarios.

1. Introduction

As emerging AI technologies break barriers in many industries, I explore how they can be applied to improve a real world situation involving single-engine aircraft. Single engine aircraft carry multiple advantages over multi-engine aircraft such as being more fuel efficient, less expensive to maintain, and cheaper to purchase. However, all single-engine aircraft have a dangerous flaw: in the case of an engine failure, the aircraft will be left without thrust, and the best hope of survival is for the human pilot to maximize glide distance and find a possible location for landing.

Although modern systems such as Garmin Smart Glide [8], Garmin Emergency Autoland [9], and the Cirrus Airframe Parachute System [10] can reduce pilot workload or improve survivability during certain emergencies, these systems primarily provide emergency guidance, autonomous landing, or parachute recovery rather than an adaptive control system designed to optimize glide distance in a thrustless single-engine aircraft [7,8,9,10].

To maximize flight distance, pilots are often told by the handbook or through the aforementioned guidance systems to obtain an optimal glide speed or optimal glide angle [2], and there are two major problems with this approach. First, there may not always be a single most optimal glide speed or glide angle during a thrustless descent due to varying air pressure, winds, and other external factors [3]. Second, a human attempting to maintain a constant angle or speed is imperfect due to human errors and limitation on adjustment granularity and frequencies and, hence, results in loss of energy and glide distance.

To solve each of these problems, I investigate two approaches, one using a traditional feedback-based PID control system and the other using an evolutionary neural network ("ENN") approach. The first, the PID controller, attempts to maintain a singular constant glide angle with significantly more accuracy than the human, solving the second problem but not the first. The second system, the ENN-based AI controller, is trained via a genetic algorithm to independently find any means to maximize distance, theoretically solving the first problem, and partially solving the second problem by having significantly greater accuracy than the human, but less than the PID controller. In the rest of this paper, I will discuss the physics model and simulator, the detailed design of each approach mentioned, the results and our conclusion.

2. Physics Model

To evaluate various approaches, I build a Python-based simulator with a multitude of considerations. First, I use a lot of specifications to make the plane similar to a Cessna 152 through online sources. I set the mass to 757kg [2], the wing area to 14.86m^2 [2], the mean aerodynamic chord estimated as 1.46m through calculation

$$c=\frac{S}{b}=\frac{160\,ft^2}{33.33\,ft}=4.8\,ft$$

the moment of inertia approximated as 1400kgm^2 [4], the induced drag factor estimated as 0.09, and the moment coefficient, coefficient of lift, and coefficient of drag at different angles approximated from the graphs available for the NACA 2412 airfoil [1], the airfoil of the primary wing in the Cessna 152.

Second, to make the system dynamic, the density of air varies continuously as a function of altitude. I calculate the density at a given altitude using the Barometric Formula in the U.S. Standard Atmosphere Model.

The formula for air pressure at a given altitude h in meters is given by

$$P=P_0\cdot\left(1-\frac{Lh}{T_0}\right)^{\frac{gM}{RL}}$$

where $P_0=101,325\,Pa$ as air pressure at sea level, $L=0.0065\,K/m$ as the temperature lapse rate, $T_0=288.15\,K$ as temperature at sea level, $g=-9.81\,m/s^2$ as gravity, $M=0.0289644\,kg/mol$ as molar mass of air, and $R=8.31477\,J/molK$ as the ideal gas constant.

Third, I run the physics simulator using two methods. One takes in the current elevator input as well as the 6 state variables, which are velocity in the x axis, velocity in the y axis, position in the x axis, position in the y axis, current pitch relative to the horizon, and rate of change of pitch angle. It then returns the derivative of all of the state values, calculated from the previous state values as well as the elevator input. I calculate lift and drag directionally proportional to the aircraft using common formulas (using the dynamic air density), and parasite drag is excluded for the purposes of the simulation. The formula for force of lift and drag respectively with the state variables as parameters are given by

$$F_L=\frac{1}{2}C_L\rho V^2 S$$

$$F_D=\frac{1}{2}C_D\rho V^2 S$$

where ρ is the dynamic air density calculated by the above function, the lift coefficient and S are aforementioned constants, and V is the velocity of the aircraft, calculated from the velocity of its x and y components.

I estimate a stall function using the NACA2412 tables, and lift would collapse and drag would massively grow when the pitch angle to the horizon exceeded 15 degrees. I then normalize the variables to earth-frame kinematics using trigonometry, and I divide the forces by mass to return accelerations. For torque, I multiply the elevator angle input by an elevator effectiveness coefficient and add it to the rest of the values in the moment coefficient to simulate using the elevator, so a modified formula for pitching moment is used, given by

$$M=\frac{1}{2}\left(C_M+C_E\epsilon\right)\rho V^2 Sc$$

where c is the mean aerodynamic chord, C_E is an arbitrary coefficient of elevator effectiveness, and ϵ is the current elevator angle. Although this does not exactly physically represent elevator input, the result of the equation is effectively the same for the results of the experiment, since the elevator input must have varying effects as a function of air density, velocity squared, wing area, and chord as well. I then divide the pitching moment by the estimated moment of inertia to return angular acceleration.

$$\alpha = \tau / I$$

In the second method, to update each state variable for each time step, dt, I chose the Runge-Kutta 4 (RK4) function. I prefer an RK4 method over Euler's method because of the significant advantage in accuracy over extended periods of time, and standard practice in similar simulators [5]. The RK4 method uses the flight dynamics method described in the previous paragraph and returns the updated state values, effectively performing a step over a period of customizable time, dt. The formula for the RK4 function is given by

$$X_{n+1} = X_n + \frac{dt}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right)$$

calculated from Simpson's Rule, where

$$k_1 = f\left(t, X_n\right)$$

representing the exact slope of each state variable at the start of the step,

$$k_2 = f\left(t + \frac{dt}{2}, X_n + \frac{k_1}{2}\right)$$

representing the estimated slope of each state variable at the midpoint of the step,

$$k_3 = f\left(t + \frac{dt}{2}, X_n + \frac{k_2}{2}\right)$$

representing a more refined estimate of the slope of each state variable at the midpoint of the step, and

$$k_4 = f\left(t + dt, X_n + k_3\right)$$

representing the estimated slope of each state variable at the end of the step, where X_n is a state variable, t is the current time, and f(a,b) is the respective function in the flight dynamics method.

The simulator is then run with parameters to sustain sufficient accuracy. Notably, dt is set to 0.01s, and the elevator input is a range between -10 degrees and 10 degrees. Although the actual Cessna 152 offers a range from -18 degrees to 25 degrees, this range is favored to simplify control and training for the human and the AI respectively.

3. Human Trials

The first set of trials were performed at a starting state of 1524m (5000ft) and a starting velocity of 61.73 m/s (120kts). These conditions were chosen as somewhat arbitrary values that maintained reasonable level flight and energy in testing. The human would have access to an ipywidget slider that would control the elevator from -10 degrees to 10 degrees. The simulator would then walk through 200 RK4 steps (2 seconds), while the human could change the elevator input during any step as the simulator ran, and update 5 graphs for the human to observe through matplotlib. These 5 graphs show position (displaying the path of the aircraft), velocity to time, pitch angle relative to horizon to time, angle of attack to time, and elevator input to time respectively.

Notably, it is hard for a human to control the simulator in finer-grained fashion since the human could realistically only receive and process new information every 2 seconds. As a result, with less fine tuned inputs, maintaining the optimal speed was significantly more difficult. The average distance among the ten trials was 12,545m and the greatest distance was 13,710m.

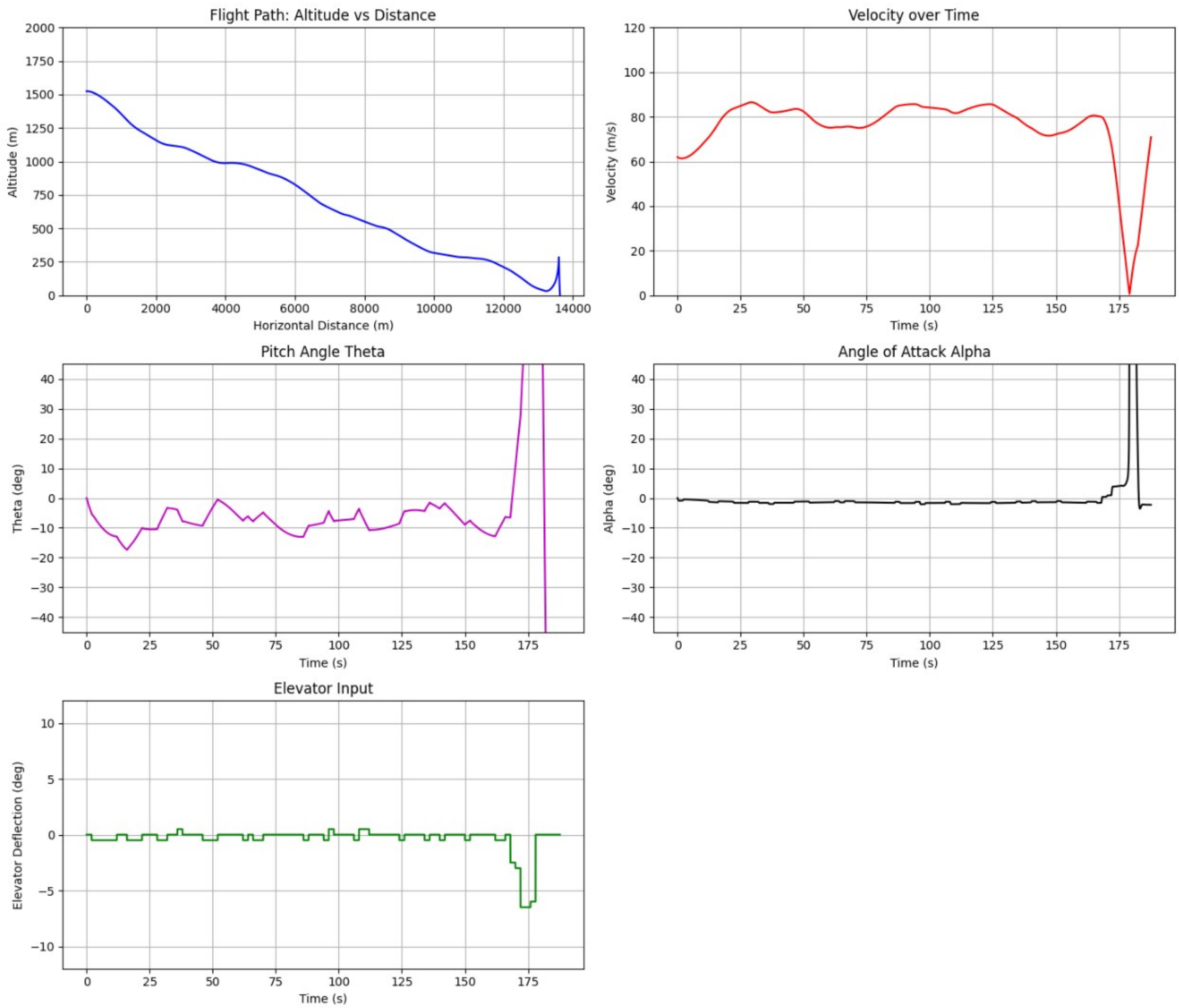

Figure 1: The charts showing key values of the best human trial.

4. PID Controller and Trials

The Proportional-Integral-Derivative (PID) controller is designed to maintain a constant target glide angle for the aircraft. It does this by using the standard PID formula [11] with coefficients chosen to optimize performance for our purpose, where the proportional term produces a directly proportional and antipodal input, the integral term softly accumulates past error to correct longstanding changes, and the derivative term reduces overshoot by predicting future outcomes.

The selected coefficients are chosen as 2.0 for proportional, 0.5 for integral, and 1.0 for derivative. The largest value is chosen for proportional, since the direct and instantaneous minute changes would be a large factor in maintaining a steady angle. Integral is the smallest chosen value, and notably smaller than the derivative value in order to avoid a phugoid oscillation, a common phenomenon that occurs when the PID controller overfits, especially in aerospace [6]. Hence, the formula for elevator input ϵ is approximately given by

$$\epsilon = -\left(2.0\, e(t) + 0.5 \int_0^t e(\tau)\, d\tau + 1.0 \frac{d\, e(t)}{dt}\right)$$

where the integral and derivative terms are calculated as

$$\int_0^t e(\tau)\,d\tau = I(t) = I(t-\Delta t) + e(t)\,\Delta t$$

$$\frac{de(t)}{dt} = \frac{e(t) - e(t-\Delta t)}{\Delta t}$$

where

$$e(t) = \theta_{target} - \theta$$

at time t, where θ is the pitch angle to the horizon.

With these values, the PID controller has the same initial state as the first set of the human trials at 1524 m and 61.73 m/s. Initially, the standard glide angle for the Cessna 152 is, calculated as:

$$\theta = -\arctan\left(\frac{1000\,ft}{1.6\,nmi \cdot \frac{6076\,ft}{1\,nmi}}\right) = -5.87\,deg$$

When I choose an estimated value of -6 degrees as the target angle to start, then the resulting distance is 17,101m. The graphs show that the PID is able to consistently hold the -6 degree target angle, besides a small adjustment period at the beginning of the simulation.

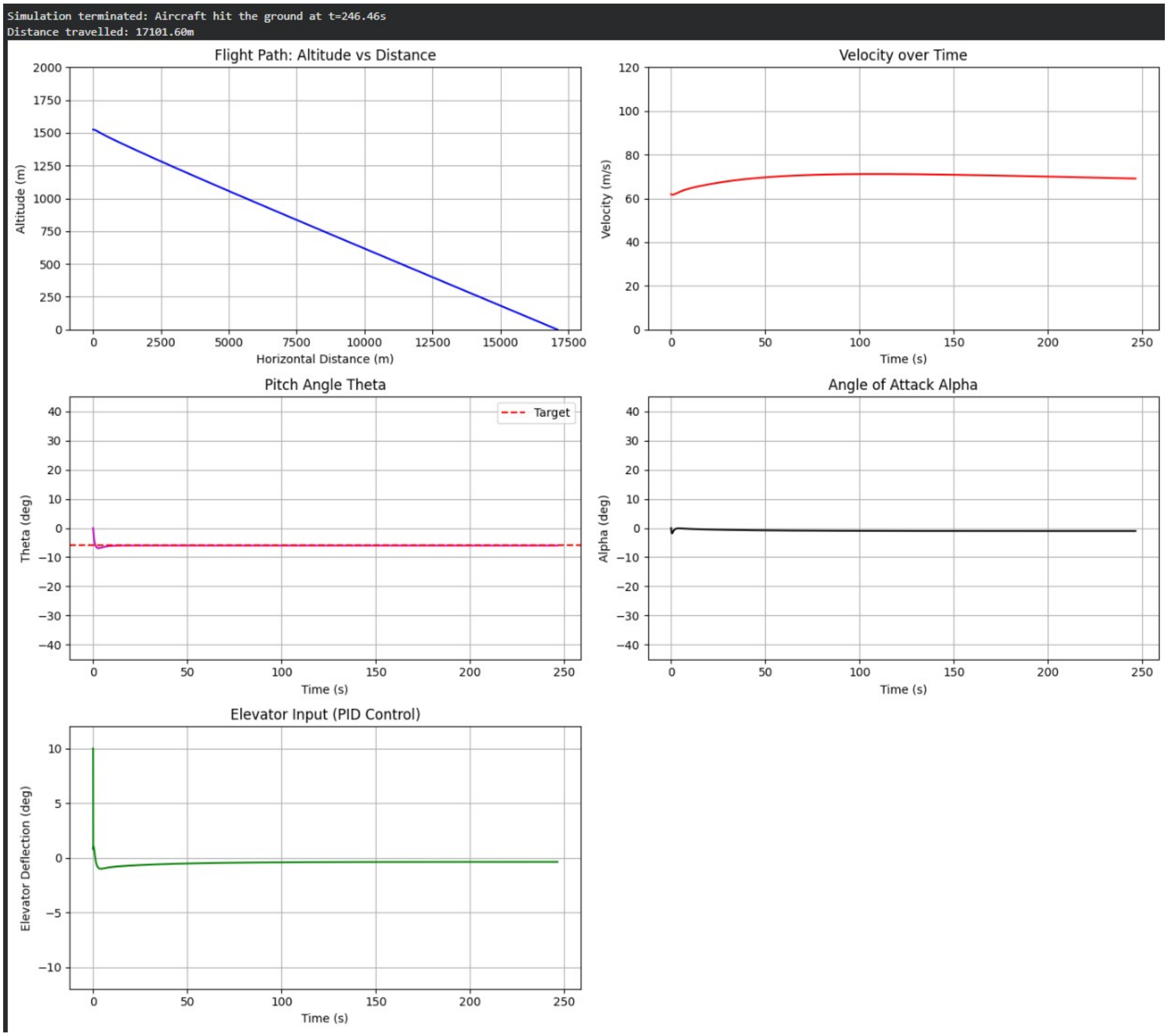

Figure 2: Key charts of the PID controller holding the -6 degree target angle.

The graphs in Figure 2 display a higher velocity than expected as well as a subzero angle of attack, suggesting that a more shallow target angle would very likely result in a further distance. At -3.5, -3.0, -2.5, and -2,0 degrees target angles, the distances that the aircraft travelled are 21,712m, 22,106m, 22,218m, and 22,051m respectively. At finer -2.6 degrees and -2.4 degrees target angles, the distances that the aircraft travelled are identical 22,218m and similar 22,206m respectively, strongly indicating that the optimal glide angle is around -2.5 degrees (see Figure 3), and that the optimal distance whilst holding a constant glide angle to the best of the PID controllers ability is 22,218m.

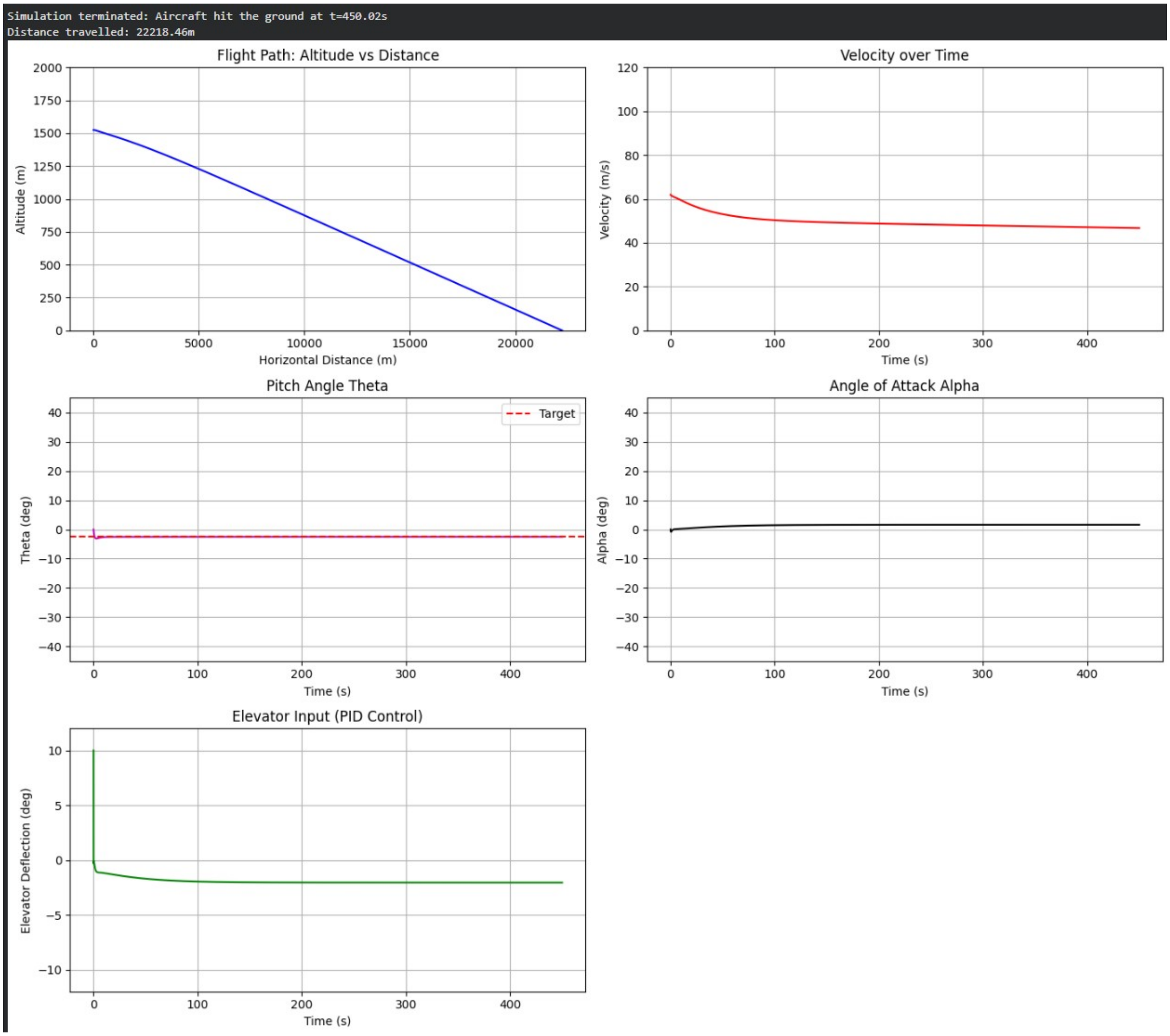

Figure 3: Key charts of the PID controller holding the -2.5 degree target angle.

5. AI Controller, Training, and Trials

I further considered multiple different AI models to find a near-optimal distance and I chose an evolutionary neural network approach (the "ENN Approach") to optimize the neural network's weights in finding a glide distance better than earlier approaches [12]. Our ENN approach uses all 6 state inputs and its update is governed by dt, which is set to 0.01 seconds for all experiments. In order to yield faster training, the 7 inputs are normalized first, given by the formulas,

$$W_1 = x/100000$$

$$W_2 = y/1524$$
$$W_3 = v_x/61.73$$
$$W_4 = v_y/61.73$$
$$W_5 = \theta/30$$
$$W_6 = q/30$$
$$W_7 = dt/0.01$$

where W_n is one of the node inputs and the state variables are horizontal distance, vertical distance, horizontal velocity, vertical velocity, pitch angle, angular velocity, and time step size, respectively.

After being normalized, the method takes the hyperbolic tangent of the weighted sum of each node input, given by,

$$L_{1_n} = \tanh\left(a_n W_1 + b_n W_2 + c_n W_3 + d_n W_4 + e_n W_5 + f_n W_6 + g_n W_7\right)$$

and passes it into the 16 nodes of the first hidden layer, a set of 16 nodes that exist solely for mathematical computation to be passed into the next layer, where every letter $a_n, \ldots, g_n$ represents a different weight in the neural network. The hyperbolic tangent is chosen as it is a very common tool in these neural networks for approximating curves, as it delinearizes the network, and could have been used interchangeably with the sigmoid function, another typically used function .

The process then repeats for the next 16 nodes of the second hidden layer, given by

$$L_{2_n} = \tanh\left(h_n L_{1_1} + i_n L_{1_2} + \cdots + w_n L_{1_{16}}\right)$$

where, once again, every letter $h_n, \ldots, w_n$ represents a different weight in the neural network.

The final step returns an elevator input based on the 16 previous inputs, taking the hyperbolic tangent of the weighted sum and multiplying the value by 10, to increase the range from [-1, 1] to [-10, 10] to match the elevator ranges of the other control methods. This is given by the formula

$$\epsilon = 10 \cdot \tanh\left(x_n L_{2_1} + y_n L_{2_2} + \cdots + z_n L_{2_{16}}\right)$$

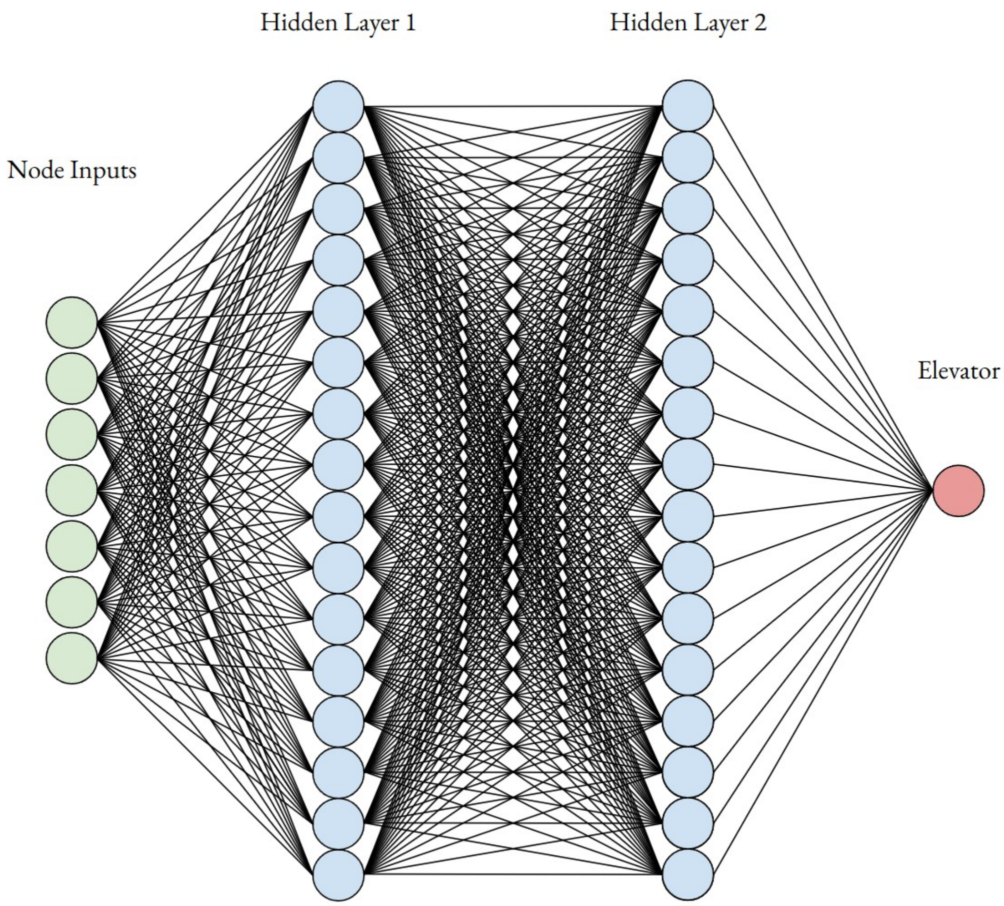

Figure 4: A graphical display of our two-layer 16-node evolutionary neural network.

Since I seek to maximize the distance travelled, I make the reward function in a one way training with a genetic algorithm. This genetic algorithm approach is favored over traditional gradient-based methods because the goal is to optimize and find an excellent control system instead of matching an already known output dataset, which I showed in the previous section that a PID controller can do well.

For the first generation, I generate 7 neural networks with random weights, and each of them controls the plane at the starting state at 1524 m and 61.73 m/s until hitting the ground. Out of the 7 networks, only the network with the greatest distance travelled saves its weights and moves onto the next generation, and it leads to 6 new neural networks using the saved weights with modifications. Specifically, using a parameter σ, which effectively served as a mutation rate, the other 6 networks are generated by going through each weight in the parent's network and randomly modifying them by an arbitrary amount from $-\sigma$ to σ.

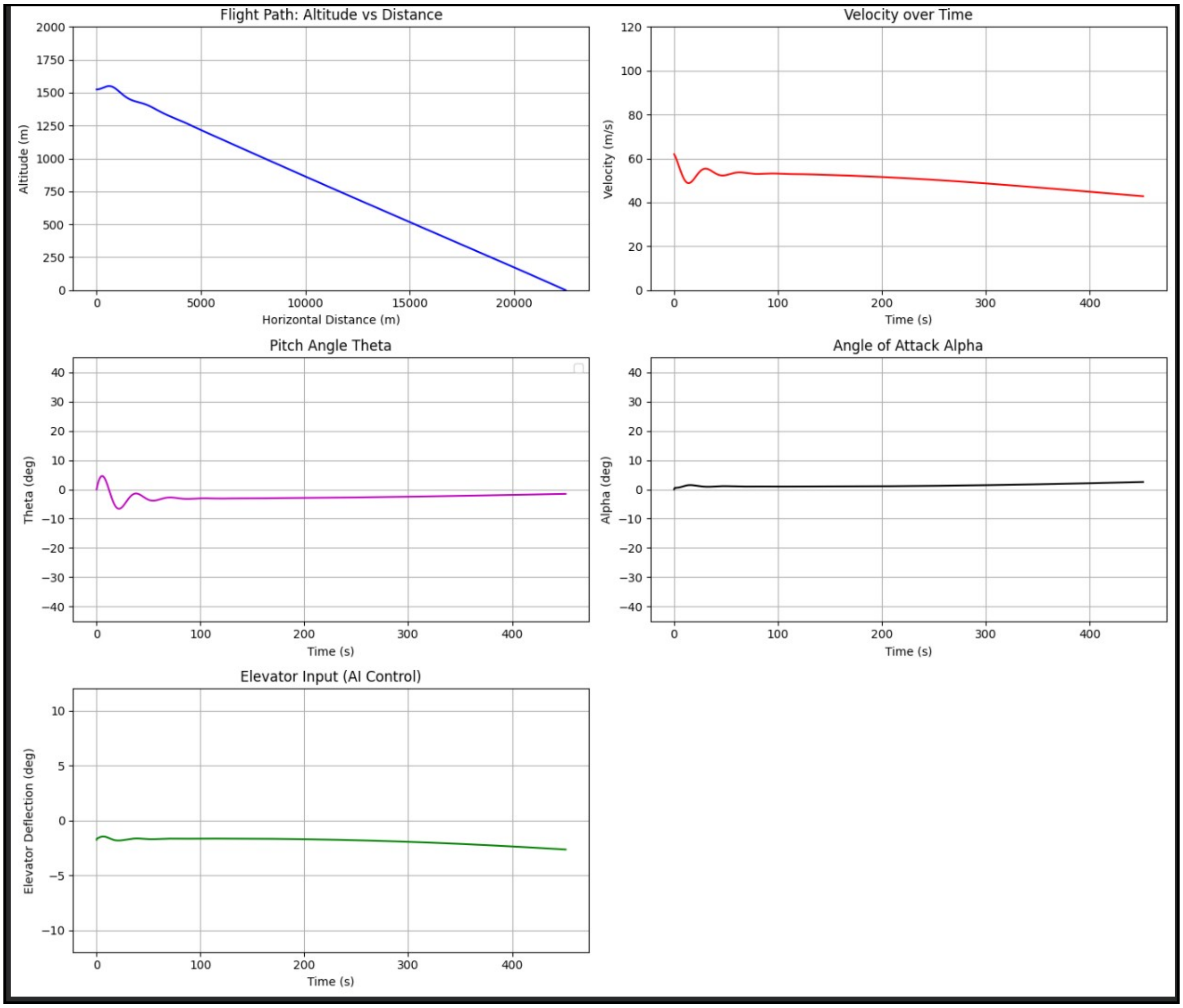

Figure 5: Key charts of the AI trained by the genetic algorithm after 300 generations.

In our experiments, our ENN approach is trained for 300 generations with different random initial weights. The best distances traveled for the 0th, 10th and 100th generations are 12,603m, 21,453m, 22,369m, 22,398m respectively. After 300 generations, the best distance traveled is 22,463m, marginally greater than the best result of the PID controller.

```
Gen  298 | best reward: 22462.7 m
Gen  299 | best reward: 22462.7 m
Training complete. Final reward: 22462.7 m
Simulation terminated: Aircraft hit the ground at t=451.55s
```

6. Analysis, Areas of Improvement, and Conclusions

The human trial results have the lowest average distance (12,545m) and lowest peak distance (13,710m). This is primarily due to the human inability to stably maintain the proper pitch angle through only the elevator input. Beyond the simulator and for future research, I conjecture that a trained pilot using a yoke instead of a widget slider to control the elevator would have significantly higher distances, but would not exceed the PID controller or AI controller.

The PID controller results are notably robust and successful, achieving a peak distance of 22,218m at a target angle of -2.5 degrees. Due to the low computation required to calculate and adjust error, the PID controller would be excellent in real-time for implementation into aircraft. The PID controller could potentially be improved with different values K_P, K_I, and K_D, making the initial adjustment more energy efficient, and possibly increasing the distance. For future research, the PID controller may be more efficient with a different system that accommodates multiple glide angles at different sections of air density, or possibly a PID controller that follows a mechanically calculated ideal glide angle. I conjecture that such a sophisticated PID, while computationally more expensive, would significantly boost the performance of the PID.

The ENN controller's results are the best, with its peak distance slightly higher than the PID controller's, at 22,463m. In Figure 5, I observe two key differences in the graphs of the AI model that may have given it a better performance. First, the ENN controller chooses to initially go up for a small period of time, maximizing altitude and potential energy before the final descent. Second, observing the pitch angle to time graph, I notice that the ENN controller chooses a slightly curved descent path, contributing to additional final distance travelled. For roughly a 1.1% marginal advantage over the PID, the ENN controller requires 300 computationally expensive generations of training; however, after training, the weights of the evolutionary neural network can be frozen, implemented into an aircraft, and placed into any scenario, with the only computational expense being hyperbolic tangents and arithmetic, similar to the computational cost of the PID controller. For future research, the AI could be trained for a significantly longer amount of time, with a lower value of σ in order to fine-tune the evolutionary neural network further.

Controller	**Furthest distance achieved (m)**
Human	13,710
PID	22,218
ENN	22,463

Figure 6: Table summarizing results of controllers

In these simulations, I observe the distances exceeding a 14:1 glide ratio for a Cessna 152, significantly higher than the predicted 9:1 glide ratio observed in reality [2]. The primary reason for this discrepancy is our choice for simplicity to ignore parasite drag, which allows the aircraft to retain significantly more energy than a real world scenario. Further research could implement parasite drag by scaling to match the results found in the Cessna Aircraft Company handbook or by implementing a complete polar curve.

Moreover, in order to reduce discrepancies with reality, further research could incorporate wind vectors, turbulence, updrafts, and many other external events which I would expect to benefit the ENN controller significantly more than the PID controller, since the ENN approach has a significant advantage in finding complex non-linear solutions. I also plan to improve our simulator and release it to those who are interested for further research.

Overall, the findings suggest that PID controllers and ENN controllers offer significantly more efficient descents in maximizing distance travelled compared to the human controller. Most importantly, the PID and ENN controllers both outperform the human by up to 62.1% and 63.8% respectively. For emergency engine failures, the PID controller and pre-trained neural network both offer low computation, high efficiency solutions that, if implemented into aircraft, could significantly increase the survivability of such aviation incidents by removing human error and fatigue in these high-stress scenarios.

7. Acknowledgements

I thank Kyle Kennedy, my mentor, for providing valuable insights, discussions, and resources that strongly improved the depth and clarity of this research.

References

[1] Airfoil Tools. “Polars for NACA 2412 (naca2412-il).” *Airfoil Tools*, http://airfoiltools.com/airfoil/details?airfoil=naca2412-il#polars. Accessed 13 April 2026.
[2] Cessna Aircraft Company. *Pilot’s Operating Handbook Cessna 152*. 1978. Long Island Aviators, Cessna Aircraft Company, https://longislandaviators.com/wp-content/uploads/2018/08/1978-Pilots-Operating-Handbook-Cessna-152.pdf. Accessed 12 April 2026.
[3] Federal Aviation Administration. *Glider Handbook*. 2018, https://www.faa.gov/regulations_policies/handbooks_manuals/aviation/glider_handbook/gfh_chapter_5.pdf. Accessed 3 June 2026.
[4] Krawczyk, Zack. “Evaluating Reduced-Order Urban Wind Models for Simulating Flight Dynamics of Advanced Aerial Mobility Aircraft.” *Aerospace* 2024, p. 13. MDPI, https://www.mdpi.com/2226-4310/11/10/830. Accessed 13 April 2026.
[5] Shah, Kamal, et al., editors. *Qualitative and Computational Aspects of Dynamical Systems*. IntechOpen, 2023. Accessed 3 June 2026.
[6] Suda, Gunasekar, and S. N. Deepa. “Optimization for PID Control Parameters on Pitch Control of Aircraft Dynamics Based on Tuning Methods.” *Applied Mathematics & Information Sciences*, 2016. Accessed 3 June 2026.
[7] Garmin International. “Garmin Continues to Enhance Aviation Safety with the Introduction of Smart Glide.” *Garmin Newsroom*, 21 July 2021, https://www.garmin.com/en-US/newsroom/press-release/aviation/garmin-continues-to-enhance-aviation-safety-with-the-introduction-of-smart-glide/. Accessed 4 June 2026.
[8] Garmin International. “Smart Glide.” *Garmin Autonomí*, Garmin, https://www.garmin.com/en-US/autonomi/smart-glide/. Accessed 4 June 2026.
[9] Garmin International. “Garmin Autoland Achieves FAA Certification for General Aviation Aircraft.” *Garmin Blog*, 18 May 2020, https://www.garmin.com/en-US/blog/aviation/garmin-autoland-achieves-faa-certification-for-general-aviation-aircraft/. Accessed 4 June 2026.
[10] Cirrus Aircraft. “CAPS Syllabus - Pilot Edition.” *Cirrus Aircraft*, March 2013, https://cirrusaircraft.com/wp-content/uploads/2014/12/CAPS_Syllabus_Pilot_Edition.pdf. Accessed 4 June 2026.
[11] Åström, Karl J., and Tore Hägglund. *PID Controllers: Theory, Design, and Tuning*. 2nd ed., Instrument Society of America, 1995.
[12] Yao, Xin. “Evolving Artificial Neural Networks.” *Proceedings of the IEEE*, vol. 87, no. 9, Sept. 1999, pp. 1423-1447, https://sci2s.ugr.es/keel/pdf/algorithm/articulo/yao1999.pdf. Accessed 4 June 2026.