A Model Predictive Control Based Autopilot for Aircraft

Luke Keating
Dept. of Mechanical, Manufacturing & Biomedical Engineering
School of Engineering, Trinity College Dublin
Dublin, Ireland
keatinl1@tcd.ie

Dermot Geraghty
Dept. of Mechanical, Manufacturing & Biomedical Engineering
School of Engineering, Trinity College Dublin
Dublin, Ireland

***Abstract*—This paper examines the use of model predictive control as an autopilot (for aircraft) that can follow prescribed flight trajectories. Aircraft dynamics were researched and it was found that the multiple inputs of the aircraft (elevator, aileron, rudder); affected each of the multiple outputs of the system (latitude, altitude, longitude) and therefore could not be coupled separately. Given this, multiple single-input single-output controllers would not suffice as the controllers would hinder each other's efforts to control their set variables. Therefore a multi-input multi-output (MIMO) controller which could produce actuations that considered all objectives was needed. Model predictive control (MPC) is a widely used MIMO controller in many engineering fields e.g. autonomous driving and industrial process. The controller was researched and was found to be a good fit for the application. A realistic flight trajectory was made using FlightGear flight simulator and was then exported. MPC was then implemented to control a model of the aircraft in a Simulink simulation to follow the trajectory from FlightGear. The conclusions found were that MPC was a viable autopilot strategy for aircraft if the trajectory was not too aggressive and small delays in timing were allowed.**

Keywords—MPC, autopilot, autonomous, control, aircraft

I. Introduction

More than 70% of aviation accidents can be attributed to human factors [1]. Fatigue is a pivotal human factor that can lead to poor decision making which can prove fatal in an aviation context. In 2006, a study conducted by Jackson and Earl found that in a sample of 162 pilots who flew regularly into their discretion hours, 75% were severely fatigued [2] which is a troubling thought when the number of lives that depend on pilots performing their job to the best of their abilities is considered. To limit the fatigue pilots experience there are aids available to them, autopilots are one of these aids. Autopilots can improve the accuracy and reliability of flying operations, and reduce the pilots' workload [3]. This reduction in workload could translate to a reduction in stress and mental fatigue placed on the pilot. It could be said therefore that developing an autopilot that is even more capable, leaves even less for the pilot to do and in theory, would further reduce fatigue and lead to safer flying.

There are three types of autopilot, single-axis, two-axis and three-axis. Single-axis are known as wing levellers and control just the roll of the aircraft for stability. Two-axis control pitch and roll, this means that the aircraft can climb and descend while staying stable autonomously. Finally, three-axis autopilots add yaw control which allows the aircraft to move laterally. Therefore, the autopilot can fly in any direction and perform any manoeuvre autonomously [4]. This leaves less for the pilot to do and leads to safer flying as mentioned.

To achieve the three axis control and to provide optimal control a multi-input multi-output (MIMO) controller was needed. Model predictive control (MPC) was chosen as it provides optimal control for multiple inputs.

The objective of the study was now: To implement a model predictive controller to control the multiple inputs and multiple outputs of a 6-DOF aircraft so it can follow a given trajectory in 3-dimensional space.

II. Methodology

A. Reference trajectory

The reference trajectory is what the user wants the system to do. The task of creating or sourcing a reference trajectory was difficult as flight data such as this is often classified. In the flight data that was found online for commercial aircraft, the time step between data points was too large, seconds at a time. The model would not predict how the system would behave accurately that far into the future.

Therefore, a different approach was needed in sourcing the reference trajectory. Flight simulators are used often in industry [5] and much time and effort is spent by teams of engineers ensuring they are as true to real-life as possible. It was decided that extracting flight data from a simulator would be an appropriate way to verify the simulation given the faith that companies and labs have in them, and the amount of engineering effort that goes into their development. FlightGear is a free to download simulator [6] and is used by universities such as the University of Naples, Italy [7] and labs such as the Institute for Scientific Research, Virginia [8], in their research of autonomous aircraft and simulations of models. Conveniently, FlightGear also has a way to export position data at a time step of the user's choosing to a CSV file. See figure 1 and 2 for the reference trajectory generated in FlightGear.

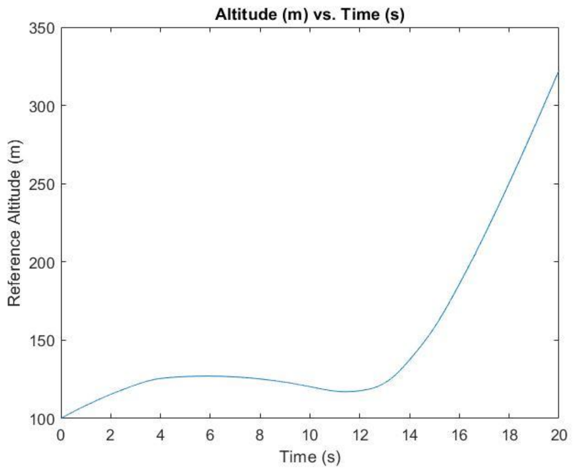

Fig. 1. Altitude trajectory

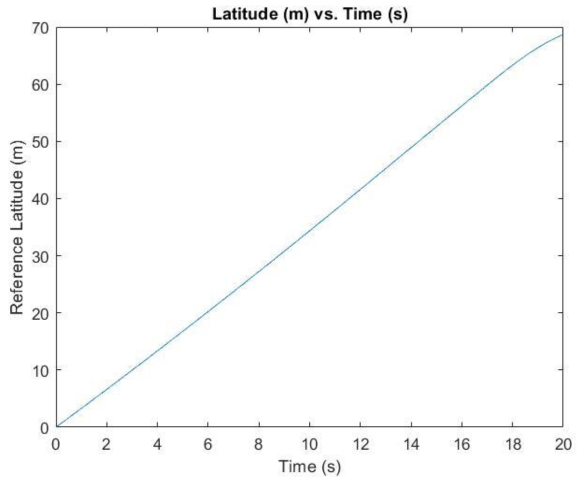

Fig. 2. Lateral trajectory

B. *State-Space*

The first step of designing the plant was to determine the continuous time state space. The equations for how the rudder, elevators and ailerons are given in figure 3. The partial derivative of each of these was taken with respect to each of the states to get the A matrix and with respect to each of the inputs to get the B matrix. Shown in (1) is how the state space should be organised. C is an identity matrix and it extracts the state as the current state, D is a matrix of zeros as the system does not respond instantaneously to the input u(t).

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t) \qquad (1)$$

$$m\dot{u} - \mathring{X}_u u - \mathring{X}_{\dot{w}}\dot{w} - \mathring{X}_w w - (\mathring{X}_q - mW_e)q + mg\theta cos\theta_e = \mathring{X}_\eta \eta \mathring{X}_\tau \tau$$
$$-\mathring{Z}_u u + (m - \mathring{Z}_{\dot{w}})\dot{w} - -\mathring{Z}_w w - (-\mathring{Z}_q + mU_e)q + mg\theta sin\theta_e = \mathring{Z}_\eta \eta + \mathring{Z}_\tau \tau$$
$$-\mathring{M}_u u - \mathring{M}_{\dot{u}}\dot{u} - \mathring{M}_w w + I_y \dot{q} - \mathring{M}_q q = \mathring{M}_\eta \eta + \mathring{M}_\tau \tau$$
$$\dot{\theta} = q$$

$$m\dot{v} - \mathring{Y}_v v - p\mathring{Y}_p - (\mathring{Y}_r - mU_e)r - mg\phi = \mathring{Y}_\xi \xi + \mathring{Y}_\zeta \zeta$$
$$-\mathring{L}_v v + I_x \dot{p} - \mathring{L}_p p - I_{xz}\dot{r} - \mathring{L}_r r = \mathring{L}_\xi \xi + \mathring{L}_\zeta \zeta$$
$$-\mathring{N}_v v + I_x z\dot{p} - \mathring{N}_p p - I_z \dot{r} - \mathring{N}_r r = \mathring{N}_\xi \xi + \mathring{N}_\zeta \zeta$$
$$\dot{\phi} = p$$
$$\dot{\psi} = r$$

Fig. 3. Equations of orientation

When the partial derivatives were taken, some assumptions were made about the state of the aircraft in order to change the state space from a matrix containing partial differential equations to a numeric matrix. Both the matrix and assumptions were taken from Cook's book [9]. The matrix is now a numerical matrix which represents the equations of motion in continuous time. The state-space then needed to be discretised in order to simulate it being controlled by a discrete-time controller. Zero-order hold is the method which was used to discretise the dynamics. Equation (2) shows the equations for the zero-order hold method.

$$Ad = \exp(AT)$$
$$Bd = \int \exp(A\eta)\, d\eta B \qquad (2)$$
$$Cd = C$$
$$Dd = D$$

C. *Flight Dynamics*

When the state space produced the new orientation, how the velocities were affected by this was calculated using the equations in (3). The calculated change in velocity was added to the constant velocity of the aircraft.

$$\dot{u} = X/m - g \cdot \sin(\theta) + r \cdot v - q \cdot w$$
$$\dot{v} = Y/m - g \cdot \sin(\phi) \cdot \cos(\theta) - r \cdot u + q \cdot w \qquad (3)$$
$$\dot{w} = Z/m - g \cdot \cos(\phi) \cdot \cos(\theta) - q \cdot u - p \cdot v$$

When the new velocities were found they could then be transformed to the inertial frame from the body frame. This was done using the rotation matrix shown in figure 4.

$$R(\phi,\theta,\psi) =$$
$$\begin{bmatrix} cos\theta cos\psi & sin\phi sin\theta cos\psi - cos\psi sin\psi & cos\psi sin\theta cos\psi + sin\phi sin\psi \\ cos\theta sin\psi & sin\phi sin\theta sin\psi + cos\phi cos\psi & cos\psi sin\theta sin\psi - sin\phi cos\psi \\ -sin\theta & sin\phi cos\theta & cos\phi cos\theta \end{bmatrix}$$

Fig. 4. Rotation matrix

Finally when the inertial velocities were found, they could be integrated with respect to time in order to find the new inertial positions.

D. *Model Predictive Controller*

As the simulation was done in Simulink, the MPC toolbox was used. An MPC block was loaded into the simulation and the reference and state space blocks were connected, see figure 5. The number of controlled outputs needed to be defined, this was the number of reference trajectories supplied which was 2, the number of inputs to the system also needed to be defined, this was 3 and they were the elevator, aileron and rudder angles. Once these steps were completed the toolbox would linearise and the step responses of the system were shown to see if the system was stable. Also, the measured outputs (mo) enter the controller, as per MPC being a feedback controller. The MPC block takes care of the cost function and optimiser for the user.

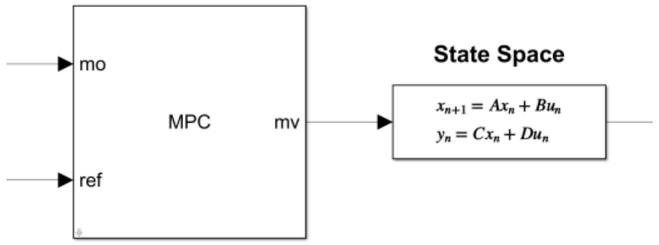

Fig. 5. MPC block and state space block

III. Results

A. Altitude

The difference between the reference and the output was calculated to show the performance of the controller. Figure 6 shows the difference between the reference and the output at any point in the simulation. Figure 7 shows a comparison of the reference altitude and the altitude the simulation produced. The mean percentage error (MPE) for altitude was -2.5119%.

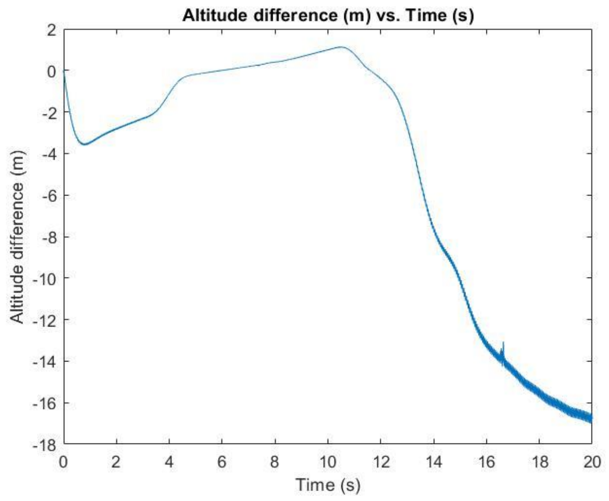

Fig. 6. Difference between the reference and the output altitude

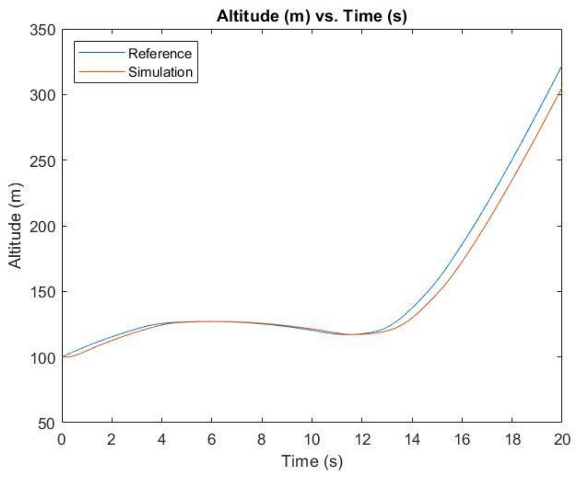

Fig. 7. Plot of reference and simulation output altitude

B. Lateral Position

Again the differences between the latitudes were plotted vs. time. Figure 8 shows the difference between the two trajectories at any point in the simulation. Figure 9 shows a comparison of the reference altitude and the altitude the simulation produced. The MPE for latitude was -2.1702%.

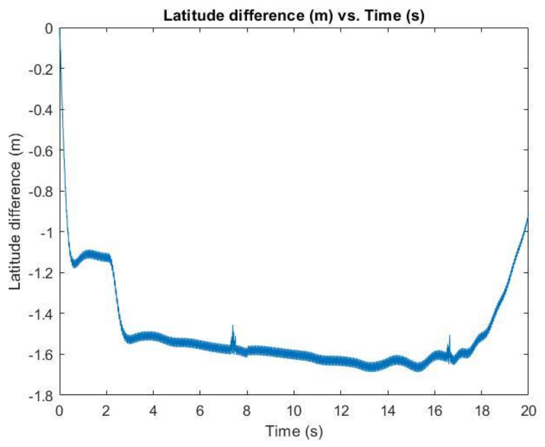

Fig. 8. Difference between the reference and the output latitude

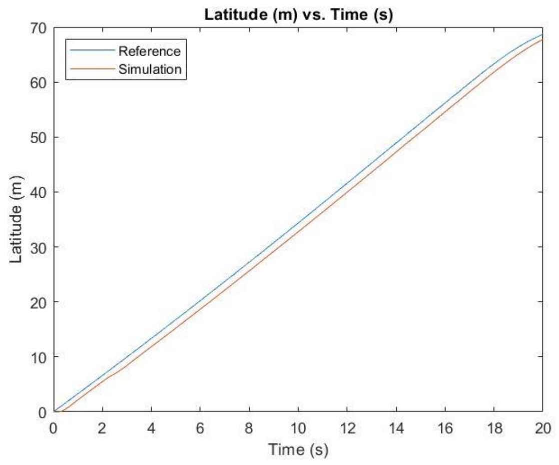

Fig. 9. Plot of reference and simulation output latitude

C. *Altitude with noise*

To test the robustness of the controller, noise was introduced to the readings of position. The reference was kept the same as before and a noise level of 0.01m was introduced every 0.1 seconds. A level of 0.01m was decided upon as many GPS have accuracy to the level of 1cm [10]. Figure 10 shows the difference between the reference and the output altitude when noise is introduced into the position readings. Figure 11 is a comparison of the desired altitude and the output altitude. The MPE for altitude with noise was 2.58%.

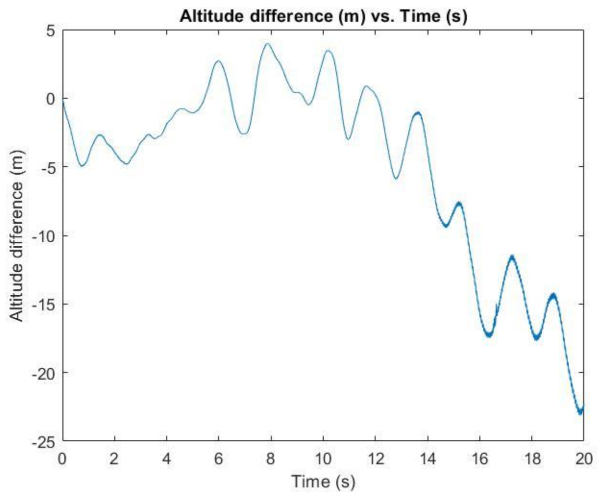

Fig. 10. Difference between the reference and the output altitude (with noise)

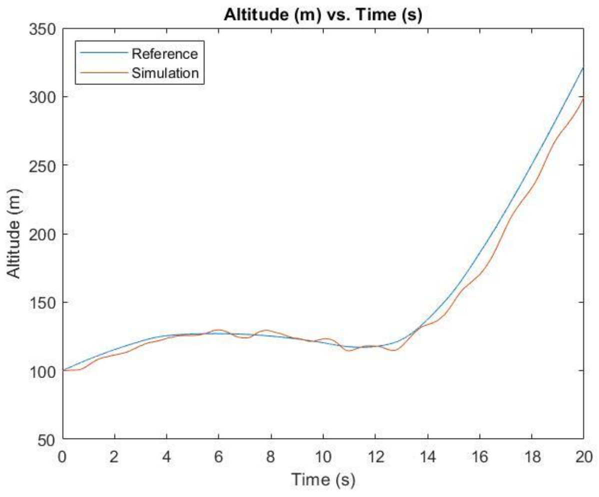

Fig. 11. Plot of reference and simulation output altitude (with noise)

D. *Lateral Position with noise*

The differences between the latitudes were plotted vs. time this time with noise introduced. Figure 12 shows the difference between the two trajectories at any point in the simulation. Figure 13 shows the comparison of desired latitude and the latitude produced by the simulation when noise was introduced. The MPE for latitude with noise was -1.06%.

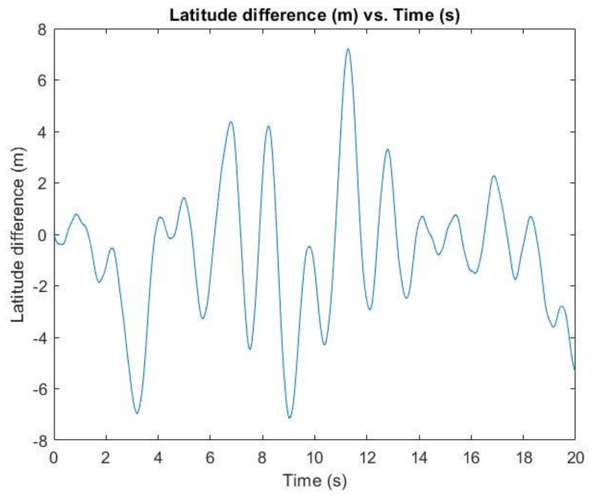

Fig. 12. Difference between the reference and the output latitude (with noise)

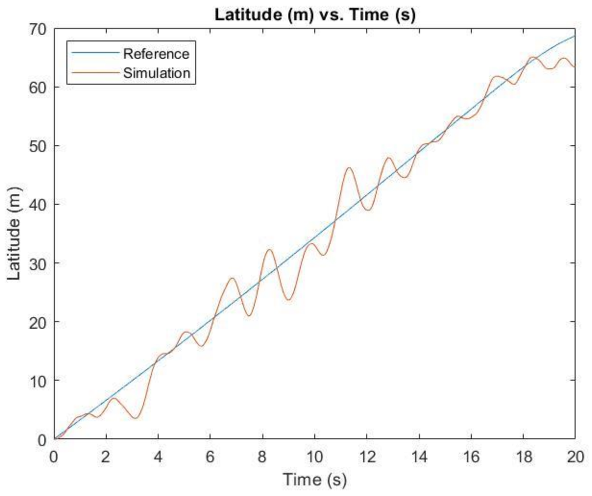

Fig. 13. Plot of reference and simulation output latitude (with noise)

IV. Discussion

The trajectory had a gentle, almost linear lateral movement with a non-linear ascent. It was seen that the aircraft was able to follow the trajectories comfortably with minimal errors as seen in figures 7 and 9. This shows the controller was capable of controlling the model to follow a realistic trajectory well. Therefore, the controller could in theory control the aircraft which was modelled in real life also.

Figure 6 showed the difference in trajectory and the simulation altitude. There was minimal deviation from the goal trajectory. It was seen that as the controller followed the reference and as the non-constant slope of altitude became steeper, the controller began to lag behind where it should have been. A possible cause of this was an overly conservative constraint on the inputs, the maximum and minimum flap angles assumed were 20° or 0.349 radians as this data was unavailable online. Also, what this could mean is that the controller struggles to follow aggressive bouts of ascent. Figure 8 showed the difference in trajectory and the simulation latitude. There was also minimal deviation here and the controller almost caught up to the reference by the end of the manoeuvre.

When noise is introduced into the position readings the autopilot is still capable of following the trajectory with a similar level of accuracy (roughly 2.5%). However it can still be seen that the controller lags when the rate of ascent begins to rise. In the case of latitude, it can be seen that noise improved the output accuracy (-2.17% to -1.06%). This is because the controller was forced to overshoot the desired trajectory as a result of believing to be off course. This balanced the position errors more, rather than being below the desired consistently, the aircraft was both above and below, leading to a more accurate result.

The results found in this study were in line with the results of other studies. It can be seen in the paper by Emami and Banazadeh [11] that MPC when used in a fast-moving application like an aircraft, the controller lags the reference with its outputs. This shows the study conducted in this project has similar limitations to results obtained by published researchers. The results of this paper were compared to the results of their robust MPC, as the adaptive MPC that was also studied in their paper has the advantage of an adaptive state space meaning its predictions for inputs are more accurate. It was seen in their paper that as the trajectory became more aggressive, the controller began to lag behind the trajectory, just like in this study. This further verifies that the results of this study are comparable to results of similar work in the past. This may prove that the simulation in this study is a good representation of how the real aircraft would behave when controlled by an MPC.

V. Conclusion

To conclude, this report found MPC to be a viable control strategy for an aircraft, provided that the lateral movements were not overly aggressive. When literature is inspected online, it can be said that there is a significant lack of research into using MPC as an autopilot. The importance of this research was discussed in the introduction section. To reiterate, the use of individual SISO controllers to control different actuators, while simple to implement is not optimal and so a MIMO controller such as the one used in this study is preferable. This strategy relieves the pilot of more stress and fatigue leading to safer flying while optimally controlling the aircraft.

A. Findings

• Model predictive control is a viable autopilot strategy for aircraft.

• The controller lag begins to increase when the rate of ascent is too aggressive.

• The controller always lags behind the reference by a small time step.

• The controller is robust enough to follow the trajectory accurately when noise is introduced to position readings.

References

[1] Seungyoung Lee and Jin Ki Kim. Factors contributing to the risk of airline pilot fatigue. Journal of Air Transport Management, 67:197–207, 2018. doi: https://doi.org/10.1016/ j.jairtraman.2017.12.009.

[2] Craig A Jackson and Laurie Earl. Prevalence of fatigue among commercial pilots. Occupational medicine, 56(4):263–268, 2006. doi: https://doi.org/10.1093/occmed/kql021.

[3] Xue Chengqi, Qiu Cen, and Zhang Yan. Design and research of human-computer in teraction interface in autopilot system of aircrafts. In 2009 IEEE 10th International Conference on Computer-Aided Industrial Design Conceptual Design, volume 10, pages 1498–1501, 2009. doi: 10.1109/CAIDCD.2009.5374997.

[4] Marc E. Cook. Autopilot basics, 2017. URL https://www.aopa.org/ training-and-safety/students/crosscountry/special/autopilot-basics. Accessed: 2022–03-15.

[5] Robert T. Hays, John W. Jacobs, Carolyn Prince, and Eduardo Salas. Flight simulator training effectiveness: A meta-analysis. Military Psychology, 4(2):63–74, 1992. doi: https://doi.org/10.1207/s15327876mp0402 1.

[6] David Murr. Flightgear - download central, 1996. URL https://www.flightgear.org/ download/.

[7] Domenico Coiro, Agostino De Marco, and Fabrizio Nicolosi. A 6dof flight simulation environment for general aviation aircraft with control loading reproduction. AIAA Modeling and Simulation Technologies Conference and Exhibit, page 1, 2007. doi: 10.2514/6.2007-6364.

[8] Eric Sorton and Sonny Hammaker. Simulated flight testing of an autonomous unmanned aerial vehicle using flightgear. In Infotech@ Aerospace, page 1. Institute for Scientific Research, 2005. doi: https://doi.org/10.2514/6.2005-7083.

[9] Michael V. Cook. Flight Dynamics Principles. Elsevier Ltd., 2 edition, 2007.

[10] Michael G Wing, Aaron Eklund, and Loren D Kellogg. Consumer-grade global positioning system (gps) accuracy and reliability. Journal of forestry, 103(4):169, 2005

[11] Martina Mammarella and Elisa Capello. A robust mpc-based autopilot for mini uavs. In 2018 International Conference on Unmanned Aircraft Systems (ICUAS), volume 2018, pages 1227–1235, 2018. doi: 10.1109/ICUAS.2018.8453290.