Aircraft Trajectory Generation Using Conditional Generative Adversarial Networks

Abstract

This paper proposes a conditional generative adversarial network model which learns the distributions of trajectory data for the purpose of generating similar trajectories. The physics of aircraft are defined using airspace variables representing the dynamic behaviour of an aircraft. Proposed detect and avoid systems are difficult to test and train in the real world due to excessive costs, and thus, software simulations have been used as a baseline for theoretical testing. In this project, it is found that the conditional generative adversarial network architecture provides statistically significant improvements to a baseline generative adversarial network model in certain turning airspace variables. Additional statistically insignificant benefits to the remaining variables are seen except for the accuracy of the aircraft speed in the generated trajectories.

1.0 Introduction

Urban air mobility (UAM), or the introduction of novel aircraft systems such as remotely piloted aircraft systems (RPAS, or drones) to existing airspace present new challenges for engineers [1]. With stringent government regulations generally requiring aircraft systems to limit serious failures or accidents to under 1/100,000 to 1/10,000,000 [2], significant validation is necessary prior to UAM implementation. One proposed risk minimization involves designing detect and avoid (DAA) systems intended to identify intruder aircraft and take corrective flight maneuvers to avoid collision.

As testing these systems using real aircraft is prohibitively expensive, software simulations have been used to quantify the likelihood of collision. Recent studies in detect and avoid (DAA) rely upon simulated intruder aircraft [3] with software simulations providing a theoretical test for the design of these systems. However, this testing is limited by both the accuracy of simulated intruders and the quantity of data which is available. An approach in software simulation is to generate synthetic paths approximating the nature of how aircraft fly and placing them in test cases with potential collision risks. Therefore, the generation of aircraft trajectories, which are realistic and simulate flight in urban areas, could provide a large enough synthetic data sample necessary to reach the safety requirements required by government regulators.

1.1 Aircraft Trajectories

Aircraft trajectories in urban areas occur in correlated airspace, or areas controlled by air traffic control. These flight patterns often consist of maneuvers which are standardized to avoid potential collisions. These trajectories can be represented using a cartesian coordinate system which consists of three arbitrary dimensions. As a convention, the positive X direction represents a northerly direction, the positive Y direction represents an easterly direction, and a positive Z direction represents a positive altitude (or an east, north, up convention). In this paper, all calculations and distances of the aircraft are done relative to an arbitrary inertial origin representing the starting point of the aircraft. Therefore, no consideration has been made for air-relative or aerodynamic effects and wind effects.

Additionally, to represent the aircraft physics, a simplification is made using three airspace variables as depicted in Figure 1. The aircraft turn rate (or heading) quantifies the rate at which an aircraft turns in a non-vertical direction. The speed of an aircraft is measured by comparing the distance between two observations in three dimensional cartesian space (although shown in Figure 1 for 2D). Lastly, the pitch rate of an aircraft measures the rate of change in the angle of ascent or descent in the aircraft.

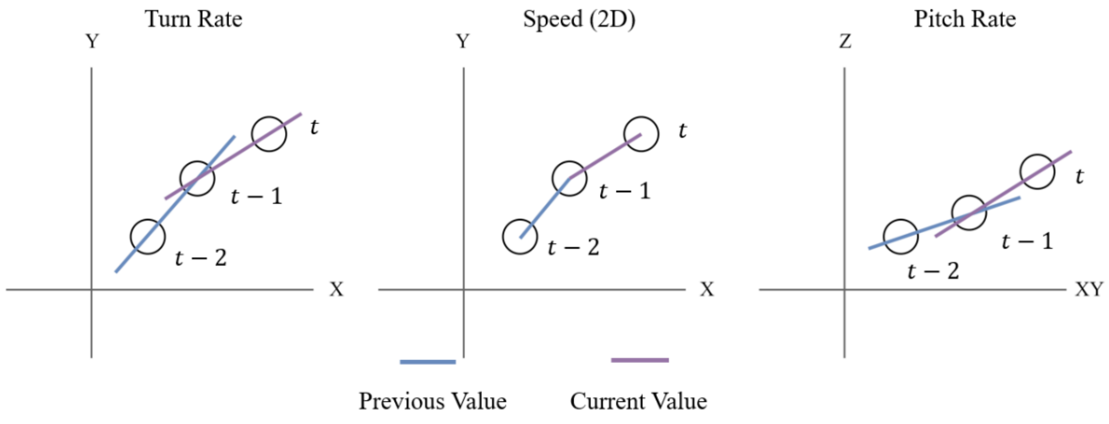


Figure 1: Airspace variables and calculation methods.

These airspace variables when in certain combinations, typically indicate a specific type of behaviour. For example, aircraft on takeoff and cruise generally have few variations in inertial pitch because steady behaviour is required for these flight patterns. These variables can either be constant, have minimal changes, or be variable during each stage of flight. For further reference, refer to Table 1 for a generalized summary of the aircraft patterns and variable trends discussed above.

Table 1: Aircraft patterns and Variable Trends

Aircraft Pattern	Airspace Variables			
	Vertical Rate	Turn Rate	Pitch Rate	Speed
Landing	Constant	Variable	None	Variable
Takeoff	Variable	Minimal	Variable	Variable
Cruise	None	Minimal	None	None

Accurately representing the maneuvers involved in different stages of flight is crucial for developing a robust DAA model, as well as building a comprehensive understanding of aircraft trajectories. Therefore, it is essential to adopt an approach that provides sufficient variety and accuracy in generating these trajectories.

1.2 Generative Adversarial Training

To generate trajectories that meet the necessary specifications, a generative adversarial network (GAN) architecture, along with an encoder, has been proposed in multiple papers. GAN training for aircraft trajectories was proposed by [4] where the generator and discriminator together aid in producing new trajectories and the encoder serves as an anomaly detector. A limitation of existing traditional non-machine learning approaches and the regular GAN method is the lack of long-term temporal dependencies [5]. This effect is minimized by [6] who proposed a multi-layer encoder and decoder long-short term memory (LSTM) GAN by using high-dimensional meteorological data and flight plans. This deep generative model can predict multi-dimensional trajectories over a significantly larger geographic area, spanning across the continental United States. [7] implements an attention and LSTM based model to predict 4D aircraft trajectories trained using ADS-B (Automatic Dependent Surveillance Broadcast) data assisted by a series of crucial preprocessing steps for a significant improvement in accuracy. However, the effectiveness of these approaches has only been tested on a limited range of data, with trajectories being restricted to either a single airport or airport pair.

1.3 Aims and Goals

In addition to generating physically accurate aircraft trajectories for DAA models using generative machine learning architectures, the aim of this paper is to create a more generalizable model applicable to a broad range of urban areas. Therefore, the generation of aircraft trajectories that are applicable to urban air flights have various constraints which can simplify the problem. Only trajectories under a pressure altitude of under 15,000 m above sea level (ASL) are considered. Additionally, rotary-wing aircrafts such as helicopters or gyrocopters are ignored in the dataset as they have significantly different flight characteristics. Finally, all abnormal trajectories resulting from emergencies, anomalous weather, and/or mechanic failures will be filtered out to avoid overfitting to outliers and insufficient data.

Furthermore, it is essential to account for the variation in aircraft, and to address this, the International Civil Aviation Organization (CAO) has proposed a Wake Turbulence Category (WTC) system [8]. The wake turbulence class was and can act as a simple categorization scheme of the approximate behaviour of aircraft. The WTC is based upon the takeoff weight of aircraft with aircraft classified as light being under 7,000 kg (15,500 lb), medium being between 7,000 kg and 136,000 kg (15,500 lb to 300,000 lb), and heavy exceeding 130,000 kg (300,000 lb). Aircraft representative of each WTC are summarized in Figure 2.,

a) Cessna 172: Light WTC. Figure from: [8] **b) Airbus A220: Medium WTC Figure from: [9]** **c) Boeing 747: Heavy WTC Figure from: [10]**

Figure 2: Wake turbulence category of aircraft

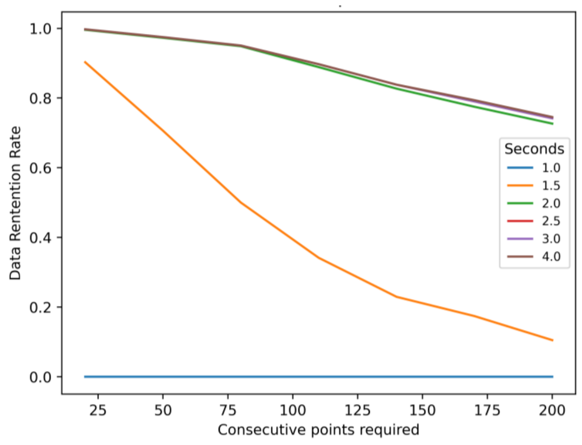


Figure 3: Data rejection with consecutive time points.

To achieve these goals, a conditional generative adversarial network (cGAN) model is proposed with the wake turbulence class (WTC) as a label to be used in training. It is argued that weight differences in aircraft have a significant impact on aircraft behaviors and maneuvers in urban areas, and therefore, provide significant prior information that is beneficial to the generative problem. It is shown this model generates aircraft trajectories with greater accuracy with respect to physics informed methods including the turn rate, speed, and pitch rate as summarized in Figure 1 and that some of these improvements are statistically significant relative to a baseline GAN architecture.

2.0 Methods

In this section, we summarize the preprocessing methodology, followed by the machine learning architecture proposed for the generator and discriminator, followed a description of the validation methodology.

2.1 Data and Preprocessing Methodology

Utilized data was obtained from the OpenSky database (opensky-network.org), a crowdsourced multi-year collection of ground-based ADS-B data. Around 320 random days in 2020 and 2021 of aircraft trajectory information are sampled for nine airports: 1) Montreal (YUL), 2) Ottawa (YOW), 3) Vancouver (YVR), 4) Victoria (YYJ), 5) Toronto (YYZ), 6) Billy Bishop (YTZ), 7) Calgary (YYC), 8) Quebec City (YQB). 72,643,485 individual observations which are suitable and approximately 1 second apart are extracted from the server. Utilized data consists of longitude, latitude, pressure altitude, an ICAO (International Civil Aviation Organization) identifier, and aircraft type of each observation.

While high in quantity, this data has two primary issues: 1) no unique identifier is available to explicitly determine which observations belong to which aircraft, 2) following the removal of invalid observations, differences in two-time observations can exceed 1 second. The first issue is addressed by the creation of fake unique identifier which is formed by concatenating the ICAO identifier, aircraft type, and finding observations which are in the same temporal period. The latter issue is addressed by removing all trajectories with two observations exceeding a certain threshold.

This is done to limit the need to create custom processing or smoothing algorithms such as Kalman filters. The filter limits the maximum time difference between two observations with the total amount of data insufficient based on the applied tolerance summarized in Figure 3,

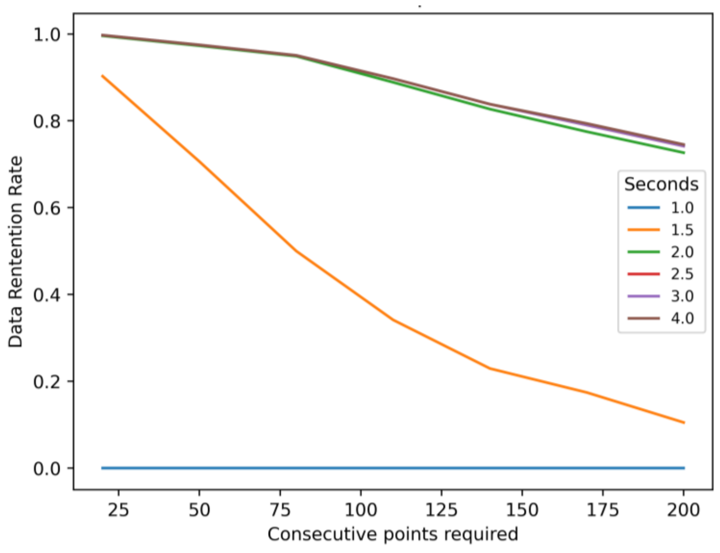


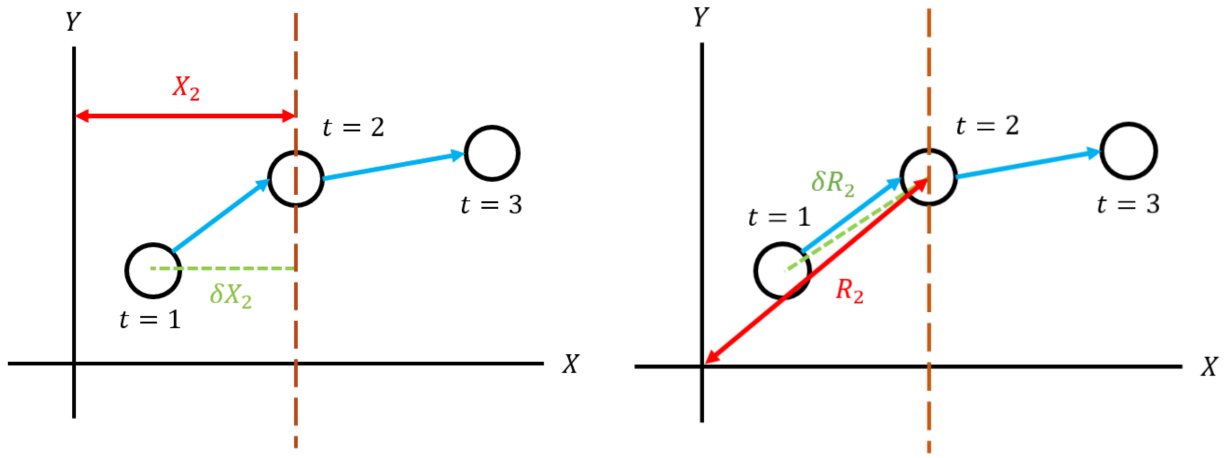
Figure 3: Data rejection with consecutive time points.

When decreasing the tolerance for the time interval between two consecutive observations, the number of available trajectories decreases. This decrease occurs at a significant rate when attempting to limit the maximum time difference to more than 2 seconds. To limit queries of the OpenSky database and improve the speed of data collection, trajectories with observations which are a maximum of 2 seconds apart are accepted. However, it is noted by the authors that most time observations are close to 1 second intervals with noise causing only occasionally causing delayed or skipped observations. Following this processing, 158,992 viable trajectories are generated with 200 consecutive points.

As the prediction of aircraft trajectories is intended to be generalized, a cartesian coordinate system which is normalized to the origin of the flight trajectory is used. This ensures equivariance of aircraft trajectories by ignoring the original point in which they are starting from. To convert latitude and longitude into values in a cartesian coordinate system, the Universal Transverse Mercator (UTM) coordinate system [11] is used. UTM coordinates approximate meters and approximate the distortions of the latitude and longitude due to the curvature of the earth.

Further preprocessing is done to convert these trajectories into a problem that is more easily solved by the GAN architecture. Data normalization is considered by converting the X, Y, and Z coordinates into deltas, or the difference between two points. Therefore, instead of predicting the value relative to the origin, each point predicts the difference relative to the previous point. This is formulated both for the cartesian coordinate space

(δX, δY, and δZ) and the cylindrical coordinate system (δr, $\delta\theta$, and δZ). In Figure 4, the red line shows the distance relative to the origin. In contrast, the green line shows the prediction using the deltas.



a) Cartesian deltas **b) Cylindrical deltas**

Figure 4: Deltas preprocessing using two methodologies.

As trajectories will often move away from the origin point closer to the end of the trajectory, delta preprocessing acts as a form of normalization with each value of the output vector having a similar magnitude. Additionally, it minimizes the noise of the trajectory as each prediction is done relative to the previous point limiting the necessity of a convolutional layer to smooth the overall trajectory.

2.2 Generator and Discriminator Architecture

In this adversarial training, the outputs of the generator are used to train the discriminator with the average loss between one batch of generated data and real data defining the loss for the discriminator. The generator loss is based on its capability to fool the discriminator. Both models are trained with an equal batch size of 1,000 trajectories and at learning rates of $2 \times 10^{-4}, 1 \times 10^{-4}$, and 5×10^{-5}. Training occurs for 150 epochs on the *PyTorch* library [12] using a mean squared error loss function on Compute Canada servers (alliancecan.ca).

The generator shown in Figure 5 takes in two inputs: 1) the latent matrix and the 2) label vector. The latent matrix is generated randomly, and the label vector contains information about the WTC of aircraft whose trajectory is being generated. The two inputs are fed into two separate fully connected layers which are the first layers of the generator. These dense layers later converge into a series of 1-dimensional convolutional layers, fully connected layers, and up-sampling layers. The output of the generator is set as the same shape as the original trajectory data.

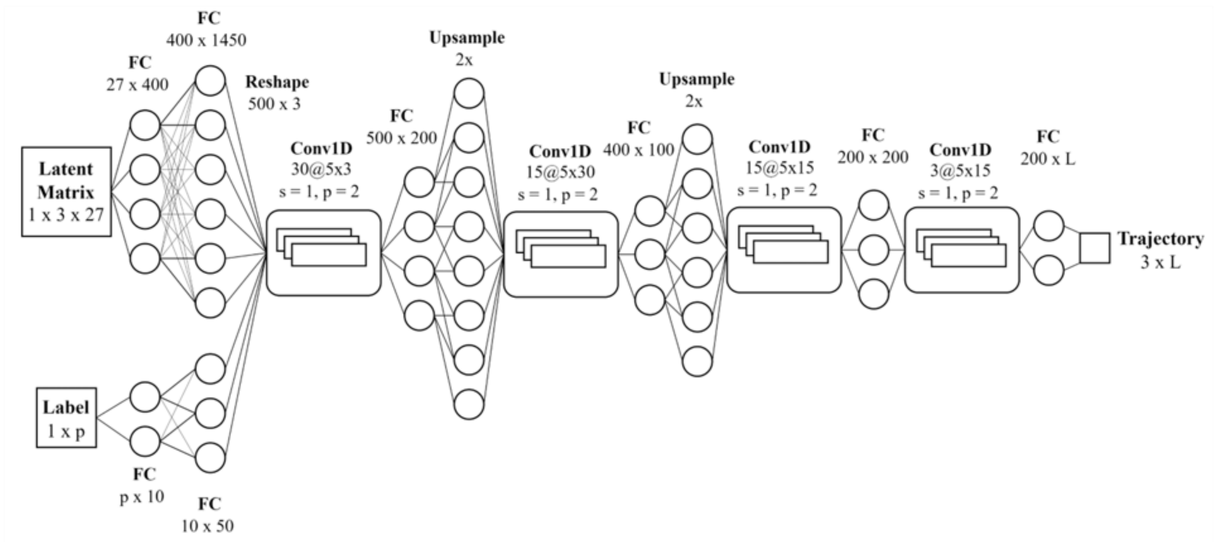


Figure 5: Generator architecture for conditional generative adversarial network.

Fully connected layers are applied to the latent matrix to increase the dimensionality of the trajectories. Because the label has few input variables to the latent matrix, a separate pretraining is performed to allow the connected weight to provide sufficient impact. Following a reshape to match the cartesian dimensions in the trajectory, the 1-dimensional convolutional layers are intended to apply features related to the flight maneuvers seen in Table 1 at random. A focus on shorter term effects is applied with 5 second convolutional filter sizes with same padding and no stride. Upsampling is used to increase the dimensionality of the trajectory while limiting noise which may occur after fully connected layers. After each fully connected layer, a ReLU activation function is applied with no dropout.

The discriminator takes in the output of the generator or a real trajectory as its input. It initially passes through a fully connected layer which later connects to a series of 1-dimensional convolutional layers and more fully connected layers. The architecture is shown in Figure 6,

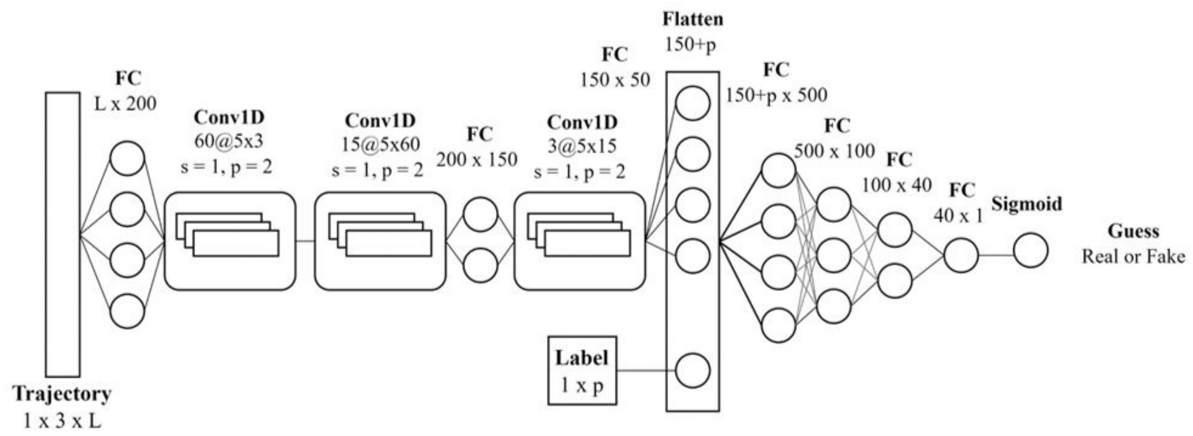


Figure 6: Discriminator architecture for conditional generative adversarial network.

The filters are also standardized with a low kernel size of 5 seconds and identical padding. These convolutional layers are designed to analyze abnormal or incorrect flight patterns within short intervals using similar short-term filters. The label is not considered until after the flattening layer to allow for the discriminator to take advantage of the timeseries nature of the data. After flattening, the label is concatenated to the fully connected layers which are followed by a sigmoid activation function for a classification problem. The discriminator uses LeakyReLU activation (negative slope of 0.05) after the first convolutional layer, and every fully connected layer after the second.

2.3 Baseline Model

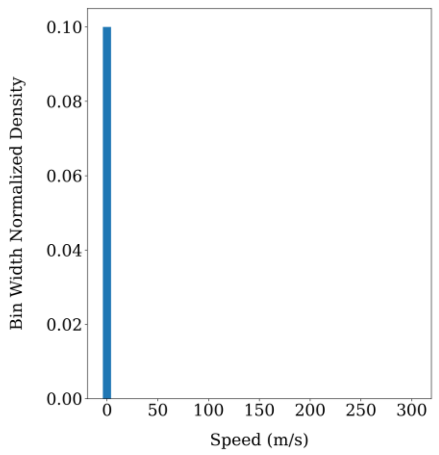
Validation of the effectiveness of using cGAN is done by comparing the results to a GAN baseline with a similar architecture. These architectures are identical to the cGAN networks shown in Figures 5 and 6 apart from the label input, which is removed. As existing baseline models summarized in Section 1.2 were often designed for different applications, the capacity of these models is not suitable for the proposed task. Thus, the proposed baseline shares similar parameters with the existing cGAN architecture, and any statistical improvements observed can be attributed to the label input.

2.4 Validation

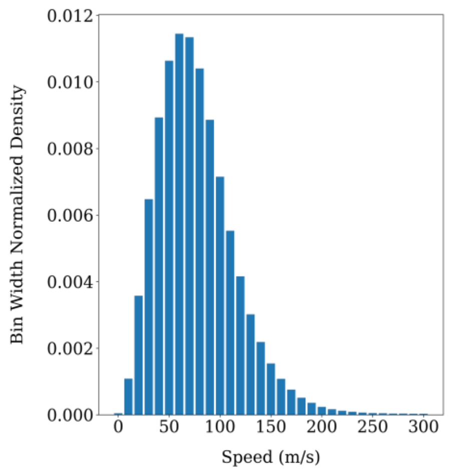
Additionally, although the noise from sensors during trajectory measurement can be ignored for the purposes of this project, the underlying physics of aircraft trajectories must be accurate to use the generated results in aerospace models. Therefore, the proposed validation methodology involves comparing the generated distribution of aircraft turn rate, speed, and pitch rate to the baseline distribution to identify any differences.

The evaluation was performed by using the trained models to generate 10,000 trajectories. For the cGAN, which requires labels for generation, the WTC distribution of generated trajectories was made to match that of the original dataset. Turn rate, pitch rate, and speed were new features calculated from both the original dataset and then generated data.

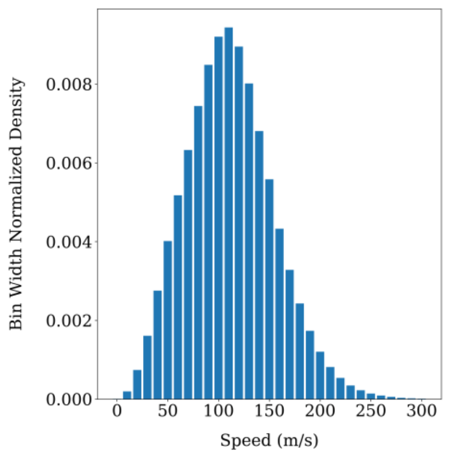
To validate the accuracy of the underlying physics, a comparison is made between the histograms of these three variables in the GAN-generated dataset and the original dataset. This similarity is obtained using the Wasserstein distance, which is a distance measure of 2 probability distributions and has been used as a loss function for GAN networks [13]. To calculate this, the *wasserstein_distance* function was imported from the *SciPy* library [14]. Figure 7 shows a visual representation of the Wasserstein distance on the speed variable.



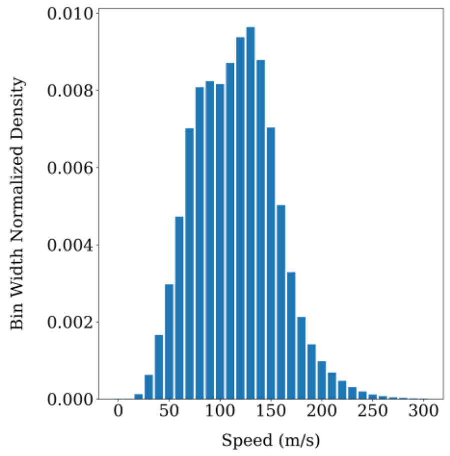
a) 114.805



b) 43.516



c) 4.989



d) Baseline (ground truth)

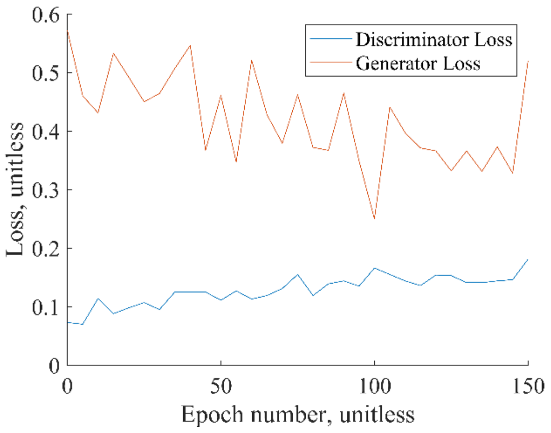
Figure 7: Visual impact of Wasserstein metric and differences relative to baseline

It is proposed that the usage of Wasserstein distance is superior to standard metrics such as an independent RNN/NN, or inception score as they may focus more on shorter term relations such as the noise of the trajectories in comparison to the fundamental physics. Additionally, given the proposed application of the generations described in Section 1.1, this metric is argued to be more important than a subjective accuracy metric.

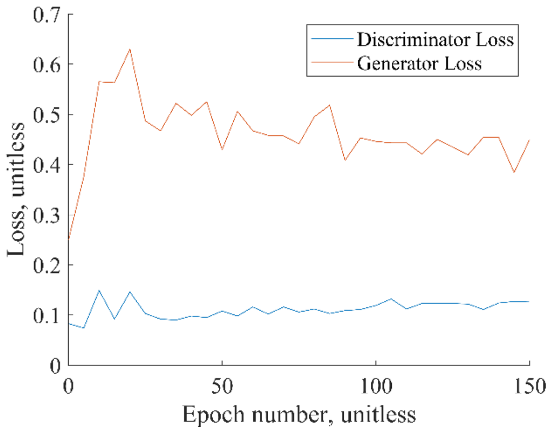
Lastly, by creating a secondary task that is unrelated to the loss function of the GAN, or accounting for the fundamental physics of the generated trajectories, it is proposed this validation methodology avoids overfitting and therefore does not require a train validation split.

2.5 Model Training

During training, it was observed that the discriminator learns significantly faster than the generator with discriminator loss being reduced significantly in epochs before 50. Two changes are made in addition to dropout regularization to limit the learning rate of the discriminator. Firstly, one-sided label smoothing as proposed by [15] is used to modify the true data labels from 1 to 0.9 and penalizes the generator for overconfidence. Secondly, the weight updates after each batch for the discriminator are skipped if loss is less than 20% of the generator loss. This was essential to avoid mode collapse, which occurs when the discriminator becomes too powerful and provides little gradient information for the generator to improve upon. Figure 8 shows the training history of the generator and discriminator for both the cGAN and the GAN baseline.



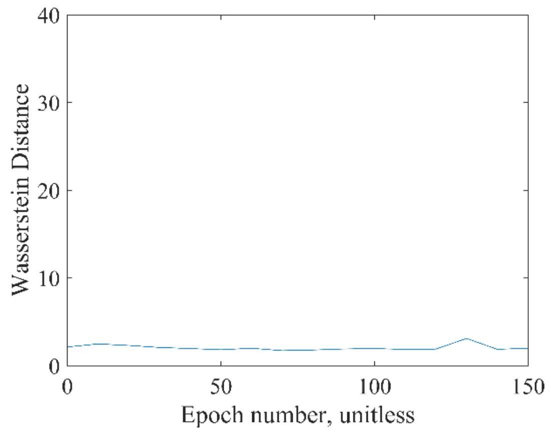
a) cGAN loss over epochs



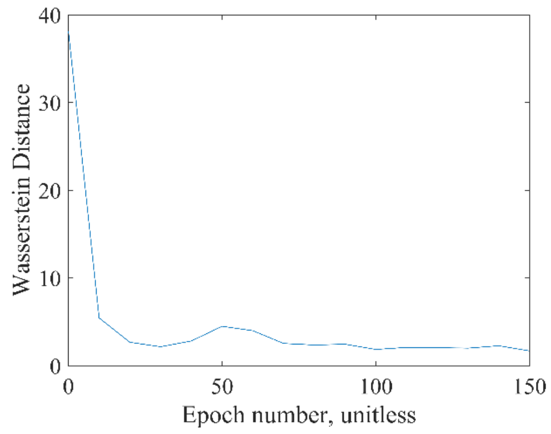
b) GAN loss over epochs

Figure 8: Loss and evaluation of model training over epochs

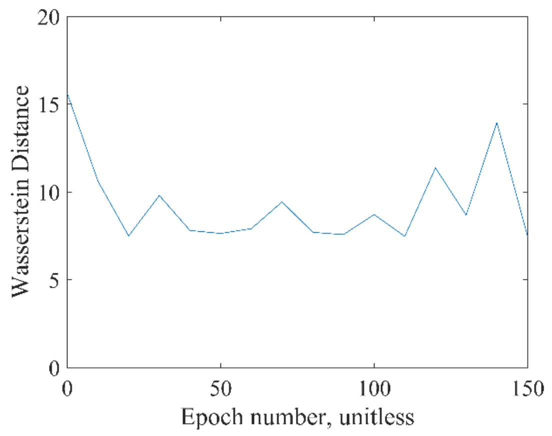
In both models, the discriminator is still seen to significantly overpower the generator, however, the loss of the generator increases at a faster rate in the cGAN case suggesting the training is approaching a stable equilibrium. In contrast, the GAN has a slower rate of increase for the discriminator loss. The loss graph for the GAN model is smoother, but still exhibits fluctuations without a clear indicator of convergence. Despite this lack of convergence on the arbitrary loss function, the variables eventually converge to reasonable values as shown in Figure 9.



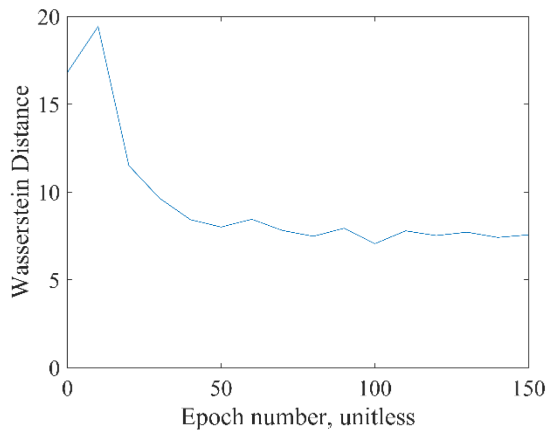
a) cGAN pitch rate



b) GAN pitch rate



c) cGAN vertical rate



d) GAN vertical rate

Figure 9: Loss and evaluation of model training over epochs

In Figure 9, the Wasserstein distance of the pitch rate and vertical rate are shown. Greater instability is seen in the cGAN model although Wasserstein distance metric starts lower in the cGAN and ends on a lower value on average. In contrast, the Wasserstein distance for the cGAN shows sharp increases and decreases within a small-time interval, while for the GAN, it has a smooth decrease over epochs.

3.0 Results

3.1 Preprocessing Validation

The validation of preprocessing methodologies is done on a synthetic dataset with an easily observable function. The baseline function is predefined using two randomly generated variables: distance (D) which is a random value between 1000 and 1200, theta (θ) which is a random value between 0 and $\pi/2$, and radius (r) which is a random value between 500 and 510. Lastly, a random value between 0 and 1 is represented using x. A trajectory with a length L is formed at each timestep t from 0 to L is done using the cartesian equations shown in Equations 1, 2 and 3.

$$X_t = D\frac{t}{L}\cos(\theta) + r[cos(\theta)\frac{t}{L} + (0.01x - 0.005)] \tag{1}$$

$$Y_t = D\frac{t}{L}\sin(\theta) + r[sin(\theta)\frac{t}{L} + (0.01x - 0.005)] \tag{2}$$

$$Z_t = 0.1\frac{D^{1.5}}{L} + 0.1r^{0.5} \tag{3}$$

These equations are intended to provide a predictable curve and used as the target generation for the cGAN model described in Section 2.2 using no processing and the two preprocessing methods described in Figure 4. The target trajectory shape and the effectiveness of the preprocessing is shown at epochs 20, 30, and 40 in Figure 8.

Epoch 20	Epoch 30	Epoch 40

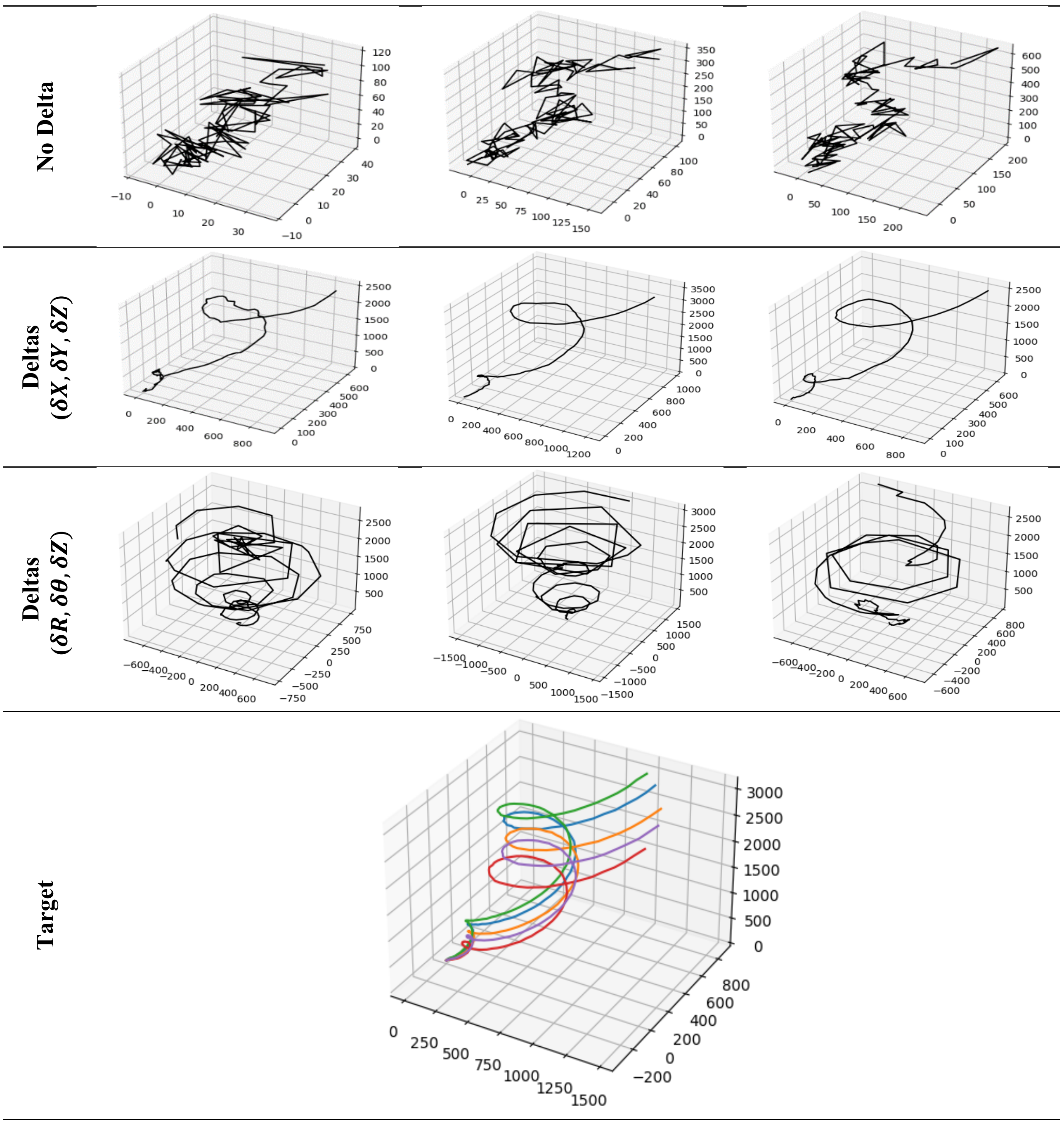
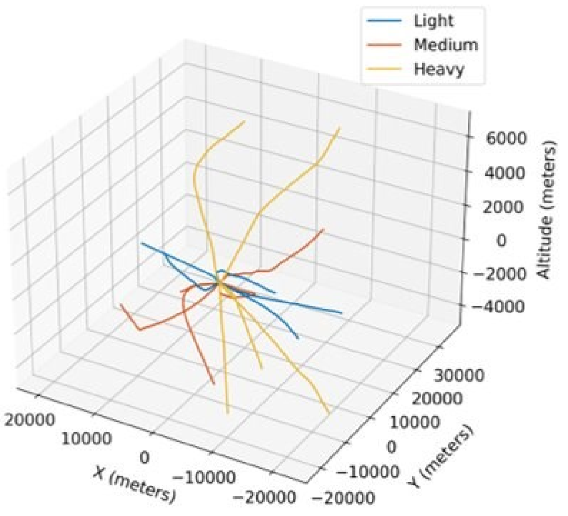


Figure 9: Impact of preprocessing methodologies on synthetic data

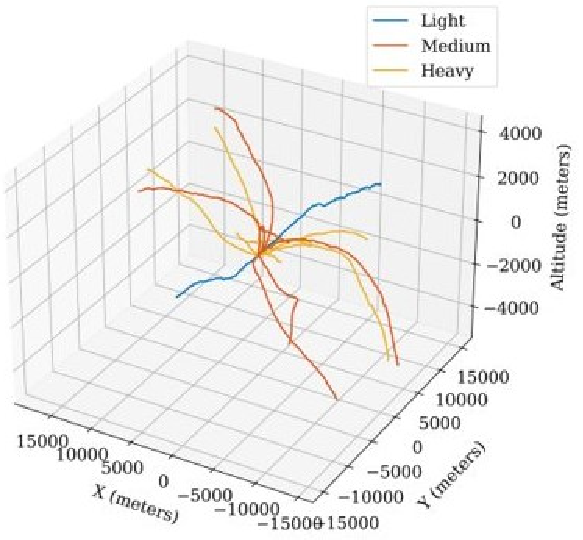
With no preprocessing, the trajectory is gradually approximated but with a significant amount of noise in the trajectory which is still significant by epoch 40. In contrast, the cylindrical deltas, while smoother, fail to capture the underlying shape of the line and do not converge to the approximate trajectories. In contrast, the cartesian deltas both capture the shape of the trajectory in a few numbers of epochs and have significantly superior smoothness compared to the alternative methods. For this reason, they are used in the original problem.

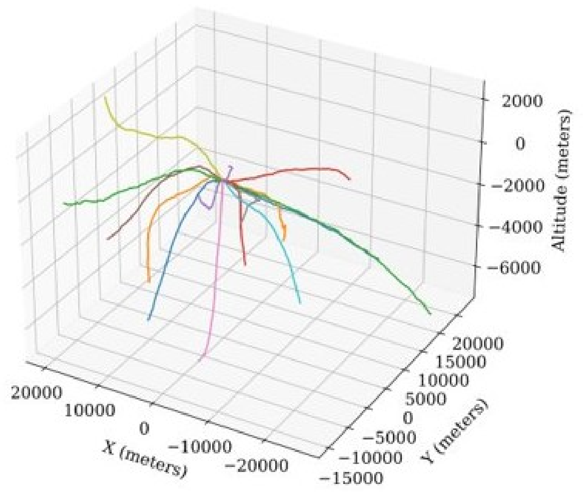
3.2 Trajectory Generation

Using the cartesian deltas preprocessing, trajectory generation is done through training both the GAN and cGAN models for 150 epochs. Trajectories from the original dataset, generated by the cGAN, and those generated by the GAN are shown in Figure 9.



a) Original





b) cGAN **c) GAN**

Figure 10: Original, baseline, and generated aircraft trajectories.

The original trajectories in Figure 9a resemble both the ones generated in the cGAN and GAN in Figures 9b and 9c respectively. Despite this, slightly greater noise is observed for both generated trajectories compared to the base data. This noise is potentially due to the models either 1) not passed through enough epochs, or 2) an overcompensation due to the noise present in the training data. Additionally, certain irregularities were observed in the generated trajectories which visually appeared to show significant drops in vertical altitude inconsistent with expected aircraft trajectories.

3.3 Statistical Comparison

A statistical comparison of the baseline GAN and cGAN model performances were ran at all three learning rates with the results of the best learning rate used for the statistical analysis. Airspace variable distributions are calculated and compared to the base truth using the Wasserstein metric. These metrics are averaged and reported with the standard deviation in parenthesis in Table 2.

Table 2: Wasserstein Distance and Standard Deviation

	Accuracy			**Variability**	
Runtime Parameters	**cGAN** n = 10 $\alpha = 2 \times 10^{-4}$	**Baseline GAN** n = 10 $\alpha = 5 \times 10^{-5}$		**cGAN** n = 10 $\alpha = 2 \times 10^{-4}$	**Baseline GAN** n = 10 $\alpha = 5 \times 10^{-5}$
			Epoch 50		
Turn Rate	9.04 (2.29)*	17.68 (5.00)			
Pitch Rate	2.02 (0.51)*	3.71 (1.70)	Vertical Rate	8.03 (0.96)*	8.30 (0.62)
Speed	13.62 (8.25)*	19.73 (10.3)			
			Epoch 100		
Turn Rate	5.93 (3.06)*	10.92 (2.32)			
Pitch Rate	2.52 (1.42)	2.39 (0.57)*	Vertical Rate	7.62 (0.61)*	7.87 (0.62)
Speed	15.84 910.5)	12.20 (4.34)*			
			Epoch 150		
Turn Rate	5.05 (2.94)*	7.77 (1.30)			
Pitch Rate	2.34 (1.33)	1.97 (0.17)*	Vertical Rate	7.45 (0.47)*	7.68 (0.43)
Speed	14.89 (9.69)	11.10 (4.14)*			

*: Indicates lowest average in the epoch and variable combination

These results show Wasserstein distances under 15 for all variables indicating distributions relatively accurate to the base truth. Most airspace variables have lower Wasserstein metric averages in the cGAN relative to the baseline GAN indicating superior performance of the cGAN model. Despite this, variability is higher in the cGAN model except for the accuracy metrics at epoch 50 where GAN has very high variability. Given the low values for the Wasserstein distance in both models shown in Table 2, the trajectories which are generated by the model appear to match the behaviour of existing aircraft meeting the requirements to generate both accurate and diverse trajectories.

The statistical significance of these improvements is calculated using the Mann Whitney U Test [16], a statistical test requiring no assumption on a gaussian prior and indicates a statistically significant enhancement to a pre-defined confidence level. This statistical test is applied to all airspace variables and summarized in Table 3,

Table 3: U-Metric and Rank of Airspace Variables

	Accuracy			**Variability**	
Runtime Parameters	**cGAN** n = 10 $p < 0.05$ $\alpha = 2 \times 10^{-4}$	**Baseline GAN** n = 10 $p < 0.05$ $\alpha = 5 \times 10^{-5}$		**cGAN** n = 10 $p < 0.05$ $\alpha = 2 \times 10^{-4}$	**Baseline GAN** n = 10 $p < 0.05$ $\alpha = 5 \times 10^{-5}$
			Epoch 50		
Turn Rate	98 (57)**	2 (153)			
Pitch Rate	86 (69)**	14 (141)	Vertical Rate	66 (89)*	34 (121)
Speed	70 (85)*	30 (125)			
			Epoch 100		
Turn Rate	91 (64)**	9 (146)			
Pitch Rate	64 (91)*	36 (119)	Vertical Rate	66 (89)*	34 (121)
Speed	59 (114)	41 (96)*			
			Epoch 150		
Turn Rate	90 (65)**	10 (145)			
Pitch Rate	52 (103)*	48 (107)	Vertical Rate	69 (86)*	31 (124)
Speed	37 (118)	63 (92)*			

*: Indicates lowest test statistic for the Wilcoxon rank sum test for each variable
**: Indicate statistically significant improvement with a 95% confidence interval

With a sample size of 10 tests in both classes, a value in each airspace variable comparison under 23 indicates a statistically significant result. This is seen a total of four times, with the turn rate at epochs 50, 100 and 150 in addition to the pitch rate at epoch 50. 6 other variables, including all vertical rate variables show improvements to the average but are not statistically significant to $p < 0.05$. Therefore, the two cases where the GAN has higher averages appear to be statistically insignificant in contrast to four cases of statistically significant improvements using the proposed cGAN.

3.4 Benefits and Downsides of Proposed Model

These statistically significant improvements occur with the addition of the class label which requires minimal additional computation time or model complexity. Therefore, we propose it has few limitations and downsides relative to existing baseline models. However, the usage of WTC limits the model to account for differences in aircraft purely in three discrete buckets while aircraft may have significant flight differences within each WTC. Additionally, the model does not account for non-fixed-wing flight and would fail to capture these behaviours in aircraft trajectories.

4.0 Conclusion

The project evaluated the performance of how generative adversarial neural network models were able to generate variable trajectories for air traffic control simulations. It was determined that the cGAN model converges faster and outperforms the baseline GAN model at creating varying trajectories. The turn rate and pitch rate accuracies were statistically significant at epoch 50 and the improvement was less marginal at higher epochs, with the cGAN model demonstrating a slight improvement in vertical rate and turn rate, but not in the average pitch rate and speed. Nonetheless, the cGAN model consistently provided more variable trajectories and converged faster than the GAN model, indicating that it provides better performance on average at low training epochs.

Additionally, it is important to note that used preprocessing methodologies had a significant impact on the performance of the GAN models and that the estimation of deltas increased the rate of convergence, as well as improved training quality. Additionally, it is important to note that the cGAN model architecture has many interesting properties that allows for the generation of many different types of aircraft.

References

[1] A. Bauranov and J. Rakas, "Designing airspace for urban air mobility: A review of concepts and approaches," *Progress in Aerospace Sciences,* vol. 125, p. 100726, 2021/08/01/ 2021, doi: 10.1016/j.paerosci.2021.100726.

[2] J. Rushby, "New challenges in certification for aircraft software," 2011: ACM, doi: 10.1145/2038642.2038675. [Online].

[4] G. Jarry, N. Couellan, and D. Delahaye, "On the Use of Generative Adversarial Networks for Aircraft Trajectory Generation and Atypical Approach Detection," in *Air Traffic Management and Systems IV*, Singapore, I. Electronic Navigation Research, Ed., 2021// 2021: Springer Singapore, pp. 227-243.

[5] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[6] X. Wu, H. Yang, H. Chen, Q. Hu, and H. Hu, "Long-term 4D trajectory prediction using generative adversarial networks," *Transportation Research Part C: Emerging Technologies,* vol. 136, p. 103554, 2022/03/01/ 2022, doi: 10.1016/j.trc.2022.103554.

[7] P. Jia, H. Chen, L. Zhang, and D. Han, "Attention-LSTM based prediction model for aircraft 4-D trajectory," *Scientific Reports,* vol. 12, no. 1, 2022, doi: 10.1038/s41598-022-19794-1.