Aerial Object Trajectory Classification by Training on Flight Controller Data and Testing on RADAR Generated Tracks

Henry Holbrook[†], Paris Garrett[†], and Nikhil Behari[†]
NASA Langley Research Center, Hampton, VA, 23681, USA

Chester V. Dolph[*]
NASA Langley Research Center, Hampton, VA, 23681, USA

Chris M. Morris [¶]
Analytical Mechanics Associates, Hampton, VA, 23666, USA

George N. Szatkowski [‡]
NASA Langley Research Center, Hampton, VA, 23681, USA

Onboard collision avoidance is needed to enable safe, autonomous flight operations for NASA projects such as Advanced Air Mobility (AAM), as well as many commercial applications. Real-time aerial object classification will improve onboard collision avoidance algorithm decision making and may reduce unnecessary activation of avoidance systems. This work trains an aircraft trajectory classifier using trajectories from flight controller logs and tests the classifier using RADAR collected trajectories during air to air experiments and ground to air experiments. In contrast to RADAR data, these flight controller logs are relatively abundant, which makes the possibility of substituting flight data for RADAR data an attractive, cost-effective option. The SVM model developed in this work achieved a 79.7% classification accuracy on the first second of radar trajectories of GA, multirotor sUAS, and fixed wing sUAS. Findings from this work indicate that it is feasible to classify sensor collected trajectories using a classifier trained on flight controller data.

I. Nomenclature

V_x	=	Velocity X Estimated
V_y	=	Velocity Y Estimated
V_z	=	Velocity Z Estimated
V_n	=	Velocity North Estimated
V_e	=	Velocity East Estimated
V_d	=	Velocity Down Estimated
sUAS	=	small Unmanned Aerial System
SVM	=	Support Vector Machine
GA	=	General Aviation

II. Introduction

The global market for small Unmanned Aerial Systems (sUAS) was estimated to be $2.84 billion US dollars in 2019 alone and is predicted to grow to an outstanding $11.31 billion by 2027 [1]. Pushing this valuation is the potential for autonomous-sUAS to increase the efficiency of many current operations in a variety of fields, as well as enable many operations that were previously technologically possible but economically impractical. Examples of both include sUAS

[†]Computer Vision Intern, Aeronautics Systems Engineering Branch, AIAA Member.
[*]Aerospace Engineer, Aeronautics Systems Engineering Branch, AIAA Member.
[¶]Engineer Level IV, Safety Critical Avionics Branch.
[‡]Electrical Engineer, Safety Critical Avionics Branch.

improving emergency response operations [2], and sUAS bringing crop monitoring into the budget of the typical farmer [3]. In addition, sUAS can perform a host of operations that are simply not feasible with traditional crewed aircraft, such as door-to-door package delivery in crowded environments [4].

In order to realize the potential gains possible with increasing sUAS use, steps need to be taken in order to ensure the safety of such sUAS operation. sUAS need robust avoidance systems to autonomously deliver packages to a consumer's doorstep to prevent midair collision with other sUAS delivering to adjacent homes. The NASA software architecture Independent Configurable Architecture for Reliable Operations of Unmanned Systems (ICAROUS), takes a step towards the creation of such a system. ICAROUS provides a variety of autonomy functions, notably including an avoidance system that seeks to enable real-time collision avoidance for non-cooperative sUAS [5].

This ICAROUS avoidance system has been tested using RADAR track data collected in real-time from encounters with a General Aviation (GA) plane and a fixed-wing sUAS. However, due to the high frequency of Not-An-Aircraft (NAA) tracks returned by the RADAR instrument, many false activations of ICAROUS occurred [6]. This eventually led to the development of machine learning models designed to classify the intruder objects and parse out the NAA in an effort to better inform ICAROUS in the future and cut down on false activations [7]. Initial analysis showed that it is feasible to classify GA, NAA, and fixed-wing sUAS using radar trajectories using a dataset collected over 2 days by achieving greater than 90% testing accuracy for leave-one-out.

This work improves aforementioned aircraft classification models by substituting aircraft trajectories sourced from flight controller logs for the training data. The addition of these logs brings with it many possible advantages; first and foremost, it is possible that the logs will increase the robustness of the classification. The much larger training set captures a much wider range of aircraft trajectories than a preliminary data collection by sensors. In addition, these logs should be more available than RADAR data in general, as most aircraft will be recording these logs during all of their flights, regardless of use case, while collecting RADAR data on an aircraft takes deliberate design. Utilizing flight controller logs for model training may reduce the amount of resources needed to collect field data while improving robustness, provided that the flight controller data can sufficiently model new sensor data.

III. Prior Work

This section outlines the details of previous work, described in [7], in which aircraft classification was performed using only RADAR data for both testing and training. As mentioned in the Introduction, this work was born out of the fact that an aircraft crash avoidance system, ICAROUS, frequently falsely activated to NAA tracks captured by the RADAR. The work outlined in this section aimed to reduce unnecessary activations of the ICAROUS collision avoidance system through classification of intruder objects as GA, fixed-wing sUAS, and NAA.

This aim was pursued via taking the RADAR data captured during the ICAROUS flight testing and repurposing it into a dataset for a machine learning model to be trained and tested on. This dataset contained 3 classes—NAA, Fixed Wing sUAS, and GA planes. With the RADAR from these 3 classes, two different datasets were made—Initial and Time Interval, and both used t as a parameter. The parameter t, measured in seconds, determines how large of a slice of data is to be used as a sample. For example, if t is 1, then the classifier will train and test on 1-second segments of data; if t is 2, the classifier will train and test on 2-second segments of data. The idea behind the parameter t is that it helps provide insight into how different lengths of data affect model performance. For the Initial Dataset, only the first t seconds of each track are included as samples in the dataset, while in the Time Interval Dataset, each track is sliced into t second segments, and each segment is a sample in the dataset.

From each of these two datasets a feature vector was made, containing the 24 features outlined in Table 1. These features were chosen for two reasons—first because they work well with the t parameter of the model. For example, if the t value is 1, then these features would be calculated over each raw data source for each 1 second sample of data. This 1 second sample of data could be either from the Initial Dataset or the Time Interval Dataset, depending on which was being used, but they would be treated just the same. Secondly, these features were selected because, when taken in concert, they not only effectively capture the distribution of the raw data source, but they also help to highlight potential differences between the classes—differences that can be hopefully be used by the model for classification.

After a feature vector was made, it was then used to train and test a machine learning model. The machine learning model used for classification was a Support Vector Machine (SVM). However, in an attempt to boost performance, a Multilayer Autoencoder was tested as a preprocessing step, reducing the number of features from 24 to 12.

The results of this work were encouraging, as they demonstrated the feasibility of classifying a RADAR trajectory with only a few seconds of data—for instance, an 84.4% leave-one-out accuracy was attained with a t value of 1 with the Initial Dataset. Overall, however, the Time Interval Dataset garnered better results than the Initial Dataset, with

Table 1 Prior Work Features

Average	Maximum	Minimum	Variance	Number of Sign Changes
XZ Component of Velocity	XZ Component of Velocity	XZ Component of Velocity	XZ Component of Velocity	X Component of Velocity
Y Component of Velocity	Y Component of Velocity	Y Component of Velocity	Y Component of Velocity	Y Component of Velocity
Magnitude of Velocity with Z sign	Magnitude of Velocity with Z sign	Magnitude of Velocity with Z sign	Magnitude of Velocity with Z sign	Z Component of Velocity
Magnitude of Velocity	Magnitude of Velocity	Magnitude of Velocity	Magnitude of Velocity	Magnitude of Velocity
Estimated radar Cross Section	Estimated radar Cross Section	Estimated radar Cross Section	Estimated radar Cross Section	

the Time Interval Dataset having a higher leave-one-out accuracy for every t value. The autoencoder only increased classification accuracy when used with the Time Interval Dataset, however, which could be due to the fact that the Initial Dataset only had 250 samples [7].

This work aims to fill the lack of training data by using a large dataset of flight controller collected data to explore if flight controller data may be used to classify sensor collected trajectories.

IV. Experimental Methodology

A. Data Collection

1. RADAR Data

The testing of the ICAROUS avoidance system occurred across two consecutive days, with each day being dedicated to testing its response against one of the two classes of aircraft—sUAS and GA, respectively. The two flight days were August 1st and 2nd, 2019. The ICAROUS avoidance system was installed aboard an 8-propeller multirotor model BFD 1400-SE8, shown in Fig. 1, with an Echoflight Frequency Modulated Continuous Wave RADAR as the input device. The two intruder objects were a USAUAS Tempest sUAS, as shown in Fig. 2, and a Cessna 172s GA plane, as shown in Fig. 3. In addition, many NAA were recorded during the flights, and these NAA would cause ICAROUS to activate and perform an avoidance maneuver when it was not supposed to [7]. On June 15th, 2021, flights were conducted with an Alta X multirotor, as shown in Fig. 4, in which it flew above a grounded version of the same RADAR device used for the flights described above, an Echoflight Frequency modulated Continues Wave RADAR.

The tracks recorded from this RADAR instrument during these flights is what is hereby referred to as the RADAR dataset.

Fig. 1 BFD 1400-SE8 [7]

Fig. 2 Tempest sUAS [8]

Fig. 3 Cessna 172s [9]

Fig. 4 Alta X [10]

2. Flight Controller Data

Flight controller data came from a variety of sources internal at NASA Langley. The aircraft from which the flight controller logs were taken are shown in Fig. 5 and fall into three classes: fixed wing sUAS, GA plane, and multirotor sUAS. The fixed wing sUAS consisted of a SIG-EdgeTRA, FQM-117B, and a USAUAS Tempest. The only GA plane was a Cessna 206H, while the multirotor sUAS aircraft included an Alta 8 Pro Rotocopter, a Tarot T960, and a modified DJI S1000, ISAAC.

(a) SIG-EdgeTRA[11]

(b) FQM-117B[10]

(c) USAUAS Tempest[8]

(d) Cessna 206H[12]

(e) Alta 8 Pro[10]

(f) Tarot T960 [10]

(g) ISAAC [13]

Fig. 5 Flight Controller Aircraft

B. Data Overview

1. RADAR Data

The RADAR received detections at a frequency of 9 Hz and used a Kalman filter in order to assign detections to tracks. Additionally, in an effort to lower false positives, the RADAR would only return a track once it hit a certain threshold of confidence that the track was indeed an object. It was in this track form that the data was returned [7].

A given track provides the following information about the corresponding object. This information was updated approximately 9 times per second:

1) X, Y, and Z Estimated Position (in meters)
2) X, Y, and Z Estimated Velocity (in m/s)
3) Estimated Radar Cross-Section (RCS)

An overview of the RADAR dataset is shown in Table 2.

Table 2 RADAR Dataset Overview

Aircraft Class	Aircraft Name	Ownship Altitude (Meters)	Target Altitude (Meters)	Number of Tracks	Hours of Flight
Fixed Wing	USAUAS Tempset	190	190	22	.13
			160		
		160	190		
GA Plane	Cessna 172s	190	366	21	.14
Multirotor	Alta X	Ground to Air Data	110	16	.21
			80		

2. Flight Controller Data

The flight controller logs had a variety of different data sources that provided position and velocity measurements that were specific to the aircraft. The data source chosen was an extended Kalman filter that not only takes the GPS

as input, but also takes inputs from the other available sensors on the craft into account, potentially including inertial measurement units, gyroscopes, accelerometers, and barometers. This extended Kalman filter feature was selected as the data source for this project due to its increased number of sensors being utilized, leading to more robust measurements. When this feature was not available, the raw GPS log was used instead. The information provided by the Kalman filter or the raw GPS was as follows:

1) Latitude and Longitude (in degrees)
2) Velocity North, East, and Down (in m/s)
3) Altitude (in meters)

An overview of the flight controller dataset is shown in Table 3.

Table 3 Flight Controller Dataset Overview

Aircraft Class	Aircraft Name	Median Altitude (Meters)	Flight Controller Sampling Rate (Updates per Second)	Number of Flights	Hours of Flight	Total Number of Flights by Class	Total Hours of Flight by Class
Fixed Wing	SIG-EdgeTRA	166.55	25	154	22.46	229	178.17
	FQM-117B	217.62	25	19	3.59		
	Tempest	247.8	25	56	152.12		
GA Plane	Cessna 206H	1045	200	1	1.60	1	1.60
Multirotor	Alta 8 Pro	77.27	10	99	11.23	192	24.21
	Tarot	214.61	10	44	4.22		
	ISAAC	224.47	10	49	8.75		

V. Methods

As mentioned previously, the aim of this paper is to improve upon the classification process outlined in Prior Work and shown in Fig. 6, via the substitution of trajectories garnered from flight controller logs as training data. This new approach is summarized in Fig. 7.

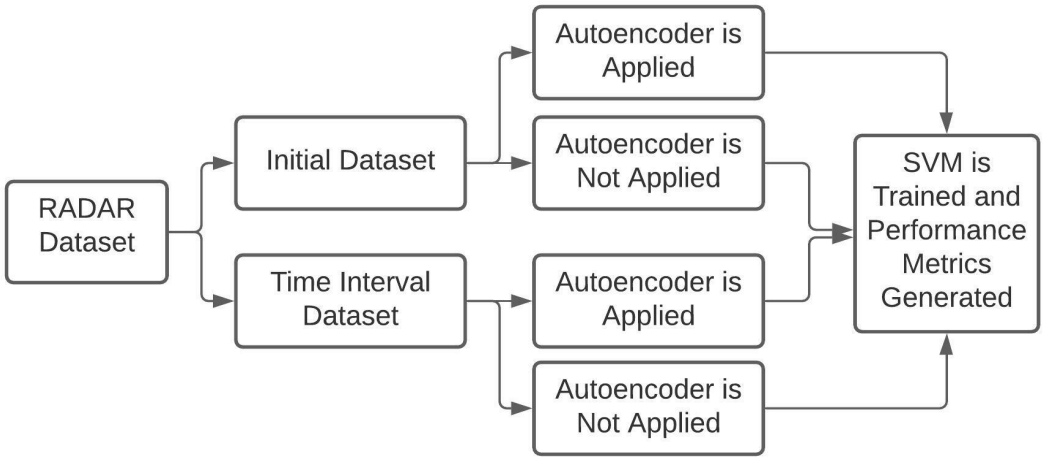

Fig. 6 Classification Pipeline outlined in Prior Work.

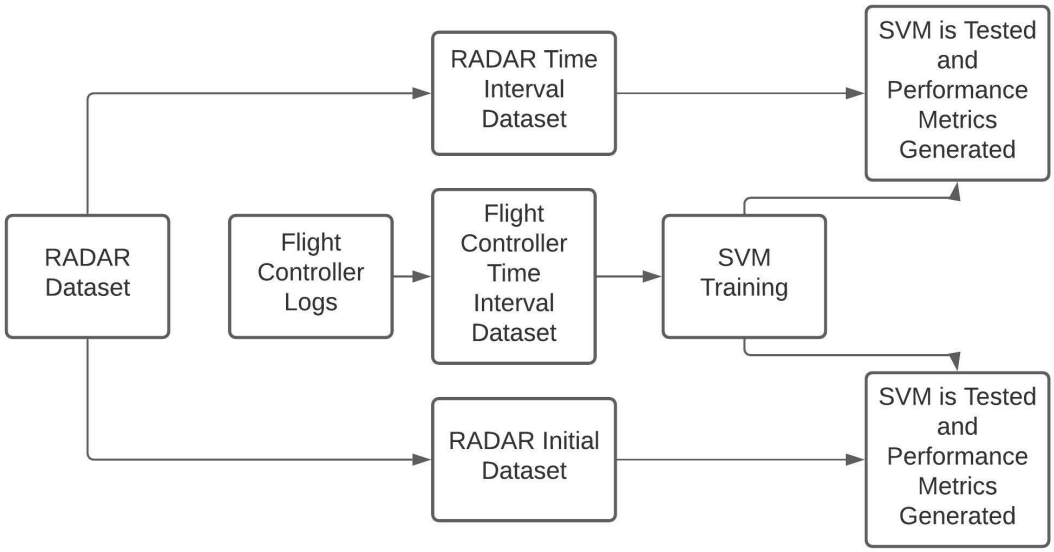

Fig. 7 New Classification Pipeline.

A. Flight Controller Data Processing

All flight controller data was resampled to a frequency of 9 Hz, as to match the refresh rate of the RADAR data. This was done via polyphase filtering, with upscaling and downscaling factors calculated for each particular log [14].

B. RADAR Data Processing

As the aim of this work is to measure the benefit of using flight controller data for classifying radar generated trajectories, estimated RCS was removed from the dataset [7] because this was a feature not available from the flight controller logs, as it is unique to RADAR. This left only X, Y, and Z Estimated Velocity (in m/s) as features, hereby referred to as V_x, V_y, and V_z. Finally, just as the RADAR data was previously used to generate two datasets—Initial and Time Interval—as outlined in Prior Work, the RADAR data was again made into these same two datasets.

C. Feature Vector

Two different feature vectors were constructed—one generated from the flight controller logs and one from the RADAR data. The reasoning for this is that the model was trained on one of these feature vectors, the flight controller one, and tested on the other, the RADAR one. Additionally, the format of the data sources was slightly different for RADAR and flight controller, so different formulas were needed to generate the same features.

Despite this, the two resulting feature vectors have the same collection of 13 features shown in Table 4. These 13 features are a collection of first order statistical features of different velocity measurements taken over a period of time ranging from 1 to 5 seconds depending on the t value of the specific model.

Table 4 Features

Magnitude of Horizontal Velocity	Vertical Velocity	Magnitude of Total Velocity
Average	Average	Average
Maximum	Maximum	Maximum
Minimum	Minimum	Minimum
Variance	Variance	Variance
	Number of sign changes	

1. Flight Controller Feature Vector

For the flight controller feature vector, the data was sourced from the entirety of each flight controller log, in the same manner as the Time Interval Dataset described in Prior Work.

With this dataset, the following were calculated, which were then used to create the 13 features shown in Table 4:

Magnitude of Horizontal Velocity, V_h:

$$V_h = \sqrt[2]{\|V_n^2\| + \|V_e^2\|} \tag{1}$$

Vertical Velocity, V_v:

$$V_v = -V_d \tag{2}$$

Magnitude of Total Velocity, V_t:

$$V_t = \sqrt[2]{\|V_n^2\| + \|V_e^2\| + \|V_d^2\|} \tag{3}$$

2. RADAR Feature Vector

Unlike the flight controller feature vector, there were two different types of RADAR feature vector tested as part of this work—one for both the Initial Dataset and the Time Interval Dataset described in Prior Work. This is for two main reasons. First, as mentioned earlier, the beginning of a RADAR track is of particular importance as it represents when the intruder object first comes into the sensor's range. The robustness of the classifier in these first few seconds is paramount as it represents how quickly, and with what accuracy, the model will be able to classify an aircraft coming into the airspace of the ownship. The second reason is that despite the importance of the Initial Dataset, it is still valuable to test on the totality of the data available, especially considering the relatively small size of the RADAR dataset.

In order to mirror the flight controller feature vector, V_h, V_v, and V_t were calculated as follows:

Magnitude of Horizontal Velocity, V_h:

$$V_h = \sqrt[2]{\|V_x^2\| + \|V_z^2\|} \tag{4}$$

Vertical Velocity, V_v:

$$V_v = V_y \tag{5}$$

Magnitude of Total Velocity, V_t:

$$V_t = \sqrt[2]{\|V_x^2\| + \|V_y^2\| + \|V_z^2\|} \tag{6}$$

D. Machine Learning

Once feature creation was completed for a specified t value and either Time Interval or Initial Dataset was selected for the RADAR data, the two feature vectors would be ready to be used for machine learning—the flight controller feature vector for training, and the RADAR feature vector for testing. These feature vectors would then be used by the classification model—a Support Vector Machine—for training and testing, respectively.

A Support Vector Machine, or SVM, is a type of machine learning model that learns the patterns of a dataset and attempts to subdivide the feature space of the dataset into different classes using a series of hyper-planes. The SVM utilized in this project used a Radial Basis Function, with the regularizer, $\boldsymbol{C}$, set to .02 and γ defined as:

$$\gamma = \frac{1}{(Number\ of\ features\ *\ Variance\ of\ feature\ vector)} \tag{7}$$

$\boldsymbol{C}$ regulates the cost of misclassification of training samples with the margin size of the decision boundary, while γ controls the radius of influence of each training sample. The SVM, per [15] and [16], was implemented in Python 3.7 [17], using the scikit-learn [18] and libsvm libraries [19].

VI. Results

This section begins with an overview of the performance metrics, followed by the results from the Initial Dataset, and finally concludes with the results from the Time Interval Dataset. The results of the models are reported according to their t values, which is the parameter that describes length of time used for training and testing, in units of seconds. For instance, if a model has a t value of 5 seconds, then that model was trained on 5 seconds of data and tested on 5 seconds of data. As described in the Methods section, all times series were resampled to match the 9 Hz of the radar.

A. Performance Metrics

The classification models were evaluated using performance metrics defined in Table 5. The trajectory classification metric captures overall performance of the model on the testing dataset, while training accuracy reveals how well the model learned the training data. Finally, precision and recall for the individual classes reveal how well the model performed on each separate class.

Table 5 Performance Metrics

Performance Metric	Definition
Trajectory Classification Accuracy	$\frac{\textit{No. Correctly Classified RADAR Trajectories}}{\textit{No. Correctly Classified} + \textit{No. Incorrectly Classified RADAR Trajectories}}$
Training Accuracy	$\frac{\textit{No. Correctly Classified Flight Controller Trajectories}}{\textit{No. Correctly Classified} + \textit{No. Incorrectly Classified Flight Controller Trajectories}}$
Fixed Wing* Precision	$\frac{\textit{No. Correctly Predicted Fixed Wing RADAR Trajectories}}{\textit{No. of Fixed Wing Predictions}}$
Fixed Wing* Recall	$\frac{\textit{No. Correctly Predicted Fixed Wing RADAR Trajectories}}{\textit{No. of Fixed Wing Aircraft Trajectories}}$

*Fixed wing is substituted with the class of which metrics are desired.

B. Initial Dataset

Table 6 contains the results of the SVM classification on the Initial Dataset. Trajectory classification accuracy is strong overall, with the majority of results being at or above 86%, and the highest being 88.9% for the t value of 4. Training accuracy is consistently high, with all t values having an accuracy around 99%. As for the individual class performances, the results are all similar for both precision and recall, except for GA Plane recall, which was 100% across all 5 t values.

Table 6 SVM Initial Dataset

t Value	Trajectory Classification Accuracy	Training Accuracy	Fixed Wing Precision	Fixed Wing Recall	GA Plane Precision	GA Plane Recall	Multirotor Precision	Multirotor Recall
1	0.797	0.989	0.750	0.682	0.875	1.000	0.733	0.688
2	0.810	0.989	0.750	0.714	0.875	1.000	0.786	0.688
3	0.860	0.989	0.889	0.762	0.840	1.000	0.857	0.800
4	0.889	0.989	0.889	0.842	0.870	1.000	0.923	0.800
5	0.865	0.990	0.933	0.737	0.818	1.000	0.867	0.867

C. Time Interval Dataset

Table 7 contains the results of the SVM alone on the Time Interval Dataset. Trajectory classification accuracy was consistently high, all around 88%, with a maximum of 88.7% with a t value of 3. Likewise, all training accuracies were consistently high—around 99%. Precision and recall results were strong as well, except for Fixed Wing recall, which hovered around 65%.

Table 7 SVM Time Interval Dataset

t Value	Trajectory Classification Accuracy	Training Accuracy	Fixed Wing Precision	Fixed Wing Recall	GA Plane Precision	GA Plane Recall	Multirotor Precision	Multirotor Recall
1	0.877	0.989	0.882	0.659	0.972	0.925	0.825	0.981
2	0.881	0.989	0.888	0.667	0.970	0.933	0.832	0.978
3	0.887	0.989	0.925	0.667	0.962	0.943	0.832	0.983
4	0.886	0.989	0.911	0.673	0.946	0.938	0.844	0.978
5	0.886	0.990	0.930	0.646	0.955	0.944	0.834	0.986

VII. Discussion

Overall the results are promising and are comparable to the results of the prior work, in which a very similar model was trained and tested on RADAR data alone. Every model trained as part of this work achieved a training accuracy around 99%, indicating that the models were able to effectively learn the patterns of the flight controller dataset, but that some of these patterns may not have been present in the RADAR dataset. For example, flight controller data is not subject to radar sensor noise or signal attenuation.

Trajectory classification accuracy in the Initial Dataset was 79.7% for a t value of 1, as shown in Table 6. With higher t values, performance improved. With a t value of 4, for instance, the SVM alone was able to achieve an accuracy of 88.9%. This result was expected because as t increases, the model benefits from more time series data to classify and thus more opportunities to discriminate between classes. Regardless of t value, however, GA plane recall remained at 100%, perhaps due to the fact that the GA plane tended to travel at higher velocities than the fixed wing or multirotor, making it easier to distinguish. However, despite this, the GA plane precision was very similar to the precisions of the other two classes. When designing an avoidance system, it always needs to activate to avoid GA aircraft. The GA initial precision of 0.8 would cause unnecessary activations of the avoidance system, however, the extra activations may not excessively degrade mission performance as precision is relatively strong.

When comparing the results of the SVM on the Initial Dataset between this work and the pipeline described in Prior Work, this pipeline achieves a higher maximum trajectory classification accuracy—88.9% versus 84.4%—a higher maximum fixed wing precision—93.3% versus 51.6%—and the same maximum fixed wing recall—84.2%. Finally, this pipeline has a higher maximum GA plane precision—87% versus 85.7%—and a higher maximum GA plane recall—100% versus 70% [7]. While both works have the classes fixed wing and GA plane, the prior work in [7] included Not-An-Aircraft (NAA) instead of the multirotor sUAS as its third class."

VIII. Conclusion

The trajectory classification accuracies of this work in the high 80%'s are certainly comparable to the results garnered from prior work with training on RADAR tracks, and they demonstrate that training an aerial object classification model on flight controller logs and validating on RADAR data is possible. With the high availability and relatively low resource cost of flight controller logs, these results are encouraging. Additionally, due to the fact that this pipeline is classifying off of trajectory data alone, future work will generalize the method to other sensors, such as vision extracted trajectories. Finally, developing auto-correlation features as described in [20] should improve classification performance by providing a fitness metric to different aerial object performance envelopes.

References

[1] "Small UAV Market Size to Surpass Around US$ 11.31 Bn by 2027," , Jan 2021. URL https://www.precedenceresearch.com/small-uav-market.

[2] Industries, I. A., "Drones Assisting Emergency Response," , Dec 2020. URL https://www.iai.co.il/news-media/features/drones-assisting-emergency-response.

[3] Roesler, S., "Do inexpensive unmanned aerial systems help with crop health?" , May 2021.

URL https://www.agupdate.com/farmandranchguide/news/technology/do-inexpensive-unmanned-aerial-systems-help-with-crop-health/article_81a817b0-aa03-11eb-8ba7-478dfdc4b77c.html.

[4] Council, N. R., *Autonomy Research for Civil Aviation: Toward a New Era of Flight*, The National Academies Press, Washington, DC, 2014. https://doi.org/10.17226/18815, URL https://www.nap.edu/catalog/18815/autonomy-research-for-civil-aviation-toward-a-new-era-of.

[5] Balachandran, S., Munoz, C., Consiglio, M., Feliu, M. A., and Patel, A., "Independent Configurable Architecture for Reliable Operation of Unmanned Systems with Distributed Onboard Services," 2018, pp. 1–6. https://doi.org/10.1109/DASC.2018.8569752.

[6] Szatkowski, G. N., Kriz, A., Ticatch, L. A., Briggs, R., Coggin, J., and Morris, C. M., "Airborne Radar for sUAS Sense and Avoid," *2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC)*, 2019, pp. 1–8. https://doi.org/10.1109/DASC43569.2019.9081656.

[7] Dolph, C. V., Szatkowski, G. N., Holbrook, H., Morris, C. M., Ticatch, L. A., Malekpour, M. R., and McSwain, R. G., "Aircraft Classification Using RADAR from small Unmanned Aerial Systems for Scalable Traffic Management Emergency Response Operations," 2021. https://doi.org/10.2514/6.2021-2331.

[8] "The Tempest - Professional UAV by UASUSA: AeroExpo," , 2009. URL https://www.aeroexpo.online/prod/uasusa/product-180876-26497.html.

[9] Bakema, P., "F-HATZ: Cessna 172S Skyhawk SP," , Apr 2010. URL https://www.jetphotos.com/photo/6817606.

[10] Mochinski, R., "sUAS - Langley Research Center," , May 2019. URL https://www.nasa.gov/offices/amd/nasa_aircraft/small_unmanned_aircraft_systems/larc.

[11] Manufacturing, S., "Sig Edgetra Arf," , 2017. URL https://sigmfg.com/products/sigrc107eparf-sig-edgetra-arf.

[12] "Cessna 206H," , April 2017. URL https://airbornescience.nasa.gov/category/aircraft/Cessna_206H_-_LaRC.

[13] Gillard, E., "Avoidance is the name of the game at UAS Safety Testing," , Dec 2018. URL https://www.nasa.gov/feature/langley/avoidance-is-the-name-of-the-game-at-uas-safety-testing.

[14] Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, Vol. 17, 2020, pp. 261–272. https://doi.org/10.1038/s41592-019-0686-2.

[15] Guyon, I., Boser, B., and Vapnik, V., "Automatic Capacity Tuning of Very Large VC-Dimension Classifiers," *Advances in Neural Information Processing Systems*, Vol. 5, edited by S. Hanson, J. Cowan, and C. Giles, Morgan-Kaufmann, 1993. URL https://proceedings.neurips.cc/paper/1992/file/eaae339c4d89fc102edd9dbdb6a28915-Paper.pdf.

[16] Cortes, C., and Vapnik, V., "Support-vector networks," *Machine Learning*, Vol. 20, No. 3, 1995, pp. 273–297. https://doi.org/10.1007/BF00994018, URL https://doi.org/10.1007/BF00994018.

[17] Van Rossum, G., and Drake, F. L., *Python 3 Reference Manual*, CreateSpace, Scotts Valley, CA, 2009.

[18] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, Vol. 12, 2011, pp. 2825–2830.

[19] Chang, C.-C., and Lin, C.-J., "LIBSVM: A library for support vector machines," *ACM transactions on intelligent systems and technology (TIST)*, Vol. 2, No. 3, 2011, pp. 1–27.

[20] Petrizze, D., Koorehdavoudi, K., Xue, M., and Roy, S., *Distinguishing Aerial Intruders from Ambient Trajectory Data: Model-Based and Data-Driven Approaches*, 2021. https://doi.org/10.2514/6.2021-2332, URL https://arc.aiaa.org/doi/abs/10.2514/6.2021-2332.