Thirteenth USA/Europe Air Traffic Management Research and Development Seminar (ATM2019)

Clustering Aircraft Trajectories on the Airport Surface

Andrew M. Churchill, Ph.D.
Mosaic ATM, Inc.
Leesburg, VA, U.S.A.
achurchill@mosaicatm.com

Michael Bloem, Ph.D.
Mosaic ATM, Inc.
Leesburg, VA, U.S.A.
mbloem@mosaicatm.com

Abstract—**In this paper, we describe an approach for clustering aircraft taxi trajectories on the airport surface. The resulting clusters can enable improved or novel analyses and optimization of airport surface traffic. In particular, we seek to identify anomalous taxi trajectories. While statistically anomalous trajectories may be planned or expected by a human controller, they may also be unplanned, and thus may represent flights that could pose safety risks. We developed a novel hierarchical clustering algorithm that groups taxi paths in space and then in time. We present results for Charlotte Douglas International Airport (KCLT), showing the common taxi trajectories represented by the clusters, and then discuss leveraging those clusters to identify anomalous trajectories in this dataset. This unsupervised machine learning approach is able to successfully differentiate between typical and anomalous trajectories in a post hoc setting. We have begun to validate the anomalies with subject matter experts as being a combination of infrequently-used paths and true anomalies. In addition, by clustering in time the trajectories in a shape-based cluster, we can separate free-flowing trajectories from those with stops and identify some common stopping points. Finally, we identify numerous extensions of this approach, and other applications for the underlying clustering methodology.**

Keywords-airport operations; taxi trajectories; airport safety; machine learning; clustering; anomaly detection

I. INTRODUCTION

The surface of a large airport is a complex environment, with aircraft arriving and departing on the runways, other aircraft taxiing to and from runways and maintenance facilities, and a variety of motor vehicles performing other missions. Some automation already supports controllers to help ensure safety and to maximize efficiency. The importance of these systems will increase as the airport surface becomes more congested and new air vehicles with different operating characteristics are introduced. Many approaches for maintaining safety (e.g., identifying flights off their expected taxi route, identifying flights that might intrude into the runway), and improving efficiency (e.g., deconflicting taxi routes, scheduling runway operations) rely on having simplified representations (i.e., a common specification, whether point or line-based, across multiple flights) of taxi trajectories. To that end, in this paper we describe an unsupervised machine learning approach for learning these simplified representations from historical trajectory data. We provide details on one application of these trajectories—identifying anomalous taxi paths—and then propose other ways that these clusters may support airport operational improvements.

A trajectory, in our context, is a collection of observations of a particle's path through space over time. These are clearly of interest in a variety of transportation applications (e.g., taxiing aircraft, cars on a freeway, long-distance cargo ships) and other domains, but working with such data directly can be challenging for several reasons:

1. *Volume of data*: depending on the frequency of observation, number of vehicles, etc., the number of records may be huge, presenting problems for computation and visualization.
2. *Noise in observation*: position observations are generally imprecise, with the error dependent on vehicle dynamics and the surveillance technology employed.
3. *Uneven sampling*: observations for different vehicles may not occur at the same instants, or the same rate, or may occur at an irregular rate.

Because of these challenges, analyzing taxi trajectories is often facilitated by assuming an aircraft is using some standard path, hopefully a path learned from observed data. Generating these representative paths for groups of flights is an example of clustering, a powerful machine learning approach for finding relevant patterns in an unlabeled data set [1]. A variety of research efforts across domains have examined the problem of clustering trajectories, fueled particularly in recent years by computational advances. Several recent papers provide a thorough review of this area [2] [3].

Several recent lines of inquiry specific to aviation have examined the development of trajectory clustering approaches and their applications, including for flights in the terminal area [4] [5], and the en-route airspace [6]. The research described in [4] and [6] uses DBSCAN for clustering. The approach taken in [5] was a bit different—they represented trajectories as sequences of turning points ("waypoints") and then clustered by using NASA's SequenceMiner engine to look for long common subsequences of such "waypoints" in trajectories [7]. A similar "partition-and-detect" approach for detecting outlier sub-trajectories based on shape only has been applied to hurricane and animal trajectories [8].

Our work is inspired in part by the approach taken in each of these papers, as we describe in section 2. However, a significant difference between airborne and surface trajectories is that aircraft often stop while they are on the airport surface. These wide variations in and periods of zero velocity introduce significant complications to the pre-processing and clustering approaches used in this type of analysis, motivating novel contributions in our algorithmic approach, particularly the second-level time-based clustering within shape-based clusters.

One particular application of these clustering techniques drives the technical work presented here: the identification of anomalous flight trajectories on the airport surface. Anomaly detection can be performed without clustering (e.g., see [9], [10] for related work for airborne flights). However, clustering can enable anomaly detection if data points are considered anomalies when they are assigned to small clusters or deemed not suitable for assignment to any cluster.

Little research has been conducted to identify anomalies in airport surface surveillance data. In an offline, post hoc analysis, anomalous observations can be mined to identify risks that can be mitigated with improved procedures, communications, and monitoring protocols. In an online, real-time application, partial trajectories can be matched to known clusters, and those without a suitable match tagged for interrogation by a controller. In this paper, we do not attempt to solve all the potential challenges derived from these applications, but are motivated by them to develop a robust and flexible approach for clustering flight trajectories on the airport surface.

The remainder of the paper focuses on describing the developed clustering and anomaly detection approach, the data used for the demonstrations, the results of the demonstrations, and directions for continued work.

II. Clustering and Anomaly Detection Approach

Our algorithmic approach to developing clusters of taxi trajectories has several steps, but has the DBSCAN clustering algorithm at its core. Of the many clustering approaches that exist, DBSCAN (Density-based spatial clustering of applications with noise) is one of the most powerful and flexible [11]. DBSCAN builds clusters by iteratively finding "core" points with many nearby neighbor points. Points with too few or too distant neighbors are assigned to no cluster.

DBSCAN has been successfully applied for clustering trajectories of airborne aircraft [4] [5] [6]. Such applications of DBSCAN typically involve resampling trajectories at even time intervals and then clustering the resulting trajectory points. Unfortunately, two related considerations complicate the application of DBSCAN for airport surface trajectory clustering and anomaly detection:

1. *Aircraft can stop*: even those trajectories sharing the same 2D shape can involve relatively large and operationally-relevant variations in speed, such as long periods of no movement. This is particularly true for departures, which may stop in queue, or to absorb assigned delays.
2. *Measurement errors*: errors in recorded surface trajectory point locations can be large relative to sometimes very low trajectory velocities.

We overcame this issue by developing a two-level hierarchical clustering approach outlined in Figure 1. In the first level of clustering, we clustered trajectories based on their 2D shapes. Then, at the second level, we clustered trajectories that shared the same shape (i.e., in the same shape-based cluster) based on the time they took to traverse the steps in the shared trajectory shape.

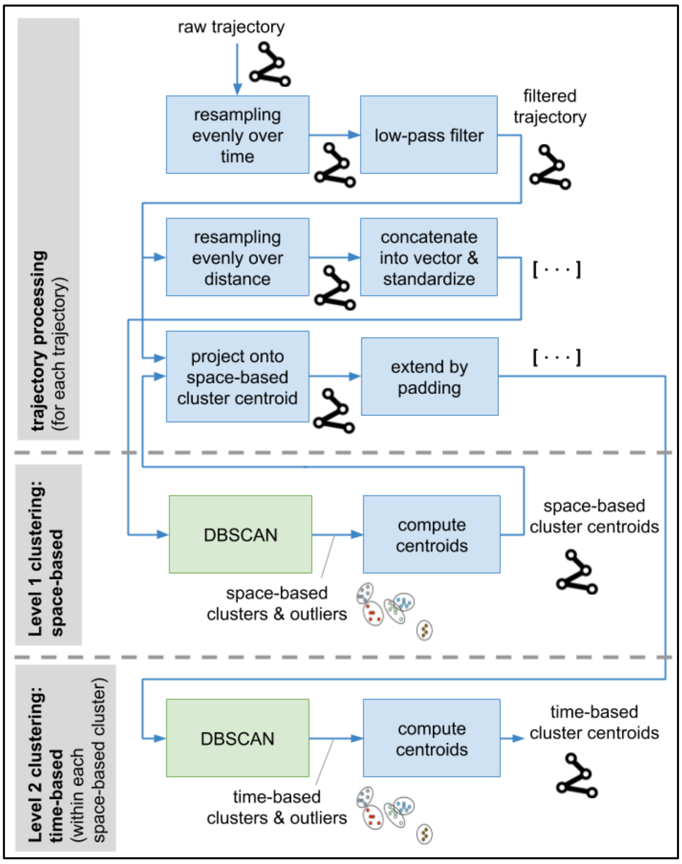

Figure 1. Algorithm flowchart

This approach aids in the interpretation of clusters and outliers by specifying them in distinct dimensions (i.e., space and time) that are both important in the airport surface context but in different ways that may involve different actions.

The first stage of our approach is processing the trajectories, during which we resample and standardize the trajectories to provide vectors of standard length to the clustering algorithm. We evaluated using Dynamic Time Warping as a distance metric [12], which may have eliminated the need for resampling, but determined that our approach would produce more explainable results. Furthermore, unlike Dynamic Time Warping, our approach provides a natural method for deriving cluster centroids from the shape-based clusters' member trajectories, and these centroids are key for computing the distance metric we use in the time-based clustering step, as described below.

In our first level of clustering in the hierarchy, we resampled trajectories after even intervals of distance traversed and then clustered trajectory shapes represented by the resulting sample points. Unfortunately, this approach is vulnerable to measurement noise, which can lead to erroneous computation of distance traversal, particularly when aircraft are moving slowly relative to the magnitude of the measurement noise and measurement sampling rate. Inspired by [6], we overcame this issue by first resampling the trajectory in time to achieve an even 1Hz sample rate (with the traces Python package [13]) and then passing the 1Hz trajectory latitude and longitude points through a low-pass finite-impulse response filter to remove measurement noise. We accomplished this filtering with signal processing software available in the `scipy` Python package [14]. We hand-tuned the filter settings based on visual inspection of the filtered trajectories, ultimately selecting transition widths of 0.2 Hz for departures and 0.4 Hz for arrivals, and attenuation stop bands of 40 dB and cutoff frequencies of 0.001 Hz for both departures and arrivals.

The resulting filtered trajectories were again resampled, but this time at even intervals of distance traversed. More precisely, we resampled to achieve 50 evenly-spaced points for arrivals and 100 for departures. The filtered and resampled latitude and longitude points were concatenated into one row vector per trajectory. Arrival trajectories were then put into an arrival matrix with the number of rows equal to the number of arrival trajectories and the number of columns equal to 100 (50 latitude points followed by 50 longitude points). A corresponding departures matrix was created. Prior to clustering, these matrices were standardized so that variations in distances associated with degrees latitude and longitude did not have undue influence on the ensuing clustering.

After this pre-processing, the resulting trajectories were clustered using the `scikit-learn` Python package [15] implementation of DBSCAN. While a distance metric like the physical distance between the aircraft on the two trajectories summed over sample points would have been more explicitly related to the physics of surface trajectories, we instead used the default Euclidean distance, emboldened by the success of clustering based on this metric in other ATM trajectory clustering efforts [4] [6].

DBSCAN has two parameters that must be tuned: the distance `eps` within which there must be `min_samples` other samples for a given sample to be considered a "core" sample. We developed an approach informed by our anomaly detection application to optimize the selection of these values. First, we performed an exhaustive search across 132 pairs of parameters for arrival clusters and 99 pairs for departure clusters. We then selected parameters that maximized the Silhouette coefficient [16], a cluster quality metric that is higher when samples in the same cluster are close to each other but far from samples in other clusters, while keeping the fraction of outliers below a threshold we deemed appropriate for our proposed anomaly detection application (0.05). Figure 2 shows how the desiderata of high cluster quality and a low fraction of trajectories identified as outliers can be traded off against each other. Achieving higher-quality clusters as quantified by the Silhouette coefficient often can only be accomplished if more trajectories are deemed outliers that do not belong to any cluster. Our 0.05 threshold for the fraction of outliers led us to select 0.6 and 1.25 for `eps` and 4 and 30 for `min_samples`, respectively, for arrival and departure clustering (achieving Silhouette coefficients of 0.80 and 0.87, respectively).

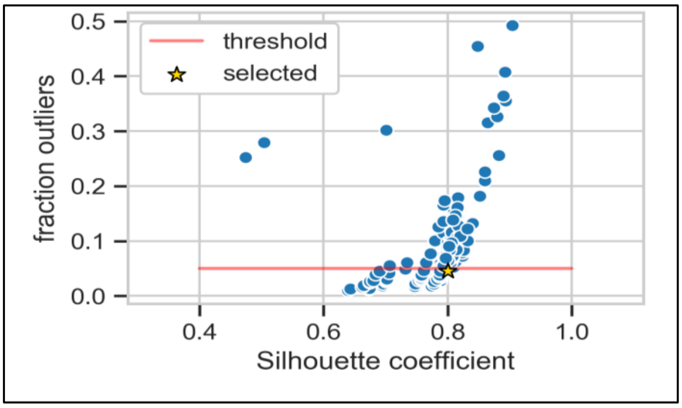

Figure 2. Arrival clustering parameter selection approach

The second level in the hierarchical clustering approach involves clustering the trajectories within each space-based cluster based only on how they moved over the shared trajectory over time. The first step in this process is computing the median trajectory shape (or cluster "centroid") from the trajectories in the shape-based cluster by computing the median over all the trajectories of the latitude and longitude values for each of the 50 or 100 points in the evenly-spaced representations of the trajectories. Next, to completely remove any variations in trajectory shapes and isolate the time spent traversing the trajectory, each of each trajectory's filtered 1Hz sample points was projected to the closest point on the shape-based cluster centroid trajectory. After this projection, each trajectory was represented as a vector in which the tth element contains the aircraft's distance along the cluster centroid trajectory t seconds after the trajectory began, expressed as a fraction of the total centroid trajectory distance to facilitate clustering parameter tuning.

The trajectories were made to have a uniform length equal to that of the longest trajectory in the cluster. Shorter trajectories padded with ones to represent that those flights already arrived at the end of the trajectory by those later times. In this second level of the clustering hierarchy, we used the ℓ_1-norm of the difference between two trajectory vectors as the clustering distance metric. More precisely, the distance between the distance-along-centroid versions of trajectories *i* and *j* is:

$$D(i,j) = \sum_{t=0}^{T} \left| d_i(t) - d_j(t) \right|$$

where di(t) is the distance along the cluster centroid traversed by trajectory i at time t expressed as a fraction of the total length of the centroid trajectory (i.e., the tth component in its representative vector), dj(t) is the corresponding value for trajectory j, and T is the duration of the longest trajectory in the shape-based cluster. We again clustered with DBSCAN, but this time left `min_samples` at the default value of 5 and set `eps` to 20% of the median time duration of the trajectories in the shape-based cluster. This value for `eps` would be achieved by two median-duration (in time) trajectories that, on average, stay at a distance equal to 20% of the total trajectory length from each other. The output of this second-level clustering is a time-based cluster or outlier assignment for each trajectory within each shape-based cluster.

In the next section, we describe the airport surface surveillance data we fed into this algorithm to derive clusters and identify outliers.

III. Data for Study

To demonstrate the filtering and clustering approach described in the previous section, we used surface surveillance data taken from an archive that Mosaic ATM maintains. In this section, we describe the data preparation that preceded our modeling effort.

At most large airports in the U.S., surface surveillance is provided by Airport Surface Detection Equipment, Model X (ASDE-X). This system fuses position reports from several types of radar, multilateration sensors, and Automatic Dependent Surveillance - Broadcast (ADS-B) sensors [17]. The resulting data is displayed to controllers to provide situational awareness, and is distributed through the FAA's SWIM Terminal Data Distribution System (STDDS).

To generate test and evaluation datasets for this research, we parsed the archived XML records from STDDS to create tabular data. In the interest of supporting repeatable research, we now describe a few details on the fields extracted from the XML and the filtering applied. Each XML record may contain position reports for multiple flights at a single airport, and we parsed each of these, extracting the timestamp, STDDS Track ID (STID), callsign, latitude, longitude, altitude, speed, and heading. According to the schema, records always include airport, time, STID, latitude, and longitude, and we allowed the other four fields of interest (callsign, altitude, speed, heading) to be null. We infer the callsign for the trajectory as the most-commonly appearing non-null value for this field.

After this parsing and filtering, we group data into individual trajectories using the STID. All records for each STID are then processed together. Using standard airport adaptation data that Mosaic maintains, containing ramp and runway polygons, we scan the trajectory to determine the first and last times a flight is observed in each runway, and the first and last times it is observed in a ramp. Flights that have a last ramp time before first runway time are marked as departures, with the opposite relationship implying arrivals. Some ambiguous cases are identified, and are not included in this analysis at present. To simplify our analysis, the trajectory is then filtered to show only points between the ramp and runway. Flights that do not include a surveillance hit both in a ramp and in a runway are excluded from this dataset, although expanding beyond this scope is clearly relevant for future work.

We selected Charlotte Douglas International Airport (KCLT) to demonstrate our analysis approach. A diagram of the airport is shown in Figure 3. The airport has four runways—three oriented north-south, and one diagonal, although this one is little-used for takeoffs and landings. We processed one week of data (November 4-10, 2018) for this demonstration. The dataset included 5100 arrivals and 4991 departures, in line with the typical KCLT schedule of 700-800 each of arrivals and departures per day.

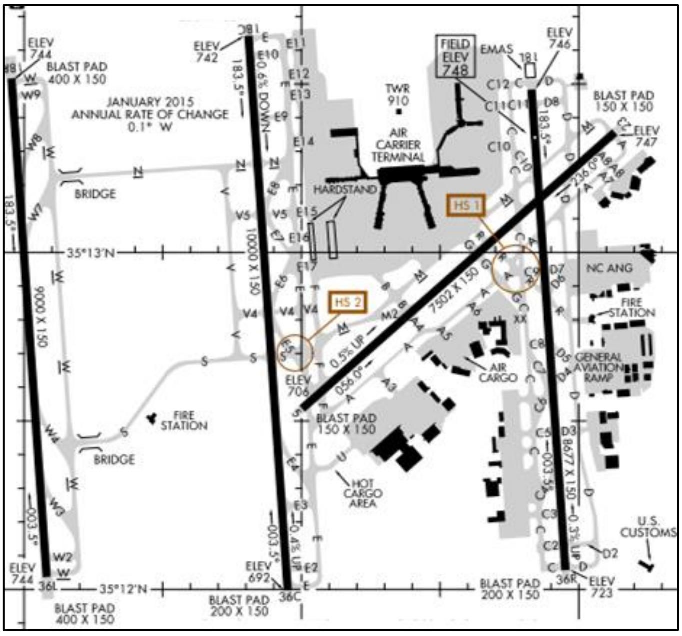

Figure 3. KCLT airport layout (FAA)

IV. Results

We have applied our proposed algorithm to the dataset described in the previous section, and here present results of this analysis.

Our first area of interest is in clustering taxi trajectories according to their shape alone (i.e., the first level of our hierarchical approach). Figure 4 shows the sizes of the shape-based clusters for arrivals and departures, as well as the number of outliers for each. For both arrivals and departures, the number of trajectories assigned to each cluster decays roughly exponential with the rank of the cluster size, suggesting that the bulk of trajectories fall into one of a few common clusters, but that many much smaller clusters exist as well. Many more clusters were identified for arrivals than departures, including many small clusters with less than 10 trajectories, likely because arrivals use a greater variety of taxiways and runways than departures at KCLT. The number of outliers was nearly the same for arrivals and departures, which was by design and enforced in our DBSCAN parameter selection process. The number of outliers could easily be made higher or lower, with corresponding tradeoffs in the quality of the shape-based clusters.

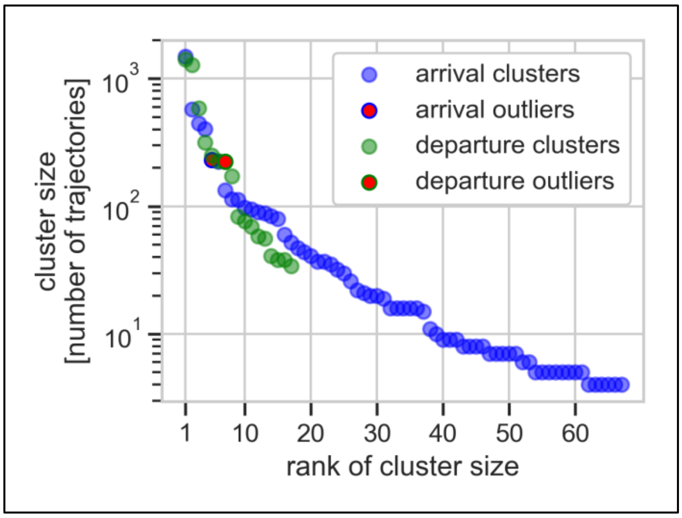

Figure 4. Shape-based cluster size

However, simply generating clusters is not enough without knowing whether they represent reasonable structure in the data. Necessarily, there is some subjectivity to judging this, but in Figure 5, we show the five largest arrival clusters. Two are paths exiting runway 36L taxiing to the main ramp (upper left), while two others are paths exiting runway 36R taxiing to the main ramp (upper right). One path (lower left) is for exiting 18R taxiing to the main ramp. These paths reflect a mix of north and south flow operations, and are consistent with existing subject matter expertise about taxiway usage at KCLT.

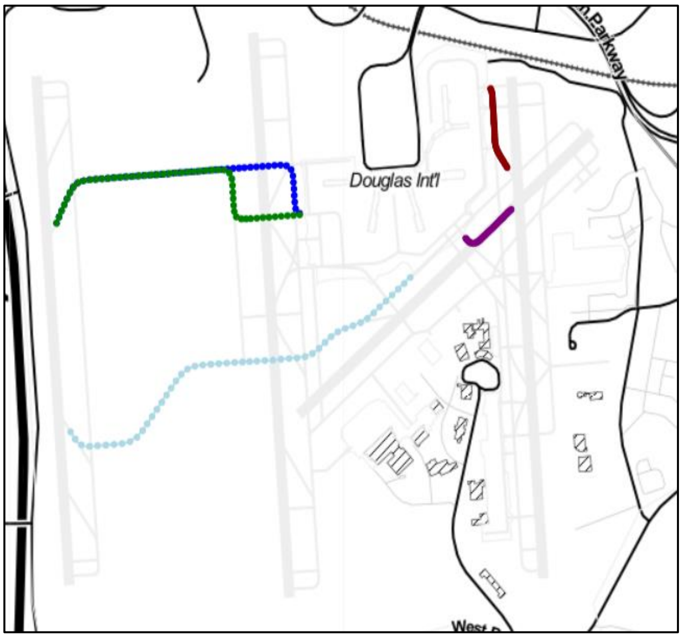

Figure 5. Largest shape-based arrival clusters

The centroid of largest of these arrival clusters is represented in blue, shown in Figure 6 with 50 randomly-selected individual cluster member raw trajectories represented in faint black. The centroid here (in blue) follows high-speed exit W7 off 36L, to taxiway N, crossing runway 36C, turning south on taxiway E, then entering the ramp along E15 at handoff spot 9W. Note that one of the randomly-selected taxi trajectories in this figure taxis further south along E and enters the ramp along E16 at handoff spot 10W. It is interesting that the clustering algorithm grouped these with those arrivals entering at 9W, presumably because so much of the trajectory was identical, and their operational impact is essentially equivalent.

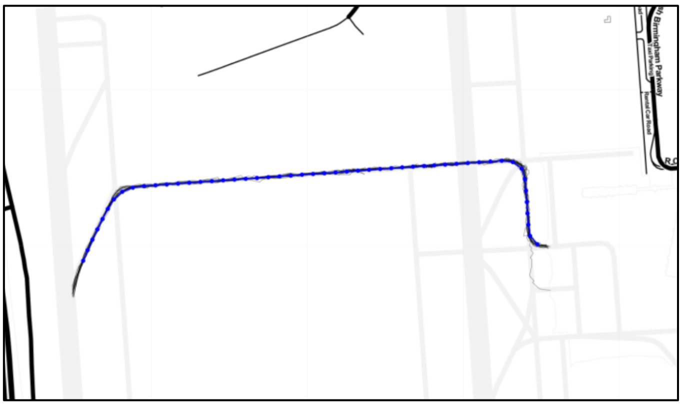

Figure 6. Detail on largest shape-based arrival cluster

In Figure 7, we show that the algorithm is equally adept at learning realistic structure from departure taxi trajectories. In this map, we see three cluster centroids approaching the departure queue for runway 36C, and one each approaching the departure queue for runways 18C and 36R. Consistent with the mixed north/south flow usage, we see both directions of runway being used during the one-week study period.

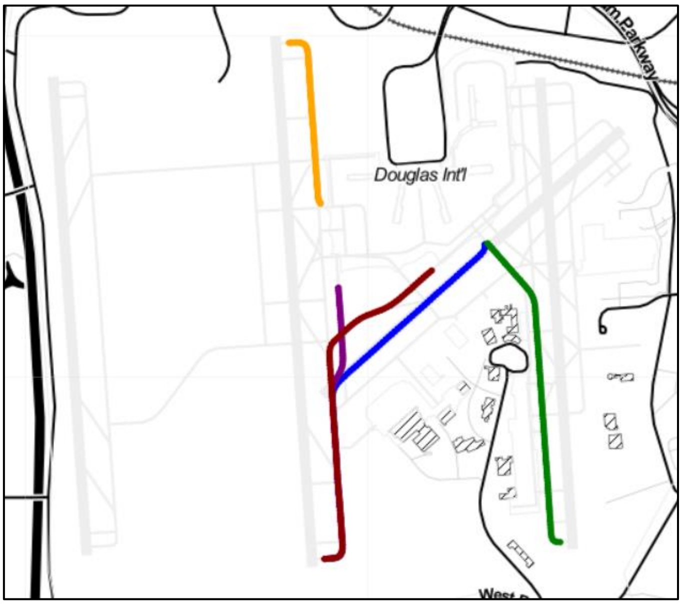

Figure 7. Largest shape-based departure clusters

The largest of these departure clusters is represented in blue, shown in Figure 8 with 50 randomly-selected individual cluster member raw trajectories represented in faint black. These flights are transiting from the main ramp to depart on runway 36C. They exit the ramp at handoff spot 24, then turn onto runway 23 to taxi to the southeast, as is the typical present use of this runway. Interestingly, in this cluster, some flights turn off runway 23 at A4 to reach taxiway A. Most flights, however, continue along runway 23, eventually reaching taxiway E to travel south to the queue for runway 36C. As for the largest arrival cluster, these two different, but functionally identical, paths are joined together into the same cluster.

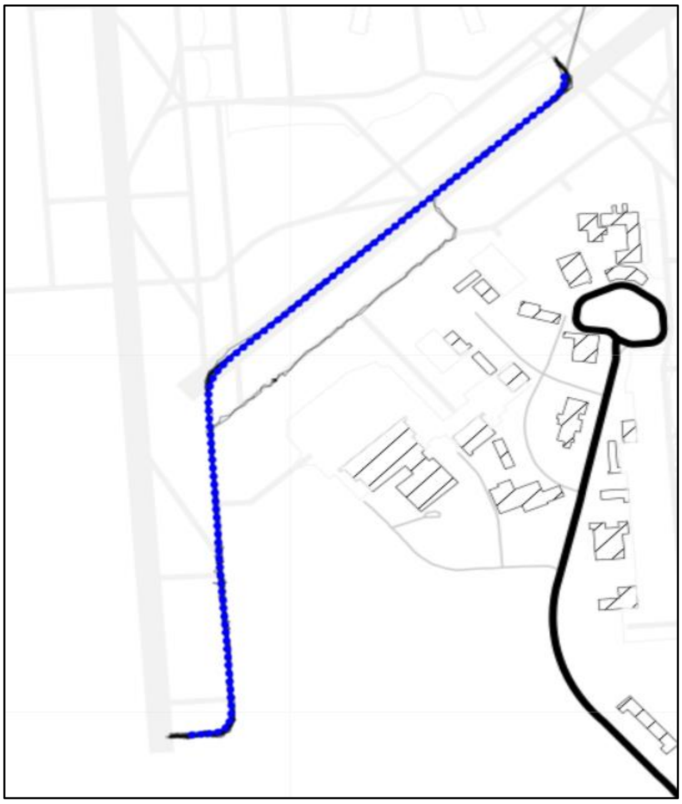

Figure 8. Detail on largest shape-based departure cluster

Identifying spatial structure in the taxi trajectory data is an important function that we believe may serve many applications. However, we are also strongly interested in automatically detecting anomalous taxi trajectories. A byproduct of our algorithmic approach leveraging DBSCAN is that some fraction of our input trajectories is automatically is not affiliated with any cluster, and can then be categorized as anomalous. Next, we show several interesting trajectories not assigned to any cluster by the first level, and describe the nature of the anomaly. In Figure 9, a flight arrived on runway 36C (by itself somewhat unusual) but then taxied directly to the maintenance hangars south of runway 23. This does not represent a safety-related anomaly, but is not activity typically observed, or otherwise easily extracted from airport surveillance data.

Figure 10 shows a departure leaving from the northeast corner of the main ramp via handoff spot 29, then taxiing south along taxiway C to the departure queue for runway 36R. Following C all the way south is a common path during north flow operations. However, this flight turned off C at runway 23, and taxied a short way down 23 before turning off on G to rejoin C. This maneuver avoids a known congestion hotspot where C, A, and R converge (highlighted on the section of the FAA chart included in Figure 3), but was little-observed in our dataset, and was flagged as an anomaly. This is a useful finding in a real-time application, because the flight may have turned errantly off C.

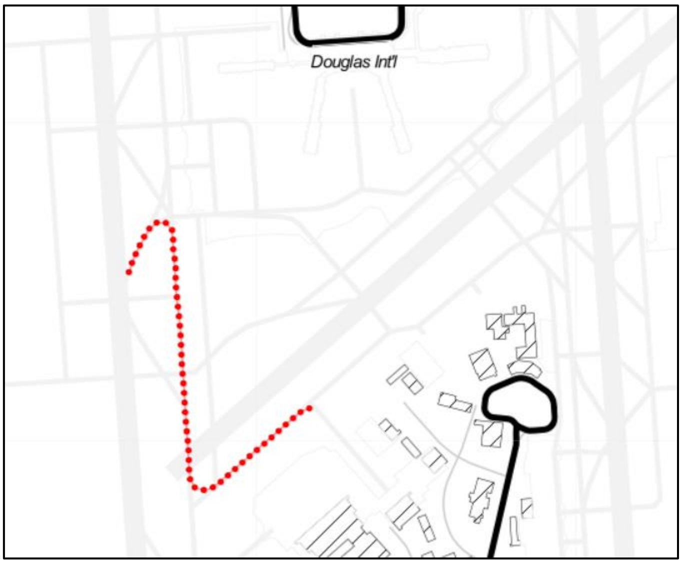

Figure 9. Example arrival trajectory anomaly

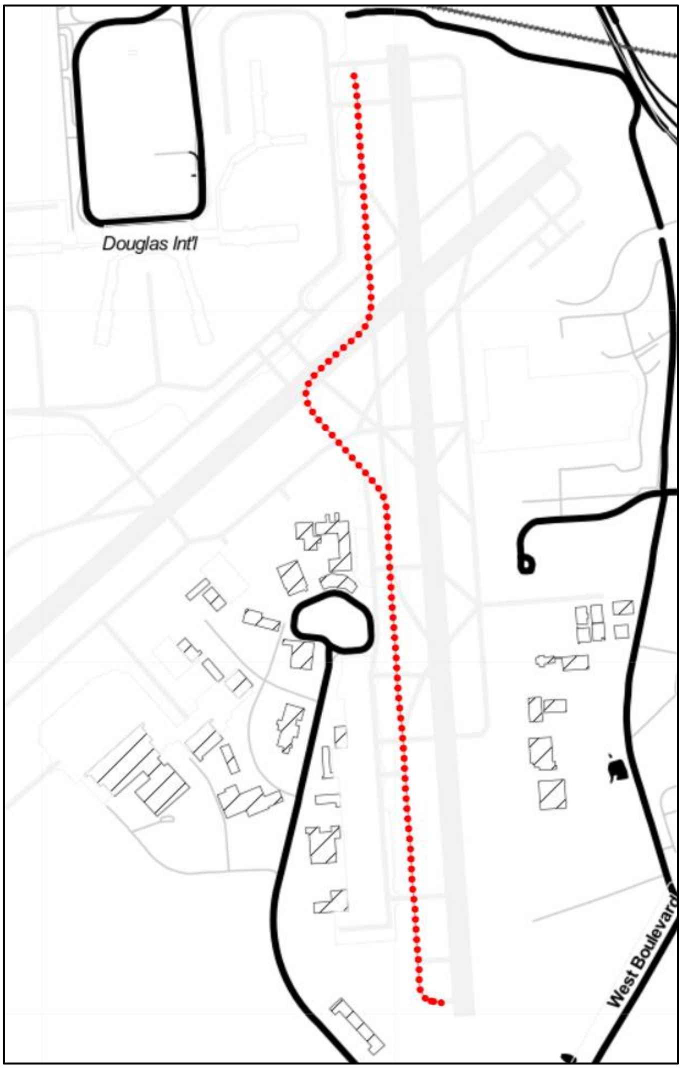

Figure 10. Example departure trajectory anomaly

In addition to these trajectories not explicitly assigned to a cluster by the algorithm, some of the small clusters may also represent anomalous behavior. Increasing the size of the input data, or increasing the `min_samples` parameter may help to separate some of these small clusters into more concrete "typical" or anomalous groupings. Recall that for this analysis, this parameter value was set higher for departures, and so there are no corresponding examples for departures to the arrival ones shown here.

For example, in Figure 11, we show four arrival trajectories (the minimum) that form one cluster. These flights do three unusual things on their route from runway 18R to the main ramp: first, they require a longer landing distance and exit on W3 instead of the typical W4, they use V4 to cross 18C instead of S, and they enter the main ramp at handoff spot 12S instead of 22W.

Figure 11. Sample small arrival cluster

In Figure 12, we show four arrival trajectories that form another cluster. These flights land on 36R and turn off on 23, as in one of the largest arrival clusters shown in Figure 5. However, instead of turning off 23 at R to enter the main ramp at handoff spot 25, the flights taxi much further down 23 and turn on B to M to enter the ramp at handoff spot 22W. This likely happened to avoid congestion in the southeast corner of the ramp near spot 25, but is unusual behavior.

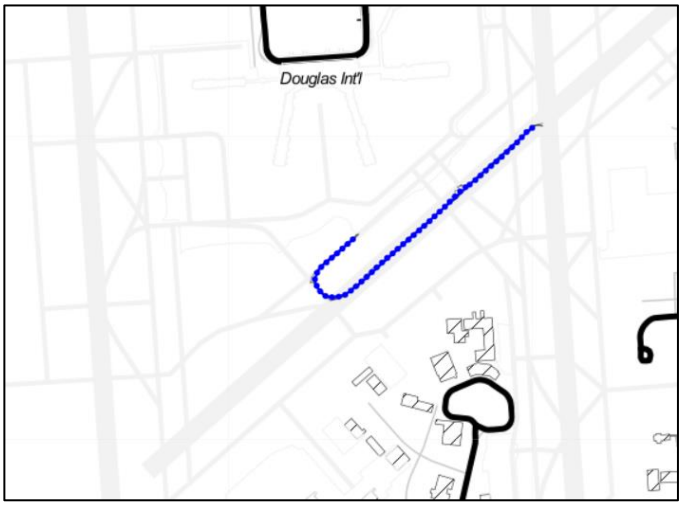

Figure 12. Sample small arrival cluster

Next, we examine the impact of the second level of our clustering hierarchy by looking within shape-based clusters to identify time-based characteristics. In Figure 13, we show the fraction of trajectories within each shape-based cluster that are not assigned to a time-based cluster, i.e., are an outlier in the second level of the hierarchy. From this figure, we can see that approximately 50% of shape-based clusters have 5% or fewer outliers when clustering in the time domain. In other words, within many shape clusters, there is considerable homogeneity with respect to time. However, because of this relative homogeneity, these clusters are particularly interesting from an anomaly detection perspective, as we will show next. Conversely, there are some shape-based clusters for which the time-based clustering found no structure, e.g., 6 of the 66 space-based arrivals clusters, and each had the minimum number of member trajectories (4). As a result, there is no value added in terms of being able to identify anomalies from the time-based clustering in these space-based clusters. This lack of structure provides one mechanism for categorizing entire space-based clusters as anomalous.

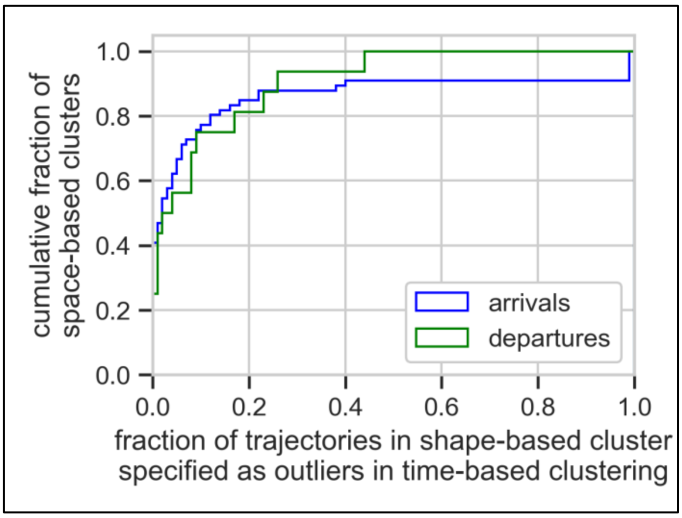

Figure 13. Time-based clustering behavior within shape clusters

Next, we show the time-based structure identified within each of the largest shape-based arrival and departure clusters. For this arrival cluster (Figure 14), only one time-based cluster is identified. A number of outlier trajectories are also identified (shown in red); each of these had stops at different locations and of different durations.

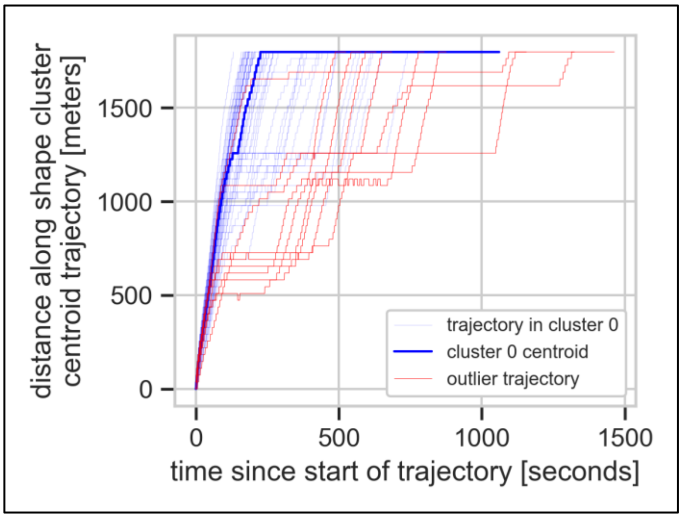

Figure 14. Time-based structure within largest arrival cluster

In the largest shape-based departure cluster (Figure 15), two time-based clusters are identified—one has minimal delay in reaching the runway, while the other represents trajectories that generally had multiple stops. Two other space-based departure clusters also had two time-based clusters within.

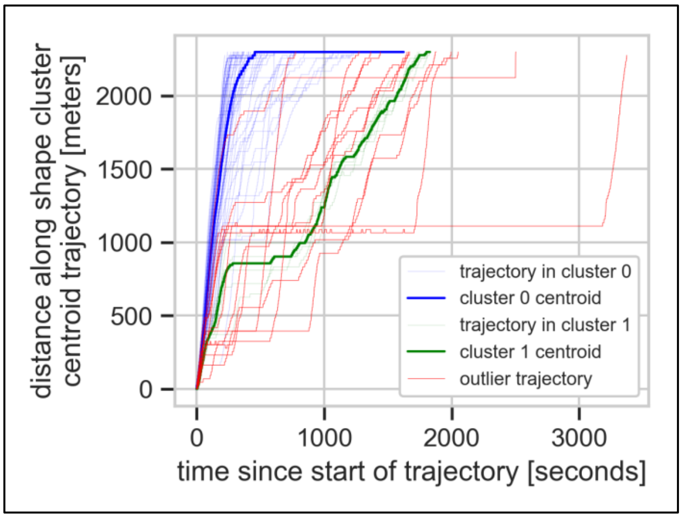

Figure 15. Time-based structure within largest departure cluster

In this figure, note that a number of trajectories were still left out of any time-based cluster because of the diversity in when and where flights stopped along their trajectory. In most cases, these stops do not seem to correspond with being in the departure queue (i.e., near the end of the taxi trajectory) but rather occur throughout. This distribution indicates that there is not likely a single hotspot of congestion causing these stops, but instead a variety of other causes.

Next, in Figure 16, we provide an example of the importance of employing the time-based clustering approach in identifying anomalous behavior by looking more deeply at one of the outliers from the previous figure. In terms of shape, this flight followed an expected, typical path. However, when clustering in time, it is exhibits anomalous behavior. At the point highlighted with the red circle, the flight stopped for approximately 25 minutes, as shown in time-space plot in Figure 17. Stopping at one point in the active movement area of the airport is clearly unusual, indicating that the flight was dealing with some maintenance problem or unexpected delay. In a real-time setting, this flight could be identified to a controller for further investigation, likely after verifying that there was no previously-known delay mechanism to explain the stop (e.g., ground delay program, ground stop, APREQ).

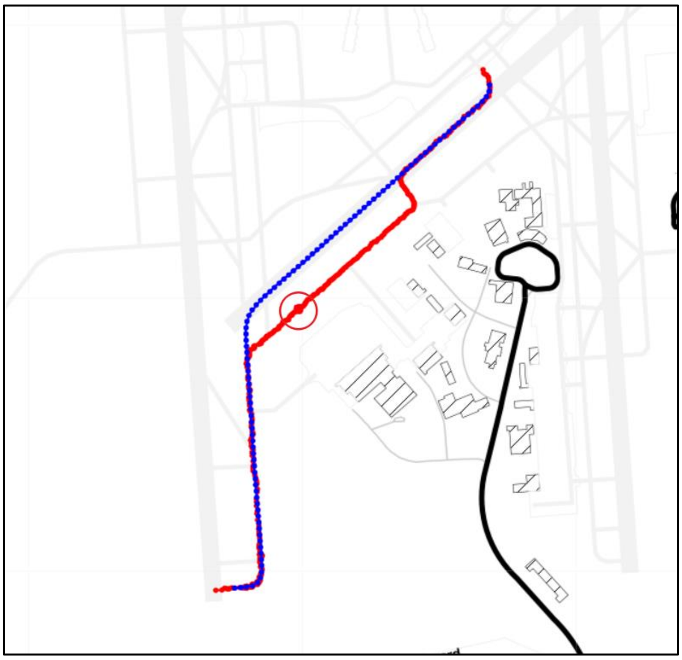

Figure 16. Example time-based anomaly

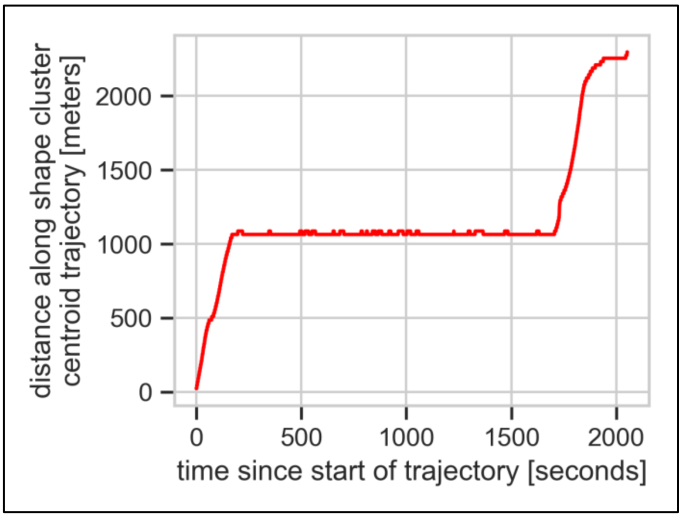

Figure 17. Space-time depiction of example time-based anomaly

In this section, we have shown the results of the analysis applying our proposed hierarchical clustering approach to a one-week data set. We believe that these results provide a compelling view into the potential insights available from this approach.

V. Continuing Work and Conclusions

In this paper, we have described our research adapting clustering techniques to develop meaningful results for taxi trajectories on the airport surface. We have demonstrated technical feasibility in applying them to the post hoc anomaly detection application outlined in the introduction. Trajectories are grouped into logically-reasonable spatial clusters, with commonly-used taxi routes appearing frequently, and together. Little-used routes and other anomalies are successfully segregated from the clusters representing typical paths. In the time domain, we can successfully separate taxi trajectories that represent unimpeded motion from those that stop, including identifying common stopping points. The time-based level of the hierarchy also provides value in helping to identify space-based clusters with little or no internal time-based structure, which may be a useful proxy for labeling entire clusters as anomalous behavior.

The techniques extend the state of the art and demonstrate promise for continued research and development. We have identified a number of areas for improving our technical approach, including:

1. Improved computational performance to support analysis over longer time periods
2. Refined approach to time-based clustering to recognize equivalence of stopping in same spot for some range of times
3. Inclusion of other aircraft data to refine clusters and diagnose performance

In this paper, we have focused on applying our clustering methodology to identify anomalous taxi trajectories in a post hoc setting, and have demonstrated the feasibility of this approach. However, we believe that there are a number of other valuable applications for these clusters that should be pursued. We intend to develop a real-time system leveraging this clustering approach to flag potentially-troublesome trajectories as they unfold. In addition, we believe this clustering approach can be very useful to inform other ongoing analysis and modeling efforts supporting other automation systems to improve airport safety and operational performance.

Acknowledgment

The authors gratefully acknowledge funding for this work from the National Aeronautics and Space Administration (NASA), under contract NNA16BF04C.

References

[1] Hastie, T., R. Tibshirani, J. Friedman, “The Elements of Statistical Learning: Data Mining, Inference, and Prediction,” 2nd edition, Springer, 2009.

[2] Yuan, G., P. Sun, J. Zhao, D. Li, C. Wang, “A review of moving object trajectory clustering algorithms,” Artificial Intelligence Review, Vol. 47, No. 1, 2017. doi:10.1007/s10462-016-9477-7

[3] Zheng, Y., “Trajectory Data Mining: An Overview,” ACM Transactions on Intelligent Systems and Technology, Vol. 6, No. 3, 2015. doi:10.1145/2743025

[4] Li, M.Z., M.S. Ryerson, “Novel terminal arrival airspace robustness metrics via topological density clustering,” ICRAT2018 conference, Barcelona, Spain, 2018.

[5] Gariel, M., A.N. Srivastava, E. Feron, “Trajectory Clustering and an Application to Airspace Monitoring,” IEEE Transactions on Intelligent Transportation Systems, Vol. 12, No. 4, 2011. doi:10.1109/TITS.2011.2160628

[6] Murça, M.C.R., R.J. Hansman, “Identification, Characterization, and Prediction of Traffic Flow Patterns in Multi-Airport Systems,” IEEE Transactions on Intelligent Transportation Systems (in press), 2018. doi:10.1109/TITS.2018.2833452

[7] Budalakoti, S., A.N. Srivastava, M.E. Otey, “Anomaly Detection and Diagnosis Algorithms for Discrete Symbol Sequences with Applications to Airline Safety,” IEEE Transactions on Systems, Man, and Cybernetics, Part C, Vol. 39, No. 1, 2009. doi:10.1109/TSMCC.2008.2007248

[8] Lee, J.G., J. Han, X. Li, “Trajectory Outlier Detection: A Partition-and-Detect Framework,” IEEE International Conference on Data Engineering, Washington, DC, USA, 2008. doi:10.1109/ICDE.2008.4497422

[9] Janakiraman, V.M., B. Matthews, N. Oza, “Using ADOPT Algorithm and Operational Data to Discover Precursors to Aviation Adverse Events,” 2018 AIAA Information Systems Conference, 2018. doi:10.2514/6.2018-1638

[10] Deshmukh, R., I. Hwang, “Anomaly Detection Using Temporal Logic Based Learning for Terminal Airspace Operations,” AIAA Scitech 2019 Forum, 2019. doi:10.2514/6.2019-0682

[11] Ester, M., H. P. Kriegel, J. Sander, X. Xu, “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”. Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, Portland, OR, AAAI Press, pp. 226-231, 1996.

[12] Sakoe, H., S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 26, No. 1, 1978. doi:10.1109/tassp.1978.1163055

[13] Stringer, M., V. Seghete, Y.P. Leong, “traces: A Python library for unevenly-spaced time series analysis,” https://github.com/datascopeanalytics/traces.

[14] Jones, E., T. Oliphant, P. Peterson, “SciPy: Open source scientific tools for Python,” 2001, http://www.scipy.org/.

[15] Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, et al, “Scikit-learn: Machine Learning in Python,” Journal of Machine Learning Research, Vol. 12, 2011.

[16] Rousseeuw, P.J., “Silhouettes: a Graphical Aid to the Interpretation and Validation of Cluster Analysis,” Computational and Applied Mathematics, Vol. 20, 1987. doi:10.1016/0377-0427(87)90125-7

[17] Federal Aviation Administration, Airport Surface Detection Equipment, Model X (ASDE-X), https://www.faa.gov/air_traffic/technology/asde-x/

Author Biographies

Andrew Churchill is a principal analyst with Mosaic ATM and Mosaic Data Science. In this role, he performs research and development applying data science techniques for U.S. government clients including FAA, NASA, and the National Academies, as well as commercial clients in a variety of verticals. He holds a Ph.D. in Civil Engineering from University of Maryland.

Michael Bloem is a principal data scientist with Mosaic Data Science. In this role, he leads and executes the design, development, and deployment of data science-enabled solutions with organizations in a variety of verticals. Michael researched air traffic management at the NASA Ames Research Center for eight years. He holds a PhD in operations research from the Department of Management Science & Engineering at Stanford University.