Airport Layout Updater Using Flight Surveillance Information

Tomek Gradzki,[*] and Dr. Susan Hotle[†]
Virginia Tech, Blacksburg, VA, 24061, USA

Accurate airport layout information is essential for airport surface movement research. This study evaluates if flight movements from the Airport Surface Detection Equipment - Model X combined with taxiway design standards from the Federal Aviation Administration can automatically update surface polygons on airport layouts and distinguish whether the new polygon is a taxiway or apron. The study updates the layouts for 6 United States airports. The Airport Layout Updater developed uses K-means to detect the taxiway centerlines before applying the widths and fillets based on the design aircraft that used that surface. New apron areas are detected where the transponders are turned on/off, indicating a gate-out/gate-in event. Airport surfaces are removed if they overlap with the runway polygons, which the Federal Aviation Administration updates every 28-days. The Airport Layout Updater had a 98.8% success rate of detecting a new surface and 81.0% of generating a surface that accurately represents the bearing, shape and size of the real-life surface. Future research could improve the fillet generation algorithm by differentiating between non-existing fillets and those that exist but were not used by flights. Similar limitations were found for apron areas, where flights do not travel near the apron perimeter.

I. Introduction

DUE to the increasing number and size of planes, United States (US) airports originally built in the 1950s-1970s frequently modify their aprons, taxiways and runways. It is vital to have updated records of airport layouts to efficiently manage traffic, keep track of pavement conditions, and perform taxi pattern analyses. Despite this necessity,

[*] Graduate Student, Department of Civil and Environmental Engineering, tomekg@vt.edu
[†] Associate Professor, Department of Civil and Environmental Engineering, shotle3@vt.edu

there is currently no system in place to automate surface layout updates with pavement categories (e.g. apron area, taxiway, runway) in a format useful to Geographic Information System (GIS) software.

This study analyzes the accuracy of automatically updating airport surface layouts using only outdated airport layouts, flight position data and the Federal Aviation Administration (FAA) design standards. It creates the Airport Layout Updater (ALU), a Python program that takes flight data compiled by the Airport Surface Detection Equipment — Model X (ASDE-X) to update airport GIS layouts. ASDE-X collects information including a flight's airline, aircraft, spatial location, bearing, origin, and destination [1]. When airplane movements are combined with airport layout information, instances where flights use surfaces that do not exist in the layout can be identified.

Therefore, this study first creates algorithms for detecting locations of missing surface infrastructure as well as identifying the design aircraft and the type of missing infrastructure, including apron areas, linear taxiways, non-linear taxiways, and present or absent fillets. The thresholds in these detection algorithms are based on experimenting with the data across the study airports. This study then evaluates the accuracy of automating airport layout updates by taking these "flight off-roading" events with the taxiway and apron design principles in FAA Advisory Circular 150/5300 [2]. This Advisory Circular regulates the apron dimensions, minimum taxiway width and fillet dimensions based on the design aircraft. New construction on any airport located in the United States is required to follow these design specifications. For unique circumstances where an airport cannot meet a design standard, such as due to limited land availability, airports can file a Modification of Standards (MOS) to the FAA for a case-by-case review for design approval. Ultimately, the goal of this study is to create a single program that could most accurately update the layout of any airport in the United States and to measure its accuracy.

II. Literature Review

Mapping transportation networks using person or vehicle movements has been conducted across multiple modes of transportation. This includes application to mining road networks [3], road networks (thorough literature review of automated road network mapping by Badran et al. [4]), pedestrian networks [5, 6], pipelines [7] and railway networks [8]. Several methods have been tested to map transportation infrastructure using Global Positioning System (GPS) movement data, predominately fitting into two categories: density-based map construction and movement-based map construction [9]. Density-based map construction takes "the density distribution of all the trajectory position samples and extract the density peak lines with a maximum local sampling density" [9]. Density-based methods include kernel density estimation (KDE), which is a grid-based method, and point clustering, which is a non-

grid-based method that sometimes uses K-means clustering. Movement-based map construction "utilize the movement aspect of the trajectories, i.e. not just a point cloud, to construct a road network and preserve those features in the resulting network" [9] and construct the network incrementally. Both methods create a link-node network (e.g. road centerlines) which can be expanded by applying lane widths to create map polygons.

Applications of map construction algorithms on airport surface networks is limited in literature, where most use satellite imagery of the airport and only update the runway surfaces or culminates with a taxiway node and link network, overlooking taxiway width and geometry characteristics. These characteristics are important to navigation and should be accurate as the network link widths dictate which aircraft types can travel each route.

Han et al. [10] identified runway boundaries from satellite imagery using edge tracing and speeded-up robust features (SURF). By smoothing an image and analyzing changes in color gradient and pixel angle, the study identified runways through this means due to their straight nature. However, this approach cannot work for more complex surfaces such as taxiways and aprons. This process does not distinguish between runways and other linear, mono-colored surfaces, meaning false positives are likely for other surface types.

Jackson et al. [11] proposed an alternative edge tracing method using satellite imagery that improved the software's ability to differentiate runways from other linear surfaces. Jackson's method identified regions of interest (ROIs) in the image, determined by the location of runway centerlines and runway markings. This method identified runways with a 0% false positive rate when tire markings are considered. However, this method cannot apply to taxiways and aprons due to the method working only for straight segments that are clearly marked.

Kim et al. [12] used airport surface images captured by an unmanned aircraft system (UAS) to create an algorithm that parsed the images to check runway design compliance with the FAA regulations. It used the runway centerline width to extrapolate other distances, such as runway width, runway length and runway centerline separation from the nearest parallel taxiway centerline. Similar to Han and Jackson, this study only considered infrastructure near the runway and not the taxiways. It was tested on the one runway at Roosevelt Memorial Airport in Atlanta.

The described approaches exclude taxiways and rely on the runway's physical characteristics to differentiate from other surfaces like taxiways or aprons. Non-runway surfaces have less linear arrangements and cannot easily be differentiated by appearance. The only paper on automating taxiway polygons, not just link-nodes, is Pschierer et al. [13]. The authors proposed a proof-of-concept approach to generate a link-node structure of the Atlanta-Hartsfield-

Jackson Airport (ATL). This approach manually determined the links, nodes and taxiway and intersection geometries using airport satellite images.

The purpose of this study is to automate airport layout updates by using ASDE-X flight surveillance position data. Thirty-five airports in the US are equipped with ASDE-X sensors [1], emitting the second-by-second positions of aircrafts on the airport surface. This data is available through the FAA System Wide Information Management Application Programming Interface (SWIM API) SWIM Terminal Data Distribution System (STDDS) feed [14, 15]. In addition to the time and position, ASDE-X data contains the flight ID (e.g. DAL101) and aircraft type.

In this paper, we use the K-means algorithm to follow a density-based map construction method. Our study not only updates the links and nodes of the airport network, but also applies the characteristics of the design aircraft (largest taxiway design group aircraft to travel the link) to automatically determine the link width and intersection geometry. This means the width of the polygons generated in our study are accurate to the design regulations put forth by the FAA and not a simple "bundling" of trajectories to determine the link thickness, as was done by Karagiorgou and Pfoser [16, 17]. Our study overcomes the limitations of previous research by updating all surface pavements (runway, taxiways, and apron areas) including all geometry features (pavement width, fillet radius and intersection geometry).

III. Data Selection

This study uses .mat files that contain outdated airport layouts for the following 6 airports: Charlotte Douglas International Airport (CLT), Newark Liberty International Airport (EWR), John F. Kennedy International Airport (JFK), LaGuardia Airport (LGA), Chicago O'Hare International Airport (ORD), and Philadelphia International Airport (PHL). This data was collected by hand-mapping the airport surfaces using satellite imagery in 2012 and it categorizes 3 surface types: aprons (surfaces near the gates), runways (surfaces used to take off or land), and taxiways (surfaces used to move between aprons and runways). These files were provided by the FAA and are used for the Taxi Event Extractor (TEE) program, as highlighted in Mirmohammadsadeghi, et al. [18] and Li, et al. [19]. The ASDE-X flight trajectory data was collected from SWIM, cleaned, and provided by the FAA Office of Performance Analysis. Using the Python Matplotlib package, flight points outside all 2012 surfaces were detected (Figure 1). Simply creating new surfaces using all flight pings outside of the file's surfaces would not be accurate as a stationary flight can have signal interference (Figure 1 box).

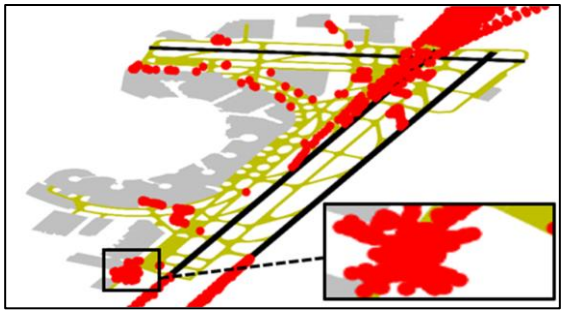

Fig. 1 Flight Pings Outside Existing Surfaces at EWR

IV. Methodology

Before new surfaces can be generated, the surfaces that no longer exist are removed from the 2012 airport layout files. While it is simple to remove any surfaces that do not experience any traffic, this runs the risk of accidentally removing surfaces that were not used during the data timeframe. Therefore, the program only removes surfaces that conflict with the runway polygons provided by the FAA 28-day subscription [20], which are updated every 28 days.

Flight pings occurring outside of surfaces in the 2012 airport layouts are used to create new surfaces. There are 3 primary criteria that all must be met for a given flight data point can be considered part of a new surface: 1) the flight's location is outside of all existing surfaces on record, 2) the flight being outside is not due to scatter caused by interference and 3) enough data exists to suggest that multiple flights used the potential new surface.

This study is only interested in new taxiway and apron surfaces as the runways are provided by the FAA every 28 days. The first criterion for new surfaces is that a plane is traveling less than 20 m/s outside of an existing surface, which removes movements on a runway. The second criterion determines when flights are stationary while accounting for signal interference. An aircraft's current location is compared to its location in 15 to 25 seconds. If any of the pings in that 10-second period are within 44 m of its current location, the flight is considered stationary and the point is removed from consideration. These parameters were determined by experimenting with the data highlighted in Figure 1 until only the points in the boxed area were removed. The final criterion requires that enough points are grouped to define a unique surface as illustrated in Figure 2. Point groups containing less than 15 points are discarded. The final set of "Point Groups" are shown in Figure 2.

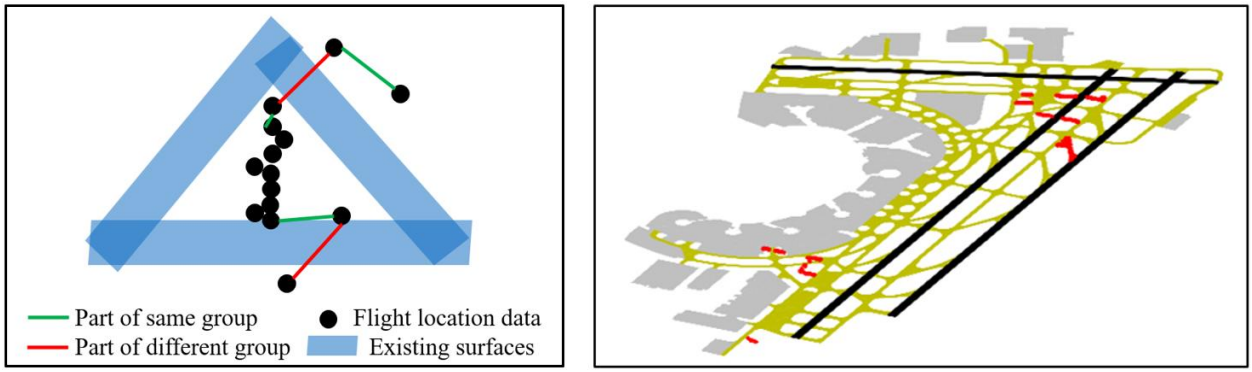

Fig. 2 Grouping Points by Using Existing Surfaces (Left) and Point Groups (Right) at EWR

While the point groups organize the flight pings bound by the same set of existing surfaces, it does not differentiate if there are multiple groupings within the boundary. To divide the point groups into individual surfaces, the flight pings that are geographically near each other must also be close to each other within the list of data in the point group. Therefore, the flight pings in each point group are sorted from south to north and then from west to east. Then the circular area around each point is analyzed. The radius of this area is determined by the group's proximity to existing apron surfaces (37 m if the point group is near an existing apron, 263 m if the point group is not near an existing apron). If none of the previous 15 points are within the circular area, then the point is part of a different surface group than the points before it. A new group must contain more than 15 points to be considered a group. Figure 3 shows how new apron surfaces tend to be near existing aprons and have denser point distributions than taxiways.

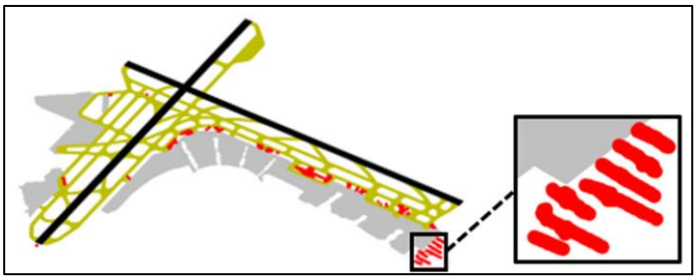

Fig. 3 Taxiway and Apron Point Distributions at LGA

To ensure the data split properly, a second check occurs by looking at the northmost, southmost, eastmost, westmost cardinal points for each point group. Lines are drawn between each pair of cardinal points to form a rough shape of the group. After determining points groups, the groups are divided into four distinct categories:

1. Linear Single-Segment Taxiways (LSSTs): Simple, straight taxiways that directly connect two existing surfaces with little deviation in width or bearing.
2. Non-Linear Taxiways (NLTs): Taxiways that deviate from a linear path by at least 15 degrees or otherwise

have characteristics that result in the point group failing to fit in a linear approximation.

3. Apron: Larger surfaces that typically surround terminals and consist of point data that travels in multiple directions and at varying speeds.
4. Extension: Additional part of existing surface meant to increase width or fillet size. A fillet is added pavement at the intersection of two taxiways, making the corners more rounded in shape.

A. LSSTs

The ALU requires two components to determine the shape of an LSST: the surface direction/bearing and the centroid. The SciKit-Learn Python package is used to perform K-means clustering on the point groups [21]. K-means is a machine-learning algorithm that divides a group of points into "n" clusters, where n is determined by the user. The K-means function used for an LSST surface was for 5 clusters, a maximum of 300 iterations.

The centroids of the first and last clusters are nearly on the boundary of the existing surfaces closest to the point group and are used to determine the LSST centerline and bearing (shown in blue in Figure 4). Next, the edges of the taxiway can be determined by creating a buffer around the centerline. The buffer is determined using the Taxiway Design Groups (TDG) of the plane with the largest required taxiway width that used the polygon. This results in an approximation of the LSST, as shown in red in Figure 4. The program opts to use the midpoint of the centerline as the centroid instead of the average of the points. This minimizes the effect of scattering as the aircraft moves down the LSST (notable "spikes" along edges).

A centroid will not lie along the true bearing of the taxiway if there is a fillet, such as in Figure 4. If the 1) angular difference between either the first two segments or final two segments are greater than 45 degrees and 2) difference between the second and second to last segments is less than 30 degrees, then the centroid immediately following or preceding the offending centroid determines the bearing. This is shown in Figure 4 by how the red boundaries do not follow the blue line connecting the first and last centroids in the illustrated taxiway.

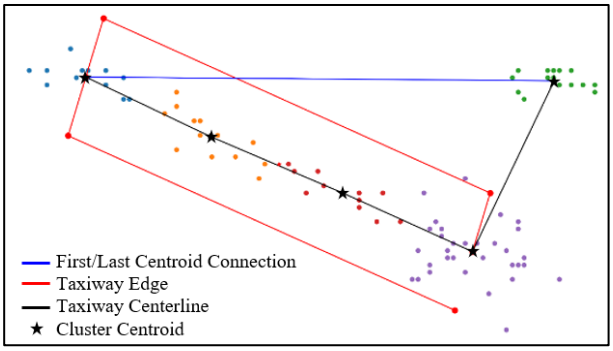

Fig. 4 Example of LSST Where Final Centroid is Separate from Main Taxiway

Next, the ALU expands the LSST to generate a surface. It extends the LSST to the nearest existing surfaces by shifting each of the four corners and the two endpoints along the approximate centerline in 9-meter increments until the point encounters an existing surface. The endpoint is then moved by 0.9-meter increments, then 0.09-meter increments. The process ends when the surface fails to include points in the point group for 5 increments, such as when an LSST does not end at an existing surface. If a substantial portion of the LSST overlaps an existing surface, the surface is an extension and does not undergo the fillet generation process. The results are shown in Figure 5.

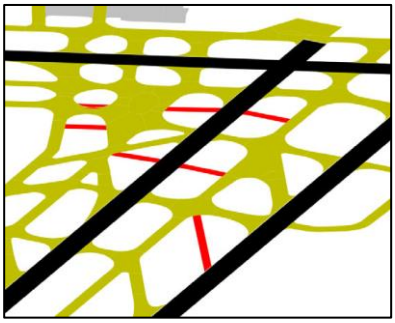

Fig. 5 LSST's at EWR

Fillet dimensions are dictated by the FAA's TDG categories and are affected by the angle of intersection between the two surfaces and whether one of the surfaces is a runway. Since the intersection angle can vary depending on the number of inlets into the intersection, the LSST fillet points are determined separately for each corner. The intersection angle is taken as the angle between the bearings of the centerline and the line connecting the endpoint and corner under consideration. If an intersecting surface is a runway, the fillet dimensions need to account for the increased speed of aircraft heading onto and off the runway. The tables for the basic dimensions as determined by intersection angle and tables for the adjustments due to being a high-speed exit taxiway come from Advisory Circular 150/5300 – 13B [2]. The taxiway and fillets are then combined into a single polygon, as shown in Figure 6.

The fillets overlap existing surfaces and should be adjusted. Each point in the LSST polygon is checked to see if the line between the point and the preceding point intersects any existing polygon. If it does, the point is moved along the line in 0.09 m increments until the line segment no longer intersects any surface. If after 200 increments, the line still intersects an existing surface, then the point is reset to its original position and will move along the line connecting itself and the previous point in the taxiway. This continues until the intersection issue is resolved. The final step is to fix the taxiways connections to the existing taxiways with no space in between. The program generates 9 equally spaced points on the edges of the taxiway (blue dots in Figures 6) and shifts them until they encounter a surface.

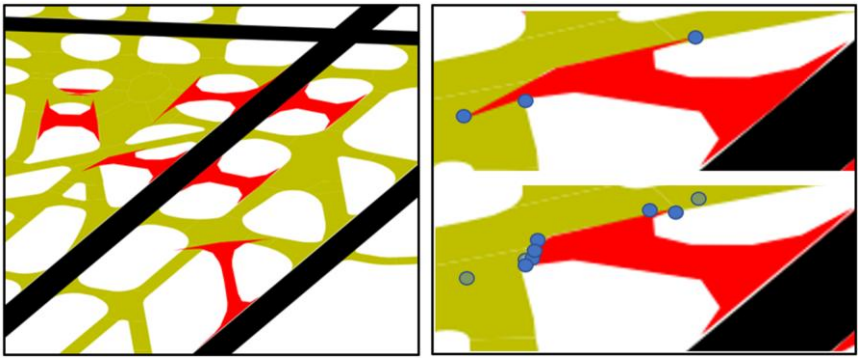

Fig. 6 Initial LSSTs with Fillets (left) and Polygon Adjustment Process (right)

B. NLTs

Additional processes are needed for complex surfaces such as aprons and NLTs. The program differentiates between these two surface types using two factors: the surface angularity and the percentage of points that fall inside a linear approximation. For surface angularity, the centroids generated by using K-means give a rough estimate of the path of the surface. For the purposes of this research, a surface group containing 2 or more angles greater than 45 degrees is considered either an apron or NLT.

For linear approximation, rectangular surfaces were created that begin and end at the boundary of the existing surface. Most points in a group representing an LSST will fit inside a rectangular surface with the length, width, and orientation defined in the LSST processes. Conversely, NLTs will have more points inside the rectangular surface, while most points in an apron group will be outside the rectangular surface, as shown in Figure 7.

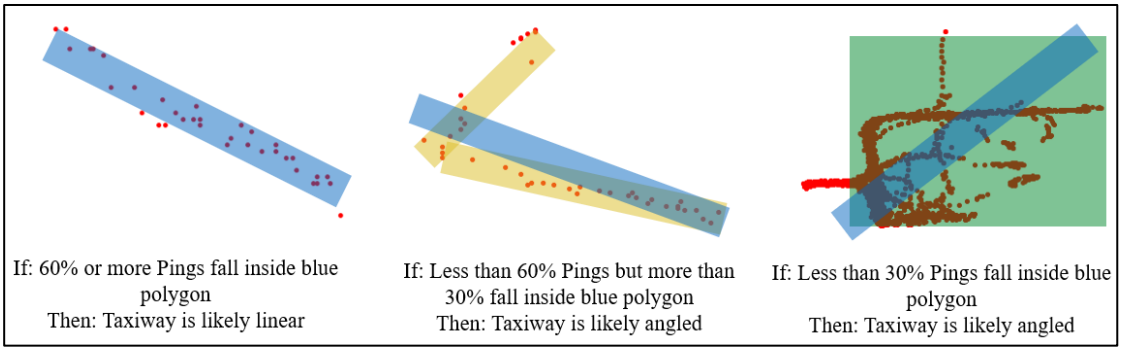

Fig. 7 Surface Fit to the Rectangular Approximation

The overall process of determining the dimensions of NLT's is similar to how LSST's are determined. K-means clustering centroids determine the initial polygon boundary with the TDG-determined widths. Where the NLT process differs is how many points are generated per surface and how this process handles intersecting taxiways. Unlike the LSST process that used 5 K-means clusters, the NLT process defines the number of clusters based on the surface shape complexity. Using too many clusters on a simple taxiway or one that does not have enough data points can result in a fluctuating bearing, resulting in a ragged-looking taxiway (See Figure 8). Using too few clusters on a complex taxiway can result in an oversimplified polygon. The equation used to determine the exact number of clusters used for an NLT polygon is as follows:

$$N = 10 + 0.003X + 5Y$$

Where:

X = Length of Taxiway in Feet

Y = Number of Interior Angles Greater Than 45-Degrees

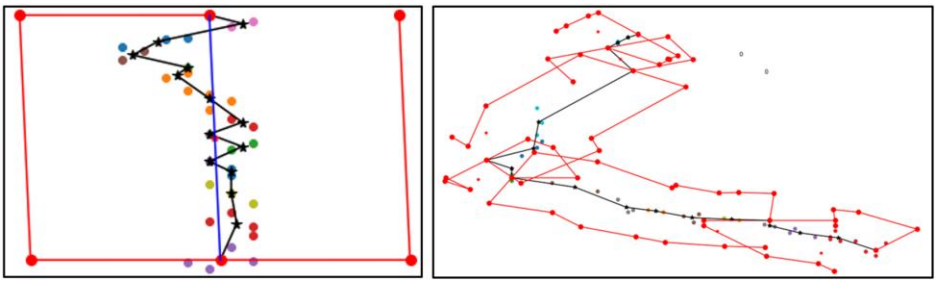

Fig. 8 Too Many Clusters on Simple (left) and Poor Data (right) Taxiways

Next, the centroids are ordered from lowest to highest latitude/longitude, then lowest to highest longitude/latitude depending on the taxiway orientation (north to south taxiways are ordered latitude first, then ordered by longitude). The bearings are determined from each centroid to the next centroid following the previous order. If any consecutive

bearings have an angular difference greater than 45 degrees, such as the added taxiway in the southeast part of CLT airport, this point is flagged as the intersection of two taxiways to be split into two distinct taxiways later in the process.

After the NLT polygon generation, the fillet process is almost identical to that used for LSSTs except that ends of an NLT do not need bearings in the same or opposite directions. The program looks at the list of bearings generated during the creation of the initial polygons and takes the first and last bearings for the extension process. The extension process for NLTs uses two bearings while the process for LSSTs uses one bearing and simply adds/subtracts 180 degrees. At this point, the ALU has finished generating new surfaces and appends them to the .mat layout file.

C. Aprons

The ALU then defines the apron boundaries. While it makes sense to progress from the least to most complex surfaces, aprons are determined second for one important reason; taxiways that connect the new aprons to the existing layout may accidentally be included in apron groups, such as the one shown in Figure 9. Because the process of defining NLT segments is similar to defining LSSTs, it is more efficient to determine the apron boundaries, isolate the points that represent a connecting taxiway, and process them along with the NLTs.

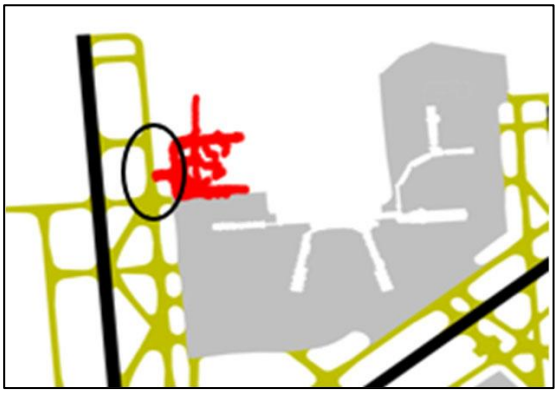

Fig. 9 Apron Group with an LSST for CLT

For aprons, the centroid is determined by averaging the coordinates of all flight pings in a potential new apron. The centroid provides a valuable apron reference point that determines other points' relationships to the apron as a whole. It is also possible to define the aprons by including data that was previously filtered out due to scatter. Points previously removed due to scatter are added to apron groups depending on whether they meet the following criteria:

1. The distance between the point and apron centroid is less than or equal to 0.0035 degrees latitude (1008.7 m)
2. No surfaces (taxiways, runways, current aprons) interrupt the line between the considered point and the centroid

Existing aprons can be used to determine the boundaries of an adjacent new apron. Adjacent aprons are identified

by slightly increasing the size of the existing apron and checking to see if any points in the prospective apron group are inside. Then, the points on the existing apron closest to the prospective apron are put into a separate group.

To align new aprons properly with existing aprons, a simple quadrilateral is first created by using the geometry of the existing apron. The ALU identifies the closest corners of the adjacent apron and the bearings of the new apron's boundaries. The first step is to determine the bearing of each line segment in the group of apron points. Then, the relative differences between each of these bearings are calculated and the line segments making up a corner are identified if the following patterns occurs among three consecutive segments:

1. The difference between the bearings of the first two segments is less than 1 degree (two nearly colinear line segments) while the difference between the bearings of the second and third segments is greater than 10 degrees.
2. The difference between the bearings of the first two segments is greater than 10 degrees while the difference between the bearings of the second and third segments is less than 1 degree (two nearly colinear line segments).

If either of these patterns hold, then it is likely that the point is a corner of the apron. Since multiple points make up the boundary of any one side of the existing apron, notable corners can be identified simply by looking at the relative differences in the angles at each point.

Next, the apron boundaries are identified. The new apron should contain nearly all the points in the new apron data set. Figure 10 shows how the two new apron surfaces cover taxiways connecting to the aprons. The borders of the aprons need to be shifted to exclude the points that make up the taxiways. The process begins by taking the existing corners of each apron and projecting points a small distance (0.9 m) away from each corner in each of the directions that the apron boundary runs. These points are illustrated in Figure 10. The program then determines where the boundaries of the apron need to be moved to exclude the connecting taxiways. The shrinking process stops when either the 1) number of points excluded after the polygon shrinking exceeds 80 or 2) new apron edge coincides with the adjacent existing apron edge. Points outside new apron boundaries are later converted into a taxiway.

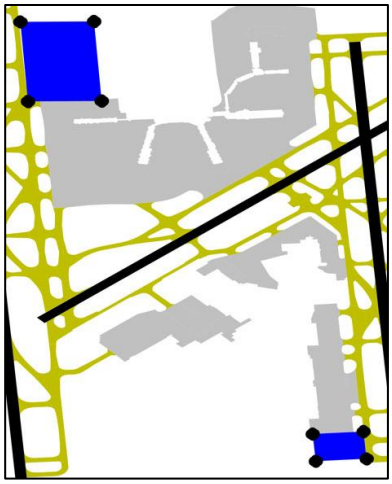

Fig. 10 Points Projected from the New Apron Corners

The final step in apron creation is to account for the terminal buildings. The program extracts and separates the first (gate-out) and last (gate-in) track points of all flights within the new apron. The locations are where the transponders are turned on/off close to the terminal, as shown in Figure 11.

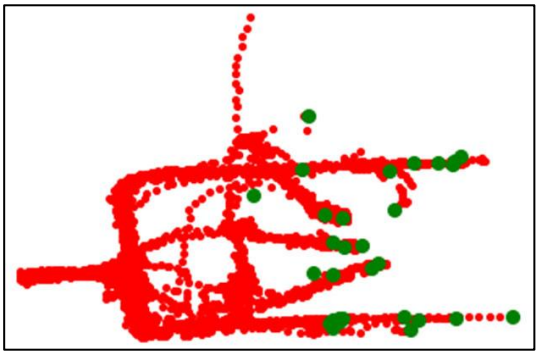

Fig. 11 First and Last Flight Points within a New Apron in CLT

Endpoints within 0.9 m from the edges of the apron are eliminated. As shown in Figure 12, several endpoints occur in an empty space. Therefore, these points need to be isolated using a radar approach, where it defines a circular wedge with an interior angle of 15 degrees that has a radius of 3 m for each endpoint under consideration. It then rotates this wedge around the endpoint and determines the number of track points in it. In this way, it can be determined if the endpoint should be kept, as illustrated Figure 12.

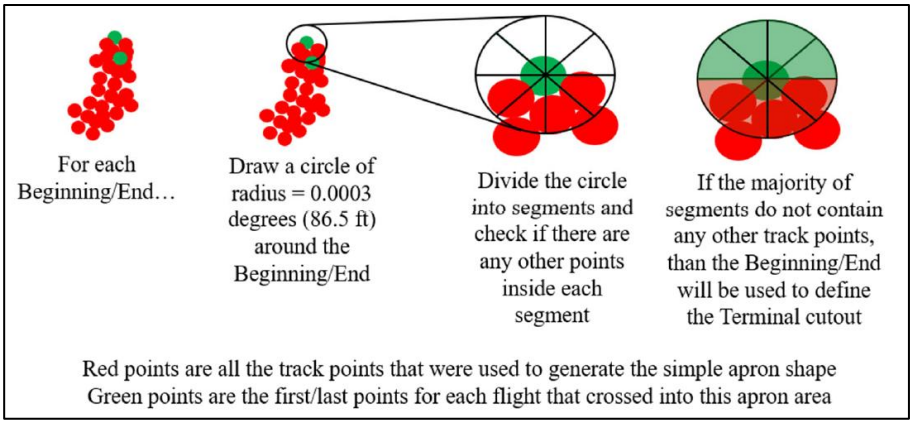

Fig. 12 Rotated Wedge Process used to Identify True Flight Endpoints

If any flight endpoint is further than 5 m from any other flight's endpoint, it is eliminated from consideration. The remaining endpoints are moved slightly inward to accommodate flights that turn off their transponders before coming to a complete stop. The resulting points are then bound to the appropriate side of the apron, as shown in Figure 13.

Fig. 13 Terminal Cutout within a New Apron

V. Results

The ALU generated updated layout files for time periods in Table 1. Figures A1-A6 in the Appendix compares the program's updates to the 2012 layouts, GoogleMaps satellite imagery [22] and the FAA diagram [23]. The Updated 2023 Layout for each airport shows the modifications made by the ALU. Added pavement, whether through new taxiways, widened taxiways or expanded aprons are shown in darker grey than the pavement already present in the 2012 layouts. For example, the ALU extended the apron area in the southeast part of LGA (Figure A4). To measure the program's ability to generate accurate surface data, the newly generated surfaces were analyzed using three different criteria to determine the program's accuracy: surface location, surface shape and orientation, and surface detail.

Table 1 ASDE-X Data Summary

Airport	Time Periods (UTC)	Time (Hours)	Flights	Flight Pings
CLT	11:28-16:13 02/22/2022	13	306	266,638
	08:59-16:23 11/01/2022		550	560,385
EWR	12:04-15:23 02/08/2022	11	162	157,229
	12:14-15:31 02/15/2022		151	202,657
	11:29-15:59 02/22/2022		158	366,614
JFK	10:59-15:47 02/22/2022	5	202	194,913
LGA	11:41-16:23 02/22/2022	5	234	210,064
ORD	07:41-10:30 12/12/2022	3	11	11,520
PHL	11:55-15:39 02/22/2022	4	124	83,151

A. Surface Location Analysis

In total, the ALU generated 78 taxiways and 6 aprons. A breakdown of the exact number of taxiways and aprons generated in each airport and whether the surface accurately represents a real newly constructed surface, a real extension to an existing surface, or was incorrectly added can be found in Table 2. The overall success rate was 98.8% (83/84) for determining whether a group of points represented an addition to an airport's layout. One limitation to this approach is that some existing surfaces naturally receive little or no flight traffic. For example, the northwest apron in CLT is smaller in the output than it is in real life because few flights used it during the study time period.

Table 2 Surface Generation Accuracy

Airport	New Taxiway	Taxiway Extension	No Basis Taxiway	New Apron	Apron Extension	No Basis Apron	Total
CLT	8	0	0	2	1	0	11
EWR	11	2	0	0	2	0	15
JFK	3	5	1	0	0	0	9
LGA	2	8	0	1	0	0	11
ORD	22	1	0	0	0	0	23
PHL	9	5	0	0	0	0	14
Total	55	21	1	3	3	0	84

B. Surface Path and Orientation Analysis

While the program was able to accurately determine the location of new additions to an airport's layout, that does not guarantee that the generated shapes resemble the real-life surfaces. To accurately represent the surfaces, the generated surfaces should be oriented in the correct direction with minimal angular difference and resemble the overall shape. Every updated airport was compared to its real-life layout for the following metrics: 1) the difference between the bearing of generated and actual surface does not exceed 5 degrees, 2) the path of generated surface matches path of corresponding existing surface and 3) the width of generated surface does not exceed width of corresponding existing surface. The results of this analysis are summarized in Table 3, with 81.0% (68/84) of surfaces generated

having no issues with representing the bearing and size of the real-life surface.

Table 3 Breakdown of Surfaces Generated by the ALU by Type of Flaws Present

Airport	Surfaces with Bearing Difference > 5 degrees	Surfaces that do not Match Existing Surface Path	Surfaces with Larger Width than Existing Surface	Surfaces with no Notable Issues	Total Number of Surfaces
CLT	0	0	1	10	11
EWR	0	0	0	15	15
JFK	1	1	0	8	9
LGA	2	7	0	2	11
ORD	0	0	1	22	23
PHL	0	3	0	11	14
Total	3	11	2	68	84

There is one clear primary issue among the surfaces that had notable issues. The taxiways that were constructed between two parallel taxiways with less than 50 feet space between them usually resulted in generating a taxiway that was too wide. This was not from applying the wrong width, but rather how the fillets are generated for the ends of the taxiway. The fillets for each taxiway were generated under the assumption that every taxiway was designed to accommodate turns for the aircraft type with highest TDG number in that taxiway group for all directions.

When combining the fillets to form a relatively short taxiway, sometimes the fillets overlap with one another, creating a very wide taxiway, much wider than in reality (see Updated 2023 CLT Layout in Appendix). Unfortunately, there is no way to tell whether a specific surface does not exist or if it exists but is simply unused by aircraft. Beyond this one issue, the generated surfaces not only matched the locations of existing surfaces, but also matched the orientations and paths of the existing surfaces while not exceeding their width with an 81.0% success rate. Most of the problematic surfaces were the LGA apron extensions. Considering how the point paths through these surfaces are mostly linear and short, the program is unable to differentiate these point groups as aprons.

C. ALU Performance

The ALU exists as an alternative to the previous method of determining surface boundary points. While it clearly takes less time than the process of hand-determining the coordinate points for non-runway surface boundaries, it is important to know its efficiency. As such, the parameters of the data used to generate each new airport layout, as well as the time that the program took to perform the update is compiled in Table 4. The total amount of time needed to update each airport was heavily dependent on the amount of data being analyzed. Also, the code run time is dependent on the number and complexity of the surfaces that need to be generated considering how ORD (the airport with the most added surfaces) is the only case where majority of time was spent generating polygon data which spans the

processes from Data Grouping to 2nd Round of Taxiways.

Table 4 Algorithm Duration by Process and Airport in Minutes

	CLT	EWR	JFK	LGA	ORD	PHL
Initial Data Load	1.27	1.48	0.34	0.33	0.02	0.12
Data Filtering	30.76	38.50	13.65	5.11	1.06	1.90
Existing Surface Removal	0.01	0.00	0.00	0.00	0.00	0.00
Data Grouping	0.32	1.36	4.82	0.39	1.85	0.16
1st Round of Taxiways	0.17	0.03	0.88	0.25	1.39	0.10
Linear & Non-Linear Taxiway/ Apron Separation	0.06	0.07	0.09	0.06	0.08	0.06
Aprons	0.01	0.02	0.02	0.01	0.02	0.02
2nd Round of Taxiways	0.06	0.07	0.08	0.07	0.86	0.06
MAT File Update	0.03	0.05	0.09	0.03	0.07	0.04
Total Minutes	32.69	41.57	19.96	6.25	5.35	2.45

VI. Conclusion

This is the first study that generates airport surface polygons from flight surveillance information and the FAA design criteria while categorizing them as taxiways and aprons. The updated runway polygons are already provided by the FAA's 28-day subscription [20]. The results generated by the ALU prove that it is not only possible to locate missing surface data using flight-level data, but that it is possible to create new surfaces that closely mimic the real surfaces that they are based off in less than an hour. Additionally, performance analysis would suggest that the ALU can be optimized even further for both time and accuracy.

This study contributes to literature as it shows that it is possible to use algorithms to update airport network infrastructure layouts using aircraft movement and label the pavement as a taxiway, runway or apron area. However, the more cosmetic pieces of the layouts that are not as important to operations research, specifically the fillets geometry, will have a reduced accuracy given the changing design rules regarding their dimensions over time. This means future algorithms for airport layout updates will need to expand upon the current automatic network update methods in literature to account for the noise in the position data and the dimensions of fillets depending on more than the aircraft size and the angle between taxiways.

VII. Limitations and Future Research

This study's limitations include that sometimes the ALU exaggerates the dimensions of fillets. This is because it uses the dimensions in the Advisory Circular 150/5300-13B [2] for the maximum TDG aircraft traveling the surface to determine fillet sizes. However, not every taxiway corner has a fillet. This could be improved by using a longer time period or, as suggested by Milne and Watling [24], having a continuous flow through of the FAA SWIM data.

Additionally, the algorithms that determine the apron boundaries use a quadrilateral approximation. Sometimes an apron segment is circular or complicated (See EWR apron boundaries in Appendix). Considering how important the distinction between aprons and taxiways are in terms of responsibility for delay (traditionally the airlines are responsible for delay in the aprons and FAA in the taxiways), this is an area that needs improvement. It is difficult to say exactly how this should be implemented given that flights typically do not travel around the apron edges.

Finally, the program should ultimately be capable of not only adding new surfaces, but also removing all old surfaces that are not currently in use. While it is simple to remove polygons that contain no pings, it is more difficult to remove polygons that are only partially used, such as when a new taxiway intersects with an older decommissioned taxiway. A possible solution would be to determine the amount of surface that is used by comparing the coordinates of the relevant flight data and the polygon to see if the polygon area used falls under a percentage threshold.

Appendix

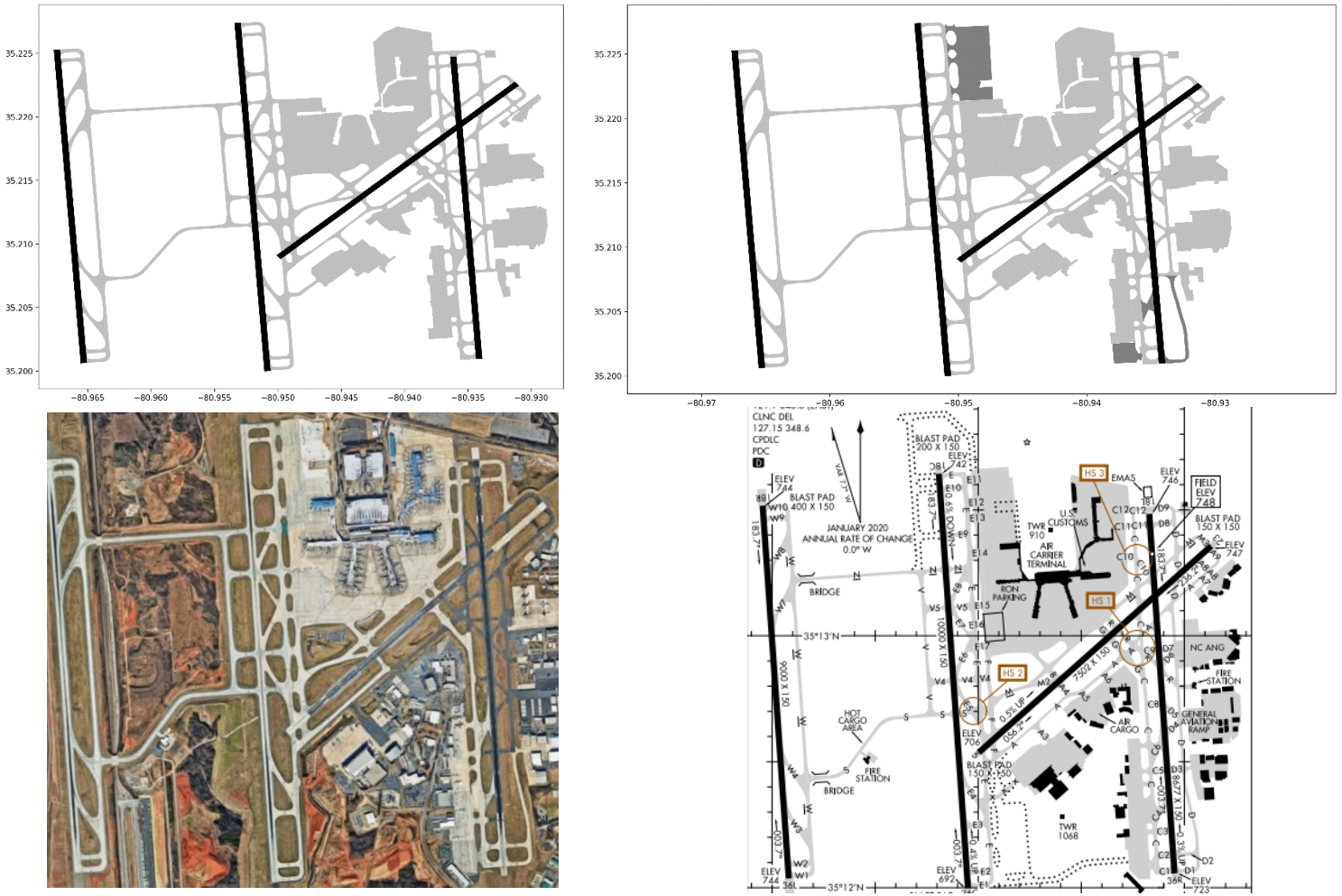

Fig. A1 From Top-Left in Clockwise Order: 2012 CLT Layout, Updated 2023 CLT Layout, 2023 CLT FAA Diagram [23], CLT Google Maps Satellite Image [22]

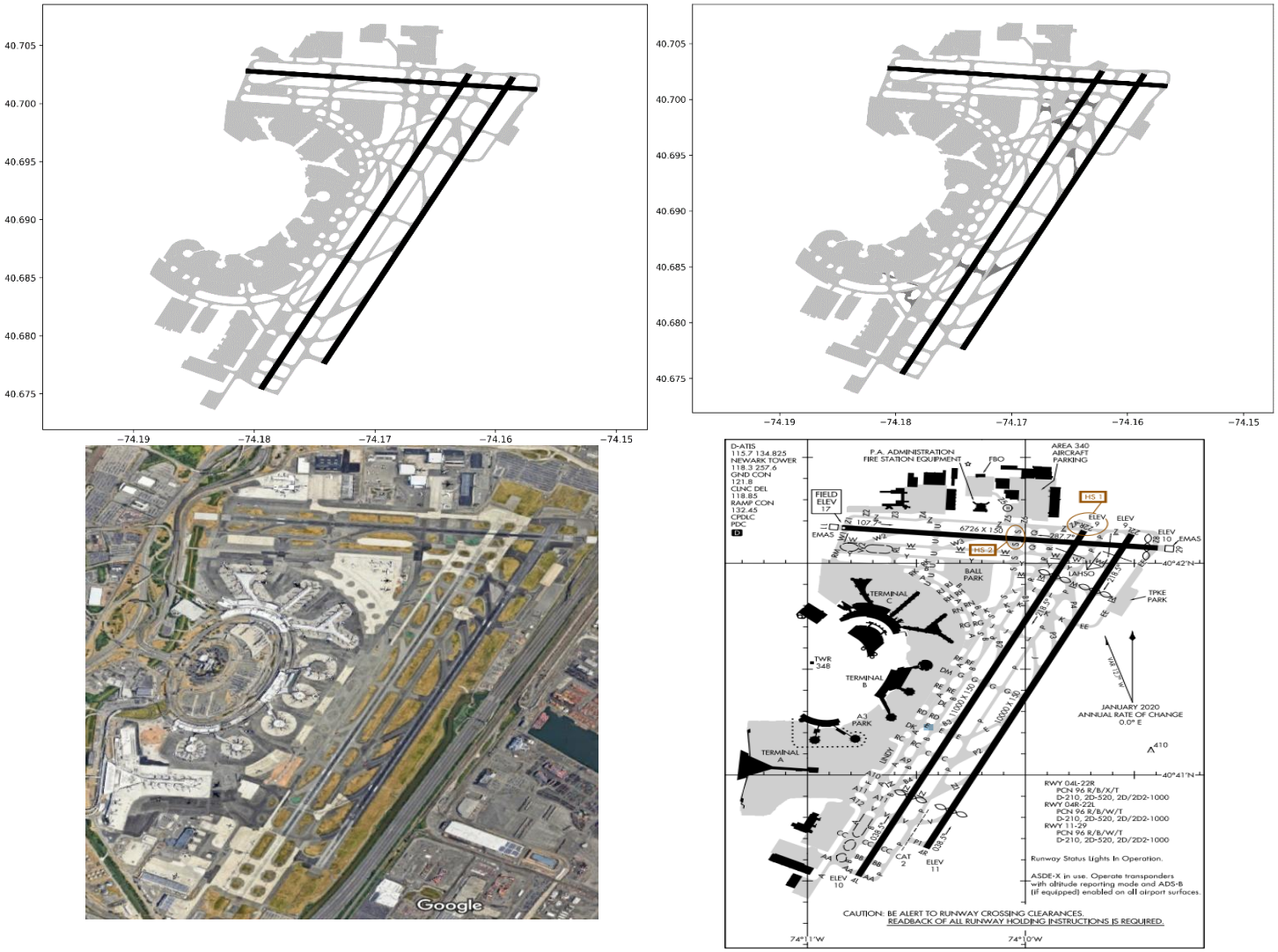

Fig. A2 From Top-Left in Clockwise Order: 2012 EWR Layout, Updated 2023 EWR Layout, 2023 EWR FAA Diagram [23], EWR Google Maps Satellite Image [22]

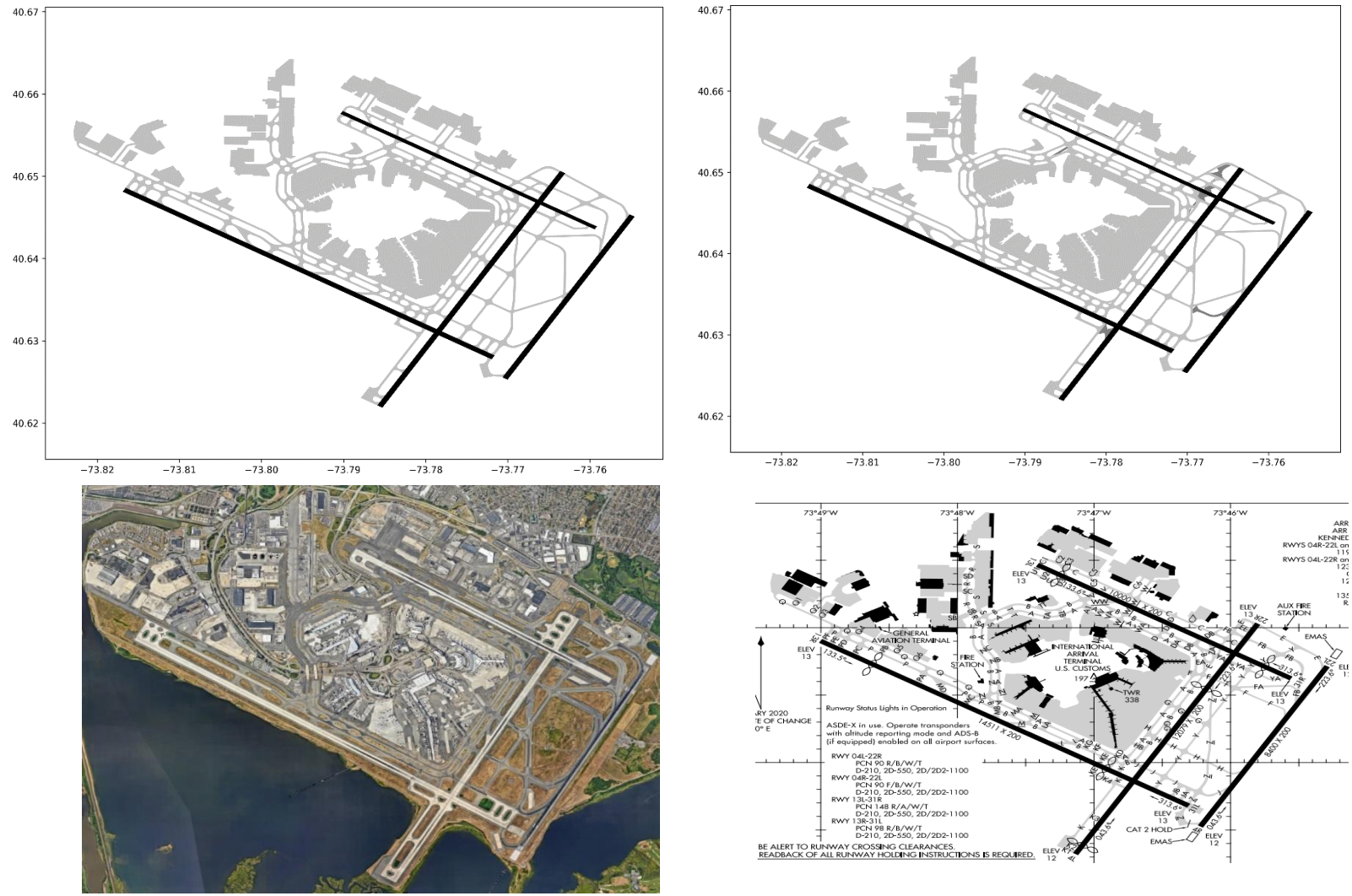

Fig. A3 From Top-Left in Clockwise Order: 2012 JFK Layout, Updated 2023 JFK Layout, 2023 JFK FAA Diagram [23], JFK Google Maps Satellite Image [22]

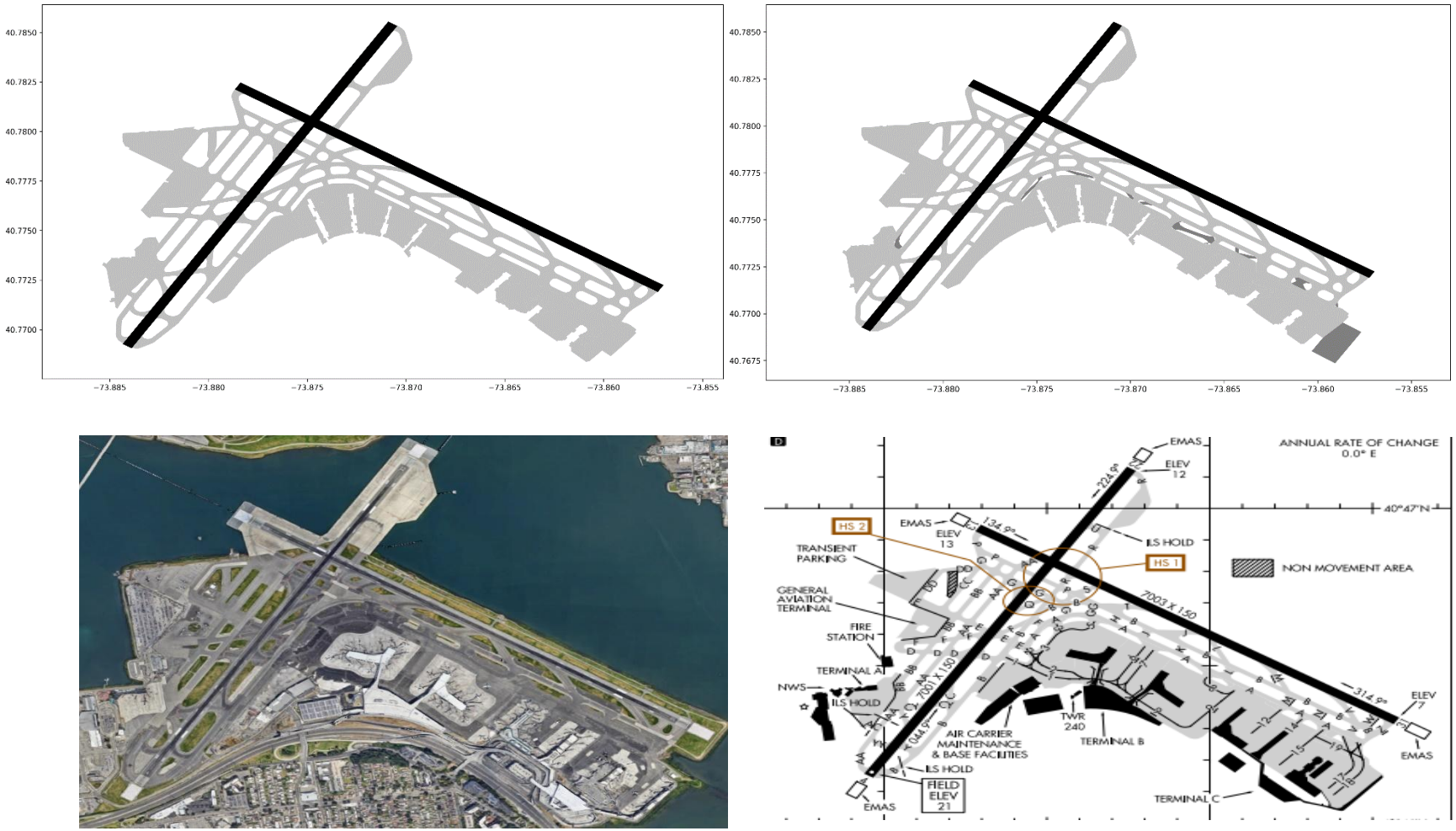

Fig. A4 From Top-Left in Clockwise Order: 2012 LGA Layout, Updated 2023 LGA Layout, 2023 LGA FAA Diagram [23], LGA Google Maps Satellite Image [22]

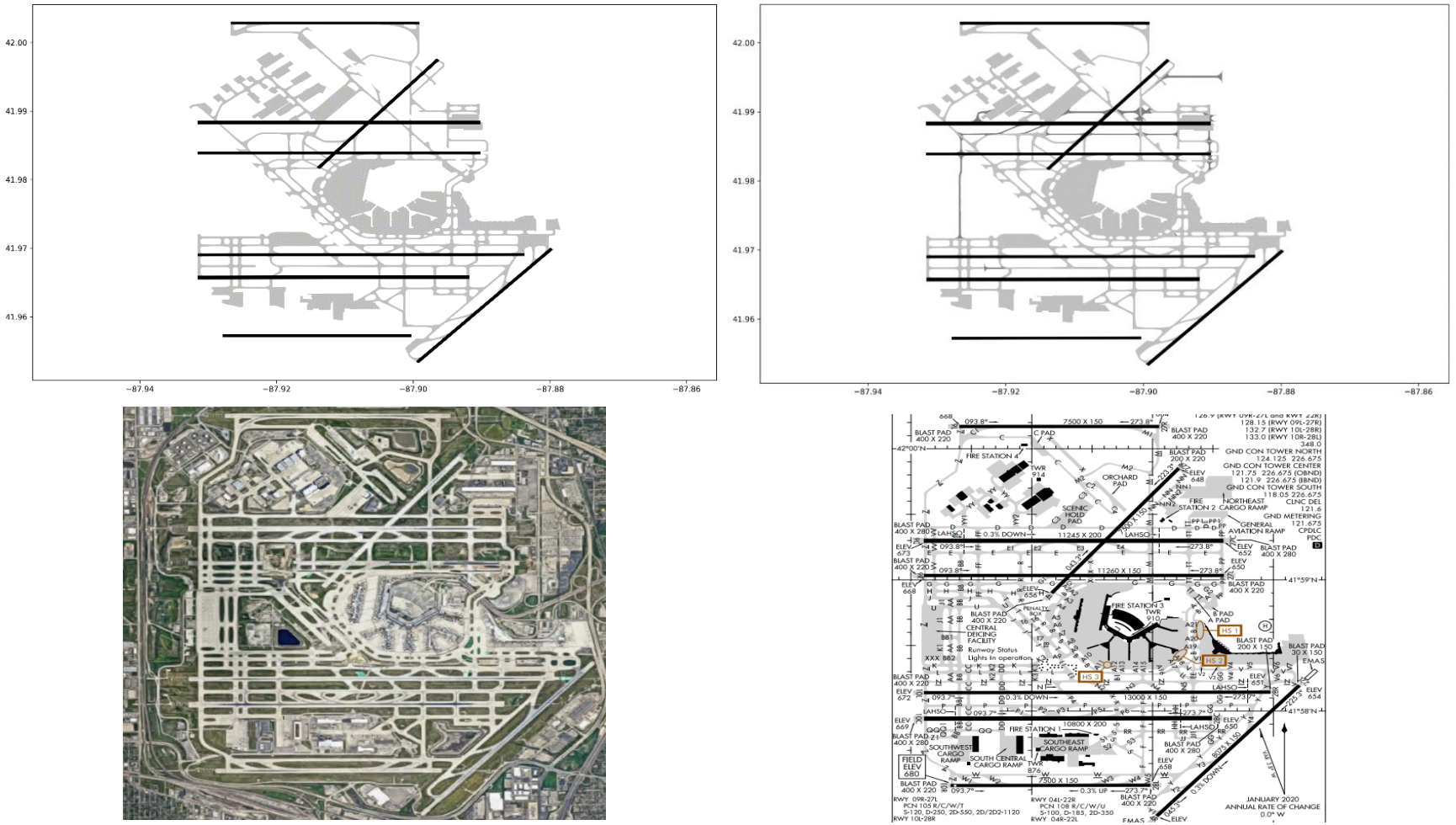

Fig. A5 From Top-Left in Clockwise Order: 2012 ORD Layout, Updated 2023 ORD Layout, 2023 ORD FAA Diagram [23], ORD Google Maps Satellite Image [22]

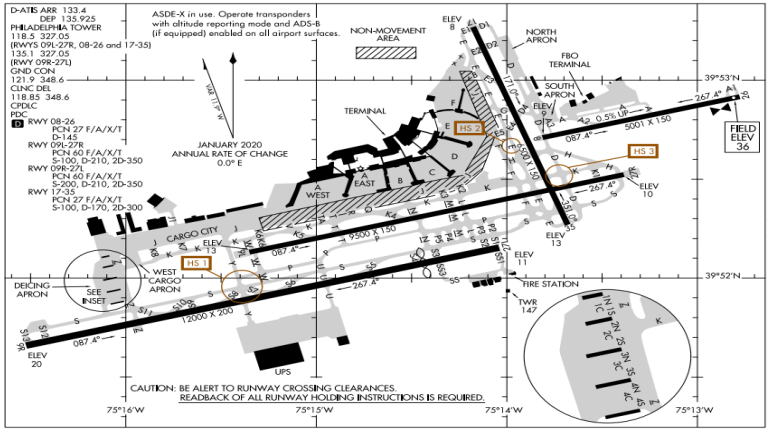

Fig. A6 From Top-Left in Clockwise Order: 2012 PHL Layout, Updated 2023 PHL Layout, 2023 PHL FAA Diagram [23], PHL Google Maps Satellite Image [22]

Funding Sources

This project was funded by the Federal Aviation Administration as part of NEXTOR III (693KA9-20-D-00004).

Acknowledgments

The authors thank Marc Meekma of the Federal Aviation Administration's Office of Performance Analysis for feedback on results of the study.

References

[1] Federal Aviation Administration (2024). Airport Surface Detection Equipment, Model X (ASDE-X). https://www.faa.gov/air_traffic/technology/asde-x.

[2] Federal Aviation Administration (2024b). Advisory Circular, AC 150/5300-13B. https://www.faa.gov/airports/resources/advisory_circulars/index.cfm/go/document.current/documentnumber/150_5300-13.

[3] Seiler, K. M. (2022). Haul road mapping from GPS traces. *arXiv preprint arXiv:2206.13936*.

[4] Badran, A., El-Geneidy, A., & Miranda-Moreno, L. (2024). A review of techniques to extract road network features from global positioning system data for transport modelling. *Transport Reviews*, *44*(1), 69-84. https://doi.org/10.1080/01441647.2023.2229521.

[5] Duran, D., Sacristán, V., & Silveira, R. I. (2016). Map construction algorithms: an evaluation through hiking data. *5th ACM SIGSPATIAL International Workshop on Mobile Geographic Information Systems* (pp. 74-83). https://doi.org/10.1145/3004725.3004734.

[6] Kasemsuppakorn, P., & Karimi, H. A. (2013). A pedestrian network construction algorithm based on multiple GPS traces. *Transportation research part C: emerging technologies*, *26*, 285-300. https://doi.org/10.1016/j.trc.2012.09.007.

[7] Ahmed, M., Karagiorgou, S., Pfoser, D., & Wenk, C. (2015b). A comparison and evaluation of map construction algorithms using vehicle tracking data. *GeoInformatica*, *19*, 601-632. DOI 10.1007/s10707-014-0222-6.

[8] Mintsis, G., Basbas, S., Papaioannou, P., Taxiltaris, C., & Tziavos, I. N. (2004). Applications of GPS technology in the land transportation system. *European journal of operational Research*, *152*(2), 399-409. https://doi.org/10.1016/S0377-2217(03)00032-8.

[9] Lyu, H., Pfoser, D., & Sheng, Y. (2021). Movement-aware map construction. *International Journal of Geographical Information Science*, *35*(6), 1065-1093. https://doi.org/10.1080/13658816.2020.1863409.

[10] Han, X., Zhao, Y., & Qing, K. (2018) Detecting Airports in TM Satellite Images Based on Edge Tracing and SURF. Proceedings of the International Workshop on Environment and Geoscience (IWEG 2018), pages 496-500. Beijing

Research Institute of Uranium Geology. ISBN: 978-989-758-342-1.

[11] Jackson, P. (2016). Automatic Update of Airport GIS by Remote Sensing Image Analysis (Master's Thesis, Durham University). https://etheses.dur.ac.uk/11651/.

[12] Kim, S., Gan, Y., & Irizarry, J. (2021). Framework for UAS-integrated airport runway design code compliance using incremental mosaic imagery. *Journal of Computing in Civil Engineering*, *35*(2), https://doi.org/10.1061/(ASCE)CP.1943-5487.0000960.

[13] Pschierer, C., Gilbert, B., DeBisschop, C., & Van der Stricht, S. (2011). Aerodrome mapping databases supporting taxi routing functions. In *2011 IEEE/AIAA 30th Digital Avionics Systems Conference* (pp. 8E4-1). IEEE. DOI: 10.1109/DASC.2011.6096157.

[14] Federal Aviation Administration (2024d). SWIM Terminal Data Distribution System (STDDS). https://www.faa.gov/air_traffic/technology/swim/stdds.

[15] Federal Aviation Administration (2024e). Welcome to the SWIM Industry-FAA Team (SWIFT) Initiative. https://www.faa.gov/air_traffic/technology/swim/swift.

[16] Karagiorgou, S., & Pfoser, D. (2012). On vehicle tracking data-based road network generation. In *Proceedings of the 20th international conference on advances in geographic information systems* (pp. 89-98). https://doi.org/10.1145/2424321.2424334.

[17] Ahmed, M., Karagiorgou, S., & Wenk, C. (2015a). Map construction algorithms. *Springer International Publishing*. ISBN 978-3319251660.

[18] Mirmohammadsadeghi, N., Hotle, S., Trani, A., & Gulding, J. (2020). Taxi event extraction from surveillance for surface performance evaluation. Journal of Air Transportation, 28(2), 28-35. https://doi.org/10.2514/1.D0160.

[19] Li, M. K., & Hotle, S. (2020). Evaluation of Taxiing Behavior by Airport and Flight Characteristics. In AIAA AVIATION 2020 FORUM. https://doi.org/10.2514/6.2020-2883.

[20] Federal Aviation Administration (2024a). 28 Day NASR Subscription. https://www.faa.gov/air_traffic/flight_info/aeronav/Aero_Data/NASR_Subscription/.

[21] SciKit-Learn (2024). KMeans. https://scikit-learn.org/1.5/modules/generated/sklearn.cluster.KMeans.html.

[22] Google (2024). Google Maps. https://www.google.com/maps/.

[23] Federal Aviation Administration (2024c). FAA Airport Diagrams. https://www.faa.gov/airports/runway_safety/diagrams/.

[24] Milne, D., & Watling, D. (2019). Big data and understanding change in the context of planning transport systems. *Journal of Transport Geography*, 76, 235-244. https://doi.org/10.1016/j.jtrangeo.2017.11.004.