Modeling and Estimating Airspace Movements Using Air Traffic Control Transcription Data
A Data-Driven Approach

Max Z. Li
Department of Electrical and Systems Engineering
University of Pennsylvania
Philadelphia, Pennsylvania
lizhaoyu@seas.upenn.edu

Megan S. Ryerson, Ph.D.
Department of City and Regional Planning
Department of Electrical and Systems Engineering
University of Pennsylvania
Philadelphia, Pennsylvania
mryerson@design.upenn.edu

Abstract—High-fidelity models of airport and airspace capacity enable researchers to study modernization strategies that optimize capacity. The design and development of airport and airspace capacity models require volumes of detailed aircraft movement data in the terminal airspace. This data, while it exists in the public realm, is highly challenging and cumbersome to collect in large quantities. In this study, we present a methodology to develop our titled Approach Airspace Characterization (AAC) database, fed by flight data scraping techniques and transcriptions of Air Traffic Control voice commands in the terminal airspace. We illustrate the mathematical mechanisms required to assign aircraft movements to specific arrival fixes to develop high-fidelity models of movements. We then present innovative new ways of measuring delay and refining arrival airspace models based on this fine-grained data.

Keywords: Air Traffic Control; Airport Capacity; Airspace Capacity; Arrival Fix Characteristics; Data Transcription

I. INTRODUCTION

Understanding airport and airspace capacity in a detailed way is a topic of great interest to those involved with modernizing and modifying the terminal airspace and airport infrastructure. The research community plays a large part in airspace modernization; researchers study strategies to optimize the airspace and airport capacity and test different scenarios. For researchers to quantify the impact of modernization initiatives such as the Federal Aviation Administration’s (FAA) Next Generation Air Transportation System (NextGen), detailed models of airport and airspace capacity are necessary. However, the data required to develop such models is highly challenging and cumbersome to collect in large quantities. The data is voluminous and collected and protected by airlines or Air Navigation Service Providers. In the following study, we present a methodology to develop an Approach Airspace Characterization (AAC) database, fed by flight data scraping techniques and transcriptions of Air Traffic Control voice commands in the terminal airspace. We conclude by describing a methodology to extract a new delay metric, just one of the many uses of the AAC database.

There is a large body of literature modeling and characterizing airport and airspace capacity. Liu, Hansen, and Mukherjee collected Airport Arrival Rates (AAR) metrics provided by publically available FAA databases, and built probabilistic arrival rate scenarios using historical arrival rates [21]. Others have built optimization models providing a thorough treatment of capacity optimization from a classical capacity-curve-based approach [10], to a minimization of cost-to-change runway configurations framed in the context of a hybrid optimal control problem [17]. Other studies attempt to characterize capacity by relating it directly to controller workload and utilizing simulation models to deduce capacity [24].

Researchers have already established that arrival capacity at an airport is highly dependent on meteorological conditions in the terminal airspace and worked to develop models of capacity and other airspace characteristics with weather information. Kamarpour, Dadok, and Tomlin zoom in to the scope of individual trajectories and innovate ways to dynamically generate and optimize flight trajectories given incoming weather forecasts [16]. Buxi and Hansen expanded on the methodology of Liu, Hansen, and Mukherjee [21] and designed capacity profiles that incorporate weather forecasts in the form of terminal aerodrome forecasts (TAF) and STRATUS [4].

A smaller yet active body of research integrates arrival fix dynamics into models of airport capacity. Gilbo modeled the role of arrival and departure fixes in the terminal airspace by integrating them with runways to form a single unit, with the objective of maximizing capacity utilization [11]. The author showed that an unevenness in fix utilization has negative consequences on airport capacity utilization. This conclusion was echoed by Kim and Clarke; choosing instead to minimize fuel consumption and terminal airspace environmental impact, they found that delay was unevenly distributed amongst the
arrival fixes [18]. Both Gilbo and Kim and Clarke did not have access to arrival fix throughput data at the hourly, individual flight level. Thus, they had to estimate a proportional allocation of arrival demand for each arrival fix. Gilbo examined Chicago O’Hare International Airport (ORD), and for model simplicity assumed that there were only four arrival and four departure fixes.

Some research efforts have collected reliable and chronologically accurate aircraft movement data in the terminal airspace by retrieving and working with ATC communications data. As the task of retrieving reliable and chronologically accurate ATC communications data must be done manually, incurring time and labor costs, automation is critical. Researchers have tried to adapt automated speech recognition to be targeted specifically towards ATC communications, yet they ran into sensitivity and comprehension problems with automation [5, 15]. Thus, gaps in our understanding of how to automate the collection of ATC communications data remain. We note that attempts to introduce ATC communications as a modeling dimension has been done, but using simulated ATC directives based off of a probabilistic analysis of ATC transcripts [25].

In this research, we present a methodology to develop the AAC database, fed by flight data scraping techniques and transcriptions of Air Traffic Control voice commands in the terminal airspace. Such data could be used to validate the models developed by researchers such as Gilbo and Kim and Clarke. The data can also provide insights into real-life scenarios where airspace and airport capacity models or assumptions behave poorly. We are motivated by our belief that the approaches to model airspace and airport capacity can be strengthened with detailed information regarding hourly – and even more finer-grained moment to moment – throughput rates at the arrival fixes within the terminal airspace.

The remainder of the manuscript is organized as follows: Section II introduces our data collection methodology and the design of the AAC database. This methodology is repeatable and flexible – other researchers can use it and customize it for their capacity studies. Section III provides an overview of our methodology of counting arrival fix utilization. Section IV presents derived metrics for delay that use the collected data. We illustrate this concretely using a proof-of-concept study with Philadelphia International Airport (PHL). Section V concludes our work with a summary and directions for future research.

II. DATA COLLECTION METHODOLOGY AND DISCUSSION

A. Air traffic control (ATC) transcription

Airspace models tend to lack the crucial dimension of pilot-controller interaction, especially if it can be merged chronologically with positional and movement data. However, transcription of ATC recordings is a time-consuming task, taking at minimum the length of the recording. Speech ambiguity and noise corruption result in unclear vocal transmissions that need to be replayed in order to accurately determine the issued command. The most critical delays in transcription occur when the transcriber cannot distinguish the flight identifier. This type of occurrence is magnified at PHL due to American Airlines’ hub presence, many aircraft carry the same “American” callsign.

Due to the inherent difficulties in obtaining ATC communications data, we focus on a few specific timeframes (days and times) over which we collect, transcribe, and ultimately determine the exact routings of flights in the PHL terminal approach airspace. We select a narrow but nontrivial timeframe for analysis: 4 pm to 5 pm EDT (2100Z to 2200Z) for three days in the Winter of 2015-2016. This time frame represents a good sampling of the afternoon and early evening increase in traffic movements (observed from the Airport Arrival Demand Chart (AADC) observations for PHL [8]) and provides a diverse mixture of aircraft types. Aircraft type diversity was noted during proof-of-concept transcriptions; we observed a number of “heavy” aircraft operating transatlantic routes as well as domestic flights arriving at PHL. We chose to analyze ATC recordings on three dates: December 29, 2015 and January 6 and 9, 2016. These three days had publically available ATC recording data and represented three different weather conditions: visual, marginal, and instrument.

ATC recordings were obtained from LiveATC.net in a MP3 file format [19]. Recordings can be freely accessed and downloaded if they are less than 30 days old (older recordings can be obtained at cost by contacting LiveATC.net) [22]. No processing was done on the audio files. oTranscribe, a free online transcription platform, was used for the transcription process [1]. oTranscribe integrates dynamic playback control of the audio file with a native word processor. The crucial component of oTranscribe worth highlighting is the ability to timestamp the transcribed text with the corresponding playtime.

We designed transcription shorthand to accommodate our transcription of the fast-paced recorded ATC-pilot verbal communications. Our shorthand covers all commands given by ATC to pilots in the terminal approach airspace. While we do not present the entire list for the sake of space, a subset of these commands includes ICAO identifier, Flight Number, Weight Class, Descend, Altitude, Intercept-Localizer, Speed, and Waypoints. For example, ATC Phraseology of “Lufthansa Four-Two-Six Heavy, descend and maintain five-thousand” is converted to “DLH426H dec 5000”.

Li and Ryerson | Modeling and Estimating Airspace Movements Using Air Traffic Control Transcription Data
The output from transcribing the timestamped ATC communications from oTranscribe is a plain text file. Lines containing transcribed ATC verbal commands are broken up into “tokens.” Every transcribed line begins with a timestamp token. Multiple commands can be linked together to reflect multiple directives in one verbal transmission. We illustrate this structure in Figure 1. Indicator variables are created to keep track of whether or not a certain type of command was issued.

![Figure 1. Methodology for parsing the transcribed ATC shorthand](image)

LiveATC.net also archives the relevant Meteorological Aerodrome Report (METAR) associated with each audio file. We retrieve the METAR in raw format, translating it using iFlightPlanner METAR and TAF translator [14]. This is our primary source of information with regards to meteorological conditions experienced during the timeframe of our ATC transcription.

### B. Merging ATC transcription data with positional data:

**“Match-validate-retrieve” methodology**

Our transcribed ATC communications data characterizes all of the chronologically-sequenced commands dictated by ATC to a target aircraft within the investigation timeframe. We next merge this with timestamped positional and movement data from FlightAware to add ten critical dimensions to our data: latitude, longitude, current heading, direction the aircraft is facing, groundspeed in knots and miles-per-hour, altitude in feet, ascend/descend rate, the day and time of day the observation was made (down to seconds), and the reporting facility [9]. FlightAware data and the ATC transcript data are matched on airline identifiers and flight numbers. This step provides two benefits: 1) we can transition into automating the data collection efforts and 2) it allows us to cross-check our transcription accuracy. We will be alerted to errors due to the occurrence of any combination of the following inconsistencies that will appear between erroneous subsets of ATC data and FlightAware data: 1) Flight destination is not PHL, 2) Actual time of arrival was not within 4 pm to 6 pm (we give a one hour “worst-case” buffer for the scenario of a flight being transcribed close to 5 pm), 3) The flight was not operated on the day of the 4 pm to 5 pm investigation timeframe, 4) The flight does not exist, indicating that the flight ID was not found. The database creation methodology is outlined in Figure 2.

![Figure 2. “Match-validate-retrieve” methodology to merge ATC communications data with positional and movement data obtained from FlightAware. This process is applicable to both manual and automated implementation.](image)

**C. Automation of “match-validate-retrieve” process using R package ‘rvest’ and custom function scrapeFW**

We automated the “match-validate-retrieve” process using a statistical software and internet data scraping techniques. We used the R package ‘rvest’ [27] and a custom function termed scrapeFW. We developed scrapeFW, a R function utilizing rvest designed to scrape positional and movement data from FlightAware for a specified flight. Data scraping techniques have been applied to aviation research before, ranging from analyzing Twitter regarding airline consumer sentiments [2] to gaining insight into airline revenue management systems and how they match lowest fare seat availability [23]. scrapeFW takes in three inputs, flightID, flightDate, and searchDate, all of which must be formatted strings. flightID is the ID of current aircraft query (“DLH426”), flightDate is the date of the investigation timeframe, and searchDate is the proxy search date used in our “proxy date matching” technique.

We developed a “proxy date matching” workaround to address an issue that occurs due to the way we handle web navigation. rvest contains three functions to handle web browsing: html_session, follow_link, and read_html. To begin, html_session takes in a Uniform Resource Locator (URL) and navigates to the website internally within R. From there, the user is able to use follow_link to abstractly “browse” the web from the starting website. Because of this primitive level of web browsing, we cannot perform complex tasks such as logging in as a registered user to FlightAware and accessing the full archive of flight history. In the internal R web session, we only have direct access to the individual flight histories going back roughly two weeks. Therefore, we implement the “proxy date matching” workaround:

1. Ensure that the flight or list of flights to be queried operated no more than 4 months prior to the current date. If the flight or list of flights of interest are outside of this range, access to premium archive data from FlightAware must be purchased.
2.) Manually determine the day-of-week that the desired flight or list of flights operated on. Suppose that the day-of-week is Thursday. From the current date, find the date of the most recent Thursday. The critical assumption of our “proxy date matching” workaround is that airports operate very similar or identical schedules on the identical day-of-week.

3.) Input the desired flight or list of flights into scrapeFW as the parameter flightID, the actual date of the flight as flightDate, and the proxy search date as searchDate.

scrapeFW will navigate to the “Flight History” page for flight flightID, exploiting the URL commonalities shared by all “Flight History” pages. If our “proxy date matching” assumptions hold, then follow_link will find the link in “Flight History” matching searchDate, and navigate to the flight information page of the proxy flight. This proxy flight, under our workaround’s assumptions, is the identical flight to the actual flight of interest in all aspects except for the date of operation.

When a valid webpage is targeted and navigated to by a web-navigation function in rvest, an R object is generated. An attribute of that object is the URL of the current webpage that the internal web session is on. Once we have navigated to the flight information page of the proxy flight, the URL attribute of the associated R object is the end goal of our “proxy date matching” workaround. We were not able to directly access the desired flightID on flightDate because the URL for this page additionally requires the flight departure time and the ICAO airport code of the origin and destination airport. All of these previously missing components are contained within the string of the end-goal URL. Parsing through this URL using various substring and string-concatenation functions, we are now able to access the desired flightID on flightDate.

Exploiting URL similarities once again, we navigate to the “tracklog” page of the desired flightID on flightDate. The “tracklog” page consists primarily of a table with columns containing the ten positional and movement data mentioned previously. On FlightAware, the table in all “tracklog” pages share a common XPath node address, which we uncover using Google Chrome’s built-in webpage element inspector [12]. In the case of FlightAware, every XPath for the table of positional and movement data is given by ‘//*[id="tracklogTable"]’. This is a valid XPath as long as the current browsing session is correctly at a valid “tracklog” page. Two rvest functions, html_nodes and html_table, properly parses the XPath and pulls the corresponding table into a R data frame. Built-in “successful query” detection will print a success prompt if the correct data is found and pulled, giving a visual cue that the search was successful. Otherwise, scrapeFW will terminate prematurely. Additionally, an error will be thrown, usually by a function within rvest. We have now detailed the automation process of obtaining FlightAware data using web-scraping techniques for flights observed in the ATC transcripts.

Our methodology to collect and merge high-fidelity and detailed data, which has the ability to capture controller-side, crew-side, and airspace operations is summarized in Table I. Note that while this report focuses on one specific time window at Philadelphia International Airport as a proof-of-concept, it is applicable to virtually any time window at any airport.

### Table I. Collection Methodology Strengths & Constraints

| ATC Communications Data Transcription | • Primary information source  
| | • High-fidelity  
| | • ATC commands to crew characterized completely  
| | • in time and content  
| | • Airport and airspace configurations known for time  
| | • of recording  
| | • Investigative timeframe must be feasible in terms  
| | • of length  
| | • Data quality subject to transcription accuracy  
| | • Relies on availability of LiveATC.net archives  
| FlightAware Positional and Movement Data | • Position, orientation, speed, and altitude known at  
| | an individual flight level  
| | • Sufficient data sampling rate allows for first-order  
| | interpolations  
| | • Cross-checks ATC communications transcription  
| | • for inaccuracies and errors  
| | • Automated collection greatly reduces process time  
| | • Relies on availability of FlightAware flight  
| | history archives  
| | • Some flights do not broadcast certain data on  
| | ADS-B (e.g. altitude)  

### III. Arrival Fix Capacity Metrics

We have now completely characterized the selected investigation window with our merged ATC communications data and flight-by-flight positional and movement data. We proceed to explore the process of extracting arrival fix utilization and throughput, and visualize the results in meaningful ways.

#### A. Methodology for obtaining arrival fix utilization metric and validation of methodology

In the following section we present a methodology to obtain an arrival fix utilization metric from the AAC database we created. This requires matching, and ultimately assigning, every flight to its most likely arrival fix. Figure 3 shows the flight tracks of observed arrivals for our three study days based on their longitude and latitude as captured by the AAC. Trajectories marked in red were on January 9, 2016, with METAR indicating marginal visual flight conditions (MVFR). Trajectories marked in blue were on December 29, 2015, with METAR indicating instrument flight conditions (IFR). Trajectories marked in green were on January 6, with METAR indicating visual flight conditions (VFR). The airport and the arrival fixes for PHL are also plotted according to their latitude and longitude coordinates given in Table II. Coordinates for the airport and arrival fixes were obtained from openNav [20], and the orthodromic distances between arrival fixes and PHL were obtained from SkyVector, an online flight planning tool used by aviators [26]. The plots were generated through GSPVisualizer [13].
calculate the orthodromic distance $d_{i,a}^t$ between flight $i$ and arrival fix $a$ at time $t$, explicitly written as:

$$d_{i,a}^t = 2r \arcsin(\sqrt{\hav(\Delta \varphi) + \cos \varphi_a \cos \varphi_i^t \hav(\Delta \lambda)})$$  \hspace{1cm} (1)$$

$\Delta \varphi$ and $\Delta \lambda$ are latitude and longitude magnitude differences, respectively. Explicitly, $\Delta \varphi = \varphi_a - \varphi_i^t$ and $\Delta \lambda = \lambda_a - \lambda_i^t$. The haversine function $\hav(\theta)$ is explicitly defined as $\hav(\theta) := \sin^2(\theta/2)$, with $\theta$ measured in radians. The constant $r$ is the radius of the Earth, which we calculate by taking the arithmetic average of the meridional radius $r_m = 3420.86$ nautical miles (nmi.) and the polar radius $r_p = 3455.50$ nmi. [28], giving us a mean radius $r = 3438.18$ nmi. Table II shows a validation of equation (1) by calculating the orthodromic distance $d_{a,PHL}$ between all arrival fixes $a$ and PHL, comparing it to given orthodromic distances $\hat{d}_{a,PHL}$ obtained from SkyVector. The haversine validation resulted in an average absolute error of 0.14 nmi. (around 260 meters).

Figure 3 confirms that it is not practical to assume that at some time $t \in T$ each flight $i$ will pass directly over a given fix $a$ such that $d_{i,a}^t = 0$ nmi. We define a utilization detection radius $r_u$ for each arrival fix $a$, and set $r_u = 3$ nmi, $\forall a \in A$. We overlay the detection boundary with radius $r_u$ onto each arrival fix in Figure 3. By visual inspection, we observe that some detection boundaries completely characterize arrival fix usage for that specific arrival fix $a$, while others will underestimate usage due to detection insensitivity.

Due to how the setting is specified, we are working in domain $\mathbb{D}(t, d, c)$ which relates spatial information $d$ and ATC communications $c$ to the current observed time $t$. A nontrivial combination $(t, d)$ that exists for all flights is the time at which the flight touches down at the threshold of a certain runway. Therefore, we define the set $\mathcal{R}$ of all runways $R$, and another utilization detection radius $r_u = 1$ nmi. for each threshold of runway $R \in \mathcal{R}$. Since this proof-of-concept focuses on PHL, let set $\mathcal{R}$ contain the elements $\{B, 9L, 9R, 17, 26, 27L, 27R, 35\}$. A simplified airport diagram of PHL is shown in Figure 4 [7]. We reason that a 1 nmi. utilization detection radius is justified for runway arrival detection, because the visualization in Figure 3 shows that positional and movement data display much more precise convergence to the runway threshold, compared to the level of dispersion at each arrival fix. The process of calculating a distance $d_{i,R}^t$ between a flight $i$ and runway threshold $R$ at time $t$ is identical to the procedure for arrival fix distance $d_{i,a}^t$.

We define an indicator variable $I_{i,a}^t = \{1, 0\}$ to be 1 if flight $i$ at time $t$ returns $d_{i,a}^t \leq 3$ nmi. per (1), and 0 otherwise. Let $t_i < t_j$, with $t_i, t_j \in T$; if $I_{i,a}^{t_i} = 1$, then $I_{j,a}^{t_j} = 0$ regardless of

Table II. Arrival Fix and Airport Geographical Descriptors with Haversine Distance Calculations

<table>
<thead>
<tr>
<th>Name</th>
<th>(Lat, Long) in radians</th>
<th>$d_{a,PHL}$ (nmi.)</th>
<th>$\hat{d}_{a,PHL}$ (nmi.)</th>
<th>Absolute Error (nmi.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PHL</td>
<td>(0.6959, -1.3132)</td>
<td>23.7</td>
<td>23.7</td>
<td>0.0</td>
</tr>
<tr>
<td>JIMS</td>
<td>(0.6901, -1.3084)</td>
<td>20.5</td>
<td>20.6</td>
<td>0.1</td>
</tr>
<tr>
<td>DQO</td>
<td>(0.6925, -1.3196)</td>
<td>26.5</td>
<td>26.6</td>
<td>0.1</td>
</tr>
<tr>
<td>BUNTS</td>
<td>(0.6996, -1.3221)</td>
<td>35.9</td>
<td>35.8</td>
<td>0.1</td>
</tr>
<tr>
<td>SPUDS</td>
<td>(0.7060, -1.3166)</td>
<td>38.9</td>
<td>38.9</td>
<td>0.0</td>
</tr>
<tr>
<td>HOGEY</td>
<td>(0.6860, -1.3203)</td>
<td>25.6</td>
<td>26.4</td>
<td>0.8</td>
</tr>
<tr>
<td>PTW</td>
<td>(0.7020, -1.3188)</td>
<td>40.1</td>
<td>40.1</td>
<td>0.0</td>
</tr>
<tr>
<td>ESSSO</td>
<td>(0.6856, -1.3204)</td>
<td>23.7</td>
<td>23.7</td>
<td>0.0</td>
</tr>
<tr>
<td>VCN</td>
<td>(0.6900, -1.3084)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The task at hand – assigning flights to the arrival fix they visited before landing and then calculating arrival fix utilization – is inherently a geometric problem. To that extent, we present some mathematical notations to simplify our description. Let the set $\mathcal{A}$ be defined as the set of all arrival fixes utilized by PHL. Let $(\varphi_a, \lambda_a)$ be the latitude-longitude coordinates for each individual arrival fix $a \in \mathcal{A}$. Let $F_d$ be the set of all unique flight IDs obtained from the ATC transcription during the investigation timeframe $T$ on some given day $d$. A flight $i \in F_d$ is indexed sequentially in set $F_d$ by the order of first appearance of flight $i$‘s flight ID in the ATC transcription. Let time index $t \in T$ indicate any particular instant in time within the valid timeframe, specified up to seconds. At some time $t \in T$, a particular flight $i \in F_d$ is located at latitude-longitude coordinate $(\varphi_i^t, \lambda_i^t)$. We define coordinates $(\varphi_a, \lambda_a)$ and $(\varphi_i^t, \lambda_i^t)$ to be converted to radians instead of degrees, $\forall a \in \mathcal{A}, \forall i \in F_d, \forall t \in T$. We use the haversine formula [3] to
Figure 4. Simplified airport diagram with runways defined in set $\mathcal{R}$ labeled what value $d_{ij}$ gives, $\forall j$. This constraint is to prevent counting a specific flight $i$ utilizing fix $a$ multiple times, as in actuality this did not occur. We also define a runway landing indicator $l_{iR} \in \{0,1\}$ which has the same characteristics as $l_{ia}$ but applies to the threshold of runway $R \in \mathcal{R}$. These specifications and constraints are diagrammed in Figure 6. This model disregards 1) multiple arrival fixes being used by one flight, as this is rare in practice, and 2) indication of a flight $i$ holding at arrival fix $a$. In Section IV we discuss possible future research motives that will investigate incorporating airborne holding.

To validate our arrival fix utilization counting methodology, we compute an analytical utilization error $e_u$ defined as the difference between the arrival fix utilization count summed over all PHL arrival fixes for the entire investigation timeframe $T$ and the number of unique observed arrivals obtained from ATC transcriptions, which we counted manually to be $\mathcal{A} = 78$. In essence, we are performing a first-order validation of our utilization detection radii $r_a$ and $r_R$, $\forall a \in A$ and $\forall R \in \mathcal{R}$.

$$e_u = \sum_{VFET} \sum_{VFDA} \sum_{VAGA} I_{iA}^t - \mathcal{A}$$

(2)

A runway utilization detection error $e_\ell$ is similarly calculated:

$$e_\ell = \sum_{VFET} \sum_{VFDA} \sum_{VRER} I_{iR}^t - \mathcal{A}$$

(3)

Our results are shown in Table III. We note that the utilization error $e_u$ was 1, indicating that we counted 79 total detected arrival fix utilizations, overcounting by 1. The runway utilization detection error was larger in magnitude, with $e_\ell = -14$, indicating 64 total detected runway utilizations, undercounting by 14. An important caveat is that all $\mathcal{A} = 78$ observed flights in the ATC transcripts may not have landed. One prominent example would be a flight who checks in with the final approach controller at the very end of the one-hour investigation timeframe. This might account for the low runway utilization detections. However, since future analysis is much more contingent on accuracy regarding arrival fix throughput, the very low $e_u$ error is an encouraging result.

B. Arrival fix usage and time-space trajectories

Important quantities in air transportation include delay metrics, interarrival times, and queue formations, each involving a component of time, distance, or both. Therefore, an intuitive
TABLE III. UTILIZATION DETECTIONS AND ERROR MEASUREMENTS

<table>
<thead>
<tr>
<th>Arrival Fix</th>
<th>Utilization Detections</th>
<th>Cumulative Arrival Fix Detection Count</th>
<th>$\alpha$</th>
<th>$e_u$</th>
<th>$e_t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPUDS</td>
<td>5</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PTW</td>
<td>1</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BUNTS</td>
<td>30</td>
<td>36</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DQO</td>
<td>6</td>
<td>42</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JIIMS/VCN$^a$</td>
<td>15</td>
<td>57</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HOGEY/ESSSO$^b$</td>
<td>22</td>
<td>79</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Runway:</td>
<td>64</td>
<td>---</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

and informative way to visualize the arrival fix data is to plot the distance between a flight $i$ and fix $a$ as function of time $t$. We call this function a trajectory $T$, $T \in \mathbb{D}(t, d, c)$ and maps $T : (i, a, t) \rightarrow d$. The time-space diagrams are shown in Figures 6, 7, and 8. Each figure shows the series of six time-space diagrams for each arrival fix or arrival fix pairs, for the three chosen days. We will highlight some key features of these time-space diagrams, as well as make some qualitative observations based on various aircraft trajectories we see plotted in time and space. These observations serve as a launch-pad for future analysis and modeling, where our collection methodology and resulting data play a critical role.

Figure 6. Arrival fix time-space diagram, December 29, 2015 (IFR)

These diagrams show the orthodromic distance $d_i^a$ between a particular flight $i$ and an arrival fix $a$ at some given time $t$, calculated through equation (1). To highlight an example, a trajectory for aircraft $k$ that noticeably decreases with some negative slope, approaching $d_{k,a}^{t_0} \approx 0$ at some time $t = t_0$, and then rapidly increases with a positive slope of roughly equal magnitude is an indication that flight $k$ was assigned and utilized arrival fix $a$, reaching that fix at time $t_0$, and continuing the final approach to PHL. Let the trajectory $T_i^{t_0-a}$ designate the set of all trajectories with this characteristic. We observe from the time-space series for all three days that the arrival fixes BUNTS and JIIMS/VCN has the most trajectories $T \in \{T_i^{t_0-BUNTS}, T_i^{t_0-JIIMS/VCN}\}$, indicating heaviest arrival fix loads. This is confirmed by our CDF column of arrival fix utilization in Table III.

Figure 7. Arrival fix time-space diagram, January 6, 2016 (VFR)

Figure 8. Arrival fix time-space diagram, January 9, 2016 (MVFR)

IV. DELAY METRICS AND FUTURE ANALYSIS DISCUSSION

A. Deriving delay from trajectory characteristics

One measure of delay is the positive differential between the nominal arrival time and actual arrival time. Our trajectories $T : (i, a, t) \rightarrow d$ allow us to generate a delay metric from this set of arrival fix throughput data, which can be related directly to airborne holding. We illustrate this with several example flights, all taken from the January 6, 2016 observation set. Let flight $i = SWA358$, observed from the perspective of $a = BUNTS$ from $t_s = 3:46:53$ pm to $t_f = 4:16:14$ pm. The space-space and time-space diagram of this flight is shown in Figure 9.
\( \mathcal{T}_{\text{SWA358-BUNTS}} \) be the trajectory for SWA358, and returns the distance between SWA358 and BUNTS at a valid time \( t \). It is apparent by inspection of Figure 9 that SWA358 experienced no in-air holding; it was vectored direct to BUNTS, and followed a smooth path to PHL. We generalize this observation to assert that any flight assigned to fix \( a \) which did not experience airborne holding should also expect to have a similar trajectory from the perspective of \( a \).

We now inspect two other flights which were observed in the same 4 pm to 5 pm window of the same day: American Airlines Flight 1714 and PSA Airlines Flight 5164. Their space-time and time-space diagrams are shown in Figure 10 and 11, respectively.

We immediately note that whatever functional form these two flights have, namely \( \mathcal{T}_{\text{AAL1714-JIIMS/VCN}} \) and \( \mathcal{T}_{\text{JIAS5164-JIIMS/VCN,HOGYE/ESSO}} \), is not the same functional form as the trajectory for a flight who did not experience airborne holding. Hence, we denote the set of trajectories for flights \( i \) who most likely experienced airborne holding observed from the perspective of fix \( a \) during time interval \( T_a \subseteq T \) as \( \mathcal{T}_{\text{H_i-a}} \). An interesting problem with flavors of functional analysis, trajectory generation, arrival fix throughput, and ATC response could be posed, attempting to characterize arrival delays and arrival queue formation by looking at the functional differences between the set of trajectories \( \mathcal{T}_{\text{H_i-a}} \) and \( \mathcal{T}_{\text{i-a}} \).

**B. Future potential topics for analysis**

We conclude this section by suggesting a medley of other ways to utilize the data that we have collected and processed in a manner that enhances our understanding of how to better model arrival fixes in the context of the entire terminal airspace. We mentioned previously that our arrival fix utilization counting methodology did not take into account airborne holding possibilities. One future topic could be to find a suitable “no-count” time difference \( \delta \) such that only if \( t_j \in (t_i, t_i + \delta) \subset T \), the no-multiple-count constraint is active. Define a hold indicator variable \( H_{i,a} = \{1, 0\} \). If \( t_f \in [t_i + \delta, t_f] \subset T \) where \( t_f \) is the last observed time in \( T \), and if \( d_{i,a} \leq 3 \text{ nmi.} \), then \( I_{i,a} = 1 \) and \( H_{i,a} = 1 \) as well. The counting methodology can be refined further by dynamically varying the detection radius for each arrival fix depending on a calculated flyover dispersion rate at that specific arrival fix.

Finally, given an investigation timeframe \( T \), suppose we examine a specific fix \( a \). We retrieve aircraft type information from FlightAware for each observed flight utilizing fix \( a \). The set of aircraft types that we retrieve is the fleet mix for \( a \). We can map this to an interarrival time between any pair of flights utilizing fix \( a \). Our time-space diagrams allow for a natural
calculation of interarrival times at arrival fix \( a \) by looking at the Euclidean distance between two trajectory curves. We have all the information we need to model an arrival queue stemming from fix \( a \), because we have the sequence of flights as well as the time separation between each flight, which is directly related to the derived interarrival times. Arrival aircraft are vectored from their designate arrival fix to the airport by ATC; we know the time at which ATC assigned a particular altitude and speed to an aircraft, as well as the time at this this altitude and speed were achieved. In essence, we know the flow of aircraft prior to arriving at fix \( a \), and the processing time and spatial information after passing fix \( a \) and beginning the final approach to PHL. We can develop a much more nuanced model of hourly arrival fix throughput, incorporating more realism by adding a dimension of ATC directives, completing \( B(t, d, c) \).

V. CONCLUSION

Motivated by the need for high-fidelity models of airport and airspace capacity, we present here a methodology to develop a database of detailed aircraft movements in the terminal area along with arrival fix assignments. We use data scraping techniques and a transcription methodology of Air Traffic Control voice commands in the terminal airspace. We illustrate the mathematical mechanisms required to assign aircraft movements to specific arrival fixes to develop high-fidelity models of movements. We then present innovative new ways of measuring delay and refining arrival queuing models based on this fine-grained data. We believe that the methodology proposed and described herein positions the field to understand and model airport and airspace capacity in an even more dynamic and fine-grained way, through publically available data.

ACKNOWLEDGMENT

The authors are grateful for the input provided by Ralph Tamburro of the Port Authority of New York and New Jersey and Andrew Churchill of Mosaic ATM.

REFERENCES


