Methods and sources for underpinning airport Ground Movement decision support systems

Alexander E.I. Brownlee *[1], Jason A.D. Atkin[2], John R. Woodward[3], and Edmund K. Burke[4]

[1]University of Stirling, UK
[2]University of Nottingham, UK
[3]Queen Mary University of London, UK
[4]University of Leicester, UK

March 19, 2020

Abstract

The airport Ground Movement problem is concerned with the allocation of routes to aircraft for their travel along taxiways between the runway and the stands. It is important to find high quality solutions to this problem because it has a strong influence on the capacity of an airport and upon the environmental impact. The problem is particularly challenging. It has multiple objectives (such as minimising taxi time and fuel consumption). It also has considerable uncertainty, which arises from the complex operations of an airport. It is an active and topical area of research.

A barrier to scientific research in this area is the lack of publicly available realistic data and benchmark problems. The reason for this is often concerned with commercial sensitivities. We have worked with airports and service providers to address this issue, by exploring several sources of freely-available data and developing algorithms for cleaning and processing the data into a more suitable form. The result is a system to generate datasets that are realistic, and that facilitate research with the potential to improve on real-world problems, without the confidentiality and commercial licensing issues usually associated with real airport data. Case studies with several international airports demonstrate the usefulness of the datasets. The algorithms have been implemented within three tools and made freely-available for researchers. A benchmark Ground Movement problem has also been made available, with results for an existing Ground Movement algorithm. It is intended that these contributions will underpin the advance of research in this difficult application area.

Keywords— a irport ground movement; taxiing; data sets; benchmarks

*sbr@cs.stir.ac.uk, `www.cs.stir.ac.uk/~sbr`

1 Introduction

The airport Ground Movement problem (Atkin, 2013; Atkin et al., 2010) connects together the problems of runway scheduling and gate allocation, often tackled separately in the literature. Ground Movement involves allocating routes for aircraft as they proceed along the taxiways between the runways and the gates (stands), and timings or orders in which to take them. The aim is to find a schedule that reduces delays, reduces the fuel burn associated with taxiing, and is resilient to last-minute changes. This is often challenging because there are typically several bottlenecks where congestion is more likely to occur, and the uncertainty inherent in aircraft landing times, push-back times and taxi speeds means that routes need to be constantly updated to reflect the current situation. This means that any solution methodology must be efficient enough that it can be executed within a couple of minutes, at most, to accommodate incoming data. It is also undesirable (and is avoided at most airports) for a new route to be assigned to an aircraft that has started its allocated movement. This means that any route allocation should be robust enough to cope with the uncertainties that arise during ground movements, such as variations in taxi speed.

Innovation in the Ground Movement problem is potentially limited by the difficulty in accessing real-world datasets. While some freely-available problems exist in the literature (e.g. the "Generic Ground Movement Problems" at ASAP, University of Nottingham: `http://www.asap.cs.nott.ac.uk/external/atr/benchmarks/index.shtml`), none truly reflect the inherent complexity of operations at a real airport. The existing literature has made use of data which had been provided through partnerships with airports. However, there are several problems with this:

- such data is typically the subject of a non-disclosure agreement making reproducibility and comparisons impossible
- the data is focused on one airport, so not general
- this prevents blue sky research and new entrants to the field.

It requires the development of working relationships with airport staff to obtain relevant data. We intend to resolve this problem in this paper via three contributions: tools top generate taxiway layouts, a concrete benchmark problem, and tools to clean movement data.

This paper explores several freely-available sources of data related to airport Ground Movement and considers the ways in which these can be combined to confront this challenge. The sources include layout information derived from open street map (OSM) (`http://www.openstreetmap.org`), which is free to use and redistribute under an Open Database Licence. OSM data can be supplemented by data from other sites, although care must be taken to ensure compliance with licensing depending on the source. This paper describes the relevant parts of the OSM data format for Ground Movement research, and an algorithm to generate layouts from raw data obtained from OSM.

Furthermore, this work has explored live flight track information originating from ADS-B transmissions. These tracks cover the majority of flights at most European and US airports and are detailed enough to allow analysis of the movements of aircraft at an airport. However, due to measurement errors, although the recorded data appears to be *precise*, it is not *accurate*. For example, on occasions an aircraft is recorded as travelling along a track parallel to the actual runway rather than along it. In order to retain as much valuable data as possible, these tracks can often be "repaired" by applying a linear transformation to the recorded trajectory and forcing the recorded track to coincide with the closest runway, correcting any systematic errors due to calibration. To recover the most likely path, the raw track data is first cleaned to remove outlying coordinates and noise, then snapped to the taxiways. From the snapped path, taxi speeds and routes taken can be analysed. The repair approach could easily be applied to snapping raw aircraft movement data coming from any sources that include latitude, longitude and time intervals. The procedure is described in detail to allow it to be reproduced by other researchers. We have demonstrated the repair method using data taken from the Flight Radar 24 (FR24) site (`http://www.flightradar24.com`).

In order to maximise the benefit to the research community of our work, we have implemented the algorithms described in two tools: *TaxiGen* and *SnapTracks*. These are supplemented by the *GM2KML* tool which is designed to enable the analysis of the data which is output by the other tools. We have also made available some example Ground Movement layouts for several international airports, including a full benchmark Ground Movement problem for Manchester Airport, the third busiest airport in the UK. Taxi time results for an existing ground movement algorithm are provided to be matched or beaten. These benchmark problems will facilitate research with the potential to improve on real-world problems, without the confidentiality and commercial licensing issues usually associated with real airport data. We conclude the paper by presenting a case study of FR24 flight tracks for several airports, discussing a number of potential research uses for the data.

2 Related work

Airport Ground Movement is a difficult problem which has been the focus of limited research over the past couple of decades. Comprehensive reviews of research in this area can be found in (Atkin et al., 2010) and (Atkin, 2013).

Approaches such as those of Gotteland and Durand (2003); Gotteland et al. (2001); Pesic et al. (2001) use predefined routes, with a heuristic search algorithm choosing route and wait points for each aircraft. More recently, Jiang et al. (2013) use a genetic algorithm to evolve the routes rather than choosing predefined ones. Alternative efforts including Clare and Richards (2011) and Yin et al. (2012) have formulated Ground Movement as a mixed-integer linear programming problem. Ravizza et al. (2013a) describe the Quickest Path Problem with Time Windows (QPPTW) algorithm, an adaptation of Dijkstra's short-

est path algorithm that takes account of the movements of previously-allocated aircraft.

A key issue is that each approach cited above has been developed for and demonstrated on only one or two airports. This makes comparison of methods difficult, and it is anticipated that our provision of benchmark problems will address this. It also hinders the development of generic methods suitable for application to many airport configurations. This will be addressed by our tools to recreate taxiway layouts for a wide range of airports.

Methods to tackle Ground Movement often assume fixed start or end times and taxi speeds. Some research (e.g. Gotteland et al., 2001; Clare and Richards, 2011; Lesire, 2010; Koeners and Rademaker, 2011; Pfeil and Balakrishnan, 2012; Khadilkar and Balakrishnan, 2014) has also attempted to account for the uncertainty that is inherent in this problem. This includes variations in push-back times, landing times and taxi speeds. In order for these methods to be successful, it is essential that real-world data is available from which accurate models of uncertainty can be generated. There have been some attempts to quantify and model uncertainty using data provided by airports or airlines. Simić and Babić (2015) and Rappaport et al. (2009) have also presented some insights from analysis of real-world aircraft movements. Modelling approaches have included reinforcement learning (Balakrishna et al., 2010); regression (Ravizza et al., 2013b; Simaiakis and Balakrishnan, 2016); fuzzy rule-based systems (Chen et al., 2011) and distribution functions (Truong, 2012). Research has also compared the different modelling approaches (Ravizza et al., 2014) and attempted to measure the impact of uncertainty using simulation (Lee and Balakrishnan, 2012; Koeners and Rademaker, 2011). Overall, there is a similar issue: all of the above methods have only been demonstrated for a small number of airports, with no common point of reference. The tools we have developed to clean and process freely available aircraft movement data will allow much wider-ranging and general research in this area.

As far as we are aware, the only other researchers to have published results using data from FR24 are Petersen et al. (2013), Turner et al. (2013), Ptak et al. (2014), and Eele and Maciejowski (2015), all of which focused only on airborne aircraft movements.

3 Taxiways and stand coordinates

A specific Ground Movement problem instance consists of a static element (the graph of taxiways) and a transient element (the required aircraft movements). The airport layout can be modelled as a graph, with edges representing taxiways and runways, and nodes representing the end coordinates of edges including stands. Aircraft movements comprise start and end nodes, a target start or end time, and expected taxi speeds. In this section, we focus on determining the taxiway graph from one of the freely available sources.

OpenStreetMap (OSM) (`http://www.openstreetmap.org`) provides freely-available mapping data, accessible to and editable by anyone with internet ac-

Figure 1: OpenStreetMap's data for taxiways and stands at London Gatwick Airport

cess. Its licence makes OSM map data "free to copy, distribute, transmit and adapt our data, as long as you credit OpenStreetMap and its contributors" (`http://www.openstreetmap.org/copyright`). OSM has seen use in other areas of transport research; recent examples including Gupta et al. (2012); Mayr and Navratil (2014) and Zilske et al. (2011). Two alternative sources that have layout information for a large number of airports are `www.flightsim.com` and `library.avsim.net`. These sites host libraries of airport layouts intended for use in flight simulator games: again they have a high level of detail, with full coordinates for taxiways and stands, the latter having associated label, size, push-back direction and airlines using the stand. Many are available in the BGL file format, which can be read by BG2XML (`www.scruffyduck.org/bgl2xml`) and exported to a simple XML format. As the process for these is more convoluted, and the licensing varies depending on the individual files (though typically free for non-profit use), we focus on OSM here.

The OSM data has some imperfections but it is surprisingly accurate. Moreover, it can easily be edited to fix any issues that are found. Several of the busiest UK airports (such as Gatwick - Figure 1, Stansted and Manchester) have both taxiways and stand locations included, whilst most of the rest have taxiways without the stands (see below).

Raw data can be exported from OSM in XML format. For Ground Movement, the two key elements are `<node>` and `<way>`. Nodes represent unique latitude / longitude (lat/lon) coordinates, and a way connects a series of nodes to form a taxiway, runway or some other path. An example segment of OSM XML is given in Listing 1, and an explanation of the other important details of

the syntax follows.

3.1 OSM Syntax

Key items of syntax in the OSM XML are:

- `<node>`: a single coordinate, with attributes for a unique ID `id`, latitude `lat` and longitude `lon`.
- `<way>`: a path connecting a sequence of nodes. An attribute specifies a unique ID `id`. Child elements are `<nd>` and `<tag>`.
- `<nd>`: identifies a node forming part of a way. The `ref` attribute is the unique ID of the node.
- `<tag>`: specifies metadata for the way as `k` and `v` attributes (key/value pairs). For Ground Movement, the important tags are `aeroway` (values runway/taxiway/parking_position), `ref` (string giving the name of the runway/taxiway/stand) and `disused` (yes/no).

Unfortunately there is some inconsistency within OSM regarding the types of `<way>` elements representing stands. Stands are typically `<way>` elements with an `aeroway` tag `taxiway` or `parking_position`. The former appears to apply to most UK airports, although the OSM documentation states a preference for the latter. The `way` for a stand starts on the taxiway and terminates at the point where the aircraft's nose wheel comes to rest. Furthermore, at some airports, `<node>` elements with an `aeroway` tag of `gate` indicate stands.

A key issue is that individual stands are not present in the mapping for many airports. Of the busiest UK airports, only Stansted, Gatwick and Manchester currently have stand data, although it is open for anyone to add this to OSM. The missing information can, however, be sourced from elsewhere. NATS AIS (Aeronautical Information Service) (`http://www.nats-uk.ead-it.com`) and the Eurocontrol European AIS Database (`https://www.ead.eurocontrol.int`) hold charts and data for UK airports that are publicly available via the web for private use. These include Gate Parking/Docking charts with lists of stand coordinates. Licensing prevents us re-sharing this data, but researchers can easily obtain this for themselves.

Once the coordinates of stands have been obtained, together with the taxiway that they connect to, edges can be added to the taxiway graph by assuming that the stands are (as is often the case in reality) perpendicular to the taxiways. Alternatively, an OSM node ID can be specified to attach the stand to. Whichever approach is taken, the process of completing a graph of taxiways can follow the procedure in sub-section 3.2.

Note that latitude and longitude cannot be used as simple x/y coordinates in a Ground Movement implementation. This is because the distance on the ground represented by a degree of longitude varies depending on the latitude. A simple solution to this problem is to convert all lat/lon coordinates for a specific airport to UTM coordinates, which represent an approximation to a square grid

```
<node id="25256057" ... lat="53.3479585" lon="-2.2776300"/>
<node id="25255969" ... lat="53.3493709" lon="-2.2793476"/>

<way id="4232478" visible="true" ... >
  <nd ref="25256057"/>
  <nd ref="25255969"/>
  <tag k="aeroway" v="taxiway"/>
  <tag k="ref" v="U"/>
  <tag k="source" v="Yahoo"/>
  <tag k="width" v="23"/>
</way>

<node id="379746103" ... lat="53.3319280" lon="-2.3107605"/>
<node id="20012053" ... lat="53.3320505" lon="-2.3105072"/>
<node id="25256064" ... lat="53.3334113" lon="-2.3076954">
  <tag k="aeroway" v="turning_circle"/>
</node>
...

<way id="3895636" visible="true" ... >
  <nd ref="379746103"/>
  <nd ref="20012053"/>
  <nd ref="25256064"/>
  <nd ref="379771699"/>
  <nd ref="379746054"/>
  <nd ref="25256076"/>
  <nd ref="25256069"/>
  <nd ref="25256056"/>
  <nd ref="25256057"/>
  <tag k="aeroway" v="runway"/>
  <tag k="length" v="3200"/>
  <tag k="loc_name" v="Runway Two"/>
  <tag k="ref" v="05R / 23L"/>
  <tag k="source" v="Yahoo"/>
  <tag k="surface" v="grooved_asphalt"/>
  <tag k="width" v="45"/>
 </way>
```

Listing 1: OSM XML for a taxiway and a runway, each defined by a `way` element. Only a subset of the `nd` elements are included: '...' marks where superfluous data has been omitted to save space.

within a local area. Libraries such as JCoord (`http://www.jstott.me.uk/jcoord`) can easily facilitate this process.

3.2 Generating a taxiway graph

Algorithm 1 gives our proposed procedure to take the taxiway, runway and stand information from OSM, combine it with additional stand location information from another source if required, and generate a graph forming the basis of a Ground Movement problem.

Starting with an empty graph G, the OSM data is parsed in steps 2 and 3, the relevant nodes and edges being added to G. Nodes for separately specified stands (e.g. from the NATS data) are added to G in step 4. Edges are then added to G connecting stand nodes to nodes on the taxiways. As noted in Section 3.1, it can be assumed that the stand will be perpendicular to the taxiway. To achieve this, the lat/lon coordinates of the stand and the nearest edge to it on the taxiway are converted to UTM references (these being having equal dimensions in x and y directions). A perpendicular line is projected from the edge to the stand coordinate, and a node is inserted into the taxiway at the point of intersection (steps 5 to 16). The stand may not always be perpendicular to the taxiway. Where this applies, a node ID on the taxiway can be specified for the connection (step 17). Finally, long edges are divided up by inserting intermediate nodes at predefined intervals (step 18). We used a value of 50m since it is large enough to accommodate one aircraft, allowing us to simulate a queue of aircraft along the edge. The *TaxiGen* tool (Section 6.1) implements this procedure.

4 Manchester Airport Benchmark Problem

We have made available a benchmark problem of the real-world requirements at Manchester Airport. Manchester is the third busiest airport in the UK according to CAA statistics (Civil Aviation Authority (2013b) and Civil Aviation Authority (2014b)), in both annual passengers (20.7M) and aircraft movements (159 000). The airport has several features which make it interesting from a Ground Movement perspective. It has three terminals, two runways (05L/23R and 05R/23L) and 148 aircraft stands. 54 stands are *shadowed* so that they cannot be used when larger aircraft are on adjacent stands. 57 stands are served by terminal piers, and 91 are remote (accessed by bus transfer from the terminal). Access to runway 05R/23L is achieved by crossing 05L/23R. Access to stands on the apron serving terminal 2 and part of terminal 1 are often via only a single taxiway, since the use of several stands may block the alternative taxiway. Owing to the limited number of terminal stands, aircraft on longer stopovers are often towed to remote stands, placing further demand on the taxiways. A stylised diagram of the airport is presented in Figure 2.

Through a combination of the freely-available data from OSM, and aircraft movement data provided by the Airport itself, we can now provide a set of

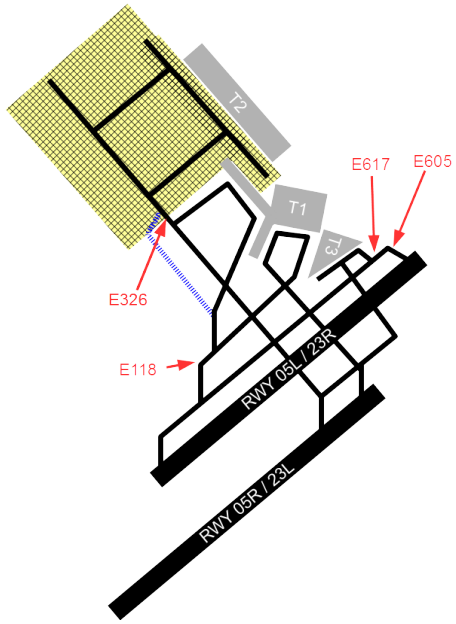


Figure 2: Stylised diagram of Manchester Airport showing the terminals (grey), runways (black) and major taxiways (solid black and dashed blue). During busy periods, the dashed blue taxiway could be blocked by aircraft on stands, so that the northern apron (yellow with black hatching) can only be reached by a single remaining taxiway. This apron has around half of the airport's stands, including those for T2, part of T1, and remote stands. The numbers in red identify edges referred to by the case study in Section 7.1.

Algorithm 1 Generating the taxiway graph

```
Initialise graph G
Extract taxiways and runways from OSM data, adding nodes and edges to G
Identify ways tagged as stands, adding a stand node and connecting edge to G
Load and create nodes s for stands from NATS data; insert s into G
for all s without a specific node to attach to do
    Find all edges e related to taxiway specified for s
    Compute the nearest coordinate i_e to s on each e
    for the single e nearest to s do
        if I is within 1m of the end of e then
            Set i_e to the end of e
        else
            Split e into two edges at i_e
        end if
        Add an edge connecting s to i_e
    end for
end for
Attach all nodes with a specified attachment point
Add intermediate nodes to divide long edges
Return G
```

example Ground Movement problems which are intended to act as a benchmark for comparisons. The problems are available at: `http://www.asap.cs.nott.ac.uk/external/atr/benchmarks/index.shtml`. Additional layouts without movement data for several international airports derived from OSM are also available at the same URL.

The datasets include a set of real-world movement requirements provided by Manchester Airport for the period 00:00 on 29 August 2011 to 23:59 on 3 September 2011. These comprise 1596 arrivals and 1615 departures, as well as 284 stand-to-stand towing manoeuvres that took place during the period. Specified for each aircraft are the runway time, the runway entry/exit used, and the stand.

Also included are alternative datasets with varying-difficulty versions of the problem, having decreased and increased traffic levels, in two groups. Group one covers the same period of 6 days, with varying traffic levels up to 2078 arrivals, 2073 departures, and 525 towing manoeuvres (approximately 1.34x standard traffic levels). Group two covers a single day (Thursday 1 September 2011), with varying traffic levels up to 1.64x standard levels. The sets were generated by selecting flights at random and either deleting them, or copying them by duplicating the runways used and flight times (with a 2 minute offset to recreate typical runway separations). Where flights were duplicated, stands must be reassigned to aircraft, because two aircraft cannot be assigned to the same stand. Stands have been assigned according to a set of rules typical of those

Table 1: Summary of varying density datasets covering six days of operations for Manchester Airport

Set name	Mult	Arr	Dep	Tow	Total	Mult orig	t_{min}	t_{QPPTW}
MANC_6days_0.8_GM.txt	0.8	1380	1374	227	2980	0.85	650 572	677 273
MANC_6days_0.9_GM.txt	0.9	1551	1549	319	3418	0.98	741 596	776 881
MANC_6days_1.0_GM.txt	1.0	1733	1725	382	3839	1.10	832 598	875 702
MANC_6days_1.1_GM.txt	1.1	1902	1901	432	4234	1.21	921 300	979 610
MANC_6days_1.2_GM.txt	1.2	2078	2073	526	4676	1.34	1 015 651	1 093 475

Table 2: Summary of varying density datasets covering a single day of operations for Manchester Airport

Set name	Mult	Arr	Dep	Tow	Total	Mult orig	t_{min}	t_{QPPTW}
MANC_1day_0.8_GM.txt	0.8	230	230	43	503	0.84	111 787	112 753
MANC_1day_0.9_GM.txt	0.9	259	259	57	575	0.96	124 654	125 835
MANC_1day_1.0_GM.txt	1.0	289	289	62	640	1.07	138 518	140 114
MANC_1day_1.1_GM.txt	1.1	318	318	65	701	1.17	154 298	156 494
MANC_1day_1.2_GM.txt	1.2	347	347	96	790	1.32	170 980	173 682
MANC_1day_1.3_GM.txt	1.3	376	376	99	851	1.43	184 290	187 335
MANC_1day_1.4_GM.txt	1.4	405	405	114	924	1.55	199 861	203 612
MANC_1day_1.5_GM.txt	1.5	443	438	139	1020	1.64	221 557	226 800

used at Manchester, using a Breakout Local Search algorithm, detailed by Benlic et al. (2014). The sets are summarised in Tables 1 and 2. For each data set, the multiplier (Mult) for the initial deletion or duplication of flights is given. This is followed by the number of arriving, departing, towing and total movements in the set, with this latter total expressed as a multiple of the movements in the original data provided by Manchester Airport (Mult orig). "Mult" and "Mult Orig" usually differ because, depending upon the gate assignments used, the need for towing manoeuvres varies as more use is made of stands. The "t" values are times, and are explained below.

Finally, it is also important to have realistic speeds for taxiing aircraft. Research in this area is ongoing (e.g. Section 5); however, a basic set of rules which can be adopted for the purposes of benchmarking are: 10.3 m/s for straights, 9.2 m/s for turns. These speeds were determined by measuring the real-world movements of aircraft at Manchester Airport over a week, using the data described in Section 5.1.1. The benchmark definition files identify whether any given edge is a straight or a turn, dependent on the previous edge (any movement tighter than 30 degrees with 100m being classified as a turn). For benchmarking, two times are also given in the tables, both being total taxi times for all aircraft in the data set. t_{min} is the theoretical minimum taxi time for all aircraft, assuming that all were able to take the shortest path, unimpeded by other aircraft, in terms of distance. t_{QPPTW} is the taxi time taken when the aircraft were routed using the Quickest Path Problem with Time Windows (QPPTW) algorithm of Ravizza et al. (2013a), assuming a minimum separation between aircraft of 60m.

5 Real-world Ground Movement

The second part of this paper concerns transient data, that is, timings and movements of real aircraft. For commercial and security reasons this is particularly hard to obtain, although most airports and sites such as FlightStats (`http://www.flightstats.com`) offer live updates of aircraft departure and arrival times via their websites. The increasing take-up of Automatic Dependent Surveillance – Broadcast (ADS-B) means that more detailed information such as taxiways taken and taxi speeds can be obtained by either collecting the data directly using a receiver or via websites such as Flight Radar 24 (FR24) (`http://www.flightradar24.com`), PlaneFinder (`http://planefinder.net`), Radar Box 24 (`http://www.radarbox24.com`) or several others. The usage policy for data from each site varies and should be carefully examined before commencing research. For example, at the point where the data sets were collected (12/3/2015), FR24 gave permission "to temporarily download one copy of the materials (information or software) on FR24's website for personal, non-commercial transitory viewing only" (FR24, 2015), so data cannot be stored long-term or redistributed, nor can it be used for commercial activity. More recently the data has been made available via an API on a commercial basis.

In order to develop a realistic Ground Movement problem, it is important to include the full traffic level at an airport. This cannot be achieved by an analysis of the ADS-B data alone, which (as noted later) currently only includes 50-60% of all traffic. Often, airports or civil aviation authorities will publish total aircraft movements for airports, although this is usually aggregated into monthly or annual statistics (e.g. Deutsche Flugsicherung, 2014; Sheremetyevo Airport, 2014; Zurich Airport, 2014; Civil Aviation Authority, 2014a). Whilst not all flights will appear on public channels, at commercial airports, the number of private flights are limited (for example, discussions with our partner airports revealed this to be around 1% of all flights at their specific locations). The FlightStats.com website offers an API which can be easily parsed automatically and reveals detailed information regarding flights including runway and on/off-block times, gate and terminal used, aircraft used and passenger numbers. However, investigations with several UK airports found that often the data available is more limited. Times and terminals will be present, but usually not gates (particularly for arrivals). We suggest that if combined with a gate allocation algorithm using realistic rules to fill in the missing information, the data obtained from such a source could form the basis of a real-world Ground Movement problem for research.

Movement data originating from ADS-B can be noisy, with positional or timing errors requiring cleaning before the data is usable. In the remainder of this section, we propose a cleaning process that transforms a raw flight track into a path across the graph of taxiways. There are three broad stages: cleaning noise from the raw data (Section 5.2), snapping the raw coordinates to the graph of known taxiways and runways (5.3), and determining the times that aircraft reached each node in the graph (Section 5.4). We have developed and made available the *SnapTracks* tool to implement this process, detailed in Section

6.2. The process is generic and could be used to snap any track of coordinates with timings to a taxiway graph, regardless of origin. We will now demonstrate the process with data gathered from FR24.

5.1 Flight Radar 24

Data representing real aircraft movements is available via FR24, and has also been used to gather airborne flight tracks by Petersen et al. (2013), Turner et al. (2013), Ptak et al. (2014), and Eele and Maciejowski (2015). FR24 gathers ADS-B messages that are transmitted by many aircraft and contain the latitude, longitude and altitude of the aircraft at typical intervals of 5s to 10s. The website offers a playback facility to provide an archive of flight movements for the past month. Clicking on a single aircraft in the FR24 web interface displays a trail showing the aircraft's precise track. The majority of aircraft continue broadcasting their location during taxiing, and from this it is possible to determine the path taken and speed profile for that aircraft's ground movement. The coordinates have a resolution of 10^{-4} degrees, approximately 10 metres at the latitude of the UK. Not all aircraft broadcast ADS-B data (the equipment has either not yet been installed or has been switched off), and of those that do, there are sometimes calibration errors or other causes of corruption that need cleaning. However, enough flight movements are available that insights can be made into taxi times and other areas of interest.

The usage policy of FR24 does not permit the sharing of derivative works, so our public datasets do not include data derived from this source. Collecting the data from the website is relatively simple (it is sent to the browser in JSON format).

5.1.1 Validation and demonstration

In order to determine the accuracy and value of this work, an analysis was performed using data which was collected from FR24 for several airports covering a variety of sizes and aviation authorities. The airports studied are listed below. Annual passenger numbers and aircraft movements for each are given in Table 3.

Cologne-Bonn (CGN), the sixth largest airport in Germany by aircraft movements. It has three runways and 110 stands excluding shadows. Annual statistics were taken from Köln-Bonn Airport (2013), and monthly flight figures from Deutsche Flugsicherung (2014).

Edinburgh (EDI), the sixth largest airport in the UK by aircraft movements. It has two runways and 36 stands excluding shadows. Annual statistics were taken from Civil Aviation Authority (2013a, 2014b), and monthly flight figures from Civil Aviation Authority (2014a).

Manchester (MAN), the third largest airport in the UK by aircraft movement. It has two runways and 94 stands excluding shadows. Annual

statistics were taken from Civil Aviation Authority (2013a, 2014b), and monthly flight figures from Civil Aviation Authority (2014a).

Melbourne International (MEL), the sixth largest airport in Australia by aircraft movements (although the second largest in terms of passenger numbers, several regional airports with high numbers of small aircraft have more movements). It serves as a major hub for Qantas and Virgin Australia. It has two runways and 50 stands excluding shadows. Annual statistics were taken from Airservices Australia (2013) and Melbourne Airport (2014) and monthly statistics from Airservices Australia (2014).

Stuttgart (STR), the seventh largest airport in Germany by aircraft movements. It has one runway and 58 stands excluding shadows. Annual statistics were taken from Stuttgart Airport (2013), and monthly flight figures from Deutsche Flugsicherung (2014).

Sheremetyevo (SVO), the second largest airport in Russia by aircraft movements, one of three major airports serving Moscow, and the hub airport for Aeroflot Russian Airlines. It has two runways and 181 stands excluding shadows. Annual statistics were taken from Sheremetyevo Airport (2015), and monthly flight figures from Sheremetyevo Airport (2014).

Zurich (ZRH), the largest airport in Switzerland by aircraft movements, and the hub for Swiss International Airlines. It has three runways and 60 stands excluding shadows. Annual statistics were taken from Zurich Airport (2013), and daily flight figures from Zurich Airport (2014).

Data was collected for all aircraft with an altitude of zero (i.e. on the ground) that were within 5km of the centre of each airport. Information is provided in each of the following sections to illustrate the ways in which the data was corrected during each stage of the analysis.

For each airport studied, Table 3 gives the dates covered by the data collected (from 00:00 on the start date to 23:59 on the end date). The number of flights captured are given, with the percentage of actual flights in this period that this represents, and the number of actual flights for comparison. For Zurich Airport, this is the exact figure, because the airport routinely publishes daily statistics for traffic. For the other airports, the value is approximated by taking a fraction of the relevant monthly figures. It can be seen that at least in terms of raw data, a large proportion of actual flights is represented (around what would be expected given a take-up for ADS-B of 50-60% and increasing). For Melbourne, the figure is particularly high (over 90%): this is likely to be because Australia has already mandated the fitting of ADS-B equipment to most aircraft.

The initial tracks for the aircraft, as given by the raw data, are illustrated for Manchester Airport in Figure 3, and superimposed on top of an aerial photograph of the airport using Google Earth™.

Table 3: Summary of airports studied and the data collected from FR24.

Airport	Pax/yr (2013)	Flts/yr (2013)	Dates	Flights captured	Actual flights
CGN	9.1M	120 400	24/5 - 12/6/2014	4499 (61.2%)	7346 approx.
EDI	9.8M	111 736	27/6 - 17/7/2014	4851 (63.3%)	7662 approx.
MAN	20.7M	169 497	5/11 - 12/11/2013	1767 (55.0%)	3211 approx.
MEL	30.5M	220 824	27/1 - 10/2/2014	8617 (91.3%)	9194 approx.
STR	9.7M	122 818	14/7 - 3/8/2014	5019 (67.6%)	7425 approx.
SVO	31.6M	255 570	14/7 - 3/8/2014	9810 (61.6%)	15913
ZRH	24.9M	262 227	24/8 - 25/9/2014	19871 (77.2%)	25754

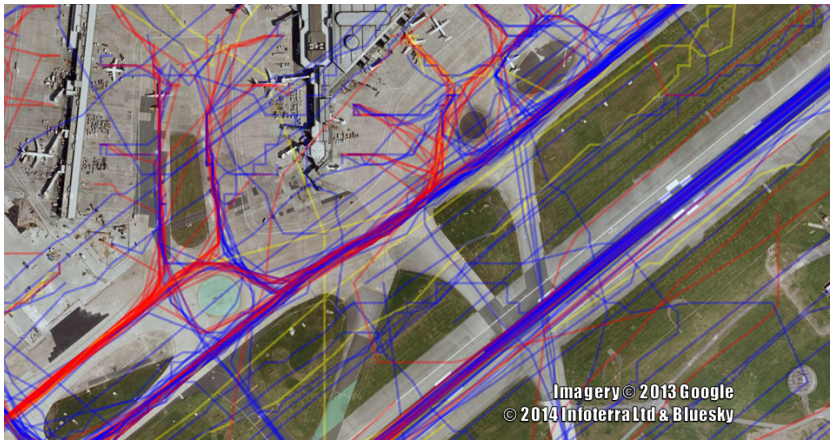


Figure 3: Ground movements at Manchester Airport: raw tracks from FR24 prior to snapping or cleaning.

5.2 Cleaning coordinates

Each time-coordinate $c = (x, y, t)$ taken from FR24 consists of a longitude x, latitude y and timestamp t. Time-coordinates in the raw data also include altitude, but this is ignored since this is always zero for Ground Movement. Being real-world data, there is considerable noise in the coordinates for a number of reasons:

- the location of the GPS receiver varies with different aircraft;
- miscalibration;
- reflected signals or corruptions in the received data;
- missing time-coordinates (either not transmitted or not received).

Much of this noise is removed by the snapping process described in Section 5.3, but some preprocessing is required to remove outlying time-coordinates that occasionally occur in the FR24 data. This process is given in Algorithm 2, in which: C denotes the list of all time-coordinates in the track; c_i is the i-th time-coordinate; F_{all} is the set of all failing time-coordinates, and F is a subset of F_{all} with size j. Two tests are performed for each time-coordinate.

1. c_i must not be more than a threshold distance from the overall track, determined by the locations of the previous and next time-coordinate. That is, the distances $c_{i-1} \to c_i$ and $c_i \to c_{i+1}$ must not be more than 5 times the distance $c_{i-1} \to c_{i+1}$. To avoid errors with very small movements, this test is only applied if $c_{i-1} \to c_i$ and $c_i \to c_{i+1}$ are both greater than 100m.

2. c_i must not introduce a sharp turn above a certain speed. The angle turned to take an aircraft through $c_{i-1} \to c_i \to c_{i+1}$ and the average speed of the aircraft from c_{i-1} to c_{i+1} are both calculated. The test fails if the aircraft turns more than 60° at over 50m/s, 90° at over 30m/s, 120° at over 20m/s, 130° at over 16.7m/s, 140° at over 13.3m/s or more than 150° at over 10m/s. These numbers are chosen based upon observations of the aircraft movements and the normal taxi speeds and turning angles, but were successful in detecting most outlying time-coordinates in the test data.

An outlying time-coordinate can cause time-coordinates on either side of it to also fail, because the resulting track has a sharp deviation. For this reason, the algorithm does not simply remove all failing time-coordinates from the track. Once a set of failing time-coordinates is found, each failing time-coordinate is removed in turn (steps 12 to 18 of Algorithm 2, with size $j = 1$) and the tests are re-run on the updated tracks, until a track with all time-coordinates passing the tests is found. If this fails, the number of time-coordinates removed in each pass (the size, j) is increased until a valid track is found. An outer loop repeats this whole process in case the removal of all initially-failing time-coordinates

results in a track with additional failures. This can result in a large number of alternative tracks being tested: a maximum n *choose* k for $1 \leq k \leq n$ where n is the number of outlying time-coordinates. However, most of the tracks in the raw data sets we processed had a maximum of one or two outlying time-coordinates.

Algorithm 2 Cleaning process

```
1: F_all ← ∅
2: for all c_i in C do
3:    if c_i fails the tests then
4:       Add c_i to F_all
5:    end if
6: end for
7: repeat
8:    if F_all contains more than 80% of C then
9:       Discard C and stop
10:   end if
11:   for j = 1 to |F_all| do
12:      for all F of size j in F_all do
13:         Create a new track T, a copy of C with F removed
14:         Test all c_i in T
15:         if all c_i in T pass then
16:            Record success, replace C with T, and stop
17:         end if
18:      end for
19:   end for
20:   /* Only reach here if, having removed F_all, some new c_i in C fail */
21:   Repeat steps 2 to 6
22: until No c_i in T fail the tests
```

5.2.1 Validation and demonstration

Figure 4 illustrates an outlying time coordinate (in red) along with the corrected track (in blue). In this case, only one coordinate in the raw data was an outlier, being located on the runway. It can be observed that this was successfully removed by the coordinate cleaning process.

The number of tracks handled at each stage of cleaning, for the studied airports, are given in Table 4. For most of the airports, the majority of tracks passed all of the tests. Typically a few of the tracks were found to be empty (that is, having no time-coordinates) and were discarded. For each airport, the number of tracks failing at least one test is given, along with the numbers that were successfully cleaned or found to be uncleanable and so discarded. The final column gives the number of tracks returned by this stage and this expressed as a percentage of the raw tracks.

Table 4: Number of flight tracks handled by each stage of the cleaning process for each airport.

Airport	Passing	Empty	Failing	Cleaned	Uncleanable	Tracks returned (% of raw)
CGN	3754	25	720	719	1	4473 (99.4)
EDI	4597	53	201	201	0	4798 (98.9)
MAN	1521	6	240	239	1	1760 (99.6)
MEL	4745	142	3730	4745	1	8474 (98.3)
STR	4353	27	638	633	5	4986 (99.3)
SVO	7644	101	2065	2065	0	9709 (99.0)
ZRH	10453	164	9254	9254	0	19707 (99.2)

Figure 4: Example of cleaning time-coordinates on a movement at Manchester Airport. The original track is in red, the cleaned track is in blue (blue overlaying red for most of the track). One coordinate in the raw data was an outlier, located on the runway, which was successfully removed.

Table 5: Notation used in Algorithm 3. Capitals are used for sets of multiple items, lowercase for single items in a set.

Symbol	Description
c	An original time-coordinate from GPS or other source
C	The cleaned list of time-coordinates generated by Algorithm 2
c_i	Time-coordinate at position i in C
c_{first}, c_{last}	First and last time-coordinates C
e	An edge on the taxiway graph
E	A set of edges
E_i	A set of potential edges for a coordinate c_i
E_{first}, E_{last}	Edges sets for first and last coordinates
s	The stand visited in a single aircraft movement
P	The path corresponding to an aircraft movement (a list of edges representing a traversal across the graph)
t_{i_in}, t_{i_out}	Times that an aircraft entered and exited edge e_i
$\|E\|$	Vertical bars used to indicate size of sets. Here, the number of edges in E

5.3 Snapping coordinates

In order to perform useful analysis of real aircraft movements, it is necessary to map the raw (cleaned) time-coordinates of flight tracks onto the graph of taxiways at the airport. The overall algorithm for completing this process is given in Algorithm 3, and described in more detail below. The notation used within the algorithm is given in Table 5. A tool implementing this procedure has been made freely-available, and is detailed in Section 6.2.

The full snapping process can be summarised in 7 stages. For the purposes of illustration, reference is made to the track of one aircraft movement at Manchester Airport. The original flight track, overlaid on the airport surface and taxiways in Google Earth$^{\text{TM}}$ is presented in Figure 5, and the final resulting snapped path is shown in Figure 11. A pre-processing stage (step 1) iterates over the raw track C, measuring the distance of each time-coordinate from the airport: if the track leaves the vicinity of the airport then returns, it is divided into multiple tracks which are processed separately. This stage also screens out tracks with fewer than a fixed number of 10 coordinates. This was the minimum used in our example: given the complexity of the taxiway graph a track with fewer coordinates was unlikely to have enough detail to fully determine the path taken.

1. Select candidate snap edges (steps 4-7): a threshold radius π is predetermined (10m in this work). For every coordinate c_i, get the list E_i of edges that are within π. If there were not enough coordinates within the threshold radius of an edge, abort. This is illustrated for a single coordinate in Figure 6. The raw GPS coordinate c_i is shown in blue and labelled "org" (for original). 5 edges are within the threshold distance from coordinate org, and the nearest coordinate on each edge is marked by a numbered circle (coordinates 0 and 3 are co-located at the coordinate where their corresponding edges meet). The edges are coloured differently for clarity. Figure 7 shows all of the candidate edges

Algorithm 3 Snapping process

1: If C has multiple visits to airport, divide. Discard tracks with small number of coordinates.
2: $R \leftarrow \varnothing$
3: **repeat**
4: **for all** c_i in C **do**
5: Allocate E_i, the set of taxiway, stand & runway edges within 10m of c_i
6: **end for**
7: Check track validity: if fewer than 80% of E_i have $|E_i| > 0$, skip to step 32
8: Reduce E_{first} and E_{last} to runway or stand edge nearest c_{first} and c_{last}
9: Determine which stand s was visited; remove any competing stands or edges from all E
10: Use pairs of c_i with $|E_i| > 1$ to reduce edge sets between them to "most likely" edges
11: **for all** E_i with $|E_i| > 1$ **do**
12: delete all edges in E_i except the closest edge to c_i
13: **end for**
14: **for all** E_i with $|E_i| > 1$ **do**
15: **if** E_i contains taxiway edges **then**
16: remove stand and runway edges from E_i
17: **end if**
18: **end for**
19: **for all** E_i with $|E_i| > 1$ **do**
20: choose one edge at random in E_i and delete the rest
21: **end for**
22: **for all** E_i with $|E_i| = 1$ **do**
23: Add e_i to R
24: **end for**
25: Complete R by taking the shortest path between disjoint edges
26: Remove branches from R that do not take place on or within a maximum distance of a stand
27: **for all** e_i in R **do**
28: Calculate t_{i_in} and t_{i_out}, given times for the c_i, c_{i-1} and c_{i+1} nearest the boundary between e_i, its predecessor e_{i-1} and its successor e_{i+1}.
29: **end for**
30: Drop runway edges from R
31: If R has more than 1 runway visit, split into separate movements, and validate each.
32: **if** $R = \varnothing$ **then**
33: Displace coordinates
34: **end if**
35: **until** $R! = \varnothing$

which were found for the track. Runways are included in the set of possible edges for snapping so that the full path of the aircraft on the ground can be determined and the ends of its movement verified. These are removed in step 30.

2. Ensure termini validity (steps 8-9): choose the most likely stand visited, check that at least 1 runway is visited, illustrated in Figure 8.

3. Refine snappings (steps 10-19): if any time-coordinate has only one potential edge to snap to, it is snapped to this edge. We can then use these edges as end-points for parts of the path, retaining only candidate edges that form a path between them. For a pair of time-coordinates c_a and c_b that have only one candidate edge each, the n-shortest paths between c_a and c_b are calculated. For each c_i between c_a and c_b, the multiple candidate edges E_i are reduced to only edges on one of these paths. After this, for any c_i which still has multiple candidate edges, the nearest edge is chosen. Where a coordinate is equally distant from two edges (for example, if it happens to be located precisely on the join between two edges): if any of the edges is part of a taxiway, any stand or runway edges are dropped. Finally, if there are still multiple candidate edges for a c_i, then one edge is chosen at random. Figure 9 shows the example path after this stage.

4. Complete path (step 25): fill in gaps with the shortest paths to make a complete path (see Figure 10).

5. Remove unwanted branches (step 26): occasionally branches remain in the path due to snapping errors earlier in the process. For example, if the aircraft was slightly to the left of the taxiway, one of the coordinates on the track may have snapped to a taxiway diverging to the left. To find such branches, we iterate over the edges in the path looking for a U-turn (a pair of visits to the same edge, in opposite directions). We then work outward from the turn point, removing matching pairs of edges until an unmatching pair is found. Branches that terminate on a stand are retained, as in this case they are almost certainly the turn associated with push-back. This process is illustrated in Figure 14, where nodes are black circles, edges are labelled with letters, and steps in the snapped path are numbered. The original path was 8 steps, visiting ABCDDCEF. A U-turn found at steps 4 and 5 (DD) is removed, reducing the path to ABCCEF. Working outwards from there, steps 3 and 6 are also paired (CC) so will be removed, reducing the path to ABEF. Steps 2 and 7 do not match (B and E), so will remain in the final path.

6. Time calculation (step 27): using the times associated with the original time-coordinates, determine when the aircraft reached each edge. This process is explained in more detail in Section 5.4.

7. Final cleaning (steps 30-31): edges associated with the runway are now removed. If the movement comprises an arrival and a departure (runway to stand to runway) then it is split into two movements. If any of the final paths are invalid (that is, not being either runway to stand or stand to runway) they are discarded. The final resulting snapped path for the example is shown in Figure 11.

If the process is unsuccessful in constructing a valid path (we deem this to

be the case if $\leq 80\%$ of the coordinates snapped to an edge, or the path does not start and end on both a stand and a runway), the coordinates are displaced and the process is repeated from step 2. Displacements follow a grid of increasing distances from the original coordinates, separated by 10m, following the pattern shown in Figure 12. An example of a displaced raw flight track is given in Figure 13.

It may be desirable to add a repair operation for invalid paths. For example, if the path does not visit a stand, one could explore the neighbourhood of the last edge to find the nearest runway node or gate node. However, this is not done at present because it could be argued that adding path information in this way diverges from the original raw data too much.

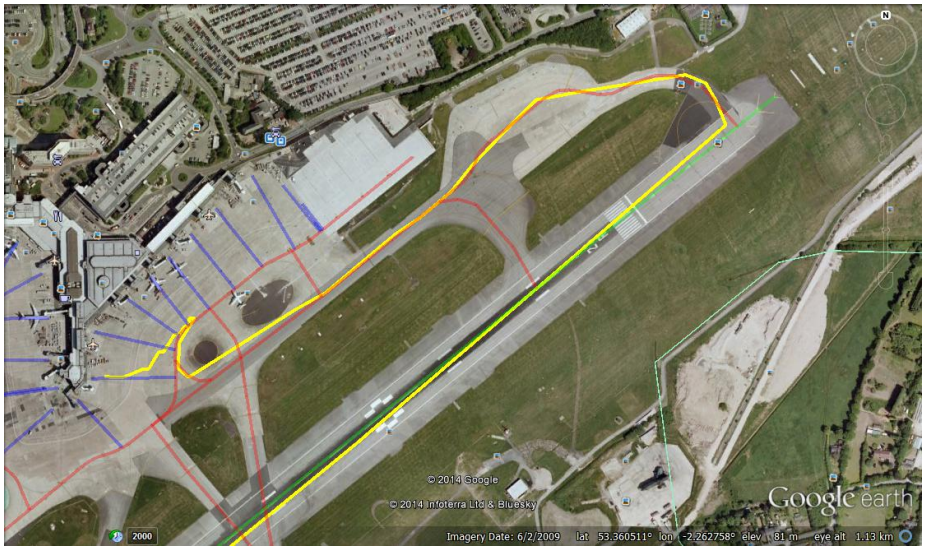


Figure 5: The raw flight track to be snapped (yellow) overlaid on the taxiways (red), stands (blue) and runway (green).

5.3.1 Validation and demonstration

By following these steps, the example data was successfully snapped, generating a usable dataset. For illustration, Figure 15 shows the final paths for the flights at Manchester Airport which were shown in Figure 3, following the cleaning and snapping process.

Table 6 reports the number of tracks handled at various stages of the snapping process. For each airport, "*Tracks*" gives the number of tracks returned by the cleaning stage, from which "*Add 1*" additional tracks were separated out from tracks having multiple visits to the airport (step 1) in Algorithm 3. "*Disc.*" shows the number of tracks having fewer than 10 time-coordinates, which were discarded at stage 1. This means that, for each airport, the algo-

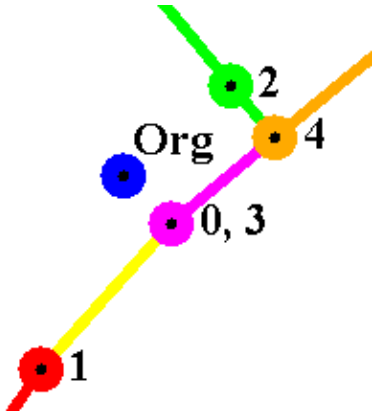


Figure 6: Snapping, after stage 1. Candidate edges for one coordinate.

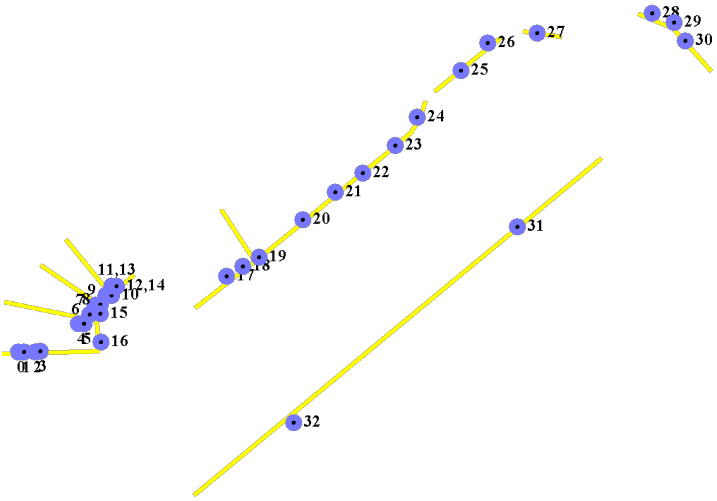


Figure 7: Snapping, after stage 1. All candidate edges for the track, with original coordinates. Left-most are multiple candidates for stands.

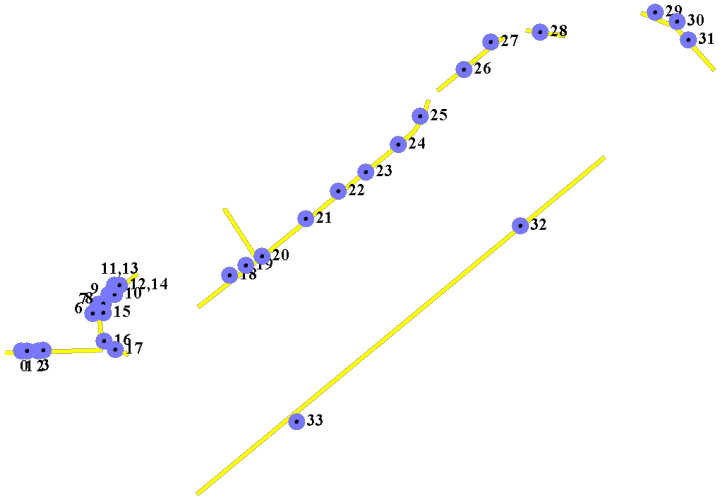


Figure 8: Snapping, after stage 2. Candidate edges, after location of stand and runway. Note that multiple stands on the left have been removed, but U-turn near the stands and possible alternative edge near coordinate 20 remain.

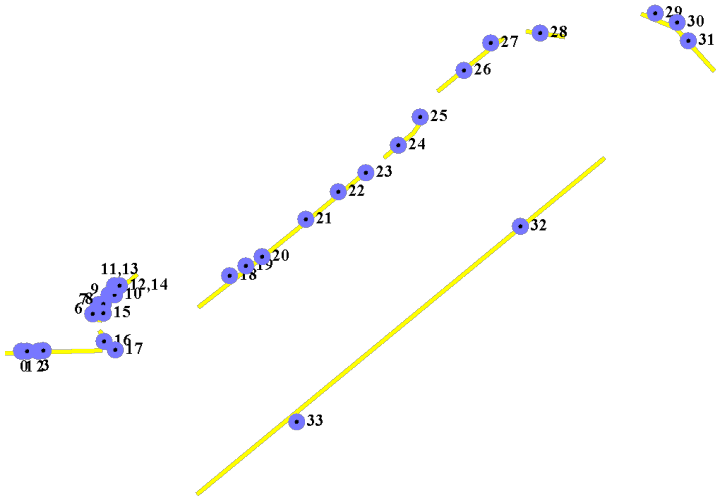


Figure 9: Snapping, after stage 3. Candidate edges, after choosing 'most likely' edges - those falling on the shortest path between coordinates where there is only one candidate edge. Note that the U-turn near the stands remains, but the possible alternative edge near coordinate 20 has been removed.

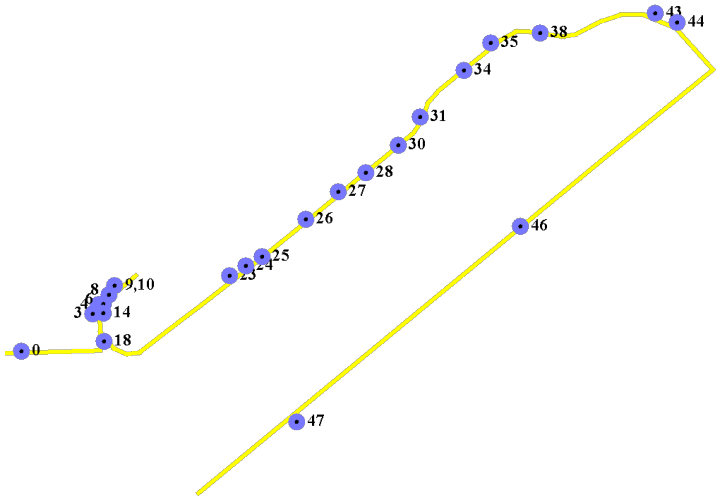


Figure 10: Snapping, after stage 4. The completed path, with the shortest path assumed between gaps in the real coordinates. In this case, no branches remain to be trimmed.

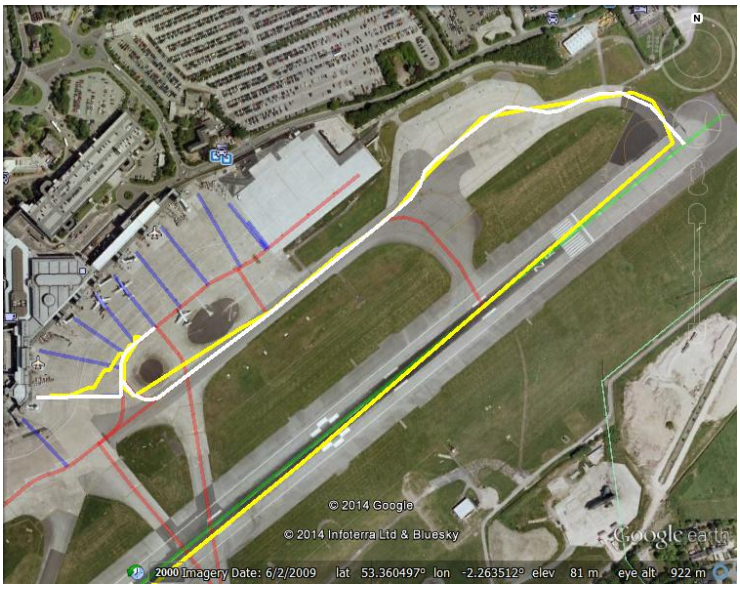


Figure 11: The final snapped path (white), overlaid on the raw flight track to be snapped (yellow), taxiways (red), stands (blue) and runway (green).

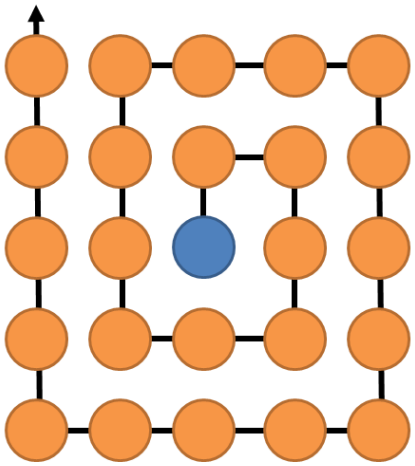


Figure 12: Pattern in which coordinates are displaced in an attempt to correct for an error consistent across a whole track. Original coordinate is in the centre.

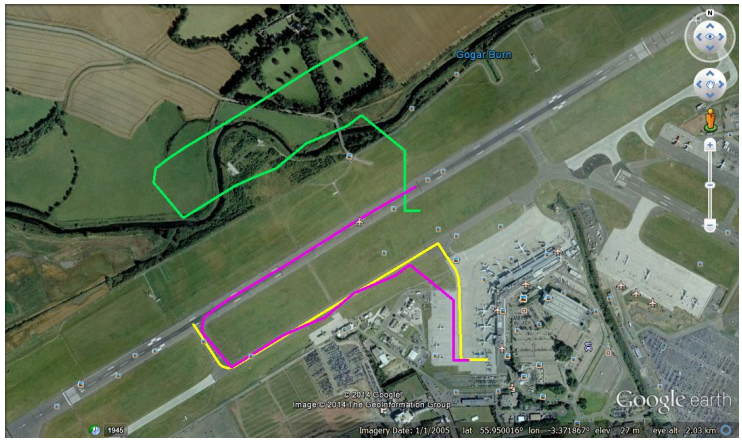


Figure 13: Snapping, displacement. Track at Edinburgh Airport, with original track coordinates in green, displaced track coordinates in purple and final snapped path in yellow. (Edinburgh was used for this figure as this flight was a particularly good example of the issue)

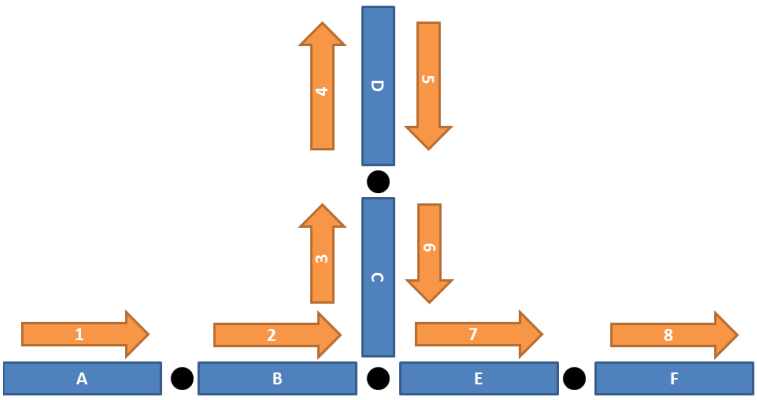


Figure 14: Snapping, branch trimming.

rithm attempted to snap "*To snap*" tracks to the taxiways. "*Snapped*" gives the number of tracks that were snapped, and "*Add 2*" gives the number of these which were found to contain multiple movements at the airport, and were split in step 31. *Out* gives the number of aircraft ground movement paths remaining in the data at the end of this process.

A substantial number of tracks for CGN and SVO Airports were found to contain only a few time-coordinates on the ground, with ADS-B transponders either being disabled or out of range near the terminal building. For most airports, between 21% (MAN) and 39% (CGN) of tracks could not be snapped successfully (this rose to 66% for SVO): visual inspection of the tracks revealed that the reason for this was that many of these were missing time-coordinates near the stands. Again, this is likely to be caused by the ADS-B transponder being disabled or out of range near the terminal. It is reasonable to assume that with increasing take-up of ADS-B, and as the system becomes mandatory, the amount of missing data will decrease, thereby reducing the impact of this issue. For MEL, the threshold distance for a coordinate to snap to an edge had to be raised to 25m as there seemed to be an increased tendency for coordinates to cover the full width of taxiways at that airport. For all of the airports, except CGN and SVO, tracks amounting to over 40% of real traffic levels were successfully snapped to paths along the taxiways.

The methodology has successfully processed data from publicly available websites to produce near-realistic datasets. However, not all aircraft will appear, since some will not have been present in the ADS-B data, and the tracks of others may not have been usable and will have been removed in the snapping and cleaning phases. For each airport, "*% of FR24*" gives the resulting paths as a percentage of the original tracks from FR24. "*% of actual*" gives the number of paths as a percentage of the known movements during the sample period (taken from the references given in Section 5.1.1).

Each movement in the dataset contains an actual taxi path taken, with the real time at the start and end of the path, as well as some intermediate

Table 6: Number of flight tracks handled at different stages of the snapping process for each airport

Airport	Tracks	Add 1	Disc.	To snap	Snapped	Add 2	Out	% of FR24	% of actual
CGN	4473	0	2337	2136	1294	4	1298	28.9% of 4499	17.7% of 7346
EDI	4851	0	207	4591	3358	2	3360	69.3% of 4851	43.9% of 7662
MAN	1760	110	79	1791	1416	4	1420	80.4% of 1767	44.2% of 3211
MEL	8474	0	2409	6065	4801	0	4801	55.7% of 8617	52.2% of 9194
STR	4986	0	598	4388	2831	2	2833	56.5% of 5018	40.2% of 7056
SVO	9709	41	4641	5109	1755	1	1756	17.9% of 9810	11.0% of 15913
ZRH	19707	0	6313	13394	10320	50	10370	52.2% of 19871	40.3% of 25754

coordinates. It can, therefore, be used to determine timings as well as spatial paths, as explained in the next sub-section.

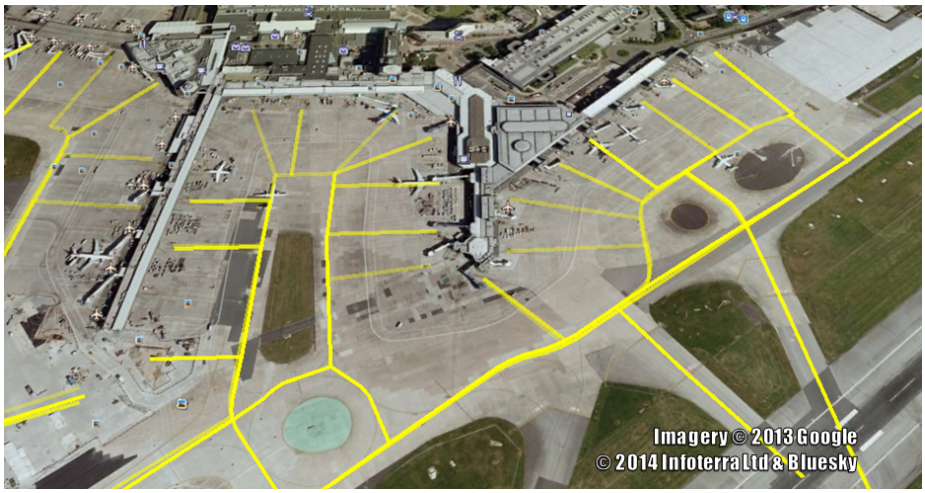


Figure 15: Ground movements at Manchester Airport: paths resulting from snapping FR24 tracks.

5.4 Timings

Once the time-coordinates have been snapped to edges on the taxiway graph, the time that the aircraft reached each edge can be determined. This allows analysis of taxi speeds and interactions between different aircraft. Currently, we are interested in the traversal time for each edge, which is equivalent to having the time that an aircraft passes each node. Each raw coordinate has an associated timestamp t. The time that the aircraft reached the end of an edge (node n) is calculated by taking the t corresponding to the two c_i on the path spatially nearest to n, and assuming a constant speed between them (clearly a more sophisticated approach, allowing for acceleration, could include timings from all coordinates). A perpendicular line is projected from the edge to the coordinates that snapped to it. The aircraft is estimated to have reached the

coordinate where this line intersects the edge at t. This is illustrated in Figure 16. The speed can then be calculated from the distance along the edge and the time between the coordinates.

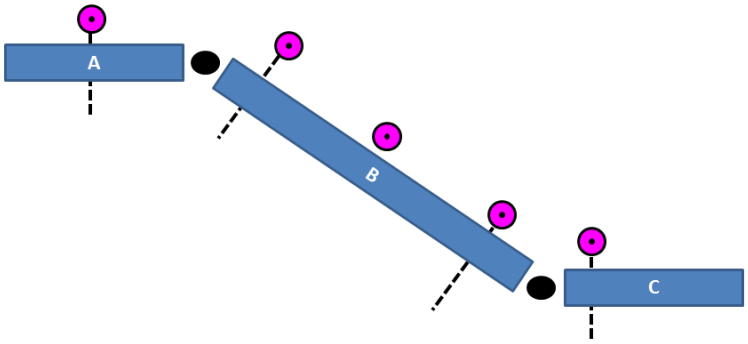


Figure 16: Calculation of edge timings after time-coordinates have been snapped to edges.

6 Tools for Generation, Snapping and Analysis

In Section 3, we described an algorithm for constructing taxiway layouts from OSM and stand coordinate data. In Section 5, we described several algorithms to snap real-world flight tracks taken from a site such as FR24 to the taxiway graph. These algorithms have been implemented in two tools, which have been made freely-available under the MIT Software Licence. The tools are implemented in Java and both source code and compiled Java™ (JAR file) executables are available. By making the source available it is hoped that the research community will be able to adopt them as needed, with improvements being similarly shared. The tools and corresponding instructions for their use can be obtained from `https://github.com/gm-tools/gm-tools/wiki`, with source code available from the main project page `https://github.com/gm-tools/gm-tools`. A preliminary demonstration of their capabilities was given in Brownlee et al. (2014).

The tools make use of two third-party libraries for manipulating XML. The JavaAPIForKML (`https://code.google.com/p/javaapiforkml`) was used to generate the KML and the JCoord (`http://www.jstott.me.uk/jcoord/`) library was used for handling lat/lon coordinates.

6.1 Generating a taxiway graph

The *Taxigen* tool has been developed to parse OSM data, with additional stand location information from another source if required, following the procedure in Section 3.2. It outputs a Ground Movement Definition (GM) file, the same

format as that already used by some simple benchmarks at the University of Nottingham's ASAP group (`http://www.asap.cs.nott.ac.uk/external/atr/benchmarks/index.shtml#groundmovement`).

TaxiGen takes two files as input: an OSM XML file, and a file either naming existing OSM edges to use as stands or giving coordinates of stands for which the tool will generate new attachment edges. A text file is output, giving the nodes and edges of the taxiway graph, with node coordinates in both lat/lon and x/y (in the form of UTM grid references), and edge lengths in metres. Additionally, the angles between adjacent edges can be output to a separate file. TaxiGen can also write out a KML file to represent the taxiways graphically in a tool such as Google Earth^{TM}. This can be helpful for debugging purposes.

6.2 Snapping coordinates

The algorithms described in Sections 5.2 and 5.3 have been implemented in the *SnapTracks* tool. SnapTracks takes two files as inputs. The first is a GM file describing the static positions of the stands, runways and taxiways as generated by the TaxiGen tool. The second is a tab-separated flatfile in which each row corresponds to an aircraft movement, with an ID and other metadata relating to the aircraft, and a sequence of raw lat/lon coordinates, optionally including separation times between coordinate pairs. The tool can also be configured with a coordinate and radius to identify an area in which to include flights (in case the raw data includes flight paths from other airports), parameters such as the step interval and maximum radius for displacing tracks if a set of raw coordinates cannot be snapped to the taxiways, and parameters to tune the cleaning process for removal of outlying coordinates. SnapTracks outputs:

- a flatfile with the cleaned raw flight tracks;
- a flatfile with the raw flight tracks translated to the snapped positions;
- two flatfiles giving data on the times that each aircraft travelled each edge;
- a GM file with the snapped paths for all aircraft;
- KML files visualising the aircraft movements.

6.3 Processing and analysis

GM2KML has been created to support the analysis of the data processed by TaxiGen and SnapTracks. It performs the following processing and visualisation tasks:

- count aircraft visiting each stand, either for the whole period or grouped into fixed-length intervals;
- count aircraft traversing each edge, either for the whole period or grouped into fixed-length intervals;

- compute average speeds for each edge, either for the whole period or grouped into fixed-length intervals;
- visualise the taxiways, stands and runways of a GM file in KML;
- visualise the number of aircraft visiting each stand in KML;
- visualise the movements of aircraft, either for the whole period or grouped into fixed-length intervals;
- visualise the paths taken by aircraft to reach each stand, either for the whole period or grouped into fixed-length intervals;
- visualise average speeds for each edge.

GM2KML was used to develop most of the visualisations in Section 7.

7 Case Studies

To illustrate some of the benefits of having this level of detail in the data, we now present several case studies, using the captured data to analyse the aircraft movements at the airports identified in Section 5.1.1. As a reminder, after cleaning and snapping the data for CGN, EDI, MAN, MEL, STR, SVO and ZRH airports contains 1298, 3360, 1420, 4801, 2833, 1756 and 10370 unique aircraft movements respectively. This compares with 7346, 7662, 3211, 7056, 15913 and 25754 actual movements during the same periods. There are many uses for this data and we now outline a few possibilities to demonstrate the value of this work.

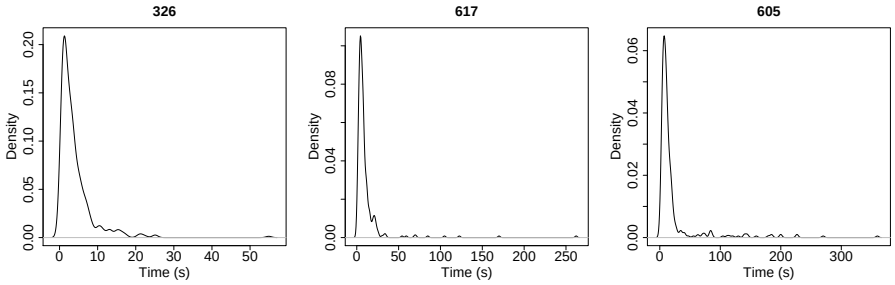
Data like this is crucial to developing a better understanding of Ground Movement at real airports. Both qualitative and quantitative analysis of Ground Movement will be of benefit to researchers in this area. Quantitative measures of taxi times will be useful in developing the realistic models of taxi speed such as (Chen et al., 2011; Ravizza et al., 2014), that are required for optimisation of real-world taxi routes.

7.1 Case Study 1: Taxi times

To illustrate the level of detail in the snapped FR24 data, Figure 17 gives density plots generated by the *R* statistical package for the calculated taxi times on three edges at Manchester Airport. Edge 326 is part of a major taxiway en route to the stands where most aircraft pass through quickly, but the tail shows occasional delays due to increasing traffic levels. Edge 617 is part of the taxiway leading to the departure runway, with times in two groups corresponding to busy and light traffic conditions. Edge 605 is near the runway entrance, with a much wider spread of taxi times, up to a minute, corresponding to aircraft queueing to take off. The approximate locations of these edges are shown in Figure 2.

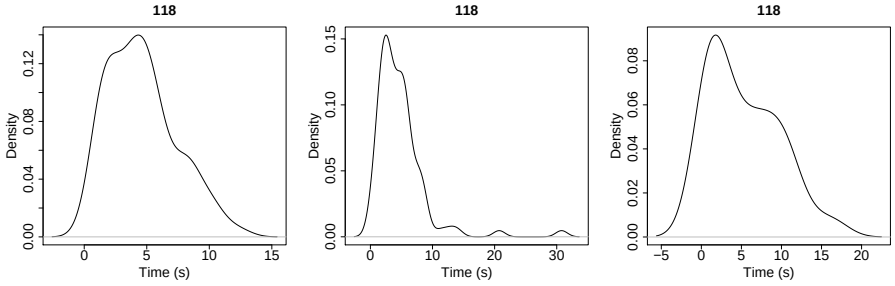
Figure 18 shows density plots for Edge 118 during different 3-hour periods of the day, aggregated over 5 weekdays. Edge 118 forms part of the taxiway that is

Figure 17: Density plots of taxi times for different edges.



(a) Edge 326 (b) Edge 617 (c) Edge 605

Figure 18: Density plots of taxi times for edge 118 during different time periods.



(a) 12:00 to 15:00 (b) 15:00 to 18:00 (c) 21:00 to 24:00

taken by arrivals en route to the stands during the studied period. Note that the uncertainty in the taxi times varies depending on the time of day. In particular, during the busiest period (15:00-18:00), the distribution of times is narrower, but the outliers represent a larger delay than in the other time periods.

Alternative representations of this data are possible. In Figure 19, taxiway edges are coloured according to average taxi speed during the period. The tool can also show colourings corresponding to speeds within equally distributed intervals over the data period.

7.2 Case Study 2: Breakdown of taxi speeds

Approaches in the Ground Movement literature make various assumptions about taxi speeds, including: constant speed according to aircraft type (Roling and Visser, 2008), speed limited by taxiway type (Balakrishnan and Jung, 2007), or speeds influenced by the number of straights and turns in the path (Weiszer et al., 2014; Pesic et al., 2001; Gotteland et al., 2001). To study this latter approach in more detail, for each of the example airports, the average speeds

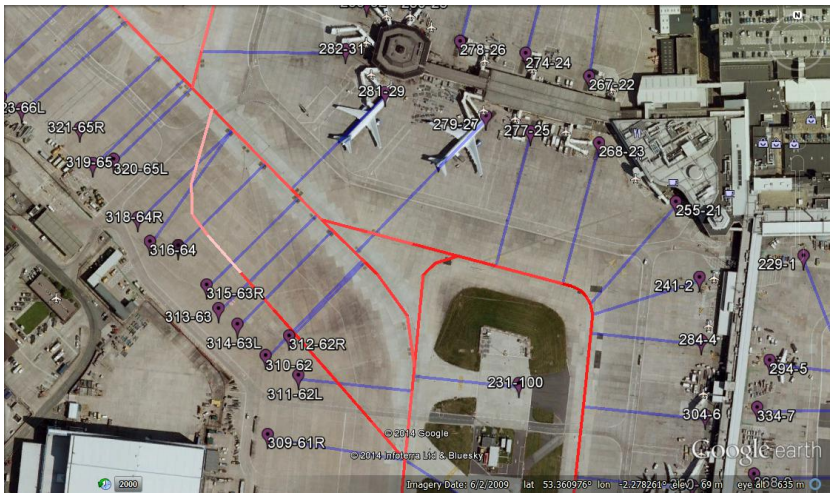


Figure 19: Visualisation of average taxi speeds at Manchester airport during the sample data period. Taxiways are red for low speeds incrementally changing to white for high speeds. Stands (no speeds associated) are blue.

were determined over all movements for edges forming parts of straight and turn paths respectively. Two rules were used to classify segments of paths as straights or turns. The first classified a segment as a turn if it turned through more than 30 degrees in 100m or less, otherwise it was deemed to be straight. The second classified a segment as a turn only if it turned through more than 60 degrees within 100m. A path taken by an aircraft is illustrated in Figure 20, with segments of the path coloured to show straights and turns when using the two rules. A similar analysis for ground movement at Detroit Metropolitan Wayne County Airport was conducted by Rappaport et al. (2009).

Speeds for straights and turns are given in Table 7. These figures exclude edges with speeds over 50m/s, to prevent taxiways near runways and the occasional noise from FR24 data skewing the mean figures. The table shows at each airport, for the two classification rules: the total number of measurements for edges analysed and the number filtered out for having speeds over 50m/s (Edges / Filtered); the total distance covered by aircraft within turns and straights respectively; and the average speed and standard deviation for all aircraft on turns and straights.

For most airports, few edges were excluded from the aggregation due to speeds over the 50m/s threshold. This was only substantial for Cologne-Bonn, with over one third of edges excluded. This is likely to be because of the issues with noisy data that impacted on the snapping process for CGN. As one would expect, increasing the angle required to classify a segment as a turn always increases the fraction of the total distance classified as straights.

For all airports, there is a high standard deviation in the speeds for both turns and straights. Furthermore, the difference in the average speed for turns

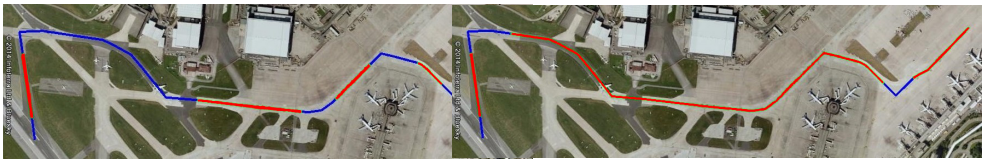
Table 7: Average speeds, for edges classified as straights and turns, according to two different rules.

Airport	Edges (Filtered)	Turn if...	Turns (m)	Straights (m)	Avg Spd Turns (sd)	Avg Spd Straights (sd)
CGN	45734	$> 30°$ in 100m	402 749	1 175 226	12.47 (10.17)	13.4 (10.72)
	(16497)	$> 60°$ in 100m	171 583	1 406 392	10.24 (8.54)	13.63 (10.72)
EDI	117132	$> 30°$ in 100m	1 637 379	2 915 511	7.28 (5.8)	9.98 (5.87)
	(479)	$> 60°$ in 100m	1 174 968	3 377 923	6.45 (4.61)	9.5 (6.15)
MAN	73766	$> 30°$ in 100m	1 131 004	1 415 316	9.81 (9.05)	10.52 (8.43)
	(1071)	$> 60°$ in 100m	582 408	1 963 911	9.19 (8.88)	10.27 (8.55)
MEL	137641	$> 30°$ in 100m	2 424 726	4 117 813	13.35 (10.91)	13.2 (10.01)
	(16981)	$> 60°$ in 100m	1 907 574	4 634 965	12.99 (10.9)	13.7 (10.23)
STR	193966	$> 30°$ in 100m	1 182 544	4 480 050	9.93 (7.64)	13.37 (8.32)
	(4135)	$> 60°$ in 100m	619 861	5 042 733	9.22 (7.37)	12.67 (8.3)
SVO	68768	$> 30°$ in 100m	932 286	1 461 945	9.38 (9.71)	11.39 (10.2)
	(8371)	$> 60°$ in 100m	651 846	1 742 385	9.14 (9.64)	11.15 (10.19)
ZRH	423014	$> 30°$ in 100m	4 660 431	8 747 144	11.25 (10.27)	10.74 (9.16)
	(49010)	$> 60°$ in 100m	2 774 655	10 632 921	10.59 (9.54)	10.62 (9.21)

and straights is not very large. This is because the taxi speed is affected by far more than just turning: factors including airport congestion, proximity to runway or stands, prevailing weather, aircraft type and flight crew will all impact on taxi speed to some degree. This provides strong motivation for using a more sophisticated model than the simple rule-based approach for determining taxi speed for realistic ground movement.

It is also worth noting that the average speed for turns at Zurich is higher than than for straights. Given the high standard deviation it is difficult to interpret much from this: a partial cause might be the delay imposed by runway crossings (which are straight) at a busy airport. This will be explored further in section 7.5.

Figure 20: Illustration of the same path taken by an aircraft at Manchester Airport, with the edges categorised as straights (red) or turns (blue) according to different rules.



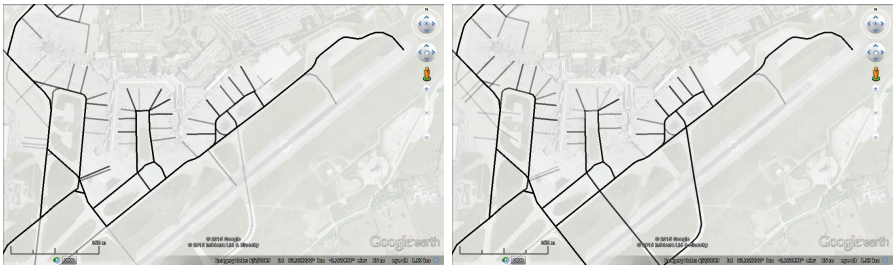
(a) Turn if $> 30°$ in 100m (b) Turn if $> 60°$ in 100m

7.3 Case Study 3: Traffic levels

Figure 21 shows traffic levels, in terms of the number of movements per taxiway at different times of the day. This is useful for the purpose of identifying likely conflict areas and bottlenecks. The paths taken by individual aircraft to a gate are rendered in KML with a high level of visual transparency. Paths taken

by multiple aircraft gradually become more opaque, readily showing the more frequently used paths.

Figure 21: Number of aircraft using taxiways at different times in the day. Note increased use of runway crossings in the late afternoon (corresponding to a switch in airport operating mode during busy period, where runway 05R/31L comes into use).



(a) 12:00-15:00 (b) 15:00-18:00

7.4 Case Study 4: Stand use

The gate or stand allocation problem (Atkin, 2013; Dorndorf et al., 2007) is closely related to Ground Movement. A common objective for this problem is satisfying airline or airport preference for particular gates: a proxy for this being the frequency with which real aircraft of an airline visit each gate. The FR24 data includes airlines and flight numbers for most flights (1359 of the 1767, or 77% in the case study data for Manchester Airport). Furthermore, for Ground Movement, it is useful to know which gates are more frequently used in order to predict bottlenecks. The number of aircraft using each stand during the period can be easily determined from the snapped paths, and a visualisation of stand use in KML is also possible. The number of aircraft visiting a subset of the stands at Manchester is plotted in Figure 23, and rendered in Figure 22.

An analysis of the taxiways used to visit a particular gate is also of relevance to stand allocation. This will allow appropriate constraints to be placed on allocations to reduce conflicts between stands that share taxiways. This is illustrated in Figure 24. Rendering is achieved by overlaying faint lines, as in Figure 21.

7.5 Case Study 5: Runway crossings

Many airports have taxiways which cross live runways. Taxiing aircraft proceed to a holding point and the flight crew await clearance from controllers to cross the runway. Inevitably, this leads to delays which should be accounted for in realistic ground movement modelling.

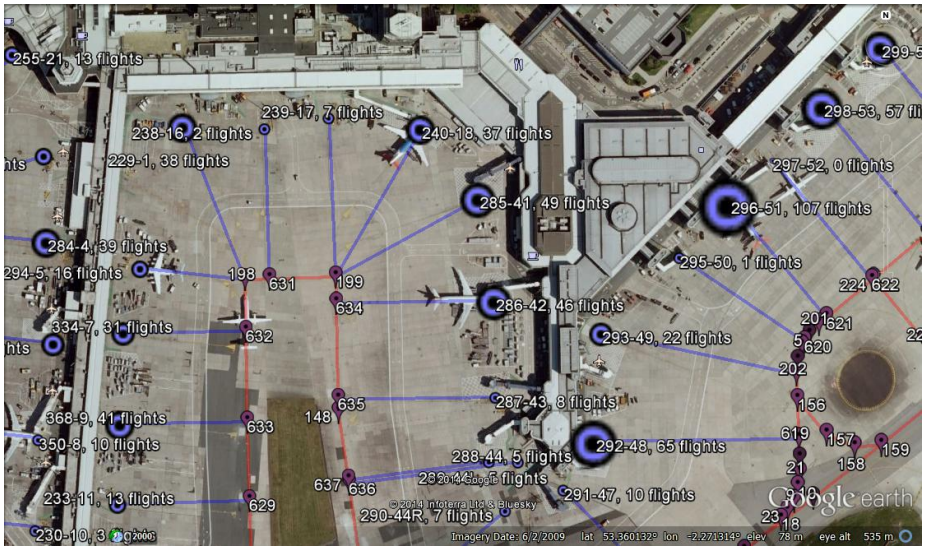


Figure 22: Visualisation of the number of aircraft visiting each gate at Manchester Airport during the sample data period.

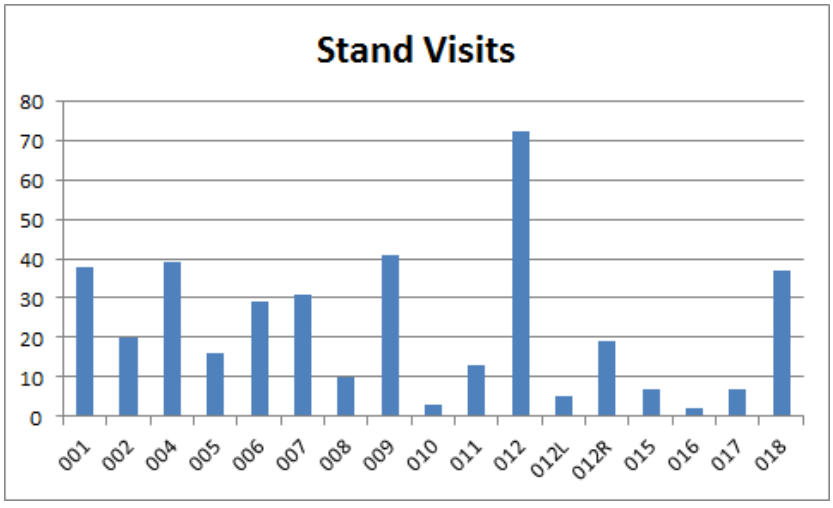


Figure 23: Bar chart showing the number of aircraft visiting each gate at Manchester Airport during the sample data period.

Figure 24: Illustration of the taxiways used to visit gate 5 at Manchester airport during the sample data period. The majority of aircraft take the path immediately adjacent to the gate, though a faint line shows that a few take the parallel taxiway on the right. The location of the turn associated with push-back manoeuvres is also clearly shown.

In this study, the movements at Zurich Airport were analysed to measure the delay caused by two runway crossings. At ZRH, taxiways J and K cross runway 10/28, connecting the main terminal stands with runway 14/32. These are illustrated in a stylised diagram of Zurich Airport in Figure 25. In the analysed data, most movements are northbound on J (for departure on runway 32) and southbound on K (arrivals from runway 14). Nodes on the taxiway graph were selected on either side of the runway crossings. These nodes were at the first intersection on either side of the runway, so any aircraft passing through both nodes had completed a runway crossing. The time difference between the two is the wait time plus the time to complete the crossing itself. The holding point for each is between these nodes and the runway, so the delay accumulated for one aircraft by waiting to cross the runway would be counted this way. Any additional queuing time outside these points will be missed.

Movement north along J was measured between nodes 39 to 352, a distance of 236m. Movement south along K was measured between nodes 355 and 356, a distance of 234m. In the data, there were 554 movements north on J, and 1329 movements south on K. Given the average taxi speed for straight movements at the airport of 10.74m/s (Section 7.2), these movements should be completed in an average time of 22.0s and 21.8s respectively. Histograms of the times taken by aircraft for these two movements are given in Figure 26. Both have a long tail, which is indicative of most aircraft having a short delay with occasional delays that are much longer. This is reflected in the difference between the average and median times for the movements. The average time for northbound crossings

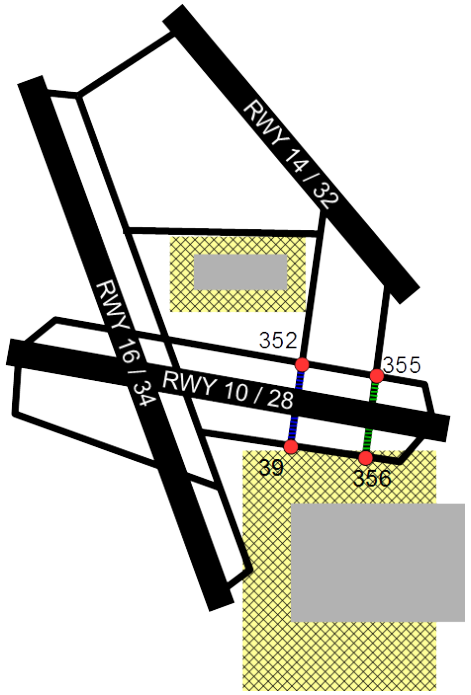
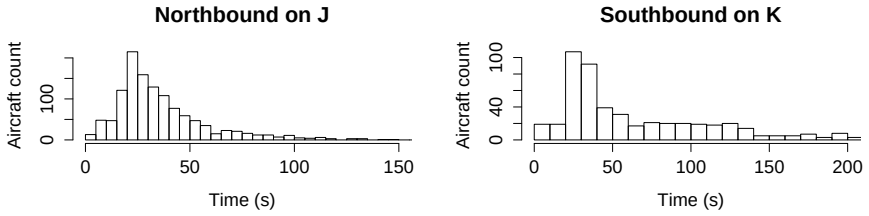


Figure 25: Stylised diagram of Zurich Airport showing the terminals (grey), runways (black), major taxiways (black) and major aprons (yellow with black hatching). The red circles with numbers identify nodes referred to by the case study in Section 7.5. Taxiway J (352-39) is dashed in blue and taxiway K (355-356) is dashed in green.

was 36.2s, with a median time of 29.0s. The average time for southbound crossings was 72.6s, with a median of 43.3s. Comparing the average times, the average delay for northbound crossings was 21.4s, and for southbound was 50.8s. There are at least two reasons for this. Firstly, northbound movements are departures, which can be better timed to coincide with gaps in runway use, leading to less delay. Secondly, according to Ravizza (2013), conversations with field staff at Zurich Airport revealed that, as long as no heavy winds occur, the usual operating modes for the airport mean that runway 32 is used for departures during quieter periods, before 7am and after 9pm. In contrast, runway 14 is used for arrivals during the daytime, covering the periods of high traffic. This is reflected in the larger number of movements southbound (arrivals from 14) than northbound (departures on 32). The runway being crossed (10/28) is used for departures during the day and arrivals after 9pm, but not at all before 7am. In combination, this means that there are fewer movements northbound on J that are likely to conflict with use of the runway, so there will be less chance of delay.

Figure 26: Time taken by aircraft to move between nodes on either side of runway 10/28 at Zurich Airport.



(a) Northbound on Taxiway J

(b) Southbound on Taxiway K

8 Conclusions

The airport Ground Movement problem is challenging, and finding route allocations for aircraft on the ground which are both achievable and practical is a complex problem for automated systems. Access to ground movement controllers' expertise is difficult to obtain and realistic data has not often been available for researchers in the past, either due to confidentiality issues or because recordings do not exist for the target airport. However, such information is a pre-requisite for the development of practical and efficient decision support tools for these airports. The research in this paper presents a solution for this problem. A method has been described here which will enable researchers to make use of information from publicly available websites in a way that has never before been possible. Information feeds from a number of different sources have been analysed and an amalgamation approach has been described. The paper

has also described an approach for cleaning and snapping real ground tracks taken from a site such as FR24, to allow analysis of the real movements of aircraft at an airport. The results of the approach have been shown graphically, and it has been observed that up to 80% of the ADS-B track information has been converted into usable aircraft movements along defined taxiways in the examples given. Further research may potentially be able to improve upon this. In the example given, usable datasets have been produced which contain up to 52% of the actual movements at seven international airports.

This research has resulted in the development of a number of useful tools, which will both automate the amalgamation of data sources and the cleaning and snapping of the data, as discussed in Sections 5.2 to 5.4, and allow a visualisation of the results in a number of ways. These tools are being made freely available to the community and can be found here: `http://www.github.com/gm-tools/gm-tools`.

A case study was presented in Section 7, illustrating the use of the developed tools upon the resulting datasets to analyse the traffic along various edges of the graph. It was interesting to observe that the density plots had obvious meanings, increasing the confidence in their validity and the accuracy of the datasets, however the data provides a richer source of information than was previously freely-available to researchers.

Although these freely-available sources of data are no substitute for the quality data that can be obtained through a partnership with an airport, they do provide a means for researchers entering the area to start making a contribution. Furthermore, they provide a means for a more comprehensive study covering a wide range of different airports globally, when obtaining partnerships with each individual airport may be impractical. Finally, they provide a means to potentially augment the data which airports can provide, allowing longer time periods to be analysed without increasing the costs for the airports.

Example layouts for several international airports have been made available, including a full benchmark Ground Movement problem for Manchester Airport, the third busiest airport in the UK. Additional benchmark problems would still be a welcome contribution to the research community, and we intend to make further data available in due course. The tools which we have made available also allow others to do so relatively easily and we look forward to accepting further contributions to the dataset repository.

In summary, this work has made a substantial contribution to the research community in this area, not only analysing the ways in which available data can be utilised and investigating algorithms for cleaning the data, but also in providing the tools for others to be able to expand their research.

Acknowledgment

This work was funded by the UK EPSRC [grant number EP/H004424/2]. Data processing was mostly conducted using the Microsoft Azure cloud platform, under a Microsoft Azure for Research award “Integrated Airport Operations

Optimisation". We are grateful to Manchester Airport for providing the data used in compiling the benchmark Ground Movement problem, and Zurich Airport for valuable discussion on ground movement operations.

Data access statement

All data and source code arising from this research is available via the URLs in the paper, or under doi:10.5281/zenodo.18730, with the exception of data arising from FR24 which cannot be shared due to licensing restrictions.

References

Airservices Australia, 2013. Movements at Australian Airports - Calendar Year 2013.
URL http://www.airservicesaustralia.com/wp-content/uploads/Airport-Movement-Calendar-YTD-April-2013.pdf

Airservices Australia, 2014. Movements at Australian Airports - January and February 2014.
URL http://www.airservicesaustralia.com/publications/reports-and-statistics/movements-at-australian-airports/

Atkin, J. A., 2013. Airport airside optimisation problems. In: Uyar, A. S., Ozcan, E., Urquhart, N. (Eds.), Automated Scheduling and Planning. Vol. 505 of Studies in Computational Intelligence. Springer Berlin Heidelberg, pp. 1–37.

Atkin, J. A., Burke, E. K., Greenwood, J. S., 2010. TSAT allocation at London Heathrow: the relationship between slot compliance, throughput and equity. Public Transport 2 (3), 173–198.

Balakrishna, P., Ganesan, R., Sherry, L., Dec 2010. Accuracy of reinforcement learning algorithms for predicting aircraft taxi-out times: A case-study of Tampa Bay departures. Transportation Research Part C: Emerging Technologies 18 (6), 950–962.

Balakrishnan, H., Jung, Y., Aug. 2007. A framework for coordinated surface operations planning at Dallas-Fort Worth International Airport. In: Proceedings of the AIAA Guidance, Navigation, and Control Conference, Hilton Head, SC, USA. Vol. 3. pp. 2382–2400.

Benlic, U., Burke, E., Woodward, J., 2014. Breakout local search for gate allocation problem. Submitted manuscript.

Brownlee, A., Atkin, J., Woodward, J., Benlic, U., Burke, E., 2014. Airport ground movement: Real world data sets and approaches to handling uncertainty. In: Proc. of the Practice and Theory of Automated Timetabling. York, UK.

Chen, J., Ravizza, S., Atkin, J. A., , Stewart, P., 2011. On the utilisation of fuzzy rule-based systems for taxi time estimations at airports. In: 11th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems. pp. 134–145.

Civil Aviation Authority, 2013a. Aircraft movements 2013. Retrieved 5/2/2015. URL http://www.caa.co.uk/docs/80/airport_data/2013Annual/Table_03_1_Aircraft_Movements_2013.pdf

Civil Aviation Authority, 2013b. Aviation trends - quarter 3 2013. Retrieved 5/2/2015.
URL http://www.caa.co.uk/docs/80/AviationTrends_Q3_2013.pdf

Civil Aviation Authority, 2014a. Aircraft movements 2014 (June and July). Retrieved 5/2/2015.
URL http://www.caa.co.uk/default.aspx?catid=80&pagetype=88&pageid=3&sglid=3

Civil Aviation Authority, 2014b. Size of reporting airports February 2013 - January 2014. Retrieved 5/2/2015.
URL http://www.caa.co.uk/docs/80/airport_data/201401/Table_01_Size_of_UK_Airports.pdf

Clare, G., Richards, A., 2011. Optimization of taxiway routing and runway scheduling. Intelligent Transportation Systems, IEEE Transactions on 12 (4), 1000–1013.

Deutsche Flugsicherung, 2014. Monthly statistics for May–August 2014.
URL https://www.dfs.de/dfs_homepage/en/About%20DFS/Facts%20and%20figures/Statistics/

Dorndorf, U., Drexl, A., Nikulin, Y., Pesch, E., June 2007. Flight gate scheduling: State-of-the-art and recent developments. Omega 35 (3), 326–334.
URL http://ideas.repec.org/a/eee/jomega/v35y2007i3p326-334.html

Eele, A., Maciejowski, J., 2015. Sequential Monte Carlo Optimisation for Air Traffic Management. Tech. Rep. CUED/F-INFENG/TR.693, Cambridge University Engineering Department.
URL http://arxiv.org/abs/1506.02869

FR24, 2015. Terms and conditions of use. Retrieved 12/3/2015.
URL http://www.flightradar24.com/terms-and-conditions

Gotteland, J.-B., Durand, N., 2003. Genetic algorithms applied to airport ground traffic optimization. In: Proc. IEEE CEC. IEEE, Canberra, Australia, pp. 544–551.

Gotteland, J.-B., Durand, N., Alliot, J.-M., Page, E., 2001. Aircraft ground traffic optimization. In: Proceedings of the 4th International Air Traffic Management R&D Seminar (ATM2001).

Gupta, R., Srivastava, B., Tamilselvam, S., 2012. Making public transportation schedule information consumable for improved decision making. In: 15th International IEEE Conf. on Intelligent Transportation Systems (ITSC). IEEE, pp. 1862–1867.

Jiang, Y., Liao, Z., Zhang, H., 2013. A collaborative optimization model for ground taxi based on aircraft priority. Mathematical Problems in Engineering 2013, 1–9.

Khadilkar, H., Balakrishnan, H., 2014. Network congestion control of airport surface operations. AIAA Journal of Guidance, Control and Dynamics 37 (3), 933–940.

Koeners, J., Rademaker, R., Aug 2011. Creating a simulation environment to analyze benefits of real-time taxi flow optimization using actual data. AIAA Modeling and Simulation Technologies Conference.

Köln-Bonn Airport, 2013. Annual report 2013. Retrieved 5/2/2015.
URL http://www.koeln-bonn-airport.de/uploads/tx_download/KBA_GeBer_2014_2014_07_11_web_Eng.pdf

Lee, H., Balakrishnan, H., 2012. Fast-time simulations of Detroit Airport operations for evaluating performance in the presence of uncertainties. IEEE/AIAA 31st Digital Avionics Systems Conference (DASC).

Lesire, C., 2010. An iterative A* algorithm for planning of airport ground movements. In: Proceedings of the 2010 conference on ECAI 2010: 19th European Conference on Artificial Intelligence. IOS Press, Amsterdam, The Netherlands, The Netherlands, pp. 413–418.
URL http://dl.acm.org/citation.cfm?id=1860967.1861049

Mayr, M., Navratil, G., 2014. Open street map for multi-modal freight transport planning. In: Proc. REAL CORP. Vienna, Austria, pp. 433–441.
URL http://programm.corp.at/cdrom2014/papers2014/CORP2014_35.pdf

Melbourne Airport, 2014. Melbourne airport achieves 8for 2014.
URL http://melbourneairport.com.au/about-melbourne-airport/media/media-releases/media-release-archive/2015/melbourne-airport-achieves-8-total-passenger-growth-for-2014-1650.html

Pesic, B., Durand, N., Alliot, J.-M., 2001. Aircraft ground traffic optimisation using a genetic algorithm. In: Proc. GECCO 2001.

Petersen, C., Mühleisen, M., Timm-Giel, A., 2013. Evaluation of the aircraft distribution in satellite spotbeams. In: Bauschert, T. (Ed.), Advances in Communication Networking. Vol. 8115 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 46–53.

Pfeil, D. M., Balakrishnan, H., 2012. Identification of robust terminal-area routes in convective weather. Transportation Science 46 (1), 56–73.

Ptak, P., Hartikka, J., Ritola, M., Kauranne, T., Jul 2014. Long-distance multistatic aircraft tracking with VHF frequency doppler effect. IEEE Transactions on Aerospace and Electronic Systems 50 (3), 2242–2252.

Rappaport, D., Yu, P., Griffin, K., Daviau, C., Sep 2009. Quantitative Analysis of Uncertainty in Airport Surface Operations. American Institute of Aeronautics and Astronautics, pp. 1–16.

Ravizza, S., 2013. Enhancing decision support systems for airport ground movement. phd.
URL http://etheses.nottingham.ac.uk/3358/

Ravizza, S., Atkin, J. A. D., Burke, E. K., Oct 2013a. A more realistic approach for airport ground movement optimisation with stand holding. Journal of Scheduling 17 (5), 507–520.

Ravizza, S., Atkin, J. A. D., Maathuis, M. H., Burke, E. K., 2013b. A combined statistical approach and ground movement model for improving taxi time estimations at airports. J. Oper. Res. Soc. 64 (9), 1347–1360.

Ravizza, S., Chen, J., Atkin, J. A., Stewart, P., Burke, E. K., 2014. Aircraft taxi time prediction: Comparisons and insights. Applied Soft Computing 14 (Part C), 397 – 406.
URL http://www.sciencedirect.com/science/article/pii/S1568494613003384

Roling, P. C., Visser, H. G., 2008. Optimal airport surface traffic planning using mixed-integer linear programming. International Journal of Aerospace Engineering 2008, 1–11.

Sheremetyevo Airport, 2014. Press releases 9 October and 14 August on traffic levels since the beginning of the year. Retrieved 10/2/2015.
URL http://www.svo.aero/en/news/2014/

Sheremetyevo Airport, 2015. Press release - Sheremetyevo Airport has provided services to more than 31 million passengers since the beginning of the year.
URL http://www.svo.aero/en/news/2015/3468/

Simaiakis, I., Balakrishnan, H., 2016. A queuing model of the airport departure process. Transportation Science 50 (1), 94–109.

Simić, T. K., Babić, O., 2015. Airport traffic complexity and environment efficiency metrics for evaluation of ATM measures. Journal of Air Transport Management 42 (0), 260 – 271.

Stuttgart Airport, 2013. Retrieved 9/2/2015.
URL http://www.flughafen-stuttgart.de/das-unternehmen/presse/pressemitteilungen/2015/01/rund-9,7-mio-passagiere-in-2014-gewinnerwartung-ueber-vorjahr/

Truong, T. V., 2012. The distribution function of airport taxi-out times and selected applications. Journal of the Transportation Research Forum 50 (2), 33–44.

Turner, R., Bottone, S., Stanek, C., December 2013. Online variational approximations to non-exponential family change point models: With application to radar tracking. In: Proceedings of NIPS 26. The Neural Information Processing Systems Foundation, Lake Tahoe, Nevada, pp. 306–314.
URL http://media.nips.cc/nipsbooks/nipspapers/paper_files/nips26/237.pdf

Weiszer, M., Chen, J., Ravizza, S., Atkin, J., Stewart, P., Jul 2014. A heuristic approach to greener airport ground movement. 2014 IEEE Congress on Evolutionary Computation (CEC), 3280–3286.

Yin, K., Tian, C., Wang, B. X., Quadrifoglio, L., 2012. Analysis of Taxiway Aircraft Traffic at George Bush Intercontinental Airport, Houston, Texas. Transportation Research Record: Journal of the Transportation Research Board 2266 (1), 85–94.

Zilske, M., Neumann, A., Nagel, K., 2011. OpenStreetMap for traffic simulation. In: Proceedings of the 1st European State of the Map–OpenStreetMap conference. No. 11-10. pp. 126–134.

Zurich Airport, 2013. Statistic report 2013. Retrieved 5/2/2015.
URL http://www.zurich-airport.com/~/media/FlughafenZH/Dokumente/Das_Unternehmen/Flughafen_Zuerich_AG/Statistikbericht_2013.pdf

Zurich Airport, 2014. Noise Bulletins for August and September 2014. Retrieved 7/2/2015.
URL http://www.zurich-airport.com/the-company/noise-policy-and-the-environment/noise-monitoring/noise-bulletin