RECONSTRUCTING FLIGHT PLANS FROM TRAJECTORIES FOR TRAFFIC SIMULATIONS

Belana Roman
German Aerospace Center (DLR), Institute of Flight Guidance,
Lilienthalplatz 7, 38108 Braunschweig, Germany

Abstract

In Air Traffic Management (ATM), accurate flight trajectory prediction is crucial for evaluating operational concepts and assessing fuel consumption, environmental impact, and other performance metrics. This prediction typically relies on Flight Management Systems (FMS), which use flight plans as input and provide critical information on fuel burn, aircraft weight and other performance parameters. However, common air traffic data sources such as Automatic Dependent Surveillance-Broadcast (ADS-B) often only provide 4D waypoints and do not include the original filed flight plan. At the same time, the initial flight plan does not account for possible detours or controller interventions that would be needed for a complete analysis. This paper addresses these gaps by proposing an algorithm that reconstructs a suitable flight plan from a given trajectory, providing a complete representation of its path. To achieve this, the algorithm first generates a candidate flight plan based on predefined routes. Then, it compares the given trajectory to this candidate flight plan to identify possible detours, additional turns, or altitude constraints. The resulting flight plan combines these findings and provides a list of lateral waypoints that guide the aircraft from departure to arrival, as well as a cruise flight level estimate, altitude constraints, and additional information on turns. The algorithm was implemented as an additonal feature within the Future Air Traffic Simulator (FATS), an advanced simulation platform for the evaluation and validation of complex air traffic scenarios developed by the Institute of Flight Guidance at the German Aerospace Center (DLR), and validated using both synthetic and real-world data. The synthetic dataset consists of 862 short- and long-haul flights with introduced lateral and vertical deviations, while the real-world dataset includes a smaller number of ADS-B trajectories from the OpenSky Network. Using the trajectories as input, the algorithm successfully reconstructed suitable flight plans representing the entire flight path, including complex holding procedures. Accuracy is evaluated by comparing the original trajectory and reconstructed flight plan on a discretized grid. The results show a high level of agreement in the lateral dimension, whereas alignment in three dimensions is lower. This is mainly due to the current focus on level segments. Modelling the complete climb and descent procedures with altitude constraints will be incorporated in future work.

Keywords

4D Trajectories; Flight plans; ADS-B; Air traffic simulation; Bresenham algorithm

ABBREVIATIONS

ADS-B	Automatic Dependent Surveillance - Broadcast
ATC	Air Traffic Control
FATS	Future Air Traffic Simulator
FMS	Flight Management System
MOT	Mid of Turn
SOT	Start of Turn

1. INTRODUCTION

The Future Air Traffic Simulator (FATS) is a tool for simulating and analysing complex air traffic scenarios. By combining air traffic and meteorological data it creates a comprehensive simulation environment [1]. For analyses of fuel consumption or climate impact, detailed trajectory information is required. While fuel consumption can also be directly estimated from

doi: 10.25967/650175

existing trajectories, the Flight Management System (FMS) uses flight plans as input to generate consistent trajectories and provide detailed information on fuel consumption, aircraft weight, and other performance parameters along the flight. The flight plan also provides detailed flight path information, such as turn segments, cruise flight levels, and waypoints, which can be easily retrieved for each trajectory. While common air traffic data sources, including Automatic Dependent Surveillance - Broadcast (ADS-B), provide four-dimensional waypoints, the original filed flight plan is usually not included [2]. At the same time, the originally submitted flight plan does not take into account potential detours due to traffic or weather conditions, or trajectory changes made during the simulation. Such changes may be required to resolve conflicts or research new routing approaches within the simulation. Therefore, reconstructing a suitable flight plan from a given trajectory is essential to enable accurate FMS-based simulations and analyses.

Previous research on reconstructing flight plans from trajectories is limited. Lee et al. [3] proposed an algorithm to extract a possible flight plan from recorded ADS-B data. In this approach, multiple candidate routes are generated using information from the Aeronautical Information Publication (AIP), which contains common predefined routes and procedure data. These candidate routes are then scored based on their closeness to the original trajectory and the route with the highest score is selected. The focus of [3] is on identifying the originally filed flight plan for HITL simulations, without considering interventions by air traffic controllers due to traffic, restricted airspace, or weather conditions. The reconstruction algorithm presented here follows a similar approach. It uses predefined routes to generate candidate routes, but also evaluates the given trajectory directly. This allows to detect additional turns, level segments, and changes in cruise altitude. Where necessary, synthetic waypoints and connections can be created to capture the actual flight path more accurately. In this way, the reconstructed flight plan accurately reflects the entire trajectory, including any potential deviations caused by operational constraints or changes made during the simulation.

Following the introduction, Section 2 briefly outlines the information contained in a flight plan and its representation in FATS. Section 3 then provides a detailed description of the proposed flight plan reconstruction algorithm. The validation methodology and results are presented in section 4. Section 5 concludes with a discussion of the algorithms limitations and potential future improvements.

2. BACKGROUND: FLIGHT PLANS

For most flights, operators are required to submit a flight plan to the appropriate Air Traffic Control (ATC) unit prior to departure. A flight plan typically contains flight-specific information (e.g., aircraft identification and equipment), aircraft-specific information (e.g., type and wake turbulence category), and routing information that specifies the intended path of the flight [4,5]. The routing component consists of a lateral path, describing the geographical route, and a vertical path, defining altitude and speed constraints along that route. The vertical profile is influenced by the aircraft's performance, configuration, and location-specific restrictions [6]. While flight- and aircraft-specific information can usually be obtained directly from air traffic data, the intended route must be reconstructed from the observed trajectory. Therefore, the following sections focus on routing information rather than on complete flight plans.

Within FATS, routing information is represented by a structure called a Constraint List. This list specifies the departure and arrival airports and runways, as well as a sequence of spatial points defining the lateral path. These points may correspond to named waypoints or navigation aids from a navigation database, or to synthetic waypoints used only within the simulation. Each point is defined by a name and geographic coordinates and may include constraints related to altitude, time, or speed. These constraints can be specified either at the point itself or between two consecutive points. The segments connecting consecutive points are referred to as legs and are approximated as great-circle segments. Further details on filling the Constraint List are given in Section 3.

3. METHODOLOGY

To reconstruct a suitable flight plan from a given trajectory, the algorithm first identifies suitable departure and arrival runways, along with their corresponding airports. Based on this information, a preliminary flight plan is calculated using the navigation database within FATS. This flight plan is derived from predefined waypoints and routes and only acts as a reference, since it may not fully align with the original trajectory. In order to quantify the similarity between the proposed flight plan and the observed trajectory, both are rasterized using the Bresenham algorithm [7, 8] and then compared. This way, it is possible to identify all segments where the trajectory and the proposed flight plan coincide and all the segments where they differ. Based on this comparison, a refined flight plan is constructed: matching segments are retained from the proposed flight plan, while differing segments are updated with new waypoints and altitude constraints. Despite these adjustments, some discrepancies may remain as the final flight plan may not fully capture the complexity of the original trajectory. The overall workflow is summarized in Figure 1, while the following sections describe the individual steps of the algorithm in detail.

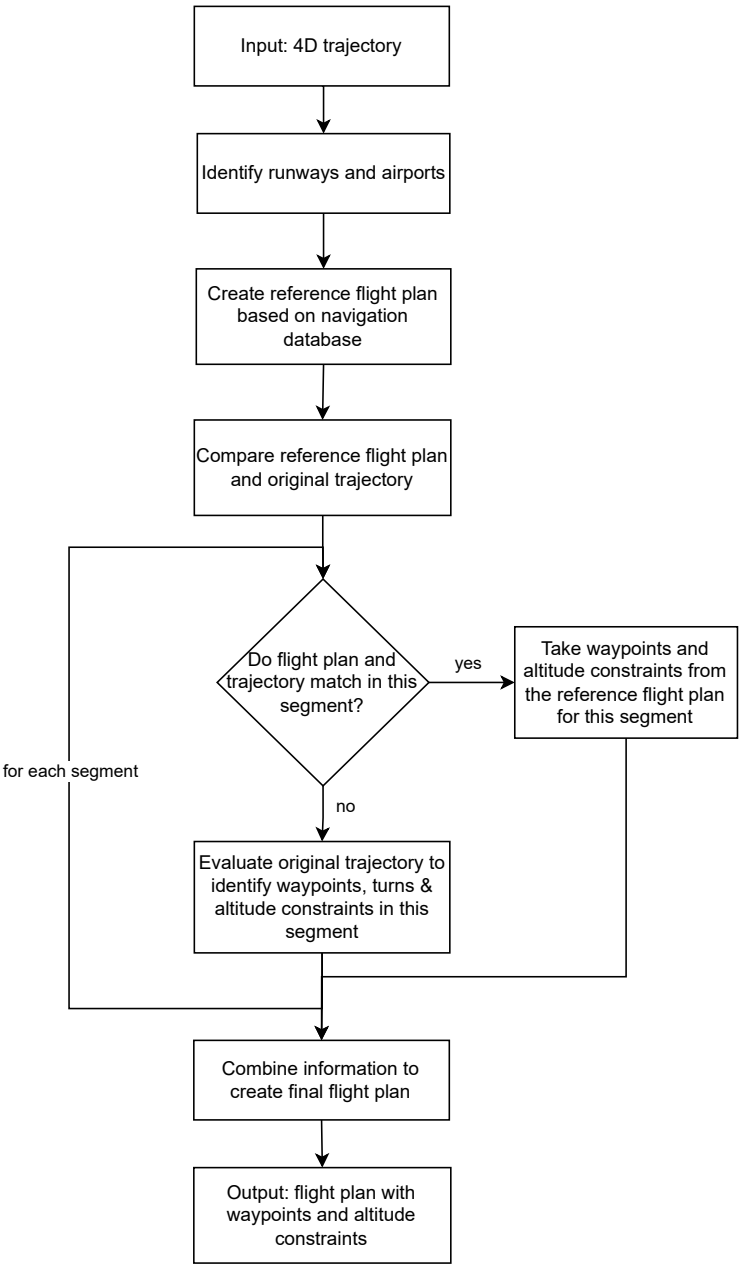

FIG 1. Overview of the flight plan reconstruction. The algorithm generates a preliminary flight plan, compares it with the observed trajectory using Bresenham rasterization, and refines the plan by updating deviating segments.

3.1. Identification of airports and runways

To determine the most suitable airports and runways, complete trajectories that include both departure and arrival procedures are considered. The first and last trajectory points are assumed to be closest to the departure and arrival runways and are therefore used for the search. Based on the positions of these points, all runways within a radius of three nautical miles are retrieved from the FATS navigation database. This search radius provides sufficient tolerance to account for variability in the recorded trajectory data, particularly in ADS-B datasets. Depending on the data, ground movements may or may not be included, and records can begin after takeoff or end shortly before landing [9]. Allowing for such variations ensures a sufficiently flexible search for candidate runways. From the resulting list, the most likely runway is selected by evaluating two factors: the lateral distance between the trajectory point and the runway centreline, and the angular difference between the aircraft's current heading and the runway bearing. These discrepancies are combined into a single error metric, and the runway with the lowest value is chosen. Figure 2 illustrates the components of this error metric, showing both the lateral distance to the runway centreline and the angular difference between heading and bearing. Once the runways are determined, the corresponding airports are retrieved from the navigation database.

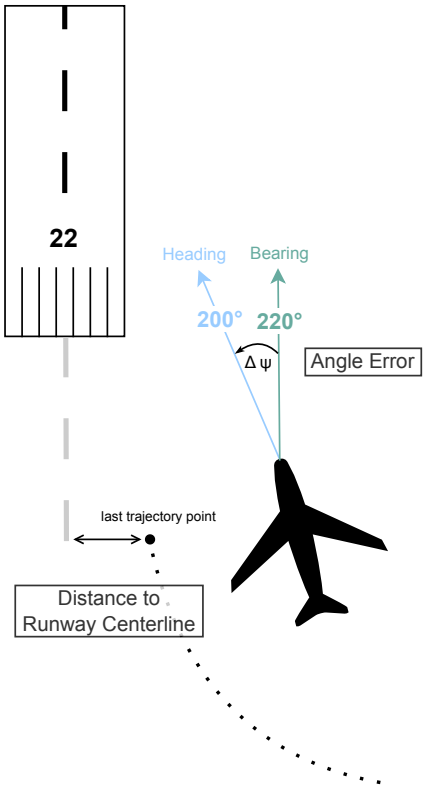

FIG 2. Schematic representation of distance and angle error for runway identification

3.2. Preliminary flight plan generation from navigation data

Once the departure and arrival runways and airports have been identified, the algorithm uses the navigation database of FATS to work out a feasible flight plan for this configuration. The proposed flight plan is based on predefined waypoints and routes stored in the database, following standard departure (SIDs), en-route, and arrival (STARs and approach) procedures. However, it may not fully align with the original trajectory. Therefore, it serves only as a reference to identify segments where the trajectory matches the suggested flight plan and segments with deviations. If no viable reference flight plan can be generated, the final flight plan is constructed only from the characteristics of the observed trajectory. In this case, the next step — measuring the similarity — is skipped, and the algorithm proceeds directly to the identification of waypoints.

Internally the proposed flight plan is stored in the previously introduced Constraint List data structure. The paths connecting the points of the Constraint List are referred to as legs and are represented by approximated great-circle segments. Turns are always assumed to follow a fixed-radius arc and can be represented by either fly-over or fly-by waypoints. A fly-over waypoint lies directly on the aircraft's path and marks the exact location where the turn is initiated. In FATS, this point is referred to as the Start of Turn (SOT). By contrast, a fly-by waypoint is not located directly on the flight path. Instead, the aircraft begins its turn before reaching the waypoint. In this case, the waypoint lies at the intersection of the inbound leg, i.e. the segment leading towards the waypoint and the outbound leg, i.e. the segment leading away from it [10]. In FATS, this point is referred to as the Mid of Turn (MOT). The turn kind (SOT or MOT) and its turn radius is included at each point of the Constraint List as additional turn information. It should be noted that all waypoints in the Constraint List are labelled with a turn kind and radius, even if there is no visible turn, because the waypoints lie on approximated great-circle segments and small deviations leading to non-zero turn values are to be expected. In these cases, the waypoint is also labelled as MOT and assigned a default turn radius of 3 nautical miles during departure and approach phases, and 5 nautical miles in the en-route phase. This convention provides consistency in the data structure and makes it easier to process the data in a standardized way. Each waypoint may contain altitude constraints, which are represented as an altitude interval with an upper and lower bound, as illustrated in figure 3. A flag specifies whether the constraint is defined locally at the waypoint itself or between two consecutive waypoints. Although time and speed constraints are part of the data structure, they are not yet handled by the current flight plan reconstruction algorithm.

3.3. Measuring the similarity of trajectories using the Bresenham Algorithm

Both the original trajectory and the trajectory resulting from the proposed flight plan are rasterized onto a common grid using the Bresenham line algorithm.

No_Waypoints : 21

Waypoints

Nr,	Name ,	Lat ,	Long ,	Turn,	Radius,	AltC,	Upper ,	Lower
1	RW27R	52.4682428	9.6526914	mot	3.00	no	60000	0
2	RW27R@4	52.4690931	9.5970220	sot	3.00	local	60000	600
3	DV204	52.4708109	9.5575887	mot	3.00	local	60000	2600
4	DV206	52.4027276	9.2300948	mot	3.00	no	60000	0
5	VAXEV	52.3956959	9.0880277	mot	3.00	no	60000	0
6	OSN	52.2001340	8.2855198	mot	5.00	no	60000	0
7	ABAMI	51.4249997	7.2805556	mot	5.00	no	60000	0
8	NEREL	51.0875030	6.5630553	mot	5.00	no	60000	0
9	AGENI	50.7499994	6.0333332	mot	5.00	no	60000	0
10	LNO	50.5859144	5.7102751	mot	5.00	no	60000	0
11	DINAN	49.8319434	5.3313891	mot	3.00	no	60000	0
12	FF101	49.6711130	4.4583334	mot	3.00	no	60000	0
13	XERAM	49.5966673	4.0672223	mot	3.00	no	60000	0
14	ENORI	49.4746938	3.7779441	mot	3.00	no	60000	0
15	DEVIM	49.4501632	3.6322220	mot	3.00	local	16000	0
16	LORNI	49.4194171	3.4513889	mot	3.00	local	15000	11000
17	PG520	49.2745010	3.3259724	mot	3.00	no	60000	0
18	PG520@3	49.2309722	3.2885241	mot	3.00	no	60000	0
19	PNW17	49.0485856	2.9842501	mot	3.00	local	60000	5000
20	PNW15	49.0454984	2.9230249	mot	3.00	no	60000	0
21	RW27R	49.0266949	2.5616889	mot	3.00	no	60000	0

FIG 3. Excerpt from a proposed flight plan for a flight from Hanover (EDDV) to Paris (LFPG)

The Bresenham algorithm is a computationally efficient method for approximating straight lines on discrete grids by determining which pixels to highlight between two points [7]. This enables a pixel-level comparison of the flight paths.
To define when two trajectories are sufficiently close or overlapping, maximum separation thresholds are introduced: 100 m in the lateral dimension and 50 m in the vertical dimension. These thresholds determine the spatial resolution of the grid cells. If two trajectories occupy the same grid cell, they are classified as identical in that segment. By plotting both trajectories on the grid, the algorithm can identify all pixels that are occupied by both trajectories. This shows segments where the trajectories overlap. For this purpose, the Bresenham algorithm has been implemented in both 2D and 3D within FATS [8]. Based on this comparison, a refined flight plan is constructed. Matching segments are retained directly from the proposed flight plan, while differing segments are updated with new waypoints and altitude constraints.

3.4. Trajectory-Based Waypoint Extraction

In order to identify all necessary waypoints, the 2D Bresenham algorithm is applied as explained in the previous section. Deviating segments are simplified using a 2D Douglas-Peucker algorithm, to reduce the number of trajectory points while preserving the essential geometry [11]. This results in a trajectory with an increased point density at turns and a decreased density along straight segments, as shown in figure 4. Along straight segments, suitable waypoints from the navigation database are simply assigned to the remaining trajectory points after the simplification. If no predefined waypoint is found within a 20-metre radius, a synthetic waypoint is generated and added to the flight plan. Turn segments, however, are handled differently by the algorithm. This is described in more detail in the following section.

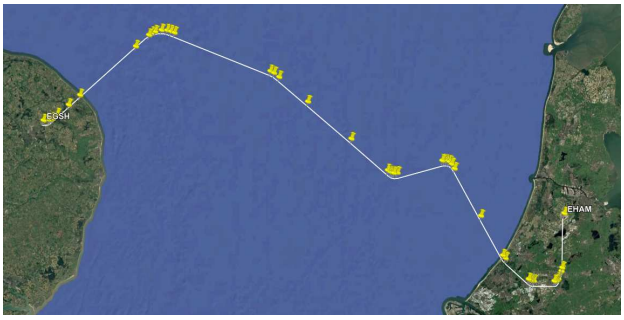

FIG 4. Visualization of 2D trajectory simplification. High point density is preserved at turns, low density on straight segments.

3.4.1. Identification of turn points

Possible turn segments are identified by their high point density after the previously mentioned 2D Douglas-Peucker simplification. Within turns the distance between consecutive trajectory points is low, while in straight segments they are more sparsely distributed and therefore further apart. However, the turn is also verified by analysing changes in the heading of the aircraft. For each valid turn segment, the Mid of Turn point must be determined. As described in section 2, the Mid of Turn lies at the intersection of the inbound and outbound legs of the turn. Here, the inbound leg is defined as the great-circle segment that connects the last trajectory point before the turn to the first point within the turn. The outbound leg, on the other hand, is defined as the great-circle segment connecting the final point of the turn to the next point beyond the turn. By extending both the inbound and outbound legs, they intersect at the Mid of Turn point. Figure 5 illustrates this principle. The algorithm described in this paper typically attempts to identify a suitable Mid of Turn point rather than a Start of Turn, as this achieved better accuracy in modelling the turns. Based on this principle, the algorithm attempts to find a suitable waypoint within approximately 20 metres of the intersection point using the navigation database.

If a waypoint is found within this range, it is added to the flight plan. If no waypoint from the database is located at this position, a synthetic waypoint is created and included in the flight plan.

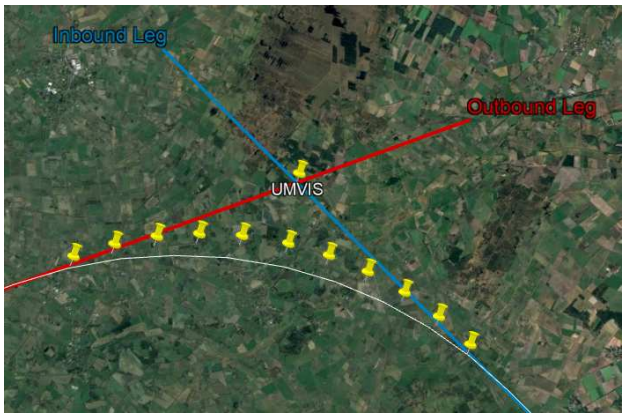

FIG 5. Intersection of Inbound and Outbound leg to find Mid of Turn waypoint

In addition to the waypoint that defines the location of the turn, the turn radius is essential for accurately modelling the turn. The turn radius is calculated using the inbound and outbound legs, which can be considered tangents of the circle representing the turn.
To calculate the radius, the centre of the circle is identified first. This is achieved by constructing perpendicular lines from the point where the tangents touch the circle, since tangents to a circle are always perpendicular to the radius at the point of contact. The point where these two perpendicular lines intersect defines the centre of the circle. Once the centre is located, the turn radius can be calculated and added to the flight plan.
Combining these steps, the algorithm constructs an adapted lateral flight plan, including the correct waypoints and turn information. Figure 6 shows the flight plan constructed in this manner. The red frames indicate where changes were made in comparison to the initially proposed flight plan, shown in figure 3. Waypoints starting with an '@' are synthetic waypoints that have been added because no existing waypoints were found in the navigation database at the respective coordinates.

3.5. Identification of altitude constraints

After all relevant waypoints have been identified and the lateral flight plan has been adapted accordingly, the similarity between the adapted plan and the original trajectory is evaluated once

No_Waypoints : 20

Waypoints

Nr,	Name ,	Lat ,	Long ,	Turn,	Radius,	AltC,	Upper ,	Lower
1	RW27R	52.4682428	9.6526914	mot	3.00	no	60000	0
2	RW27R@4	52.4695269	9.5970484	sot	3.00	local	60000	600
3	DV204	52.4708109	9.5575887	mot	3.00	local	60000	2600
4	UMVIS	52.5588862	8.6769451	mot	5.00	no	60000	0
5	OSN	52.2001340	8.2855198	mot	5.00	no	60000	0
6	@0	51.5515870	7.4415867	mot	5.00	no	60000	0
7	NEREL	51.0875030	6.5630553	mot	5.00	no	60000	0
8	@1	50.6167424	5.7709246	mot	5.00	no	60000	0
9	@2	50.2686009	5.2684698	mot	5.00	no	60000	0
10	@3	50.1178108	5.0623480	mot	5.00	no	60000	0
11	@4	49.9765898	4.8693089	mot	5.00	no	60000	0
12	FF101	49.6711130	4.4583334	mot	3.00	no	60000	0
13	XERAM	49.5966673	4.0672223	mot	3.00	no	60000	0
14	ENORI	49.4746938	3.7779441	mot	3.00	no	60000	0
15	DEVIM	49.4501632	3.6322220	mot	3.00	local	16000	0
16	LORNI	49.4194171	3.4513889	mot	3.00	local	15000	11000
17	PG520	49.2745010	3.3259724	mot	3.00	no	60000	0
18	PG520@3	49.2308014	3.2889890	mot	3.00	no	60000	0
19	PNW05	49.0331460	2.6831363	mot	3.00	local	60000	1980
20	RW27R	49.0266949	2.5616917	mot	3.00	no	60000	0

FIG 6. Constructed flight plan assumed to be correct laterally

again, now using the 3D Bresenham algorithm. This step highlights all segments where the vertical profile still deviates from the original trajectory and where altitude constraints must be introduced. The cruise flight level is estimated based on the maximum altitude reached by the trajectory. It is assumed that flights are planned efficiently and therefore reach their cruise level as early as possible. As a result, climb and descent phases are not considered in the algorithm at this stage. Instead, the algorithm focuses on identifying level segments and adding them to the flight plan as altitude constraints.
To detect level segments, the original trajectory is used in full resolution, as the simplified version would not provide enough detail to capture the vertical profile accurately. Once a level segment is identified, it must be mapped to the nearest waypoints in the flight plan. If the nearest waypoints are within 100 metres of the estimated position, they can be considered close enough, and the level segment can be defined between them. For turns, the allowed distance to the estimated position is extended to the turn radius. However, if the waypoints are further away from the estimated start and end points of the level segment, synthetic waypoints must be added to ensure the altitude constraints are set at the correct locations. After the relevant waypoints have been determined, the corresponding information is added to the flight plan. Level constraints are defined between two consecutive waypoints by setting the altitude constraint type in the Constraint List to level. The altitude itself is represented as an interval: the upper bound corresponds to the identified level altitude, while the lower bound is set 100 m below.

This tolerance accounts for minor numerical deviations and ensures that the constraint is stored as a feasible altitude band rather than a single exact value.

4. VALIDATION

The proposed flight plan reconstruction algorithm has been validated by comparing the reconstructed flight plans to the original trajectories used as input. The objective is to assess how accurately the reconstructed flight plan represents both the lateral path and the vertical profile of the reference trajectory.

4.1. Data

The validation is based on a set of 862 flights departing from or arriving in Europe, including both short- and long-haul operations.
These flights were originally generated using the navigation database of FATS, but were later deconflicted both laterally and vertically. This process introduced deviations from the usual routes, as well as additional level segments, making the dataset a suitable test case for the algorithm. As this synthetic dataset is free of measurement noise and the trajectories are available at a high temporal and spatial resolution, it is useful for testing and validating the overall concept.
In addition to this synthetic dataset, the algorithm was also tested on a smaller set of ADS-B data recorded and downloaded from the OpenSky Network using the Python library Traffic [12, 13]. In contrast to the FATS data, the ADS-B dataset contains measurement noise and outliers, especially in the vertical profile. This makes it a valuable basis for evaluating the robustness of the reconstruction under real-world conditions. To minimize the most severe effects, a smoothing filter was applied using the filter() method provided by the Traffic library [12]. The filter was applied several times with different kernel values, specifically using altitude = 17 followed by altitude = 53, to remove noise in the altitude measurements.
Figure 7 shows an example of an unfiltered vertical profile from the ADS-B dataset, along with the corresponding filtered profile.

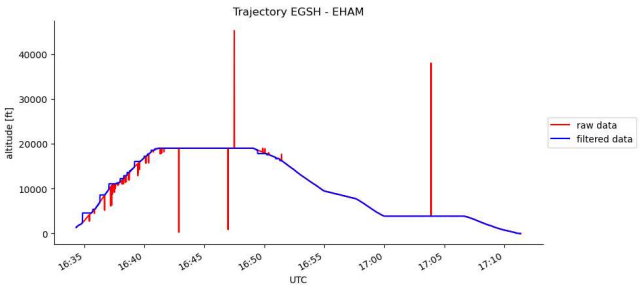

FIG 7. Vertical profile of an example ADS-B trajectory before and after filtering.

4.2. Validation approach

In order to validate how well the final flight plan fits the original trajectory, a new trajectory is once again calculated from the final flight plan and compared to the original one using the Bresenham algorithm, which checks for overlaps between both trajectories on a discretized grid. Different maximum separation thresholds were tested for both lateral and vertical dimensions.

4.3. Results

Table 1 shows the average matching percentage in 2D and 3D between the original trajectory and the final flight plan for different separation thresholds for the synthetic dataset containing 862 flights. The results indicate that the average match rate between the trajectories is high, even when considering the smallest matching distance of 100 m in the lateral dimension and 50 m in the vertical layer. As expected, the 3D matching rate is consistently lower than the 2D rate, showing current limitations in modelling the vertical profile. The cruise altitude and level segments are typically successfully identified and reconstructed, however, there are still discrepancies in the climb and descent phases, since these phases are not yet specifically modelled in this algorithm.

TAB 1. Accuracy of the reconstructed flight plan on synthetic data (862 flights) using different separation thresholds

Separation Maxima	2D (%)	3D (%)
100 m lateral, 50 m vertical	97.06	70.27
100 m lateral, 100 m vertical	97.06	76.06
200 m lateral, 150 m vertical	97.93	80.82

For the small ADS-B dataset, the matching percentages are shown in table 2. Here, an overall lower score can be achieved, however, this dataset included more complex trajectories. For instance, the ADS-B dataset contained trajectories from a day with severe thunderstorms around Milan Malpensa Airport (LIMC). Those trajectories included holding patterns to delay landings. As shown in Figure 8, the trajectory based on the reconstructed flight plan successfully captures the behaviour laterally with only smaller deviations of a few hundred metres: the red trajectory corresponds to the original data, the blue trajectory to the reconstructed trajectory. The cruise flight level is estimated correctly, however, the descent procedure is not yet modelled by the algorithm, resulting in large deviations here.

TAB 2. Accuracy of the reconstructed flight plan on real-world ADS-B data using different separation thresholds

Separation Maxima	2D (%)	3D (%)
100 m lateral, 50 m vertical	72.72	30.06
100 m lateral, 100 m vertical	72.72	32.01
200 m lateral, 150 m vertical	78.53	35.05
300 m lateral, 300 m vertical	82.93	42.48

5. CONCLUSION & DISCUSSION

In this paper, an algorithm for reconstructing flight plans from recorded trajectories was presented and validated using both synthetic flights and real-world ADS-B data. The algorithm adapts the lateral and vertical components of a candidate flight plan to match the given trajectory. By combining predefined routes and waypoints with trajectory characteristics to identify turns and altitude constraints, a flight plan can be created that accurately reflects the entire trajectory, including deviations caused by operational constraints or simulation interventions. In this way, a matching flight plan can now be reconstructed for any recorded trajectory. This provides accurate flight plans as essential input for reliable FMS predictions, enabling a wide range of further analyses, such as fuel consumption assessments.

The validation results show that the method achieves a high level of accuracy, particularly in reproducing the en-route phase of the original trajectories.

However, there is still room for improvement. In particular, altitude constraints need to be handled more effectively by modelling climb and descent procedures explicitly. Furthermore, integrating speed and time constraints would enable a more comprehensive reconstruction that better reflects actual flight behaviour. Future work will therefore focus on extending the algorithm accordingly.

FIG 8. Example of a more complex trajectory with holding patterns due to thunderstorms near Milan. The original ADS-B trajectory is shown in red, the reconstructed trajectory in blue.

Contact address:

belana.roman@dlr.de

References

[1] Alexander Kuenz. FATS (Future Air Traffic Simulator), 2025.

[2] Automatic Dependent Surveillance - Broadcast (ADS-B). https://skybrary.aero/articles/automatic-dependent-surveillance-broadcast-ads-b. [Accessed 02-03-2025].

[3] Hyeonwoong Lee and Hak-Tae Lee. Extracting flight plans from recorded ads-b trajectories. *International Journal of Aeronautical and Space Sciences*, 24(2):581–589, Apr 2023. DOI: 10.1007/s42405-022-00539-3.

[4] EUROCONTROL. Flight plan filing and management. https://www.eurocontrol.int/service/flight-plan-filing-and-management. [Accessed 17-04-2025].

[5] Federal Aviation Administration (FAA). Appendix A. FAA Form 7233-4, International Flight Plan. https://www.faa.gov/air_traffic/publications/atpubs/fs_html/appendix_a.html. [Accessed 17-04-2025].

[6] B. D. Dancila and R. M. Botez. Vertical flight path segments sets for aircraft flight plan prediction and optimisation. *The Aeronautical Journal*, 122(1255):1371–1424, 2018. DOI: 10.1017/aer.2018.67.

[7] Mohammed Khalid Kaleem, Dhanraj Verma, and Mohammad Javed Idrisi. Generalization of line drawing algorithm – an effective approach to minimize the error in the existing bresenham's line drawing algorithm. In *2021 International Conference on Emerging Smart Computing and Informatics (ESCI)*, pages 516–521, 2021. DOI: 10.1109/ESCI50559.2021.9396940.

[8] X-W Liu and K Cheng. Three-dimensional extension of bresenham's algorithm and its application in straight-line interpolation. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 216(3):459–463, 2002. DOI: 10.1243/0954405021519979.

[9] Manuel Waltert and Benoit Figuet. Using ads-b trajectories to measure how rapid exit taxiways affect airport capacity. *Journal of Open Aviation Science*, 1(2), 2023.

[10] International Civil Aviation Organization (ICAO). *Procedures for Air Navigation Services (PANS) - Aircraft Operations - Volume I Flight Procedures (Doc 8168)*. 2018.

[11] Saeed Mehri, Navid Hooshangi, and Navid Mahdizadeh Gharakhanlou. A novel context-aware douglas–peucker (cadp) trajectory compression method. *ISPRS International Journal of Geo-Information*, 14(2), 2025. DOI: 10.3390/ijgi14020058.

[12] Xavier Olive. traffic, a toolbox for processing and analysing air traffic data. *Journal of Open Source Software*, 4:1518, 2019. DOI: 10.21105/joss.01518.

[13] Matthias Schäfer, Martin Strohmeier, Vincent Lenders, Ivan Martinovic, and Matthias Wilhelm. Bringing up opensky: A large-scale ads-b sensor network for research. In *IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*, pages 83–94, 2014. DOI: 10.1109/IPSN.2014.6846743.