A Data-Driven Framework for Long-Range Aircraft Conflict Detection and Resolution

SAMET AYHAN*, University of Maryland
PABLO COSTAS, Boeing Research & Technology Europe
HANAN SAMET, University of Maryland

At the present time, there is no mechanism for Air Navigation Service Providers (ANSPs) to probe new flight plans filed by the Airlines Operation Centers (AOCs) against the existing approved flight plans to see if they are likely to cause conflicts or bring sector traffic densities beyond control. In the current Air Traffic Control (ATC) operations, aircraft conflicts and sector traffic densities are resolved tactically, increasing workload and leading to potential safety risks and loss of capacity and efficiency.

We propose a novel Data-Driven Framework to address a long-range aircraft conflict detection and resolution (CDR) problem. Given a set of predicted trajectories, the framework declares a conflict when a protected zone of an aircraft on its trajectory is infringed upon by another aircraft. The framework resolves the conflict by prescribing an alternative solution that is optimized by perturbing at least one of the trajectories involved in the conflict. To achieve this, the framework learns from descriptive patterns of historical trajectories and pertinent weather observations and builds a Hidden Markov Model (HMM). Using a variant of the Viterbi algorithm, the framework avoids the airspace volume in which the conflict is detected and generates a new optimal trajectory that is conflict-free. The key concept upon which the framework is built is the assumption that the airspace is nothing more than a horizontally and vertically concatenated set of spatio-temporal data cubes where each cube is considered as an atomic unit. We evaluate our framework using real trajectory datasets with pertinent weather observations from two continents and demonstrate its effectiveness for strategic CDR.

CCS Concepts: • **Information systems → Data analytics**; • **Computing methodologies → Dynamic programming for Markov decision processes**; • **Applied computing → Aerospace**;

Additional Key Words and Phrases: Aircraft Conflict Detection and Resolution; Air Traffic Management; Prescriptive Analytics; Hidden Markov Model; Time Series

ACM Reference Format:
Samet Ayhan, Pablo Costas, and Hanan Samet. 2019. A Data-Driven Framework for Long-Range Aircraft Conflict Detection and Resolution. *ACM Trans. Spatial Algorithms Syst.* 5, 4, Article 24 (September 2019), 23 pages. https://doi.org/10.1145/3328832

1 INTRODUCTION

In the present Air Traffic Management (ATM) system, Airline Operations Center (AOC) personnel file a flight plan for a particular flight. The filed flight plan usually contains 2D coordinates of the fixed way points and the planned initial cruise speed and cruise altitude in addition to their speed and level changes along the route. It does not include the time information for the way points and the existing information is not routinely updated

*Senior Engineer at Boeing Research & Technology.

Authors' addresses: Samet Ayhan, University of Maryland, Department of Computer Science, College Park, Maryland, 20742, sayhan@cs.umd.edu; Pablo Costas, Boeing Research & Technology Europe, Av. Sur del Aeropuerto de Barajas, 38, Madrid, Spain, 28042, pablo.costas@boeing.com; Hanan Samet, University of Maryland, Department of Computer Science, College Park, Maryland, 20742, hjs@cs.umd.edu.

2374-0353/2019/9-ART24 $15.00
https://doi.org/10.1145/3328832

by the ATM systems. Moreover, the filed flight plan is not checked against other flight plans to probe potential interference with other aircraft or identify sector traffic complexity before the aircraft departs. Airspace sector complexity and aircraft separation assurance are dealt with tactically, usually just a few minutes before the aircraft enters the sector or is likely to have a conflict with other aircraft.

It is estimated that by 2040 the USA alone can expect an increase of more than 68% in the commercial air traffic [22]. Hence, a new concept of operations is needed to accommodate this increase in volume. To meet this challenge ahead of us, new technologies and procedures for next generation ATM are being developed in the context of the Next Generation Air Transportation System (NextGen) [20] in the USA and the Single European Sky ATM Research (SESAR) [55] in Europe. The SESAR and NextGen concept of operations requires a paradigm shift from a highly structured and fragmented system that is heavily reliant on tactical decision making and with few strategic planning functions based on uncertain information, to an integrated one based on collaborative strategic management of trajectories and information sharing [59]. In the future ATM systems to be built under NextGen and SESAR, the trajectory becomes the fundamental element of new sets of operating procedures collectively referred to as Trajectory-Based Operations (TBO). The underlying idea behind TBO is the concept of a business trajectory that the airline agrees to fly and the ANSP and airports agree to facilitate, given the safe separation provision. A business trajectory is the representation of an airline's intention with respect to a given flight that will best meet the airline's business interests. The TBO concept of operations and the notion of a business trajectory will result in increased overall predictability of air traffic, reduced costs and emissions due to lowered fuel consumption and/or time, and increased capacities. Thus, the future ATM system should offer flexibility to accommodate business trajectories, while bearing the primary goal of safety in mind. Overall, the next generation ATM paradigm shift requires more safe and efficient CDR due to fact that maintaining the separation minima among the vast volume of aircraft that fly their versions of most optimal business 4D trajectories becomes more challenging.

Hence, we introduce a Data-Driven Framework to address the long-range aircraft CDR problem. The framework performs CDR in two steps: In the first step, given a set of predicted trajectories in the form of a 4D joint spatio-temporal data cubes on a 3D grid network, the framework declares a conflict if one of the aircraft's predicted trajectory segment overlaps with the other's protected zone at the same time interval in the future. Obviously, the more accurate the predicted trajectories, the more accurate the detection of the conflict as the separation minima will be on the containment volume of each predicted trajectory. In the second step, the framework builds a stochastic model, HMM that learns from the historical trajectories and their correlation with the pertinent weather parameters. Using a variant of the Viterbi algorithm, the framework prescribes an optimal solution by perturbing at least one of the 4D trajectories involved in the conflict.

In summary, the contributions of this paper are as follows:

- We propose a cube-shaped protected zone surrounding each aircraft that can be expanded by joining the neighboring cubes horizontally and vertically, yielding a protected zone with a desired size. This idea resonates with the assumption that an airspace is nothing more than a set of concatenated spatio-temporal data cubes around a 3D grid network, where each cube is considered as an atomic unit.
- We propose a scalable Data-Driven Framework to strategically address the aircraft CDR problem. Given a set of predicted trajectories, the framework declares a conflict when a protected zone of an aircraft on its trajectory is infringed upon by another aircraft at the same time interval in the future. Upon a conflict, the framework executes our conflict resolution algorithm and prescribes a solution that resolves the conflict by perturbing at least one of the 4D trajectories involved in the conflict.
- We conduct extensive experiments based on real trajectory and weather data from two continents and demonstrate that our framework can detect and resolve conflicts with lateral and vertical accuracies that are within the boundaries of conventionally accepted minimum separation values, set by the ANSPs. This

translates to the fact that, ANSPs can now detect and resolve potential conflicts before the aircraft depart, resulting in safer and greener airspace with more efficiency and capacity, and thereby reducing the air traffic controller workload.

In our previous work [9], we presented our Aircraft Trajectory Prediction System that shares a common ground with the current framework in a way that both the previous system and the current framework attempt to find the most likely sequence of aircraft positions given a set of weather observations. However, they differ in the following respects: *i)* The previous system [9] aimed at predicting aircraft trajectories without considering any potential obstacles along the route. The proposed framework detects and resolves conflicts among the aircraft. In fact, the framework we propose in this work uses predicted trajectories generated by the previous system [9] as input. *ii)* The previous system [9] utilized historical trajectories and pertinent weather observations to build an HMM which was fed into the Viterbi algorithm. Although our current framework uses the same input for an HMM during the conflict resolution phase, it employs a variant of the Viterbi algorithm. Unlike the regular Viterbi algorithm, the variant generates optimal trajectories bypassing the spatio-temporal data cubes in which the conflict is detected. *iii)* Unlike the previous system [9], the current framework is scalable. It can detect and resolve conflicts among multiple aircraft. Moreover, this paper is an extended version of work published in [6]. We extend our previous work [6] by providing the following additional content:

- We elaborate more on the ideas of TBO and business trajectory in Section 1.
- We provide a more detailed explanation as to where our framework falls in comparison to the other systems in Section 2.
- We expand Section 4.3 with an Octree data structure figure and we provide the pertinent explanation.
- We provide time complexities for Algorithm 2 and Algorithm 3, where we present CDR for multiple aircraft in Section 4.3.
- We provide 8 additional test cases in Section 5, where we evaluate CDR for multiple aircraft.
- We provide Table 3 for the additional test cases in Section 5.1.
- We provide Figure 6 with a set of boxplots showing trajectory prediction errors that can be attributed for the CDR accuracy in Section 5.2.
- We update Figure 5 and Figure 7 based on additional test cases in Section 5.2.

Our Data-Driven Framework can be used as a ground-based strategic CDR system by air traffic flow managers to resolve potential interference among large volume of aircraft and identify high density and complex sector traffic before the aircraft depart. The set of optimized resolutions should improve ATM automation and reduce the workload of air traffic controllers. The rest of the paper is organized as follows. Section 2 reviews related work. Section 3 introduces preliminary concepts followed by Section 4 where our Data-Driven Framework is described. Section 5 presents the results of our experiments, while Section 6 discusses the results. Concluding remarks are drawn in Section 7.

2 RELATED WORK

Our study involves trajectory data which have been the subject of much work in the spatial domain with an emphasis on cars along roads [53]. The focus has been on their generation (e.g., [54]), queries (e.g., [36, 38–40, 49, 51, 52]), and matching (e.g., [26, 35, 50]). This data is collected continuously and is quite voluminous. Instead, our focus here is on the flight domain. The problem of aircraft CDR remains an active area of research in the spatial domain and in the aviation community and has attracted the attention of many researchers. An excellent survey of various CDR methods is presented in [27]. In this survey, Kuchar et al. propose a taxonomy to categorize the basic functions of CDR modeling methods. The proposed taxonomy includes: dimensions of state information (lateral, vertical, or three-dimensional); method of state propagation (nominal, worst-case, or probabilistic); conflict detection threshold; conflict resolution method (prescribed, optimized, force field, or manual); maneuvering options (speed change, lateral, vertical, or combined maneuvers); and management of multiple aircraft conflicts (pairwise or global).

Conflict detection methods can be classified as nominal, worst-case, and probabilistic techniques. The nominal technique projects the current states into the future along a single trajectory without taking uncertainties into account [18, 21, 61]. The worst-case technique assumes that an aircraft will perform any of a set of maneuvers and a conflict is predicted if any of the maneuvers could cause a conflict [11, 58, 62]. The disadvantage of the worst-case technique is that it can declare a conflict as soon as there is a minimum likelihood of a conflict within the definition of the worst-case trajectory model thereby leading to false positives. The probabilistic approach offers a balance between relying on either a single trajectory model as in the nominal technique or a set of worst-case maneuvers. Instead it models uncertainties to describe potential changes in the future trajectory [13, 29–32, 41, 64, 65].

Once a conflict has been detected, the next step in the CDR process is to initiate the conflict resolution phase by determining the course of action. The conflict resolution method and the maneuvering options are two major factors in defining the course of action. Conflict resolution methods can be categorized as *a)* prescribed, *b)* optimized, *c)* force-field, and *d)* manual. The prescribed resolution method provides a fixed maneuver based on a set of predefined procedures [14]. Hence, depending on the nature of the conflict, the predefined resolution maneuver is automatically performed, minimizing the response time. However, it does not compute the optimal resolution path for the aircraft, thereby resulting in a less efficient trajectory. The optimized method provides a conflict resolution strategy with the lowest cost based on a certain cost function (separation, fuel, time, workload, etc.) [21]. In the force-field resolution method, each aircraft is treated as a charged particle and the resolution maneuvers are defined using repulsive forces between the aircraft [66]. Although it is practical when properly applied, it may require a high level of guidance on the flight deck especially when the aircraft vary their speed over a wide range. The manual resolution method allows users to generate potential resolution options and provide feedback if the option is viable [32, 64].

During the resolution phase, some CDR approaches only offer a single maneuver such as speed change [16, 24] or lateral maneuver [42] or vertical maneuver [17, 21], while others offer a combination of these maneuvers [13, 25, 33]. Obviously, the more maneuvering options the CDR approach offers, the more likely an efficient solution can be provided to a conflict. Our CDR approach has some similarities with [15, 16] due to fact that both approaches consider the airspace as a set of cubic cells, called the grid model, declare a conflict if one of the aircraft's predicted trajectory segments overlap with the other's protected zone, and propose an optimized 4D trajectory for the conflict resolution. However, unlike our study, they attempt to address tactical CDR (short-term and medium-term) by changing speed profiles [16] or using Particle Swarm Optimization [15].

Summarizing the above literature and comparing to the proposed prescriptive analytics approach, we make the following remarks: *i)* Our CDR modeling method considers dimensions of state information as 4D (latitude, longitude, altitude, and timestamp), *ii)* Due to input trajectories being probabilistically computed, our conflict detection method is also considered probabilistic, *iii)* Our conflict resolution method is optimized, and it offers a combination of lateral and vertical maneuvers. *iv)* Our framework is scalable, i.e., it can address the CDR problem among multiple aircraft. Given a set of aircraft trajectories in 4D, our framework declares a conflict if any of the aircraft's predicted trajectory segments overlaps with the other's protected zone at the same time interval in the future. In our approach, one or more cubic cells forming the airspace is considered to be a conflict detection threshold. Next, we compute and prescribe an optimized solution which is a conflict-free 4D trajectory. Our approach addresses the CDR problem strategically over a time horizon of several hours to compute conflict-free 4D trajectories for optimal flight plans and less complex sector traffic densities. In particular, the proposed data-driven approach exploits machine learning techniques to predict conflicts and prescribe an optimized solution based on constraints introduced to the framework.

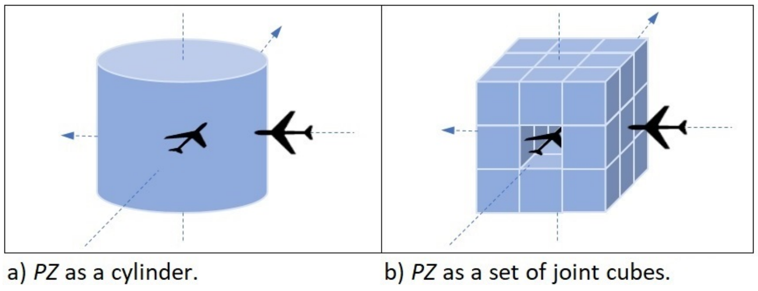


Fig. 1. A sample crossing conflict and a *PZ*. (left) regular representation, (right) our unique representation.

3 PRELIMINARIES

The primary concern of the ANSPs, for example; the FAA in the USA and EUROCONTROL in Europe is to assure safety, which is quantified by the number of resolved conflicts.

Definition 3.1. A **conflict** is an event in which two or more aircraft come closer than a certain distance to one another.

Definition 3.2. **Separation minima** are encoded by lateral and vertical separation, forming a bounding volume around each aircraft, a *protected zone(PZ)* also known as a buffer zone or corridor in Geographic Information Systems (GIS) applications (e.g., [1–3, 57]) and can be represented using quadtree medial axis transforms [44, 45]. Currently, the minimum lateral separation for en-route airspace is 5nmi. It is 3nmi inside the terminal radar approach control (TRACON) area. The minimum vertical separation is 2000 ft above the altitude of 29000 ft (FL290) and 1000 ft below FL290. Due to fact that lateral and vertical separation are specified by single distance values, the resulting *PZ* becomes a cylinder. Each aircraft is assumed to be surrounded by a *PZ* that moves along the aircraft. An interesting idea is to use the Hausdorff distance [35] to define the cubes.

Definition 3.3. **Conflict detection** is a process that evaluates the separation between any pair of aircraft, by comparing the distance between them with the separation minima. Formally, given a pair of predicted aircraft trajectories formed by a set of aircraft positions $T_i = [p_{i1}, p_{i2}, ..., p_{im}]$, $T_j = [p_{j1}, p_{j2}, ..., p_{jn}]$ where each point p is defined by its 4D spatio-temporal parameters *(latitude, longitude, altitude,* and *timestamp)*, distance values between them $d_{i,j}$ are computed and compared with the separation minima d_s and a conflict is declared if any of distance values is less than the separation minima $d_{i,j} < d_s$.

Definition 3.4. **Conflict resolution** is a process that generates a feasible safe alternative trajectory by fulfilling the separation minima criteria. Formally, given a pair of predicted aircraft trajectories formed by a set of aircraft positions $T_i = [p_{i1}, p_{i2}, ..., p_{im}]$, $T_j = [p_{j1}, p_{j2}, ..., p_{jn}]$, upon conflict resolution, all the distance values $d_{i,j}$ between the pairs of aircraft positions are greater than the separation minima $d_{i,j} > d_s$.

Definition 3.5. **Long-range CDR** is a process in which conflict detection and resolution are carried out several hours before the potential conflict occurs. Hence, long-range CDR is strategically performed before the departure for better planning, whereas mid-range and short-range CDR are tactically performed while the aircraft is airborne.

Unlike online CDR approaches in which distance between predicted future aircraft positions are constantly computed and compared with separation minima upon receipt of each new aircraft position, our approach is offline and uses a 3D grid network as a reference system. In our approach, raw trajectories are transformed into *aligned trajectories* causing aircraft to move along grid points. This results in conflict queries being computed

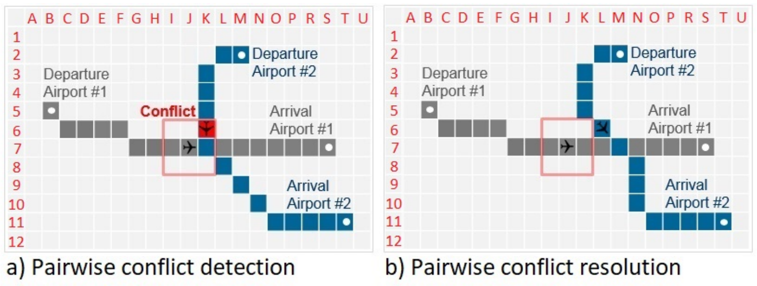


Fig. 2. Simplified depiction of a pairwise CDR.

at grid points only. In addition, unlike most other CDR approaches, our framework considers a cube shaped PZ surrounding each aircraft. This idea resonates with the fact that our approach creates virtual data cubes around grid points, forming an overall airspace. Each cube is defined by its centroid, the original grid point, and associated weather parameters that remain homogeneous within the cube during a period of time. With this vision, we define trajectories as a set of 4D joint cubes.

This uncommon representation of 4D trajectories enables us to view conflicts and PZ from a unique perspective. Hence, in our view, PZ is a cube that can be expanded by joining the neighboring cubes horizontally and vertically. The process yields a PZ with a desired size. In our study, we use a PZ of variable size. It can be made up of a single cube or expanded by a number of cubes in each direction on each axis reaching a larger volume. Figure 1 illustrates a PZ in two different forms. The PZ on the left is formed by a cylinder. The PZ on the right is formed by 27 cubes. Figure 2 illustrates a sample pairwise CDR. Aircraft #1 departs before aircraft #2. Both aircraft move one cube at a time. In Figure 2a. aircraft #1 and #2 are located at cube $J7$ and $K6$, respectively. This causes a conflict as aircraft #2 intrudes aircraft #1's PZ outlined in red. In Figure 2b. the conflict is resolved by a lateral shift to cube $L6$ by aircraft #2. No conflict occurs from here on as aircraft #1 follows cubes in gray ($K7, L7, M7, N7, O7, P7, R7, S7$) and aircraft #2 follows cubes in blue ($M7, N8, N9, N10, O11, P11, R11, S11, T11$) until they land at their pertinent airports.

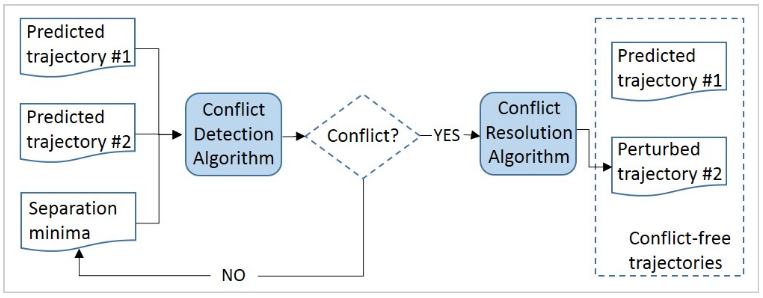


Fig. 3. Overview of our Data-Driven Framework.

4 DATA-DRIVEN FRAMEWORK FOR CDR

Figure 3 shows an overview of the proposed Data-Driven Framework for a simplified pairwise CDR. Predicted trajectory #1 and #2 are generated by our previous Aircraft Trajectory Prediction System [9]. Our conflict detection algorithm takes predicted trajectories along with separation minima as input. The output of the process is a data cube defined by its 4D position, where conflict, if any, occurs. The next process in the pipeline is the conflict resolution, where a variant of the Viterbi algorithm is performed to avoid the conflicting trajectory segment. The process perturbs at least one of the trajectories involved in the conflict, generating a new optimized path and thereby resulting in conflict-free trajectories. Separation minima may be expanded to probe what-if scenarios if the conflict detection algorithm doesn't generate a conflict.

4.1 Pairwise Conflict Detection

The predicted trajectories along with the size of the PZ are fed into our pairwise conflict detection algorithm, presented in Algorithm 1. The algorithm declares a conflict if a PZ of an aircraft is infringed upon by another aircraft at the same time interval in the future.

Formally, given a pair of predicted trajectories $T_i = [p_{i1}, p_{i2}, ..., p_{im}]$ and $T_j = [p_{j1}, p_{j2}, ..., p_{jn}]$ where each trajectory is formed by a set of segments defined by their 4D spatio-temporal centroid parameters *latitude, longitude, altitude*, and *timestamp*, along with PZ for trajectory T_i in data cubes, we want to return the very first trajectory segment p_{js}, if a conflict occurs, *null* otherwise. Note that each aircraft position along predicted trajectories are recorded once every minute. To compute this, we start with the departure point of the aircraft that departs beforehand, and keep moving forward, one grid point at a time. With the departure of the second aircraft, we compare the trajectory segment p_{js} with PZ_{is} at each time instance t_s to see if p_{js} overlaps with PZ_{is}.

Algorithm 1: Pairwise Conflict Detection

```
Result: Detected conflict or no conflict
Input   : Trajectory pairs $T_i$, $T_j$, Protected Zone $PZ_i$
Output : Conflicting trajectory segment $p_{js}$ or null

1 $T_i \leftarrow [p_{i1}, p_{i2}, ..., p_{im}]$
2 $T_j \leftarrow [p_{j1}, p_{j2}, ..., p_{jn}]$
3 foreach $t_s \in (p_{is} \cap p_{js})$ do
4     if $p_{js} \subset PZ_{is}$ then
5         return $p_{js}$
6     end
7 end
8 return null
```

Note that Algorithm 1 assumes that all the PZs have the same definition.

4.2 Pairwise Conflict Resolution

Our conflict resolution approach shares a common ground with our previous Trajectory Prediction System [9] as they both attempt to address an optimization problem; *given a set of weather observations, what is the most likely sequence of aircraft positions*? However, unlike the previous system [9], the current conflict resolution algorithm avoids the spatio-temporal data cubes where the conflict is detected. To recapitulate our approach to the optimization problem, we briefly review our previous Trajectory Prediction System [9] here.

Given a set of historical trajectories along with pertinent weather observations, the system works based upon an assumption that the weather observations are realizations of hidden aircraft positions i.e. trajectory segments and the transitions between the underlying hidden segments following a Hidden Markov model [43]. This assumption considers a finite set of states, each of which is associated with a probability distribution over all possible trajectory segments. Transitions among the states are managed by a set of probabilities. The states are not visible, but the pertinent observations are. Given a sequence of observations, the system trains an HMM, a statistical Markov model, and derives a sequence of hidden states, aircraft positions that correspond to the sequence of weather observations. The system computes the most likely sequence of aircraft positions in three steps:

- In the training data processing step, the system transforms raw trajectories into aligned trajectories and combines weather parameters for each grid point along aligned trajectories. To achieve this, the system uses a 3D grid network with a spatial resolution of 6km x 6km as a reference system.
- In the test data processing step, the system resamples the weather parameters to generate buckets with distinct ranges and feeds them into the time series clustering algorithm [10] to produce input observations.
- In the final step, the HMM parameters generated in the first two steps and the flight time computed by our Estimated Time of Arrival (ETA) Prediction System [5] are used as input to the Viterbi algorithm. The output is the optimal state sequence, joint 4D cubes defining aircraft trajectories.

During the conflict resolution stage, our current framework makes use of the first two steps, outlined above. However, unlike the 3rd step of the process, our current framework avoids the cubes where the conflict is detected. This translates to a perturbation of at least one of the trajectories. Hence, the framework prescribes an optimized solution by perturbing at least one of the 4D trajectories, involved in the conflict. The process executes as follows: In addition to the regular HMM parameters of transition, emission, and initial probabilities, the framework uses conflict-free probabilities where each state is assigned a probability value indicating how conflict-free it is. The parameters are fed into a variant of the Viterbi Algorithm, in which the framework computes the optimal state sequence by considering the maximum HMM probabilities. Due to fact that the trajectory segments that are part of the first aircraft's trajectory are assigned low conflict-free probabilities, the framework avoids selecting them during the Viterbi process, yielding a conflict-free trajectory for the second aircraft.

Now, we present a variant of the Viterbi algorithm. Note that our previous Trajectory Prediction System [9] characterized an HMM by the following elements:

- N, the number of states in the model. States $S = \{S_1, S_2, ..., S_N\}$ are represented by reference points' coordinates *(latitude, longitude, altitude)* that form aligned trajectories. We denote state at time t as q_t.
- M, the number of distinct observation symbols per state. Observations $V = \{v_1, v_2, ..., v_M\}$ are represented by weather parameters *(temperature, wind speed, wind direction, humidity)* recorded at grid points.
- The state transition probability distribution $A = \{a_{ij}\}$ is the probability of an aircraft discretely transitioning from one state i to another j along its aligned trajectory, where

$$a_{ij} = P[q_{t+1} = S_j | q_t = S_i], \quad 1 \leq i, j \leq N$$

- The observation symbol probability distribution in state j, $B = b_j(k)$ is the probability of discrete weather parameters having been observed at that specific state, where

$$b_j(k) = P[v_k \ at \ t | q_t = S_j], \quad \begin{array}{c} 1 \leq j \leq N \\ 1 \leq k \leq M \end{array}$$

- The initial state distribution $\pi = \{\pi_i\}$ is the probability of an aligned trajectory beginning at a state i, where

$$\pi_i = P[q_1 = S_i], \quad 1 \leq i \leq N$$

These parameters form an HMM, compactly denoted by $\lambda = \{A, B, \text{and } \pi\}$. Now, we propose an additional parameter;

- The conflict-free probability distribution in state j, $C = c_j(k)$ is the probability of a conflict not occurring at that specific state, where

$$c_j(k) = P[v_k \ at \ t | q_t = S_j], \quad \begin{array}{c} 1 \leq j \leq N \\ 1 \leq k \leq M \end{array}$$

Hence, the lower the conflict-free probability for a particular state, the lower likelihood of that state to be included in the most probable path. With this new parameter, C an HMM can be expanded and denoted by $\lambda = \{A, B, \pi$ and $C\}$

The next step in the process is to choose a corresponding state sequence $Q = q_1, q_2, \ldots, q_T$ that best explains the observation sequence $O = O_1, O_2, \ldots, O_T$ given the model λ. A variant of the Viterbi algorithm [63] that is based on dynamic programming addresses this problem. The key component in the algorithm is the *optimal probability*, $\delta_t(j)$, and is computed as follows:

$$\delta_t(j) = \max_{q_1, \ldots, q_{t-1}} \pi_{q_1} b_{q_1}(o_1) c_{q_1}(o_1) \prod_{j=2}^{t} (a_{q_{j-1}, q_j} b_{q_j}(o_j) c_{q_j}(o_j))$$

Due to their low conflict-free probabilities, a variant of the Viterbi algorithm avoids trajectory segments where the conflict is detected, and generates a new optimized path.

4.3 CDR for Multiple Aircraft

Our pairwise CDR solution can be scaled to address CDR for multiple aircraft which can be used towards better planning of airspace sector densities. Similar to the current flight planning procedures, we propose predicted trajectories to be processed on a first come first served basis. For the sake of simplicity, consider an empty airspace. The airspace will be fully available to the first predicted trajectory. Hence, the first trajectory's optimal set of data cubes will be reserved in the airspace. This process will introduce a set of constraints in the form of PZs by the first trajectory to the second and following trajectories during its flight time. Any conflict between the first and second trajectory will be detected and resolved. Once resolved, a new set of constraints will be introduced by the second trajectory. The next trajectory will need to satisfy the constraints introduced by the previous trajectories and so on. This also means that the next trajectory will need to avoid the PZs in the constraints list while generating its optimal set of data cubes during the conflict resolution phase. This process is repeated until one or more flights land, which will result in the removal of all pertinent constraints from the list which will free up the particular sections of airspace.

Formally, given a new predicted trajectory $T_j = [p_{j1}, p_{j2}, \ldots, p_{jl}]$ and a time series of existing constraints list $CL_{PZ} = [PZ_1, PZ_2, \ldots, PZ_i, \ldots, PZ_m]$, where $PZ_i = [pz_{i1}, pz_{i2}, \ldots, pz_{ik}]$ along their existing trajectories $T = [T_1, T_2, \ldots, T_i, \ldots, T_n]$, where $T_i = [p_{i1}, p_{i2}, \ldots, p_{ik}]$, we want to detect and resolve conflicts among these trajectories T_j and T (i.e. *one vs. all*). Note that PZ_i is formed by a set of data cubes, each defined by 4D spatio-temporal parameters around its centroid p_{ij}. With this approach, the implementation and management of the constraints list is of central importance. In our implementation, we map time instances to data cubes forming PZs, which are stored in an Octree data structure, where there is one Octree for each time instance. Hence, when a new trajectory comes in, the pertinent Octree is located based on the trajectory's time instance. We declare a conflict if the trajectory segment is found in the Octree. We process all pairs of possibly conflicting data cubes using spatial indexing techniques to prune the search (e.g., [46, 48]).

Our scalable conflict detection algorithm for multiple aircraft is presented in Algorithm 2. To detect a conflict, Algorithm 2 performs time instance lookup in constant time $O(1)$ for each trajectory segment. Next, Algorithm 2 performs a lookup operation for each trajectory segment in each Octree that has been created for each time instance, which have a running time of $O(logn)$, where n is the number of trajectory segments in the pertinent Octree. Given a new trajectory with k segments, the worst-case running time of Algorithm 2 is $O(k * logn)$, as

Algorithm 2: Conflict Detection for Multiple Aircraft

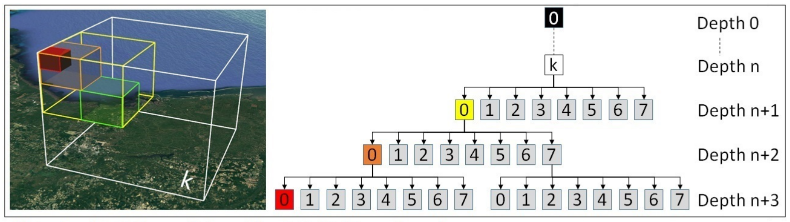
```
Result: Detected conflict or approved optimal trajectory
Input  : A new predicted trajectory $T_i$, constraints list $CL_{PZ}$
Output : Conflicting trajectory segment $p_i$ or updated constraints list $CL_{PZ}$
1  $T_i \leftarrow [p_{i1}, p_{i2}, ..., p_{in}]$
2  $CL_{PZ} \leftarrow [PZ_1, PZ_2, ..., PZ_m]$
3  foreach $t_s \in (p_i \& CL_{PZ})$ do
4      PruneAndSearch
5      if $p_i \subset CL_{PZ}$ then
6          return $p_i$
7      end
8      else
9          Insert $p_i$ into $CL_{PZ}$
10     end
11 end
```

the conflict is declared as soon as a new trajectory segment is found in any Octree for k time instances. Here and in the rest of this work, we assume that the Octree is balanced.

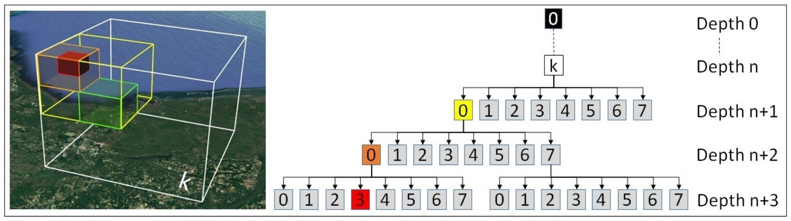
To keep the constraints list up-to-date, we periodically check and delete the pertinent Octree if any of the time instances has expired. Once a conflict has been detected, a variant of the Viterbi algorithm is performed where all the data cubes included in the constraints list for the pertinent time instance are avoided to find a conflict-free, optimized path for the new trajectory. To achieve this, we assign a minimum conflict-free probability value, such as 1×10^{-100} to those data cubes included in the constraints list.

Figure 4 is an illustration of a simplified case where two consecutive trajectory segments of aircraft #1 and #2 are inserted into two separate Octree data structures at time instances *t* and *t+1*. For the sake of simplicity, the size of aircraft #1's *PZ* is considered as a single data cube. Assuming an empty airspace, the very first *PZ* set for aircraft #1 is allocated in the airspace without being considered for a potential conflict. This translates to the fact that an Octree for time instance *t* and another Octree for time instance *t+1* are created as they don't exist yet. Next, given the predicted trajectory for aircraft #1, each consecutive *PZ* in solid red is inserted into each Octree at time instance *t* and *t+1* as shown in Figures 4a and 4b, respectively. When the predicted trajectory segments of aircraft #2 in solid blue in Figures 4c, and 4d are received, the Octree data structure at each time instance *t* and *t+1* are searched to see if the pertinent Octree node is empty. To achieve this, each Octree is looked up based on its index. Next, the pertinent Octree is searched as each time instance is mapped to an Octree. If found empty, each consecutive trajectory segment is inserted into each Octree at time instance *t* and *t+1*, as shown in Figures 4c and 4d, respectively. The insert operation for the first trajectory segment of aircraft #2 in solid blue is performed by traversing the *node k* at *depth n* illustrated as white cube mesh, followed by the *node 0* at *depth n+1* illustrated as yellow cube mesh, followed by the *node 7* at *depth n+2* illustrated as green cube mesh, and finally reaching to the leaf *node 7* at *depth n+3* and populating it with solid blue as shown in Figure 4c. Upon search, if the leaf node is not found empty, which translates to a conflict, our conflict resolution Algorithm 3 is executed to perturb the predicted trajectory for aircraft #2 to find an alternative optimal solution that results in conflict-free trajectories.

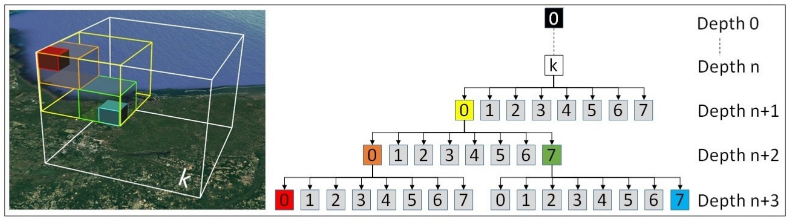
Formally, given the number of states N, number of distinct observations per state M, state transition probability distribution A, observation probability distribution B, and initial state probability distribution π, along with a conflict-free probability distribution for all data cubes included in the constraints list at time instances t =



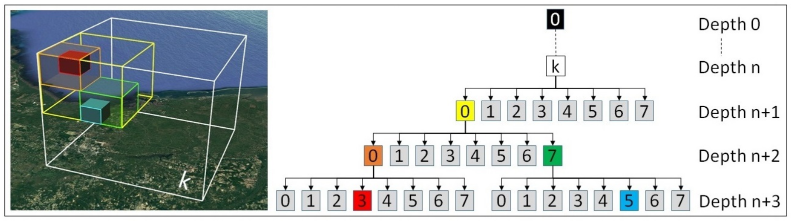
a) A single *PZ* in solid red in OCTREE at time instance *t*.



b) A single *PZ* in solid red in OCTREE at time instance *t+1*.



c) *PZ* in solid red for aircraft #1 and *PZ* in solid blue for aircraft #2 in OCTREE at time instance *t*.



d) *PZ* in solid red for aircraft #1 and *PZ* in solid blue for aircraft #2 in OCTREE at time instance *t+1*.

Fig. 4. Octree data structure for efficient aircraft conflict detection and resolution.

$[t_1, t_2, ..., t_n]$, we want to form an HMM λ, and train it to find the optimized path that is conflict-free. Our conflict resolution algorithm for multiple aircraft is presented in Algorithm 3.

To resolve a conflict, Algorithm 3 assigns a conflict-free probability of 1×10^{-100} to each state in the state space when a trajectory segment in the new trajectory is found in any Octree that exists for k time instances (i.e., the number of segments in the new trajectory), which has a running time of $O(k * logn)$. Algorithm 3 assigns a conflict-free probability of 1 to the remaining states in the state space, which has a running time of $O(N - k)$, where N is the number of states in the state space. Hence, the running time of Algorithm 3 between line 3 and line 11 is $O(k * logn + N - k)$. Algorithm 3 performs a variant of the Viterbi algorithm between line 12 and line 13, which has a running time of $O(k * |N^2|)$. Lastly, Algorithm 3 inserts conflict-free trajectory segments into the

Algorithm 3: Conflict Resolution for Multiple Aircraft

```
Result: A conflict-free optimized trajectory
Input  : # of states $N$, # of distinct observations $M$, transition probabilities $A$, emission probabilities $B$, initial
         probabilities $\pi$, constraints list $CL_{PZ}$
Output : A conflict-free optimized trajectory $T_o$ and updated constraints list $CL_{PZ}$
1  $S \leftarrow [p_1, p_2, ..., p_i, ..., p_n]$
2  $CL_{PZ} \leftarrow [PZ_1, PZ_2, ..., PZ_m]$
3  foreach $t_s \in ((p_i \in S)\ \&\ CL_{PZ})$ do
4      PruneAndSearch
5      if $p_i \subset CL_{PZ}$ then
6          $c_{p_i} \leftarrow 1 \times 10^{-100}$
7      end
8      else
9          $c_{p_i} \leftarrow 1$
10     end
11 end
12 VariantOfViterbi
13 $T_o \leftarrow \max_{s_1,...,s_{t-1}} \pi_{s_1} b_{s_1}(o_1) c_{s_1}(o_1) \prod_{j=2}^{t} (a_{s_{j-1},s_j} b_{s_j}(o_j) c_{s_j}(o_j))$
14 foreach $t_s \in ((p_o \in T_o)\ \&\ CL_{PZ})$ do
15     Insert $p_o$ into $CL_{PZ}$
16 end
```

constraints list between line 14 and line 16, which has a running time of $O(k * logn)$. Hence, the total running time of Algorithm 3 is $O(k(2logn + |N| - 1) + N)$.

5 EVALUATION

To evaluate our framework, we generated a number of test cases using real trajectory and weather data from Europe and USA. Table 1 shows the European and USA airports forming the routes we used in our evaluation. Although the ideal evaluation would use real trajectories in actual conflicts, this is infeasible due to the nature of ATC operations, where the controller would interfere and separate the aircraft as soon as they are likely to infringe upon one another, so there would be no conflicts to find. Hence, we used a total of 24 test cases formed by European and USA flights that were in close proximity to cause potential conflicts when a minimal perturbation was applied. 16 out of these 24 cases were used to test pairwise CDR, while 8 cases were used to test CDR for multiple aircraft.

5.1 Setup

Due to the fact that there exists no system that continuously records and stores exact positions of an aircraft's original trajectory [8], only a discrete set of sample data are recorded and stored which presumably represent a close approximation of the original trajectory. We call this a raw trajectory. The raw trajectory data from Europe was provided by Spanish ANSP, ENAIRE using a radar surveillance feed with a 5 seconds update rate. The raw data was wrangled as part of the Data-driven AiRcraft Trajectory prediction research (DART) project under the SESAR Joint Undertaking Work Programme [56]. The European trajectory data contains all commercial domestic flights for Spain, a total of 119,563 raw trajectories and 80,784,192 raw trajectory points for the period of January through November 2016. The fields of the raw trajectory data are as follows: *flight no, departure airport, arrival*

Table 1. A set of European and U.S.A. airports.

AirportCode	*AirportName*
LEAL	Alicante–Elche Airport
LEBL	Barcelona–El Prat Airport
LECO	A Coruña Airport
LEIB	Ibiza Airport
LEMD	Adolfo Suárez Madrid-Barajas Airport
LEMG	Málaga Airport
LEMH	Menorca Airport
LEPA	Palma de Mallorca Airport
LEVC	Valencia Airport
LEVX	Vigo–Peinador Airport
LEZL	Seville Airport
KATL	Hartsfield-Jackson Atlanta International Airport
KBOS	Boston Logan International Airport
KDFW	Dallas/Fort Worth International Airport
KJAX	Jacksonville International Airport
KLGA	NewYork LaGuardia Airport
KMIA	Miami International Airport
KORD	Chicago O'Hare International Airport
KPIT	Pittsburgh International Airport

airport, date, time, aircraft speed in *X, Y, Z* directions, and position information *(latitude, longitude, altitude)*. Note that, as a preprocessing step, we downsampled raw trajectory data from the original resolution of 5 seconds to 60 seconds and aligned them to our 3D reference grid [9]. The raw trajectory data from the USA was extracted from an Aircraft Situation Display to Industry (ASDI) data feed which is recorded once every 60 seconds, provided in near real-time by the FAA [23], and stored in an aviation data warehouse [4, 7]. The USA trajectory data contains flights between 8 major airports, a total of 4,628 raw trajectories and 450,919 raw trajectory points for the period of May 2010 through December 2015. The fields of the raw trajectory data are as follows: *source center, date, time, aircraft Id, speed, latitude, longitude* and *altitude*. Both European and USA weather data were extracted from the Global Forecast System (GFS), provided by NOAA [34]. The original data has 28-km spatial and 6-hour temporal resolution and it contains over 40 weather parameters including *atmospheric, cloud* and *ground* attributes for each grid point as part of its 3D weather model. Hence, for this study's geographic volume and time period of interest, over 160TB of weather data was collected.

Due to fact that our current framework aims at addressing the CDR problem before departure, at least a pair of predicted trajectories are needed as input. Hence, we used our previous system [9] to generate a pair set of predicted trajectories. Next, we searched and found a number of trajectory points between the two flights' trajectories in the first and second pair, where the *date, latitude, longitude,* and *altitude* values matched, and *time* mismatched. By perturbing one of the flights' departure time we *virtually* created conflicts, where both aircraft traversed the same trajectory point at the same time. Table 2 shows the final size of training and test data in number of trajectories (#$trjs$) and points (#pts).

To evaluate our framework on multiple CDR test cases, we generated a set of three flights that is a combination of the existing routes listed in Table 2. The additional test cases for multiple CDR with their final size of test data

Table 2. Sizes of training and test datasets for pairwise CDR.

$TestCase\#$	$Route\#1$	$TrainingSetSize$		$TestSetSize$		$Route\#2$	$TrainingSetSize$		$TestSetSize$	
		$\#trjs$	$\#pts$	$\#trjs$	$\#pts$		$\#trjs$	$\#pts$	$\#trjs$	$\#pts$
1	LEAL-LEBL	1118	55116	200	9860	LEMD-LEIB	2572	125623	200	9769
2	LEAL-LEBL	1118	55116	19	937	LEMD-LEMH	1056	68141	19	1226
3	LEAL-LEBL	1118	55116	152	7493	LEPA-LEMD	5116	306128	152	9095
4	LEBL-LEMG	1704	127451	43	3216	LEMD-LEMH	1056	68141	43	2775
5	LEBL-LEMG	1704	127451	180	13463	LEPA-LEMD	5116	306128	180	10771
6	LEBL-LEZL	2404	183343	41	3127	LEMD-LEAM	1434	70128	41	2005
7	LEBL-LEZL	2404	183343	46	3508	LEMD-LEMH	1056	68141	46	2968
8	LEBL-LEZL	2404	183343	164	12508	LEMG-LEMD	1403	75408	164	8815
9	LEBL-LEZL	2404	183343	210	16016	LEPA-LEMD	5116	306128	210	12566
10	LEIB-LEBL	1360	53443	259	10178	LEPA-LEMD	5116	306128	259	15498
11	LEIB-LEBL	1360	53443	158	6209	LEPA-LEVC	1426	50467	158	5592
12	LEMG-LEBL	1563	114767	38	2790	LEMD-LEAM	1434	70128	38	1858
13	LEMG-LEBL	1563	114767	46	3378	LEMD-LEIB	2572	125623	46	2247
14	LEZL-LEBL	2380	186299	40	3131	LEMD-LEIB	2572	125623	40	1954
15	KDFW-KJAX	1355	153970	19	2159	KATL-KMIA	1296	108005	19	1583
16	KLGA-KORD	1155	131454	62	7056	KPIT-KBOS	822	57490	62	4339

Table 3. Sizes of training and test datasets for multiple aircraft CDR.

$TestCase\#$	$Route\#1$	$TestSetSize$		$Route\#2$	$TestSetSize$		$Route\#3$	$TestSetSize$	
		$\#trjs$	$\#pts$		$\#trjs$	$\#pts$		$\#trjs$	$\#pts$
17	LEAL-LEBL	5	253	LEMD-LEMH	5	322	LEMD-LEPA	5	304
18	LEBL-LEMG	3	226	LEMD-LEMH	3	197	LEBL-LEZL	3	233
19	LEBL-LEMG	30	2244	LEMD-LEMH	30	1935	LEMD-LEPA	30	1795
20	LEBL-LEMG	22	1672	LEPA-LEMD	22	1321	LEBL-LEZL	22	1694
21	LEBL-LEZL	27	2059	LEMD-LEMH	27	1742	LEMD-LEPA	27	1616
22	LEBL-LEZL	164	12508	LEMG-LEMD	164	8815	LEPA-LEMD	210	12566
23	LEMG-LEBL	38	2790	LEMD-LEAM	38	1858	LEMD-LEIB	46	2247
24	LEMG-LEBL	6	441	LEMD-LEIB	6	384	LEMD-LEPA	6	363

in number of trajectories ($\#trjs$) and points ($\#pts$) are listed in Table 3. Table 3 does not include training data for each test case as they are already included in Table 2. Note that test case #22 and #23 use two separate *virtual* conflicts in each test case, where each conflict is addressed in sequence, i.e., first the *virtual* conflict between route #1 and route #2 is detected and resolved, next the *virtual* conflict between route #1 and #3 is detected and resolved. Unlike test cases #22 and #23, test cases #17 through #21 and #24 address multiple CDR all at once, i.e., all three routes share a common trajectory point. Hence, to form a set of three flights for test case #17 through #21 and #24, we searched and found a number of trajectory points between these three flights' trajectories, where the *date, latitude, longitude,* and *altitude* values matched, and *time* mismatched. Assuming that the predicted trajectories were received in the order of route #1, #2, and #3 respectively, first we perturbed the 2nd flights' departure time to create a *virtual* conflict between the first and second flight, which is detected and resolved using our multiple CDR approach. As an outcome, route #2 was perturbed at the conflict location, yielding a common trajectory point between route #1 and route #3. Next, we perturbed the 3rd flights' departure time to create a *virtual* conflict between the first and third flight, which is also detected and resolved using our multiple CDR approach. In all multiple CDR test cases listed in Table 3, we used an Octree data structure and executed Algorithm 2 and Algorithm 3 as highlighted in Section 4.3.

Note that overall, the trajectory data alone contains over 4 million trajectory points. In Tables 2 and 3, the test data represents the conflicting trajectories. Aside from these 1,677 trajectory pairs, we also bootstrapped by drawing 100 additional trajectory pairs with replacement from the trajectory set to test for the false positive cases. Hence, we evaluated our framework's effectiveness with a total of 3,554 (3,354 + 200) test trajectories on 16 test cases for pairwise CDR and an additional set of 1,234 test trajectories formed by three flights on 8 test cases for multiple CDR.

Figures 5a through 5p illustrate the pairwise test cases with their cumulative training and test data in white and yellow, respectively, whereas Figures 5r through 5y illustrate the test cases of three flights with their cumulative training and test data in white, yellow, and blue, respectively.

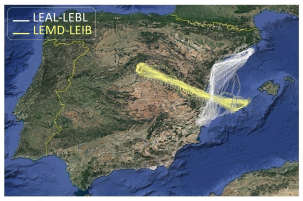
5.2 Results

We evaluated our framework by comparing the output of each step through the framework's pipeline with the ground truth on all 24 test cases. These steps include conflict detection and conflict resolution. However, due to fact that the accuracy of conflict detection and resolution significantly depend on the accuracy of predicted trajectories, we also computed trajectory prediction errors. For that purpose, we used trajectory prediction accuracy metrics as outlined in [37] and computed horizontal and vertical errors e_{horiz}, e_{vert} based on predicted trajectories generated by our previous system [9] versus raw trajectories of pertinent flights. The box plots in Figure 6 capture the aggregated errors for horizontal, along-track, and cross-track errors for each flight in all 24 test cases. The horizontal error is unsigned whereas the along-track, cross-track, and vertical errors are signed errors. Note that the mean value for the cross-track error and vertical error along the entire test trajectories are 7.693nmi and 1589.459ft respectively, when the sign is omitted.

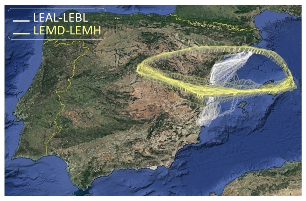
To evaluate our conflict detection capability, we used the same accuracy metrics as in [37] and computed horizontal and vertical errors by comparing the output of conflict detection and conflict resolution of our framework with the ground truth on all 24 test cases. To compute the errors, we fed a pair set of predicted trajectories for test case #1 through #16 and a set of three predicted trajectories for test case #17 through #24 into our conflict detection algorithm and compared the locations of *virtual* conflicts versus locations of conflicts detected by our framework. Next, we created 4 bin sizes, where each bin size is an integer multiple of 5nmi of lateral and 2000ft of vertical distances, conventionally accepted as minimum separation values for enroute airspace by ANSPs.

Table 4 presents the pertinent condition for each bin. Using horizontal and vertical error values, we counted the number of conflicts in each case and found the bin size to which they belong. The outcome is presented as a set of histograms in Figure 7. Our algorithm detected **87.1%** of the conflicts within the first bin size and **99%** of the conflicts within the first two bin sizes on all 24 test cases. Note that the conflict detection can only be as accurate as the predicted trajectories. Hence, these errors are attributed to the accuracy of our previous system [9], defined by the horizontal, and vertical error of **14.983nmi** and **1589.459ft** respectively along the entire test trajectories.

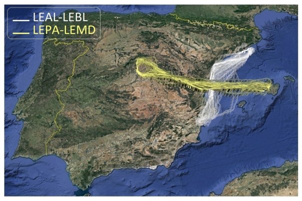
To resolve the conflicts, we ran our conflict resolution algorithm as highlighted in Section 4.2. Figure 8 provides a closer look at one of the detected and resolved conflicts by our framework for a pairwise CDR test case. In all figures, the yellow cubes and white cubes represent the first and second flight's, respectively, predicted trajectories. The red line parallel to the white cubes represents the first fight's actual trajectory and the red line parallel to the yellow cubes represents the second flight's actual trajectory. As both aircraft move one cube at a



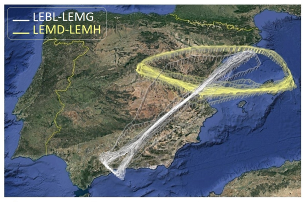
a) Test case #1: LEAL-LEBL and LEMD LEIB.



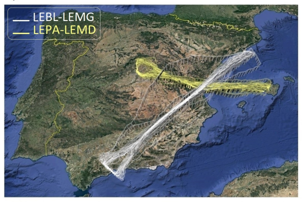
b) Test case #2: LEAL-LEBL and LEMD-LEMH.



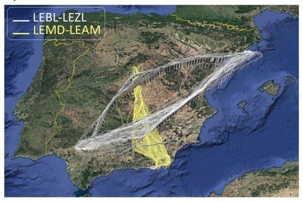
c) Test case #3: LEAL-LEBL and LEPA-LEMD.



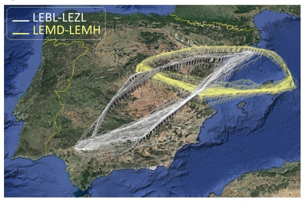
d) Test case #4: LEBL-LEMG and LEMD-LEMH.



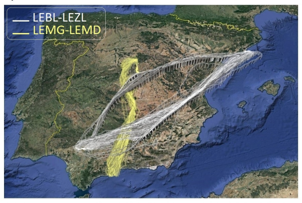
e) Test case #5: LEBL-LEMG and LEPA-LEMD.



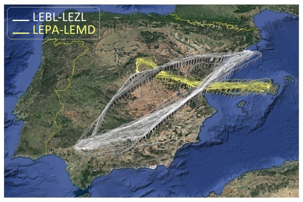
f) Test case #6: LEBL-LEZL and LEMD-LEAM.



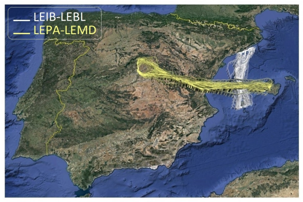
g) Test case #7: LEBL-LEZL and LEMD-LEMH.



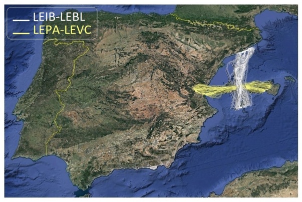
h) Test case #8: LEBL-LEZL and LEMG-LEMD.



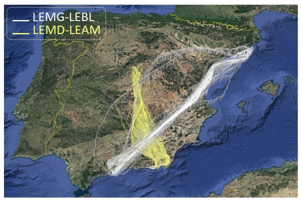
i) Test case #9: LEBL-LEZL and LEPA-LEMD.



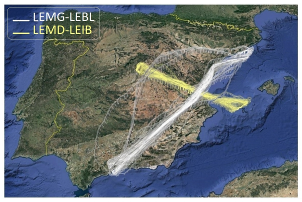
j) Test case #10: LEIB-LEBL and LEPA-LEMD.



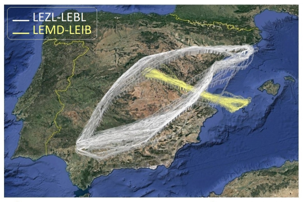
k) Test case #11: LEIB-LEBL and LEPA-LEVC.



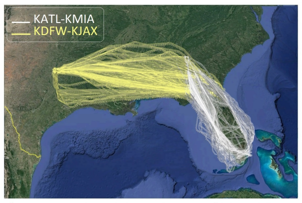
l) Test case #12: LEMG-LEBL and LEMD-LEAM.



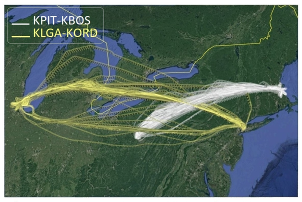
m) Test case #13: LEMG-LEBL and LEMD-LEIB.



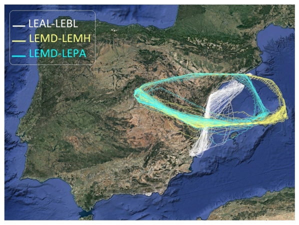
n) Test case #14: LEZL-LEBL and LEMD-LEIB.



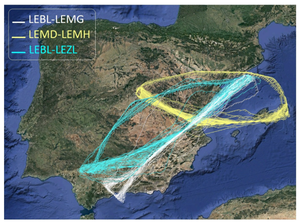
o) Test case #15: KATL-KMIA and KDFW-KJAX.



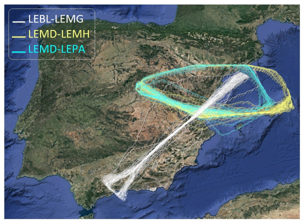
p) Test case #16: KPIT-KBOS and KLGA-KORD.



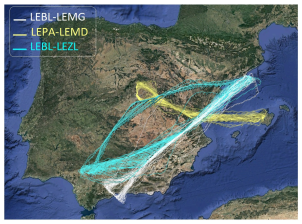
r) Test case #17: LEAL-LEBL and LEMD-LEMH and LEMD-LEPA.



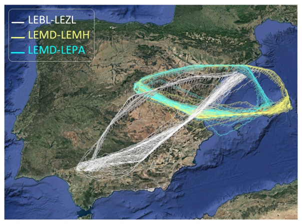
s) Test case #18: LEBL-LEMG and LEMD-LEMH and LEBL-LEZL.



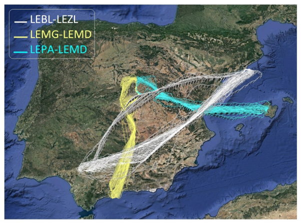
t) Test case #19: LEBL-LEMG and LEMD-LEMH and LEMD-LEPA.



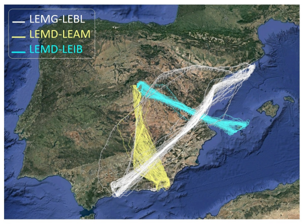
u) Test case #20: LEBL-LEMG and LEPA-LEMD and LEBL-LEZL.



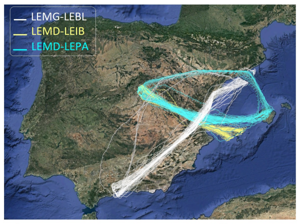
v) Test case #21: LEBL-LEZL and LEMD-LEMH and LEMD-LEPA.



w) Test case #22: LEBL-LEZL and LEMG LEMD, LEBL-LEZL and LEPA-LEMD.



x) Test case #23: LEMG-LEBL and LEMD-LEAM, LEMG-LEBL and LEMD-LEIB.



y) Test case #24: LEMG-LEBL and LEMD-LEIB and LEMD-LEPA.

Fig. 5. Visual representation of training and test data for conflicting flights.

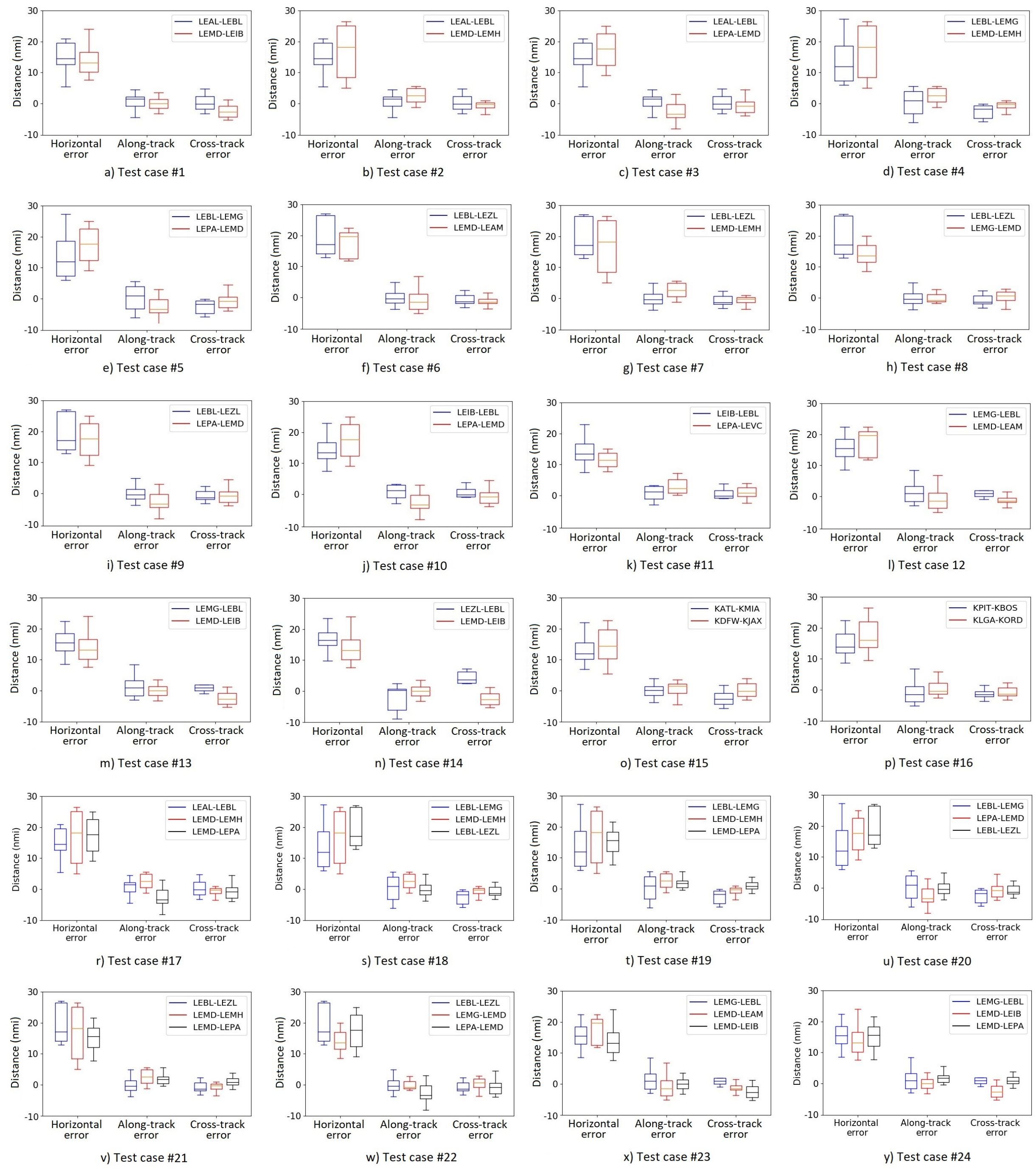


Fig. 6. Trajectory prediction errors in the form of box plots.

Table 4. Horizontal and vertical error ranges for each bin size.

$BinSizeId$	$Condition$	$Size$
1	0nmi $\leq e_{horiz} \leq$ 5nmi $\wedge$ 0ft $\leq e_{vert} \leq$ 2000ft	5nmi,2000ft
2	5nmi $< e_{horiz} \leq$ 10nmi $\vee$ 2000ft $< e_{vert} \leq$ 4000ft	10nmi,4000ft
3	10nmi $< e_{horiz} \leq$ 15nmi $\vee$ 4000ft $< e_{vert} \leq$ 6000ft	15nmi,6000ft
4	15nmi $< e_{horiz} \leq$ 20nmi $\vee$ 6000ft $< e_{vert} \leq$ 8000ft	20nmi,8000ft

time in the flight direction, the PZ illustrated in the cyan cube around the current position of the first flight also moves forward in the form of a sliding window. The first figure on the far left shows where each aircraft is at time t_{s-1}, represented respectively by the solid white cubes for the first and yellow cubes for the second flight. The second figure from the left captures the conflict detected at time interval t_s by our framework illustrated with a solid red cube. The actual conflict occurs where the red lines intersect. Note that both the predicted conflict position and the actual conflict position are within the first flight's PZ. The third figure from the left is the 3D view of the second figure from the left. The actual and predicted conflict positions are only 2 cube sizes away from each other, considering the center of the cube as the predicted position of the aircraft. The figure in the far right illustrates the optimized solution to the conflict by our framework. The first flight's trajectory has been perturbed vertically and the conflict has been resolved by our framework. The altitude of the second flight has been elevated resulting in conflict being resolved. Note that due to assignment of low conflict-free probabilities to the 3D grid points inside of the PZ, the framework avoids selecting them during the conflict resolution stage, generating conflict free trajectories.

As the final evaluation step, we executed our conflict resolution algorithm on all conflicts formed by a total of 4,788 (3,554 + 1,234) trajectories and computed accuracy values based on the number of successful resolutions, i.e. generating a feasible safe alternative trajectory by fulfilling the separation minima criteria. Due to the fact that our conflict resolution algorithm resolved the vast majority of conflicts on all test cases, we provide aggregated results overall, rather than provide results for each test case. Table 5 presents accuracy values for each bin size. Note that, to resolve the conflicts, we treated each bin size differently based on their varying sizes so that only relevant 3D grid points were avoided. The process perturbed the trajectory for the latter flight, generated an optimal alternative and yielded conflict-free trajectories.

6 DISCUSSION

During the trajectory prediction step, errors were computed by comparing the locations of *virtual* conflicts versus locations of conflicts detected by our framework. The trajectory prediction had mean horizontal and vertical errors of **14.983nmi** and **1589.459ft**, respectively along the entire trajectory points over the full test set from two continents. Using the predicted trajectories, our conflict detection algorithm had **6.013nmi** of mean horizontal error. Note that this value is considerably less than 14.983nmi, the mean horizontal error by our previous Trajectory Prediction System [9] along the entire trajectory points including climb, cruise and descent phases of a flight. This is due to two major facts: 1) Trajectory prediction accuracy during the cruise phase is often considerably higher than the climb and descent phases of the flight. 2) All conflicts we used in our experiments took place during the cruise phase of the flights. With 6.013nmi of mean horizontal error on the conflict positions, our conflict detection algorithm found **99%** of conflicts within the first two bin sizes of separation minima (10nmi, 4000ft). We also verified that between none of the additional 100 trajectory pairs where the *virtual* conflict never occurred was falsely detected as a conflict by our framework. Our conflict resolution algorithm's mean accuracy was over **97%** on all test cases. Though, it is interesting to see that it was unable to reach 100% accuracy, given the fact that all it had to do was avoid the selected 3D grid points when generating the new conflict-free optimal

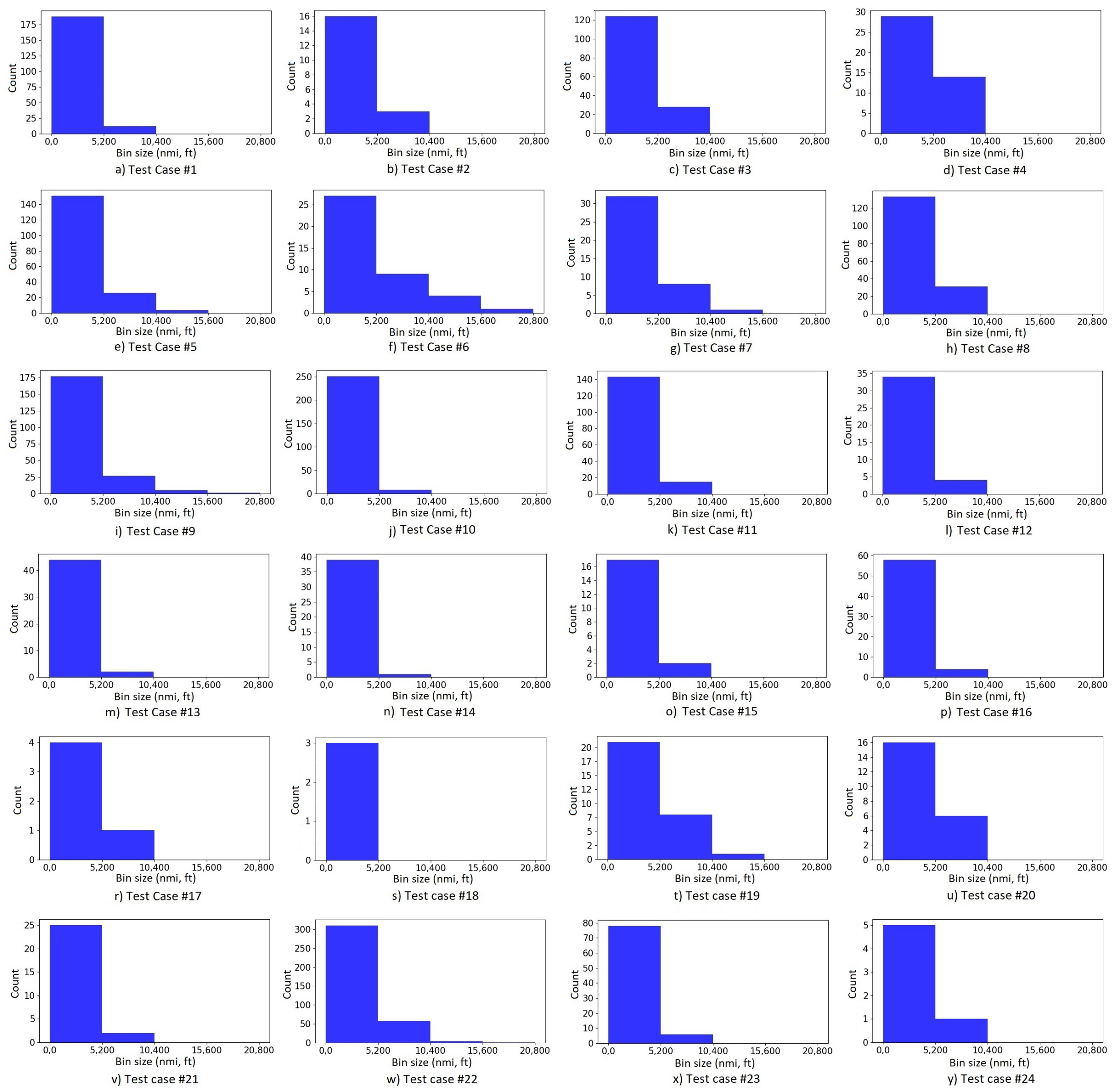


Fig. 7. Conflict detection histograms showing how many conflicts were captured by our framework in each bin size.

trajectory. The reason for that was the sparse distribution of data cubes, i.e. lack of data over the 3D grid network. The algorithm was unable to connect the new trajectory segments from start to end due to disconnection. Hence, our framework craves for more data to reach higher accuracy values.

These results validate the effectiveness of our framework on the long-range CDR problem. However, this is not to say that strategic CDR will detect and resolve conflicts once for all and no more conflicts will occur during

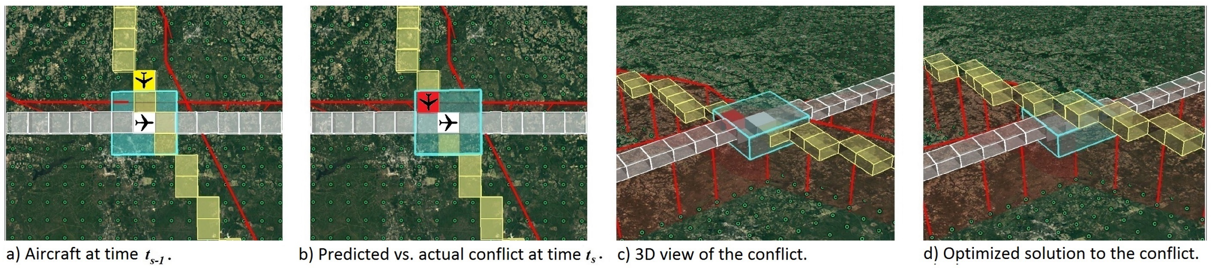

a) Aircraft at time t_{s-1}.
b) Predicted vs. actual conflict at time t_s.
c) 3D view of the conflict.
d) Optimized solution to the conflict.

Fig. 8. Illustration of a detected and resolved pairwise conflict by our framework.

the flight. There will likely be some convective weather patterns shaping after departure. These sudden changes causing potential conflicts should be addressed tactically by short and or medium-range CDR systems while the aircraft is airborne.

Although we were not able to find, there may also likely be some exceptional cases where false positive conflicts may be found. These cases should also be addressed tactically by short and or medium-range CDR systems. Note that the larger the selected *PZ* value, the higher probability of finding and resolving the conflicts between trajectories. However, that also means less denser sectors resulting in inefficient use of airspace. Hence, the tradeoff should be handled carefully by the ANSPs.

Overall, we propose our framework to be used by AOCs to file more realistic flight plans. Our framework can also be used by ANSPs to validate that the filed flight plans do not cause conflicts with previously approved flight plans or increase any sector traffic complexities. Air traffic density and complexity (a count of aircraft predicted to be in conflict with another) are two major factors defining the metric of air traffic controller workload [28]. These goals can be achieved in the planning phase before the aircraft depart, resulting in improvement in the four ATM key performance areas; safety, capacity, efficiency, and environmental impact, and thereby improving ATM automation and reducing the air traffic controller workload.

Table 5. Accuracy of our conflict resolution algorithm based on each bin size.

BinSizeId	*Size*	*Accuracy*
1	5nmi,2000ft	98.3%
2	10nmi,4000ft	97.5%
3	15nmi,6000ft	93.7%
4	20nmi,8000ft	100.0%

7 CONCLUSION

We have presented a novel Data-Driven Framework addressing a long-range CDR problem. Using a set of predicted trajectories, the framework delivers two major capabilities; *i)* conflict detection, and *ii)* conflict resolution before the flight depart. In the conflict detection stage, the framework declares a conflict when a *PZ* of an aircraft on its predicted trajectory is infringed upon by another aircraft at the same time interval in the future. In the conflict resolution stage, upon a conflict, the framework executes our conflict resolution algorithm that is derived from the Viterbi algorithm and prescribes a solution that resolves the conflict by perturbing at least one of the 4D trajectories.

Our experiments on real trajectory and weather datasets from two continents verify that our framework achieves lateral and vertical accuracies that are within the boundaries of conventionally accepted minimum separation values, set by the ANSPs. This translates to the fact that, ANSPs can now detect and resolve potential conflicts long before the aircraft depart, resulting in safer and greener skies with higher efficiency and capacity, and thereby reducing the air traffic controller workload. Some future work could involve adding a spatial browsing capability [12, 19, 47] for the trajectories as well as incorporating our methods in a distributed spatial environment [60].

ACKNOWLEDGMENTS

This work was supported in part by the NSF under Grants IIS-13-20791 and IIS-18-16889. We are grateful to the Boeing Research & Technology Europe, a member of the project DART [56] team for providing the European trajectory data used in this study. The project DART has received funding from the SESAR Joint Undertaking under Grant Agreement No. 699299 under the European Union's Horizon 2020 Research and Innovation Programme.

REFERENCES

[1] A. Amir, A. Efrat, P. Indyk, and H. Samet. 1999. Efficient regular data structures and algorithms for location and proximity problems.. In *Proceedings of the 40th IEEE Annual Symposium on Foundations of Computer Science*. New York, 160–170.

[2] A. Amir, A. Efrat, P. Indyk, and H. Samet. 2001. Efficient algorithms and regular data structures for dilation, location and proximity problems. *Algorithmica* 30, 2 (June 2001), 164–187.

[3] C. H. Ang, H. Samet, and C. A. Shaffer. 1990. A new region expansion for quadtrees. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 12, 7 (July 1990), 682–686.

[4] S. Ayhan, P. Comitz, and V. Stemkovski. 2009. Aviation Mashups. In *Proceedings of the IEEE/AIAA 28th Digital Avionics Systems Conference*. Orlando, FL, 6.D.5–1–6.D.5–9.

[5] S. Ayhan, P. Costas, and H. Samet. 2018. Predicting Estimated Time of Arrival for Commercial Flights. In *Proceedings of the 24nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. London, UK, 33–42.

[6] S. Ayhan, P. Costas, and H. Samet. 2018. Prescriptive Analytics System for Long-Range Aircraft Conflict Detection and Resolution. In *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. Seattle, WA.

[7] S. Ayhan, J. Pesce, P. Comitz, G. Gerberick, and S. Bliesner. 2012. Predictive Analytics with Surveillance Big Data. In *Proceedings of the 1st ACM SIGSPATIAL International Workshop on Analytics for Big Geospatial Data*.

[8] S. Ayhan and H. Samet. 2015. DICLERGE: Divide-Cluster-Merge Framework for Clustering Aircraft Trajectories. In *Proceedings of the 8th ACM SIGSPATIAL International Workshop on Computational Transportation Science*. Seattle, WA.

[9] S. Ayhan and H. Samet. 2016. Aircraft Trajectory Prediction Made Easy with Predictive Analytics. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. San Francisco, CA, 21–30.

[10] S. Ayhan and H. Samet. 2016. Time Series Clustering of Weather Observations in Predicting Climb Phase of Aircraft Trajectories. In *Proceedings of the 9th ACM SIGSPATIAL International Workshop on Computational Transportation Science*. San Francisco, CA, 25–30.

[11] K. Bilimoria, B. Sridhar, and G. Chatterji. 1996. Effects of Conflict Resolution Maneuvers and Traffic Density on Free Flight. In *AIAA GNC Conference and Exhibit*. San Diego, CA, 1–11.

[12] F. Brabec and H. Samet. 2007. Client-based spatial browsing on the world wide web. *IEEE Internet Computing* 11, 1 (January/February 2007), 52–59.

[13] D. J. Brudnicki, K. S. Lindsay, and A. L. McFarland. 1997. Assessment of Field Trials, Algorithmic Performance, and Benefits of the User Request Evaluation Tool (URET) Conflict Probe. In *Proceedings of the 16th DASC. AIAA/IEEE Digital Avionics Systems Conference*. Irvine, CA, 35–44.

[14] B. Carpenter and J. Kuchar. 1997. Probability-based Collision Alerting Logic for Closely-spaced Parallel Approach. In *Proceedings of the 35th Aerospace Sciences Meeting and Exhibit*. Reno, NV, 1–8.

[15] J. Cobano, D. Alejo, G. Heredia, and A. Ollero. 2013. 4D Trajectory Planning in ATM with an Anytime Stochastic Approach. In *Proceedings of the 3rd International Conference on ATACCS*. Naples, Italy, 1–8.

[16] J. A. Cobano, D. Alejo, A. Ollero, and A. Viguria. 2012. Efficient Conflict Resolution Method in Air Traffic Management Based on the Speed Assignment. In *Proceedings of the 2nd International Conference on Application and Theory of Automation in Command and Control Systems*. London, UK, 54–61.

[17] G. Dowek and C. Munoz. 2007. Conflict Detection and Resolution for 1,2,...,N Aircraft. In *7th AIAA Aviation Technology, Integration and Operations Conference*. Belfast, Northern Ireland, 1–13.

[18] V. N. Duong and E. G. Hoffman. 1997. Conflict Resolution Advisory Service in Autonomous Aircraft Operations. In *Proceedings of the 16th DASC. AIAA/IEEE Digital Avionics Systems Conference*. Irvine, CA, 10–17.

[19] C. Esperança and H. Samet. 2002. Experience with SAND/Tcl: a scripting tool for spatial databases. *Journal of Visual Languages and Computing* 13, 2 (April 2002), 229–255.

[20] FAA. 2009. *The NextGen.* https://www.faa.gov/nextgen/

[21] FAA. 2012. *Traffic Alert and Collision Avoidance System (TCAS II).* https://www.faa.gov/other_visit/aviation_industry/airline_operators/airline_safety/info/all_infos/media/2012/InFO12010.pdf

[22] FAA. 2015. *TAF.* http://taf.faa.gov/Downloads/TAFSummaryFY2015-2040.pdf

[23] FAA. 2016. *Aircraft Situation Display to Industry.* http://www.fly.faa.gov/ASDI/

[24] G. B. M. Heuvelink and H. A. Blom. 1998. Analysis and Optimization of Systems. Springer-Verlag, London, UK, Chapter An Alternative Method to Solve a Variational Inequality Applied to an Air Traffic Control Example, 617–628.

[25] R. Irvine. 1998. GEARS Conflict Resolution Algorithm. In *AIAA GNC Conference and Exhibit.* Boston, MA, 786–796.

[26] E. Jacox and H. Samet. 2008. Metric space similarity joins. *ACM Transactions on Database Systems* 33, 2 (June 2008), 7.

[27] J. Kuchar and L. Yang. 2000. A Review of Conflict Detection and Resolution Modeling Methods. *IEEE Transactions on Intelligent Transportation Systems* 1, 4 (December 2000), 179–189.

[28] I. V. Laudeman, S. G. Shelden, R. Branstrom, and J. L. Brasil. 1998. *Dynamic Density: An Air Traffic Management Metric.* Technical Report. NASA Ames Research Center, Mountain View, CA.

[29] T. Lauderdale. 2012. Probabilistic Conflict Detection for Robust Detection and Resolution. In *12th AIAA ATIO Conference and 14th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference.* Indianapolis, IN, 1–12.

[30] W. Liu and I. Hwang. 2011. Probabilistic Trajectory Prediction and Conflict Detection for Air Traffic Control. *Journal of Guidance, Control, and Dynamics* 34 (November 2011), 1779–1789.

[31] Y. Matsuno. 2013. *Probabilistic Conflict Detection in the Presence of Uncertainty.* Springer Japan, Tokyo, Japan, 17–33.

[32] D. McNally, R. Bach, and W. Chan. 1998. Field Test Evaluation of the CTAS Conflict Prediction and Trial Planning Capability. In *AIAA GNC Conference and Exhibit.* Boston, MA, 1686–1697.

[33] P. Menon, G. Sweriduk, and B. Sridhar. 1999. Optimal Strategies for Free-Flight Air Traffic Conflict Resolution. *Journal of Guidance, Control, and Dynamics* 22 (March 1999), 202–211.

[34] NOAA. 2016. *NCEP Global Forecast System.* https://www.ncdc.noaa.gov/data-access/model-data/model-datasets/global-forcast-system-gfs

[35] S. Nutanong, E. H. Jacox, and H. Samet. 2011. An incremental Hausdorff distance calculation algorithm. *PVLDB* 4, 8 (August 2011), 506–517.

[36] S. Nutanong and H. Samet. 2013. Memory-efficient algorithms for spatial network queries. In *Proceedings of the 29th IEEE International Conference on Data Engineering.* Brisbane, Australia, 649–660.

[37] M. Paglione and R. Oaks. 2007. Implementation and Metrics for a Trajectory Prediction Validation Methodology. In *AIAA GNC Conference and Exhibit.* Hilton Head, SC, 1–18.

[38] S. Peng and H. Samet. 2015. Analytical queries on road networks: An experimental evaluation of two system architectures.. In *Proceedings of the 23rd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems.* Seattle, WA.

[39] S. Peng and H. Samet. 2018. DOS: A spatial system offering extremely high-throughput road distance computations.. In *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems.* Seattle, WA, 199–208.

[40] S. Peng, J. Sankaranarayanan, and H. Samet. 2016. SPDO: High-Throughput Road Distance Computations on Spark Using Distance Oracles. In *Proceedings of the 32nd IEEE International Conference on Data Engineering.* Helsinki, Finland, 1239–1250.

[41] M. Prandini, J. Hu, J. Lygeros, and S. Sastry. 2000. A Probabilistic Approach to Aircraft Conflict Detection. *IEEE Transactions on Intelligent Transportation Systems* 1, 4 (December 2000), 199–220.

[42] M. Prandini, J. Lygeros, A. Nilim, and S. Sastry. 1999. A Probabilistic Framework for Aircraft Conflict Detection. In *AIAA GNC Conference and Exhibit.* Portland, OR, 1047–1057.

[43] L. Rabiner. 1990. Readings in Speech Recognition. Morgan Kaufmann, San Francisco, CA, Chapter A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, 267–296.

[44] H. Samet. 1983. A quadtree medial axis transform. *Commun. ACM* 26, 9 (September 1983), 680–693.

[45] H. Samet. 1985. Reconstruction of quadtrees from quadtree medial axis transforms. *Computer Vision, Graphics, and Image Processing* 29, 3 (March 1985), 311–328.

[46] H. Samet. 2006. *Foundations of Multidimensional and Metric Data Structures.* Morgan-Kaufmann, San Francisco, CA.

[47] H. Samet, H. Alborzi, F. Brabec, C. Esperança, G. R. Hjaltason, F. Morgan, and E. Tanin. 2003. Use of the SAND spatial browser for digital government applications. *Commun. ACM* 46, 1 (January 2003), 63–66.

[48] H. Samet and M. Tamminen. 1986. An improved approach to connected component labeling of images. In *Proceedings of Computer Vision and Pattern Recognition'86.* Miami Beach, FL, 312–318.

[49] J. Sankaranarayanan, H. Alborzi, and H. Samet. 2005. Efficient query processing on spatial networks. In *Proceedings of the 13th ACM International Symposium on Advances in Geographic Information Systems.* Bremen, Germany, 200–209.

[50] J. Sankaranarayanan, H. Alborzi, and H. Samet. 2006. Distance Join Queries on Spatial Networks. In *Proceedings of the 14th ACM International Symposium on Advances in Geographic Information Systems*. Arlington, VA, 211–218.

[51] J. Sankaranarayanan and H. Samet. 2009. Distance oracles for spatial networks. In *Proceedings of the 25th IEEE International Conference on Data Engineering*. Shanghai, China, 652–663.

[52] J. Sankaranarayanan and H. Samet. 2010. Query processing using distance oracles for spatial networks. *IEEE Transactions on Knowledge and Data Engineering* 22, 8 (August 2010), 1158–1175.

[53] J. Sankaranarayanan and H. Samet. 2010. Roads belong in databases. *IEEE Data Engineering Bulletin* 33, 2 (June 2010), 4–11.

[54] J. Sankaranarayanan, H. Samet, and H. Alborzi. 2009. Path oracles for spatial networks. *PVLDB* 2, 1 (August 2009), 1210–1221.

[55] SESAR. 1999. *Single European Sky ATM Research*. http://www.eurocontrol.int/dossiers/single-european-sky

[56] SESAR. 2017. *DART*. http://dart-research.eu/the-project/

[57] C. A. Shaffer and H. Samet. 1988. An algorithm to expand regions represented by linear quadtrees. *Image and Vision Computing* 6, 3 (August 1988), 162–168.

[58] M. Shewchun, J. Oh, and E. Feron. 1997. Linear Matrix Inequalities for Free Flight Conflict Problems. In *Proceedings of the 36th IEEE Conference on Decision and Control*. San Diego, CA, 2417–2422.

[59] M. Soler, A. Olivares, E. Staffetti, and J. Cegarra. 2011. Multi-phase Optimal Control Applied to 4D Business Trajectory Strategic Planning in Air Traffic Management. In *Proceedings of the 1st International Conference on ATACCS*. Barcelona, Spain, 68–78.

[60] E. Tanin, A. Harwood, and H. Samet. 2005. A distributed quadtree index for peer-to-peer settings. In *Proceedings of the 21st IEEE International Conference on Data Engineering*. Tokyo, Japan, 254–255.

[61] A. Valenzuela and D. Rivas. 2009. Conflict Detection and Resolution in Converging Air Traffic. In *9th AIAA ATIO Conference*. Hilton Head, SC, 1–13.

[62] A. Vink, S. Kauppinen, J. Beers, and K. Jong. 1997. Medium-term Conflict Detection in EATCHIP Phase III. In *Proceedings of the 16th DASC. AIAA/IEEE Digital Avionics Systems Conference*. 45–52.

[63] A. Viterbi. 1967. Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm. *IEEE Transactions on Information Theory* 13, 2 (April 1967), 260–269.

[64] L. Yang and J. Kuchar. 1998. Using Intent Information in Probabilistic Conflict Analysis. In *AIAA GNC Conference and Exhibit*. Boston, MA, 797–806.

[65] Y. Yang, K. Cai, and M. Prandini. 2017. Fast Algorithm Based on Computational Geometry for Probabilistic Aircraft Conflict Detection. In *Proceedings of the 2017 International Conference on Robotics and Artificial Intelligence*. Shanghai, China, 66–70.

[66] K. Zeghal and E. Hoffman. 1999. Design of Cockpit Displays for Limited Delegation of Separation Assurance. In *Proceedings of the 18th Digital Avionics Systems Conference*. St. Louis, MO, 1–8.