Test scenario generation for validation of a black-box automated aircraft trajectory generator

Miguel Consiglieri Pedroso Mendes Dias
miguelmendesdias@tecnico.ulisboa.pt

Instituto Superior Técnico, Universidade de Lisboa, Portugal

November 2021

Abstract

Considerable progress in Artificial Intelligence, supported by unprecedented hardware capabilities, promises to revolutionize aviation. In an emergency, an aircraft autonomous navigation system should propose a safe trajectory until a suitable diversion runway to save the aircraft in real time. However, aviation is safety-critical, since failures might incur loss of human lives. Validating these automatic functions is therefore fundamental. Due to systems increasing complexity, or black-box nature, formal validation methods become impractical.

We propose a data-driven framework to support the development and validation of a black-box automated emergency procedure generator. Smooth complete trajectories are reconstructed from FlightRadar24 historical measurements through convex optimization. These allow to evaluate whether the simplified terrain representation used by the system is sufficiently precise to accept usually flied routes as safe. Trajectory clusters are obtained thanks to Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) and for each cluster a list of suitable diversion airports is constructed from historical data, ensuring that an option always exists within a certain distance through a set cover problem. Each cluster is represented by a group of cells and the system is evaluated on them, ensuring relevant search space coverage. Representative environments are constructed using real restricted areas and historical weather obstacles. A genetic algorithm guides the search for challenging scenarios.

The proposed framework was validated by applying it on a prototype under development, successfully identifying several axis of improvement, hence, proving itself useful to support the design process.
Keywords: Black-Box Safety Validation, Falsification, Optimization, Trajectory Clustering, Big Data

1. Introduction

In 2018, IATA forecast a significant growth in air passenger numbers for the next two decades [1]. A shortage of qualified pilots was therefore expected. Although COVID-19 has greatly impacted passenger traffic, as IATA does not expect it to return to pre-COVID-19 levels until 2024 [2], the issue remains for the long-term. Additionally, pilots costs are significant part of aircraft operating costs.

For these reasons, working towards autonomous aircraft is important. It is equally important to develop functions to assist pilots in stressful emergency situations. In both of these contexts, a system capable of proposing in real-time a suitable diversion target and a safe and flyable trajectory to reach it and save the aircraft in case of emergency is useful. Several works on this topic can be found in the literature [3] [4]. However, validating such a system is a far from trivial necessity. Since aviation is a safety-critical domain, any system must undergo extensive validation and testing before certification and deployment. As systems become increasingly complex, formal validation methods become impractical, in favor of black-box techniques not requiring internal system knowledge.

This work addresses the design and implementation of an efficient strategy for the evaluation and validation of a black-box system as the one described above. A major challenge lies on the huge size of the search space, composed by aircraft and environment states. The framework should allow to either validate the system (ensuring good coverage of the search space) or find axis of improvement to support the design and development. To ensure that test scenarios are representative, these are constructed from FlightRadar24 recordings, real restricted areas and historical weather obstacles. The framework is tested on a use case prototype and successfully identifies axis of improvement.

2. Background

2.1. Safety Validation of Black-box Autonomous Systems

Autonomous systems are becoming increasingly capable, promising to revolutionize aviation. However, since aviation is safety-critical, these systems must undergo extensive validation and testing prior

to certification and deployment. A System Under Test (SUT) can be considered safe if no failure is found after adequate exploration of the input space, or if the failure probability is below an acceptable threshold. Since conventional testing does not scale well to the complexity and unknown internal nature of next-generation systems, black-box validation techniques are required.

The work in [5] has proposed an effective method for automatic generation of test inputs for embedded control systems by maximizing a coverage function with the help of a Genetic Algorithm.

In [6], a falsification problem is defined as:

1) Given model $\mathcal{M}$ (takes input $\mathbf{u}$ and outputs $\mathcal{M}(\mathbf{u})$) and a specification φ (temporal formula),

2) find an error input, that is, a signal $\mathbf{u}$ such that the corresponding output $\mathcal{M}(\mathbf{u})$ violates φ.

Real-valued robust semantics of temporal logics assigns a value to represent not only if a specification is true or false but also how robust, reducing hybrid system falsification to an optimization problem of generating inputs in the direction of decreasing robustness. To be confident about accepting the system if no falsifying inputs are found, guaranteeing good search space coverage is important. This is possible if enough exploration is performed.

A time-staged approach is followed to find a falsifying sequence of inputs. The approach can be converted to an adversarial form of Reinforcement Learning (RL). A two-layered optimization framework that balances exploration of new areas and exploitation of promising ones is proposed by [6]. In an upper layer, Monte Carlo Tree Search (MCTS) is used for high-level planning and controls this balance, picking a region from the input space. A lower layer performs hill-climbing optimization within the region selected to sample the input. The system is then simulated to obtain the corresponding output and the robustness is computed and fed back to the MCTS.

For the safety validation of next-generation Airborne Collision Avoidance Systems, [7] formulates the search for the most likely state trajectory leading to an event given only a simulator of the system as an RL problem and successfully solves it using MCTS. Figure 1 illustrates the test input generation cycle. The reward was designed to be maximized if an event is found and favor more likely sequences otherwise. It is defined by (1).

$$R(s_t, s_{t+1}) = \begin{cases} 0 & \text{if } s_t \in E \\ -\infty & \text{if } s_t \notin E, t \geq T \\ \log P(s_{t+1} \mid s_t) & \text{if } s_t \notin E, t < T \end{cases} \quad (1)$$

MCTS is one of the most successful sampling-based online approaches to RL. A search tree is incrementally built using sampling and forward simulation to inform the search and focus on the most promising areas. Several variations of MCTS exist. A detailed review of these is presented in [8].

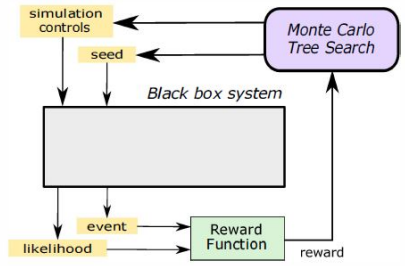

Figure 1: MCTS for test input generation. From [7].

Reference [9] presents a comprehensive survey of algorithms for black-box safety validation. The problem formulation is represented in Figure 2. There is an environment in which the system takes actions based on its observations. The problem is finding a disturbance trajectory that leads to failure. Disturbances could represent positioning errors, unexpected closing of restricted areas or yet unexpected temporal evolution of critical weather. The adversary has one of the following goals:

1. Falsification: find any disturbance trajectory that leads to a failure;
2. Most likely failure analysis: find the most likely disturbance trajectory that leads to a failure;
3. Estimation of the probability of failure: determine how likely any failure will occur.

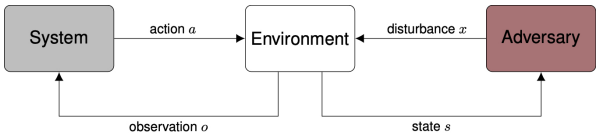

Figure 2: Problem formulation for the safety validation of black-box autonomous systems. From [9].

This work focuses on falsification. The taxonomy in [9] categorizes three approaches for this end: optimization (adaptive sampling of disturbance trajectories guided by a cost function) path-planning (trajectories built by choosing disturbances that bring the environment to unexplored regions) and Reinforcement Learning (algorithms select disturbances based only on the current state). Path-planing is not directly applicable to black-boxes.

This work only addresses a ”static” validation of the SUT, meaning its ability to produce a suitable solution considering the information initially available. An optimization approach seems suitable for this. The capacity to deal with disturbances (like an unexpected evolution of critical weather) that render the solution found and engaged invalid while flying it is not tested yet. However, finding a disturbance trajectory leading to failure is a fundamental

stream of future work. It could probably be well addressed by a Reinforcement Learning approach.

Several works have focused on falsification or on providing runtime assurances and ensuring search space coverage. However, it is important to interpret the decision-making process of the autonomous system and how environmental factors affect it. The work in [10] proposes to focus on regions where small changes in the scenario result in transitions between performance modes. Sampling preferentially from these regions allows to maximize the information returned by a limited number of runs. In a first step, adaptive sampling is used to search the parameter space. In a second step, unsupervised learning is used to determine performance modes and performance boundaries.

2.2. Trajectory Reconstruction

Reconstructing a complete and smooth trajectory from a set of recorded measurements can be formulated as a convex optimization problem [11].

Let q represent a sampling rate. If all trajectories are uniformed to have a same duration T_{ldg}, the number of points defining each reconstructed trajectory is $\mathrm{N} = T_{\text{ldg}}/q$. Let $P = (p_1, p_2, \ldots, p_N) \in \mathbb{R}^{N\times 3}$ be the optimization variable (trajectory to be reconstructed). The ith row of P, $p_i = (\text{ lat }_i, \text{lon }_i, \text{alt }_i)$, is the reconstructed position at time i.

To ensure reconstructed trajectories stay truthful to observations, a least-squares regression term $\|AP - \widehat{P}\|_F^2$, where $\|.\|_F$ defines the Frobenius norm, is inserted in the objective function. $\widehat{P} \in \mathbb{R}^{N\times 3}$ contains in its ith row measurements of time i if these are available and zero otherwise. $A \in \mathbb{R}^{N\times N}$ is a diagonal matrix with $A_{ii} = 1$ if an observation is available at time i and $A_{ii} = 0$ otherwise.

Realistic flight dynamics needs to be included when reconstructing trajectories. To ensure that the acceleration is low on average, a term $\lambda_1 \|D_2 P\|_F^2$ is introduced. The scalar λ_1 is a regularization hyper-parameter to be tuned and $D_2 \in \mathbb{R}^{N-2\times N}$ is the second-order difference matrix representing the acceleration operator, defined by (2).

$$(D_2)_i = e_i - 2e_{i+1} + e_{i+2},\ i \in \{1, \ldots, N-2\} \quad (2)$$

where e_i denotes the i^{th} standard unit vector.

To ensure that the jerk (change of acceleration) is also low on average, the term $\lambda_2 \|D_3 P\|_F^2$ is also introduced in the objective function. Here, λ_2 is another scalar regularization hyper-parameter to be tuned and $D_3 \in \mathbb{R}^{N-4\times N}$ is the third-order difference matrix representing the jerk operator (3).

$$(D_3)_i = -e_{i-1} + 2e_i - 2e_{i+2} + e_{i+3},\ i \in \{1, \ldots, N-4\} \quad (3)$$

Summing the terms just introduced, the objective function is defined and a convex unconstrained optimization problem is formulated by (4).

$$\underset{P}{\text{minimize}} \|AP - \widehat{P}\|_F^2 + \lambda_1 \|D_2 P\|_F^2 + \lambda_2 \|D_3 P\|_F^2 \quad (4)$$

Regarding the tuning of λ_1 and λ_2, the authors in [11] suggest a selection for each trajectory through out-of-sample validation, which is performed by randomly holding out measurements from the trajectory, fitting trajectories with varying λ_1 and λ_2 using the measurements not held out, and selecting the parameters that have the lowest loss on held-out measurements.

2.3. Trajectory Clustering

There are four basic clustering frameworks: partitioning, hierarchical, density-based and grid-based [12].

Given a set of n objects, partitioning methods construct K partitions and each object is assigned to exactly one (this may be relaxed). Usually, an initial partitioning is iteratively relocated based on the distance between objects until convergence. The two most popular algorithms are K-means and K-medoids. Major drawbacks are that K is a required input and they are fit to cluster spherical-shaped data, not data of irregular shape.

Hierarchical methods partition the data into groups at different levels, as in a hierarchy. They can be agglomerative (a bottom-up strategy where individual objects start as clusters and are iteratively merged to form larger clusters) and divisive (a top-down strategy where all objects start belonging to a single cluster and are iteratively divided into smaller clusters).

Density-based methods are based on the notion of probability density, rather than distance between objects. They can identify clusters of data of any irregular shape and identify and filter out outliers. The most popular algorithm is DBSCAN [13]. In a probabilistic framework, some approaches allow for the estimation of the optimal K from data [14].

Grid-based methods partition the space into cells, regardless of the distribution of input objects. Clustering operations are then performed on such a grid structure. For more details refer to [12].

The techniques described were originally introduced to cluster points. Clustering aircraft trajectories is more difficult, since one needs to choose not only a suitable algorithm but also a metric for similarity or dissimilarity between trajectories [15].

Euclidean distance is simple and intuitive because it is parameter-free. However, two trajectories being compared must be composed by a same number of segments with corresponding times. This is hard to ensure, since trajectories do not necessarily have the same temporal nor distance lengths. Noise may have a strong impact, but some

variations exist to address this problem. Other metrics allow to consider shape or time dimension.

Hausdorff distance is used to measure the maximum mismatch between two trajectories, considering them close if every sampling point of either trajectory is close to some sampling point of the other one. Being defined by the point in each trajectory that is farther to the other trajectory, it is sensitive to noise. It does not seem suitable to give an overall idea of how similar or dissimilar trajectories are.

Dynamic Time Warping (DTW) finds the optimal alignment between two trajectories, instead of using predefined correspondences like Euclidean distance does. Each sample from each trajectory must be monotonically non-decreasingly mapped to a sample of the other trajectory. An advantage is that the trajectories lengths can be different, so different sampling rates can be handled. The main inconvenience is the sensitivity to noise and need to have quite continuous trajectory data points.

At last, instead of a distance, one can also compute the Longest Common Subsequence (LCS). Several combinations of algorithms and metrics have already been successfully applied to problems involving aircraft trajectory clustering.

In [16], agglomerative hirearchical clustering was used to find trajectory patterns in the context of aircraft arrival times prediction. DTW was chosen to compute the distance between trajectories.

The work in [17] presents two clustering methods. The first one is way-point-based: it identifies turning points, uses DBSCAN to identify the main ones, represents trajectories as sequences of these and uses LCS to cluster them. It shows good empirical results. However, it only keeps trajectories going over way-points. A parallel trajectory might be seen as an outlier, while one containing a large rerouting period will belong to a cluster if it contains its way-points. In the second method, trajectories are resampled to obtain time series of equal length. The data is augmented and a Principal Components Analysis is run to reduce data dimensionality. Clusters and outliers are obtained using DBSCAN.

In [11], Euclidean distance and K-means++ are used. A Gaussian Mixture Model is constructed based on intra-cluster covariance matrices, allowing accurate inference and realistic generation of trajectories. The probabilistic generative model may be used for anomaly detection tasks as well.

3. System Under Test (SUT)

3.1. System Purpose and Interfaces

The SUT computes automatically and in real-time an emergency diversion solution, either to assist pilots or be automatically engaged by an autonomous aircraft, depending on the use case. It receives the aircraft position (latitude, longitude and altitude), speed (horizontal and vertical) and heading, restricted areas and critical weather obstacles to be avoided and a list of diversion options. It selects the diversion target from the list of options and outputs it and then computes and outputs a safe trajectory starting at the aircraft and ending at the runway. A safe trajectory avoids terrain, restricted areas and critical weather obstacles.

To ensure terrain avoidance, the SUT uses a simplified terrain representation obtained from a raster with elevation data such as NASA's Shuttle Radar Topography Mission (SRTM) database.

3.2. Restricted Areas and Weather Construction

In order to ensure test scenarios as realistic as possible, real prohibited areas, danger areas and restricted areas were recovered from Aeronautical Information Procedures (AIPs). The declared Spanish and French areas retrieved are represented on the left side of Figure 3.

An easy way to reproduce realistic weather obstacle shapes is to retrieve historical data. Historical weather images are retrieved and from them a library of representative shapes enclosing weather obstacles is created. A variety of scenarios can be created by selecting some shapes from this library and playing with their positioning and orientation. A weather scenario generated with this strategy is represented on the right side of Figure 3.

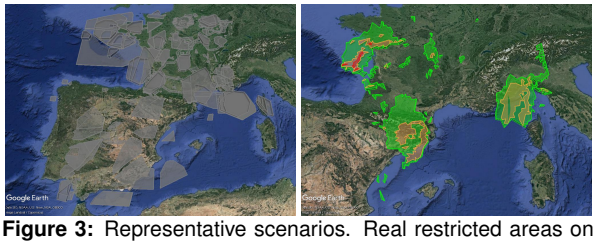

Figure 3: Representative scenarios. Real restricted areas on the left and weather generation from historical data on the right.

3.3. Validation of Solutions

The trajectory produced must be safe in light of the terrain. To ensure that such is the case, the terrain profile under the lateral trajectory is retrieved from NASA's Shuttle Radar Topography Mission (SRTM) [18] database. Such a terrain profile is represented in red in Figure 4 for one trajectory produced by the SUT. The trajectory is considered safe with respect to the terrain, since its vertical profile is always above the terrain elevation profile. Likewise, all the restricted areas and weather obstacles fed to the SUT that are placed along the trajectory produced are retrieved and the trajectory is valid because its vertical profile avoids these.

Terrain and obstacle avoidance is verified until the Final Approach Fix. For the final approach, it is checked if the trajectory finishes on the runway threshold and if the glide slope and Instrument Landing System procedure are coherent.

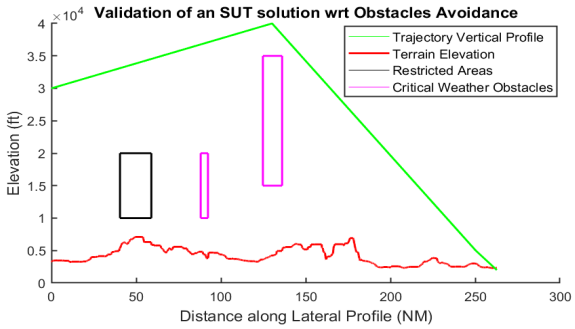

Figure 4: Validation of the safety of a trajectory.

3.4. Performance Metrics and Cost Functions

A diversion option is well selected if the trajectory generated by the SUT to reach it is shorter than the trajectory it would generate to reach any of the other options. Let n be the number of options in the list. The $n-1$ options not selected are sorted by increasing order of ground distance from the aircraft (so in index 1 is the closest alternative, and so on). Let l_{SUT} be the ground length of the trajectory produced to reach the option selected and l_i the ground length of the trajectory produced to reach option i (obtained by passing a list with only option i to the SUT). Cost function f_i defined by (5) evaluates the pertinence of the diversion selection.

$$f_i = \frac{l_{SUT}}{l_i} \tag{5}$$

The selection is not good in case $f_i > 1$ for any $i \in \{1, ..., n-1\}$. A search for sub-optimal trajectories could be guided by a maximization of l_{SUT}. However, this would favor scenarios where the aircraft is far from the airport and would not identify sub-optimal trajectories when the aircraft is closer to it. Let d_{SUT} be the ground distance between aircraft and runway. The search for sub-optimal trajectories could be guided by f_{SUT} defined by (6).

$$f_{SUT} = \frac{l_{SUT}}{d_{SUT}} \tag{6}$$

However, f_{SUT} might be large for optimal trajectories if obstacles need to be avoided or the aircraft is close to the runway at high altitude and needs to dissipate substantial energy. In future work, it could be interesting to design a cost function to guide the search based on energetic considerations.

Ideally, if another algorithm could compute optimal solutions (not necessarily in real-time) with ground length l_{opt}, cost function g defined by (7) could drive the search for non-optimal trajectories.

$$g = \frac{l_{SUT}}{l_{opt}} \tag{7}$$

A search for scenarios where a prototype a produces trajectories of length l_a longer than those produced by b of length l_b could be driven by h (8).

$$h = \frac{l_a}{l_b} \tag{8}$$

In case prototype a is an improved version of prototype b, h can drive a search for regressions, supporting the design process.

The SUT being real-time, the last term of the cost function that should drive the validation scenario choice is proportional to the execution time, t. The principle of h also allows to find scenarios in which a prototype is faster or slower than another.

4. Exploratory data analysis and preprocessing

4.1. Dataset - FlightRadar24

FlightRadar24 is a data source that compiles Automatic Dependent Surveillance-Broadcast (ADS-B) data from most aircraft worldwide. The flight recordings dataset used comes in two tables. The first one contains flight metadata, including parameters such as flight id, code of departure, scheduled and arrival airports or yet flight phase during which ADS-B transmission began and ended. The second one contains the time-series for each flight, including timestamp, altitude, heading, latitude, longitude, speed and vertical speed. The datasets used contain over 200 million flights between February 2014 and December 2018.

4.2. Diversions in the Dataset

Historical diversions can be identified (scheduled and arrival airports are different). Figure 5 presents diversions from LIS-MUC or MUC-LIS flights.

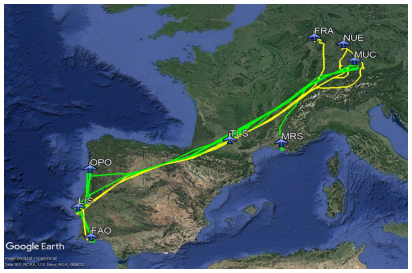

Figure 5: Diversions that occurred during LIS-MUC (in yellow) or MUC-LIS (in green) flights.

There were three different diversion moments: right after take-off (a probable cause is the aircraft having problems during take-off), during cruise (to LFBO and LFML, both a problem with the aircraft or a passenger feeling sick are plausible causes) and already close to the scheduled airport (severe weather conditions could be the cause). Regardless of the causes, we know which airports were chosen by pilots or airlines OCC when a diversion was necessary. They should therefore be introduced in the list of diversion options.

4.3. Data of Interest

For clustering purposes and to allow testing the SUT throughout whole flights, the trajectories used

should start before or during take-off and finish during or after landing, so that they are complete. The majority of the trajectories in the dataset is complete, so those that are not were filtered out. A minority of complete trajectories that do not start or finish exactly on a runway was also filtered out.

4.4. Trajectory Reconstruction

Trajectory reconstruction from historical recordings was formulated as an unconstrained optimization problem in Section 2.2, for which there is a closed form solution defined by (9).

$$P = \left[A^T A + \lambda_1 D_2^T D_2 + \lambda_2 D_3^T D_3\right]^{-1} A^T \widehat{P} \quad (9)$$

The reconstructed trajectory P is obtained from available measurements $\widehat{P}$ as a function of scalar regularization parameters λ_1 and λ_2, which are greater than 0 to be tuned. To understand their effect, a LIS-MUC trajectory was reconstructed using different values of λ_1 and λ_2. Figure 6 presents the lateral profile of reconstructed trajectories, zooming on the departure from Lisbon. Vertical profiles are more easily interpolated.

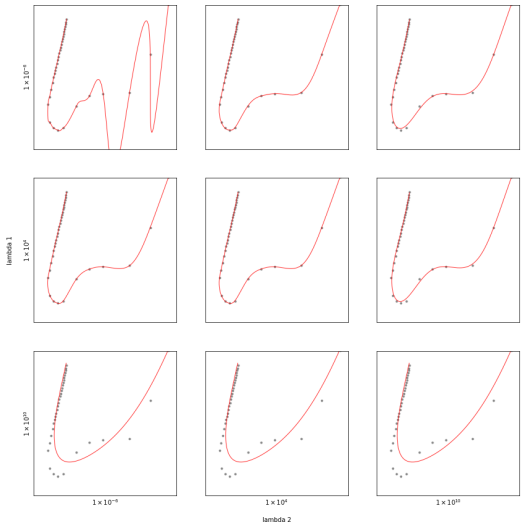

Figure 6: Effect of λ_1 and λ_2 on the reconstruction of smooth trajectories from historical measurements.

As Figure 6 shows, if regularization parameters are too small, rapid and aggressive maneuvers that are unrealistic from an aircraft dynamics point of view can exist to ensure a position error almost nonexistent. On the contrary, if they are too high, aircraft performances are too limited, not allowing the proper reconstruction of the actual aircraft trajectory neither. An out-of-sample-validation strategy (see Section 2.2) is adopted to find a balance.

5. Data-driven terrain representation validation

The SUT terrain representation must fully enclose the real terrain, so that any trajectory judged safe is so. This was verified by checking that each triplet of (latitude, longitude, elevation) values from the SRTM raster is considered to be inside or on the surface of the SUT inner terrain representation.

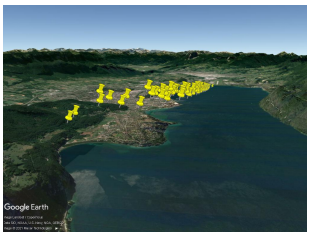

Figure 7: FlightRadar24 recordings considered to be inside the SUT inner terrain representation.

The representation cannot be too simplistic either, so that common aircraft positions are not considered as unsafe and the SUT can launch computations from them. An interface allows to communicate with the SUT software that determines the safety (or not) of a point or part of a trajectory. Therefore, a data-driven approach using FlightRadar24 data is followed to understand if common aircraft trajectories are accepted as safe.

Recordings were tested for collisions with the SUT terrain representation and points inside the latter were plotted on Google Earth (see Figure 7).

The representation needs to be more precise close to airports, since good part of departure and arrival procedures are declared as colliding with the terrain. This means that the SUT cannot produce emergency trajectories when flying these procedures for now, even though it should. Since emergencies can occur during these phases, accepting them as safe is an improvement required.

The height of collisions with respect to the relevant runway was computed and an histogram was plotted. From its analysis it was determined that the minimum height above the runway at which the terrain representation is already sufficiently precise regardless of the scenario is 5000 ft. Therefore, in this work the SUT can only be tested after reaching 5000 ft above the departure runway and until reaching 5000 ft above the arrival runway.

As the representation becomes more precise, trajectories should be considered instead of points. Consecutive points may be considered safe without the trajectory connecting them being so.

6. SUT Evaluation

To evaluate the SUT performance on the environment it was designed for, a data-driven strategy is proposed. Route trajectories are first clustered. Then, for each cluster a realistic and as optimal as possible diversion list is constructed and the SUT is evaluated, first under a nominal environment and then under a challenging one. A general search for challenging scenarios is also performed.

6.1. Commercial Route Trajectories Clustering

Two clustering algorithms are considered: HDBSCAN and Gaussian Mixture Model (GMM). They are evaluated on the LIS-MUC route.

HDBSCAN [19] is a density-based, hierarchical clustering method that provides a clustering hierarchy with all DBSCAN*-like solutions for an infinite range of density thresholds. Let $\mathrm{X} = \{\mathrm{x}_1, \cdots, \mathrm{x}_n\}$ be a dataset of n objects and $d(\mathrm{x}_p, \mathrm{x}_q)$ the distance between $\mathrm{x}_p, \mathrm{x}_q \in \mathrm{X}$. HDBSCAN defines density-based clusters based on *core objects*, accordingly to the definitions presented in [19], that follow.

Definition 1. (Core Distance): For an object $\mathbf{x}_p \in \mathbf{X}$ w.r.t. m_{pts}, $d_{\text{core}}(\mathbf{x}_p)$ is the distance from $\mathbf{x}_p$ to its m_{pts}-nearest neighbor (including $\mathbf{x}_p$).

Definition 2. (ε-Core Object) : An object $\mathrm{x}_p \in \mathbf{X}$ is ε-core for every ε such that $d_{\text{core}}(\mathbf{x}_p) \leq \varepsilon$.

Definition 3. (Mutual Reachability Distance): For $\mathrm{x}_q, \mathrm{x}_p \in X$, it is defined as $d_{\text{mreach}}(\mathbf{x}_p, \mathbf{x}_q) = \max\{d_{\text{core}}(\mathbf{x}_p), d_{\text{core}}(\mathbf{x}_q), d(\mathbf{x}_p, \mathbf{x}_q)\}$.

Definition 4. (Mutual Reachability Graph): Complete graph, $G_{m_{pts}}$, in which the objects of $\mathbf{X}$ are vertices and edge weights are mutual reachability distances between respective pairs of objects.

If graph $G_{m_{pts},\varepsilon} \subseteq G_{m_{pts}}$ results from removing all edges from $G_{m_{pts}}$ having weights greater than ε, then connected components of ε-core objects in $G_{m_{pts},\varepsilon}$ correspond to DBSCAN* clusters w.r.t. m_{pts} and ε. All DBSCAN* partitions for $\varepsilon \in [0, \infty)$ may be produced in a hierarchical way by removing edges in decreasing order of weight from $G_{m_{pts}}$.

A flat partitioning is obtained through local optimal cuts through the cluster tree (while DBSCAN uses a global density threshold) to maximize an overall cluster stability measure proposed in [19].

A Gaussian Mixture is a function containing K Gaussians, each defined by a mean μ_k, a covariance Σ_k defining the width and a mixing probability π_k defining the cluster frequency (probability of observing a sample from it) such that (10) holds.

$$\sum_{k=1}^{K} \pi_k = 1 \tag{10}$$

The problem is determining optimal values for π_k, μ_k and Σ_k. They correspond to the Maximum Likelihood Estimates of the Gaussian density function differentiated with respect to the mean and covariance and equalled to zero. Trajectories are assigned to each cluster with a certain probability. This soft clustering allows identifying outliers, unlike K-means. Regarding the choice of K, several models can be fit and the best one is chosen. This is called model selection. Two probabilistic model selection options based on performance and complexity used are AIC [20] and BIC [21].

Clusters from the LIS-MUC route obtained with HDBSCAN and GMM are presented in Figure 8.

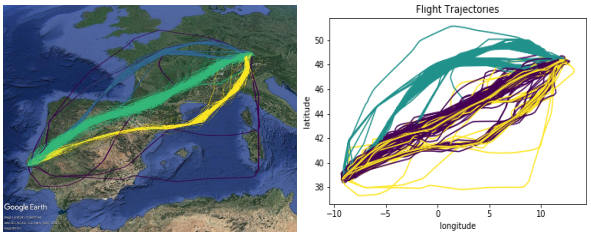

Figure 8: LIS-MUC clusters (yellow, green and blue) and noise (purple) found by HDBSCAN (left) and clusters (yellow, purple and green) found by GMM (right).

HDBSCAN identifies three clear distinct clusters, whereas two of the clusters identified by GMM are very intertwined. HDBSCAN is therefore chosen to be used henceforth for clustering route trajectories.

6.2. Diversion List Construction

To ensure that the approach of the procedure computed is safe, auto-land-compliant and accepted even under low visibility conditions, only runways with ILS category II or III means are considered.

There should always be a suitable diversion airport within a distance corresponding to one hour of flight. To be very conservative, a distance of 250 Nautical Miles (NM) is considered. To ensure this, each cluster is first represented by a set of points from a grid. Each recording is assigned to the grid point it is closest to and only grid points with at least one recording assigned to them are kept. Cluster representative points are illustrated in Figure 9.

Airports selected for diversion while flying the route of interest in the past are first inserted in the lists of pertinent clusters. Determining the minimal number of airports to ensure all uncovered cluster representative point are within 250 NM of at least one of them is formulated as a set cover problem.

Set Cover Problem. Given universe U, a collection of subsets of U, $S = \{S_1, ..., S_k\}$, and a cost function $c : S \longrightarrow \mathbf{Q}^+$, find a minimum cost subcollection of S that covers all elements of U [22].

Universe U consists of uncovered cluster representative points. Each set S_i contains points from U within 250 NM of airport i. Airports farther than 250 NM from any points from U are disregarded. Among the approaches in the literature to solve the problem, defining it as an Integer Linear Program was chosen. A variable x_{S_i} is assigned for each set $S_i \in S$. This variable is allowed 0 (set not picked) or 1 (set picked) values. For each element $e \in U$ at least one of the sets containing it must be picked. The problem is formulated by (11).

$$\begin{aligned} \min_{x_{S_i}} \quad & \sum_{S_i \in S} x_{S_i} \\ \text{subject to} \quad & \begin{cases} \sum_{S_i : e \in S_i} x_{S_i} \geq 1, & e \in U \\ x_{S_i} \in \{0,1\}, & S_i \in S \end{cases} \end{aligned} \tag{11}$$

Figure 9 presents the construction of the diversion list for one LIS-MUC cluster.

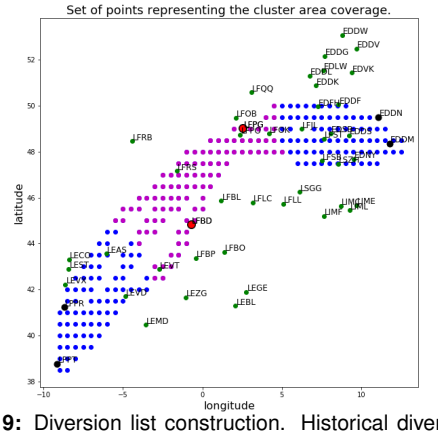

Figure 9: Diversion list construction. Historical diversion airports are in black. Cluster representative points less than 250 NM from these are in blue and those not yet covered by any airport in magenta. The two airports selected are in red. Other airports from the database are in green.

6.3. Evaluation of the SUT on Commercial Routes

6.3.1 Nominal Environment

To ensure cluster coverage, clusters are represented by a group of hyper-cubes, by creating a grid in the search space and keeping cells containing recordings. Figure 10 shows the latitude-longitude representation of the LIS-MUC route.

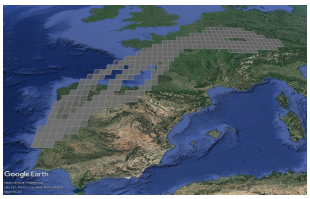

Figure 10: Cells representing the LIS-MUC route.

The diversion list corresponding to a cluster is selected and n random samples are taken from each cell and fed to the SUT. The cost functions proposed in Section 3.4 are then computed.

The execution time was very low, with a mean under 0.01 seconds and maximum under 0.1 seconds. f_{SUT} was judged a poor cost function to search for sub-optimal trajectories, since when the aircraft is close to the target at high altitudes f_{SUT} is large even if the aircraft dissipates its energy in an optimal manner. Should an alternative prototype be available, cost function g would be more suitable to guide a search for sub-optimal trajectories. For the moment, a visual analysis of trajectories with different f_{SUT} values does not allow to find sub-optimal trajectories from a length point of view under a nominal environment. Scenarios with a high value of f_1 allowed to find sub-optimal diversion selections, as illustrated in Figure 11.

Figure 11 shows scenarios where the diversion option chosen is slightly closer to the aircraft than the alternative but the trajectory to reach the alternative is shorter due to the heading. It is also more intuitive to keep the heading rather than turn back.

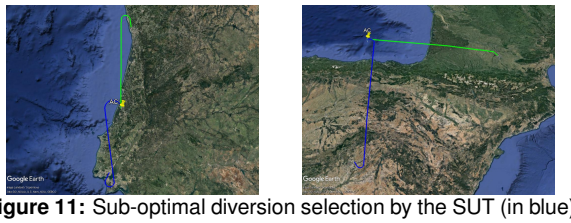

Figure 11: Sub-optimal diversion selection by the SUT (in blue) vs optimal selection of diversion option 1 (in green).

6.3.2 Challenging Environment Search

This section addresses the search for challenging environments created from the libraries presented in Section 3.2 when the aircraft state belongs to a given cell. An optimization approach is suitable for this. To define the aircraft and environment states, the optimization variable, $\mathbf{x}$, is defined by (12).

$$\mathbf{x} = \left[x_{AC}, n_{restr}, n_{wx}, x_{wx}, \theta_{wx}\right]^\top \tag{12a}$$

$$x_{AC} = \left[lat, lon, alt, hdg, spd, vSpd\right]^\top_{AC} \tag{12b}$$

$$n_{restr} = \left[nr_1, nr_2, nr_3, nr_4\right]^\top \tag{12c}$$

$$n_{wx} = \left[nwx_1, nwx_2, nwx_3, nwx_4\right]^\top \tag{12d}$$

$$x_{wx} = \left[lat_1, lon_1, lat_2, lon_2, lat_3, lon_3, lat_4, lon_4\right]^\top \tag{12e}$$

$$\theta_{wx} = \left[\theta_1, \theta_2, \theta_3, \theta_4\right]^\top \tag{12f}$$

where x_{AC} defines the aircraft state, n_{restr} is an array of integers indicating which restricted areas are selected, n_{wx} is an array of integers indicating which critical weather obstacles are selected, x_{wx} defines their location and θ_{wx} their orientation.

It was a choice to only allow the selection of a maximum of 4 restricted areas and 4 weather obstacles. An environment filled with obstacles would obviously be more challenging, but less realistic.

To tackle the optimization problem, a Genetic Algorithm approach was chosen due to its success in the past for several applications. An initial population is generated randomly or heuristically. It evolves iteratively thanks to evolution-theory-inspired operators, which favor the survival of the fittest. On each iteration, individuals are evaluated through a fitness function and in the selection phase those with a higher fitness are more likely to be chosen for reproduction. A new population is generated from the selected individuals thanks to crossover and mutation operations. Crossover is a re-combination operator, taking two parents and swapping some of their genes to produce two children. It favors exploitation, biasing the search towards promising search space regions. Mutation favors exploration, by randomly sampling new points from the search space. Hence, preventing premature convergence to local optima (using a population also prevents this). The algorithm provides a set of solutions. It is suitable for global optimization of black-box functions that may or not be

continuous, since no local gradient information is required, only the ability to compute fitness values. The main risk is not choosing well the parameters.

The search could quickly and successfully find challenging scenarios with the aircraft restrained to a given cell. Figure 12 presents two of them.

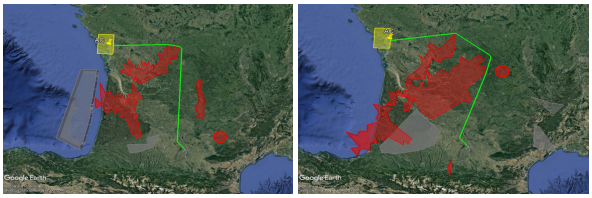

Figure 12: Challenging scenarios found when searching for maximum execution time (left image) and maximum f_{SUT} (right image) values from a given aircraft cell (in yellow). Weather is represented in red and restricted areas in grey.

Obstacles are placed between the aircraft and the diversion option, obliging the trajectory to go around these. The orientation is such that most concavities face the aircraft. The design team was informed of this to support future improvements.

6.4. Worst Scenario Search

The genetic algorithm presented in Section 6.3.2 can be used to find the most challenging scenario to reach a fixed diversion option (constraining aircraft positions only to be within 250 NM of it) or scenarios where the diversion selection is the worse possible (given a list of diversion options and not constraining the aircraft state to a cell). Figure 13 represents two of the scenarios successfully constructed during a quick search.

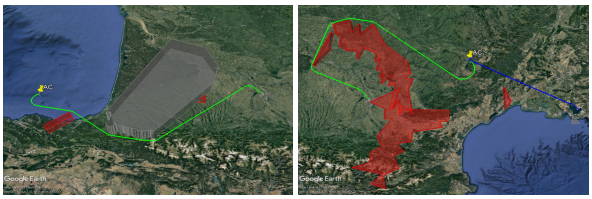

Figure 13: Challenging scenarios. Forcing a diversion to LFBO, it took the SUT 80.56 seconds to produce the solution for the left scenario. Searching for sub-optimal diversion selections the scenario on the right was constructed. The trajectory produced to reach the SUT choice (in green) is 2.53 times longer than the one produced to reach diversion option 1 (in blue).

From the left scenario, it is clear that dealing with a restricted area defined by many concavities facing the aircraft is very challenging. From the right scenario, it seems like the environment between the aircraft and airports is not taken into account when selecting the most suitable diversion option, yielding a trajectory much longer than needed.

7. Conclusions and Future Work

This paper proposed a complete framework to support the development and validation of a real-time black-box system for automated selection of the most suitable diversion option from a list and generation of a safe and flyable trajectory to reach it and save an aircraft in case of emergency. To ensure that test scenarios are representative, they are constructed from historical aircraft and weather data, as well as real restricted areas. The framework was successfully tested on a prototype under development, identifying axis of improvement.

Smooth and complete trajectory reconstruction from FlightRadar24 data was formulated as a convex optimization problem, showing good performance. A data-driven strategy was proposed to evaluate the SUT's internal representation of the terrain. It allowed to verify that the representation fully encloses the real terrain but also understand that it should be more precise close to airports. In fact, many recordings belonging to departure and arrival procedures are considered not to be safe, not allowing to launch the SUT from them. In future work, as the representation becomes more mature, reconstructed trajectories should be tested instead of single positions. Additionally, it is important to take into account aircraft navigation performances.

The capacity to cluster trajectories from a route was required by the strategy designed to evaluate the SUT performance. HDBSCAN and Gaussian Mixture Models were implemented and their performance tested on a use case, with the former showing significantly better performance than the latter. HDBSCAN was therefore chosen.

Airports historically chosen (by pilots or airlines OCC) for diversion were introduced in the diversion list of pertinent clusters. In a second phase, Integer Linear Programming was successfully used to solve a set cover problem and ensure that anywhere on the cluster a diversion option exists within 1 hour of flight. Since there are usually several solutions, it would be interesting to introduce tie-breakers such as the airport capacity or existence of emergency response means in future work.

Cost functions were proposed to evaluate the SUT performance and compare it with that of alternative prototypes. Further improving them would be interesting. To evaluate the performance on a cluster, the cluster is represented by a group of cells and scenarios are sampled from each cell to ensure cluster coverage. The SUT has shown very good performance on nominal environments. However, an axis of improvement was identified. When the aircraft is between two airports the diversion selection is not optimal when the aircraft heading favors the solution not chosen by the system.

A genetic algorithm was proposed to efficiently search for challenging environments. To ensure they are representative, real restricted areas were retrieved and a library of historical critical weather shapes was constructed. The algorithm selects which areas and weather obstacles are to be considered and also the placement and orientation

of the weather selected. It has successfully constructed challenging environments, mainly by selecting complex shapes and placing them between the aircraft and airport. It was found that the diversion selection does not take into account the environment very well. Sometimes the diversion option selected is closer to the aircraft but requires it to go around several obstacles, yielding a solution considerably longer than the one produced to reach an alternative from the list. The algorithm could be improved with problem-specific crossover and mutation operators. It could also be interesting to allow it to define the actual shapes of obstacles instead of selecting them from libraries.

The framework will once again be useful to test future versions and identify potential regressions.

The SUT was only tested with static inputs so far. In future work, its capacity to react to disturbances while flying the computed trajectory must be evaluated. The background presented may support the design and implementation of strategies to discover sequences of disturbances leading to failure. MCTS could be an interesting approach. Computing error probabilities is another interesting stream of work. At last, it is important to develop approaches allowing to better understand how the system works to build confidence on it.

References

[1] IATA. IATA Forecast Predicts 8.2 billion Air Travelers in 2037. *Pressroom*, 62, October 2018.

[2] IATA. Recovery Delayed as International Travel Remains Locked Down. *Pressroom*, 63, July 2020.

[3] Fallast, A., Messnarz, B. Automated trajectory generation and airport selection for an emergency landing procedure of a CS23 aircraft. *CEAS Aeronaut J*, 8:481–492, June 2017.

[4] Meuleau et al. An Emergency Landing Planner for Damaged Aircraft. *Twenty-First Conference on Innovative Applications of Artificial Intelligence*, July 2009.

[5] Zhao et al. Generating test inputs for embedded control systems. *IEEE Control Systems Magazine*, 23(4):49–57, August 2003.

[6] Zhang et al. Two-Layered Falsification of Hybrid Systems Guided by Monte Carlo Tree Search. *CoRR*, August 2018.

[7] Lee et al. Adaptive stress testing of airborne collision avoidance systems. *2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC)*, pages 6C2–1–6C2–13, 2015.

[8] C. B. Browne et al. A Survey of Monte Carlo Tree Search Methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43, March 2012.

[9] Corso et al. A Survey of Algorithms for Black-Box Safety Validation. *CoRR*, May 2020.

[10] Mullins et al. Automated generation of diverse and challenging scenarios for test and evaluation of autonomous vehicles. *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1443–1450, 2017.

[11] Barratt et al. Learning Probabilistic Trajectory Models of Aircraft in Terminal Airspace From Position Data. *IEEE Transactions on Intelligent Transportation Systems*, 20(9):3536–3545, sep 2019.

[12] Han et al. *Data mining: concepts and techniques*. Morgan Kaufmann Publishers, San Francisco, 3rd edition, 2011.

[13] Ester et al. A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[14] Christopher M. Bishop. Pattern Recognition and Machine Learning. *Sprinter-Verlag*, 2006.

[15] Yuan et al. A review of moving object trajectory clustering algorithms. *Artificial Intelligence Review*, 47:123–144, January 2017.

[16] S. Hong and K. Lee. Trajectory prediction for vectored area navigation arrivals. *Journal of Aerospace Information Systems*, March 2015.

[17] Gariel et al. Trajectory Clustering and an Application to Airspace Monitoring. *IEEE Transactions on Intelligent Transportation Systems*, 12(4):1511–1524, December 2011.

[18] NASA. Shuttle Radar Topography Mission. https://www2.jpl.nasa.gov/srtm/.

[19] Campello et al. Density-based clustering based on hierarchical density estimates. *PAKDD 2013: Advances in Knowledge Discovery and Data Mining*.

[20] Hirotugu Akaike. Information theory and an extension of the maximum likelihood principle. *Second International Symposium on Information Theory*, pages 267–281, 1973.

[21] G. Schwarz. Estimating the Dimension of a Model. *Ann. Statist.*, 6(2):461 – 464, 1978.

[22] Vijay V. Vazirani. *Approximation Algorithms*. Springer-Verlag, 2003.