Predicting Aircraft Trajectories via Imitation Learning

Theocharis Kravaris
University of Piraeus
Piraeus, Greece
hariskrav.unipi@gmail.com

Alevizos Bastas
University of Piraeus
Piraeus, Greece
bastas.a@gmail.com

George A. Vouros
University of Piraeus
Piraeus, Greece
georgev@unipi.gr

ABSTRACT

The current Air Traffic Management (ATM) system worldwide - something to be continued in the post Covid-19 era- has reached its limits in terms of efficiency and cost effectiveness. Different initiatives worldwide propose trajectory-oriented transformations that require high fidelity aircraft trajectory prediction capabilities. Recently proposed data-driven trajectory prediction approaches provide promising results, albeit with important limitations. In this paper we propose approaching the data-driven trajectory prediction problem as an imitation learning task: We aim to imitate experts with conflicting objectives "shaping" the trajectory. Towards this goal we are formulating the trajectory prediction as an imitation task and we are using the Generative Adversarial Imitation Learning method to predict trajectories. This approach, compared to other state of the art approaches to trajectory prediction, can provide very accurate results for large look-ahead time predictions, both at the pre-tactical and at the tactical operation stages.

KEYWORDS

Imitation Learning, Inverse Reinforcement Learning, Trajectory Prediction, Aircraft Trajectory

1 INTRODUCTION

Nowadays, the Air Traffic Management (ATM) system is based on an airspace management paradigm that introduces uncertainties which impact multiple stakeholders, including airspace users (i.e.airlines), the air navigation service providers (ANSPs, providing services for Air Traffic Management), the air traffic controllers, as well as ground operators, and of course, passengers. Confronting and adopting to uncertainties is costly for all and with cascaded effects to the whole ATM system. With the aim of overcoming ATM system drawbacks, different initiatives, dominated by Single European Sky ATM Research (SESAR) in Europe and Next Gen in the US, have promoted the transformation of the current environment towards a new trajectory based ATM paradigm. In this Trajectory Based Operations (TBO) paradigm the trajectory becomes the cornerstone upon which all the ATM capabilities will rely on, supporting the whole trajectory life cycle: From the trajectory planning, to the negotiation and agreement, execution, amendment and modification stages.

The proposed transformation requires high fidelity aircraft trajectory prediction capabilities, supporting the trajectory life cycle at all stages efficiently. Indeed, predictability is considered as the main driver to enhance operational key performance areas, such as capacity of the airspace, effectiveness regarding all stakeholders' objectives, and, of course, safety. Bringing more automation into all stages of operations by enhancing predictability, emerges both, as a mid-term need (the European Network Manager EUROCONTROL, forecasts increases in traffic of +50% in 2035 compared to 2017, meaning 16 million flights across Europe) and as a long-term need (2035+).

Current trajectory predictors are based on deterministic (mechanistic) formulations of the aircraft motion problem. These sophisticated solutions are intrinsic simplifications to the actual aircraft behavior, incurring a computational cost. Predictors' outputs are generated based on a-priory knowledge of the flight plan (i.e. airline's planned and intended trajectory), the expected command and control strategies released by the pilot, or the Flight Management System (FMS) instructions (known as Aircraft Intent [20]), and the aircraft performance. Aircraft Intent together with very precise aircraft performance models such as Base of Aircraft Data (BADA) [1] has helped to improve the prediction accuracy. These mechanistic, model-based approaches require a set of parameters that typically are not precisely known (i.e. initial aircraft weight, pilot/FMS flight modes, etc.) at prediction time, given that these include airspace users (AU, i.e. airlines) business sensitive information. Therefore, accuracy is reduced considerably beyond a limited prediction horizon (look-ahead time) of approx. 10 minutes. Adding the fact that these solutions are not able to predict how the trajectory is finally "shaped", based on evidence from past flights, showing trends, preferences and strategies of stakeholders, they can be of limited use towards implementing the TBO paradigm.

Recent efforts in the field of aircraft trajectory prediction have explored the application of statistical analysis and machine learning techniques. Linear regression models [17] [13] and neural networks [18] [5] [19], have returned successful outcomes for improving the trajectory prediction accuracy for traffic flow forecasting. Generalized Linear Models [6] have been applied for the trajectory prediction in arrival management scenarios and multiple linear regression [29] [15] for predicting estimated times of arrival (ETA). These efforts include as input historical surveillance data, and they usually require additional supporting data for accurate trajectory predictions (e.g. flight plans, airspace structure, Air Traffic Control procedures, airline strategy, weather forecasts, etc.), depending on their objectives. However, these approaches are either restricted to a specific operational/tactical phase, have a limited prediction horizon, consider specific constraints for trajectories (e.g. flight plans), and typically need large amounts of historical data with excessive training requirements. Furthermore, it is important to note that the lack of benchmarks hinders the systematic comparison of different trajectory prediction methods. In addition to that, we cannot directly compare predictions made for trajectories in Europe, or for different routes in Europe, with those in USA, without considering how predictions are affected by training data, due to the different operational concerns.

In this paper we approach trajectory prediction as a data-driven imitation learning problem. Aiming to imitate the experts "shaping"

trajectories by learning policy models that incorporate their preferences, strategies, practices etc. in an aggregated way, we study the use of Generative Adversarial Imitation Learning (GAIL) [14] state of the art method, using a critic method for estimating the state advantage instead of estimating Q, directly. Results show that GAIL can be effective (in terms of accuracy of predictions) even with a small number of historical trajectories, and can provide accurate long-term predictions, compared to state of the art approaches.

Major contributions that this paper makes are as follows:

(a) It formalizes the trajectory prediction problem as an imitation learning process, given a set of historical trajectories provided as "expert" demonstrations, considering that these trajectories have been "shaped" by aggregating individual stakeholders' policies, preferences and objectives.

(b) It studies the use of state of the art imitation learning methods, which are able to learn trajectory models without making any assumption on the form of a cost function, in continuous state-action spaces, with no specific requirements on specifying trajectory constraints, and with minimal data pre-processing requirements.

(c) It provides extensive experimental results that concern trajectories between Origin-Destination (OD) airports' pairs with different characteristics, demonstrating the far in the future prediction abilities of the method, either at the pre-tactical or at the tactical stage of operations, as well as its abilities to generalize beyond specific OD pairs.

The structure of the paper is as follows: Section 2 provides background knowledge on trajectories, it specifies the trajectory prediction problem, and provides background knowledge on imitation learning. Section 3 formalizes the data-driven aircraft trajectory prediction problem as an imitation learning problem, Section 4 presents the proposed method using GAIL, and Section 5 presents experimental results. Finally, Section 6 presents related work and discusses the effectiveness of the proposed method w.r.t. state of the art aircraft' trajectory prediction methods. Section 7 concludes the paper.

2 BACKGROUND KNOWLEDGE

2.1 Aircraft Trajectory Prediction

In the aviation domain, a trajectory is defined as the description of movement of an aircraft both in the air and on the ground.[1]. This description can be provided by a chronologically ordered sequence of aircraft states. Most relevant state variables are airspeeds, 3D position (determined by latitude (f), longitude (l) and geodetic altitude (h)), the bearing (c) or heading (y) and the instantaneous aircraft mass (m).

Following a data-driven approach, we aim to exploit historical 4D aircraft trajectories whose states include 3D aircraft position with timestamps, in conjunction to contextual information providing useful features in the prediction process, such as weather conditions at each state, traffic, special events occurring etc. Adding variables in a trajectory state results in a trajectory with *enriched points* or *enriched states*, thus to an *enriched trajectory*:

An *enriched trajectory state* or *enriched trajectory point* of a trajectory of length $|T|$, is defined to be a triplet $s_{r,i}$=<p_i, t_i, v_i>, where p_i is a point in the 3D space, v_i is a vector consisting of categorical and/or numerical variables and t_i is a timestamp, with $i \in [0, |T|-1]$. An *enriched trajectory* T is defined to be a sequence of enriched states $s_{r,i}$=<p_i, t_i, v_i>, $i \in [0, |T|-1]$.

As already pointed out in the introductory section, uncertainties occurring during a flight have impact on multiple stakeholders, with costly cascaded effects to the whole ATM system. For instance, these may require assigning delays to flights, or choosing alternative routes to those planned for a flight, or scheduling operation with enough slack time to absorb unpredictable events. Therefore, accurate trajectory predictions are of immense importance.

A *predicted trajectory* can be defined as the future evolution of the aircraft state as a function of (a) the current flight conditions (e.g. a given state), (b) a forecast of contextual features (e.g. of weather conditions) and (c) a "policy" specifying how the aircraft intends to transit among subsequent states.

Casting the trajectory prediction to a data-driven problem, and assuming a set $\mathbf{T}_E = \{T_{E,i} | i = 1, 2, 3, ...\}$ of historical, demonstrated enriched trajectories, the *trajectory prediction problem* can be defined as follows: Given $\mathbf{T}_E$ and a cost function c, the objective is to predict a trajectory T_π, such that

$$T_\pi = \underset{\pi}{\operatorname{argmin}}\ \mathbb{E}_\pi[c(\langle p, t, v\rangle, a)] \tag{1}$$

where $\mathbb{E}$ denotes the expected cumulative costs for all states $s = \langle p, t, v\rangle$ along the predicted trajectory by following a policy $\pi(a|s)$, prescribing the probability of applying an action a at state s. Actually, according to equation 1, the ultimate objective is to find the policy π that determines the generation of a minimal-expected-cumulative-cost predicted trajectory T_π.

The cost function may take several forms depending on how the problem is approached: For instance, considering specific trajectories (e.g. flight plans, or cluster medoids) as constraints (e.g. as in [11]), and measuring the adherence of predictions to these constraints, the cost function may take the form of a distance function between these trajectories and predicted trajectories. Generally, in a data-driven trajectory prediction process, the cost function measures the adherence of predictions to given trajectories (e.g. those provided as constraints, or those demonstrated, i.e. those generated by an expert policy). We delve into this issue further while formulating the trajectory prediction problem as an imitation process, in Section 3.

Furthermore, formula 1 includes the trajectory enriched states and actions performed: The formulation indicates separately the 4D position information with timestamps and other variables enriching states. Indeed, additional features may be considered in the cost function, such as weather variables, traffic, airspaces crossed, etc. Also, different prediction processes may have different prediction objectives: For instance, one may predict the aircraft position at specific time instances, or predict the time instance that a specific position will be reached, or the position together with the corresponding timestamp, or even predict some of the contextual features, such as airspaces crossed at specific time instances. What we aim to predict in this work is the 3D aircraft position at specific time instances, given forecasts for contextual features. Actions executed at each state determine how the aircraft intends to evolve

[1] https://ext.eurocontrol.int/lexicon/index.php/Trajectory

its trajectory towards the next state. The actions set A may vary between different approaches.

2.2 Imitation Learning

We aim to apply an imitation learning method towards learning a policy that models the way stakeholders shape the evolution of trajectories w.r.t. contextual conditions (e.g. weather conditions), considering historical enriched trajectories as experts' demonstrations.

There are two fundamental approaches to imitation learning: Behavioral Cloning (BC) and Inverse Reinforcement Learning (IRL).

Behavioral Cloning [22] solves a regression problem minimizing the prediction error over the states of the historical trajectories. This technique suffers from compounding errors and a regret bound that grows quadratically in the time horizon of the task leading to poor performance [24, 25].

Inverse Reinforcement Learning [2, 9, 10, 32] on the other hand, aims at deriving a cost function that assigns minimal cost to trajectories demonstrated by experts and maximal cost to trajectories generated by other policies. Given that many policies may demonstrate the same trajectories, we can aim to find the maximum entropy policy [32]. Actually, this process comprises two steps:

The second step uses the cost function derived from the first step into a standard reinforcement learning problem, to find a policy that minimizes the expected cumulative cost, maximizing the entropy, very closely to the one specified by equation (1):

$$RL(c) = \underset{\pi \in \Pi}{\operatorname{argmin}} -\lambda_H H(\pi) + \mathbb{E}_\pi[c(s,a)] \tag{2}$$

where, Π is the set of all policies, $H(\pi)$ the entropy function and λ_H its weighting parameter.

Specific instances of this process result into apprenticeship learning methods (e.g. that in reference [2]), assuming that the cost function is given by a linear combination of basis functions, which result to feature vectors over states and actions. Making assumptions on, or handcrafting the cost function is crucial to the prediction process, as flown trajectories are shaped by several stakeholders each with own preferences and objectives. Thus, while the linearity assumption can be restrictive for making predictions in complex settings, the hand-crafted state features are a big engineering burden. Moreover, an apprenticeship learning method is computationally expensive, as it runs a Reinforcement Learning algorithm at every cost function update, to find a policy that performs optimally w.r.t. the learnt cost function.

To address linearity limitations and avoid hand-crafted state features, the Guided Cost Learning approach [10] uses neural networks to represent the cost function, and provides a more computationally efficient approach, by applying a single gradient step for each new update of the cost function. However, as mentioned in [9], Guided Cost Learning is a specific instantiation of the Generative Adversarial Imitation Learning (GAIL) framework [14].

GAIL [14] learns the optimal policy from expert demonstrations directly, quite efficiently, scaling to large, continuous state-action spaces, and does not need to derive a cost function. Actually, it aims to bring the distribution of the state-action pairs of the imitator as close as possible to that of the expert. GAIL uses an architecture similar to Generative Adversarial Networks to optimize the following objective:

$$\min_{\pi} \max_{D \in (0,1)^{SxA}} \mathbb{E}_\pi[logD(s,a)] + \mathbb{E}_{\pi_E}[log(1 - D(s,a))] - \lambda_H H(\pi) \tag{3}$$

where π is the imitator policy, π_E is the expert's policy, D is a classifier called discriminator which distinguishes state-action pairs generated from π and π_E. $H(\pi) \triangleq \mathbb{E}_\pi[-log\pi(a|s)]$ is the γ-discounted causal entropy of the policy π. As shown in [14], equation (3) provides a way to solve the two-steps imitation process described earlier.

To predict aircraft trajectories via imitation learning, we are using the GAIL framework. The GAIL implementation to address the specific problem is described after the trajectory prediction specification problem given in the next Section.

3 PROBLEM SPECIFICATION

Given the abstract specification of the data-driven trajectory prediction problem in Section 2.1, and the formulation of the imitation learning problem provided in Section 2.2, we can provide a formulation of the *data-driven aircraft trajectory prediction problem as an imitation learning task.*

Let us assume a set $\mathbf{T}_E = \{T_{E,i}, i = 1, ...N\}$ of historical, enriched aircraft trajectories generated by an expert policy π_E. These trajectories have various number of states, and therefore, various lengths $|T_{E,i}|$. The objective is to find a policy that minimizes the difference between the expected cumulative cost of the predicted trajectories and of the trajectories in $\mathbf{T}_E$, given an approximation of the cost function that penalizes any state-action pair generated by any policy in $\Pi - \{\pi_E\}$. As shown in [14], this objective is equivalent to finding a policy π that brings the distribution of the state-action pairs generated by it, as close as possible to the distribution of the state-action pairs demonstrated by trajectories in $\mathbf{T}_E$.

As pointed out in Section 2.1, in this work we aim to predict the 3D aircraft position at specific time instants, given an initial state at time instant t_0: Specifically, we aim at determining the evolution of the trajectory in space every Δt seconds, i.e. at time instances $t_i = t_0 + (\Delta t * i)$, $i = 1, 2, 3...$, given the position of the aircraft at time instance t_0.

Specifically, the *data-driven aircraft trajectory prediction problem as an imitation learning task* is specified as follows: Given a set $\mathbf{T}_E = \{T_{E,i}, i = 1, ...N\}$ of historical aircraft trajectories, and a time step Δt, we need to determine a policy $\pi \in \Pi$ which, given the initial state of aircraft s_0, optimizes the GAIL objective at any time instant $t = t_0 + (\Delta t * i)$, $i = 1, 2, 3....$

A crucial decision concerns the set A of actions considered, which should adequately and unambiguously (although, in a non-deterministic way) specify the evolution of the trajectories. In our approach, and very close to the General Adversarial Imitation from Observations approach described in [30], we are motivated to focus on states, rather than on actions that determine the movement of the aircraft from state to state. This approach is motivated by considering the following: (a) Expert trajectories do not specify in any way the actions applied in any state and thus, these have to be determined a posteriori under specific assumptions that may bring noise into the learning process; (b) there are several possibilities of instruction combinations for evolving the aircraft state, at

different levels of detail, which result in a high-dimensional state-action space, and which require considering constraints between instructions combinations; (c) what we aim to actually predict is the evolution of aircraft states in the 4D space (i.e. regarding position and time); and (d) the imitation learning approach that we take aims to bring the distribution of state-action pairs of the imitator close to the corresponding distribution of the expert.

We consider that the set A contains all the possible combinations of differences in all 3 spatial dimensions between subsequent trajectory states' position information, given the constraint that each difference must be feasible within the constant Δt period considered, w.r.t. aircraft capabilities (e.g. maximum speed). Specifically, we consider an action set, depending on how the position information is represented: Given a position in terms of longitude, latitude and altitude (l, f, h), actions take the form of $(\Delta l, \Delta f, \Delta h)$, and the position in the next state is determined by $(l + \Delta l, f + \Delta f, h + \Delta h)$.

Indeed, these actions can be determined by the demonstrated trajectories unambiguously and very efficiently, although in low-quality surveillance data space-time constraints concerning the evolution of aircraft states may be violated. This action set has three additional important effects: (a) We can tune the resolution of the predicted trajectory by changing the Δt. (b) Given a specific Δt (e.g. 5 seconds), and the evolution of the trajectory until reaching the destination airport, we can determine the estimated time of arrival (ETA), which is simply ($\Delta t * |T_\pi|$), given the predicted trajectory T_π. (c) The transition between subsequent positions is deterministic given the first state and an action.

4 GAIL: LEARNING TO IMITATE TRAJECTORIES

GAIL employs a generative trajectory model G that models π and a discriminative classifier D that distinguishes between the distribution of state action pairs generated by π and the demonstrated data. Both π and D are represented by function approximators with weights θ and w, respectively. Following the implementation described in [14], GAIL alternates between an Adam [16] gradient step on w to increase the objective function stated in equation (3) with respect to D, and a step on θ using the Trust Region Policy Optimization (TRPO) algorithm [26] to decrease the objective function (3). TRPO optimizes the following objective:

$$\begin{aligned} &\underset{\theta}{minimize}\mathbb{E}_{s\sim\rho_{\theta_{old}},a\sim q}[\frac{\pi_\theta(a|s)}{q(a|s)}Q_{\theta_{old}}(a|s)] \\ &subject\ to\ \mathbb{E}_{s\sim\rho_{\theta old}}[D_{KL}(\pi_{\theta_{old}}(\cdot|s)\|\pi_\theta(\cdot|s))] \leq \delta \end{aligned} \tag{4}$$

where $\rho_{\theta_{old}}$ is the distribution of states generated using the prior-to-update (old) policy $\pi_{\theta_{old}}$, q is an action sampling distribution that we consider equal to $\pi_{\theta_{old}}$, π_θ is the updated policy with parameters θ, $Q_{\theta_{old}}$ is the state-action value function of the old policy and δ is a constant that constraints the KL divergence between $\pi_{\theta_{old}}$ and π_θ, preventing the policy from changing too much due to noise in the policy gradient. We solve the TRPO optimization problem as described in [26] Appendix C, using the conjugate gradient method and a line search. In our setting we set $\lambda_H = 0$, so we omit $-\lambda_H H(\pi)$ from the equation (3), following the practice demonstrated in [14].

The implemented method, instead of approximating Q, utilizes a separate critic model to approximate the state advantage defined as $A_t = A(s_t, a_t|\pi) = Q(s_t, \pi(s_t)) - V^\pi(s_t)$, aiming to lower the gradient variance. We follow the Generalized Advantage Estimation (GAE) approach introduced in [27], which provides a balance between low variance and a small amount of bias introduced. Formally, we estimate the advantage from the sampled state-action pairs as follows:

$$\hat{A}_t^{GAE(\gamma,\lambda)} := (1-\lambda)(\hat{A}_t^{(1)} + \lambda\hat{A}_t^{(2)} + \lambda^2\hat{A}_t^{(3)} + \dots) \tag{5}$$

where $\gamma \in [0, 1]$ is the discounting factor, λ a hyper-parameter and

$$\hat{A}_t^{(k)} := -V(s_t) + r_t + \gamma r_{t+1} + \cdots + \gamma^{k-1} r_{t+k-1} + \gamma^k V(s_{t+k}) \tag{6}$$

Algorithm 1 shows the aforementioned procedure in more detail. Specifically, we pre-train G using Behavioral Cloning. Then, at each GAIL iteration, the algorithm samples from the initial state distribution and generates roll-out trajectories. It uses the generated state-action samples and the samples of the historical trajectories to update the D parameters w. D is updated with cross entropy loss that pushes the output for the demonstrated state-action samples closer to 0 and π_θ state-action samples closer to 1. Next, the imitation algorithm takes a policy step using the TRPO [26] update rule and $logD(s, a)$ as the cost function approximation to update θ. It must be noted that the t parameter in the denotation of the approximation of the state advantage in Algorithm 1 is left implicit, for simplicity of the presentation.

Algorithm 1 GAIL

1: **Input:** Expert trajectories $\tau_E \sim \pi_E$, initial policy π_{θ_0} and discriminator parameters w_0
2: **Output:** Policy π_θ
3: Initialize policy using Behavioral Cloning.
4: **for** i=0,1,2,... **do**
5: Sample trajectories $\tau_i \sim \pi_{\theta_i}$
6: Update D parameters w with the gradient
7: $\hat{\mathbb{E}}_{\tau_i}[\nabla_w log(D_w(s,a))] + \hat{\mathbb{E}}_{\tau_E}[\nabla_w log(1 - D_w(s,a))]$
8: Estimate advantages $\hat{A}^{GAE(\gamma,\lambda)}$, according to $\pi_{\theta_{old}}$
9: Take a policy step using the TRPO rule with cost function $log(D_w(s,a))$:
Take a KL-constrained natural gradient step with $\hat{\mathbb{E}}_{\tau_i}[\nabla_\theta \frac{\pi_\theta(a|s)}{\pi_{\theta_{old}}(a|s)}\hat{A}^{GAE(\gamma,\lambda)}]$ subject to $\mathbb{E}_{s\sim\rho_{\theta old}}[D_{KL}(\pi_{\theta_{old}}(\cdot|s)\|\pi_\theta(\cdot|s))] \leq \delta$
10: **end for**

To implement the GAIL generative model G and the discriminator D we have used two neural networks consisting of two dense layers of 100 nodes, each layer with *tanh* activation. The input for G corresponds to the position and temporal variables per state, and the other variables enriching a trajectory state. D takes as additional input the three action variables. Thus, the input of G and D depends on the way positions and actions are formulated. G has a dense output layer with size equal to the number of action variables, while the output layer of D has one node. G outputs for each action variable the mean of a Gaussian distribution with logarithm of standard deviation equal to 0.6, resulting to a stochastic policy. Behavioral Cloning minimizes the Mean Square Error between demonstrated actions and the policy actions, over the training set, using Adam optimization. This has been trained with 100 epochs and 10 fold cross validation.

5 EXPERIMENTAL EVALUATION

5.1 Experimental Setting

Datasets exploited in our experiments include (a) radar tracks for flights between 3 OD pairs: Barcelona to Madrid (BCN-MAD) during July 2019 (308 trajectories), London Heathrow to Rome Fiumicino (LHR-FCO) during July 2019 (219 trajectories), and Helsinki to Lisbon (HEL-LIS) during July 2019 (44 trajectories); (b) weather data obtained from National Oceanic and Atmospheric Administration (NOAA); and (c) aircraft models' ids. Prediction of long flights regarding LHR-FCO and HEL-LIS is the more challenging problem, as it involves large time horizon prediction, with many uncertainties during trajectory evolution.

Trajectories in these datasets have been pre-prosessed, cleaned and enriched with five (5) numerical variables corresponding to 4 meteorological features at any trajectory state position and time, provided by NOAA, and the aircraft model of each trajectory. The NOAA features are *temperature, geopotential height, u-component of wind, v-component of wind.*

The pre-processing stage interpolates points in trajectories, so that two subsequent points have a temporal distance of $\Delta t = 5$ seconds in the case of the short BCN-MAD trajectories and $\Delta t = 10$ seconds in the case of the long LHR-FCO and HEL-LIS trajectories. This task assumes constant velocity between subsequent trajectory points and calculates the position of the aircraft every Δt seconds. It finally keeps only the points occurring every Δt seconds along the original trajectory. The cleaning task aims to detect incomplete trajectories starting or finishing away from any of the airports, as well as flights showing inconsistent behavior (e.g. covering a significant distance within an unreasonably small amount of time), due to imperfections in the raw data.

We measure the prediction accuracy achieved at the pre-tactical phase (starting from a position in the origin airport) and at the tactical phase (starting from any point en-route), introducing a parameter M in {0,0.2, 0.5, 0.7}. M determines the initial state of the prediction, i.e. the state in the actual trajectory after ($M \times FlightDuration$) minutes, starting from t_0.

We report on the trajectory prediction accuracy using the following measures: **(a)** Root Mean Square Error (RMSE) in meters in each of the 3 dimensions, as well as in 3D, **(b)** Along-Track Error (ATE), **(c)** Cross-Track Error (CTE), and **(d)** Vertical deviation (V). ATE and CTE are computed according to the methodology proposed in [12]. As shown in Figure 1, the along track error is measured parallel to the predicted trajectory, while the cross track error is measured perpendicular to the predicted course. V measures the difference in altitude between the predicted and the corresponding test (actual) trajectory.

To compute the ATE, CTE and V, we align the predicted trajectory with the corresponding test (actual) trajectory in the time dimension, so as to calculate the errors between trajectory points with the same timestamp. As the predicted trajectory may have different length (different number of points) compared to the test trajectory, we compare the points of the longer trajectory (predicted or actual), to the last point of the shorter trajectory, as this is the last known position of the aircraft. Finally, we provide results on **(c)** the estimated time of arrival (ETA) error, given the predicted ETA and the arrival time of test trajectories. All errors ATE, CTE, V and

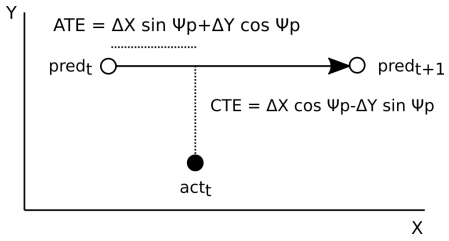


Figure 1: Along track (ATE) and cross track (CTE) errors w.r.t. the predicted trajectory's points at times t and $t+1$, denoted by $pred_t$ and $pred_{t+1}$, and the actual trajectory's point act_t at time t. $\Delta X = X_p - X_a$ is the difference in the X dimension (longitude) of $pred_t$ (X_p) and act_t (X_a). $\Delta Y = Y_p - Y_a$ is the difference in the Y dimension (latitude) of $pred_t$ (Y_p) and act_t (Y_a). Ψp denotes the bearing of the predicted trajectory (i.e. the angle between the direction of the trajectory and the North).

ETA are signed errors, but we take their absolute values in order to report on average scores from multiple experiments, providing a clear indication of the errors.

The RMSE error is computed for each predicted trajectory point after computing its corresponding point in the test trajectory using the Dynamic Time Warping (DTW) [4] [21] method. We compute the RMSE errors in each of the 3 dimensions using the formula

$$RMSE(var) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(var_{pred} - var_{actual})^2}$$

and the 3D RMSE error using the formula

$$RMSE_{3D} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\frac{\sqrt{\sum_{d=1}^{Dim}(var_{d_{pred}} - var_{d_{actual}})^2}}{Dim}}.$$

where, N is the number of trajectory points' pairs compared, Dim is the number of dimension variables considered per point, var is the variable corresponding to a certain dimension and $pred$ and $actual$ indicate the predicted and actual trajectories, respectively.

Specifically, we report on the average of RMSE for the longitude, latitude and all three dimensions (3D), as well as the average of ATE, CTE, V and ETA reported by 20 independent experiments per experimental case. The division of the historical trajectories for training and testing purposes is done randomly for each of the individual experiments using 90% of them as expert trajectories and 10% as test trajectories .

GAIL is trained for 1500 batches. At each round the policy generates a batch of 50000 state-action samples. The number of episodes needed for this number of samples is not constant. At each episode the method randomly selects a starting point regarding a trajectory in the training set and uses G to generate roll-outs. Roll-outs terminate either when a trajectory point lies within a 8km radius from the destination airport, for the cases of BCN-MAD and LHR-FCO and 14km for HEL-LIS or when the trajectory has 1000 points for the cases of BCN-MAD and LHR-FCO and 1900 for HEL-LIS, or when it evolves in positions out of the *prediction area* (defined below). These 50000 samples, along with all the expert samples, are used for training the Discriminator D. Specifically, we use Adam

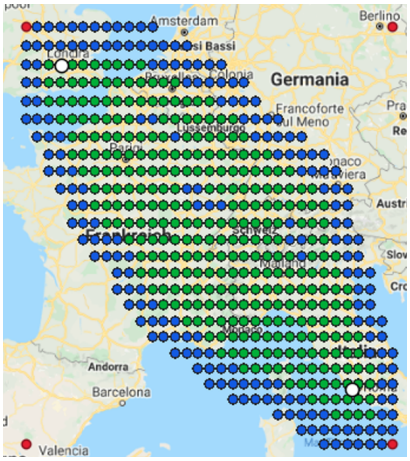


Figure 2: Specification of the bounding box and of the prediction area for LHR-FCO trajectories.

optimization and 100 epochs to maximize equation (3) w.r.t. the D parameters w.

The *prediction area* is a 3D area in which generator's roll-outs are allowed to evolve. While the prediction area for short trajectories can be the 2D trajectory bounding box (along with a maximum allowed altitude of 40000 feet), long trajectories are included in significantly larger bounding boxes (sometimes including the whole continent). To address this issue, the prediction area is the area determined by the 1×1 degree cells that the demonstrated trajectories cross, expanded by additional cells of 1×1 degree in every direction of the initial area to provide a kind of "buffer" that gives room for the generator to explore. For example, red dots in Figure 2 indicate the corners of the bounding box of the LHR-FCO trajectories. In green we specify the cells crossed by demonstrated trajectories and in blue the buffer area.

5.2 Experimental Results

Table 1 shows the average RMSE error of the predicted vs the actual (test) trajectory in meters for each of the three dimensions and in 3D, together with the average absolute ATE, CTE, and VE, in meters. It also reports the average error of the expected arrival time (ETA) in seconds for each case. The table is split to parts corresponding to the different origin destination pairs examined, starting from the short trajectories and going into the longer ones with fewer samples, and for each pair we report the results provided by the GAIL method, for different values of M.

Figures in Table 2 show box plots for all the measures. The x axis specifies the error measured. Horizontal lines of each box plot represent the 25^{th}, the 50^{th}, the 75^{th} and the 100^{th} percentile. Diamonds indicate outliers and the numbers indicate the medians. The left column provides RMSE and the right the along and cross track errors. These box plots correspond to the cases where $M = 0$.

Not surprisingly, the GAIL method provides consistently better results compared to the BC baseline: Indeed, table 2 box-plots show that GAIL reports smaller errors with narrower deviations, and very small number of outliers compared to BC. In addition to that, low deviations of predicted from the actual trajectories, compared to state of the art methods (Section 6) provide firm evidence of the imitation learning approach efficacy, even in very long trajectories spanning the European continent and with few training examples.

Table 1 shows that the proposed method is quite effective to predict the whole trajectory at the pre-tactical stage ($M = 0$), while all measures are reduced in all cases, except from some of the cases while increasing M, i.e. while we select a starting point far from the origin airport, simulating the tactical stage: This happens for instance in the prediction of very long trajectories regarding HEL-LIS. The average along and cross track errors may increase while increasing M in these cases, due to the complexities of the trajectories while approaching the destination airport (i.e. due to holding patterns, maneuvers, etc.). Thus, it seems that a more refined approach must be used to address the landing part of the trajectory more accurately. This is also the case for the ETA error: If we eliminate the holding patterns while measuring errors for the LHR-FCO pair, we get unsigned ETA errors of 67.82, 61.47, 45.94, 35.22 (signed -14.77, -24.22, -17.51, -8.74) seconds, for $M = 0, 0.2, 0.5, 0.7$, respectively. Similar patterns are recorded for the other error measures, providing evidence to our conjecture about the destination airports with complex holding patterns and multiple modes of approach.

6 RELATED WORK & DISCUSSION ON RESULTS

Reinforcement learning techniques inherently deal with trajectories, produced by policies' roll-outs in an action-state space. Such methods have been used in predicting aircraft trajectories [23], as well as humans' and vehicles' trajectories in urban spaces with traffic/crowd. The DART [7] tabular reinforcement learning approach for aircraft trajectory prediction uses an action set that includes commands executed by the aircraft Flight Management System. This approach needs a model-based trajectory prediction method in the loop to predict the next aircraft position given a set of commands, incurring a significant computational cost in the whole process, while it requires discretization of state-action parameters, and constraints on the valid combinations of commands, further incurring significant loss of accuracy in large look-ahead prediction horizons.

As pointed out in the introduction, recent data-driven efforts in the field of aircraft trajectory prediction have explored the application of statistical analysis and machine learning techniques. A comprehensive review of trajectory prediction methods in different domains can be found in [8]. As far as aircraft trajectory prediction is concerned, most approaches make specific assumptions concerning the types of aircraft considered (e.g. [6], the operational phase considered (e.g. climbing, being in terminal airspace, etc.) (e.g. [13], [31]), the short look-ahead time (as in [13] and [5]), or they consider specific constraints that aim to constrain the possible predictions [8]. State of the art approaches in the ATM domain that are closely related to our work are those in [3], [19], [8] and [28]. As also noted in the introductory section, the lack of benchmarks hinders the systematic comparison of different trajectory prediction methods. However, here we attempt a comparison of methods and findings, aiming to show the importance of our contributions.

Table 1: Prediction Errors (in meters) and ETA (in seconds)

BCN-MAD									
	M	**Long**	**Lat**	**Alt**	**3D**	**ATE**	**CTE**	**VE**	**ETA**
GAIL	0	11994.99	6214.72	282.63	8005.49	13923.3	6392.61	574.13	263.06
	0.2	10021.46	5549.65	317.87	6774.39	11541.14	6159.43	516.15	259.81
	0.5	8680.78	5103.39	391.5	5977.16	11360.9	6330.23	510.78	239.35
	0.7	6327.37	4078.96	273.41	4519.16	9751.45	5284.18	375.73	158.9
LHR-FCO									
	M	**Long**	**Lat**	**Alt**	**3D**	**ATE**	**CTE**	**VE**	**ETA**
GAIL	0	23371.12	20888.65	372.33	18351.99	18689.42	17874.3	636.2	457.1
	0.2	24427.65	20568.25	359.35	18733.01	19273.79	19362.4	621.78	615.12
	0.5	20274.53	18497.25	370.98	16209.15	15758.31	16399.46	629.06	791.83
	0.7	14313.57	14444.13	539.25	12126.93	13205.18	11294.95	659.35	910.28
HEL-LIS									
	M	**Long**	**Lat**	**Alt**	**3D**	**ATE**	**CTE**	**VE**	**ETA**
GAIL	0	88448.14	95173.02	1096.41	75950.75	77341.27	59731.61	1074.71	801.44
	0.2	91184.7	100957.56	1062.17	79921.51	81309.09	52941.16	1052.04	978.19
	0.5	90334.3	92006.24	1090.32	76575.08	81468.87	49669.47	1252.01	1080.75
	0.7	77587.38	76998.23	1966.88	64771.15	81990.64	46206.48	1691.44	1113.12

In more detail, authors in [3] introduce a stochastic approach, modeling trajectories in space and time by using a set of spatiotemporal 4D joint data cubes, enriching these with aircraft motion parameters and weather conditions. This approach computes the most likely sequence of states derived by a Hidden Markov Model (HMM), which has been trained over trajectories enriched with weather variables. The algorithm computes the maximal probability of the optimal state sequence, which is best aligned with the observation sequence of the aircraft trajectory. Given that the lateral resolution of each cube is 13km and temporal resolution is 1hr, authors conclude that the mean value for the cross-track error (12.601km when the sign is omitted or -3.444km when signed) is within the boundaries of the spatial resolution. Given that this method does not exploit other information regarding operational aspects, but flown trajectories enriched with weather variables in an OD pair (Atlanta, Miami) with distance similar to BCN-MAD, we can conjecture that our proposed method provides a much lower error in RMSE, as well as in along and cross track errors. This is achieved without limiting the resolution of trajectories' representation, while learning/predicting in continuous action-state space.

In reference [19], authors propose a tree-based matching algorithm to construct image-like feature maps from high-fidelity meteorological datasets. They then model the trajectory points as conditional Gaussian mixtures with parameters to be learned from the proposed deep generative model, which is an end-to-end convolutional recurrent neural network that consists of a long short-term memory (LSTM) encoder network and a mixture density LSTM decoder network. It must also be noted that, that approach requires flight plans, as well as a number of actual trajectory points prior to prediction. Our proposed method seems to be more effective in terms of predicted trajectory deviations from the actual trajectories in all dimensions: Without requiring any information that will guide/constrain predictions, we report on vertical errors larger than 2800 ft but with a much lower 3D RMSE (increased by 1.21) for an increase of the trajectory length by 0.76 regarding the HEL-LIS case.

The approach in [8] is a "constrained" approach, learning the deviations of trajectories from flight plans and reporting low deviations per waypoint. This is in contrast to the proposed approach, which does not exploit any information constraining the predicted trajectory. The 3D RMSE reported for this approach is 1.78 greater than the 3D RMSE of GAIL for the BCN-MAD pair, given the same set of demonstrated trajectories.

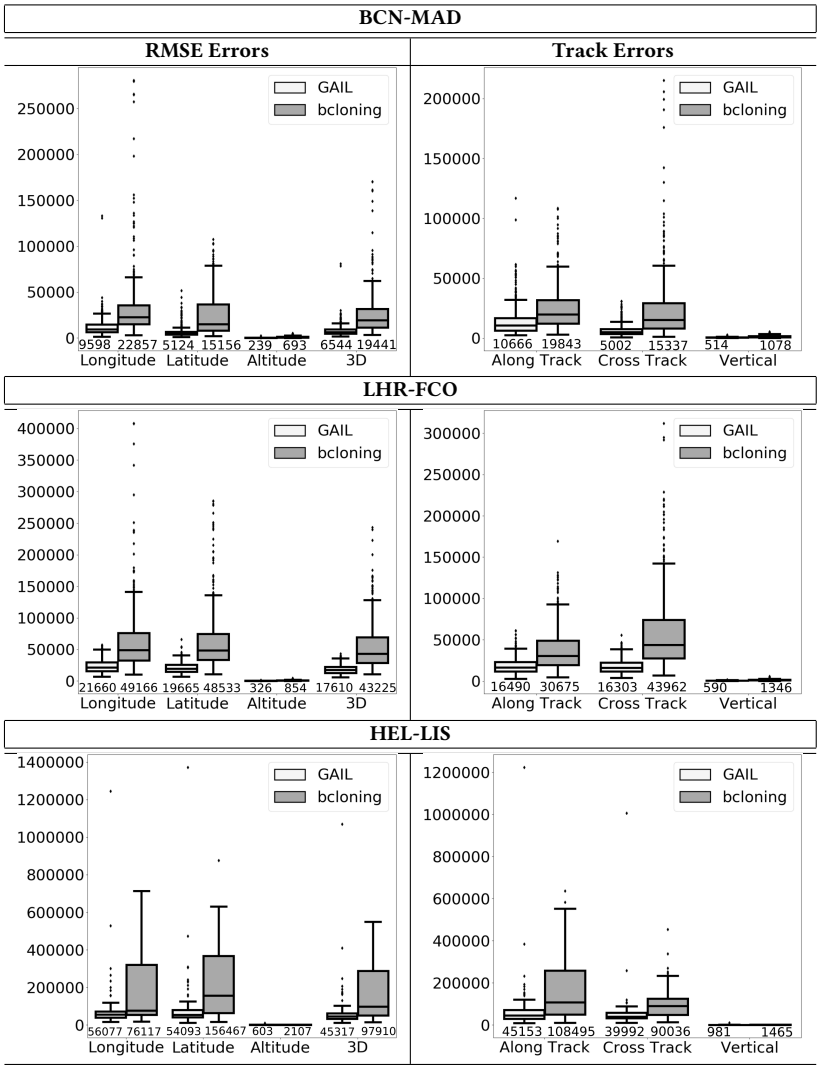
Finally, Applearn [28] is an apprenticeship learning imitation learning approach for the trajectory prediction problem, assuming that the cost function is a linear combination of basis functions on state variables. This method has been proposed as an alternative to GAIL, aiming to study the effectiveness of linear cost functions. Applearn, although quite competent to other state of the art approaches, it fails to achieve the accuracy of GAIL: The 3D RMSE reported is 2 times greater than the 3D RMSE of GAIL for the BCN-MAD pair, and approximately 7.2 and 21.3 times, for the LHR-FCO and HEL-LIS pairs, respectively, given the same data sets of demonstrated trajectories. Given these results, the linearity assumption on the cost function seems to reduce prediction accuracy, although Applearn scores better predictions when starting from a state close to the destination airport (i.e. when $M > 0.5$), which is something to be explored in the future.

7 CONCLUSIONS AND FUTURE WORK

In this paper we specify the data-driven trajectory prediction problem as an imitation learning task. Towards solving this problem we present a prediction method using the Generative Adversarial Imitation Learning state of the art method utilizing a critic model for estimating the state advantage.

Evaluation results show the effectiveness of the method to make accurate predictions for the whole trajectory (i.e. with a prediction horizon until reaching the destination airport) both at the

Table 2: Prediction Errors box plots: Numbers below the boxes indicate the medians.



pre-tactical (i.e. starting at the departure airport at a specific time instant) and at the tactical (i.e. from any state while flying) stages, compared to state of the art approaches. We discuss our findings with respect to results reported by state of the art trajectory prediction methods, although a direct and systematic comparison required methods to be trained using the same sets of demonstrated, flown trajectories.

Future Plans include (a) verifying the effectiveness of the method for different origin-destination airports, (b) exploiting flight plans to constrain the prediction pipeline, (c) extending the method to deal inherently with different modes of trajectory evolution, and (d) generalizing beyond specific origin-destination pairs.

ACKNOWLEDGMENTS

This work has been funded and completed in the context of the ENGAGE KTN Catalyst Fund project entitled "Data-Driven Trajectory Imitation with Reinforcement Learning", and it is partially funded by University of Piraeus Research Center.

REFERENCES

[1] [n.d.]. BADA, Base of Aircraft Data. https://simulations.eurocontrol.int/solutions/bada-aircraft-performance-model/.
[2] P. Abbeel and A. Y Ng. 2004. Apprenticeship learning via inverse reinforcement learning. In *ICML*. 1.
[3] S. Ayhan and H. Samet. 2016. Aircraft Trajectory Prediction Made Easy with Predictive Analytics.
[4] Donald J Berndt and James Clifford. 1994. Using dynamic time warping to find patterns in time series.. In *KDD workshop*, Vol. 10. Seattle, WA, USA:, 359–370.
[5] T. Cheng, D. Cui, and . Cheng. 2003. Data mining for air traffic flow forecasting: a hybrid model of neural network and statistical analysis. *Proc. of the 2003 IEEE Intl. Conf. on Intelligent Transportation Systems* 1 (2003), 211–215.
[6] A. de Leege, M. van Paassen, and M. Mulder. [n.d.]. *A Machine Learning Approach to Trajectory Prediction.* https://doi.org/10.2514/6.2013-4782 arXiv:https://arc.aiaa.org/doi/pdf/10.2514/6.2013-4782
[7] D. Scarlatti et al. 2018. Deliverable D2.4 – Evaluation and Validation of Algorithms for Single Trajectory Prediction. http://dart-research.eu/2018/07/10/dart-final-deliverables/

[8] H. Georgiou et al. 2018. Moving Objects Analytics: Survey on Future Location & Trajectory Prediction Methods. arXiv:1807.04639 [cs.LG]

[9] C. Finn, P. Christiano, P. Abbeel, and S. Levine. 2016. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. *arXiv preprint arXiv:1611.03852* (2016).

[10] C. Finn, S. Levine, and P. Abbeel. 2016. Guided cost learning: Deep inverse optimal control via policy optimization. In *ICML*. 49–58.

[11] H. V. Georgiou, N. Pelekis, S. Sideridis, D. Scarlatti, and Y. Theodoridis. 2019. Semantic-aware aircraft trajectory prediction using flight plans. *Intl Journal of Data Science and Analytics* (2019), 1–14.

[12] Chester Gong and Dave McNally. 2004. A methodology for automated trajectory prediction analysis. American Institute of Aeronautics and Astronautics.

[13] M. G. Hamed, D. Gianazza, M. Serrurier, and N. Durand. 2013. Statistical prediction of aircraft trajectory : regression methods vs point-mass model.

[14] J. Ho and S. Ermon. 2016. Generative adversarial imitation learning. In *NIPS*. 4565–4573.

[15] S. Hong and K. Lee. 2015. Trajectory prediction for vectored area navigation arrivals. *Journal of Aerospace Information Systems* 12, 7 (2015), 490–502.

[16] Diederik P. Kingma and J. Ba. 2014. Adam: A Method for Stochastic Optimization. http://arxiv.org/abs/1412.6980 cite arxiv:1412.6980, also in: 3rd Intl Conf. for Learning Representations.

[17] W. Kun and P. Wei. 2008. A 4-D trajectory prediction model based on radar data. In *2008 27th Chinese Control Conference*. IEEE, 591–594.

[18] Y. Le Fablec and J.M. Alliot. 1999. Using Neural Networks to Predict Aircraft Trajectories.. In *IC-AI*. 524–529.

[19] Y. Liu and M. Hansen. 2018. Predicting Aircraft Trajectories: A Deep Generative Convolutional Recurrent Neural Networks Approach. arXiv:1812.11670 [cs.LG]

[20] J. López-Leonés, M. A Vilaplana, E. Gallo, F. A Navarro, and C. Querejeta. 2007. The aircraft intent description language: A key enabler for air-ground synchronization in trajectory-based operations. In *2007 IEEE/AIAA 26th Digital Avionics Systems Conf.* IEEE, 1–D.

[21] Michael J. Pazzani and Eamonn J. Keogh. 2000. Scaling up dynamic time warping for data mining applications. (2000), 285–289.

[22] D. A Pomerleau. 1991. Efficient training of artificial neural networks for autonomous navigation. *Neural computation* 3, 1 (1991), 88–97.

[23] Boeing Research and Technology Europe (patent filed in the European Patent Office). 2017. Method and system for autonomously operating an aircraft. arXiv:EP17382412.9

[24] S. Ross and D. Bagnell. 2010. Efficient reductions for imitation learning. In *AISTATS*. 661–668.

[25] S. Ross, G. Gordon, and D. Bagnell. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*. 627–635.

[26] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. 2015. Trust region policy optimization. In *ICML*. 1889–1897.

[27] Moritz P. Levine S. Jordan M. & Abbeel a. P. Schulman, J. 2016. High-dimensional continuous control using generalized advantage estimation. (2016).

[28] Christos Spatharis, Konstantinos Blekas, and George A. Vouros. 2020. Apprenticeship Learning of Flight Trajectories Prediction with Inverse Reinforcement Learning. In *11th Hellenic Conference on Artificial Intelligence* (Athens, Greece) *(SETN 2020)*. Association for Computing Machinery, New York, NY, USA, 241–249. https://doi.org/10.1145/3411408.3411427

[29] K. Tastambekov, S. Puechmorel, D. Delahaye, and C. Rabut. 2014. Aircraft trajectory forecasting using local functional regression in Sobolev space. *Transportation Res. part C: Emerging Technologies* 39 (2014), 1–22.

[30] F. Torabi, G. Warnell, and P. Stone. 2018. Generative Adversarial Imitation from Observation. arXiv:1807.06158 [cs.LG]

[31] Y. Yang, J. Zhang, and K.q. Cai. 2015. Terminal-Area Aircraft Intent Inference Approach Based on Online Trajectory Clustering. In *TheScientificWorldJournal*.

[32] B. D Ziebart, A. L Maas, J A. Bagnell, and A. K. Dey. 2008. Maximum entropy inverse reinforcement learning. In *Aaai*, Vol. 8. Chicago, IL, USA, 1433–1438.