A generative adversarial imitation learning approach for realistic aircraft taxi-speed modelling

Pham, Duc-Thinh; Tran, Thanh-Nam; Alam, Sameer; Duong, Vu N.

2021

Pham, D., Tran, T., Alam, S. & Duong, V. N. (2021). A generative adversarial imitation learning approach for realistic aircraft taxi-speed modelling. IEEE Transactions On Intelligent Transportation Systems. <https://dx.doi.org/10.1109/TITS.2021.3119073>

<https://hdl.handle.net/10356/152909>

<https://doi.org/10.1109/TITS.2021.3119073>

© 2021 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. The published version is available at: <https://doi.org/10.1109/TITS.2021.3119073>.

Downloaded on 26 Apr 2023 19:18:52 SGT

A Generative Adversarial Imitation Learning Approach for Realistic Aircraft Taxi-Speed Modelling

Duc-Thinh Pham[§], Thanh-Nam Tran[§], Sameer Alam, and Vu Duong

Abstract—Classical approaches for modelling aircraft taxi-speed assume constant speed or use a turning rate function to approximate taxi-timings for taxiing aircraft. However, those approaches cannot predict spatio-temporal component of aircraft-taxi trajectory due to a lack of consideration of the complexity and stochasticity of airport-airside movements and interactions. This research adopts the Generative Adversarial Imitation Learning (GAIL) algorithm for aircraft taxi-speed modelling, while considering multiple operational factors including surrounding traffic on the ground and target take-off time. The proposed model can learn and reproduce the ground movement patterns in a real-world dataset under different circumstances. In addition, the characteristics of the taxi-speed model are also analyzed, especially focusing on handling conflict scenarios with surrounding traffic. Finally, the travel-time of the aircraft from starting to target positions are compared with baseline models and actual taxiing data. The proposed model outperforms all the baseline models with a significant margin. In terms of spatial completion (SC), it achieves up to 97.1% for arrivals and 88.3% for departures. The results also show significantly high performance for temporal completion. The model achieves a stable performance with low Root Mean Square Error (RMSE) (16.8 seconds for arrivals, 32.4 seconds for departures) and Mean Absolute Percentage Error (MAPE) (4.4% for arrivals and 7.6% for departures). Our model's errors are 72% lower for arrivals and 48% lower for departures when compared to other baseline models.

Index Terms—Generative Adversarial Imitation Learning, learning behavior, aircraft movement, speed prediction, inverse reinforcement learning, taxi trajectory prediction, air traffic management, ground movement.

I. INTRODUCTION

IN 2018, International Air Transport Association (IATA) forecasted a significant increase in the number of air passengers by 2037. With the continued growth in air traffic demand, the whole air transportation system is under pressure with limited resources such as airspace, airport capacity, and controller's cognitive capacity. Airports, with their fixed infrastructure, have emerged as the main bottleneck in the air transportation system [1]. They are facing the capacity challenge with prolonged congestion and delay problems [2]. To address these issues, several researches have focused on improving airport capacity by proposing better planning or decision supporting algorithms [3], [4] and system such as

Surface Management System (SMS) [5], Surface Operations Simulator and Scheduler (SOSS) [6], [7], etc. Some advanced airport supporting systems are already deployed, such as Advanced Surface Movement Guidance and Control System (A-SMGCS) and Airport Surface Detection Model (ASDE-X). These systems provide controllers more decision-support functions and generate more operational data such as spatio-temporal taxi trajectories.

All those systems and models for airport surface operations require the modelling of airport traffic system at different levels in their formulation and evaluation. However, modelling completely a complex airport traffic system is challenging, therefore majority of approaches have been using simulation with different levels of simplification for the evaluation. From the literature, based on the fidelity levels [8], the simulation modelling can be broadly classified into three groups- microscopic [3], [6], [7], [9]–[12], mesoscopic [4], [13], and macroscopic [14], [15] models. Microscopic approaches demonstrate higher performance for the potential travel-time and fuel savings compared to the other approaches [16]. Microscopic models assume control over each aircraft's precise location and velocity at all times, and the control is implemented by specifying a time-based trajectory for each departure and arrival aircraft. For instance, microscopic models such as Airport and Airspace Simulation Model (SIMMOD) [17], or Total Airspace and Airport Modeler (TAAM) [8] specify the dynamics for each aircraft which requires the simulation of individual aircraft trajectories. However, the primary drawback of such methods is the conformance of aircraft spatio-temporal trajectory and proposed schedules. The proposed simulation model with non linear control system in [18], developed for a detailed B-737 aircraft taxi model, is used to achieve high-precision taxi control while complying with taxi clearances with tight time margins. In general, precise spatio-temporal trajectory estimation is the key element in several optimization algorithms and systems in both literature [19], [20] and operation [21]. For instance, even though A-SMGCS system recommends a flight plan with taxi routes and necessary timestamps, aircraft taxiing speed profile can still vary significantly (Fig. 1). Therefore, an accurate aircraft taxi-speed model, which can estimate precisely aircraft positions during taxiing, plays an important role in improving the performance of airport airside operations.

The most straightforward way to model aircraft taxi-speed is to assume that they travel at a constant speed. This assumption was used widely in ground movement problems [10], [22]

D.T. Pham, T.N. Tran, S. Alam and V. Duong are with Air Traffic Management Research Institute, School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore (e-mail: {dtpham | thanhnam.tran | sameeralam | vu.duong}@ntu.edu.sg)

[§]Equal contribution

Fig. 1. Variations in speed profiles of departures for the same taxi-route from historical data of Changi Airport.

because it reduces the complexity of considering all aircraft speed factors. However, airport ground movement is a complex logistics operation; thus, such assumption creates a huge gap between the model and the reality. Some studies propose more complex rule-based models to obtain better estimations of aircraft speed. The work in [23] models the speed limitations as a function of turning rate to estimate the speed for each taxi segment. In airport simulators (e.g., SIMMOD [24], SOSS [25]), the aircraft speed was chosen by adjusting the nominal speed that could be a set of constant speeds for different areas in an airport. The algorithm in [20] is designed for speed decision, determined by the airport congestion level. All rule-based models usually depend on subjective assumptions and the specific implementation that is hard to generalize and do not guarantee to match real data.

As aircraft taxi-speed is stochastic with various movement patterns, one of the potential solutions for a better taxi-speed model is applying data-driven or Machine Learning approaches to reproduce the aircraft movement behaviors. The approach in [26] is proposed for predicting taxi speed by combining the average speed of straight and turn segments. However, manual work is still required to assign segment types for the new taxi route in prediction. Thus, it is challenging to train and test this model with a large dataset. A supervised learning approach using Random Forest is proposed in [27]. The model is trained with A-SMGCS data for predicting a sequence of unimpeded taxi speed on each taxi segment of the given taxi route. The adoption of this supervised learning approach for the taxi-speed modelling problem is challenging due to the large, stochastic and continuous state space of the problem.

Among machine learning (ML) techniques, imitation learning [28], [29] is widely used for this kind of problem, in which, the agent forms the policy for a task by observing how an

expert performs that task. Since the learning is based on the demonstration, which contains the information regarding how to perform the task correctly. There are two major research directions for imitation learning, namely Behavioral Cloning (BC) [30], [31] and Inverse Reinforcement Learning (IRL) [32]–[34]. BC uses directly supervised learning to approximate the mapping from observations to actions. However, BC has the compounding errors [35] and requires enormous training data [36]–[38]. Even with the recent advances in deep learning techniques, BC approaches [39], [40], trained with large datasets using deep learning, still showed the compounding errors during simulations. On the contrary, IRL tries to recover the reward function followed by the experts, assuming that they follow an optimal policy. Approaches in this direction can be generalized more effectively [41] and have been applied in several studies and applications for modeling human behavior [42], [43]. However, training the model using IRL is computationally expensive. The reward function should be learned from data, then the reinforcement learning (RL) techniques are applied with that reward function to find the optimal policy. Recently, there have been the third research direction introduced, namely Generative Adversarial Imitation Learning (GAIL) [44]. This direction attempts to imitate the expert behavior through direct policy optimization rather than learning the reward function first.

This study proposes an aircraft taxi-speed model by adopting the GAIL algorithm while considering multiple operational factors, including surrounding traffic and target take-off time. The proposed taxi-speed model is designed to control the aircraft taxi movement between two pre-defined positions (e.g., the assigned taxi-route) by providing appropriate accelerations at each time step. Noting that aircraft ground movements can be affected by two major factors - the pilot's taxiing behavior and the ground controller's interventions. Since only a small part of the ground controller's interventions can be observed from the ground movement data, such as stop-and-go for conflict resolution, this work will focus on modeling aircraft taxi-speed without explicit controller's interventions. Thus, the aircraft taxi trajectories will be analyzed to detect stop-and-go timestamps. Then they are split into multiple sessions based on those timestamps for developing the model. This work is based on another assumption in which the taxi route for each flight is provided as part of the input along with the other flight information to construct the state vector. The performance of the proposed model is accessed both in successfully modelling the aircraft taxi-speed and imitating underlying patterns in ground movements. The former involves comparing with the baseline model regarding spatial and temporal completion, including analysis of the generated travel-time errors. While the latter analyses characteristics of the proposed model, especially focusing on conflict scenarios with surrounding traffic. The experiment results have shown that the proposed model can successfully model taxi-speeds in complex traffic scenarios while producing similar patterns and behaviors, as observed in historical data. Moreover, as the model can mimic actual behaviors in data, it can also be used as an aircraft movement module in any airport ground movement simulator to generate realistic taxi movements with precise travel time.

The contributions of this paper are as follows:

- This is the first attempt to develop a data-driven aircraft taxi-speed model using inverse reinforcement learning (IRL). The proposed approach adopts the Generative Adversarial Imitation Learning (GAIL) framework to train a control policy that can model the aircraft taxi-speed while successfully considering surrounding traffic and mimicking the aircraft ground movement from historical data.
- This work extends the Generative Adversarial Imitation Learning (GAIL) framework with Proximal Policy Optimization (PPO) as the learning algorithm. Some modifications are applied to make the model converge quickly. Firstly, absorbing states are designed as the terminal state in our state space. Secondly, similar to curriculum learning, the training starts with short and straightforward fixed-length sessions and gradually increases the length to consider more complexity and challenges. Finally, to speed up the training process, we also distributed the training process of the PPO algorithm.
- A data-driven airport ground movement simulator that can simulate surface traffic scenarios for an airport (i.e. Singapore Changi Airport) is developed. Ground movement data is processed and transformed into a spatial representation by using node-link graph representation. In addition, state features, action space, and reward structure are also designed for training the model effectively.

The structure of this document is as follows: Section II introduces the data and required processing steps performed for this research, followed by the proposed approach in Section III. The data-driven baseline models are introduced in Section IV. Section V then documents the experimental settings used followed by experimental results and discussions in Section VI. The paper concludes with the conclusion and future work in Section VII.

II. DATA PREPARATION

A. Graph Representation of the Airport-airside Network

The airport-airside network can be represented by node-link graph $\mathcal{G}(N, E)$ [23] [25] where the set of nodes N and the set of edges E of airport graph corresponding to the line marking and the taxiways in the real operation. The graph $\mathcal{G}(N, E)$ in this study is constructed from public OpenStreet map data [45] using OSMNX package [46]. We use all physical infrastructure with the tag “aeroway” but filter out the tags: “aerodrome”, “apron” and “terminal”. The nodes and edges of the graph correspond to the lines constructing the taxiway. The graph is simplified by eliminating all nodes that are not junctions or dead-ends [46]. Edges (E) represent taxiways and nodes (N) represent stands, junctions and intermediate points. In the final graph, a taxiway (between two intersections) is presented by set of edges, while an intersection is presented by a set of nodes and edges. In this work, each edge is also referred to as a taxi segment. The resulting Changi Airport graph contains 518 nodes and 1686 edges (see Fig. 2). The graph network of any selected airport can be obtained using this method.

Fig. 2. An illustration of constructed graph of Changi airport. The grey dots and lines represent nodes and edges of the graph. The red line shows an example of a taxi-route on the graph.

B. Spatial Representation of Taxi Trajectory using a Node-Link Graph

A spatial of taxi trajectory from a gate to runway entry or from runway entry to gate is the sequence of edge on the airport graph or we can represent it as a polyline $P = \{e_0, e_1, \dots, e_n\}$ with $e_i \in E$. Therefore, any position of aircraft on the taxiway can be represented by a pair of information (P, q) where P is the polyline representation of spatial taxi trajectory and the ratio $q \in [0, 1]$ encode the percentage of the travel distance of the current aircraft position from the beginning on the total length of taxi trajectory. Fig. 3 shows an example of the taxi trajectory and aircraft position represented by graph. Compared to other representations such as the cellular automata (CA) model [20] that divided the airport-airside network into many discrete cells and model the movement of aircraft as the movement between cells, the representation can model any state airport traffic at any time point. As the proposed approach requires a significant amount of training iterations before the model can converge and stabilize, the computational time of feature extraction, at each time step, plays an important role. The node-link representation, developed in this work, is designed to facilitate the feature extraction process, especially conflict-related features. Besides, using the given taxi-route, the information of aircraft’s positions can also be represented in lower dimensions using node-link graph structure (Section III-B).

Fig. 3. An illustration of the proposed representation. A pre-defined taxi-route ($P_{taxiway}$) is highlighted in red color. The position of aircraft ($lat = 1.3463, lng = 103.9848$) with the given taxi-route is highlighted as a green circle and corresponding to $q = 0.46$ or 46% of the required travel distance.

C. Data Pre-processing

In this study, we have used A-SMGCS data for Singapore Changi Airport. A-SMGCS data contains information about the ground movements of aircraft, 4D trajectories and flight plan. For the movement part, every data point records the calculated position with the corresponding velocity of aircraft in the airport.

High tracking frequency of the surveillance system generates a large amount of movement data, such as noise, which may have misaligned data points (in time and space). Thus, the first pre-processing step is to apply a map-matching algorithm to standardize the trajectory data and remove noise and outliers. The applied map-matching algorithm (adopted from author's previous work [47]) includes two steps - spatial matching and temporal map-matching. In spatial matching, the tracking positions of the aircraft are matched with the airport-airside network to obtain a taxi-route that is the graph representation of aircraft movement as a sequence of taxi segments. In temporal matching, every data point is projected to the taxi-route to get the pair of information (t, q_t) where t is the timestamp and q_t is the ratio of the aircraft on the taxi-route. Given the sequence of timestamps and ratio values, we can get the position of aircraft on the taxi-route at any timestamp using linear interpolation. The outliers in terms of speeds and travel-time are also detected and removed in this step.

III. LEARNING ENVIRONMENT AND AIRCRAFT AGENT

A. Markov Decision Process

In this work, an aircraft agent or agent is a taxiing aircraft that needs to travel between two pre-defined positions in a given session. During the process, the agent interacts with the environment by adjusting its accelerations along the given route to maximize the long-term rewards. This interaction process is formulated as a Markov Decision Process (MDP) which is described by a four-element tuple $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P})$, in which, $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{R}$ is the reward function and $P : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ is the transition probability distribution. In this study we focus on episodic task so we define trajectory $\{s_1, a_1, \dots, s_T, a_T\}$ as a collection of state and action of a target aircraft with T is the termination time step. Where s_t is aircraft's state (the aircraft's state is a combination of aircraft's planning, aircraft's location and surrounding traffic) and the action a_t at the time step t can be a aircraft speed or aircraft acceleration. To unify notation for episodic and continuous tasks, we use the definition of a set of absorbing states [48] that transits only to itself and generates only rewards of zero. With this notation, the return will be $\sum_{i=0}^T \gamma^i r(s_i, a_i)$ with the discount factor γ , the termination time step T correspond to an absorbing state, and the reward $r(s_i, a_i)$ of the executing action a_i in state s_i . The optimal policy π^* is the policy that maximize expected rewards $E[\sum_{i=0}^T \gamma^i r(s_i, a_i) | \pi^*]$.

B. Modeling Spatial-Temporal Taxi trajectory

We then develop a spatial-temporal taxi trajectory model from spatial representation of taxi trajectory, given a position

of the aircraft at the time step t as (P, q_t) with the taxi-route P and the current ratio of the aircraft on its taxi-route q_t . The movement of aircraft from time step t to time step $t + 1$ is simply the travel distance of the aircraft from t to $t + 1$ that can be estimated from aircraft speed v_t at time step t . The position of the aircraft at time step $t + 1$ (P, q_{t+1}) is updated by: $(P, q_{t+1} = q_t + \frac{v_t * \Delta_t}{\text{length}(P)})$, in which Δ_t is the gap between two consequent time steps. That means with the speed information at time step t , we can model any spatial-temporal position (P, q_t) of the aircraft. Thus, the spatial-temporal taxi trajectory $\tau = \{(P, q_i)\}$ with $i = 0, 1, \dots, T$ can be considered as the problem of modelling the aircraft taxi-speed v_t at any time step t . In this work, for smoothly controlling aircraft speed, Δ_t is set to be one second. Moreover, at each time step, aircraft speed is updated by the acceleration value or the action a_{t-1} : $v_t = v_{t-1} + a_{t-1} * \Delta_t$. Fig. 4 shows the concept diagram of the taxi-speed model. The aircraft takes action (acceleration) to adjust its speed and moves to a new position. At the new position, the agent updates the new observation about airport traffic and its plan to decide the next action (e.g., increasing or decreasing the current speed) until they reach the target positions. The accelerations are selected as the actions in work because of the following reasons. Firstly, speed is continuous variable and its range is high (from 0 m/s to 18 m/s) that can generate fluctuated speed profiles if the model is trained directly using speeds. Secondly, even though the fluctuations can be reduced by adding penalty for the magnitude of speed change, model's design and training are more challenging since the feasible actions will vary significantly based on the current speed of the aircraft. Using acceleration as the actions can maintain the consistency of the action space while also reducing the action space's dimension and complexity.

Fig. 4. The MDP diagram for taxi-speed model of an aircraft agent.

C. Constructing Environment

In this work, a session is a subset of trajectory data, defined as a period the aircraft begins to taxi from the starting position to the target position (the runway entry or the position that controllers ask aircraft to stop to resolve possible conflicts). The spatial trajectory is assigned by ATCO before the aircraft

starts moving. For all surrounding traffic, actual trajectory data is extracted and used. The environment loads the session at the start of the simulating episode, then simulates it step-by-step (1 second) with actions inputted by the agent. In each episode, the agent's task is controlling the aircraft's speed by changing the acceleration in response to the current traffic situation appropriately. The episode terminates when the agent falls into an absorbing state. The absorbing state is defined if the aircraft reaches the target position, performs invalid actions, or the conflict detection module detects the possible conflict between two aircraft, which requires ATCO's intervention. Since this study does not consider ATCO's interventions, the episode will be terminated if an ATCO's intervention for conflict resolution is required. Every time the environment resets, a new session will be loaded.

(a) Distribution of acceleration for both departures and arrivals.

(b) Distribution of acceleration during the first 20 seconds of arrivals after turning off the runway.

Fig. 5. Distribution of acceleration in historical data.

As mentioned, in this work, taxi-speed model problem is developed by controlling the acceleration. The agent's actions are the value of acceleration. To reduce the noise of the data during training and narrow down the action space, the acceleration is discretized into six values. To select the appropriated values for the actions, distributions of accelerations from historical data are analysed. Fig 5a shows the acceleration distribution in historical data for both arrival and departure flights on

the airport surface. As observed from the figure, more than 60% of acceleration values falls into the range $[-0.1, 0.1]$ and more than 90% of data falls into the range $[-0.5, 0.5]$. There is also a high peak at 0 which means the aircraft has a tendency to maintain their speed. Moreover, as illustrated in Fig 5b, after arrivals turn off the runway and enter the taxiway network, they have performed high deceleration to reduce its speed. To consider this period in our model, an action with high deceleration value ("brake") will be added into the action space. Therefore, the actions are selected as follows:

- 0: acceleration is -1.5 m/s^2 ("Brake")
- 1: acceleration is -0.5 m/s^2 ("Slower")
- 2: acceleration is -0.1 m/s^2 ("Slightly slower")
- 3: acceleration is 0 m/s^2 ("Keep speed")
- 4: acceleration is 0.1 m/s^2 ("Slightly faster")
- 5: acceleration is 0.5 m/s^2 ("Faster")

To evaluate the quality of the selected actions, they are used to approximate the actual data and the difference is insignificant. The speed profiles constructed by discrete acceleration vary just 5.06% from true speed profiles, and time difference is only 1.01 second on average. Fig 6 shows an example of the similarity between the actual speed profile and the approximated speed. In case where the actual acceleration is out of the range $[-1.5, 0.5]$, the approximate speed can be different from the actual speed in a few time steps, but it can quickly catch up later.

Finally, all the aircraft movements in the data, which are used for training and testing, are approximated by this set of actions.

Fig. 6. Comparison between actual speed profile and approximate speed profile.

D. State Encoding

State of every time step is defined by the list of features described in TABLE I. We define 5 groups of features as:

- Flight plan features: related to the planning of the flight that can affect the movement of aircraft, such as the time aircraft will depart (Target take-off time (TTOT)). The flight plan features reveal the intention of the pilots taxiing the aircraft. For instance, if the aircraft is late

compared to the expected time of arrival, it will be speed up to reach the target position at the intended time.

- Flight status: related to the current status of the flight, such as current position, previous speeds. The flight status information, such as the previous speed, affects the decision for the current speed. For example, in a given situation when an aircraft is close to the runway entry point and it is moving at a high speed, it will likely to slow down. Otherwise, if an aircraft is moving at a low speed, it will likely maintain their speed.
- Taxi-route features: related to the taxi-route characterizations, such as turning angle, area of high speed or low speed. Taxi-route features provide physical constraints that can affect the aircraft taxi speed. For instance, an aircraft usually keeps slow speed at turns and the speed management depends on how big the turning angle is.
- Traffic features: related to the traffic situation surrounding the flight, such as traffic density, conflict information. The traffic situation is one of the main factors affecting the taxi speed. For instance, if another aircraft is moving ahead at a slower speed, then the current aircraft will also moves slowly to maintain a safe distance.
- Aircraft features: the physical characteristics of the aircraft that should affect the aircraft's movement. At this time, we consider aircraft type.

E. Adversarial Imitation Learning

In RL, the policy tries to learn to maximize the expected rewards. However, we do not have specific reward rules for guiding policy learning to mimic the aircraft movement. We can manually set the reward rules, but defining reward rules and reward function that can be optimized effectively and encoding the correct aircraft movement task as we expect can be challenging in practice [33]. Another solution is to apply inverse reinforcement learning (IRL) or imitation learning to train policy directly from demonstration data in the A-SMGCS system.

The target of imitation learning is training the agent to perform a task from the demonstration by learning a mapping between observations and actions [49]. In this study, we use the Generative adversarial imitation learning (GAIL) framework to train the agent that mimics the aircraft ground movements from the demonstration data. Fig. 7 shows the general concept of GAIL. GAIL borrows the idea of Generative adversarial networks (GAN) [50] that train simultaneously two models: a generator G and a discriminator D . In GAN, the discriminator tries to classify its inputs as either the output of the generator or actual samples from the underlying data distribution $p(x)$. On the other hand, the generator's goal is to produce outputs classified by the discriminator as coming from the underlying data distribution. In the GAIL context, the policy π acts as the generator and the reward function acts as a discriminator that distinguish whether the data come from the policy or the demonstration data. The reward from discriminator will assign the high values for actions similarly to the ones in demonstration data and low values for actions which are totally different from demonstration data. This signal is used to train the policy

to generate the data close to the demonstration to confuse the discriminator. As mentioned, unlike IRL approaches where the reward functions are explicitly approximated from data, GAIL uses the typical unconstrained form of discriminator as the guidance for learning rather than the typical reward function. Particularly, the reward function remains implicitly within the discriminator and cannot be used directly. The final result of the training process is only the aircraft taxi-speed policy.

Fig. 7. A Conceptual representation of the Generative adversarial imitation learning framework adopted in this work.

Algorithm 1: Proposed algorithm adopted from GAIL [44] and distributed PPO [51]

```

1 initialization: fixed-length  $T$ , demonstration dataset  $B_E$ , shared buffer  $B_i$ ;
2 for  $i = 0, 1, 2, \dots$  do
3   Gradually increase  $T$  every 500 iterations;
4   for environment  $env_j = env_1, env_2, \dots, env_M$  do
5     Run policy  $\pi_{\theta_i}$  in environment  $env_j$  for fixed-length  $T$  timesteps. When one episode reach absorbing state, it will be terminated.
6     Store trajectories (trajectory is a sequence of states and actions  $(s, a)$ ) in shared buffer  $B_i$ .
7   end
8   Samples  $\tau_i \sim B_i$  and  $\tau_E \sim B_E$ 
9   Update the discriminator parameters from  $w_i$  to  $w_{i+1}$  with the gradient
      
$$\hat{\mathbb{E}}_{\tau_i} [\nabla_w \log(D_w(s, a))] + \hat{\mathbb{E}}_{\tau_E} [\nabla_w \log(1 - D_w(s, a))]$$

10  Take a policy step from  $\theta$  to  $\theta_{i+1}$  using the value  $\log(D_{w_{i+1}}(s, a))$  as reward signal to update the policy using PPO rule on dataset  $B_i$ .
11 end

```

The Algorithm 1 describes general procedures involved in our proposed algorithm. In the initialization step (line 1), we input the initial fixed-length T and the demonstration dataset B_E that contains pairs of state and action (s_t, a_t) in the actual data, and the shared buffer B_i that be used to store pairs of

TABLE I
LIST OF FEATURES USED IN STATE ENCODING

	Feature	Description
1	Remaining time to the start of the time window	The remaining time to the lower bound (-10%) of the expected end time
2	Remaining time to the end of the time window	The remaining time to the upper bound (+10%) of the expected end time
3	Remaining distance	The remaining distance to the target position
4	Number of next crossings	The number of crossing ahead (1000m) that the aircraft need to travel
5	Number of next turns	The number of turns ahead (1000m) that the aircraft need to travel
6	Next turning angle	The angle the aircraft need to take at the next turn
7	Distance to the next turn	The distance from the current position to the next turn.
8	Total angle of next turns	The total turning angle the aircraft needs to take in a fixed distance ahead (1000m)
9	Mean segment speed	The average speed on the current segment for the corresponding aircraft type
10	Standard deviation of the segment speed	The standard deviation of speeds on the current segment for the corresponding aircraft type
11	Number of surround aircraft	The distance from the current aircraft position to the next segment
12	Number of aircraft ahead	The number of aircraft moving ahead (1000m) on the target aircraft's taxi-route
13	Distance to the closest aircraft	The distance of the target aircraft to the closest aircraft that is taxiing
14	Distance to the trailing conflict	The distance of the target aircraft to the possible trailing conflict if the conflict exists
15	Speed of the other aircraft in the trailing conflict	The speed of the other aircraft in trailing conflict if the conflict exists
16	Distance to the crossing conflict	The distance of the target aircraft to the crossing conflict if the conflict exists
17	Distance of the other aircraft to the crossing conflict	The distance of the other aircraft to the crossing conflict position if the conflict exists
18	Speed of the other aircraft in the crossing conflict	The current speed of the other aircraft in the crossing conflict if the conflict exists
19	Previous speed	The target aircraft's speed at previous time step.
20	Aircraft type	The type of the target aircraft that includes two types: Medium and Heavy

state and action simulated by the environments with agent's actions. Fixed-length T is the maximum number of episode's steps simulated by an environment and a policy every iteration. The demonstration dataset B_E contains the data generating by actual actions that we extracted from historical data.

The curriculum learning technique [52], whose basic idea is to "start small" by learning easier aspects of the task and then gradually increase the difficulty level, has been applied during training. The fixed-length T is set with small value (e.g., 32) at the beginning of the training process to make sure the agent can learn how to start-up and then gradually increased fixed-length later (line 2) to consider more complex and challenging scenarios. We found that setting the small fixed-length works well in practice. At the beginning of the training process, the policy frequently faces the stuck-at-begin situation. This situation happens because the initial policy is a naive model, so it is easy to choose the invalid actions (e.g., taking deceleration at speed 0 m/s to cause the speed negative), making the shared buffer B_i dominant by invalid examples and the policy learning stuck. Setting small fixed-length helps the policy overcomes the stuck-at-begin situation faster.

Instead of running single environment, the training process is distributed by running different environments simultaneously to speed up the training process (lines 4-7). Every environment parallelly generates fixed-length T segments of experience and stores it in shared buffer B_i . Noted that when one episode reach absorbing state, it will be terminated.

Then we sample expert data from the demonstration dataset and simulate data from the shared buffer to update the discriminator parameters $D : S \times A \rightarrow (0, 1)$ (line 9). The output of the discriminator is the probability that the input data come from the demonstration dataset. The value $\log(D_w(s, a))$ will be used as reward signal to update the policy parameter θ through Proximal policy optimization (PPO) rule [51] (line 10).

In principle, we can use any on-policy RL algorithm for the

policy updating part in GAIL. In this work, we use PPO due to its robustness and data efficiency. The main idea of PPO is that it tries to update the policy with the biggest possible improvement step but avoids to step so far that can cause performance collapse. That idea is similar to Trust Region Policy Optimization (TRPO) [53] but PPO is much simpler to implement, more general, and empirically have better sample efficiency [51].

F. The Absorbing State

As mention in [54], the prevalent assumption of many imitation learning and IRL algorithms are implicitly assigning zero reward value to absorbing states. The non-zero reward for absorbing states is very important for training imitation learning policy. Especially in aircraft movement cases, we have different kinds of termination events such as conflict, invalid speed. We should have absorbing states to identify which state is valid or invalid to give appropriate penalty or bonus reward.

Drawing motivation from absorbing states implementation of [54], the state feature set is extended with one feature indicating the status of the current state. However, different from the approach in [54], our absorbing states have two types - valid and invalid corresponding to the situations that the agent successfully finishes the task or is terminated by executing an invalid action (speed out of valid range, collision, etc.). We assigned the absorbing-state feature 0 for the non-absorbing state, 1 for the valid absorbing state and -1 for the invalid absorbing state. By absorbing state indicator feature, the GAIL discriminator can recognize whether reaching an absorbing state is desirable from the expert's perspective and assigns the rewards accordingly. Note that because there is no invalid action in practical, we never observe invalid-absorbing state in demonstration data and always get the valid-absorbing state at the end of the demonstration episode. Thus, GAIL discriminator can learn to assign low reward for the invalid absorbing state and in contrast, they should assign high reward for the valid absorbing states.

IV. BASELINE MODELS

In this work, different data-driven models are considered as the baselines for comparing with the proposal model for investigating its pros and cons.

The first baseline model is the statistical model that controls the aircraft speed by setting the reference speeds for different locations (defined by taxi-segments), similar to the idea used in the SOSS simulator [17]. This model, referred to as “Mean”, uses the average speed of each segment from historical data as the reference speed to simulate the movement of taxiing aircraft.

Others are supervised learning models which are trained with the same historical data, used for the proposed model. The same feature set at each time step in the data is extracted and the corresponding movements of taxiing aircraft are the response. Two popular supervised learning models, Random Forest and Neural Network, are used for training. Moreover, both ‘speed’ and ‘acceleration’ are considered as the responses. Therefore, four models are developed: Neural Network to predict speeds (NN[s]), Neural Network to predict accelerations (NN[a]), Random Forest to predict speeds (RF[s]), and Random Forest to predict accelerations (RF[a]). The grid-search technique is applied for Random Forest models to optimize their hyper parameters before reporting final experiment results with the best models. Besides, all Neural Network models are used the same architecture as the actor network of the proposed model. This setup facilitates the analysis of the difference between supervised learning and reinforcement learning models for this problem.

V. EXPERIMENTS

For experiments, we used 3 weeks of processed A-SMGCS data of Singapore Changi Airport which includes 8072 departure and 7212 arrival flights to construct the environment for training and evaluation. Flights that have sampling rates in A-SMGCS data above $0.5Hz$ (more than 1 sample for every 2 seconds) are selected as the target (or controlled) flights. Then we detect stop-and-go positions (as interventions from the controller) in the target flights and split their trajectories into different sessions to guarantee that every session contains only the moving time. The sessions, whose duration is less than 180 seconds, are also filtered out. After finishing these steps, the final dataset includes 5806 departure and 4545 arrival sessions. The training dataset contains 2525 departure and 1758 arrival sessions, corresponding to the 1st week of the dataset, and the remaining sessions are used for evaluation. The travel-time distributions of selected sessions are visualized in Fig. 8.

The taxi-speed model described in Section III-E is implemented using Pytorch and motivated by the PPO implementation in [55]. For the critic and policy networks, we use the two-layer multilayer perceptrons (MLP) with $\tanh$ activation function. The first hidden layer has 150 units and the second hidden layer has 100 units. For the discriminator, we use the two-layer MLP with 100 hidden units and $\tanh$ activation as same as the architecture proposed in [44]. The proposed implementation uses a distributed setting similar to A2C architecture to speed up the training process. Ten

(a) The travel time distribution of selected departure sessions.

(b) The travel time distribution of selected arrival sessions.

Fig. 8. The travel-time distribution of selected sessions.

environments are run simultaneously to generate a fixed length of experiences. The fixed-length value was initialized 32 at the beginning of the training process and increased by five every 500 iterations.

Finally, there are two metrics considered in this study for analysing errors of the trajectory travel-time - Root mean square error (RMSE) and Mean absolute percentage error (MAPE):

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{t=1}^n (T'_t - T_t)^2} \quad (1)$$

$$\text{MAPE} = \frac{100\%}{n} \sum_{t=1}^n \left| \frac{T'_t - T_t}{T_t} \right| \quad (2)$$

where n is the size of evaluated data, and T'_t is the total simulated travel-time for the input trajectory τ_t with the known total travel-time T_t .

VI. RESULTS AND DISCUSSIONS

After converging, the trained model, also called the policy, can generate a sequence of actions to control aircraft taxi-speed from starting position to target position for unseen

scenarios without collision. The acceleration action is then converted into aircraft taxi speed to simulate the aircraft movement (as described in Section III-B).

The comparison on model's performances between the policy and baseline models (mentioned in Section IV) is shown in Table II. Among the baseline models, all supervised learning models using acceleration action have shown significantly poor performances. It implies that those models fail to solve this control task. On the other hand, the policy outperforms all the baseline models with a significant margin. In term of spatial completion (SC), the policy achieves 97.1% for arrivals and 88.3% for departures. It means that using the policy to control the aircraft taxi-speed, the target aircraft can successfully complete the assigned taxi-route in the given session without falling into any invalid condition. Compared to the policy, the other baseline models using speed action also achieve high spatial completion for both departure and arrival. In which, the *NN[s]* and *Mean* are the best baseline models which have the nearest performance to the policy with 90.1%, 93.3% for arrivals and 75.4%, 68.0% for departures. However, the results of the policy have shown significantly high performance for temporal completion (with two mentioned metrics RMSE and MAPE). The policy achieves a stable performance with low RMSE (16.8 seconds for arrival, 32.4 seconds for departure) and MAPE (4.4% for arrival and 7.6% for departure). Compared to the other baseline models, the policy's errors are less than 30% of their errors for arrivals and 45% of their errors for departures.

TABLE II
THE COMPARISON BETWEEN THE POLICY AND BASELINE MODELS.

	Arrival			Departure		
	SC	RMSE	MAPE	SC	RMSE	MAPE
RF[a]	26.8%	294.6s	100.7%	43.6%	237.4s	64.0%
NN[a]	32.1%	233.4s	64.1%	46.2%	211.7s	51.9%
RF[s]	86.1%	98.2s	23.3%	64.6%	115.9s	25.2%
NN[s]	90.1%	84.4s	20.0%	75.4%	62.6s	12.9%
Mean	93.3%	60.2s	14.9%	68.0%	70.2s	15.3%
Policy	97.1%	16.8s	4.4%	88.3%	32.4s	7.6%

Fig. 9 presents three examples of speed profiles that are generated by the policy similarly to the actual speed profiles. The valleys in Fig. 9a denote the aircraft turning or conflict resolution which are also highlighted in their corresponding spatial information visualizations in Fig. 9c. We can observe that the owner has low speed near the apron where the geometry and traffic are complex or before reaching the runway holding point (target position) but higher speed during the straight taxi segment. It demonstrates that the proposed model is able to learn and mimic behaviors in data in handling different traffic situations. Moreover, Fig 9b shows the time space diagram of those three trajectories. Even though small deviations can be observed during the actual trajectory, the policy can closely approximate the real aircraft spatial-temporal positions comparing to the best baseline model (*NN[speed]*). In the simulation, the time to reach the crossing points is important as the total taxi-time. The accurate simulation to crossing points supports estimating the traffic situation in the future, such as conflicts, congestion, etc.

(a) Generated speed profiles vs actual speed profile for a selected aircraft.

(b) Time Space diagram of those trajectories.

(c) Visualization of generated speed profile on the airside network.

Fig. 9. An example of a generated speed profile for an evaluated episode.

We now detail the performance of the policy in terms of behavior learning, spatial completion, spatial and temporal completion. Its performance is also compared with the two best baseline models which are *Mean* and *NN[s]*. All models are evaluated under three different minimum duration conditions (e.g., > 180 seconds, > 360 seconds and > 600 seconds). Six

(6) datasets are constructed from the two evaluating arrival and departure datasets as described in Table III. Since the discussions involve the policy's behaviors in different conflict (air-side) situations, from now onward, we will use "the owner" to refer to the target aircraft and "the intruder" to refer to the other surrounding aircraft involved in the conflict/collision situation.

TABLE III
THE COMPARISON OF SPATIAL COMPLETION BETWEEN THE POLICY AND BASELINE MODELS FOR THREE DATASETS

	Index	Name	Duration	Size
Arrival	Dataset 1	A1	> 180s	2787
	Dataset 2	A2	> 360s	1097
	Dataset 3	A3	> 600s	35
Departure	Dataset 4	D1	> 180s	3281
	Dataset 5	D2	> 360s	1625
	Dataset 6	D3	> 600s	398

Behavior Learning

As one of the objectives is to develop a realistic taxi-speed model, the proposed model, referred to as the policy from now onward, must have the ability to reproduce hidden movement patterns or behaviors in data. Thus, different aspects of the simulated data (generated by the policy) are also investigated and compared to those of the actual data.

Action and Speed Distributions: Firstly, we will analyze action (accelerations) and aircraft speed distributions (Fig. 10) to assess the policy's behavior learning. As we observe from Fig. 10a and Fig. 10c, the action distributions of the policy and actual data are quite similar with some minor differences. The policy has more actions "1", "2", and "5" and fewer actions "3" compared to actual data. This observation may imply that instead of trying to maintain the speed profile with small variations (action "3"), our policy will generate more small fluctuations by combining high acceleration (action "5") with multiple decelerations (action "1" and "2") to approximate behaviors in data. In case of arrivals, the policy also uses slightly more action "0" ("brake") to control the aircraft speed than the actual data. Secondly, the aircraft speed is also another important factor in analysing the policy's behavior. Fig. 10b and Fig. 10d show the comparison of the speed distributions for both departures and arrivals. The distributions of the policy approximate normal distributions with mean values around 6m/s for departures and 8m/s for arrivals, while actual data has "skewed right" distributions. The policy has the tendency to maintain the speed around the mean value instead of big variations as in actual data. These discussed characteristics can also be observed from the generated speed profile in Fig. 9a where the gaps at low and high speeds between speed profiles are clearly visualized.

Responses to surrounding traffic: The policy has shown its ability to handle potential conflict to a certain level in complex traffic scenarios. The first evidence is the reported spatial completion, where several potential conflicts have been successfully resolved. In this part, an example is selected and discussed to demonstrate the policy's behavior in response to the surrounding traffic. Fig. 11 shows an interesting case

Fig. 10. Illustrations of the action and speed distribution of both actual and simulated data.

where the owner has to slow down to avoid a potential conflict with the intruder (the ahead aircraft). In this case, the intruder reduces their speed because they close the crossing and are ready to take a turning (Fig. 11). By understanding the situation, the owner starts slowing down from 6m/s to 2m/s while continuously monitoring the evolution of the conflict.

Fig. 11. A selected case to demonstrate the responses of trained model to surrounding traffic, in which a potential conflict occurs between the target aircraft (referred to as the owner) and an ahead aircraft (referred to as the intruder). (a) presents the spatial information where the owner is highlighted in red, the intruder is highlighted in green and blue ones are surrounding aircraft. (b) shows the speed profiles of these two aircraft and the corresponding related distance between them within 45 seconds before the end of the episode .

Responses to taxi-route geometry: Moreover, even in unimpeded taxi movements where the aircraft movements aren't affected by the traffic or any other interventions, aircraft speeds still vary depending on the taxi-route geometry. Both studies in [26], [27] have shown unique patterns in speed profiles at straight and turn segments. Those patterns are also observed in our data and successfully reproduced by the policy.

Fig. 12 visualizes selected movement patterns at a turn and a straight segment on the airside network. The segment with its geometry is highlighted in Fig. 12a. The taxi trajectories from the datasets (actual and simulated data by the policy and by the NN[s]), which passed by this segment, are selected to extract the movement patterns. The patterns are presented by the average speeds and their variation at each spatial location along the segment in Fig. 12b. As observed from the figure, aircraft start the segments with high speed, then slow down at the turn and speed up again. We can observe the similarity in behavior between the policy's simulated data and the actual data, which implies that the policy is able to learn and imitate actual aircraft movement. On the other hand, the baseline model (NN[s]) shows only constant speeds over the segment. Besides, for the policy, we observe that the owner enters the segment with a lower speed and maintains less variation in speed comparing actual data. It is also an example of what we have discussed on the policy's overall behaviors on the action and speed distributions. A straight segment (length = 950 m) is selected as highlighted in Fig. 12c and Fig. 12d shows the corresponding movement patterns through that segment which are computed using the same techniques as above. In observed patterns, aircraft start with low speeds around $5.5m/s$; they increase their speeds, reach the peak at the ratio = 0.7 ($\approx 665m$); and finally, slow down before the end of the segment. The main difference (Fig. 12d) between them is gap in maximum speed ($\approx 1.5m/s$) at their peaks. The policy also shows similar in movement pattern with lower speeds after ratio > 0.6 ($\approx 570m$).

Spatial Completion:

The policy successfully controls aircraft, by providing appreciated taxi-speeds, to travel from starting position to target position while considering multiple operational constraints such as separation distances, target take-off time, etc. It completes 88.3% (for departure) and 97.1% (for arrival) of the evaluated episodes where the aircraft can finish the assigned taxi-route without falling into any absorbing states (e.g., collision with relative distance less than 60m).

Table IV shows the spatial completion of three models with different datasets. For arrival, the proposed model achieve the spatial completion up to 97.1% comparing to 93.3% and 90.1% of baseline models. The gaps become significant when the selected sessions are longer (e.g., A2 and A3). The policy still achieves more than 84.4% while the others only achieve less than 70%. For departure, all the models have less spatial completion comparing to arrival ones that implies that the departure sessions are more challenging than the arrival ones. However, the policy still achieves 88.3% and outperforms the baseline models with significant margins ($\approx 30\%$) with D3

Fig. 12. Illustration of the similarity in aircraft behaviors between the policy and actual data.

(duration more than 10 minutes). The reductions in spatial completions with long episodes have emphasized the complexity for controlling the aircraft to reach the target point. By contrasting to the baseline models, the proposed model has demonstrated a good performance by maintaining high spatial completions for challenging episodes.

TABLE IV
THE COMPARISON OF SPATIAL COMPLETION BETWEEN THE POLICY AND BASELINE MODELS FOR THREE DATASETS

	Arrival			Departure		
	A1	A2	A3	D1	D2	D3
NN[s]	90.1%	83.9%	54.7%	75.4%	60.8%	46.1%
Mean	93.3%	90.0%	68.8%	68.0%	59.1%	42.1%
Policy	97.1%	95.9%	84.4%	88.3%	82.2%	72.2%

Moreover, by manually investigating the incompleted cases, we figured out that there are two main reasons behind all of the collisions. The first reason is the noise in aircraft' positions. In some instances, the noises still exist in the data for surrounding aircraft' positions even though multiple cleaning and pre-processing steps have been performed. It makes the intruder block or collide accidentally with the owner. It implies a better pre-processing technique for taxi movement data should be studied. The second reason comes from the non-cooperative movements of surrounding aircraft that lead to collisions due to unexpected behaviors of the intruder. Fig. 13 shows an example of an incomplete episode where a conflict happens between the owner and the intruder (the following aircraft).

Fig. 13. An incomplete case in which a collision occurs between the target aircraft (referred to as the owner) and a following aircraft (referred to as the intruder). (a) presents the spatial information where the owner is highlighted in red, the intruder is highlighted in green and blue ones are surrounding aircraft. (b) shows the speed profiles of these two aircraft and the corresponding related distance between them within 20 seconds before the collision.

The policy navigates the aircraft on the straight segment and there is an aircraft close to the taxi-way, so the agent keeps the stable speed at 8m/s (Fig. 13a)). However, the intruder suddenly increases its speed (refer to Fig. 13b) from 10m/s to 15m/s and results in a collision (relative distance less than 60 meters). The policy fails to capture the case and anticipate the intruder's behavior, leading to delayed and inappropriate responses. Since we only control the owner with limited action space, the responses to the surrounding aircraft' abnormal behaviors are constrained.

As the objective of the study is to develop a realistic taxi-speed model, the simulation is designed to preserve pilot's behaviours in taxing the aircraft by reproducing realistic airside traffic from historical data. The current architecture of the simulator is based on single-agent architecture design which implies that the simulation environment is non-cooperative, with only one agent acting on the learned model and navigating the airside environment with other aircraft. The proposed design is also optimized for facilitate the computational time for the model's training. Therefore, simulating all aircraft with full autonomous taxing capabilities (i.e., multi-agent configuration) requires a different architecture, which we are considering for our future work.

Spatial and Temporal Completion:

The "temporal" aspect of the proposed model will be analysed and discussed herein by comparing simulated travel-time of models, the policy and the selected baselines (refer to Table V). The travel-time errors are only calculated for spatial completed episodes of each model. Even though the two sets of episodes would be different, most of the spatial completed episodes of the baselines are covered in the policy's set that ensures the validity of the following arguments. Firstly, the

Root Mean Square Error (RMSE) is computed from travel-time errors for all models. The proposed model shows a robust performance with RMSE varies between 32.4 and 38.1 seconds for departure, and between 16.8 and 21.1 seconds for arrival. While the values for RMSE of the baselines increase significantly with the increase of episode's length. For instant, the RMSE of *NN[s]* model increases from 62 to 114 seconds for departure and from 60 to 179 seconds for arrival. Moreover, Mean Absolute Percentage Error (MAPE) values are also considered to have a better understanding of the magnitude of travel-time errors. The MAPE of proposed model for departure linearly decrease from 7.6% to 4.3% corresponding to the increases of taxi time while its MAPE slightly reduces from 4.4% to 2.7% for arrival. Meanwhile, the baselines have high MAPE (12% – 15%) for departure. In case of arrival, their MAPE values even increase significantly with long episodes (A3).

Briefly, the policy, which have learnt the actual movement patterns, have higher performance in modelling taxi-speed for controlling taxiing aircraft to complete assigned taxi-routes in both "spatial" and "temporal".

TABLE V
THE EXPERIMENT RESULTS FOR TEMPORAL COMPLETIONS FOR DATASETS WITH DIFFERENT TAXI LENGTHS.

	Arrival					
	Policy		Mean		NN[s]	
	RMSE	MAPE	RMSE	MAPE	RMSE	MAPE
A1	16.8s	4.4%	60.2s	14.9%	60.7s	14.4%
A2	17.4s	3.1%	70.7s	12.3%	76s	13.0%
A3	21.1s	2.7%	147.1s	18.8%	179.9s	24.3%

	Departure					
	Policy		Mean		NN[s]	
	RMSE	MAPE	RMSE	MAPE	RMSE	MAPE
D1	32.4s	7.6%	70.2s	15.3%	62.6s	12.9%
D2	33.8s	5.2%	85.0s	12.9%	85.0s	12.2%
D3	38.1s	4.3%	111.6s	12.2%	114.5s	12.8%

VII. CONCLUSION

In this study, we adopt the Generative Adversarial Imitation Learning (GAIL) algorithm for aircraft taxi-speed model while considering operational factors including surrounding traffic and target take-off time. The movement patterns in data is then learned and reproduced by the proposed model under different circumstances. Moreover, the characteristics of the proposed model is also analysed, specially focusing on handling conflict scenario with surrounding traffic. Finally, the travel time of the target aircraft from starting positions to target positions are recorded and compared with actual data. This model may serve as a spatio-temporal taxi trajectory predictor for taxiing aircraft or a aircraft movement module in a ground movement simulator.

The effectiveness of learning by the model (the trained model) also called the policy, is shown by generating similar action distributions and speed distributions as observed in actual data. Moreover, some examples have been selected to show the policy's behavior in terms of responding to surrounding aircraft and responding to different geometry of route segments.

The results demonstrate that the policy outperforms all the baseline models with a significant margin. In terms of spatial completion (SC), it achieves 97.3% for arrivals and 88.2% for departures. The gaps between the policy and baselines become more significant when the minimum sessions' duration are increased. It means that the performance of the policy is higher and more stable than the other algorithms in successfully controlling the target aircraft without falling into any "invalid" absorbing state. Besides, the policy has significantly high performance for temporal completion. The policy achieves a stable performance with low RMSE with less than 21.1 seconds for arrivals, 38.1 seconds for departures for all evaluated datasets. Its MAPE values within the range 2.7% - 4.4% for arrivals and 4.3% - 7.6% for departures. Meanwhile, all baseline models have high RMSE values (> 60 seconds) for the best case and increase from 1.7 to 2.5 times when the minimum sessions' duration are increased. The policy's errors are at least 72% lower for arrivals and 48% lower for departures when compared to other baseline models.

For future work, we will be extending the proposed framework to a multi-agent environment, in which a centralized conflict resolution algorithm will be developed. The future extension is expected to fully model and simulate the airport airside environment.

VIII. ACKNOWLEDGEMENT

This research is supported by the National Research Foundation, Singapore, and the Civil Aviation Authority of Singapore, under the Aviation Transformation Programme. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore and the Civil Aviation Authority of Singapore.

REFERENCES

- [1] L. Adacher, M. Flamini, and E. Romano, "Airport ground movement problem: minimization of delay and pollution emission," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 12, pp. 3830–3839, 2018.
- [2] K. G. Zografos, M. A. Madas, and K. N. Androustopoulos, "Increasing airport capacity utilisation through optimum slot scheduling: review of current developments and identification of future needs," *Journal of Scheduling*, vol. 20, pp. 3–24, 2017.
- [3] H. Lee, "Airport surface traffic optimization and simulation in the presence of uncertainties," Ph.D. dissertation, Massachusetts Institute of Technology, 2014.
- [4] L. Yang, S. Yin, K. Han, J. Haddad, and M. Hu, "Fundamental diagrams of airport surface traffic: Models and applications," *Transportation Research Part B: Methodological*, vol. 106, pp. 29–51, Dec. 2017. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0191261517303843>
- [5] S. Atkins and C. Brinton, "Concept description and development plan for the surface management system," *Journal of Air Traffic Control*, vol. 44, no. 1, pp. 1–8, 2002.
- [6] Z. Wood, S. Rathinam, Y. Jung, and M. Kistler, "A simulator for modeling aircraft surface operations at airports," in *AIAA Modeling and Simulation Technologies Conference*, 2009, p. 5912.
- [7] S. Zelinski and R. Windhorst, "Modelling and simulating airport surface operations with gate conflicts," *The Aeronautical Journal*, vol. 123, no. 1259, pp. 1–19, 2019.
- [8] A. R. Odoni, J. Bowman, D. Delahaye, J. J. Deyst, E. Feron, R. J. Hansman, K. Khan, J. K. Kuchar, N. Pujet, and R. W. Simpson, "Existing and required modeling capabilities for evaluating atm systems and concepts," Tech. Rep., 2015.
- [9] S. Atkins, Y. Jung, C. Brinton, L. Stell, T. Carniol, and S. Rogowski, "Surface management system field trial results," in *AIAA 4th Aviation Technology, Integration and Operations (ATIO) Forum*, 2004, p. 6241.
- [10] H. Balakrishnan and Y. Jung, "A framework for coordinated surface operations planning at dallas-fort worth international airport," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2007, p. 6553.
- [11] C. Brinton, J. Krozel, B. Capozzi, and S. Atkins, "Improved taxi prediction algorithms for the surface management system," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2002, p. 4857.
- [12] G. J. Couluris, R. K. Fong, M. B. Downs, N. Mittler, D. Signor, A. Stassart, and T. Hsiao, "A new modeling capability for airport surface traffic analysis," in *2008 IEEE/AIAA 27th Digital Avionics Systems Conference*. IEEE, 2008, pp. 3–E.
- [13] H. H. D. Khadilkar, "Networked control of aircraft operations at airports and in terminal areas," Ph.D. dissertation, Massachusetts Institute of Technology, 2013.
- [14] I. Simaiakis, M. Sandberg, and H. Balakrishnan, "Dynamic Control of Airport Departures: Algorithm Development and Field Evaluation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 1, pp. 285–295, Feb. 2014. [Online]. Available: <http://ieeexplore.ieee.org/document/6589157/>
- [15] I. Simaiakis, "Analysis, modeling and control of the airport departure process," Ph.D. dissertation, Massachusetts Institute of Technology, 2013.
- [16] H. Lee, I. Simaiakis, and H. Balakrishnan, "A comparison of aircraft trajectory-based and aggregate queue-based control of airport taxi processes," in *29th Digital Avionics Systems Conference*. IEEE, 2010, pp. 1–B.
- [17] R. D. Windhorst, J. V. Montoya, Z. Zhu, S. Gridnev, K. Griffin, A. Saraf, and S. Stroiney, "Validation of simulations of airport surface traffic with the surface operations simulator and scheduler," in *2013 Aviation Technology, Integration, and Operations Conference*, 2013, p. 4207.
- [18] V. H. Cheng, V. Sharma, and D. C. Foyle, "A study of aircraft taxi performance for enhancing airport surface traffic control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 2, no. 2, pp. 39–54, 2001.
- [19] I. Gerdes and M. Schaper, "Management of time based taxi trajectories coupling departure and surface management systems," in *11th ATM Seminar*, 2015.
- [20] R. Mori, "Aircraft ground-taxiing model for congested airport using cellular automata," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 1, pp. 180–188, 2012.
- [21] J. N. Okunieuk, I. Gerdes, J. Jakobi, T. Ludwig, B. L. Hooey, D. Foyle, Y. C. Jung, and Z. Zhu, "A concept of operations for trajectory-based taxi operations," in *16th AIAA Aviation Technology, Integration, and Operations Conference*, 2016, p. 3753.
- [22] S. Rathinam, J. Montoya, and Y. Jung, "An optimization model for reducing aircraft taxi times at the dallas fort worth international airport," in *26th International Congress of the Aeronautical Sciences (ICAS)*, 2008, pp. 14–19.
- [23] J.-B. Gotteland, N. Durand, J.-M. Alliot, and E. Page, "Aircraft ground traffic optimization," in *ATM 2001, 4th USA/Europe Air Traffic Management Research and Development Seminar*, 2001, pp. pp–xxxx.
- [24] FAA. Simmod manual. [Online]. Available: http://www.tc.faa.gov/acb300/how_simmod_works.pdf
- [25] J. V. Montoya, R. D. Windhorst, S. Stroiney, K. Griffin, A. Saraf, Z. Zhu, and S. Gridnev, "Analysis of airport surface schedulers using fast-time simulation," in *2013 Aviation Technology, Integration, and Operations Conference*, 2013, p. 4275.
- [26] C. Gong, "Kinematic airport surface trajectory model development," in *9th AIAA Aviation Technology, Integration, and Operations Conference (ATIO) and Aircraft Noise and Emissions Reduction Symposium (AN-ERS)*, 2009, p. 7076.
- [27] T.-N. Tran, D.-T. Pham, S. Alam, and V. Duong, "Taxi-speed prediction by spatio-temporal graph-based trajectory representation and its applications," in *The 9th International Conference for Research in Air Transportation (ICRAT)*, Tampa, Florida, USA, 2020.
- [28] S. Schaal et al., "Learning from demonstration," *Advances in neural information processing systems*, pp. 1040–1046, 1997.
- [29] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.
- [30] D. Pomerleau, "An autonomous land vehicle in a neural network," *Advances in Neural Information Processing Systems; Morgan Kaufmann Publishers Inc.: Burlington, MA, USA*, 1998.

- [31] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings*, 2011, pp. 627–635.
- [32] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 1.
- [33] C. Finn, S. Levine, and P. Abbeel, "Guided cost learning: Deep inverse optimal control via policy optimization," in *International conference on machine learning*. PMLR, 2016, pp. 49–58.
- [34] A. Y. Ng, S. J. Russell *et al.*, "Algorithms for inverse reinforcement learning," in *ICML*, vol. 1, 2000, p. 2.
- [35] S. Ross and D. Bagnell, "Efficient reductions for imitation learning," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings*, 2010, pp. 661–668.
- [36] R. Hadsell, P. Sermanet, J. Ben, A. Erkan, M. Scoffier, K. Kavukcuoglu, U. Muller, and Y. LeCun, "Learning long-range vision for autonomous off-road driving," *Journal of Field Robotics*, vol. 26, no. 2, pp. 120–144, 2009.
- [37] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2722–2730.
- [38] T. Fernando, S. Denman, S. Sridharan, and C. Fookes, "Going deeper: Autonomous steering with neural memory networks," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2017, pp. 214–221.
- [39] A. Kuefler, J. Morton, T. Wheeler, and M. Kochenderfer, "Imitating driver behavior with generative adversarial networks," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 204–211.
- [40] J. Morton and M. J. Kochenderfer, "Simultaneous policy learning and latent state inference for imitating driver behavior," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2017, pp. 1–6.
- [41] C. Finn, P. Christiano, P. Abbeel, and S. Levine, "A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models," *arXiv preprint arXiv:1611.03852*, 2016.
- [42] D. S. González, J. S. Dibangoye, and C. Laugier, "High-speed highway scene prediction based on driver models learned from demonstrations," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2016, pp. 149–155.
- [43] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan, "Planning for autonomous cars that leverage effects on human actions," in *Robotics: Science and Systems*, vol. 2. Ann Arbor, MI, USA, 2016.
- [44] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Advances in neural information processing systems*, 2016, pp. 4565–4573.
- [45] OpenStreetMap contributors, "Planet dump retrieved from <https://planet.osm.org>," <https://www.openstreetmap.org>, 2017.
- [46] G. Boeing, "Osmnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks," *Computers, Environment and Urban Systems*, vol. 65, pp. 126–139, 2017.
- [47] T.-N. Tran, D.-T. Pham, and S. Alam, "A map-matching algorithm for ground movement trajectory representation using a-smgcs data," in *International Conference on Artificial Intelligence and Data Analytics in Air Transportation (AIDA-AT) 2020*, 2020.
- [48] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.
- [49] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Computing Surveys (CSUR)*, vol. 50, no. 2, pp. 1–35, 2017.
- [50] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, "Generative adversarial nets," in *NIPS*, 2014.
- [51] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.
- [52] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 41–48.
- [53] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*, 2015, pp. 1889–1897.
- [54] I. Kostrikov, K. K. Agrawal, D. Dwibedi, S. Levine, and J. Tompson, "Discriminator-actor-critic: Addressing sample inefficiency and reward bias in adversarial imitation learning," in *ICLR*, 2019.
- [55] I. Kostrikov, "Pytorch implementations of reinforcement learning algorithms," <https://github.com/ikostrikov/pytorch-a2c-ppo-acktr-gail>, 2018.

Duc-Thinh Pham obtained his Ph.D. degree of Paris Science and Letter (PSL) in Computer Science, France, in 2019. He is a Senior Research Follow at Air Traffic Management Research Institute (ATMRI), Nanyang Technological University (NTU). His research interests include on deep reinforcement learning, and application of machine learning in air traffic management. He is the leader of data-driven surface movement management project at ATMRI, NTU.

Thanh-Nam Tran obtained his M.Sc. degree in Computer Science from University of Science, Viet Nam, in 2018. He is currently a Research Associate at ATMRI, NTU. His researches focus on machine learning, deep reinforcement learning and its application on surface movement management.

Sameer Alam is an Associate Professor at the School of Mechanical and Aerospace Engineering, NTU, Singapore. He received Ph.D. degree (2008) in Computer Science with specialization in Artificial Intelligence from University of New South Wales (UNSW) Australia. He is the Deputy Director of ATMRI, NTU. He is also the editorial board member of Transportation Research Part-C. His research interests are in Machine Learning, Computer Vision, Multi-Agent Systems, applied to Air Traffic and Airport operations.

Vu N. Duong is a Professor and Director of Air Traffic Management Research Institute (ATMRI), School of Mechanical & Aerospace Engineering, Nanyang Technological University, Singapore. Vu had also been Head of Innovative Research then Senior Scientific Advisor at EUROCONTROL (1995-2012), and a member of SESAR JU Scientific Committee (2010-2012). He obtained his Ph.D. in Artificial Intelligence from Ecole Nationale des Ponts et Chaussees, France in 1990.