FAST-TIME SIMULATION OF AIRPORT SURFACE MOVEMENT

Yao Lu (1) Chengyongping Lu (1) Shirley Su (1) Yi Wang (1) Yurui Wang(1) Shuang Weng (1)
Robert Morris (2) Corina S. Păsăreanu (2) Christopher DiPrima (3)

(1) Information Networking Institute, Carnegie Mellon University Silicon Valley
(2) Researcher, Intelligent Systems Division, NASA Ames Research Center
(3) Airport Planner, Bureau of Planning and Environment Affairs, San Francisco International Airport

ABSTRACT

Fast-time simulation of airport surface operations allows for the thorough testing and analysis of modeling concepts and algorithms for managing aircraft movement. However, airport surface movement operations present a difficult, large-scale logistics problem. This paper summarizes the results of a multi-year effort to build a fast-time simulator of large airport surface movement, using San Francisco Intrnational Airport as a case study.

I. INTRODUCTION

A fast-time simulator of airport capacity compresses time so that data can be generated more quickly to allow reliable evaluations of new solutions and optimizations of existing solutions to capacity problems more quickly. In this paper we use fast-time simulation of airport surface operations for testing and analysis of modeling concepts and algorithms for controlling aircraft movement. This work is motivated in part by recent assessments [1] of the potential benefits of improving the predictability of movement in efficiency of airport operations. This work builds upon research in Integrated Arrival, Departure and Surface Operations (IADS), including the ATD-2 System developed at NASA [2].

This paper reports the results of a multi-year effort at building capabilities for fast-time simulation of large airports with complex logistical challenges. San Francisco airport (SFO) has been selected as a case study in this work. The paper presents a summary of the overall approach, the system component models and algorithms, and the tools for visualization and analysis.

II. OVERVIEW OF SIMULATOR ARCHITECTURE

We present a fast-time simulator for analyzing the performance of automation for complex airports. The technical approach in building the simulator emphasizes a high fidelity, realistic model of the movement of departing and arriving aircraft. To realize this goal we specifically focus on modeling movement uncertainty. Airport surface dynamics is unpredictable and prone to unexpected changes in operating conditions due to external factors such as weather.

Airports seek to optimize the traveler's comfort and safety through efficient and safe operations. These are difficult objectives to achieve in practice, due to the challenges posed by the presence of uncertainties, human factors, and competing stakeholder interests. The goal of the simulator is to quantify important performance metrics, specifically related to delay, in order to compare models and algorithms that could be applied to future automated tools deployed at airports.

The simulator architecture is summarized in Figure 1. The inputs to the simulator consist of the following. First, an airport *node-link model* uses latitude and longitude to represent locations on the surface, and links to represent traversable surfaces. A separate graphical model used by the scheduler was developed using only landmark nodes, including gates, spots, taxiway intersections and runway nodes. Second, a *scenario* describes a problem instance to the system, consisting of a set of arrivals and departure information, including scheduled arrival or departure times, as well as gate and runway information. Finally, a set of *numeric parameters* are used to customize the behavior of the simulator components, including uncertainty parameters, that define the degree of uncertainty on the surface; clock parameters that define the speed of the simulation; and test parameters that allow for a range of statistics to be outputted.

The simulator consists of a collection of algorithms and models that allow for the generation of a *history*: a sequence of states that represent the evolution of all aircraft movement along an airport surface over a defined period of time.

Models describe rules that govern movement. A *state model* defines the features of interest on the airport surface at a moment of time. It is used by the scheduler to generate a schedule, and by the simulator to record a history of events that happen during the simulation.

A *dynamics model* consists of rules for defining the speed of the aircraft along different parts of the surface. For example, the dynamics models defines how fast aircraft push back from a gate, move in the terminal area, and on the taxiway. A *controller* model provides a set of rules that simulate how human controllers or pilots maintain safe conditions during operations by anticipating and responding to

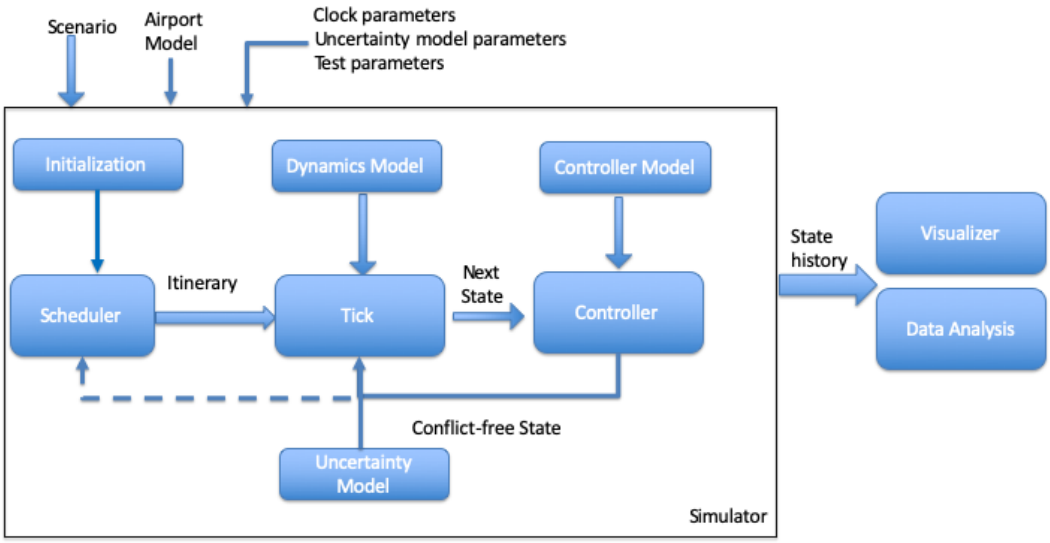

Fig. 1. Airport Surface Planner and Simulation Architecture

potentially unsafe conditions. Finally, an *uncertainty model* defines parameters that allow for the injection of unexpected changes to airport movement (for example, delays at gates) during simulation.

The output of the simulator is a *history*: a sequence of world states (states of the airport) that shows the evolution of aircraft movement. This output can be sent to a *visualizer* which can animate the movement, and to an *analysis tool* to produce statistics that will guide further testing and algorithm development.

The simulator was implemented in Python. The airport visualizations used JavaScript and CSS. To represent the airport geographic information, KML files were exported from Google Map in order to construct the node-link model.

The remaining sections describe the three core models and algorithms (scheduler, state and dynamics, control) in more detail.

II-A. Scheduler

Airport surface movement scheduling present a difficult, large-scale logistics problem with a wide range of sub-problems, including: runway sequencing and scheduling; spot or gate release scheduling; gate allocation and taxi route planning and scheduling [3], [4], [5], [6]. In our work we have thus far focused on gate release and taxi route planning and scheduling for departing flights. This work provides the basis for eventually building an integrated scheduler of all airport movement, including movement in the terminal airspace, and builds upon previous work such as [7]

As a starting point, a departure scheduler is required to generate a path on the airport surface from gate to runway, as well as a release time, for a set of departing flights in a scenario. In real operations, scheduling is performed for a set of flights over a fixed horizon in the future (e.g. flights scheduled to depart in the next hour). We simplify the model as follows. Let us begin with the generation of a *Master Schedule*, a complete set of trajectories + release times for a single day. In the simplest formulation, the scheduler of the master schedule uses pre-assigned paths from gates to runways, generated offline using airport-assigned paths. Gate release times may be initialized to scheduled release times as provided by airlines. In this simplified model, arriving flights are not scheduled, but rather appear as constraints on the overall departure schedule that are needed to maintain safe operations.

A master schedule can be revised in order to resolve *conflicts* or through rescheduling (as indicated in Figure 1 by the dotted link). By conflict is meant any indicator of potential surface congestion. These are discussed in more detail below.

A third option for managing uncertainty in scheduling would be to augment the scheduler in ways that that enables it to interleave the search for a master schedule with resolving conflicts. Recent advances in multi-agent planning can be applied to search for an optimal conflict-free schedule [8]. The result might be a schedule that is more robust to conflicts caused by uncertainty during operations. This idea is discussed in previous work by the same authors [9]. Schedulers that incorporate models of the dynamics of airport movement can also potentially be used to manage autonomous taxiing operations [10].

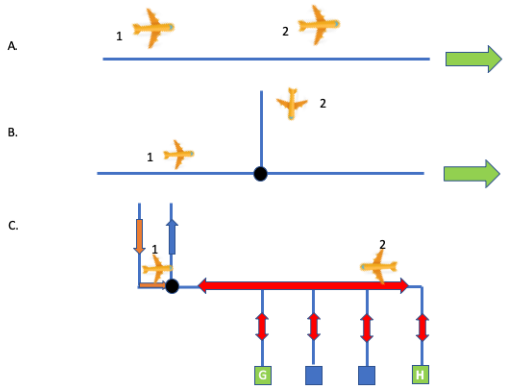

Fig. 2. Types of potential conflict. A: unsafe distance; B: Intersection approach; C1,C2: Terminal conflicts

II-B. Controller

The primary task of a controller is to manage conditions that could result in an unsafe state. For the purpose of simulator we abuse terminology a little by saying a controller could be a pilot or autonomous pilot, an ATC controller, or a terminal controller. The action performed by a controller might be direct (e.g. a pilot decelerating an aircraft), or indirect (commanding a human or automated system to do something).

Constraints imposed on the flow of aircraft on an airport surface help to simplify the types of conflict that may arise. Specifically, all conflicts can be reduced to three, each illustrated in Figure 2: unsafe distance (A), intersection approach (B) and terminal area (C).

Case A is commonly handled by the pilot. Maintaining safe distance can be modeling by applying car-following models used for self-driving cars [11].

In case B, a potential conflict can be detected when the aircraft together are at a certain distance from (within a certain radius of) the intersection. To resolve this conflict, we can assume that at the point of detection we can assign a priority that determines the right-of-way for the aircraft.

In case C, one or more arriving aircraft is approached a terminal area containing its gate. One or more departing aircraft are at various stages of pushback and approaching the exit of the terminal area. Again there is an intersecting node (black dot) and potential for nose-to-nose conflict. A terminal controller resolves potential conflicts in the terminal area by imposing a safe ordering of the aircraft through the intersecting node. This ordering must be continually updated as new aircraft enter or leave the terminal area.

II-C. State and Dynamics Model

A *state* is a description of the world in a moment of time; in the simulator, a state is a set of vectors indicating the location and speed of each active aircraft (i.e., aircraft that have entered and not yet exited the system), as well as the trajectory followed by the aircraft.

A state model contains classification of states that allow for features of interest to be quantified. Thus a departing flight might be in one of the set: *AtGate, pushback, terminal, taxi, runway queue* states; arriving aircraft have similar states. A region on the airport surface is associated with each state. Each state type can be associated with different speeds that make up a speed model. Transitions between states (e.g. from pushback to terminal states) may correspond to a change in speed calculated from an acceleration model.

II-D. Visualization and Analysis

The output of the simulator can be fed into a visualizer for viewing. The user selects a data source (history) and the airport surface map and data are loaded into the visualizer. Icons for gates, pushback ways, spots, taxiways, and runways are depicted. Other color-coded icons for active aircraft flow along the surface image when the simulator is run. Each icon can be clicked to provide information about that item. The simulation can be started and stopped at any time, and aircraft icons can be clicked to reveal its state information, including speed and state status. A dashboard to the left of the surface image shows the call sign of all active aircraft and their current status. See Figure 3 for a snapshot of the visualizer running.

Aircraft state data generated from the scheduler model are used to analyze the airport's scheduling performance. Currently, source data are extracted, transformed using python scripts, and loaded into AWS S3 buckets for data persistence. Data analysis is visualized as a Quicksight report where several metrics are calculated to evaluate the airport's load distribution, capacity and performance.

III. SUMMARY

Researchers at NASA and other research institutions have been developing systems for decades to improve the aircraft surface movement at busy airports. This paper has described an on-going project building a a system for simulation, visualization, scheduling, modeling and analysis of aircraft surface congestion at large complex airports.

IV. REFERENCES

[1] B. Aponso1, R. Coppenbarger, Y. Jung, L. Quon, G. Lohr, N. OConnor, and S. Engelland, "Identifying key issues and potential solutions for integrated arrival, departure, surface operations by surveying stakeholder preferences," *15th AIAA Aviation Technology, Integration, and Operations Conference*, 2015.

[2] A. Ging, S. Engelland, A. Capps, M. Eshow, Y. Jung, S. Sharma, E. Talebi, M. Downs, C. Freedman, T. Ngo, H. Sielski, E. Wang, J. Burke, S. Gorman, B. Phipps, and L. Ruszkowski, "Airspace technology demonstration 2 (atd-2) technology description document (tdd)," *NASA/TM-2018-219767*, 2018.

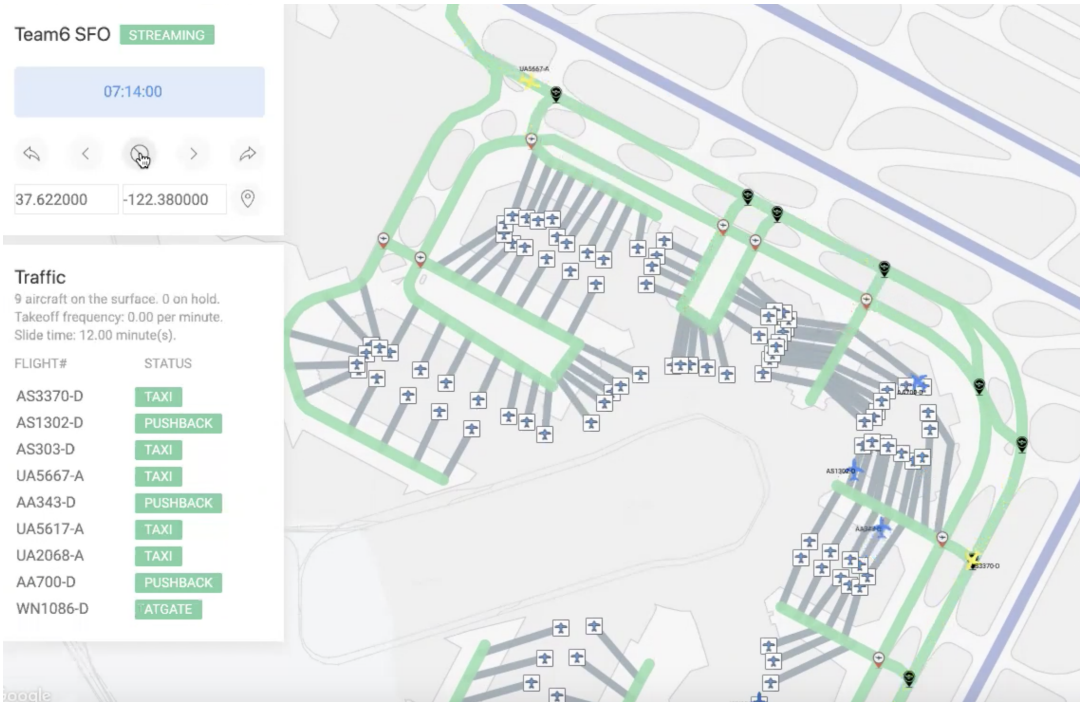

Fig. 3. Visualization of Simulator Output Showing a Surface Map of San Francisco International Airport

[3] W. Malik, G. Gupta, and Y. . Jung, "Spot release planner: Efficient solution for detailed airport surface traffic optimization," *Proceedings of the 12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*, 2012.

[4] S. Rathinam, Z. Wood, B. Sridhar, and Y. Jung, "A generalized dynamic programming approach for a departure scheduling problem," *AIAA Guidance, Navigation, and Control Conference*, 2009.

[5] J. Ravizza, S. amd Atkin and E. Burke, "A more realistic approach for airport ground movement optimisation with stand holding," *Journal of Scheduling*, vol. 17, pp. 507—-520, 2014.

[6] P. C. Roling and H. G. Visser, "Optimal airport surface traffic planning using mixed-integer linear programming," *International Journal of Aeronautical Engineering*, vol. 1, pp. 1–11, 2008.

[7] A. Capps, M. Kistler, and S. Engelland, "Design characteristics of a terminal departure scheduler," in *Proceedings of the 14th AIAA Aviation Technology, Integration and Operations Conference*, 2014.

[8] H. Ma and S. Koenig, "Optimal target assignment and path finding for teams of agents," *Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2016)*, 2016.

[9] J. Li, H. Zhang, M. Gong, Z. Liang, W. Liu, Z. Tong, L. Yi, R. Morris, C. Pasareanu, and S. Koenig, "Airport taxiway path planning under uncertainty," in *Proceedings of the AIAA Aviation Forum and Exposition (AIAA)*. AIAA Aviation, 2019.

[10] R. Morris, M. L. Chang, R. Archer, E. V. C. II, S. Thompson, J. L. Franke, R. C. Garrett, W. Malik, K. McGuire, and G. Hemann, "Self-driving towing vehicles: A preliminary report," in *Proceedings of the Workshop on AI for Transportation (WAIT)*, 2014, pp. 35–42.

[11] M. Saifuzzaman and Z. Zheng, "Incorporating human-factors in car-following models: A review of recent developments and research needs," *Transportation Research Part C: Emerging Technologies*, vol. 48, pp. 379–403, 2014.