Simulation of Demand and Supply of Urban Rail in a Multimodal Environment

Kenneth Koh*, Carlos Lima Azevedo*, Kakali Basak† and Moshe E. Ben-Akiva*
*Massachusetts Institute of Technology, Cambridge, 02139 MA, United States
{kkoh3, cami, mba}@mit.edu
†Singapore MIT Alliance for Research and Technology, 138602 Singapore
kakali@smart.mit.edu

***Abstract*—High-frequency rail transit systems play a key role in dense urban transportation. As such, decisions made in design, implementation, and operation may precondition the successful performance of the overall urban system. To fully understand the impacts of these decisions there is a need for an integrated analysis, where rail is analyzed as a component of a larger and more complex transportation system. In this paper we present a new rail simulator as an integrated component of a comprehensive agent-based mobility simulation platform. By leveraging a multi-modal activity-based demand formulation and mesoscopic dynamic network supply simulation, our proposed design aims primarly to capture the impacts of complex dynamics between individual traveler behavior, the design and operation of a rail system, and the performance of all other modes. A second aim is flexibility; through the implementation of a flexible architecture, the simulation of a wide variety of different rail operational configurations and scenarios becomes feasible. As a demonstration of capabilities the simulator is used to replicate the operations and performance of a fully automated Mass Rapid Transit line in Singapore. The simulator was calibrated using a novel sequential calibration method with only automatic fare card data, without requiring automatic train control or manually collected data, and validated with another dataset. The proposed rail simulator can be used for the assessment of different rail control strategies under regular and disruption conditions, and to evaluate the performance of the road and other public transportation networks and users.**

Submitted to the IEEE 20th International Conference on Intelligent Transportation Systems, Yokohama, Japan, 16-19/10/2017

I. Introduction

Mass transit plays a key role in dense urban transportation systems and more than 50% of residents in major cities such as Hong Kong, Singapore, Tokyo, and London, rely on public transport for their daily commute [1]. As such, the choices made in the design, implementation, and operation of rail networks are of considerable importance. Understanding and analyzing the underlying relationship between demand and supply in these systems is therefore a crucial factor in the decision process. A natural approach to this analysis is through simulation, which allows for the examination of a multitude of potential scenarios. However, in complex urban systems, it is important to use integrated mobility simulators which not only capture dynamics within the rail system, but also within other modes of transport and individual travel patterns. That is, travel demand across modes should affect the rail system performance, and changes across the many features of a multimodal supply network should affect the demand for rail. There is a reasonable amount of prior work involving the development of train simulators to investigate various problems such as train scheduling, infrastructure design or disruptions. Simple macroscopic simulators such as Bahn [2] and MetroModSim [3] have been developed in the past to provide simplified representations of rail networks; in these simulators, trains move at deterministic speeds, with minimal signal controls such as linked pairs of stop/go signals. On the other end of the spectrum, more recent commercial simulators such as OpenTrack [4], a microscopic time-based railway simulator, provide a comprehensive model of train movement across long distances. However, they (1) tend to not focus on demand-side effects, (2) are commonly used in long-distance freight transportation planning and (3) are not capable of multi-modal simulation analysis. Multi-modal simulators, such as AIMSUN and VISSIM [5][1], provide detailed road traffic simulation with limited rail capabilities often constrained by traffic flow and road vehicle motion models. Both packages allow for integration with static demand models but lack breadthwise dynamic integration with all forms of transit and associated infrastructure, and dynamic changes in demand and supply operations. In academia, rail simulators are often developed to investigate various supply-targeted policies. For example, Grube et al. [6] and Sanchez-Martinez et al. [7] both created discrete event-based simulators to increase the operational efficiency of high-frequency rail transit by implementing various holding strategies. Cha et al. [8] constructed a similar model for *Maglev* trains to investigate how varying passenger arrival rates affect system performance. Others utilize rail simulation in operations research, such as to optimize bus-based disruption recovery strategies in subways ([9], [10]). However, these models are often customized to the topic at hand or mostly data driven, and hence serve mainly as a tool to demonstrate a very specific operational strategy. For example, [6] and [7] assume constant train speeds and ignore operational restrictions (such as safe distances), while [11] utilizes historical train running time data instead of actively simulating train movement. [8] utilizes a linear dwell time based off historical data, but ignores the effects of induced passenger congestion. [9] and [10] only consider deterministic choice (that is, when the subway breaks down, passengers automatically take the substitute bus) and ignore the possibility of individual agents revising their mode or destination choice due to service disruptions. Despite these limitations, these efforts resulted in key contributions to the state-of-the-art and together form the perfect foundation for the design of a multi-purpose generic rail simulator. Multi-modal simulators do exist in academia. MATSim is an advanced integrated multimodal simulator that generates demand from a synthetic population and an activity-based model[2] [12]. However, it also focuses primarily on road transit, with the intricacies of demand-side effects (such as dwell time) notably absent from its embedded rail simulator. SUMO [13] is a microscopic time-based multimodal simulator that supports passenger mode choice; however, it lacks dynamic demand features (once a commuter makes a choice, such as mode, she/he will not change decision) and has limited rail simulation capabilities. Finally,

[1]Updates on the Aimsun and Vissim were retrieved on May 15, 2017, from https://www.aimsun.com/aimsun/new-features/ and http://vision-traffic.ptvgroup.com/en-uk/products/ptv-vissim/, respectively.

[2]Demand is derived from an individual's daily activity patterns.

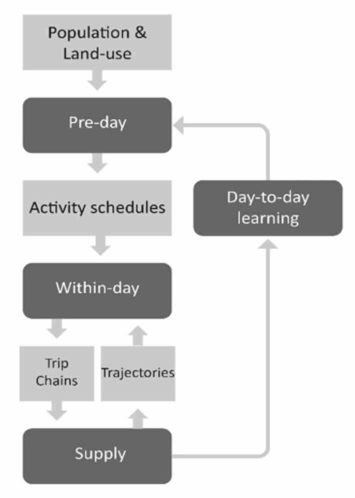

Fig. 1. Structure of the SimMobility *Mid-Term* [14]

SimMobility [13], a multi-level activity-based modeling platform in C++ also has interesting integrated simulation capabilities. It comprises three different simulation levels: the *Short-Term* simulator represents high temporal resolution (i.e. in the order of tenth of a second) events and decisions, such as vehicle lane-changing, braking and accelerating, individual pedestrian movement and agent to agent communication [25]. The *Mid-Term* simulator represents daily activity scheduling, mode, route, destination and departure time choices together with a dynamic multi-modal mesoscopic supply, and has a temporal resolution in the order of seconds or minutes [14]. Finally, the *Long-Term* simulator represents long-term choices such as house and job relocation or car ownership [15]. However, and contrary to its road and bus counterparts, a detailed rail supply model is absent in SimMobility. In summary, the current rail simulation state-of-the-art tends to macroscopically model demand (aggregated in terms of flows) and to ignore the impacts of changes in demand or supply of changes in other transportation modes. There is a need to develop a simulator where passengers decisions are modeled microscopically, capturing the different individual preferences and responses to system conditions, as well as the impacts that events occurring in the rail system have on the overall transportation network. A second consideration is developing a simulator flexible enough to simulate a wide range of different scenarios, thus reducing the need to custom-develop simulators in the future. In this paper, we fill this gap by presenting a comprehensive rail simulator integrated with the SimMobility *Mid-Term* simulator, an ideal overarching platform.

II. Framework within SimMobility Mid-Term

The *Mid-Term* contains many key functions, such as organic demand (as explained below), mode/route choice models and a road network supply time-based simulator, which allow for breadthwise integration of the rail supply. It also has a publish/subscribe mechanism [14], allowing for the development of a dynamic feedback loop between the rail system and passengers' decisions in runtime. Finally, the blank slate of the *Mid-Term* enables the rail simulator to be developed in a similarly flexible manner and allows it to easily simulate a wide range of scenarios and operating conditions. *Mid-Term* encompasses four modules schematized in Figure 1.

A synthetic population of potential travelers for a given network is generated in the *Long-Term* simulator and passed to the *Mid-Term* simulator. The demand comprises two groups of behavior models: (1) pre-day and (2) within-day.

1) The pre-day models follows an econometric Day Activity Schedule approach to decide an initial overall daily activity schedule of the agent, particularly its activity sequence (including tours and sub-tours), with preferred modes, departure times by half-hour slots, and destinations. This is based on sequential application of hierarchical discrete choice models using a monte-carlo simulation approach.
2) These schedules are then passed to the within-day module, which works in tandem with the (3) supply module transforming the activity schedule into effective decisions and execution plans. Through a publish/subscribe mechanism of event management agents may get involved in a multitude of decisions, not constrained to the traditional set of destination, mode, path and departure time depending upon their state in the event simulation cycle. Examples of triggers of this mechanism include excessive delay during the trip, specific incident or disruption information. Specific individual choice models are then called and result in rescheduling of the remainder of the day, canceling an activity, re-routing in the middle of a trip or changing mode. The time-step based supply side simply picks the latest plan information for each agent and executes it. A global events manager module checks, at each time step, which subscribed events occur and wakes up the corresponding agents. If an agent changes her/his plan due to an event, it will be reflected in the next time step, which will then be executed by the supply simulator.
3) The supply model follows a mesoscopic dynamic traffic assignment paradigm and provides feedback about the current system performance, which may cause a traveling agent to modify her/his schedules in response in the within-day module. For example, if a traveler becomes aware that the planned route is disrupted or unexpectedly congested, she/he may instead trigger the decision to take a taxi to the destination.
4) Finally, in the (4) day-to-day learning module, individuals 'learn' from their experiences and modify their choices in the subsequent runs of the *Mid-Term* simulator.

The reader is referred to [13] and [14] for the detailed description of the four modules.

In SimMobility, there is a distinction between Agents and Entities. They are both modeled microscopically with unique associated parameters; however, the former are vested with decision-making abilities, while the latter make no decisions whatsoever. For example, traveller-agents decide on daily activity schedules, trips, mode choices, and route choices in the pre-day module, and have an associated 'role' corresponding to their current action (car passenger, bus traveller, driver, etc). If a traveler selects public transportation as the mode for a given trip, a dedicated route-choice is called by the simulator. In our proposed integration, for any legs of this trip assigned to the rail system, the traveler-agent's role is then changed to `<rail passenger>` and her/his movement is handled by our new proposed rail supply module. The exact location of this individual in the rail system is calculated and updated at every simulation time-step until she/he leaves the rail system; at this juncture, her/his corresponding role changes, and is no longer visible to the rail supply module. The publish/subscribe mechanism was also considered in the proposed integration and is fundamental for opportunistic re-routing, mode choice revision or activity scheduling, thus dynamically allowing for changes in demand. Finally, the individual's realized trips within the rail supply module is returned to the overall *Mid-Term* simulator, as well as performance metrics such as capacity reliability and travel time reliability.

III. The Rail Supply Simulator

A. Components and Structure

Similar to the *Mid-Term*, the rail simulator itself consists of Entities and Agents all integrated within the SimMobility C++ code. We designed six new entities and three new agents: pools, blocks, stations, platforms, schedules and lines as entities; and rail passengers,

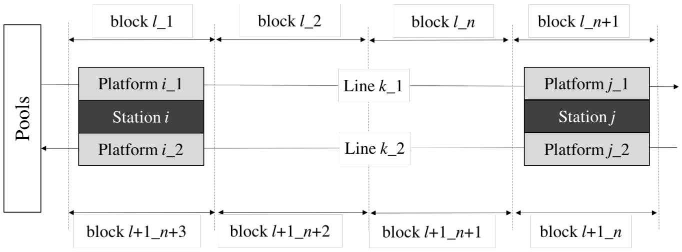

Fig. 2. Entities in the Rail Supply Simulator

trains and service controllers as agents. A *pool* is analogous to an uninvolved area such as a depot, sidings and others, forming an area where out-of-service trains wait (further elaborated below). This helps in distinguishing between different states of operation for the existing trains: active (servicing) vs. inactive (idle at a terminal station, side-track waiting for its next schedule to begin, or out-of-service). This distinction is simulated by having an 'active pool' and 'inactive pool' as ordered lists located at specific stations (generally at the end of a train line). Such list helps in simulating ordering constraints in the dispatching process. A *platform* is linked to a train service in a specific direction and is independent of the number of existing physical platforms. As such, most stations contain two platforms, with interchange stations containing more. A *station* represents a point of connection between the overall *Mid-Term* simulator to the rail simulator, where commuters are "passed" from the former to the latter. *Stations* serve as connections to the multi-modal supply network. A *block* represents a specific section of track with unique geometric and operational properties, and connects platforms to each other. A *line* represents a sequence of platforms and blocks that trains progress through sequentially during service. As such, train services are effectively represented as two different lines in the rail simulator. Finally, every train is assigned a *schedule*, which represents an ordered subset of platforms within a line that a train should serve. For example, a given line k consisting of $i, j, ...$ stations is represented in the rail simulator as two lines: k_1, which consists of ordered list of platforms $i_1, j_1, ...$; and k_2 for the opposite direction with platforms $i_2, j_2, ...$. An express train on k_1 could conceivably stop only in some of the stations (e.g. j_1). Figure 2 shows a graphical depiction of the proposed design. A *station* has walking time parameters that each *passenger* uses to determine their walking time within the station. *Block*-entities have acceleration rates and maximum train speeds associated with them that affect how fast *train*-agents decide to move on that particular *block*, and *platform*-entities have minimum and maximum dwell times that affect how long trains stop at the *platform*-entity. On the other hand, *passengers* and *trains* make decisions at every decision time-step [3] based on their own associated attributes, relevant entity attributes, and instructions from the *Service Controller*. The *Service Controller* controls all other agents' movements and behaviors via the use of internal functions that ensure smooth operation of the system. For example, the *Service Controller* may impose a second speed limit on a given train, in addition to the speed limit already defined by the *block*-entity attributes the train is on, to keep two trains a safe distance from each other. Finally, the user has ultimate control over the rail simulator. Through the use of application programming interfaces (APIs), the user is able to interact with the *Service Controller* to create operational configurations and other scenarios worthy of investigation (such as disruptions). For example, the user may specify unconventional operating logic, or introduce atypical train behavior such as break-downs, to explore the sensitivity of the system to deviations from standard operation or the robustness of proposed policies. These components are summarized in Tables 1 and 2.

[3] An agent- and decision- specific multiple of simulation time-steps

TABLE I
ATTRIBUTES OF AGENTS AND ENTITIES IN THE RAIL SIMULATOR

Name	Type	Attributes
Pool	Entity	• id • maximum capacity • train list (dynamic) • type (active/inactive) • dispatching schedule id
Block	Entity	• id • maximum acceleration rate • maximum deceleration rate • maximum speed • location (X,Y coordinates) • length
Station	Entity	• id • walking time distribution parameters (list) • traits (u-turn, bypass,interchange, regular) • maximum capacity • passenger list (dynamic)
Platform	Entity	• id • dwell time function parameters (list) • maximum capacity • traits (shared, regular) • passenger list (dynamic)
Passenger	Agent	• id • origin (station id) • destination (station id) • path (list of platform ids) • individual walking speed parameter
Train	Agent	• id • schedule id • actual dispatching time (dynamic) • list of passengers (dynamic) • maximum capacity • length
Service Controller	Agent	• id • list of APIs

TABLE II
DECISIONS MADE BY AGENTS

Name	Decisions
Passenger	• walk time selection
Train	• dwell time selection • acceleration, deceleration and speed
Service Controller	• passenger access to station • passenger access to platform • passenger access to train

B. Agent Decisions

For better understanding, each action is marked with a letter-number code that denotes whether it is an action taken by the *passenger*-agent (P), *train*-agent (V), *Service Controller* (C), or User (U) for the necessary inputs, and its position in the sequence of events within the simulation.

1) Passenger-agent Decisions: The passenger flow is essentially linear, consisting of 7 steps with minimal intervention from the *Service Controller* (see Figure 3). A *passenger* officially enters the system (P.01) when it is passed from the overarching *Mid-Term* simulator to the rail simulator (when the passenger 'taps-in' at a fare gate). It is then assigned a walking time to the platform, drawn from a distribution based on the station he enters at (P.02, C.01, U.01) and a individual specific parameter. The passenger queues in first-in, first-out (FIFO) order and is assumed to board the first train that arrives with sufficient capacity (P.03, P.04, C.02). If a train does not have sufficient capacity, the passenger is denied boarding (P.03.5), and has to wait for the subsequent train. Once boarded, the passenger-agent

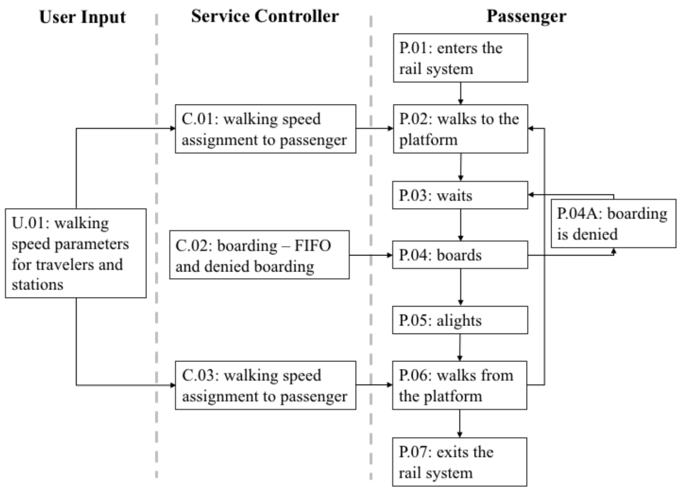

Fig. 3. *Passenger*-agent decisions

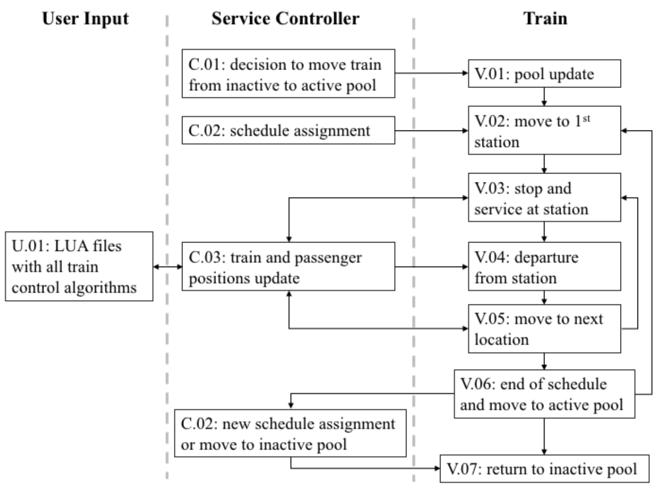

Fig. 4. *Train*-agent decisions

'sleeps', with all movement between stations controlled by the train-agent. The passenger is then 'awakened' at his desired destination (P.05), and assigned a walking time to the fare-gate (if exiting the system) or the next platform (if performing a transfer) (P.06, C.03, U.01). Note that the walking times in P.02 and P.06 are drawn from the same percentile of different distributions to maintain consistency in individual walking speeds and keep the geometric features of the stations constant. Once the passenger exits the system (P.07), the train sub-trip is determined to be completed.

2) Train-agent Decisions: In contrast to *passenger* agents, the flow of the *train* agent is significantly more complicated (see Figure 4). Service begins when a train is pulled from the "inactive pool" to the "active pool" by the *Service Controller* (V.01, C.01). The former is analogous to a train depot, while the latter is analogous to an out-of-service train waiting at a terminal station. The train then receives schedule information comprising its departure time from the first station , a list of stations to service, and its expected arrival time at those stations (V.02, C.02). When a train arrives at a station, it relays this information to the *Service Controller*, which checks for any especial instructions for the train previously specified by the user (U.01). If there are none, the *Service Controller* boards and alights passengers, calculates a dwell time for the train based on the dwell time models, and instructs its subsequent behavior (V.03, C.03). Similarly, as the train moves between stations, its relative position to other trains and stations in the line is constantly monitored by the *Service Controller*, and its speed adjusted accordingly subject to train movement models and especial instructions from the user, if any (V.04, C.03, U.01). Finally, when the train reaches the end of the line, it conveys this information to the *Service Controller*, which typically assigns the train a new schedule, usually the corresponding return route (V.05, C.04). In rare cases, the *Service Controller* will instead retire the train from the active pool (V.06, C.04), essentially sending it back to the depot. This occurs when the desired arrival frequency decreases, for example, when the system changes from peak scheduling to off-peak scheduling.

IV. An Implementation of the Service Controller

Within the proposed simulation, the central control is represented by the *Service Controller*[4]. It uses externally-written *LUA* scripts as an interface to modify the behavior of the rail simulator and thus allows for the simulation of a multitude of scenarios. For example, a `LUA` script could be used to modify train speeds and accelerations to test the effects of introducing brand-new trains into the system. It may also be used to simulate disruptions. The `LUA` scripts interact with the rail simulator through Application Program Interfaces (APIs), which allow different aspects of the rail simulator to be modified. Each API has a defined set of inputs, target functions to modify, and resulting outputs. For example, the API `reset_speed_limit` takes speed limit, start platform, end platform, and line ID as inputs to set the new maximum speed of trains along a portion of a line. There are 2 main categories of APIs: *Command APIs* that are used to instruct the simulation to perform actions that deviate from typical behavior, including modifications to train speed or schedules; and *Informational APIs* that sense the status of various elements within the simulation, such as the number of current active trains or the ID of the next train to arrive at a station. Using these APIs, `LUA` scripts can then be created:

1) The user specifies `if...then` scenarios, which modify train operations in real-time during the simulation if conditions are met (reactive control).
2) The user creates scenarios that proactively modify train operations in order to simulate and observe the effects of events that may potentially occur (proactive control).

A major advantage of the *Service Controller* is its extreme flexibility, which allows the rail simulator to be customized for vastly different scenarios.

A. Signaling and Train Movement

The primary purpose of a signaling system is to help trains in a rail network maintain safe distances from each other such that each train has adequate room to brake as necessary. A 'safe distance' is typically considered to be the braking distance of a train, plus some additional leeway known as the safe operating distance[5]. In cases where track is shared, especially when between trains running in opposite directions, signaling systems are also used to ensure that only one train runs on the track at any given time. These systems have also been utilized to implement operations research policies such as in Kariyazaki et al. [16], where train speeds are intentionally reduced to minimize delays at stations. Ultimately, a signaling system must be able to modify train speeds. There are two main types of signaling systems used worldwide: fixed-block and moving-block systems. Hill et al. [17] provides an in-depth discussion of these systems, as well as general signaling principles and their

[4]Different operators are simulated as instances of the *Service Controller*.

[5]A safe operating headway is often also implemented as a secondary measure; this can be modeled in the same way as a safe operating distance.

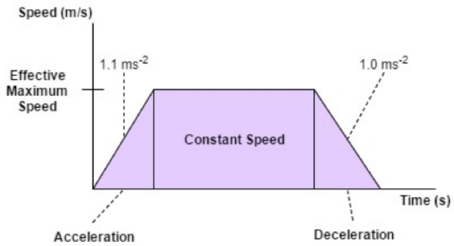

Fig. 5. Simulated Speed-Time Profile

implementation in rail simulators, and their proposed simulation framework was implemented in our simulator through a `LUA` script. In this framework, the designed *block*-entities were a valuable tool for capturing track properties at specific locations, and were used to model the impact of gradient in the maximum acceleration rate. As demonstrated in Kraft [18], train movement between stations can be approximated with a trapezoidal speed-time profile - that is, in regions of constant acceleration, constant speed, and constant deceleration - with minimal loss of accuracy (see Figure 5). This allows train movement to be calculated via relatively straightforward kinematic equations which were again implemented through `LUA` scripts (see [16] for the individual equations). Note that in this implementation, continuous equations were used in discrete time-interval computations. If the time-step used in the simulation is small, this is an acceptable approximation; however, if the time-step is large, unreasonably high deceleration values may occur due to loss of accuracy in estimating train positions. In order to help the simulator yield coherent results, a solution was introduced, where trains decelerated one time-step earlier than strictly necessary in certain cases[6].

B. Passenger-Train Interaction

Stations can have multiple properties from the following list:

1) **U-Turn**: There is a siding within, or in the vicinity of, the station such that trains can reverse and switch to the corresponding line heading in the opposite direction;
2) **Bypass**: There is a siding for the train to overtake the train in front of it, or allow trains behind it to overtake, or simply wait for further instructions without obstructing other trains;
3) **Interchange**: The station is shared by multiple lines and passengers may transfer between lines;
4) **Shared Track**: Multiple lines share the same track; trains may switch to different lines upon leaving the station.

Upon arriving at a station, a train first references its schedule, along with any altering input from the user via the *Service Controller*, to determine its departure behavior (to continue to the next station on the line, switch to a different line, wait at a siding, etc.). The train, station, and platform passenger lists are then updated according to the origin-destination matrix of passengers at the station, and the change in occupancy used by the *Service Controller* to calculate dwell times. Next, the *Service Controller* calculates any additional delays that may occur due to intended behavior, for example, a train attempting to 'U-turn' must wait until the corresponding track on the opposite line is clear; or external factors, such as holding policies specified by the user. Finally, the train departs after the total required delay resolves.

C. Dwell Time

Dwell time is affected by both system specific factors such as passenger loads and behavior, and other external factors that can affect operating conditions [19]. However, given that the latter is hard to quantitatively measure, they are usually not included in dwell time models, but considered through random parameters. Previous works [20, 21] suggest that a straightforward approach to modeling dwell time (DT) is as a linear function of:

$$DT = \beta_0 + \beta_1 * boarding + \beta_2 * alighting + \beta_3 * congestion \quad (1)$$

where boarding represents the total number of boarding passengers, alighting the total number of alighting passengers, and congestion a combined term that reflects the interactions between passengers staying on-board (through-standees) and alighting passengers, through-standees and boarding passengers, alighting and boarding passengers, interference within alighting passengers as their number increases, and interference within boarding passengers as their number increases. The composition of the congestion term will vary from system to system or indeed even from line to line, and can only be determined from an understanding of existing conditions and an analysis of which model best fits the available empirical data. Alternatively, a microscopic simulation of crowd dynamics could be used instead.

[6]It should be noted that the approach in the preceding subsections essentially applies continuous equations to discrete time-intervals and appropriate treatment should be applied for long simulation steps, see [16]

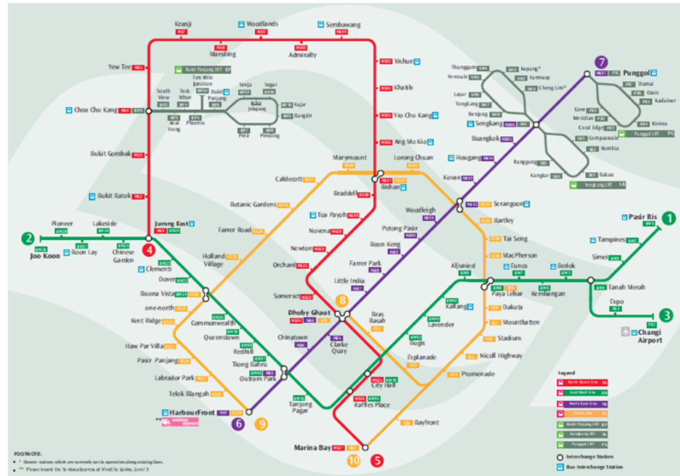

Fig. 6. Singapore MRT network

V. Case-study: Singapore's North-East Line

A. Introduction

In this section, we use the calibration of the Singapore Mass Rapid Transit (MRT) North-East Line (NEL) as an example. The NEL (the purple line in Figure 6, with end stations #6 and #7) is fully automated and has a total of 28 operating trains servicing 16 stations across 20km. It has a daily ridership of approximately 350000 commuters, and provides a direct route from the primarily residential north-east region of Singapore to the Central Business District in the south. The calibration process was conducted using EZ-Link (AFC) data provided by Singapore's Land Transport Authority (LTA): approximately 173 million trips, encompassing all rail and bus trips made in the month of August 2013. Trip data included start and end locations, start and end time, and trip date. Operating specifications for the NEL were obtained from SBS (the NEL operator) and can be found in [16].

B. Calibration

The parameters of all control functions implemented within the *Service Controller* were then fine tuned to the observed conditions in Singapore. Calibration of supply has typically been done by simply utilizing historical data, or manually collecting data at certain stations, then extrapolating and generalizing this data to the entire train network. In recent years, due to rapid advances in communications technology and the increasing ubiquity of electronic devices, automatic fare collection (AFC) via the use of smart cards (or even phones in some cities) is being adopted by an increasing number of transit agencies worldwide relaxing the burden of traditional data collection. This data has often been used with train movement data (known as automatic train control, or ATC) to match passengers to trains and therefore calibrate supply. This, unfortunately, poses a problem in cases like Singapore where ATC data is unavailable, which, due to privacy concerns, is becoming more prevalent. We propose a solution to this problem via the method of sequential calibration to calibrate the supply of trains in a rail simulator. In this method, AFC data is used together with General Transit Feed Specification (GTFS) data and minimal manual data collection to calibrate many aspects of train supply. As shown in Figure 3, the flow of a passenger through the rail system consists of 5 components: walking from the fare gate to the platform, waiting at the platform for the train to depart, traveling (that is, time spent in motion on the train) to the destination, dwelling at intermediate stations, and walking from the platform to the fare gate at the destination. This can be expressed via the following equations, where the superscripts i, j, and k represent the starting, ending, and intermediate stations of a trip, and the subscripts refer to specific trip time components:

$$\mu_{i,j} = \mu^{i}_{walk} + \mu_{wait} + \sum_{k=i}^{j-1} \mu^{k,k+1}_{travel} + \sum_{k=i+1}^{j-1} \mu^{k}_{dwell} + \mu^{j}_{walk} \quad (2)$$

$$\begin{aligned} \sigma^2_{i,j} =& \sigma^{2^i}_{walk} + \sigma^2_{wait} + \sum_{k=i}^{j-1} \sigma^{2^{k,k+1}}_{travel} \\ &+ \sigma^2_{dwell(allstations)} + \sigma^{2^j}_{walk} + 2Cov(i,j) \end{aligned} \quad (3)$$

That is, the expected travel time between stations i, j assuming no transfers is equal to the sum of:

1) The expected walking time within station i.
2) The expected waiting time, assumed to be independent.
3) The sum of expected travel-time-between-stations on the train.
4) The sum of expected dwell times at intermediate stations.
5) The expected walking time within station j.

If the components are assumed to be independent (except walking times within i and j), then the variance of the expected travel time between stations i, j can be expressed in a similar manner. Note that the variance of the dwell times is represented as a variance for total combined dwell time, since the dwell times at stations are likely to be correlated for one specific trip. The aim of sequential calibration is to, for each component:

1) Select the distribution function type.
2) Estimate the mean μ and variance σ^2.
3) Generate the function parameters.

As the calibration progresses, information from the previous stage is used to estimate parameters of the next stage. It is therefore important to constantly ensure that the calibrated parameters are coherent to improve the accuracy of the calibration procedure. In the interests of expediency, the calibration procedure makes five key assumptions:

1) Individual specific walk times from/to the fare gate to/from the platform are the same for given station i.
2) Passengers and trains arrive at stations independently from other passengers and trains respectively.
3) Between station's travel time is constant, with zero variance.
4) μ_{walk} and μ_{wait} are independent.
5) Commuters are always able to board the first train that arrives.

It should be noted that assumptions (1) and (4) are not strictly necessary: the former can be relaxed by doubling the number of walking time parameters, and the latter by introducing a second covariance term corresponding to the interaction between walking times and waiting times. However, these assumptions help expedite the calibration process; furthermore, walk time data collected in a sample station seems to support the first assumption [16]. Assumption (2) is a common assumption made in train modelling, and can also be relaxed by examining passenger arrival rates for trends to predict when passengers arrive in clusters as opposed to singly. However, that is out of the scope of this paper. Assumption (3) is a reasonable assumption given that the NEL is automated: trains are controlled by a computer, and run at almost identical speeds, with variance only occurring if a train delay affects the train behind it. However, the NEL rarely runs at peak frequency; as such, cascading delays are unlikely and travel time can be assumed to be generally constant. Finally, Assumption (5) was addressed by calibrating the rail simulator for both the off-peak and peak. In the former, denied boarding can be safely assumed to be zero; as such, it may be possible to compare trends across the two time periods and identify probabilities of denied boarding as an area of future work. The results of each disaggregate calibration were then checked against existing data and then used as input for the next calibration stage (of the next associated model). For example, passenger walking time parameters were estimated, verified against station counts, and then used to estimate train dwell time parameters. This was possible due to the nature of AFC data (which captures commuters' points and times of entry and exit into/from the system, thus providing a good picture of overall travel times), as well as well-structured individual components of travel time, which allowed for reasonable simplifying assumptions to be made. For details on each individual step of the calibration the reader is referred to [24]. Regarding the goodness-of-fit-measure for the calibration, Hollander et al. [23] provide a good summary of the advantages of different objective functions. For the purposes of this estimation, the objective function was chosen to be the root-mean-squared error (RMSE) and the root-mean-squared normalized error (RMSNE) was also calculated as control metric. The Simultaneous Perturbation Stochastic Approximation (SPSA) algorithm as described in [24,22] was run multiple times with different initial seeds for the dwell time calibration step. At the end of the entire calibration process, the estimated parameters are observed to fit simulated travel times to actual times reasonably well, with a RMSE value of 176.23 seconds, or about 18.4% of the actual average travel time of 960 seconds, while the average deviation of each simulated trip from the actual trip duration was about 29%. Approximately 57% of simulated travel times fell within +/- 15% of the actual travel time, and approximately 68% of simulated times fell within +/- 20% of the actual travel time. Figure 7 plots the CDFs of simulated and actual travel times against each other. The curves are generally similar, although the variance of actual travel times is slightly larger than that of the simulated travel times. This is likely due to the presence of difficult-to-detect factors in actual travel times that result in erroneous data, which will be subsequently elaborated on. Figure 8 compares actual and simulated travel times, with the x-axis representing the former and the y-axis representing the latter. Each '+' represents a specific trip, and the red 45° line represents the ideal 'perfect fit', where every simulated travel time matches actual travel time exactly. It can be seen that the estimated/actual travel time pairs generally lie along this line, demonstrating that a linear dwell time function results in a reasonable approximation for the data.

C. Validation

In order to validate the parameters estimated in the previous section, AFC data from an arbitrarily chosen weekday (Friday, 9 Aug 2013) was used. This data comprises approximately 170,000 trips which were then fed as demand into the rail simulator. Heatmaps were used to compare the distribution of demand in the calibration day against the validation day, as shown in Figure 9.

The numbers on the axes correspond to the station numbers (for example, '1' refers to station NE1). The distribution of origin-destination pairs is largely similar across both days, with major interchange stations such as NE12 (Serangoon) experiencing high

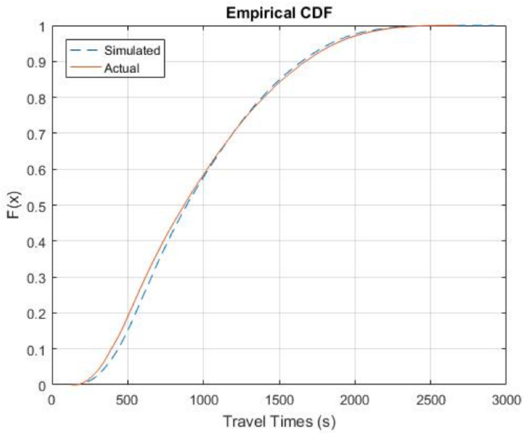

Fig. 7. Distribution of Simulated vs. Actual Travel Times - Calibration

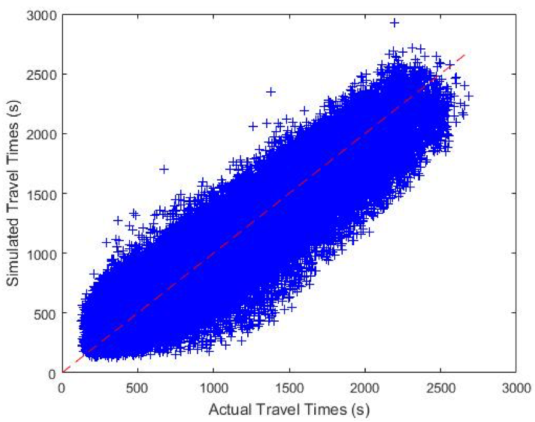

Fig. 8. Comparing Simulated and Actual Travel Times - Calibration

commuter flow. We also note that NE1 (Harbourfront) has a significantly higher passenger flow on the validation day than the calibration day, likely due to its popularity as a recreational location. The calibrated parameters resulted in a RMSE of 204.21 seconds, or 19.8% of the actual average travel time of 1034 seconds. (Note that the difference between these results and the results obtained earlier exceed the standard deviation of the RMSE, which was roughly 0.3 seconds). The RMSNE value was 0.2679, a seeming improvement over the estimated case; however, this was due to the longer average travel times in the validation case, which was likely a result of the flow of passengers to NE1 and back to the suburbs mentioned earlier. Approximately 52% of simulated travel times fell within +/- 15% of the actual travel time, and approximately 65% of simulated times fell within +/- 20% of the actual travel time. The parameters estimated are therefore shown to constitute a reasonably accurate model of travel time. Figure 10 plots the CDFs of simulated and actual travel times against each other. We observe that simulated travel times tend to be shorter than the actual travel times, once again due to the validation day having longer travel times overall than the validation day. This problem can likely be rectified via the use of a larger sample size across multiple days for estimation, by performing separate estimations for weekdays and weekends (with Friday considered a weekend), etc. Figure 11 is very similar to Figure 8, with the estimated/actual dwell time pairs generally lying along the line of perfect fit. We therefore conclude that the approach here presented is a valid method of estimating parameters, especially when the goal is to simply obtain appropriate initial seed values

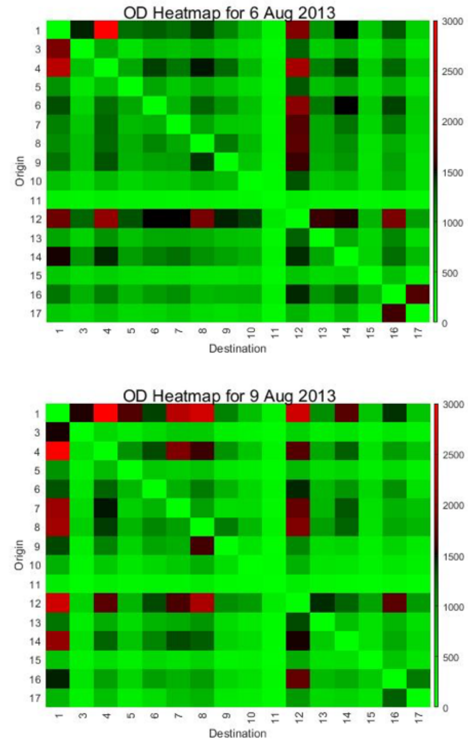

Fig. 9. Heatmaps of Calibration Demand vs. Validation Demand

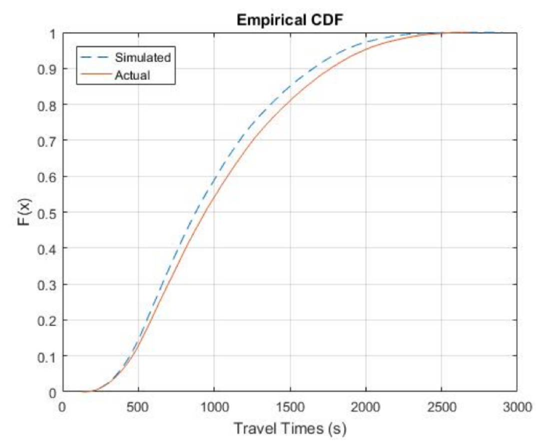

Fig. 10. Distribution of Simulated vs. Actual Travel Times - Validation

for a secondary simultaneous calibration (or in other cases where computational speed is prioritized over a high degree of accuracy). Calibration improvements can be made at the cost of additional data or computing power; these are briefly discussed in the next section.

D. Disruption Example

In [26], the proposed rail simulator was used for the assessment of a bus-bridging disruption management strategy, using additional shuttles between pairs of disrupted MRT stations. This was the first application of the simulator and provides a good demonstration of its capabilities. This bus-bridging strategy was used during an actual real incident in 2013, when the NEL service was interrupted between stations NE11 and NE17. Adnan et al. [26] compared this strategy against a no extra service scenario, where travelers have to seek for other existing alternatives namely buses, taxis, or walking. Several

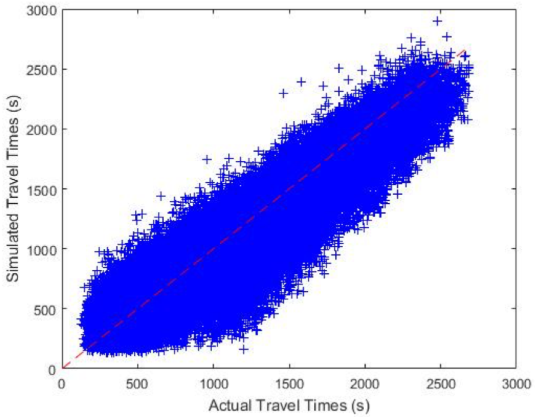

Fig. 11. Comparing Simulated and Actual Travel Times - Validation

metrics were obtained from the simulations, namely travel times, dwelling time, waiting times, denied boarding and mode share across the Singapore network. Unsurprisingly, the bus bridging strategy substantially outperforms the no extra service scenario with, for example: an average door-to-door travel time reduction of 36.8 min, for all trips that are directly affected by the disruption; a reduction of over 60 min in the average delay time for the trips to/from the Central Business District that utilized the disrupted MRT; over 50% reduction in denied boardings on the bus stops nearby disrupted MRT stations; and a reduction of the taxi mode share by 13 percentage points. Further details on this specific analysis can be fond in [26].

VI. Conclusion

We developed a new comprehensive rail simulator incorporated within the SimMobility platform, which integrates rail supply with other modes of transportation. The simulator captures not only supply and demand interactions within the rail network, but also how these interactions affect other modes of transport, and was successfully calibrated only on AFC data to yield reasonably accurate estimates of simulation parameters. The features of the proposed Service Controller, a powerful entity that allows the user to construct a wide range of scenarios, were demonstrated via the simulation of a case-study in Singapore and shown to help in the assessment of rail related policies and scenarios in complex urban scenarios. Nevertheless, there is still much room for improvement. For example, rail networks with manually driven trains (which require modeling driver behavior) were not considered in the proposed framework. More detailed models could also have been used; for example, the dwell time model could have included a non-linear congestion term in order to better capture crowding effects. A more ambitious long-term goal would be to expand the rail simulator to model light surface rail. These systems present an additional dimension of difficulty as train-agents interact not only with passenger-agents, but also with road conditions (cars, buses, pedestrians, etc.) and their own drivers (the train-driver-agents mentioned above). The models used in the Service Controller will therefore need to be significantly improved to capture the increased degree of autonomy present in such situations. Regarding the calibration presented here, a considerable number of assumptions were used to improve the efficiency of the sequential calibration process. Although this resulted in a reasonable fit, some of these assumptions could be removed to obtain more accurate estimation parameters. For example, a natural improvement would be to use a non-linear dwell time function to replicate the increasing marginal delay that each additional passenger has on dwell time. This approach was used in [21], which fit a cubic function to the data. It should also be noted that in this work, the NEL was calibrated in isolation out of necessity, as the other lines in the Singapore rail network were not yet developed in the simulation environment. The sequential calibration process should therefore be revisited once the entire Singapore rail network is modeled. Finally, any availability of AVL data would allow us to validate the overall calibration method.

Acknowledgment

The authors are thankful to Z. Huai Peng and J. Karachiwala (SMART) for their contribution to the simulator code and to the LTA for supporting this research and providing the used data.

References

[1] Di, P. (2013), *Key transport statistics of world cities*. Journeys, 2013, pp. 105-112.

[2] Bochmann, J. (1993), *BAHN version 3.11a*. Shareware Professional 4/1993, Grafing, pp. 17 (in German). Details on the simulator (v3.88) were retrieved on Feb 12, 2017, from http://www.jbss.de/

[3] Teixeira, J.M. (2008), *Circles model for metro light rail analysis*. The European Simulation and Modeling Conference (ESM2008). Details on the simulator were retrieved on May 15, 2017, from http://paginas.fe.up.pt/ feliz/man/modeller.html

[4] Nash, A. and Huerlimann, D. (2004) *Railroad simulation using OpenTrack*. Computers in railways IX, pp. 45-54.

[5] Hidas, P. (2005) *A functional evaluation of the AIMSUN, PARAMICS and VISSIM microsimulation models*. Road and Transport Research 14, 4.

[6] Grube, P., Nunez, F. and Cipriano, A. (2011), *An event-driven simulator for multi-line metro systems and its application to Santiago de Chile metropolitan rail network*. Simulation Modeling Practice and Theory 19.1, pp. 393-405.

[7] Snchez-Martnez, G. (2012), *Running time variability and resource allocation: a data-driven analysis of high-frequency bus operations*. PhD Thesis, MIT.

[8] Cha, M.H. and Mun, D. (2014) *Discrete event simulation of Maglev transport considering traffic waves*. Journal of Computational Design and Engineering 1.4, pp. 233-242.

[9] Darmanin, T., Lim, C. and Gan H. (2010), *Public railway disruption recovery planning: a new recovery strategy for metro train Melbourne*. Proceedings of the 11th Asia-Pacific Industrial Engineering and Management Systems Conference, 7.

[10] Jin, J.G., Teo, K.M. and Sun L. (2013) *Disruption response planning for an urban mass rapid transit network*. Proceedings of the 92nd Annual Meeting of the Transportation Research Board, Washington DC.

[11] Ravichandran, H. (2013), *Evaluating the robustness of crew schedules for rail transit systems*. PhD Thesis, MIT.

[12] Horni, A., Nagel, K. and Axhausen, K. W. (2016) *The multi-agent transport simulation MATSim*. Ubiquity, London.

[13] Adnan, M., Pereira, F. C., Lima Azevedo, C., Basak, K., Lovric, M., Raveau, S., Zhu, Y., Ferreira, J., Zegras, C., Ben-Akiva, M. (2016), *SimMobility: A Multi-scale Integrated Agent-based Simulation Platform*. Proceedings of the 95th Annual Meeting of the Transportation Research Board, Washington D.C., USA.

[14] Lu Y., Adnan, M., Basak, K., F. Pereira, Carrion, C., Saber, VH., Loganathan, H., Ben-Akiva, M. (2015), *SimMobility Mid-Term Simulator: a state-of-the-art integrated agent-based demand and supply model*. Proceedings of the 94th Annual Meeting of the Transportation Research Board, Washington D.C., USA.

[15] Zhu, Y. and Ferreira, J. (2014), *Synthetic population generation at disaggregated spatial scales for land use and transportation microsimulation*. Transportation Research Record 2429.

[16] Koh, K., (2017), *Developing an Integrated Rail Simulator for SimMobility*. Master of Science in Transportation Thesis, MIT.

[17] Kariyazaki, K., Naohiko, H., Shigeru, M. (2015), *Simulation analysis of train operation to recover knock-on delay under high-frequency intervals*. Case Studies on Transport Policy 3.1, pp. 92-98.

[18] Hill, J. R., Yates, T. (1992), *Modeling railway block signaling systems using discrete-event simulation*. Proceedings of the ASME/IEEE Spring Joint Railroad Conference, IEEE, 1992.

[19] Kraft, K. H. (2013) *Fahrdynamik und automatisierung von spurgebundenen transportsystemen*. Springer-Verlag Vol 17.

[20] Kraft, W. H. (1975), *An analisys of the passenger vehicle interface of street transit systems with applications to design optimizaiton*.

[21] Lin, T.M., Wilson, N.H.M. (1992), *Dwell time relationships for light rail systems*. Transportation Research Record 1361.

[22] Puong, A. (2000), *Dwell time model and analysis for the MBTA red line*. Massachusetts Institute of Technology Research Memo.

[23] Hollander, Y., Liu, R. (2008), *The principles of calibrating traffic microsimulation models*. Transportation 35.3, pp.347-362.

[24] Spall, J. C. (1998), *Implementation of the simultaneous perturbation algorithm for stochastic optimization*. IEEE Transactions on Aerospace and Electronic Systems 34.3, pp. 817-823.

[25] Lima Azevedo, C., Oh S., Deshmukh N., Marimuthu B., Marczuk K., Soh H., Basak K., Toledo T., Peh L. S., Ben-Akiva M. (2017), *SimMobility Short-term: An integrated microscopic mobility simulator*. Transportation Research Record 2622.

[26] Adnan, M., Pereira, F., Lima Azevedo, C., Basak, K., Koh, K., Loganathan, H., Huai P. Z., Ben-Akiva, M. (2017), *Evaluating disruption management strategies in rail transit using SimMobility Mid-term simulator: the Singapore MRT North-East line*. The 96th Annual Meeting of the Transportation Research Board, Washington D.C., US.