

# Next-Generation Design Tools for Intelligent Transportation Systems

Dominik Ascher<sup>1</sup> and Georg Hackenberg<sup>2</sup>

<sup>1</sup>*Faculty of Electrical Engineering and Computer Science, Technical University of Berlin, 10587 Berlin, Germany*

<sup>2</sup>*School of Engineering, University of Applied Sciences Upper Austria, Stelzhamerstraße 23, 4600 Wels, Austria*

**Keywords:** Model- and Simulation-Based Systems Engineering, Model-Based Design, Intelligent Transportation Systems.

**Abstract:** Intelligent Transportation Systems (ITS) promote new transportation paradigms such as connected and autonomous vehicles (CAV), multi-modal and demand-responsive transport systems, and enable the transportation electrification by sustainable operation of electric vehicles. Methods and tools are needed to explore the possible design space for emerging transportation paradigms, which support evaluation of system design alternatives and verification of system properties. In this work, we propose a model- and simulation-based systems engineering framework for capturing design decisions and evaluating control strategies for ITS design. In addition to capturing and evaluating different design decisions, the proposed solution allows users to guide design decisions by systematic comparison and evaluation of system configurations and control strategies.

## 1 INTRODUCTION

The transformative patterns of future mobility are enabled by interconnected as well as integrated systems and services. For this, Intelligent Transportation Systems (ITS) establish new transportation paradigms such as connected and autonomous vehicles (CAV) (Kopelias et al., 2020), multi-modal and demand-responsive transport systems (Brake et al., 2004), as well as the transportation electrification by sustainable operation of electric vehicles (Zhang and Fujimori, 2020). In order to minimize environmental impacts, complex scenarios between these systems and their requirements need to be holistically addressed, from which integrated system designs are derived, which enable transportation infrastructure and heterogeneous actor efficiency and sustainability.

To systematically support the aforementioned systems engineering task, model- and simulation-based systems engineering (Gianni et al., 2014) can be employed for abstraction of design problems through the formulation of system models and their evaluation in terms of concrete behavior at run time, where simulation aids projecting run-time information and performance metrics about the system under design and environment. However, as systems and their underlying requirements are typically imperfectly understood at the beginning of the design task, methods and tools are needed to explore the possible design space for emerging transportation paradigms, which support

evaluation of system design alternatives and verification of system properties.

With our research, we want to help improve the efficiency, effectiveness and sustainability of today's transportation systems. To achieve this goal, we work on methodologies for designing such systems and verifying their properties. Fundamentally, we promote a formal approach capturing the relevant design decisions and their relations. Furthermore, we integrate scenario-based simulation of system dynamics and evaluation of emergent properties. Finally, we exploit optimization algorithms for optimizing system dynamics as well as static design decisions.

In this paper, we explore how the next generation of design tools for ITS addressing emerging paradigms could look like. Therefore, first we want to understand which system properties and design decisions should be represented in these tools. Then, we want to learn how design decisions could be verified with respect to the desired system properties. Finally, we want to study how the relevant design information could be represented in a graphical user interface.

In the following, we first describe an overview of related work for our approach in Section 2. Then, we describe approach and methodology for a model- and simulation-based systems engineering framework for capturing and evaluating design decisions in Section 3. Thereafter, we describe a tool prototype for building system designs, conducting simulation runs, and visualizing simulation outcomes in Section 4.

## 2 RELATED WORK

We discuss related work for ITS in terms of infrastructure planning, eco-routing and driving, cooperative driving as well as mobility-on-demand systems.

**Infrastructure Planning.** Traffic simulation tools such as PTV Vissim (Fellendorf, 1994), MATSim (W Axhausen et al., 2016), SUMO (Lopez et al., 2018) as well as AIMSUN (Barceló and Casas, 2005) support analysis and decision-making for transportation systems engineering problems. For instance, SUMO can be used to assess mesoscopic to microscopic traffic scenarios, based on behavior models for passenger and vehicle behavior, and infrastructure models separating links, intersections and lanes for different transport modalities, while implementing actor and traffic control strategies.

**Eco-Routing and -Driving.** Eco-routing (Ericsson et al., 2006) describes a concept, which focuses on traffic participants and their routes in terms of targeting energy-efficient route selection and reduction of emissions, whereas the concept of Eco-Driving (Huang et al., 2018) focuses on energy-efficient intermediate driving behavior. Potential congestions may arise within transport networks of single modalities due to their excessive utilization, where energy-efficient utilization of different transport modalities can be targeted (Namoun et al., 2021).

**Cooperative Driving.** Cooperative driving of intelligent connected vehicles (Wang et al., 2022) is enabled by Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I) or Vehicle-to-Everything (V2X) communication. Specific use-cases include vehicle platooning (Jia et al., 2015), management of distributed electric vehicle fleets and their integration to stabilize ancillary systems such as the power grid in the context of Vehicle-to-Grid (V2G) (Hu et al., 2016) applications, as well as alignment of different transportation modalities (Harris et al., 2015).

**Mobility-on-Demand.** Mobility-on-demand systems refer to systems, where transportation demands get covered by transportation modalities, which make up the transportation supply. Here, Ridesharing favors shared mobility over personal mobility, bringing together passengers with a shared route in shared vehicles (Furuhata et al., 2013). In autonomous mobility-on-demand systems, autonomous transportation modalities cover transportation supply while increasing system efficiency (Fagnant and Kockelman, 2014).

## 3 METHODOLOGY AND APPROACH

Our model- and simulation-based systems engineering approach is based on (Ascher and Hackenberg, 2015; Ascher and Hackenberg, 2016; Ascher and Hackenberg, 2017; Ascher et al., 2023; Ascher and Hackenberg, 2023) and employs an integrated systems modeling technique for design of integrated ITS systems and investigation of emergent system properties. Our modeling technique supports both static property, i.e. structural and infrastructure design, and dynamic property design, i.e. behavior and control strategy design. Here, Figure 1 shows an overview of the system design methodology.

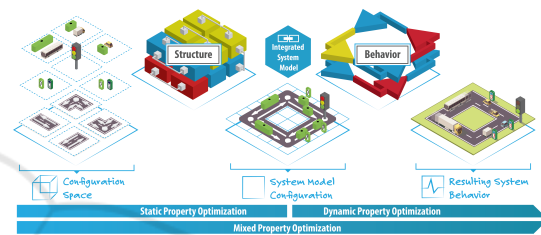


Figure 1: System design methodology.

Subsequently, first the intended usage of the framework is described in Section 3.1. Secondly, domain-specific design concepts for the framework are described in Section 3.2. Thirdly, problem solution for design problems is described in Section 3.3.

### 3.1 Intended Usage

We subsequently describe use-cases considering different roles for our systems engineering approach for investigation of static and dynamic properties:

**Transportation System Engineers (TSEs).** maintain and design transportation infrastructure.

- *Static Properties:* TSEs may be supported in the investigation of transportation infrastructure properties such as placement of intersections and links, and capacities of links such as number of lanes.
- *Dynamic Properties:* TSEs may be supported in devising traffic control strategies and semaphores incorporating demand-responsive cycle times.

**Charge Point Operators (CPOs).** maintain and operate a charging infrastructure in a defined area.

- *Static Properties:* CPOs may be supported in devising charging infrastructure such as planning of placement and capacity of charging stations along most frequented routes and points of interest.



**Discrete-Event Simulation.** is used, where events are defined as functions over a system’s trajectory of states, which indicate *when* actions need to be taken in the system simulation, thereby reducing action space dimensionality (Ascher and Hackenberg, 2023).

**Heuristic Search.** procedures are employed, which are iteratively designed by domain experts, where simulation results are utilized to adapt heuristic control logic as needed (Ascher and Hackenberg, 2015).

**Approximate Dynamic Programming.** methods employ approximations of policy and value functions (Powell, 2007), which are used to reduce DP problem complexity (Ascher et al., 2023).

## 4 TOOL PROTOTYPE

Based on the general approach explained in Section 3 we started developing an open source tool prototype hosted on GitHub<sup>1</sup>, which we subsequently describe: In Section 4.1 we describe core data structures for representing both static system configurations and dynamic system states. Then, in Section 4.2 we explain how users can implement and integrate different control strategies into the tool prototype. Next, in Section 4.3 we describe data collected during simulation runs which can be used for visualizations and possibly training. In Section 4.4 we provide an overview of the engine, which computes discrete events and updates system states accordingly. Finally, in Section 4.5 we highlight two different framework applications.

### 4.1 Data Structures

The `model` module provides the core data structures for modeling static system configurations and representing dynamic system states. Figure 3 shows the classes, their attributes, and their relationships.

The `Intersection` class represents intersections of the driving infrastructure. Each intersection stores its coordinate in three-dimensional space. Note that we use Cartesian coordinates for simplicity.

The `Segment` class represents road segments of the driving infrastructure. Each segment points to its source and target intersection and provides a method for computing its length based on Euclidean distance.

The `Location` class represents specific points on the segments of the driving infrastructure, where each

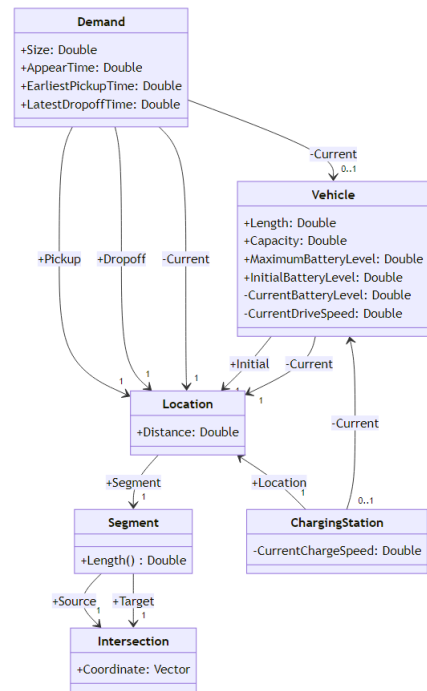


Figure 3: Configuration and simulation data model.

location points to a corresponding segment. Furthermore, each location stores a distance on this segment measured in Euclidean units of the Cartesian space.

The `ChargingStation` class represents the charging infrastructure. Each charging station stores its location on the driving infrastructure. Furthermore, each charging station optionally points to a current vehicle. Each charging station provides the current charging speed, where the simulator computes the current charging speed dynamically.

The `Vehicle` class represents the driving resources. Each vehicle provides an initial and a current location on the driving infrastructure. The initial location defines the position of the vehicle in the initial state of the simulation. Then, the simulator continuously updates the current location of the vehicle. Furthermore, each vehicle stores a length and a capacity, a maximum, an initial, and a current battery level, and a current drive speed. The length determines how much space the vehicle occupies on the driving infrastructure. The capacity defines how much demand the vehicle can carry. The maximum battery level specifies the size of the energy storage. The initial battery level stores the load state of the energy storage at simulation start. Finally, the simulator continuously updates current battery level and current drive speed.

The `Demand` class represents the transportation loads to be served. Each demand points to a pickup and a dropoff location as well as a current location

<sup>1</sup><https://github.com/ghackenberg/transport-ide>

and vehicle. Furthermore, each demand stores a size as well as an appear, and earliest pickup, and a latest dropoff time. The simulator continuously updates the current location and vehicle.

### 4.2 Control Strategies

During a simulation run, a number of control decisions have to be taken. For example, when arriving at an intersection each vehicle has to select the next outgoing road segment. Similarly, when arriving at a demand pickup location each vehicle has to decide whether to serve the demand or not. The overall system performance heavily depends on the optimality of the individual choices made during a simulation run. Therefore, one key engineering task for this class of systems is to develop an appropriate control strategy. Since desired control strategies cannot be hard-coded upfront, the simulator supports plugging in and testing different strategies, which must implement the controller interface methods depicted in Figure 4.

Controller	
SelectDemand(Vehicle v, Demand d) :	Boolean
SelectStation(Vehicle v, Station s) :	Boolean
UnselectStation(Vehicle v) :	Boolean
SelectSegment(Vehicle v) :	Segment
SelectSpeed(Vehicle v) :	Double
SelectMaximumSpeedSelectionTimeout(Vehicle v) :	Double
SelectMaximumStationSelectionTimeout(Vehicle v) :	Double

Figure 4: Methods of the controller interface.

We currently provide four different implementations of the controller interface: A **manual**, a **random**, a **greedy**, and a **smart** control strategy. Figure 5 provides an overview of the four control strategies and their decision logic. Columns of the matrix represent individual control strategies, rows represent decisions to be taken, and cells represent corresponding logic.

Control decision	Control strategy			
	manual	random	greedy	smart
route	user	random	greedy	smart
pickup	user	random	yes	yes
charge	user	random	yes	smart
level	max	max	max	max
speed	max	max	max	smart

Figure 5: Overview of the control strategies.

In the following, we describe the logics behind each of the control strategies in more detail.

#### 4.2.1 Manual Control Strategy

The manual control strategy delegates routing, demand pickup, and charge decisions to the tool user using input dialogs. The remaining control decisions are derived automatically.

Figure 6 shows the input dialog for route decisions, which pops upon vehicle arrival at an intersection. It provides the vehicle name (V in the example) and possible follow-up road segments (C->D and C->E in the example). Note that C, D, and E represent the intersection names connected through segments. The user can select the desired routing option by pressing the button for the respective follow-up segment.

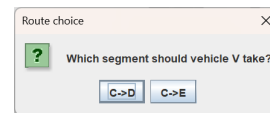


Figure 6: Vehicle route choice.

Figure 7 shows the input dialog for demand pickup decisions, which pops up when a vehicle arrives at the pickup location of an appeared and unserved demand. It provides the vehicle name (U in the example), demand data (i.e. pickup location, earliest pickup time, dropoff location as well as latest dropoff time), and two choice buttons (i.e. yes and no).

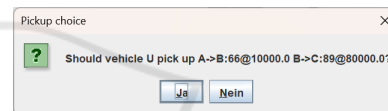


Figure 7: Demand pickup choice.

Figure 8 shows the input dialog for vehicle charging decisions, which pops up when a vehicle arrives at the location of an unoccupied charging station. The dialog provides the name of the vehicle (U in the example), the location of the charging station (A->B: 50 in the example), and the buttons for the two available choices (i.e. yes and no). Finally, the strategy al-

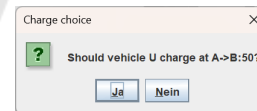


Figure 8: Vehicle charging choice.

ways selects the maximum driving speed for each vehicle without considering possible collisions and always chooses to fully charge vehicle batteries after the user decided to start the charging process.

#### 4.2.2 Random Control Strategy

The random control strategy uses a random number generator for making routing, demand pickup, and charging decisions. For each decision, it assigns equal probabilities to available choices (i.e. outgoing segments of intersections or yes and no). Driving speed and target battery charge level are handled equally to the manual control strategy.

### 4.2.3 Greedy Control Strategy

The greedy control strategy determines routing decisions upon vehicle arrival at intersections of the driving infrastructure. Its logic comprises four rules being sequentially processed until one rule applies:

1. When the battery of the vehicle is only half full or less, the strategy randomly selects an outgoing road segment with charging station if such segment is available.
2. Then, when the vehicle carries one or more demands, the strategy randomly selects the drop-off segment of one such demand if reachable directly via the intersection.
3. Next, when the vehicle carries no demand, the strategy randomly selects the pick-up segment of an unserved demand if reachable directly via the intersection.
4. Finally, when none of the above three rules apply, the strategy randomly selects any of the outgoing road segments of the current intersection with uniform probability distribution.

Note that the described routing logic only considers the next segment and does not perform a look-ahead across more segments. Finally, the greedy control strategy always chooses to pick up demands or charge vehicle batteries if arriving at an unserved demand pick-up location or charging station. The strategy always chooses to drive at maximum speed and fully charge batteries if a charging process has started.

### 4.2.4 Smart Control Strategy

The smart control strategy uses a more sophisticated strategy for determining routing decisions upon vehicle intersection arrivals. Its logic comprises four rules being sequentially processed until one rule applies:

1. When the vehicle carries one or more demands and a charging station can be reached via their drop-off location, the segment leading to the closest such demand is selected.
2. When the vehicle carries no demands and a charging station can be reached via the pick-up location of an unserved demand, the segment leading to the closest such demand is selected.
3. When the vehicle carries no demands and does not plan to pick-up a new demand, the segment leading to the closest charging station is selected if reachable with the remaining battery level.
4. Finally, when none of the above three rules apply, again the strategy randomly selects any of the outgoing road segments of the current intersection with uniform probability distribution.

When arriving at an unused charging station, this strategy only starts charging if no other charging station can be reached with the current battery level. When the charging station is occupied, but no other charging station can be reached, the strategy lets the vehicle wait at the station to become free. With this strategy vehicles always pick up unassigned demands when passing by, charge their batteries fully when at charging stations, and drive with maximum speed.

## 4.3 Data Recorder

During simulation, a number of events are recorded for visualization of system (and control strategy) performance as well as strategy training. Figure 9 shows the provided methods of the data recorder interface.

Recorder
RecordCrossing(Vehicle v, Segment p, Segment n, Double t)
RecordPickupDecline(Vehicle v, Demand d, Double t)
RecordPickupAccept(Vehicle v, Demand d, Double t)
RecordDropoff(Vehicle v, Demand d, Double t)
RecordSpeed(Vehicle v, Double s, Double t)
RecordDistance(Vehicle v, Double d, Double t)
RecordStep(Double s, Double t)

Figure 9: Data recorder interface.

The recorder tracks, when a vehicle crosses an intersection, and records associated routing decisions of the control strategy. It tracks, when a vehicle passes by an unassigned demand and the control strategy declines or accepts the pick up. Similarly, the recorder tracks, when a vehicle drops off a demand at its target location. Furthermore, it continuously tracks the speed and overall travel distance of vehicles and size of time steps made during discrete event simulation.

Currently, we use data about intersection crossing for determining driving infrastructure bottlenecks and data about pick ups and drop offs for determining per demand waiting and driving time as well as delays.

## 4.4 Simulation Engine

The simulation engine computes event times, delegates control decisions to the control strategy, updates model state, and dispatches relevant data to the recorder. It employs a simulation loop, which advances the model time and updates the model state until no more events are to be processed. The simulation loop can be divided into three main steps:

**Step 1.** Make routing decisions and update vehicle locations. Make charging decisions and update connections between vehicles and charging stations. Make pick up decisions, perform drop offs (also if vehicle battery empty), and update connections between

vehicles and demands. Make speed decisions and update vehicle speeds.

**Step 2.** Compute time of next event to be processed: Determine speed update timeout, charging speed update or charging station disconnect timeout, intersection arrival, charging station arrival, vehicle battery empty/full, demand appearance, demand overdue, demand pick up / drop off location arrival, and vehicle attach / detach.

**Step 3.** Based on current time and time until next event do the following: Update vehicle locations, vehicle battery levels, and record travelled distances of vehicles. Detect vehicle collisions based on their location, width, and overlaps along the segment line. Finally, set model time to time of next event.

### 4.5 Specific Applications

Based on the previous components we implemented two specific applications: The first application can be used for comparing the performance of different control strategies (see Section 4.5.1). The second application can be used for comparing the performance of different driving and charging infrastructures as well as fleet configurations (see Section 4.5.2).

#### 4.5.1 Control Strategy Comparison

To compare different control strategies, we apply them to the same system configuration (including driving / charging infrastructure as well as fleet setup) and scenario (i.e. demand profile). For performance evaluation, we measure the times between demand appearances at the pick up location and subsequent disappearances at the drop off location. Figure 10 shows an example system configuration, where random, greedy, and smart control strategies are applied.

The upper part of the window shows for each control strategy the current system state including the vehicle locations and the active demands. The lower part of the window shows for each control strategy the total times that have passed between demand appearances and their respective disappearances. From the diagram we can deduct that in this simulation run the smart control strategy showed superior performance over the others. Note that due to strategy randomness, the result might differ in a proceeding run.

#### 4.5.2 System Configuration Comparison

To compare system configurations, we instead apply the same control strategy and scenario (i.e. demand

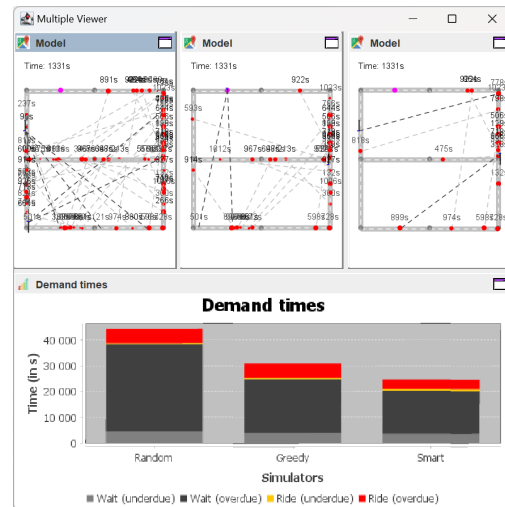


Figure 10: Control strategy comparison.

profile) to different versions of the driving / charging infrastructure as well as fleet setup. Note that differing infrastructure versions might include additional road segments not present in others and thus require that demands only reference road segments, which are included in every version. Figure 11 shows an example of such system configuration comparison. Again,

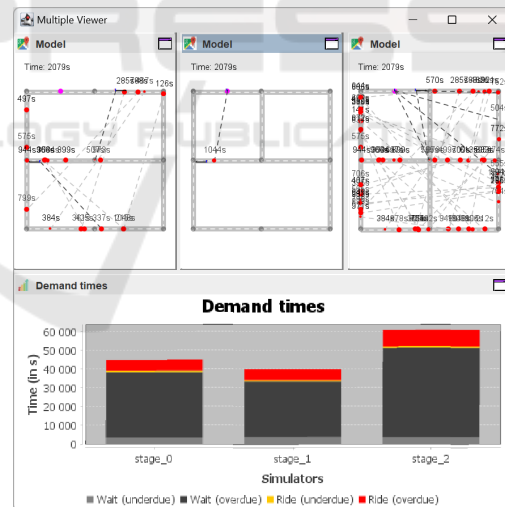


Figure 11: System configuration comparison.

the upper window part shows for each control strategy the current system state including vehicle locations and demands. The lower window part shows the total times between demand appearance and disappearance for each system configuration. In this simulation run, the configuration in the middle, which adds only one additional road segment, has the best performance. The example demonstrates that the smart control strategy may not always deliver optimal results, which is necessary for fair infrastructure comparison.

## 5 CONCLUSION

In this work, we described a design tool for a model- and simulation-based systems engineering framework for capturing design decisions and evaluating static and dynamic properties for ITS design. In addition to capturing different design decisions, users can guide design decisions by systematic comparison and evaluation of system configuration and control strategies. Application results demonstrate the feasibility of the design tool for verification of ITS design decisions with respect to static and dynamic system properties.

## REFERENCES

- Ascher, D. and Hackenberg, G. (2015). Integrated transportation and power system modeling. In *2015 International Conference on Connected Vehicles and Expo (ICCVE)*, pages 379–384.
- Ascher, D. and Hackenberg, G. (2016). The transp-0 framework for integrated transportation and power system design. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 945–952.
- Ascher, D. and Hackenberg, G. (2017). The passenger extension of the transp-0 system design framework. In *2017 5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, pages 256–261.
- Ascher, D. and Hackenberg, G. (2023). A discrete event formalism for fast simulation of on-demand transportation systems. In *International Conference on Intelligent Systems Design and Applications*, pages 185–197. Springer.
- Ascher, D., Hackenberg, G., and Albayrak, S. (2023). Model-based design of integrated transportation systems using approximate dynamic programming. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 4443–4450.
- Barceló, J. and Casas, J. (2005). Dynamic network simulation with aimsun. In *Simulation approaches in transportation analysis*, pages 57–98. Springer.
- Bellman, R. (1957). *Dynamic Programming*. Princeton University Press, New Jersey.
- Brake, J., Nelson, J. D., and Wright, S. (2004). Demand responsive transport: towards the emergence of a new market segment. *Journal of Transport Geography*, 12(4):323–337.
- Ericsson, E., Larsson, H., and Brundell-Freij, K. (2006). Optimizing route choice for lowest fuel consumption—potential effects of a new driver support tool. *Transportation Research Part C: Emerging Technologies*, 14(6):369–383.
- Fagnant, D. J. and Kockelman, K. M. (2014). The travel and environmental implications of shared autonomous vehicles, using agent-based model scenarios. *Transportation Research Part C: Emerging Technologies*, 40:1–13.
- Fellendorf, M. (1994). VISSIM: A microscopic simulation tool to evaluate actuated signal control including bus priority. In *64th Institute of Transportation Engineers Annual Meeting*, volume 32. Springer.
- Furuhata, M., Dessouky, M., Ordóñez, F., Brunet, M.-E., Wang, X., and Koenig, S. (2013). Ridesharing: The state-of-the-art and future directions. *Transportation Research Part B: Methodological*, 57:28–46.
- Gianni, D., D’Ambrogio, A., and Tolk, A. (2014). *Modeling and simulation-based systems engineering handbook*. CRC Press.
- Harris, I., Wang, Y., and Wang, H. (2015). Ict in multimodal transport and technological trends: Unleashing potential for the future. *International Journal of Production Economics*, 159:88 – 103.
- Hu, J., Morais, H., Sousa, T., and Lind, M. (2016). Electric vehicle fleet management in smart grids: A review of services, optimization and control aspects. *Renewable and Sustainable Energy Reviews*, 56:1207–1226.
- Huang, Y., Ng, E. C., Zhou, J. L., Surawski, N. C., Chan, E. F., and Hong, G. (2018). Eco-driving technology for sustainable road transport: A review. *Renewable and sustainable energy reviews*, 93:596–609.
- Jia, D., Lu, K., Wang, J., Zhang, X., and Shen, X. (2015). A survey on platoon-based vehicular cyber-physical systems. *IEEE communications surveys & tutorials*, 18(1):263–284.
- Kopelias, P., Demiridi, E., Vogiatzis, K., Skabardonis, A., and Zafiropoulou, V. (2020). Connected & autonomous vehicles—environmental impacts—a review. *Science of the total environment*, 712:135237.
- Lopez, P. A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y.-P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., and Wießner, E. (2018). Microscopic traffic simulation using SUMO. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2575–2582. IEEE.
- Namoun, A., Tufail, A., Mehandjiev, N., Alrehaili, A., Akhlaghinia, J., and Peytchev, E. (2021). An eco-friendly multimodal route guidance system for urban areas using multi-agent technology. *Applied Sciences*, 11(5):2057.
- Powell, W. B. (2007). *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. John Wiley & Sons.
- W Axhausen, K., Horni, A., and Nagel, K. (2016). *The multi-agent transport simulation MATSim*. Ubiquity Press.
- Wang, B., Han, Y., Wang, S., Tian, D., Cai, M., Liu, M., and Wang, L. (2022). A review of intelligent connected vehicle cooperative driving development. *Mathematics*, 10(19):3635.
- Zhang, R. and Fujimori, S. (2020). The role of transport electrification in global climate change mitigation scenarios. *Environmental Research Letters*, 15(3):034019.