Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

School of Computer Science and Statistics

A Framework for Building Digital Twins of Motorway Scenarios

Swati Poojary

Supervisor: Professor Vinny Cahill

August 19, 2022

A dissertation submitted in partial fulfilment
of the requirements for the degree of
Master of Science in Computer Science (Data Science)

Declaration

I hereby declare that this dissertation is entirely my own work and that it has not been submitted as an exercise for a degree at this or any other university.

I have read and I understand the plagiarism provisions in the General Regulations of the University Calendar for the current year, found at `http://www.tcd.ie/calendar`.

I have also completed the Online Tutorial on avoiding plagiarism 'Ready Steady Write', located at `http://tcd-ie.libguides.com/plagiarism/ready-steady-write`.

Signed: _____ Date: _____

Abstract

A digital twin is a virtual replica of a physical object that is constantly updated through mapping between them. Digital twins have the potential to improve transportation by providing insights into traffic conditions, assisting in traffic management applications, and measuring the effect of infrastructure changes on traffic flow.

The dissertation proposes a framework for building digital twins of the motorway by collecting various sensor data. The framework collects data from motorway sensors using an Apache Kafka-based communication mechanism and supports consuming data from four different sensors: inductive loops, motorway cameras, toll bridge cameras, and probe vehicles. The collected data will be passed to a sensor fusion model which will perform the data processing and error correction. The processed data will be fed into the agent mapping model, which will model the traffic flow. The agent mapping model will use an agent-based approach to simulate the vehicles as agents. Agent-based approach is a simulation modelling technique that treats each vehicle as an agent and models it individually based on its interactions with the environment. Simulation of Urban Mobility (SUMO), a microscopic simulator, will be used to simulate the modelled traffic flow. To advance the vehicles between the sensor readings, SUMO's car following model and lane changing model will be used.

The framework is inspired by the Kalman filter, a sensor fusion technique that predicts and corrects the state using previous data iteratively. Similarly, the concept of our framework is to iteratively estimate the vehicle's state based on the initial sensor data reading and advance the vehicles. The framework estimates the new state and corrects it based on the sensor data.

The framework is being evaluated by creating a digital twin of Dublin's M50 motorway. A SUMO simulation of the M50 motorway serves as the physical entity for this digital twin. The number of total vehicles and the average speed of the entire simulation are used to compare the physical entity and the digital twin. The dissertation also discusses potential future work for the framework to improve performance and accurately model traffic flow.

Acknowledgements

I would like to thank Professor Vinny Cahill for his constant support and guidance throughout this dissertation. The regular meetings and continuous advice helped me to stay in the right direction and complete the dissertation.

I would also like to thank family and my friends Mansi and Kapil for their continuous support throughout the year.

Contents

List of Figures

List of Tables

1 Introduction

A digital twin is a virtual replica of the physical world, and a framework is the basic underlying structure of a system that can be used to build digital twins of any motorway by collecting data from roadway sensors. This dissertation presents a framework for building digital twins of motorways that will aid in the implementation of traffic management systems and the assessment of the impact of infrastructure changes on traffic flow.

1.1 Motivation

Transportation has become an integral part of our daily lives and it is difficult to imagine a world without it. A good transportation system can improve the lives of individuals while also benefiting a country's population, economy, business community, and environment.(*Importance of Transportation in Different Aspects of Life* n.d.)

According to Transport Infrastructure Ireland (*Our National Road Network* n.d.) and the Irish Transport Trends 2021 report published by the Department of Transport's Strategic Research and Analysis Division (*Transport Trends 2021* n.d.), there are 101,316 km of roads in Ireland, of which 13.1% are regional highways and 81.6% are local roads. The primary route between cities and towns consists of 5,293 km, or 5.2% of all roads, of which 995 km, or 18.8% are motorways.

A major part of the every day commute is through roadways, where traffic congestion is a major issue. As the population grows, the transportation system may encounter a number of problems, including traffic congestion, infrastructure maintenance, efficient traffic flow management, monitoring, and alerting users to alternate routes in case of emergencies (Dasgupta et al. 2021). All of these issues must be monitored and analyzed in order to be resolved. A digital twin can help with this process by modelling the traffic flow, monitoring, and improving decision making. A digital twin creates a representation of the real object based on the data exchanged between the two. It has the potential to revolutionize traffic management and operations by creating a link between the actual roadway and the simulation model in order to estimate the product life-cycle and offer practical maintenance

advice to decision-makers.(Gao et al. 2021)

1.2 Objective

The goal of this study is to create a framework for digital twins of motorways by gathering data from the real-world and modelling it with the help of SUMO simulator. We need data that describes the vehicles travelling on the motorway in order to represent it. Data from multiple sensors is collected on real time and fused together to obtain the appropriate data required for modelling traffic flow in the digital twin.

Sensors on moving vehicles can be used to collect data, allowing us to obtain information such as the type of vehicle, its exact location, its speed, the path it follows, and the vehicles in its vicinity. Fixed sensors, such as inductive loops, can be used to collect data such as the number of vehicles that travel pass over them. Traffic signal cameras can provide information about traffic flow. Toll booths can also be used to collect comparable information. Sensor data streams from various sensors are collected at varying frequencies and latency. Some may have missing or incorrect data, while others may fail to supply the necessary information. Despite the fact that some sensors will produce the same data, this data can be fused together to improve accuracy and eliminate errors. The processed data is then fed into the model to create the digital twin.

In our framework, we will collect data from multiple sensors, process it and use agent-based approach to map the vehicles in the digital twin. For evaluating the framework, a digital twin of the Dublin's M50 motorway will be generated. A SUMO simulation of the M50 motorway will serve as the physical entity for the digital twin.

1.3 Approach

The framework supports generating digital twins using four types of sensors: inductive loops, motorway cameras, toll bridge cameras, and probe vehicles. Sensor data is collected using a Kafka-based communication mechanism to build the digital twin framework. This mechanism will collect and transmit data from the real-world simulation as a Kafka Producer. The proposed framework will read the data as a Kafka Consumer in real time and use SUMO simulation to model the traffic flow in the digital twin. The collected data will then be passed onto the senor fusion model, which will process it, correct errors, and predict the state of the vehicles in the simulation. The vehicles are then modelled in the simulation as agents using an agent-based approach.

1.4 Challenges

There are several challenges that are faced in developing the digital twin framework. Some of the challenges and the solutions used to overcome them are discussed below.

Simulating the data in real time

Obtaining real-time data and generating digital twins from it can be difficult because we do not observe the exact route that the vehicle will take, and it is possible that the sensor will stop sending data for that specific vehicle after a certain time. To address this issue, SUMO's car following model and lane changing model are used, which advance the vehicle in the simulation when no data is available for the vehicle.

Mapping data to the vehicles in the simulation

There may be hundreds of vehicles on the road at the same time, and not all sensors can send data with the vehicle's unique identifier. It is difficult to map the vehicles to the simulation in such scenarios.To overcome this, agent based approach is used where each vehicle will be treated as an agent and simulated in a microscopic simulation SUMO which focuses on modelling individual agents based on their high level data.

Evaluating the frameworks performance

The framework built can be used to develop a digital twin for any motorway. Evaluating this framework can be difficult as data from real world needs to be transmitted. For this dissertation a SUMO representation of the M50 motorway will be used as real world and digital twin will be created for this motorway and compared.

1.5 Structure

The rest of the dissertation is divided into five chapters. The second chapter will provide the necessary background work for this project as well as the state of the art for digital twins and sensor fusion. The third chapter will describe the digital twin framework's design and proposed architecture. The fourth chapter will describe the framework's implementation, and the fifth chapter will discuss its evaluation. The final chapter will provide a conclusion and limitations as well as a section on future works.

2 Background and Related Work

This chapter discusses the background information required for a better understanding of the proposed framework. The first section explains digital twins and sensor fusion. The following section discusses the current state of the art in the field of digital twin and sensor fusion. Finally, an overview of all the tools and supporting projects is provided.

2.1 Digital Twins

A digital twin is a virtual representation of a physical object that functions by interacting with the physical object by creating a mapping between them. Using real-time sensor data, the digital twin continuously adapts to changes in the environment. It has the ability to predict and detect potential issues with the physical entity. The physical entity, the sensors

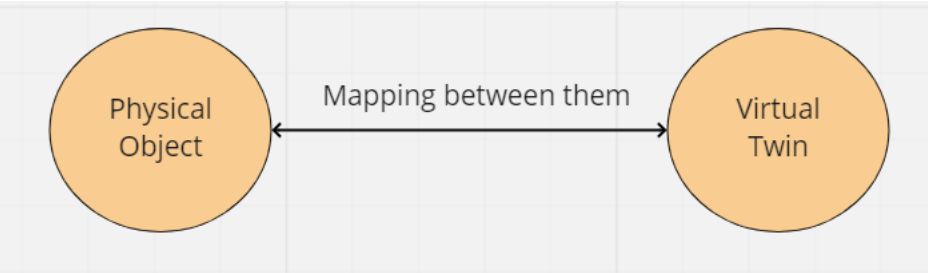

Figure 2.1: Digital Twin

that transmit data from the physical entity, the data collected from the sensors, and the models required for generating digital twins are the main components of a digital twin. It generates a virtual representation of a physical object using the data collected and the models used to process it. Digital twins are currently gaining popularity in industries such as manufacturing, transportation, healthcare, and urban intelligence. By communicating with the physical entity, digital twins can assist these domains in monitoring, evaluating, and assessing risk, as well as providing mechanisms to provide solutions and manipulate the system.

2.2 Sensor Fusion

Building the digital twin necessarily requires the collection and analysis of traffic sensor data; however, data from a single sensor is insufficient, and data gathered from multiple sources may not be consistent or amount to significant data. Sensor fusion, also known as data fusion, is the process of combining data from multiple sensors in order to create a more accurate model of the vehicle by balancing the capabilities of various sensors (*What Is Sensor Fusion?* n.d.). For accurate traffic estimation, data fusion techniques are critical for transforming available data into consistent and complete data. Different sensors have different benefits and drawbacks, such as a camera, which can distinguish between objects well but is easily blinded by dust or weather conditions. While reliable, inductive loops can only identify things based on changes in traffic speed and flow characteristics. Sensor fusion techniques can also be used to correct errors and improve the accuracy of data from a single sensor. The physical entity and the data collected from it are the primary components of the digital twin. Sensor fusion is important in this component because it fuses sensory data to make it more understandable by providing high-level insights.

Sensor fusion can be applied to data from different types of sensors or multiple sensors of

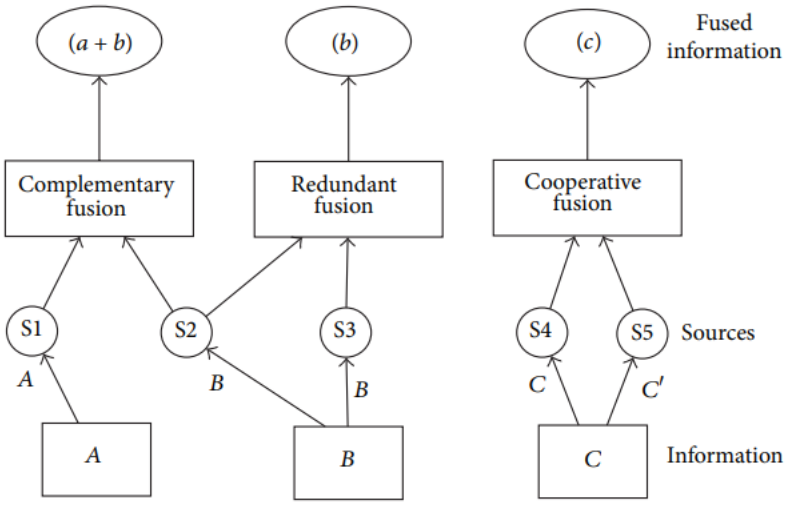

Figure 2.2: Whyte's classification based on the relations between the data sources (Castanedo 2013)

the same type based on the data that they return. Based on the relationship between the various sensors Durrant-Whyte (1988) proposes categorizing sensor fusion into three distinct categories. Complementary: Sensor fusion used for data provided by different sensors that represent different parts of the same vicinity, and by combining the data, it can provide a good overall characteristic. Redundant: The sensor fusion used for the data provided about the same characteristic of the same target. Cooperative: The sensor fusion used for the data

by different sensors which can be combined to generate a new characteristic which is not initially present (Castanedo 2013).

Varshney (1997) also proposes a three-level data fusion model, from low to high level, with the low level model's results serving as an input for the high level model. The first level of data fusion involves processing raw sensor data to estimate an object's basic state. For example, data is collected from multiple sensors such as inductive loops, cameras, and probe vehicles and combined to obtain basic traffic information such as speed and flow in order to model the traffic. The second level of data fusion is used to derive data features and patterns based on the first level's estimated states. The third level of fusion is used to predict events and make decisions based on the second level's features and patterns.

In this framework, we are processing data for the first level in order to obtain the basic traffic state estimation needed to model the traffic flow in the digital twin. Several techniques for this level of data fusion are discussed further below.(Ou 2011)

Kalman filter

The Kalman filter provides the best estimate of the state through an iterative process of predicting and correcting the state. It predicts and corrects the state iteratively using the initial data by computing the Kalman Gain to make the filter an optimal estimator.

The image 2.3 above explains the Kalman filter and implies that as the data becomes more

$$\mathbf{K}_k = \frac{\text{uncertainty process model}}{\text{uncertainty observation model \& data}} \times \text{sensitivity obs. model to state variables}$$

Figure 2.3: Kalman Filter (Ou 2011)

uncertain, more weight is placed on the prediction and vice versa. The Kalman filter also adjusts the observation model's sensitivity to changes in the state variables proportionally.

The solution can be provided for models with linear or Gaussian data, i.e. normal distribution. It is incapable of providing accurate information to the non-linear system. There are two variations of Kalman filters: Extended Kalman Filter and Unscented Kalman Filter. For slightly non-linear systems, an extended Kalman filter that linearizes the data using Jacobians and Taylor Series can be used. The unscented Kalman filter is an improvement on the extended Kalman filter. It linearizes the data using sigma point

prediction and requires a large dataset for prediction.

Particle filter

This is also an iterative processing filter with two prediction and correction phases, similar to the Kalman filter, but it can process data with nonlinearity and a non-Gaussian system. It is a simulation-based technique capable of achieving Bayesian optimal estimation. It provides a global approximation of the data and outperforms the extended and unscented Kalman filters when there is a large amount of data.

Linear Programming

This technique uses historical data to estimate the current state. It processes data in large batches to find the largest or smallest value for optimal estimation. The value is determined by objective functions and is constrained by linear inequality constraints. Linear programming has a lower computational cost than the Kalman filter and can produce results with a wide range of values.

Treiber filter

It is based on spatiotemporal characteristics; for example, in traffic travel estimation, it considers speeds in both congested and free flow conditions. It is similar to image processing in that it can only be applied to data of the same type.

2.3 State of the Art in Digital Twins

The digital twin's main concept is to collect data from various sensors and create a virtual representation of the physical world. The data collected from the sensors cannot be directly used to model the digital twins. Sensor fusion techniques must be used to ensure that the data is consistent and complete. Digital twins are used vastly in industries such as manufacturing, transportation, and healthcare. There is a lot of research going on with digital twins and sensor fusion. The following section discusses the current state of the art in digital twins and sensor fusion.

State of the Art for Digital Twins

Nowadays, transportation is such an important part of our daily lives that any minor delay or congestion in our daily commutes has a significant impact on our lives. One such mode of transportation is the airplane. Thousands of people travel around the world on flights, and

even minor delays have a significant impact. The airport is a critical factor in the smooth operation of the aerospace department. If the airport does not function properly, it may cause problems such as runway collisions and ground vehicle congestion. One such study (Saifutdinov et al. 2020) proposes the use of digital twins to develop and test solutions in the field of centralized ground transportation control at airports to address these issues. The proposed model simulates various scenarios in the transportation network that require the use of a centralized control system. The spatial coordinate details received from the surface movement radar (SMR) and onboard systems of the vehicles are used as input for these models. The scheduling-routing system also collects data about the vehicle's state. Machine learning and artificial intelligence algorithms are also used in the proposed solution. The digital twin is created by simulating the processes in the transport network of any airport under consideration using Ground Traffic Scenario Simulation (GTSS). The study is structured around three major components: 1) A simulation model that serves as a data source for the Digital Twin; 2) The Digital Twin, which is a dynamic information model of the observed and controlled process; and 3) Applications, which are computer programs that use the data collected by the Digital Twin. The 5-15 minute simulation scenarios allow the potential user to specify which vehicles are involved in traffic during the simulation time by specifying the time of their appearance as well as the route with all stop points.

Another study (Wang et al. 2021) suggests using the Unity game engine to create a framework for creating a digital twin simulation of connected and autonomous vehicles (CAVs). The proposed architecture considers both the physical and digital worlds. The physical world consists of simulated real-world objects such as road networks, ego CAVs, and other vehicles, whereas the digital world consists of the three sub layers of the unity game engine: 1)Unity game objects are used to build the hardware and as the main simulation platform, 2)Unity scripting API is used for software, and 3)External tools such as SUMO, Python, and Amazon Web Services (AWS) are used to enhance the simulation. Using data from the current trip as well as all previous trips in the driver database, the "Personalized Adaptive Cruise Control" (P-ACC) algorithm is used to generate the driver's preferred time gap value. The results of this algorithm are delivered to the ego vehicle in Unity when the driver activates the automated control, where they can be applied to the P-ACC algorithm. The paper mentions the algorithms and methodology used to build the digital twin, as well as test scenarios and applications once it is formed, but it does not go into detail about the process.

These studies explain the algorithm used as well as the input and output data, but they lack information on the evaluation process used to measure the framework's performance.

State of the Art for Sensor Fusion Techniques

Collecting data is not enough to create a digital twin; the data must also be processed and combined in order to be used as an input for the model. There are many techniques available for sensor fusion. One such technique is discussed in the study by Dong & Evans (2007) which explains the data fusion techniques to combine data from inductive loops and cameras. The main purpose of this study is to use these two sensors to determine the state of the road. The proposed system would collect sensor data, process it, and present the results to an operator via a single user interface. Because each sensor has its own set of advantages and disadvantages, combining them can improve detection rate (DR), reduce false alarm rate (FAR), and provide built-in redundancy. Loop detection methods are used to detect cars traveling through the induction loop by constructing the loops in pairs, and the vehicle is simulated from the entering loop to the exit loop, assisting in the calculation of vehicle length and speed measurement. When vehicles or obstacles are visible within the image analysis zone, image analysis algorithms are used to identify the vehicles using data from the cameras, and an image analysis alert is triggered. After applying the algorithms to the individual sensors, Bayesian analysis is used to combine the data from image analysis and loop sensors modules providing detections of the road-state and associated confidence measures. The study describes the algorithm used for the individual sensors in details but does not provide much information about the Bayesian analysis.

The study by Houbraken et al. (2015) proposes a system that fuses floating car data (FCD) with stationary detector data (SDD) (cameras, inductive loops) to construct a traffic state estimation using adaptive smoothing technique, also known as the extended and generalized Treiber-Helbing filter (EGTF). This filter combines individual data into a comprehensive traffic state estimation using kinematic wave theory applied to road traffic. To estimate the traffic state, the road network is represented as a dynamic system in which traffic flows along the road from the origin to the destination while considering free flow and congested traffic. This document offers data source normalization strategies to avoid bias toward a single source. The system was tested along a 20-kilometer stretch of the A58 highway. Individual data samples can be integrated in a single data fusion method to estimate traffic state by taking them into account and normalizing them. According to the findings of this study, this corrects individual data source bias, resulting in a more accurate prediction of traffic state.

2.4 Tools

This section gives a brief description of all the tools that are used for building the framework.

2.4.1 SUMO

Simulation of Urban Mobility (SUMO) is an open source microscopic traffic simulation engine designed to handle large road networks. As each vehicle has its own route, a microscopic simulation models each vehicle individually as it moves through the network. Every vehicle in SUMO is simulated by taking its own capabilities and driver behaviour into account. For simulating a traffic flow, we need network information, traffic infrastructure and demand. SUMO provides a number of tools for working with these components, as well as an SUMO-GUI application for visual simulation observation (*SUMO User Documentation* n.d.).

SUMO provides traffic demand modelling by defining a vehicle trip or route. A trip is the movement of a vehicle defined by the starting and ending edges, whereas a route is an expanded trip defined by specifying all the edges the vehicle will move. SUMO also has its own inbuilt car following and lane changing model for smoothly simulating vehicles. The car following model defines the vehicle's movement and speed in relation to its leading vehicle by constantly attempting to maintain a safe distance between them, whereas the lane changing model determines the lane changing strategy based on the traffic conditions around the vehicle. In this project, we use SUMO to represent the road network and simulate the vehicles in the network using sensor data. We also use SUMO's car following and lane changing model to advance the vehicle in the simulation (Abidin et al. 2015).

SUMO also has tools for generating and storing simulation output files, which can be triggered from the command line while starting the simulation. The output files can then be used to visualize the simulation results by passing them through the visualization tools. To evaluate our framework, we generate two output files: tripinfo.xml and statistics.xml.

The tripinfo.xml output file contains information about every vehicle. It includes the vehicle's departure time, speed, duration, speed factor, and many other parameters. The statistic.xml output contains average simulation details such as total vehicle load, total running vehicles, overall trip details, and so on.

```
<tripinfos xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="http://sumo.dlr.de/xsd/tripinfo_file.xsd">
    <tripinfo id="camera73S" depart="31.00" departLane="106130759.791_2" departPos="379.47" departSpeed="0.00" departDelay="0.00"
              arrival="34.50" arrivalLane="106130759.791_2" arrivalPos="394.23" arrivalSpeed="9.10" duration="3.50" routeLength="14.76"
              waitingTime="0.00" waitingCount="0" stopTime="0.00" timeLoss="2.71" rerouteNo="0" devices="tripinfo_camera73S ssm_camera73S"
              vType="DEFAULT_VEHTYPE" speedFactor="0.83" vaporized=""/>
```

Figure 2.4: Screenshot of tripinfo.xml

```
<statistics xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="http://sumo.dlr.de/xsd/statistic_file.xsd">
    <vehicles loaded="4800" inserted="4800" running="1252" waiting="0"/>
    <teleports total="29" jam="6" yield="0" wrongLane="0"/>
    <safety collisions="23" emergencyStops="0"/>
    <persons loaded="0" running="0" jammed="0"/>
    <vehicleTripStatistics routeLength="544.30" speed="8.41" duration="718.65" waitingTime="0.68" timeLoss="8.09"
                           departDelay="0.00" departDelayWaiting="0.00" totalTravelTime="3449543.50" totalDepartDelay="0.00"/>
    <pedestrianStatistics number="0" routeLength="0.00" duration="0.00" timeLoss="0.00"/>
    <rideStatistics number="0"/>
    <transportStatistics number="0"/>
</statistics>
```

Figure 2.5: Screenshot of statistic.xml

2.4.2 TraCI

The Traffic Control Interface (TraCI) provides users with access to running road simulations and allows them to retrieve and manipulate simulated objects. TraCI connects to SUMO using a client/server TCP architecture. Sumo acts as a server, waiting for all applications to connect and take control and simulate continuously until a client requests that the simulation be terminated. TraCI offers full simulation control by providing various methods for changing the behavior of the simulation's objects. We can use TraCI to introduce new vehicles into the simulation or modify existing ones. TraCI provides a number of methods for extracting and modifying vehicle states such as speed, route, and location. We can also retrieve and change the values of other objects such as edges, traffic lights, and inductive loops (*Introduction to TraCI* n.d.).

In this framework we are using TraCI along with python. We begin the simulation with TraCI and use it in our model to add new vehicles using the traci.vehicel.add method and change their location using the traci.vehicel.moveToXY method by providing longitude and latitude values. TraCI serves as the link between models and simulation in our framework, allowing us to control and modify the digital twin representation.

2.4.3 Apache Kafka

Apache Kafka is an event streaming platform that allows users to publish and subscribe to event streams. Kafka also has the ability to store and process event streams. It is a distributed system that communicates between clients and servers using the high-performance TCP protocol, making it highly elastic, fault-tolerant, and scalable. It works as a messaging system that uses a producer-consumer architecture, with producers being the applications that publish events to Kafka and consumers being the users that subscribe to receive and process the events (*Kafka documentation* n.d.).

Due to the total decoupling between producers and consumers, the producer need not wait for the consumer to connect before publishing the data. Events are categorized and

broadcast as topics, and each topic can be further subdivided into partitions distributed across multiple Kafka brokers. Multiple producers and subscribers can publish and read data simultaneously on the topics which enables parallel processing.

To retrieve data from the sensors, we use an Apache Kafka-based communication mechanism in this framework. The data is collected and transmitted by the Kafka producer, and our framework will read the data as a Kafka consumer.

2.5 Supporting Projects

This section discusses the projects that help our framework implement various digital twin components.

2.5.1 A Communication Architecture for Transportation Digital Twins using Apache Kafka

The study proposes by Fennell (2022) is a communication service architecture between the digital twin and the motorway sensors using Apache Kafka. It examines the sensor's characteristics and the type of data it produces. The data is then collected from the physical entity, which is a simulation of Dublin's M50 motorway. Data collected by the architecture is from four different sensors: inductive loops, motorway cameras, toll bridge cameras, and probe vehicles. Sensor data is divided into Topics based on sensor type, and each Topic contains partitions for individual sensors. The data is streamed to an Apache Kafka broker hosted on Amazon AWS EC2. The collected data is transmitted as a Kafka producer.

This proposed architecture will be used as a communication mechanism in our framework to collect sensor data. It serves as a middle-ware between the sensors and the digital twin. Our framework will act as a Kafka consumer, gathering data from all four sensors.

2.5.2 Quantifying the impact of connected and autonomous vehicles on traffic efficiency and safety in mixed traffic

The paper by Gueriau & Dusparic (2020) investigates the impact of connected and autonomous vehicles (CAVs) on three types of networks, simulating traffic flow with the help of SUMO of vehicles with varying levels of automation using historical Irish road traffic data. CAVs' impact is assessed on three networks: an urban network, a 17.1 km stretch of national road, and a 7 km four-lane motorway with two intersections. The evaluation traffic flow is based on data collected in Ireland between 2012 and 2019 with various traffic loads: congested, free-flow, and saturated. It provides a 24-hour simulation of traffic flow.

We will evaluate our framework using one of the research networks. The M50 simulation of a 7-kilometer four-lane motorway will be used to generate the digital twin of the M50 motorway, which will include two significant interchanges with national road junctions N7 and N9. The SUMO simulation of the motorway will serve as the physical counterpart to our digital twin.

3 Design

This chapter describes the proposed framework design by first explaining the architecture and then detailing the components of the digital twin.

3.1 Architecture

The framework's proposed architecture includes several components such as the physical entity, a communication mechanism with Kafka Producer and Consumer, data pre-processing sensor fusion models, agent mapping models, and a digital twin simulation.

The physical entity that describes the network and the sensors available on the network is represented by a SUMO simulation of the real-world motorway. These sensors are modelled in order to represent the actual sensors on the roads. The digital twin framework receives data from these sensors. To collect the data, an Apache Kafka based communication mechanism is used. The data will be collected by the communication mechanism and published as a Kafka Producer and as a Kafka Consumer, our framework will consume the data. Our framework supports generating the digital twin with four different sensors: Inductive loops, Motorway cameras, Toll bridge cameras and Probe vehicle. The data from each sensor group is transmitted as a separate Topic and our consumer will get these data from four different topics. The data is collected in real time when the physical entity simulation is running. The Kafka consumer will continue to retrieve data until the Producer publishes it.

Data from various sensor types will be sent to their respective sensor fusion models. Some of the data may be inaccurate, and some data may be missing. The sensor fusion model's primary responsibility is to process and correct data. It is also in charge of calculating the missing parameters based on the existing ones. Cameras, for example, transmits data about the vehicle's speed, direction, and distance. The sensor fusion model will calculate the vehicle's approximate location using these data.

After pre-processing, the data is sent to the Agent mapping model, which is in charge of

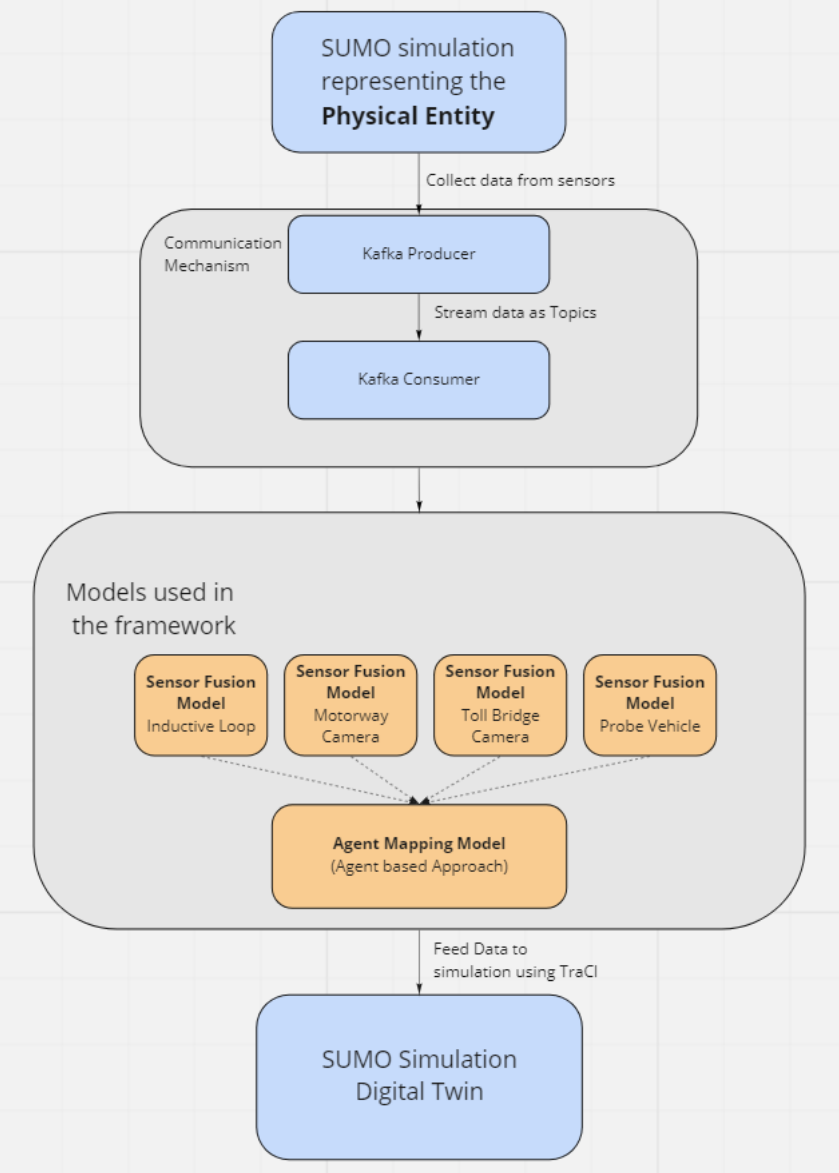

Figure 3.1: Proposed Architecture

mapping the vehicles in the simulation. To map the vehicle in the simulation, an agent-based approach is used, with each vehicle treated as an agent. SUMO simulation is used in our framework to model traffic flow which is a microscopic simulation concerned with modelling individual agents based on their parameters. The agent mapping model will determine whether the vehicle is already present in the simulation based on the parameter calculated in the sensor fusion model. If the vehicle is already present, the state and behaviour of the vehicle are updated; otherwise, a new agent is introduced into the simulation. The model and simulation interact via TraCI, a traffic control interface that allows access to retrieve and manipulate the simulation object.

3.2 Components of Digital Twin

A digital twin consists of a physical entity, a virtual simulation, and the link between them. Our proposed architecture is divided into four components: Physical motorway entity, Motorway sensors, Data collection, and Models.Each of the elements will be discussed in detail below.

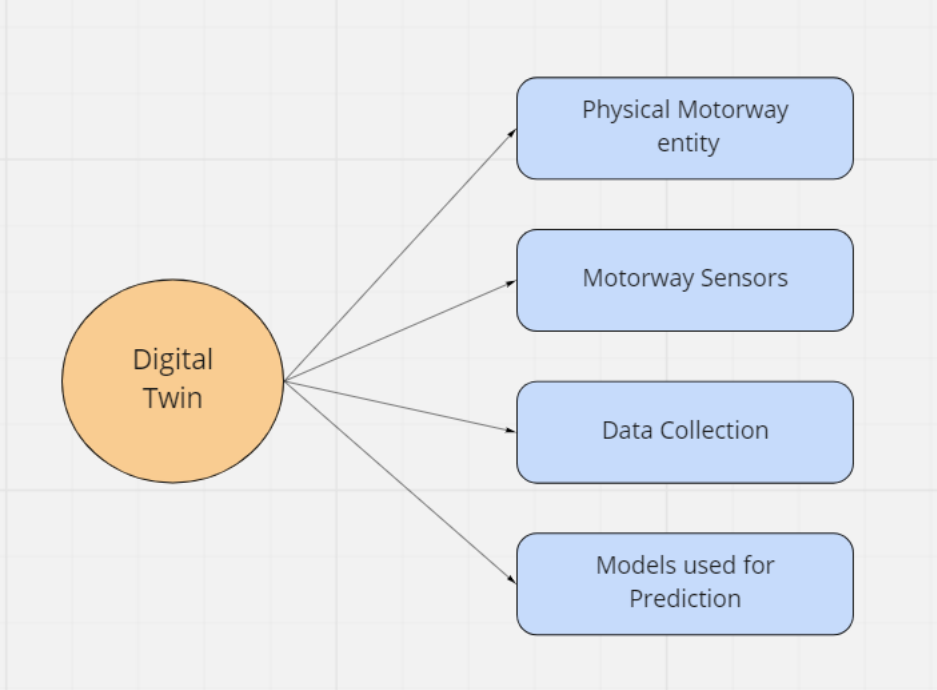

Figure 3.2: Components of Digital Twin

3.2.1 Physical Motorway entity

The most important component in creating a digital twin is the real-world motorway. The network for the digital twin will be built to replicate the lanes and intersections, as well as the sensors installed in the same locations as the physical motorway entity, in order to recreate it. Using a real-world motorway as a physical entity may be challenging due to the difficulty of collecting data from all sensors in real time; thus, a SUMO simulation representing the real-world motorway will be used as a physical entity, describing its key features.

3.2.2 Motorway Sensors

Sensors are essential when creating a digital twin because they act as a mapping between the physical world and the digital twin. Sensors collect data from the real world, which is then transmitted to the digital twins. We need a lot of data to recreate the traffic flow, such as vehicle speeds, locations, and vehicle counts. Since all of these data cannot be provided by a

single type of sensor, multiple sensors must be used to collect and combine the data. We are using a SUMO simulation as the physical entity, hence the virtual sensors are modelled in the simulation which is intended to be an accurate representation of the real-world sensors found on motorways. The proposed architecture supports four types of sensors: an inductive loop, a motorway camera, a toll bridge camera, and probe vehicles.

Inductive Loops

An inductive loop functions similarly to a metal detector, which is commonly used to detect traffic by monitoring changes in the field when moving vehicles pass over it. It is made of a coiled wire loop that is installed into or beneath the road's surface. For the duration that the vehicle is passing over the loop, a change in the magnetic field is created in the loop. This shift in the magnetic field is interpreted as a demand for the vehicle by the controller(*Inductive loops* n.d.).
In our framework, the inductive loop data is used to verify the number of vehicles that have passed through the loop, ensuring proper replication of the vehicle route. The inductive loop data is received from the publisher every second, with an estimated accuracy of 95%(Fennell 2022).

The following is the format of the data collected:

loop id	lane 1	lane 2	lane 3	lane 4	timestamp
Unique identifier for the set of inductive loops.	Vehicle count for the inductive loop on lane 1.	Vehicle count for the inductive loop on lane 2.	Vehicle count for the inductive loop on lane 3.	Vehicle count for the inductive loop on lane 4.	The simulation timestamp the measurements were taken. This is independent of the true time outside of the simulation.

Table 3.1: Inductive loop data

Motorway Cameras

Cameras can help you identify and categorize vehicles on the road. They can provide additional information such as the direction, speed, and lane of the vehicle. Cameras are less expensive than other sensors, and they can capture images of their surroundings as well as identify moving and stationary objects in their field of view.
The motorway camera data received from the publisher returns data from up to 200 meters away and is oriented north and south for ease of use. The detection accuracy ranges between 92 and 98 percent. With a 30 frames per second camera, the expected speed measurement error rate is 2%, while the distance inaccuracy is 5% (Fennell 2022) . It

returns the direction, speed, and distance between the camera and the vehicle, which can be used to predict the vehicle's location. It also returns the vehicle's lane details, which are used to predict the vehicle's route and simulate the entire flow of the vehicle.

The format of the data collected is as below:

camera id	lane id	lane index	direction	distance	speed	timestamp
Unique identifier for the camera.	Which road the vehicle is travelling on.	Which lane on the road the vehicle is in.	The direction the vehicle is travelling.	The distance between the vehicle and the camera.	The speed in kilometres per hour the vehicle is travelling.	The simulation timestamp the measurements were taken. This is independent of the true time outside of the simulation.

Table 3.2: Motorways Camera data

Toll Bridge Cameras

Toll bridge cameras have the same features and benefits as motorway cameras. The toll bridge cameras are mounted on toll booths and return the same type of data along with additional information about the vehicle's class. The accuracy and error rate of the data obtained from the publisher are the same as those of the motorway camera. Because both cameras produce the same type of data, the model used to process them is the same (Fennell 2022). The data are used in the same way to calculate the vehicles' location and route.

The format of the data collected is as below:

lane id	lane index	direction	distance	speed	class	timestamp
Which road the vehicle is travelling on.	Which lane on the road the vehicle is in.	The direction the vehicle is travelling.	The distance between the vehicle and the toll bridge.	The speed in kilometres per hour the vehicle is travelling.	The class of the vehicle.	The simulation timestamp the measurements were taken. This is independent of the true time outside of the simulation.

Table 3.3: Toll Bridge Camera data

Probe Vehicles

Probe vehicles, also known as floating cars, are vehicles that participate in traffic flow and have the ability to assess traffic conditions and transmit these details to a traffic center. The probe vehicles has a inbuilt location sensor to track its own location, and a communication sensor to communicate traffic data.

The data returned from the publisher also returns data from some third part enterprises like the real-time bus tracking, Google maps, sat-nav, and other sources that provide GPS. The GPS is accurate to a maximum of 20 meters for services like Google maps and a maximum of 3 meters for devices like the Garmin 16x GPS device. The estimated frequency of the probe vehicles is 1 HZ. The probe vehicle sensors are predicted to have an inaccuracy of up to 10 meters (Fennell 2022).

These sensors provide data with a unique identifier and the speed of the probe vehicle, as well as longitude and latitude information. These details will be used to map the vehicles to the simulation and advance the vehicles accordingly. The route of the vehicle is identified using previous locations to complete the vehicle flow.

The format of the data collected is as below:

probe id	location	speed	vehicle type	timestamp
A unique identifier for the vehicle.	The latitude and longitudinal location of the vehicle.	The speed in kilometres per hour the vehicle is travelling.	The type of vehicle.	The simulation timestamp the measurements were taken. This is independent of the true time outside of the simulation.

Table 3.4: Probe vehicle data

3.2.3 Data Collection

The data will be collected from the sensors mentioned in the previous section, but there must be a mechanism in place to carry out the data collection and transmission process. To accomplish this, an Apache Kafka-based communication mechanism Fennell [2022] will be used, which will collect and stream data to the Kafka broker deployed on the Amazon AWS EC2 instance. It serves as a producer, sending data streams to the Kafka broker endpoints. By specifying the same broker endpoint information, our project will act as a Kafka consumer and receive data from the publisher. The publisher will send the data as a topic, with a separate topic created for each sensor type. Since our framework supports four types of sensors, data will be streamed in four different topics.

3.2.4 Models used for Prediction

Once the data is collected, it must be processed to generate the digital twin, and the vehicles must be mapped to the simulation in order to model the traffic flow. To accomplish these tasks, we will create two models in this framework. An agent mapping model and a sensor fusion model. In addition, the framework will use SUMO's car following and lane changing models to help advance the vehicles in the simulation.

Sensor Fusion Model

The collected data is used in the digital twin to represent traffic flow, but it must be processed and error corrected before being fed to the model. For each type of sensor, a sensor fusion model is created that is in charge of data processing, error correction, and data fusion.

Some sensor data may have missing or incorrect values. These parameters will be corrected by this model. It will also use the existing parameters to calculate the parameters that are not present in the data. This model's input will be data returned from the sensor, and its output will be processed data.

Each sensor will have its appropriate sensor fusion model which will process the data for that particular sensor. Both motorways cameras and toll bridge cameras use the same sensor fusion model because they return the same type of data such as speed, distance, direction, and vehicle edge. We need to know the location of the vehicle in order to replicate its flow in the simulation. This model will calculate the approximate location of the vehicle based on the vehicle's direction, speed, and distance from the camera. The vehicle's path is predicted based on the edges it has traveled on to complete the vehicle flow. The inductive loop model verifies the number of vehicles that have passed through the loop for each timestamp. The probe vehicle's data will be fed into its own sensor fusion model. The model is in charge of predicting the exact location of the vehicle while accounting for any inaccuracies in the data.

Agent Mapping Model

Following the completion of pre-processing and data fusion, the data is transferred to the Agent mapping model, which employs an agent-based approach to treat each individual vehicle as an agent and map it to the agents in the simulation. Not all sensors return the unique vehicle identifier, it is up to this model to predict which agent the data will be mapped to. The processed data from the sensor fusion model will be the input for this model, and the output will be the mapping of the vehicles as agents in the simulation.

Agent based approach: [Bonabeau 2002] A powerful simulation modelling technique that models individual vehicles as agents based on their interactions with their environment is the agent-based approach. One advantage of using an agent-based approach is that it is flexible, allowing easy addition and removal of the agents. When using an agent-based approach in conjunction with microscopic modelling, the agents can be modelled individually in the simulation. Each agent will assess its own situation and make a decision based on a set of rules. The agents will constantly interact with the model in order to predict its next state.

Sumo's car following model and lane changing models are used to advance the agents through the simulation for each time step after mapping the data to the simulation. To ensure a smooth and accident-free simulation, these models are configured with a minimum gap value between vehicles while car following and strategies for efficient lane changing.

4 Implementation

The main components required for developing the digital twin framework are the physical entity and the data from the physical entity. For these components, the supporting projects described in section 2.5 are used. This chapter describes how the proposed architecture is implemented. It goes into great detail about the models used in the framework.

4.1 Framework Overview

The proposed framework allows for the creation of digital twins for motorways using four types of sensor data collected from the physical entity. Data is gathered using an Apache Kafka-based communication mechanism. As described in section 3.2.3, the Kafka Producer will collect and transmit sensor data, while our framework will act as a Kafka consumer, retrieving data in real time. The Kafka producer sends data from each sensor group as a separate topic. Each topic is also partitioned based on the number of sensors used. Regardless of the number of sensors, our Kafka consumer will consume all data and process it in a loop. After that, each data point will be forwarded to the appropriate sensor fusion model for processing. The framework's concept is to map the vehicles as agents in the simulation. To accomplish this, the processed data is passed to the agent mapping model, which uses an agent-based approach to map the vehicle in the simulation. Agent-based modelling is a technique for simulating vehicles as agents based on their interactions with their surroundings.

```
from kafka import KafkaConsumer

consumer = KafkaConsumer(bootstrap_servers=${bootstrap_servers})
consumer.subscribe(topics=${topic_name})

for msg in consumer:
    #Process the message
    #Feed it to SUMO as a input
```

Figure 4.1: Kafka consumer code

The framework idea is inspired by the Kalman Filter. The Kalman filter is a sensor fusion technique that predicts and corrects the state using previous data iteratively. Similarly, our

framework's concept is to estimate the vehicle's state based on the initial sensor data reading. Once the vehicle has been added to the simulation, Sumo's car following and lane changing models are used to advance the vehicles between senor readings. The new state estimation is calculated, and the estimated state is corrected based on the data when the senor data is read. This is an iterative process of predicting and correcting the vehicle's state estimation.

4.2 Models

The framework has two major models: sensor fusion model and agent mapping model. Sumo's car following and lane changing models, in addition to these, are used to advance the vehicles in the simulation. The detailed description of all the models are provided below.

4.2.1 Sensor Fusion Model

A sensor fusion model is in charge of data processing, error correction, and predicting the correct values for missing or inaccurate data. A sensor fusion model is created for each type of sensor. The model's input is the data gathered from the physical entity via the communication mechanism, and the model's output is the processed data.

The same sensor fusion model is used for both the motorway and toll bridge cameras because they provide similar data and have similar inaccuracies and noise. The cameras in the simulation are designed to represent real-world camera sensors mounted on the road. It has the ability to collect and return data from their north and south orientations. We know the exact location of the camera and the edges in its vision for the north and south orientations. The closest edge to the camera is determined by measuring the distance between the camera and each edge. In addition, the camera's bearing towards the nearest edge is calculated. The direction measured in degrees from true north is referred to as the camera bearing. Both of these data points can be used to calculate the vehicle's location in the simulation. The camera data returns the vehicle's direction and speed, as well as the distance between the vehicle and the camera. The approximate longitude and latitude of the vehicle are calculated using the returned sensor data and the calculated bearing and closet edge.

The probe vehicle data returns the unique identifier, speed and the location details of the vehicle. Since we have the longitude and latitude data it is easier to introduce or modify the vehicle in the simulation. Even though we have the location details we are unaware of the path that the vehicle follows . All of the edges that the vehicle has travelled on are computed using the vehicle's locations, and the path is calculated based on that. The sensor

fusion model also adjusts the vehicle's speed if it moves too far ahead or too far behind in the simulation and the sensor data is not in sync with the simulation. TraCI includes methods for slowing down and explicitly setting the vehicle's speed.

```
traci.vehicle.setSpeed(vehID=${vehicleID}, speed=${speed})

traci.vehicle.slowDown(vehID=${vehicleID}, speed=${speed},
                       duration=${duration})
```

Figure 4.2: TraCI speed methods

4.2.2 Agent Mapping Model

Once the data is prepared, the main challenge is to map the agents to the simulation. To overcome this challenge, an agent mapping model based on the agent-based approach is used. An agent-based approach makes decisions for the agents based on their interactions with their environment. The primary responsibility of this model is to map the agents in the simulation and change the state and behavior of the agents based on the data processed by the sensor fusion model.

The data from the inductive loop returns the number of vehicles that passed through the lane. This count is compared to the count of vehicles that have passed through the inductive loop of the digital twin simulation at each time step. TraCI provides a method for obtaining the ids of the vehicles that have passed through the inductive loop.

```
traci.inductionloop.getLastStepVehicleIDs(loopID=${loop_id})
```

Figure 4.3: TraCI inductive loop

For the cameras, we calculate the vehicle's approximate longitude and latitude in the sensor fusion model. The agent mapping model will compare all of the simulation's existing agents in the vicinity of the location. If an agent is present within this range, the data is mapped to the agent; otherwise, a new agent is introduced into the simulation.

Agent mapping is easier for probe vehicles as it contains a unique identifier of the vehicle. The model simply determines whether an agent with the specified id exists in the simulation; if so, the agent is updated; otherwise, a new agent is introduced. TraCI includes methods

for adding and modifying vehicles in the simulation. The moveToXY method of TraCI can be used to change the location of the vehicles based on their longitude and latitude.

```
traci.vehicle.add(vehID=${vehicleID}, routeID=${route_id})


traci.vehicle.moveToXY(vehID=${vehicleID}, edgeID=${edge_ID},
                       lane=${lane_ID}, x=${longitude}, y=${latitude})
```

Figure 4.4: TraCI adding and modifying vehicles

4.2.3 Sumo Car following model and Lane changing model

Sumo car following and lane changing models are used to advance the vehicle in the simulation between every sensor reading. It ensures that the simulation is smooth and accident free. Car following models explain how vehicles follow one another in a traffic flow one at a time. It establishes a vehicle's speed in relation to the vehicle in front of it. It demonstrates how the vehicle in front is constantly attempting to maintain a safe distance from the following vehicle and the following vehicle always adjusts to the leading vehicle's deceleration behaviour[SONGa et al. 2014]. The Intelligent Driver Model is the car-following model employed here (IDM). Based on the optimal velocity model (OVM), it only responds to the distance from the leading vehicle and ignores the speed of the car in front of it[Bieker-Walz et al. n.d.].

The lane changing model decides whether or not to change lanes for a vehicle based on the traffic conditions around it. While performing the lane change, it also adjusts the speed of the vehicle and the obstructing vehicle. In our framework we are using the Sumo's default lane changing model LC2013. [Erdmann 2014]

5 Evaluation

This chapter discusses the framework evaluation process in order to assess the feasibility of the proposed approach. It will also be useful in identifying the approach's limitations and the measures needed to extend the framework.

5.1 Test Setup

We will create a digital twin of Dublin's M50 motorway to evaluate the framework. The SUMO simulation of the M50 motorway created in the project by Gueriau & Dusparic [2020] is used for the physical entity. The simulation includes a 7-kilometer motorway network with two major interchanges with national road junctions N7 and N9.

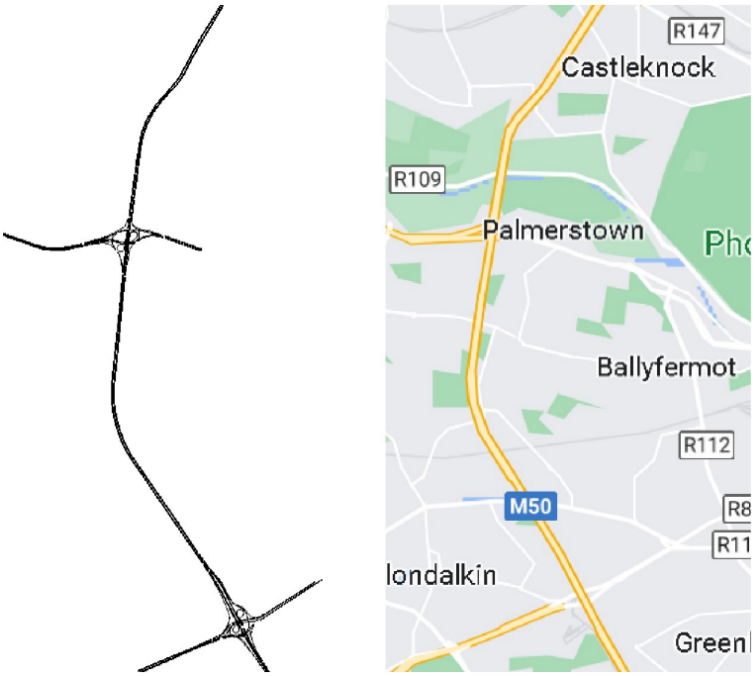

Figure 5.1: M50 motorway and its SUMO simulation [Fennell 2022]

5.2 Scenarios and Evaluation metrics

The SUMO simulation of the motorway offers 24 hour simulation. For evaluating the digital twin in all the traffic conditions, the simulation will be executed for three different time periods. All scenarios will be executed thrice to obtain the model's average results and ensure that the result is not biased.

For comparing the digital twin with the physical entity and to measure how successfully the digital twin is modelling the traffic flow, the total number of running vehicles in the simulation will be compared. In addition, the average speed of the vehicles in the entire simulation will be compared. SUMO generates a tripinfo.xml and a statictics.xml output file during the simulation to gather this data. The trip info file contains information about the vehicles in the simulation that is relevant to specific vehicles, such as the departure position, arrival position, duration, speed factor, and so on. The statistics output file contains general information about the simulation, such as total vehicles loaded, total vehicles running, trip and pedestrian details, and so on. With the generated output, bar plots will be created for quick visual comparison of data.

5.2.1 Scenario 1

The first scenario considered for evaluation is morning traffic on congested roads. From 9:00 a.m., a digital twin is created for a 15-minute simulation. The digital twin is run three times to ensure that the data is not biased, and the average data is used for comparison with the physical entity.

Comparing the total number of running vehicles

We will compare the total number of vehicles running in the simulation for this experiment. The digital twin is run three times, and the results of all three runs are compared to the physical entity's results. We are using the SUMO simulation of the M50 motorway from the supporting project as the physical entity for our digital twin for evaluation purposes. Because this is a simulation, the results will be the same for all three runs.

Figure 5.2 depicts a bar plot for the number of running vehicles in the physical entity, followed by the number of running vehicles for the three digital twin executions.
The above plot shows that the number of running vehicles is lower in the digital twin than in the physical entity. The number of running vehicles for the three execution of the digital twin also slightly differs.

The following plot in figure 5.3 shows a bar plot comparing the number of vehicles in the

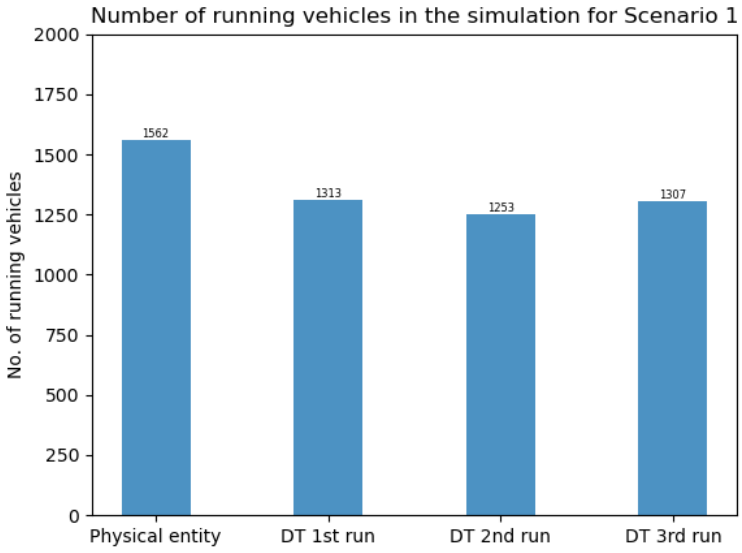

Figure 5.2: Total number of running vehicles for scenario 1

physical entity to the results of the digital twin's average of three executions.

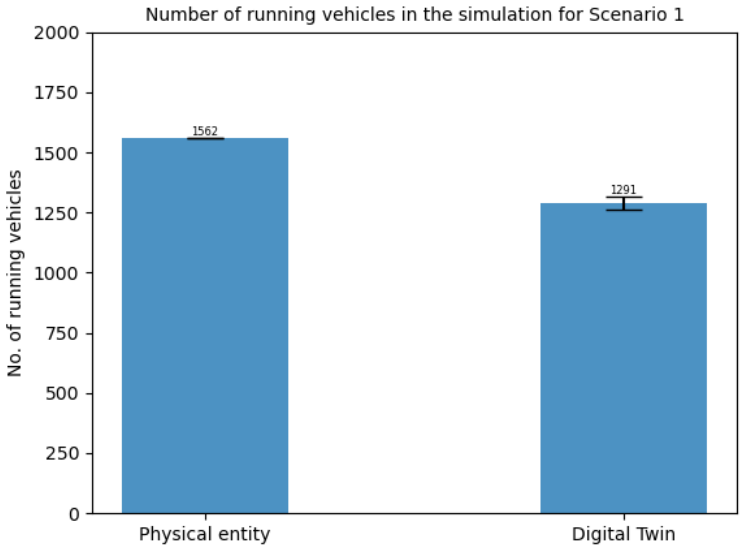

Figure 5.3: Total number of running vehicles average for scenario 1

The total number of running vehicles in the physical entity is 1562, and the average total of running vehicles in the digital twin of three executions is 1291, according to the bar plots. The digital twin is under counting vehicles by nearly 17%. Several factors can influence the number of vehicles in the digital twin. The framework only supports four types of sensors, which provides limited data for traffic modeling. The framework's models have not yet been developed to count the number of vehicles entering and exiting the motorway. Furthermore, there may be discrepancies in reading sensor data, and it is possible that sensors are not

collecting data for all vehicles, resulting in an under counting of vehicles in digital twins.

Comparing the average trip speed

The bar plots in figure 5.4 below represent the simulation's average trip speed in m/s. The first bar in the first figure represents the physical entity's average speed, while the remaining bars represent the digital twins' average speed when executed three times.

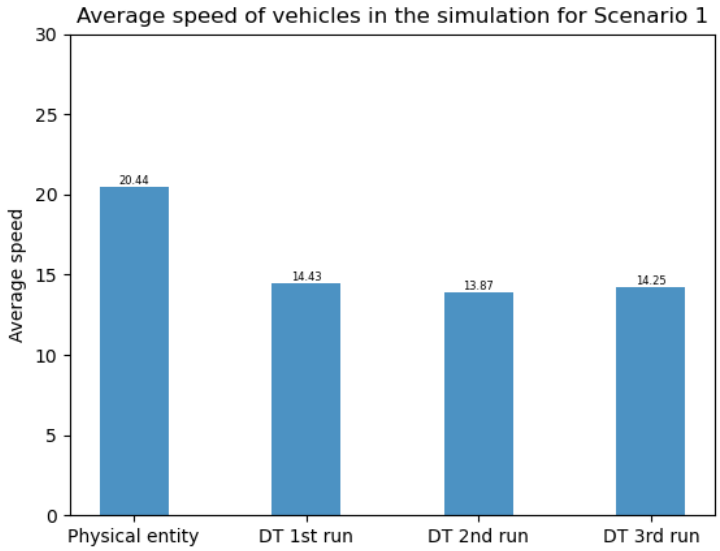

Figure 5.4: Average trip speed of 3 digital twins run for scenario 1

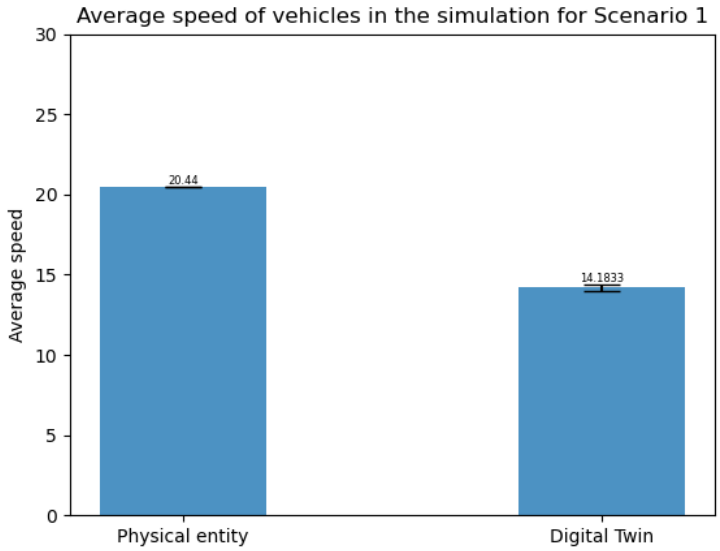

Figure 5.5: Average trip speed for scenario 1

The average trip speed of the digital twins executed three times is averaged out and shown

as a single bar in the second plot 5.5.
The plots show that the digital twin's speed is significantly lower, and the vehicles modeled in the digital twins travel at a slower speed. The physical entity's average speed is 20.44, while the digital twin's is 14.1833. There is nearly a 30% difference between the two. Some of the reasons for this could be that the data is not capturing accurate speed information and that the speed data from the sensors is inaccurate.

Comparing the speed factor distribution

The plots below represent the speed factor distribution in relation to the number of vehicles in the simulation. The speed factor is the ratio of the vehicle's speed to the maximum speed allowed. This data is obtained from the trip info file created during the simulation. It contains information about each vehicle.

The speed factor for three runs of the physical entity simulation is shown in the figure 5.6. Because this is an SUMO simulation of the M50 motorway for a 24 hour traffic flow, the data used for all runs is the same and hence the distribution is also identical for all three runs.

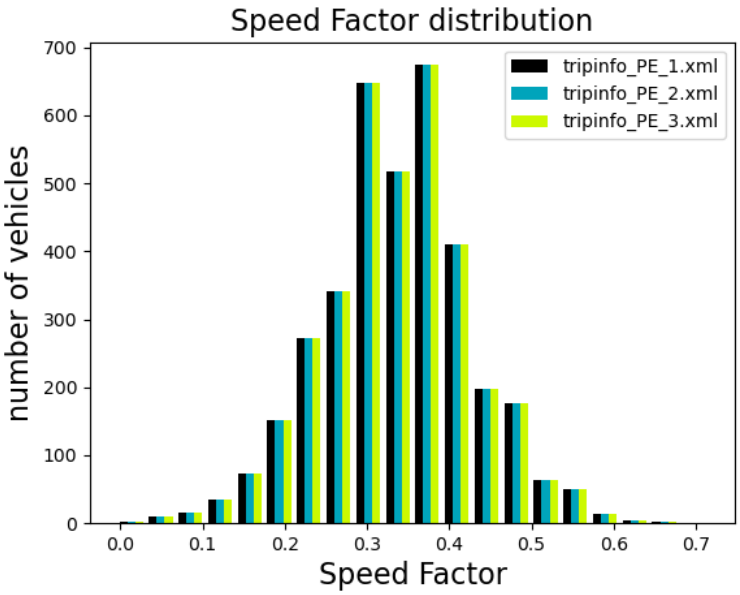

Figure 5.6: Speed factor distribution of physical entity for scenario 1

Figure 5.7 shows the speed factor distribution of the vehicles in the digital twin over three runs.
We can see from comparing the speed distribution bar plots of the physical entity and its digital twin that it follows a normal distribution with data that is symmetrically distributed. Both the physical entity and digital twin bar plots exhibit this pattern. We can see that the mean for both of them is around 0.3 to 0.4, and the speed factor of most vehicles is around

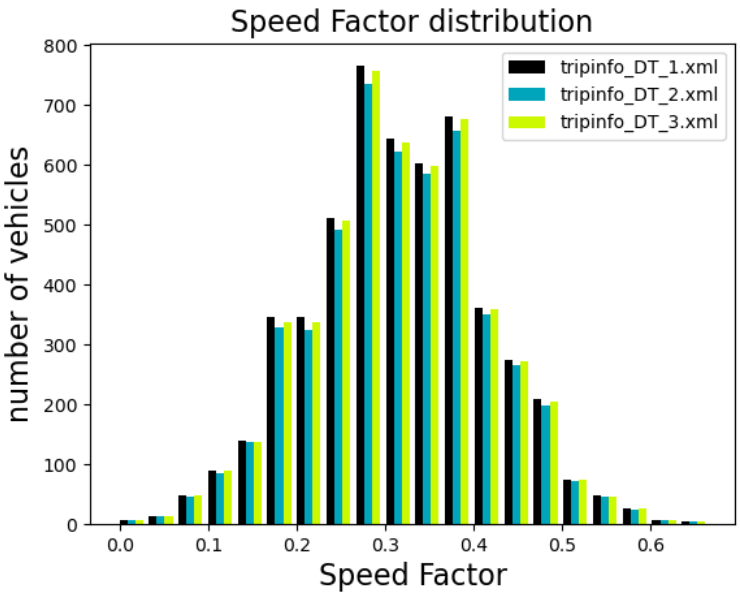

Figure 5.7: Speed factor distribution of digital twin for scenario 1

this value.

After comparing all three measures of the physical entity and the digital twin, we discovered that the digital twin models traffic relatively slower and has a lower number of running vehicles. Some potential reasons for this include sensor data inaccuracy, a lack of support for counting the number of vehicles entering and exiting, and a lack of sensors of sufficient variety to provide data. All of these are potential future framework works. In addition, the framework itself needs to be improved in order to model traffic flow.

5.2.2 Scenario 2

The second scenario for evaluation is evening traffic, when the roads are not that congested. From 19:00 pm, the digital twin is created for a 15-minute simulation. The digital twin is run three times to ensure that the data is not biased, and the average data is used for comparison with the physical entity. The plots generated for this scenario are for the same metrics as in scenario 1 and depict the same data as in scenario 1.

Comparing the total number of running vehicles

Scenario 2 runs the digital twin for the saturated traffic flow, and thus the total number of running vehicles in the physical entity is 1190, and the average total of running vehicles in the digital twin of three executions is around 813, as shown in the figure 5.8 and 5.9. As in scenario 1, the digital twin under counts the vehicles in this scenario as well. However, in this scenario from figure 5.8, we can see that the digital execution is inconsistent across the

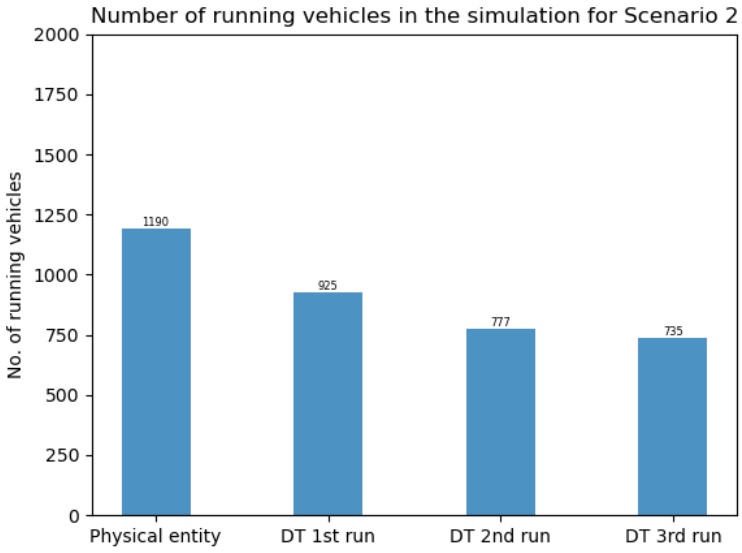

Figure 5.8: Total number of running vehicles for scenario 2

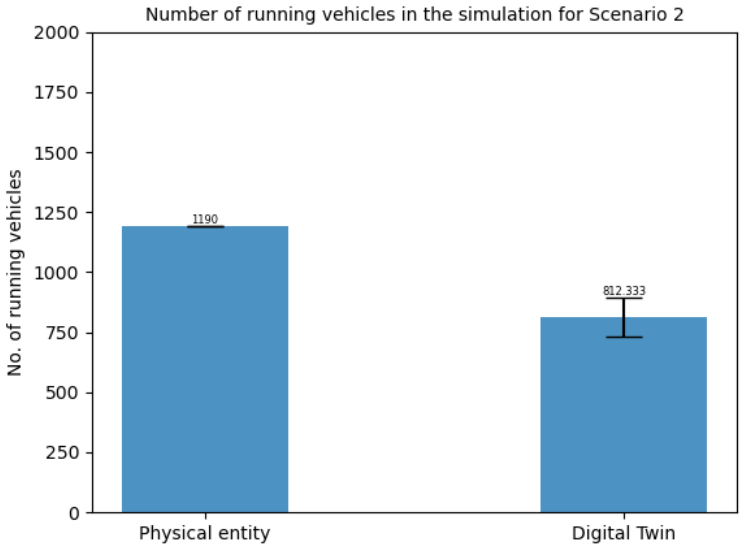

Figure 5.9: Total number of running vehicles average for scenario 2

three runs. The total number of vehicles for the first run is 925, while the remaining two executions have 777 and 735 vehicles, respectively.

Comparing the average trip speed

As in scenario 1, the digital twin models the traffic significantly slower in this scenario as well. The physical entity's average speed is 20.85 m/s, while the digital twin's is 14.24 m/s. The average speed data shows a nearly 31% difference.

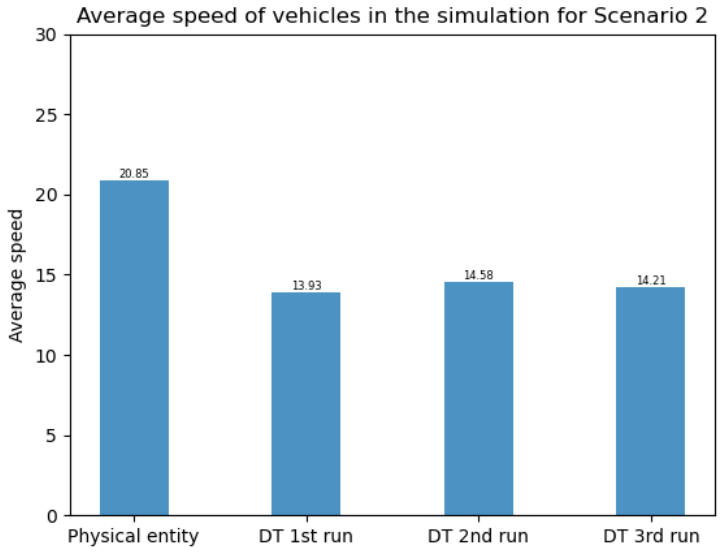

Figure 5.10: Average trip speed of 3 digital twins run for scenario 2

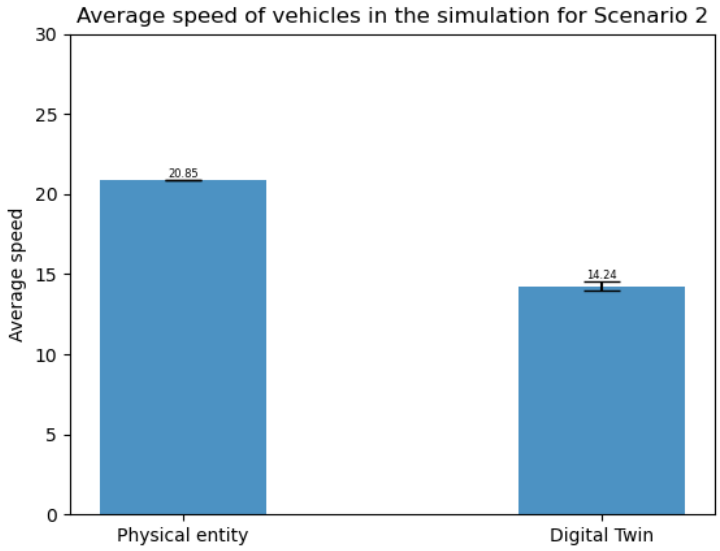

Figure 5.11: Average trip speed for scenario 2

Comparing the speed factor distribution

In this scenario, too, the speed factor distribution follows a normal distribution with the data symmetrically distributed. This pattern can be seen in both the physical entity and digital twin bar plots. We can see that the mean for the physical entity is 0.35 to 0.45 and for the digital twin is around 0.3, which corresponds to the speed factor of most vehicles.

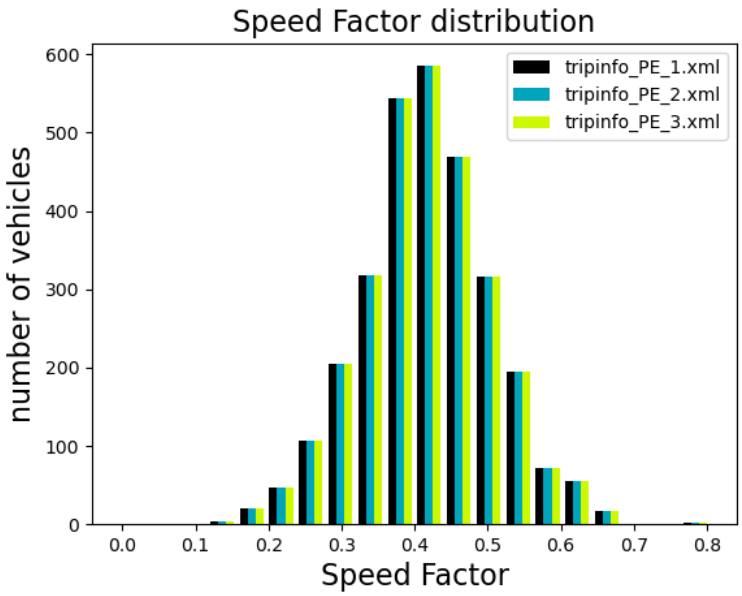

Figure 5.12: Speed factor distribution of physical entity for scenario 2

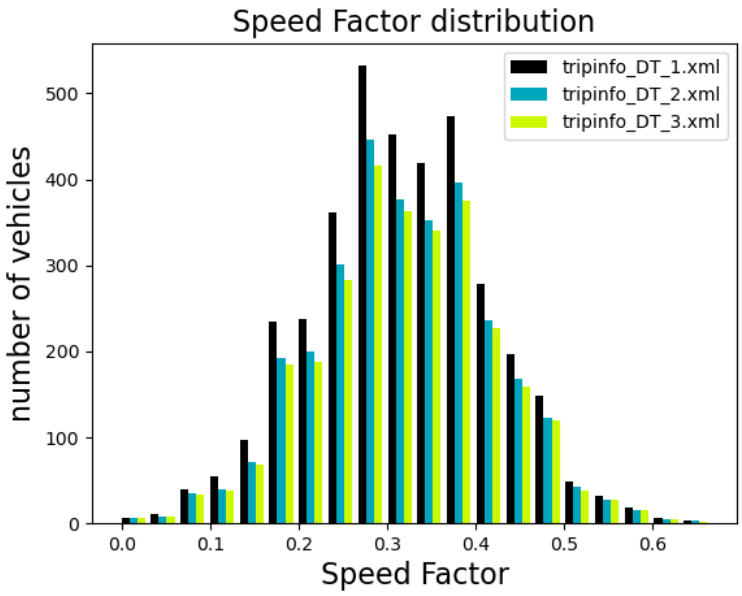

Figure 5.13: Speed factor distribution of digital twin for scenario 2

5.2.3 Scenario 3

The third scenario is evaluated for a free flow traffic at 22:00 pm. The digital twin is executed for a 15 minutes simulation and to ensure the results are not biased, it is executed thrice and the average is used for the comparison. The plots generated for this scenario are for the same metrics as in scenario 1 and depict the same data as in scenario 1.

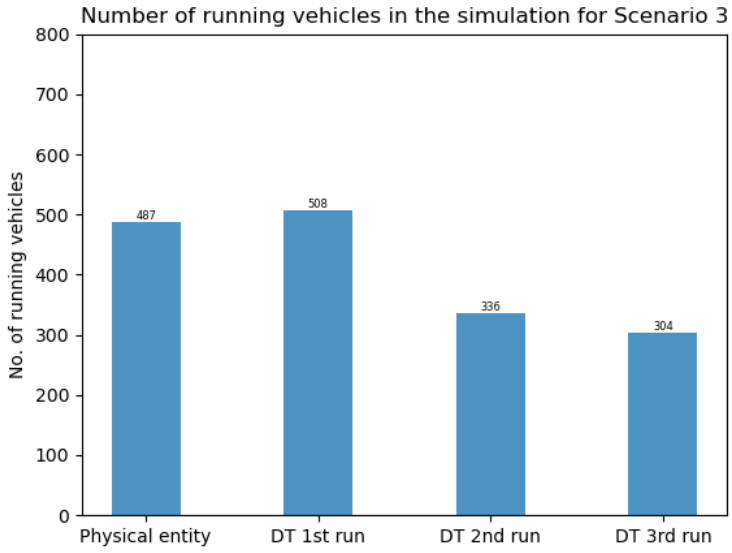

Figure 5.14: Total number of running vehicles for scenario 3

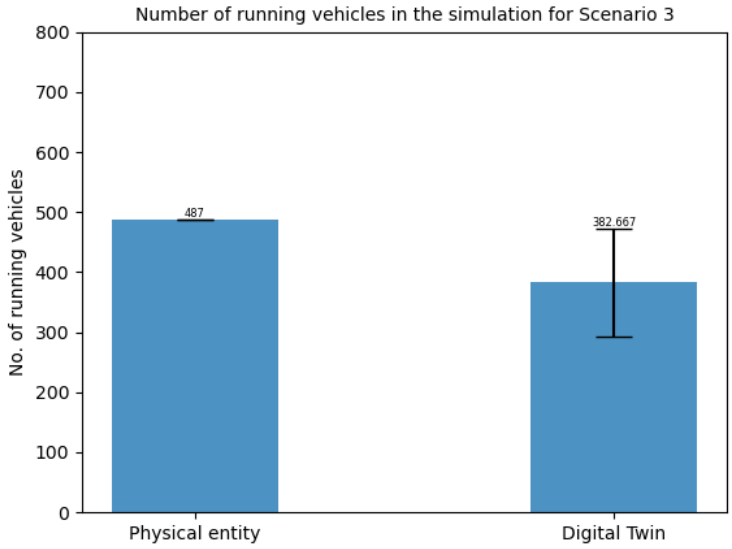

Figure 5.15: Total number of running vehicles average for scenario 3

Comparing the total number of running vehicles

Scenario 3 runs the digital twin for the free flow traffic. The total number of running vehicles in the physical entity is 487, and the average total of running vehicles in the digital twin of three executions is around 383, as shown in plots 1 and 2. In this scenario, as in scenario 1, the digital twin under counts the vehicles. However, as illustrated in Figure 1, the digital execution is inconsistent across the three runs. In fact, in this scenario, the digital twin behaves strangely, and the number of running vehicles is greater than the physical entity for the first execution. The remaining two executions have 336 and 304 vehicles,

respectively, demonstrating a significant difference between the first execution and the remaining two executions.

Comparing the average trip speed

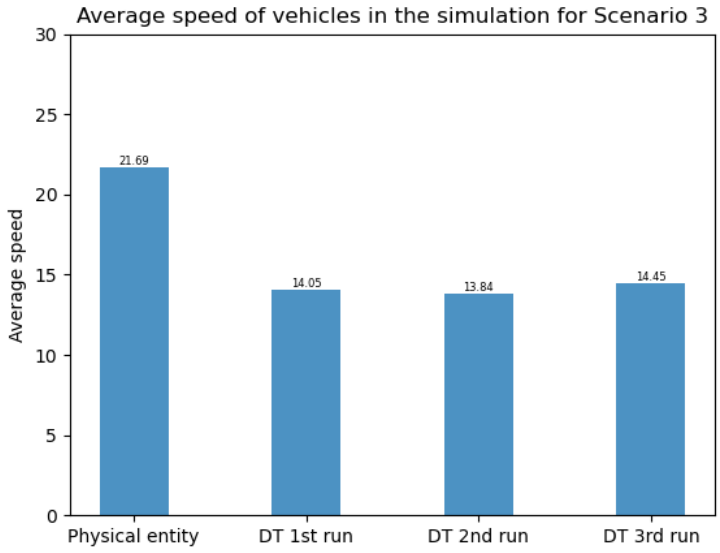

Figure 5.16: Average trip speed of 3 digital twins run for scenario 3

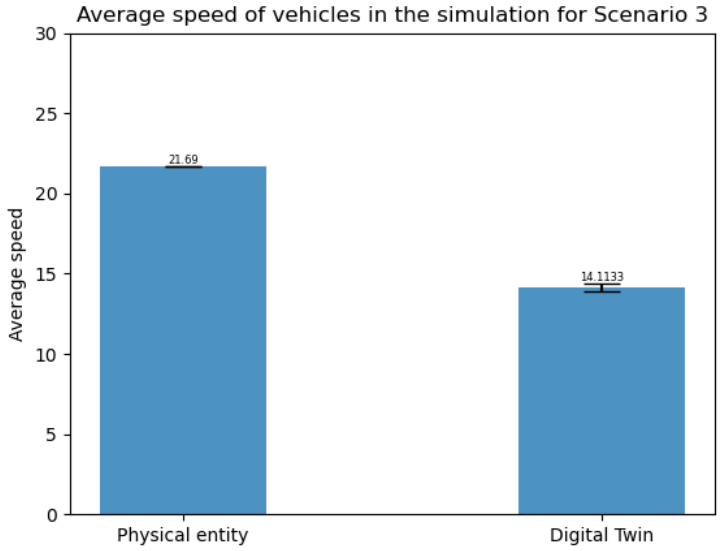

Figure 5.17: Average trip speed for scenario 3

In this scenario as well, the digital twin models traffic significantly slower. The average speed of the physical entity is 21.69 m/s, while the digital twin's is 14.1133 m/s. The difference in average speed is nearly 34%.

Comparing the speed factor distribution

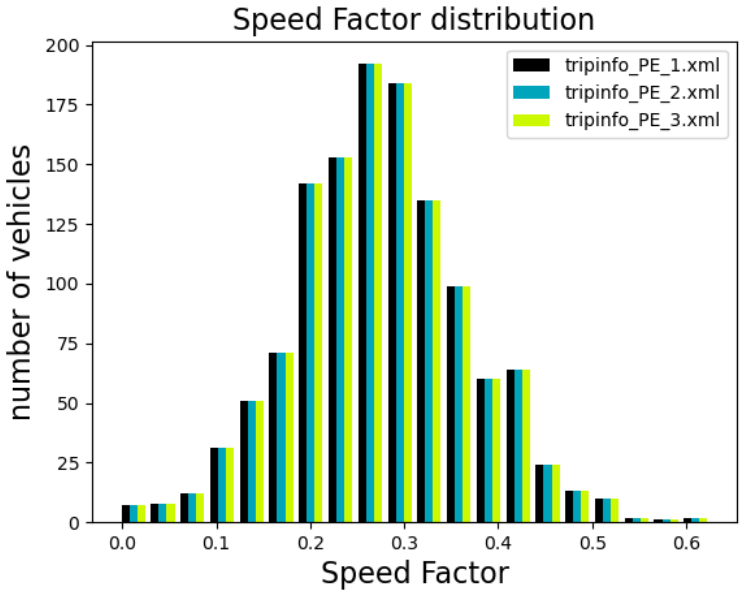

Figure 5.18: Speed factor distribution of physical entity for scenario 3

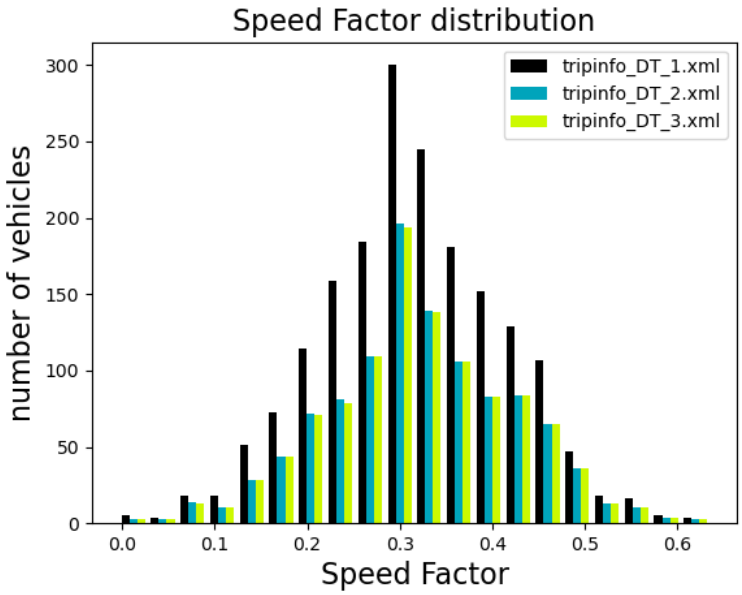

Figure 5.19: Speed factor distribution of digital twin for scenario 3

We can see from comparing the speed distribution bar plots of the physical entity and its digital twin that it follows a normal distribution with data that is symmetrically distributed. Both the physical entity and digital twin bar plots exhibit this pattern. We can see that the mean for the physical entity is around 0.25 to 3 and for the digital twin is around 0.3 to 0.35, and the speed factor of most vehicles is around this value.

5.2.4 Discussion

For evaluating the framework, the digital twin was executed for 15 mins for different timestamp which included a free flow, saturated and congested traffic flow. The digital twin was also run three times in order to average the results and compare them to the SUMO simulation of the motorway.

It is difficult to draw conclusions about the performance of the digital twin based on the results of the scenarios. The metrics used to assess the digital twin are also limited. We discovered that the speed factor distribution shows correlation in both the physical entity and the digital twin based on the plots. However, it has been discovered that the digital twin consistently under counts vehicles and models traffic flow at a slower rate than the physical entity. One of the reasons for this could be inaccurate sensor readings. Furthermore, the sensors may not be able to capture all of the vehicle's data. Many things are still missing from the framework, such as the model for counting the number of vehicles entering and exiting the motorway. Sensor fusion methods could be improved as well. All of these observations can be resolved by extending the framework, as mentioned in the section on future work.

6 Conclusion and Future works

This section gives the conclusion for the proposed framework and outlines the limitations of the framework. Furthermore, this section describes some of the potential future works that can be done to improve the framework.

6.1 Conclusion

The dissertation proposes a framework for developing digital twins of motorways, with a design that includes all of the components of a digital twin. The digital twin model is divided into different components based on the various necessary requirements by the digital twin. The significance of each component is explained, along with the approach used to implement each component.

The initial chapters of the dissertation discusses the current state of the art as well as the tools needed for the framework. The framework can collect sensor data by using an Apache Kafka-based communication mechanism. The framework was designed to support data from four types of sensors: inductive loops, motorway cameras, toll bridge cameras, and probe vehicles. The collected data is sent to the models for data processing and agent mapping, and the traffic flow is modeled using SUMO and TraCI.

To evaluate the framework, a digital twin of Dublin's M50 motorway was created using data from all four types of sensors. The physical entity used in this case was an SUMO simulation of the M50 motorway. From the evaluation section we concluded that the framework was able to successful model the traffic flow. The number of vehicles, average speed, and speed factor distribution are used to evaluate the digital twin.

6.2 Limitations

The designed framework was able to create a digital twin and model the traffic flow for the M50 motorway, but it does have some limitations. The framework's communication

mechanism reads data in batches sequentially and processes it one at a time. When compared to the physical entity, this causes a delay in traffic flow in the digital twin. The evaluation metrics used are limited and cannot accurately evaluate the digital twin. A model for predicting the number of vehicles entering and exiting the motorway is also lacking from the framework. Keeping all of these limitations in mind, the section on future work discusses methods for overcoming them.

6.3 Future work

The designed framework was capable of creating a digital twin of a motorway by collecting data from sensors and modelling traffic flow accordingly. To evaluate the framework, we built a digital twin of the M50 highway and compared its various characteristics. There are many things that can be done to improve the framework and achieve better results in future research. This section will go over some of the potential future initiatives for this framework.

The developed framework currently supports four types of sensor data: inductive loops, motorway cameras, toll bridge cameras, and probe vehicles. The framework can be expanded to support additional sensors that can provide data in addition to the current sensor data. For example, radar data can be collected for object detection. Data from the drivers mobile phone sensors could be collected as well, which could assist in determining the exact location of the vehicles. Out of the four supported sensors, the probe vehicle can provide this data, but for the rest of the vehicles, mobile phone GPS may be useful. Additional sensor fusion models that combine data from multiple sensors and perform error correction on the data could be added in future research. For example, the Kalman filter could be used to correct GPS errors. Methods for correcting and validating other sensor data could also be discovered.

The models could be expanded, and a new model for predicting the number of vehicles entering and exiting the motorway through the intersections could be introduced. Along with this, further experimentation can be done with additional evaluation metrics for measuring the performance of the framework.

Bibliography

Abidin, A. F., Kolberg, M. & Hussain, A. [2015], 'Integrating sumo and kalman filter models towards a social network based approach of public transport arrival time prediction', *International Journal of Simulation Systems Science Technology* pp. 5.1–5.9.

Bieker-Walz, L., Behrisch, M., Junghans, M. & Gimm, K. [n.d.], 'Evaluation of car-following-models at controlled intersections'.

Bonabeau, E. [2002], 'Agent-based modeling: Methods and techniques for simulating human systems', *Proceedings of the National Academy of Sciences* pp. 7280–7287.

Castanedo, F. [2013], 'A review of data fusion techniques', *The Scientific World Journal* .

Dasgupta, S., Rahman, M., Lidbe, A. D., Lu, W. & Jones, S. [2021], 'A transportation digital-twin approach for adaptive traffic control systems', *Transportation Research Board 100th Annual Meeting and publication in Transportation Research Record* .

Dong, H. & Evans, D. [2007], 'Data-fusion techniques and its application', *Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2007)* pp. 442–445.

Durrant-Whyte, H. F. [1988], 'Sensor models and multisensor integration', *International Journal of Robotics Research* .

Erdmann, J. [2014], 'Sumo's lane-changing model', *LECTURE NOTES IN CONTROL AND INFORMATION SCIENCES* pp. 105–123.

Fennell, C. [2022], 'A communication architecture for transportation digital twins using apache kafka'.

Gao, Y., Qian, S., Li, Z., Wang, P., Wang, F. & He, Q. [2021], 'Digital twin and its application in transportation infrastructure', *2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence (DTPI)* pp. 298–301.

Gueriau, M. & Dusparic, I. [2020], 'Quantifying the impact of connected and autonomous vehicles on traffic efficiency and safety in mixed traffic'.

Houbraken, M., Audenaert, P., Colle, D., Pickavet, M., Scheerlinck, K., Yperman, I. & Logghe, S. [2015], 'Real-time traffic monitoring by fusing floating car data with stationary detector data', *2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)* pp. 127–131.

Importance of Transportation in Different Aspects of Life [n.d.].
URL: *https://impoff.com/importance-of-transportation/*

Inductive loops [n.d.].
URL: *https://diamondtraffic.com/technicaldescription/124*

Introduction to TraCI [n.d.].
URL: *https://sumo.dlr.de/docs/TraCI.html*

Kafka documentation [n.d.].
URL: *https://kafka.apache.org/documentation/gettingStarted*

Ou, Q. [2011], 'Fusing heterogeneous traffic data: Parsimonious approaches using data-data consistency'.

Our National Road Network [n.d.].
URL: *https://www.tii.ie/roads-tolling/our-road-network/*

Saifutdinov, F., Jackson, I., Tolujevs, J. & Zmanovska, T. [2020], 'Digital twin as a decision support tool for airport traffic control', *2020 61st International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS)* pp. 1–5.

SONGa, J., WUb, Y., XUc, Z. & LINd, X. [2014], 'Research on car-following model based on sumo'.

SUMO User Documentation [n.d.].
URL: *https://sumo.dlr.de/docs/index.html*

Transport Trends 2021 [n.d.].
URL: *https://www.gov.ie/en/publication/182bd-transport-trends-2021/*

Varshney, P. K. [1997], 'Multisensor data fusion', *Multisensor data fusion. Electronics and Communications Engineering Journal* .

Wang, Z., Han, K. & Tiwari, P. [2021], 'Digital twin simulation of connected and automated vehicles with the unity game engine', *2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence (DTPI)* pp. 1–4.

What Is Sensor Fusion? [n.d.].
URL: *https://www.aptiv.com/en/insights/article/what-is-sensor-fusion*