Foundations for Self-Adaptive Abstraction and Approximation in Digital Twins with Real-time Requirements

Raheleh Biglari

Blissful connection, miniature by Iranian painter, Prof. Mahmoud. Farshchian

Supervisor **Prof. dr. Joachim Denil**

Thesis submitted for the degree of Doctor of Applied Engineering
Faculty of Applied Engineering | Antwerp, 2025

University of Antwerp

Faculty of Applied Engineering

Foundations for Self-Adaptive Abstraction and Approximation in Digital Twins with Real-time Requirements

Thesis submitted in fulfilment of the requirements for the degree of
doctor in applied engineering
at the University of Antwerp

Raheleh Biglari

De doctoraatsonderzoeker en promotor(en) verklaren dat het doctoraatsonderzoek werd uitgevoerd volgens de principes van de wetenschappelijke integriteit, zoals vermeld in het algemeen doctoraatsreglement en algemeen charter van de doctorandus van UAntwerpen en het integriteitscharter voor doctorandi en promotoren verbonden aan de Universiteit Antwerpen.

Antwerp, 2025

Supervisor
prof. dr. Joachim Denil

Jury

Chairman
prof. dr. Paul De Meulenaere, University of Antwerp, Belgium

Supervisor
prof. dr. Joachim Denil, University of Antwerp, Belgium

Members
prof. dr. Claudio Gomes, Aarhus University, Denmark
prof. dr. Dominique Blouin, Telecom Paris, France
prof. dr. Hans Vangheluwe, University of Antwerp, Belgium
em. prof. dr. ir. Martin Timmerman, Dedicated Systems Consult BV, Belgium

Contact
Raheleh Biglari

University of Antwerp
Faculty of Applied Engineering
Department of Electronics and Information and Communication Technology
Co-design of Cyber-Physical Systems (Cosys-Lab)
Groenenborgerlaan 171, 2020 Antwerpen, België
M: raheleh.biglari@uantwerp.be
T: +32 494 85 22 50

To my family

whose love and support

made this journey possible

Acknowledgment

I would like to take the opportunity to thank all those who supported me during this PhD journey.

First & foremost, my great great appreciation goes to Joachim, not only for being my supervisor, but also for being so much more than a promoter, mentor, or manager. Your insight has been invaluable in shaping my journey, and I have learned far more from you than just science. After each and every meeting, even those that lasted more than two hours, I always walked away feeling hopeful, fulfilled, and eager to keep moving forward. Your encouragement and infectious positive vibe made all the difference, and I consider myself very lucky to have had such an inspiring and supportive supervisor. I deeply understand that this does not happen to all PhD students!

You gave me many valuable opportunities for which I am incredibly grateful, from research visits and teaching experiences to guiding me in peer review, encouraging conference participation, and teaching me how to collaborate and build a professional network within our scientific community. You taught me not only how to become a doctor, but also how to be kind, respectful, professional, and above all, a good human being. And of course, to never forget to ***have fun****! That last lesson stuck with me from our very first online meeting during COVID (I even wrote it down right away:D).*

Each scientific work needs to be reviewed and accepted by the scientific community. I appreciate the jury members' time and effort in reviewing this thesis. I would like to thank my PhD committee and Jury, thank you Prof. Dr. Claudio Gomes, Prof. Dr. Dominique Blouin, Prof. Dr. Hans Vangheluwe, Em. Prof. Dr. Ir. Martin Timmerman, and Prof. Dr. Paul De Meulenaere. Your feedback helped me improve the manuscript to its current state.

I would also like to thank my (former) colleagues at CoSys-Lab, AnSyMo, IDLab, and M4S. Mehrdad, thank you for showing me the ropes when I first arrived at the university and for helping me settle into life in Antwerp. You taught me the importance of supporting newcomers—something I've happily paid forward. Joost, our collaboration on the Digital Twin and other courses, your support, and our great discussions have made this journey all the more enjoyable and memorable. A big thank you to Stijn and Jan for sharing the office with me and for watering the plants when I was away! And to Arkadiusz, Bert, Jan, Ken, Milan, Sahar, and Yon—I genuinely enjoyed every conversation we had, whether it was about work or completely off-topic.

I'm also grateful for the network I built within the Modelling and Simulation community & look forward to staying in touch, continuing collaborations, and contributing to our field's growth.. Thank you, Claudio, for all the insightful discussions and meetings—whether in Denmark, Belgium, or online. And thank you to you and Caroline for the warm welcome and delicious Portuguese dinner! Mamadou, thank you for being such an incredible host during our research visit to the IMS Lab in Bordeaux. The intense and inspiring discussions with you, Zeeshan, and Milad marked the beginning of my journey into the Digital Twin domain. Judith, I really appreciate our discussions and collaboration. And Hans, my academic grandfather (Joachim's

academic father, transitive relationship ;)); thank you not only for being part of my PhD jury, but also for all the meetings, workshops, and materials you shared, which played a big role in my learning and building my scientific network. This is amazing, please keep organising these things!

Thank you, Mom and Dad, for your endless love and support, even from 5,000 kilometers away. Despite the distance, I always felt your presence, support, and encouragement. Thank you, Nazanin, for all your support and encouragement, listening to all my never-ending explanations of the things, even when they didn't always make sense! Thank you all for supporting me. Without your encouragement, support, and love, this journey was completely impossible.

Thank you, Faezeh, for always inspiring me with your encouragement; thank you, Nastaran, Roshank, and Maryam, for being there for me through thick and thin; and thank you, Ali, Iman, Naser, Rasa, Reza, Roza, Sevilay, Vida, and Azadeh (in loving memory). The good times we have shared have been wonderful.

A very special thank you to Hamid for your endless support throughout this journey. Thank you for encouraging me when I doubted myself, for pushing me to move forward when things got tough, and for always reminding me of what I'm capable of.

To everyone who has been part of my journey—even if your name is not mentioned here—please know that you are not forgotten. Your support, guidance, and kindness have made a lasting impact. I'm deeply grateful to each and every one of you.

Raheleh

05 June 2025 - Antwerp - Belgium

Abstract

Digital Twins (DTs) are complex systems that increasingly support critical applications across various domains, including agriculture, transportation, home automation, automotive, aerospace, and healthcare. Essentially, a DT is a virtual representation of a physical or digital object, process, or system, serving as a bridge between the physical and digital worlds. They function by enabling real-time monitoring, simulation, and optimisation, offering significant advantages: 1) Simulation and Testing: DTs facilitate real-time monitoring and simulation of their physical or digital counterparts, allowing engineers and developers to test scenarios, predict outcomes, and optimise performance without risking the actual system. 2) Data Integration: By incorporating data from sensors and IoT devices, DTs provide a complete view of system performance in real-time, enabling early issue detection and informed decision-making. 3) Predictive Maintenance: In industries such as manufacturing, aerospace, and healthcare, DTs anticipate potential failures before they occur, enabling proactive maintenance, reducing downtime, and minimising costs. 4) Enhanced Decision-Making: DTs offer stakeholders a clear, data-driven understanding of operations, leading to improved planning, design, and management across multiple sectors.

Embedded systems, particularly Cyber-Physical Systems (CPS), are characterised by the tight integration of computational (cyber) and physical components. In this thesis, these CPS are referred to as the actual systems. Digital twin revolutionise our interactions and management of the actual system, fostering innovation and efficiency. The design and testing of DT systems need simulation models, which provide engineers with the means to analyse system behavior and ensure compliance with performance requirements. However, the complexity of the actual system and DTs translates into computationally expensive models, posing significant challenges when real-time constraints necessitate in time decision-making with strict requirements for timeliness, and accuracy.

To address these challenges, this thesis introduces the SACube framework, a novel approach for dynamically switching between high-validity model-the model with broader validity frame- and less-detailed models–models with narrower validity frame to achieve computational efficiency without compromising accuracy. This framework tackles critical questions of how, when, and where these switching happens. Moreover, we emphasise the importance of the model validity region, which is as essential as the model itself.

Sometimes the model validity is available upfront and already given, but sometimes not. To address this, we introduce the Decision-Centric Technique to determine a model's validity within the context of a decision-maker and store this information in our framework. We further demonstrate the application of this framework in a DT system with real-time requirements, providing detailed workflows.

The proposed methods in this thesis are assessed using two case studies: an automated highway lane change maneuver system and a DT of a traffic system. These use cases

illustrate the actual application of adaptive approaches to fulfil real-time requirements while preserving the accuracy and usefulness of predictive models.

This thesis presents a structured approach for optimising multi-modal system simulations, particularly in computationally intensive domains like urban traffic networks. By defining a validity frame for surrogate models, it ensures seamless mode switching while maintaining accuracy and minimising redundancy. The proposed ensemble of adaptive surrogate models enhances performance modeling, enabling efficient analysis of dynamic, multi-modal systems.

This thesis lays the foundation for scalable and robust modelling of complex systems through the integration of adaptive abstraction and approximation, machine learning, and surrogate modelling techniques. The suggested methodologies are substantiated by thorough experimentation, demonstrating their applicability across various domains and enhancing the cutting-edge in modelling and simulation for CPS and DTs.

Nederlandstalige Samenvatting

Digital Twins (DT's) zijn complexe systemen die in toenemende mate kritieke toepassingen ondersteunen in verschillende domeinen, waaronder landbouw, transport, domotica, auto's, lucht- en ruimtevaart en gezondheidszorg. In wezen is een DT een virtuele representatie van een fysiek of digitaal object, proces of systeem, die dient als brug tussen de fysieke en digitale wereld. Ze maken real-time monitoring, simulatie en optimalisatie mogelijk en bieden aanzienlijke voordelen: 1) Simulatie en testen: DT's maken realtime monitoring en simulatie van hun fysieke of digitale tegenhangers mogelijk, zodat ingenieurs en ontwikkelaars scenario's kunnen testen, uitkomsten kunnen voorspellen en prestaties kunnen optimaliseren zonder risico's te nemen met het eigenlijke systeem. 2) Gegevensintegratie: Door gegevens van sensoren en IoT-apparaten te integreren, bieden DT's een compleet beeld van de systeemprestaties in realtime, waardoor problemen vroegtijdig kunnen worden opgespoord en geïnformeerde besluitvorming mogelijk is. 3) Voorspellend onderhoud: In industrieën zoals productie, lucht- en ruimtevaart en gezondheidszorg anticiperen DT's op potentiële storingen voordat ze optreden, waardoor proactief onderhoud mogelijk wordt, de uitvaltijd wordt beperkt en de kosten worden geminimaliseerd. 4) Verbeterde besluitvorming: DT's bieden belanghebbenden een duidelijk, datagestuurd inzicht in de activiteiten, wat leidt tot betere planning, ontwerp en beheer in meerdere sectoren.

Ingebedde systemen, met name Cyber-Physical Systems (CPS), worden gekenmerkt door de nauwe integratie van computationele (cyber) en fysieke componenten. In dit proefschrift worden deze CPS de eigenlijke systemen genoemd. De digitale tweeling zorgt voor een revolutie in onze interacties en het beheer van het eigenlijke systeem, en bevordert innovatie en efficiëntie. Voor het ontwerpen en testen van DT-systemen zijn simulatiemodellen nodig, die ingenieurs de middelen geven om het gedrag van het systeem te analyseren en te zorgen dat het voldoet aan de prestatie-eisen. De complexiteit van het daadwerkelijke systeem en DT's vertaalt zich echter in rekenintensieve modellen, wat aanzienlijke uitdagingen met zich meebrengt wanneer real-time beperkingen tijdige besluitvorming met strikte eisen voor tijdigheid en nauwkeurigheid noodzakelijk maken.

Om deze uitdagingen aan te pakken, introduceert dit proefschrift het SACube raamwerk, een nieuwe benadering voor het dynamisch schakelen tussen modellen met een hoge validiteit - het model met een breder geldigheidsframe - en minder gedetailleerde modellen - modellen met een smaller geldigheidsframe - om rekenefficiëntie te bereiken zonder de nauwkeurigheid in gevaar te brengen. Dit raamwerk pakt kritische vragen aan over hoe, wanneer en waar deze omschakelingen plaatsvinden. Bovendien benadrukken we het belang van het geldigheidsgebied van het model, dat net zo essentieel is als het model zelf.

Soms is de geldigheid van het model vooraf beschikbaar en al gegeven, maar soms ook niet. Om dit aan te pakken introduceren we de Decision-Centric Technique om

de geldigheid van een model te bepalen binnen de context van een beslisser en deze informatie op te slaan in ons raamwerk. Verder demonstreren we de toepassing van dit raamwerk in een DT-systeem met realtime eisen, waarbij we gedetailleerde workflows geven.

De voorgestelde methoden in dit proefschrift worden beoordeeld aan de hand van twee casestudies: een geautomatiseerd systeem voor het wisselen van rijstroken op snelwegen en een DT van een verkeerssysteem. Deze use cases illustreren de daadwerkelijke toepassing van adaptieve benaderingen om aan real-time eisen te voldoen met behoud van de nauwkeurigheid en bruikbaarheid van voorspellende modellen.

Voor het simuleren van complexe systemen met verschillende modi en scenario's zijn rekenintensieve modellen nodig. Multimodale systemen, zoals stedelijke verkeersnetwerken, vereisen specifieke modelleringsoverwegingen voor variërende operationele en omgevingsomstandigheden. Surrogaatmodellering biedt een praktische oplossing voor het optimaliseren van de modellering van dergelijke complexe, multimodale systemen. Het schakelen tussen modi vereist echter

Dit proefschrift presenteert een gestructureerde aanpak voor het optimaliseren van multimodale systeemsimulaties, met name in rekenintensieve domeinen zoals stedelijke verkeersnetwerken. Door het definiëren van een geldigheidsframe voor surrogaatmodellen wordt naadloze modiwissel gegarandeerd met behoud van nauwkeurigheid en minimale redundantie. Het voorgestelde ensemble van adaptieve surrogaatmodellen verbetert de prestatiemodellering en maakt efficiënte analyse van dynamische, multimodale systemen mogelijk.

Dit proefschrift legt de basis voor schaalbare en robuuste modellering van complexe systemen door de integratie van adaptieve abstractie en benadering, machinaal leren en surrogaat modelleringstechnieken. De voorgestelde methodologieën worden onderbouwd door grondige experimenten, waarbij hun toepasbaarheid in verschillende domeinen wordt aangetoond en de voorhoede in modellering en simulatie voor CPS en DT's wordt verbeterd.

[Vertaald met DeepL.com (gratis versie)]

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Embedded systems, particularly Cyber-Physical Systems (CPS), are complex systems characterised by the tight integration of cyber and physical parts. CPS applications are used in various domains, including agriculture, transportation, home automation, automotive, aerospace, and healthcare. A notable example is developing an automated lane-change system for highway driving that enables the autonomous vehicle to automatically move from one lane to another.

Several challenges are involved in designing such systems, including modelling the control components and services. Prediction models significantly enhance the optimisation, analysis, and adaptability of complex engineering systems. It allows for better decision-making, increases system performance, and facilitates adaptation in dynamic contexts. Accurate predictions require the utilisation of simulation models to replicate the behavior and interactions of the actual system.

Simulation models are commonly used at design time when engineers use simulation results to develop systems that fulfil the system's requirements. More recently, using predictive models during system operation has become more prevalent. Digital twins (DT) play a vital role in this evolution. While there are several definitions of DTs, many of them vary significantly [8]. Among various definitions of DT, this thesis uses the description provided by Bordeleau et al. [9], which defines a DT as a digital representation of the actual system that is continually updated with real-time data during its lifecycle. This representation interacts with and influences the actual system [10].

Notably, the actual system can be either digital or physical. For example, the DT of a smart city is one of the applications for testing autonomous systems, specifically the decision-making algorithms of Autonomous Vehicles (AVs) [11]. Another example is the DT of a motorway. Compared to simulations that help understand what can happen when there are changes in the motorway system, DT helps understand what is happening during run-time and what can happen by particular changes, such as control strategies in the current traffic situation [12]. Twinning can also be used at the control level of CPS; models are often used at run-time to predict and control, e.g., dead-reckoning and model predictive control.

1.1 Motivation

The availability of real-time quantitative data and advanced analytics through DT considerably improves decision-making by providing deeper insights and allowing for faster responses. However, in certain situations, decision-making processes must adjust precisely to real-time constraints. Moreover, the computational demands of mathematical optimisation tasks and long simulation times make them difficult to execute in real time. Psarommatis [13] illustrates that large numbers of scenarios and simulation runs in DT manufacturing make real-time calculations difficult. To satisfy these requirements, DT models must not only run within the required timeframes but also keep sufficient accuracy to ensure reliable results [14].

There might be this idea of solving this kind of computational problem using supercomputers. While advances in supercomputers, connectivity, and cloud computing are improving system capabilities, they cannot address all challenges, particularly those with hard real-time requirements. The challenges posed by such constraints cannot always be solved by simply using more powerful computers [15]. Amdahl and Gustafson's [16, 17] law states that the speedup cannot be increased to infinity even if the number of processors is increased to infinity. By increasing the number of processors, programs and operating systems get more complex, and there are always some sequential tasks that can not run in parallel.

Effective modelling can therefore be understood as the development of valid simplifications. Simplifying or reducing complexity is essential to ensure that models can be executed efficiently on resource-limited simulators. Surrogate models or dimension reduction techniques play a critical role in this process by significantly decreasing computational effort, thus enabling real-time applicability. However, these simplified models must remain valid within a defined experimental frame to ensure reliability for their intended purpose [18, 19].

Although the most common solution is to use a single model for representing the entire system, which is appropriate for many kinds of studies, it may not be adequate when large, complex scenarios must be investigated. Complex scenarios may be too large or complex to be assessed in full detail. These findings have led to the use of a library of models that is made up of different models with different levels of detail that all reason over the same property(s) of interest.

Furthermore, the utilisation of multiple models is essential in particular fields. Goldenson et al. [20] discuss climate models of the CMIP6 (Coupled Model Intercomparison Project), which include more than 100 simulations from 53 distinct modeling centers. Since computational resources are limited, they can't downscale every GCM. Therefore, they need to prioritise which Global Climate Models (GCMs) are best suited for regional dynamical downscaling—a process that refines large-scale climate projections into higher-resolution regional simulations.

Moreover, in the traffic domain, different traffic phenomena need different traffic models to represent the different physical nature and inherent characteristics. Schiller et al. [21] present a multi-resolution traffic simulation approach that couples two distinct simulation models to enable large-scale yet high-fidelity virtual evaluations of VANET applications and ADAS systems. This study uses two models: a detailed nanoscopic

model (employing VIRES Virtual Test Drive) for high-resolution simulation within an area of interest, centered around the "EGO" vehicle, and a less detailed microscopic model (using SUMO) for simulating the surrounding traffic. The key technique is dynamic spatial partitioning, whereby the simulation area is split into a High Resolution Area and a Low Resolution Area. A hysteresis-based mechanism is applied to avoid frequent oscillations of vehicles near the partition boundary, ensuring smooth transitions between models. Furthermore, an offline preprocessing step harmonises the different road network data formats, and an online master-slave synchronisation ensures that the two simulators operate coherently in real time. This hybrid method effectively balances computational efficiency with the accuracy required for simulating safety-critical vehicular interactions in dynamic environments.

Previous approaches in the literature have relied on ad hoc methods for determining model transitions, which limits their effectiveness and consistency. These shortcomings highlight the need for innovative, systematic approaches that clearly define how, when, and where to switch between models, thereby ensuring that each model is applied within its appropriate context. This central requirement motivates the present thesis, which is dedicated to developing rigorous frameworks and techniques to address these challenges.

This motivation forms the basis for the research contributions and research questions explored in this thesis.

1.2 Research Questions

In this thesis, we address the following research questions:

RQ1) How can we effectively implement and manage the transition to less-detailed simulation models in real-time within the context of adaptive abstraction and approximation? This question addresses the mechanisms for determining when, where, and how to switch between models of varying complexity during system operation.

RQ2) To what extent can we ensure the validity of the information generated through the implementation of frameworks and techniques for adaptive abstraction and approximation? In particular, how can we leverage decision-maker equivalence and domain knowledge to establish model validity regions efficiently?

RQ3) How can the Self-Adaptive Abstraction and Approximation approach contribute to meeting real-time decision-making requirements within the context of the digital twin paradigm? This question explores the potential of adaptive abstraction and approximation frameworks and techniques to meet time-critical decision-making challenges in DT applications.

In addressing these research questions, we assume that the existence and accessibility of a model library are taken for granted. It is important to point out that an abstracted and/or approximated model increases uncertainty, and in this study, we assume that all models are verified as chosen from a library of verified models.

Furthermore, simulating complex systems across diverse modes and scenarios requires computationally expensive models. This exploration leads us to the following research

question:

RQ4) How can surrogate modelling help simulate complex systems across diverse modes and scenarios?

Each and every one of these research questions will be addressed by this thesis.

1.3 Contributions

Several contributions emerged in response to the research questions in this thesis. The following are the main contributions of this thesis:

1. **Decision-Centric Technique:** The significance of the model's validity region map is equally as crucial as the model itself, and we need to know the model's validity. A validity region map is a structured representation that defines the specific conditions or ranges of variables (such as input values, environmental factors, time, etc.) under which a particular model remains valid. The model's validity is mostly available in the form of the validity frame, but sometimes this validity frame is not available upfront. Therefore, we propose a decision-centric technique to find the validity of a model in the context of a decision-maker.

2. **SACube Framework:** the creation of a framework to deal with computationally expensive models by dynamically switching to the most appropriate model with the high priority of less-detailed models within the library of model (family), as well as to decide how, when, and where these models should be used. Furthermore, in this research, we focus on the significance of the model's validity region map, which is equally as crucial as the model itself. A validity region map is a structured representation that defines the specific conditions or ranges of variables (such as input values, environmental factors, time, etc.) under which a particular model remains valid. To allow this, we propose decision-centric technique to find the validity of a model in the context of a decision-maker and store this information for use in our framework. We investigate employing this framework in a digital twin with real-time constraints. We also define the required workflows and discuss this framework with a detailed case study.

3. **Towards a Validity Frame of Multi-modal Surrogate Models:** Multi-modal systems, such as urban traffic networks, require considering specific modelling considerations for varying operational and environmental conditions. Surrogate modelling offers a practical solution to represent these modes through different surrogate models. However, switching between modes needs careful management of the models' experimental frames. In multi-model setups, overlap between these frames introduces unnecessary complexity and ambiguity in model selection. The concept of validity frame of multi-modal surrogate models is the model's experimental frame plus the process allowing for changing this frame. This study introduces an approach to define such a validity frame, ensuring that the union of all experimental frames covers the full system operating domain without redundancy and focusing on clean separation between models and interpretable mode switching. The proposed ensemble of surrogate models, coupled with adaptive techniques,

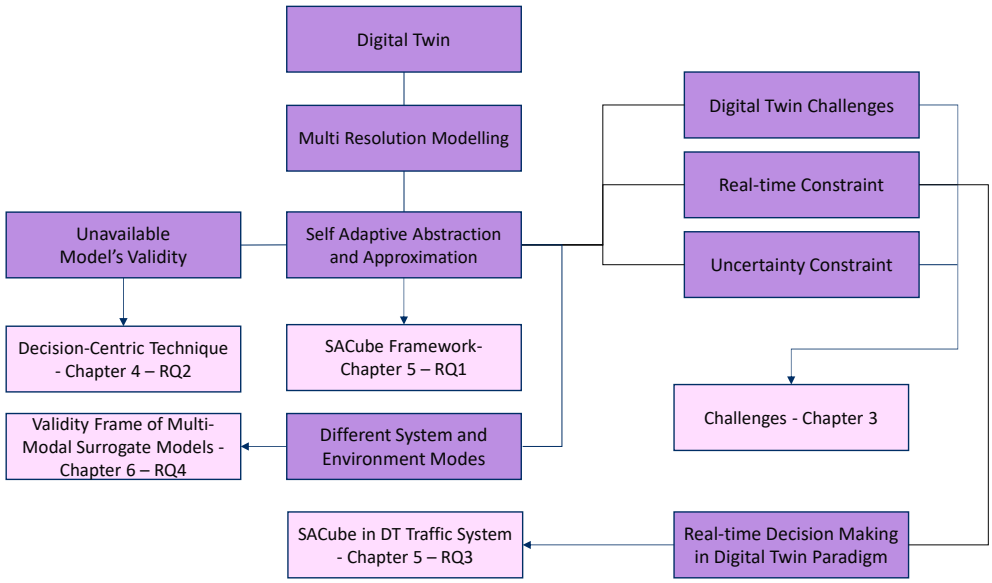

Figure 1.1: Roadmap of this thesis.

represents a significant step forward in the performance modelling of dynamic and computationally expensive models.

1.4 Roadmap of the Thesis

Figure 1.1 shows the roadmap of this thesis, outlining the logical flow of the research and its contributions. Each chapter is derived from one or more of the publications listed in appendix C. This thesis is around complex systems like CPS and DT, and the modelling and simulation of such systems with real-time constraints.

The structure of the thesis is as follows:

Chapter 2 provides the foundation by introducing key concepts, definitions, and state-of-the-art approaches in the domain of CPS, DT, and modelling and simulation. This chapter provides the theoretical background needed to understand the challenges and opportunities explored in the next chapters. The content in this chapter derives from [8, 22, 23].

Chapter 3 identifies and analyses the challenges in modelling and simulation of complex systems specifically DTs, with a particular emphasis on real-time constraints and uncertainty management. This chapter highlights critical gaps in the existing literature and establishes the motivation for this research. The content in this chapter derives from [8].

Chapter 4 tackles the second research question, RQ2, by introducing a decision-centric technique for finding the validity of simulation models. Here, the focus shifts toward ensuring the reliability and validity of models used in decision-making processes. The

content in this chapter derives from [22, 24, 25].

Chapter 5 presents the SACube Framework, which addresses the first research question, RQ1, by proposing an adaptive abstraction and approximation methodology to address the issue of computationally expensive models by dynamically switching to less-detailed models when appropriate. This framework tackles critical questions of how, when, and where these models should be used. The content in this chapter derives from [22]. Moreover, this chapter addresses the third research question, RQ3, by investigating how the SACube is applicable in the Digital Twin of a traffic system.

Chapter 6 answers the fourth research question, RQ4, by proposing a validity frame of multi-modal surrogate models. This chapter emphasises the modelling of multi-modal systems, like urban transportation networks, which requires careful consideration of operational and environmental factors. Surrogate models provide a realistic way to express these modes. Switching between modes requires careful control of the model's experimental frames. Overlapping frames in multi-model setups add complication and ambiguity when selecting models. In multi-modal surrogate models, the validity frame encompasses both the experimental frame and the mechanism for changing it. This work proposes a method for defining a validity frame that covers the entire system operating domain without redundancy, with a focus on clear model separation and interpretability. The suggested ensemble of surrogate models, along with adaptive approaches, significantly improves performance modelling for dynamic and computationally expensive models. The content in this chapter derives from [23].

Finally, the thesis concludes with Chapter 7 where the overall conclusion and potential directions for future work are discussed.

1.5 Case Studies

Throughout this dissertation, we use two main case studies.

1.5.1 Highway Lane Change Maneuver Case Study

The first case study is an automated Highway Lane Change Maneuver (LCM) system implemented using MATLAB/SIMULINK, shown in Figure 1.2 that allows a vehicle to move automatically from one lane to another. The actual system is the lane following controller, while the digital twin functions as a lane change planner that monitors the system and initiates lane changes when necessary. We use a lane change control algorithm to examine the challenges of introducing an adaptive abstraction and approximation framework in a real-time context. While not exclusively focused on this category of control algorithms, the lane change case study provides a clear and intuitive representation of the problem space, as illustrated in Figure 1.3. Such systems model both longitudinal and lateral control dynamics for automated lane changing. Utilising onboard sensors, LCM systems monitor the environment to detect the most important objects (MIOs), find an optimal trajectory that avoids potential collisions, and guide the vehicle along this trajectory. The system operates with real-time constraints, where computational tasks must adhere to predefined time budgets to maintain responsiveness. Specifically, the

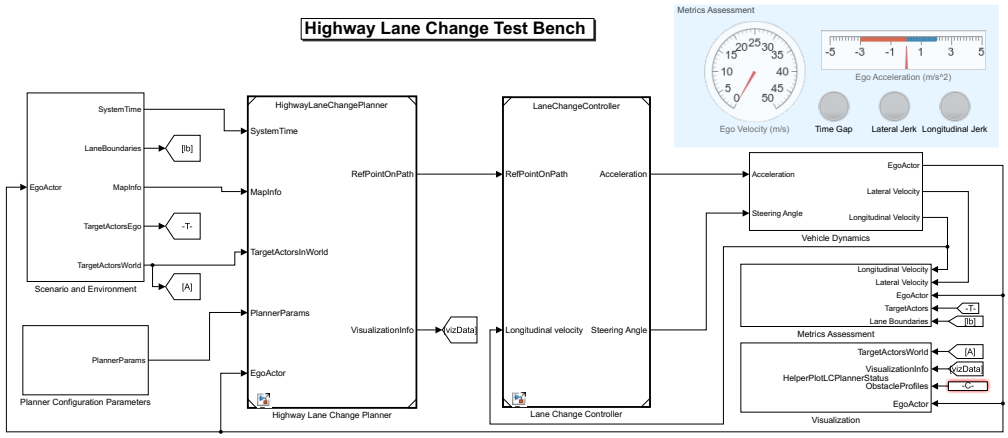

Figure 1.2: Highway lane change test bench.

predictive step of the algorithm is allocated a maximum execution time to ensure timely decision-making. As discussed in detail in Chapter 2, variations in computational load, such as an increase in detected vehicles, can impact prediction performance, requiring adaptive model switching to maintain real-time operation.

We define eight different scenarios for our contributions. Each of these scenarios has been created using the Driving Scenario Designer (Automated Driving Toolbox) and exported to a scenario file. During the simulation, the model logs signals to the base workspace as *logsout*, and we can analyse the simulation results.

In the context of this thesis, we use a Simulink-provided model to simulate the lane-changing behavior. Figure 1.4 shows the high-level architecture of the lane-changing algorithm. The algorithm uses the Frenet Coordinate System, which represents the position of the car on the road more intuitively than traditional (x, y) coordinates.This algorithm also uses a prediction step to predict the trajectories of the different actors. Afterward, path planning takes care of finding a good path in the world. Finally, a model-predictive controller steers the vehicle over the planned path.

In this study, we focus on the prediction step of the other vehicles. The prediction step simulates a trajectory relative to the ego car for each vehicle in the environment. The component receives information on the lateral and longitudinal position, velocity, and acceleration of the vehicles. Several real-time properties are present, which are discussed in Chapter 2.

1.5.2 SUMO Traffic Simulation Case Study

This thesis employs SUMO (Simulation of Urban MObility) [26] as a second case study, a renowned open-source micro-traffic simulator, to analyse the use of machine learning-based surrogate models for traffic management. Through the simulation of traffic dynamics in Wilrijk, a district of Antwerp, Belgium, we generate a robust dataset that serves as the foundation for the training and assessment of these models. Using SUMO helps

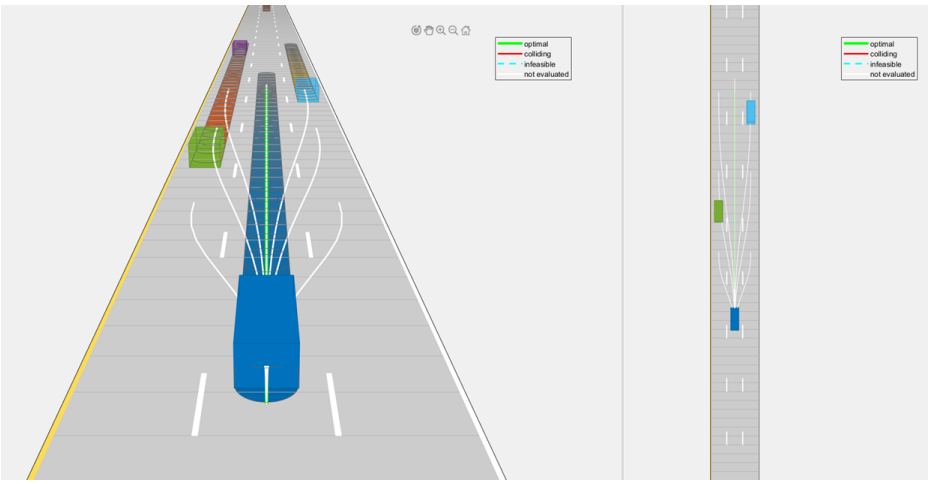

Figure 1.3: Simulation snapshot of highway lane change case study providing visualisation and intuitive reasoning.

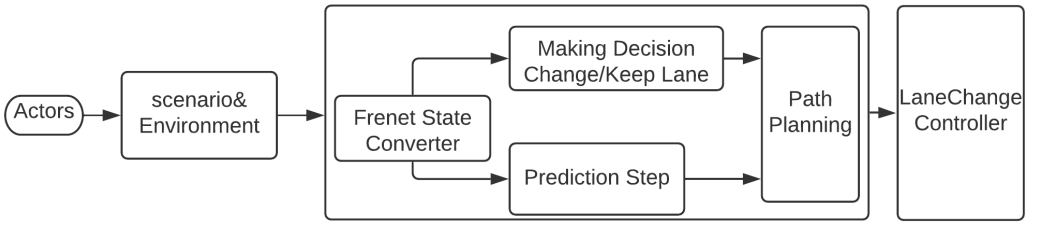

Figure 1.4: Causal block diagram of lane change maneuver case study.

us show our approach to using ML-based surrogate models, facilitating accurate prediction of intended parameters that would often necessitate computationally expensive simulations.

We model the traffic system for our intended map, Wilrijk, shown in Figure 1.5 by using SUMO. Figure 1.6 shows the simulator environment, running the traffic simulation of the Wilrijk map. This simulator is a micro-level traffic simulator, meaning each vehicle and its dynamics are modeled individually. It has several main advantages. First, it allows us to generate a large-scale map of a certain area. By importing a map from OpenStreetMap, we don't have to spend time building a large-scale network from scratch. Next, using the Traffic Control Interface (TraCI), we can connect to an ongoing simulation. This allows us to collect information about the simulation and to update the ongoing simulation. Traffic scenarios are generated using SUMO's randomTrips.py, which generates a set of random trips for a given network.

Figure 1.5: Covered traffic system in SUMO.

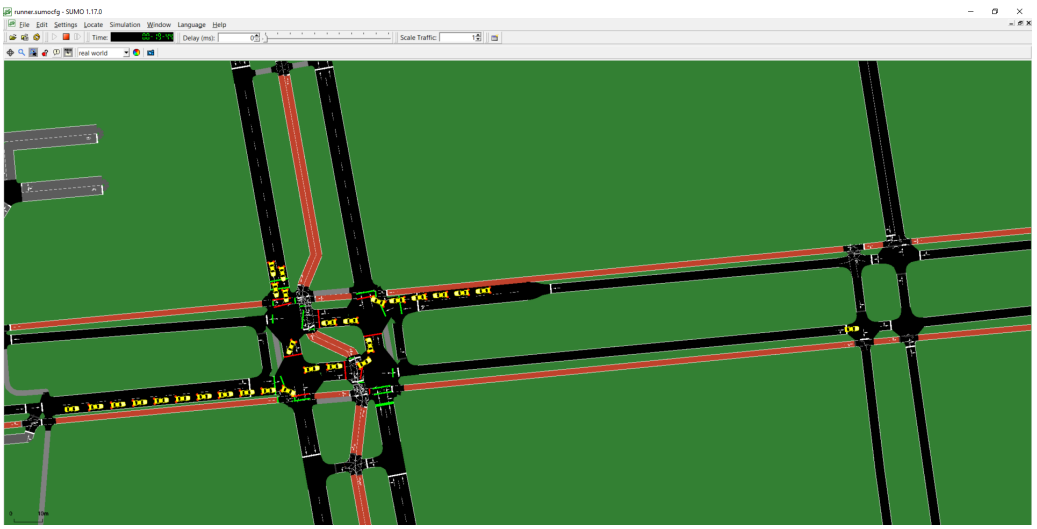

Figure 1.6: Sumo environment.

Chapter 2

Background and Related Work

This chapter provides the necessary background to frame the research presented in this thesis. It brings the foundation for understanding and discussing model abstraction and approximation within the real-time twinning paradigm, which will be addressed in the following chapters.

The discussion begins by understanding the cyber-physical systems, followed by tracing the evolution of Digital Twins from traditional modelling and simulation methodologies, highlighting the increasing complexity of these systems and the associated challenges. A detailed exploration of available techniques for simplifying complex models is presented, with a particular emphasis on self-adaptation and its widely recognised technique, MAPE-K. Furthermore, this chapter addresses the critical constraints imposed by the actual systems, the system under study, being modeled, the requirements for ensuring the validity of simulation models, and the essential principles underlying the composition of simulation models. These elements form the basis for applying self-adaptive abstraction and approximation, which serves as a core focus of this research.

In addition, this chapter includes the relevant literature, positioning our investigation within the wider research context.

This chapter is based on the following publications:

Z. Ali, R. Biglari, J. Denil, J. Mertens, M. Poursoltan, and M. K. Traoré, "From modeling and simulation to digital twin: evolution or revolution?" *SIMULATION*, vol. 100, no. 7, pp. 751–769, 2024.

R. Biglari and J. Denil, "Self-adaptive abstraction and approximation (sacube) framework for digital twins with real-time requirements," *SIMULATION*, 2025, under peer review.

R. Biglari, C. Gomes, and J. Denil, "Towards a validity frame of multi-modal surrogate models for traffic simulation," in *2025 Annual Modeling and Simulation Conference (ANNSIM)*, 2025, in press.

2.1 Understanding Cyber-Physical Systems

Cyber-Physical Systems (CPS) are integrated systems that combine computerised implementations with physical components, creating a seamless interaction between the digital and physical worlds [27]. CPS operate on the principle of mutual influence between their cyber and physical components. The cyber element initiates changes in the physical component, which then generates feedback that alters the state of the cyber system, creating a continuous interaction loop.

CPS have evolved from earlier concepts in fields such as mechatronics, embedded systems, and cybernetics. Literature attributes the coining of the term 'Cyber-Physical System' (CPS) to Hellen Guille in 2006 [27]. These systems are often seen as networks integrating multi-physical components—mechanical, electrical, biochemical, etc.—with computational processes such as control, signal processing, logical inference, and planning. CPS frequently operate in highly uncertain and challenging environments, interacting with human actors and other CPS.

CPS have diverse applications, including energy conservation, environmental control, avionics, critical infrastructure management, medical devices, traffic control, robotics, manufacturing, and smart city development. Their design drives innovation, creating new markets and economic benefits.

However, CPS are complex due to their multidisciplinary nature and inter-domain interactions, especially in safety-critical applications. Failures can lead to economic losses, environmental damage, or harm to humans. Engineering advancements have addressed these challenges by refining design languages, frameworks, and tools, leading to the widespread adoption of model-based design, which enables higher-level abstraction and automated implementation.

A key feature of CPS design is its inherently multidisciplinary approach. The complexity arises both from the application domain, such as medical, biological, or aeronautical industries, and the integration of computerised, electronic, and mechanical components. Effective CPS development requires cross-disciplinary collaboration, balancing software and physical elements while analyzing trade-offs.

2.2 From modelling and Simulation to Digital Twins

The Digital Twin concept has surfaced in many areas, e.g., aerospace [28], manufacturing [29], healthcare [30], transportation systems [31], and smart cities [32]. The DT approach landed in the top strategic technology trends, as shown in the Gartner hype cycle of 2017 [33] and 2018 [34]. However, for most M&S researchers, the DT concept seems like a natural evolution of M&S. Though seemingly evolutionary, there might be some revolutionary aspects to the concept.

This section discusses the DT concept from the viewpoint of M&S experts. It both provides literature review elements and adopts a commentary-driven approach. We first examine the DT from a historical perspective, tracing the historical development of M&S from its roots in computational experiments to its applications in various fields

and the birth of DT-related and allied concepts. We then approach DTs as an evolution of M&S, acknowledging the overlap in these different concepts. We also look at the M&S workflow and its evolution toward a DT workflow from a software engineering perspective, highlighting significant changes. We aim to give the reader a broader understanding of the digital twin concept in this particular way.

2.2.1 Digital Twin: a Historical Perspective

In the 1960s, the DT idea was born at NASA from the "living model" of its Apollo missions [35]. Figure 2.1 illustrates the historical perspective of the emergence and budding of the DT concept.

The evolution of M&S has progressed alongside the significant advancements witnessed in computer science. This is why the history of M&S dates back to the 1960s when the first computers could support simulations. During the initial decades of simulation development, the generated output primarily consisted of textual reports [36].

In the 1980s, visual animation was integrated into commercial simulations to enable all stakeholders to communicate with the model appropriately. Since the 1980s, real-time simulation (RtS) has been increasingly used in industrial and entertainment applications [37]. In this approach, the simulation model generates outputs and responds to inputs at a pace corresponding to the system's real-world dynamics. These developments influenced other industries to adopt real-time simulation techniques for training, planning, and testing purposes. RtS possesses the capability not only to model the future state of systems but also to depict the current state of real systems such as hospitals, factories, distribution hubs, supply chain networks, airports, and container ports. The term "RtS" can be traced back to the 1950s, when Rubinoff defined RtS as a simulation of the performance of a process or device at its regular operating speed. The digital simulator is responsible for keeping up with the simulated process or device [38]. However, the RtS concept has evolved over time. The MIT Servomechanisms Laboratory and the United States Air Force developed the Whirlwind I project, which was publicly announced in 1951 [39]. Historically, the project has consistently focused on the domains of real-time simulation and control [40].

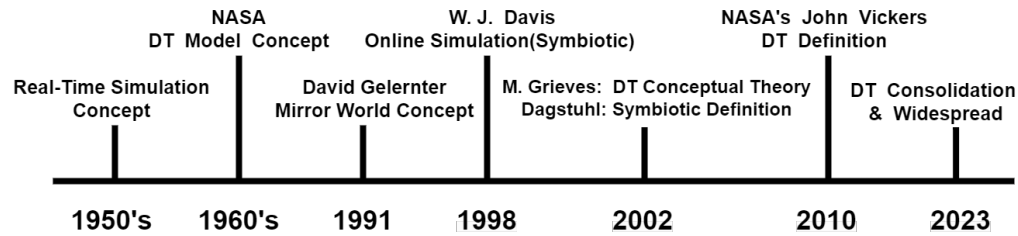

Figure 2.1: Historical perspective of the Digital Twin concept.

Most RtSs are identified by including some physical components in the simulation. This might be hardware, software, or a human(s) in the loop. In every case, the simulation

needs to be synchronous with a wall clock to ensure the correct timing of the interactions between the simulation and the external agent [41].

Simulation is one of the most important tools for evaluating system performance and assisting in decision-making. However, RtS for decision-making plays a key role in many sectors, spanning from manufacturing plants [42, 43] to sociotechnical systems such as healthcare services [44].

In 1991, we find Gelernter's concept of "Mirror Worlds", a software model of a part of the real world, fed with information streams such that the model is ever up-to-date with reality. He envisions this mirror world to be accessible by multiple users simultaneously, each of which can request and view exactly those aspects of the model they are interested in, at whatever level of detail necessary. A city Mirror World would contain the state of bridges, locations of policemen, occupancy of buildings, etc. Those occupied buildings would have a mirror world themselves, with, for example, in a hospital, digital versions of patients and doctors, but also rooms and medical inventory [45]. Gelernter's view is that of a doppelgänger and is bordering on what is nowadays called a DT.

The origin of the Symbiotic Simulation can be traced back to 1998 to Davis' concept of online simulation [46], but the term Symbiotic was introduced at the Dagstuhl seminar on Grand Challenges for modelling and Simulation in 2002 [47].

A Symbiotic Simulation System is a system in which a simulation and a physical system interact with each other through an exchange of data. The physical system sends measurements to the simulation, which sets up what-if experiments to control or influence that physical system optimally. In this initial definition, the Symbiotic Simulation System forms a closed loop. It behaves in a mutually beneficial way, but this does not need to be the case, as argued by Aydt et al. [48]. If the model used in the simulation does not accurately represent the physical system, suboptimal or even detrimental decisions might be made. Aydt et al. [48] also propose other uses of the Symbiotic Simulation System that do not require a closed loop, such as forecasting, model validation, and anomaly detection. We see that Symbiotic Simulation Systems have found their way into Industry 4.0 [49], and as Cao et al. put it succinctly [50]: "Symbiotic simulation systems describe the whole process of using a DT.".

Grieves's initial concept of DT is found in a University of Michigan presentation on Product Life-cycle Management (PLM). At the time, the slide was called "Conceptual Ideal for PLM", and over time, the concept was renamed to "the information mirroring model" initially and later on to "digital twin". This slide, reproduced in Figure 2.2, contains all the elements of what is nowadays considered a DT: a real space, which is mirrored by a virtual space (VS) that consists of any number of sub-spaces (VS1, VS2, ..., VSn), and the accompanying data flow from the real space to the virtual one, as well as an information flow in the opposite direction. The central idea is that the real space represents a physical system, and the virtual space represents all the information of this physical system throughout its lifecycle, from (prototype) production to disposal [1]. This view is similar to Gelernter's concept but is brought in the more specific context of PLM, and with the extension of an automated information flow from the virtual to the real space. It is also clear that there is a definite overlap with Symbiotic Simulation Systems. The main difference is that in Symbiotic Simulation, the one-to-one mapping of real space to virtual space is not a necessity, though it may be optionally present, e.g., in its model validation usage.

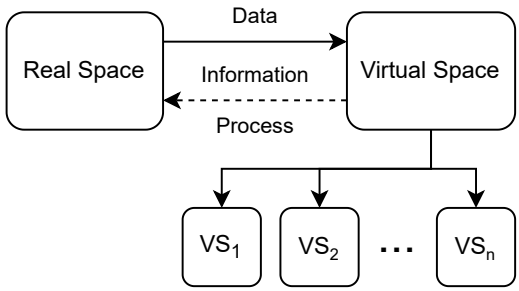

Figure 2.2: Adaptation of Dr. Michael Grieves' slide from 2002 [1].

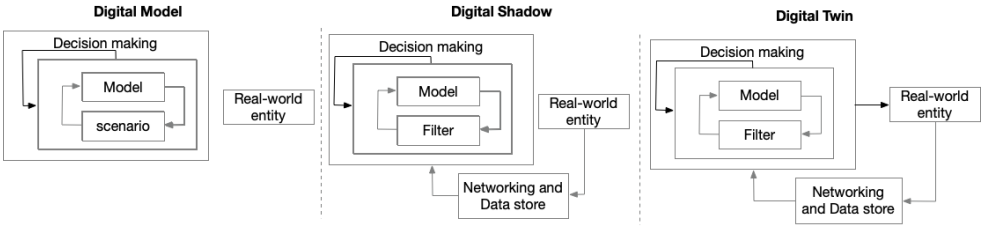

Figure 2.3: From digital model to DT according to Kritzinger et al.'s classification.

2010 brings us NASA's definition, which stems from a roadmap on modelling, simulation, and information technology [51]. NASA describes a DT as follows: "A Digital Twin is an integrated multiphysics, multiscale, probabilistic simulation of an as-built vehicle or system that uses the best available physical models, sensor updates, fleet history, etc., to mirror the life of its corresponding flying twin" [51, 52]. They consider the DT to be ultra-realistic, based on high-fidelity physical models, onboard sensor data, maintenance history, and fleet data. The central idea is that due to future missions being more complex and longer, a DT can aid by continuously forecasting the system health and the probability of mission success, as well as uncover issues before they become critical [52]. Compared to Grieves' view, this definition is less general and clearly influenced by an aeronautics background, yet the basic elements from Figure 2.2 remain present.

Several industrial consortia, individual researchers, and standardisation bodies are actively participating in the progress of DT technology. For example, "Alliance Industrie du Future" (AIF, a large French consortium of industries and academics) defines the DT as (i) an organised set of digital models representing a real-world entity designed to address specific issues and uses, (ii) updated in relation to reality, with a frequency and precision adapted to its issues and uses and, (iii) equipped with advanced operating tools including the ability to understand, analyse, predict and optimise the operations and management of the real entity [53]. As stated by the Digital Twin Consortium, a DT is a virtual replication of a real-world physical object, entity, or process, which is synchronised with its physical counterpart at a certain frequency and fidelity. The DT supports businesses through holistic understandings, optimal decision-making, and effective actions. DT predicts possible future scenarios and represents the present and past, by using real-time and historical data [54]. According to Siemens, a DT is a virtual duplicate of an object, machine, process, or a complete facility of production. It can carry all the data and models relevant to the real-world entity along all the value

chain processes, from design to production, operation, maintenance, and recycling of the product. This makes it possible to design, simulate, and manufacture products faster whilst improving the factors of economy, performance, robustness, or environmental compatibility [55]. General Electric (GE) defines the DT as a software representation of a physical asset, system, or process designed to detect, prevent, predict, and optimise through real-time analytics to deliver business value [56]. In an attempt to consolidate various DT definitions, Wright [11] distilled the following three required parts in a DT: (i) a model of the twinned system is needed, (ii) an ever-evolving dataset related to the twinned system is needed, (iii) a means of updating the model in accordance with the data is needed. This aligns with the definition given by the ISO (International Organisation for Standardisation), which defines the DT in a manufacturing context as the digital representation of an observable manufacturing element with synchronisation between the element and its digital representation [57].

Because all these definitions are rather broad, there have been attempts in the literature to define further classifications in the DT technology according to some properties. Kritzinger et al.'s classification [29] is a classification based on the level of data integration between a physical object and its digital counterpart. Specifically, the classification looks at the presence of computer-automated data/information exchange between the digital world and the real-world entity, as shown in Figure 2.3. Based on that presence, it distinguishes between a digital model, a digital shadow, or a digital twin. A digital model only has manual data exchange between the two objects, a digital shadow has automatic data exchange from the physical object to the digital one, and a digital twin has an automated data exchange in both directions; that is, the digital object can directly influence the physical one. Despite stemming from the manufacturing field, this classification is generic and broadly applicable. Babic [58] classifies DTs for smart manufacturing in two groups based on the awareness of the digital twin about the manufacturing equipment's layout. He classifies them as static twins, in which the equipment layout is configured manually, and dynamic, in which the twin automatically determines this configuration. Bao et al.'s classification [31] also stems from manufacturing, yet their classification focuses on what the DT captures: the produced product or the production process. They describe the product DT as a virtual information carrier of the product that carries information associated with that product through the various phases of its life, that is, design, manufacturing, maintenance, repair, and operations. The process DT then supports the production process and captures the appropriate attributes and manufacturing procedures in a digital way.

2.2.2 Digital Twin: an Evolution?

Some scholars posit that DT can be attributed to the evolution of the simulation model. For example, Lugaresi et al. [59] state that DT can be seen as an evolved form of simulation that has been used for years. Compared to mere simulation, the evolved features are a bidirectional data flow and synchronisation between the real and the virtual elements [59]. This idea is not just an assumption; in fact, it can be deduced from the previously mentioned DT definitions [1, 52, 11]. Shao et al. [60] argue that although concepts behind DT might be old and known by simulation experts, DT is however a prominent step over the simulation model because classic simulations typically represent what happened in the past or may happen in the future based on initial assumptions,

while DT focuses on what is happening right now and may be used to predict future states as well. VanDerHorn & Mahadevan [61] have the same viewpoint and argue that one reason for the potential confusion between simulation models and DT arises from the fact that while a simulation model is not necessarily a DT, the use of a simulation model combined with DT is prevalent.

Based on an evolved workflow, we look to clarify the evolution that has occurred.

2.2.2.1 M&S Workflow

As M&S is such a mature discipline, workflows exist for practitioners to guide them through a simulation study. The different concepts of the workflow are combined in the work of Balci, where a life-cycle model for M&S is defined [62].

Figure 3.1 defines a simplified view of such a workflow of a simulation study based on [62, 63]. Activities are shown using ellipses. The control flow between different activities is shown with a full arrow. The dashed lines show the interaction with the system under study. We distinguish the following activities in the workflow:

1.1 Model Objective Definition: This activity defines the reasons for the simulation study. The problem of interest is defined. From this high-level question, the simulation analysts and domain experts define the specific questions of the study. These specific questions are translated into the properties of interest. Furthermore, the scope of the model is defined.

1.2 Create the Conceptual Model: The conceptual model is the model that is formulated in the head of the developer [62]. Zeigler defined a hierarchy of system specification that can be used as a foundation for creating a conceptual model [64].

1.3 Create a Programmed Model: The conceptual model must be captured in an executable/programmed model. Different programming languages and simulation formalisms are available to create the executable model. Once the programmed model is available, we can check if it is a good implementation of the conceptual model and does not contain any errors. This process is often referred to as *verification*.

1.4 Calibration or Parameter Estimation: The conceptual and programmed model are typically parametrised, so for a virtual experiment, these parameters must be assigned a value. Some parameters can be taken from component data sheets or literature. However, sometimes experiments need to be set up to measure the parameter. Finally, some parameters can only be estimated using optimisation techniques to ensure the model's output is calibrated to the system's output.

1.5 Model Validation: Once a calibrated model is available, it still needs to be checked if it has any predictive capabilities within its domain of applicability. The process is called *validation*: "A computerised model within its domain of applicability possesses a satisfactory range of accuracy consistent with the intended application of the model" [65]. Different techniques and statistical metrics are available for doing model validation [3].

1.6 Model Experimentation and Decision-making: The model can now be used for its purpose. In silico experiments are conducted using designed experiments. The results are used for decision-making, understanding a system, etc.

Note that most of these activities are done iteratively. Furthermore, the simplified life-cycle model does not show feedback loops to return to previous phases when needed. We refer to Balci [62] for a more detailed treatise of the life-cycle of M&S.

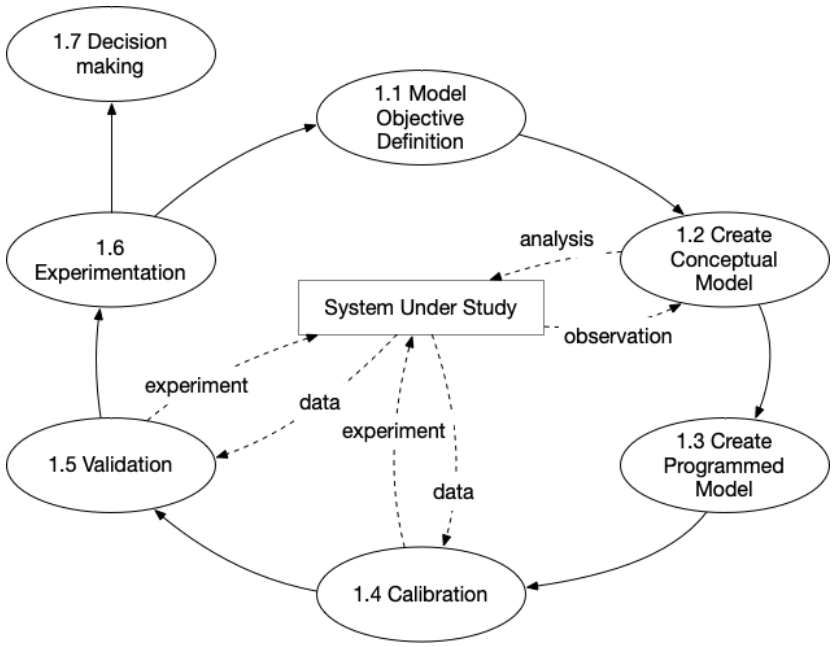

Figure 2.4: Simplified M&S workflow.

2.2.2.2 Evolution Towards DT Workflow

A simplified DT workflow is shown in Figure 2.5. This workflow is based on our experience of building digital twins and on typical model-based systems engineering workflows and standard software life-cycle models such as DevOps [66]. Compared to the simplified simulation study workflow, a lot has changed. A single step within this workflow contains the entire simulation study workflow. We see the following activities.

2.1 DT Objective definition: A digital twin is developed for a specific purpose, e.g., optimisation of performance parameters of the system, control-oriented applications, monitoring, and dashboarding. Based on these specific objectives of the DT, the developers create a set of requirements for the DT. The requirements and specifications also include the operational domain of the DT. The requirements translate into the specific properties of interest that the DT needs to work on.

2.2 Model development life-cycle: This activity contains the life cycle shown in Figure 3.1. Based on the requirements and specifications of the DT, a model needs

to be created or possibly reused from a library of models. M&S experts use the requirements of the DT to translate them into model requirements.

2.3 DT Architecture: The architecture of the digital twin is created. Decisions, such as using a distributed or centralised architecture, are taken. Besides the model, the DT needs many other components to operate, e.g., instrumentation, data collection, networking, data storage, and decision-making. Furthermore, the model is used for a specific purpose within the DT, e.g., what-if analysis or optimisation. Decision-making components, based on the simulation outcomes, are defined.

2.4 DT Create/Build: Based on the architecture, the digital twin is developed, e.g., coding and testing of software components and setting up the networking and data infrastructure.

2.5 DT Deployment: Deployment is releasing the digital twin for use. The different data streams are connected to the DT. Deployment of a DT is typically done in cloud environments. However, fog and edge computing are also considered [67].

2.6 DT Verification and Validation: As a digital twin is a complicated software-based system, current software engineering practices should be considered. Verification is the process of checking if the services offered by the DT are created correctly. Validation checks that the services provided by the DT meet the needs of the system users. Note that verification starts in the DT build phase when testing the various components.

2.7 Data Collection: From the instrumented environment and system, the data is gathered by the system for use by the DT services and validation processes. Depending on the services, the data is also made persistent for later use.

2.8 DT Services: These are the services that implement the objectives of the digital twin. The services use the data and actuate the system. The service typically runs automated simulation experiments to support the automated decision-making provided by the DT service. Matta and Lugaresi classify these services as descriptive (e.g., health-monitoring), predictive (e.g., prognosis), and prescriptive (e.g., optimisation) services [68].

2.9 DT Synchronisation: The current state of the system must be estimated to be able to use the digital twin properly. These estimates can be used in the underlying models to initialise the models. Once the state is estimated, we need to be sure that the model that is used within the digital twin is up-to-date with the system and its environment. This is necessary because the system and environment can evolve over time (e.g., wear and tear, replacement of components). Furthermore, once it is detected that the underlying models are no longer valid, the model should be brought back in synchronisation with the actual DT. The parameters must be updated if the model is valid for a larger operational domain (and/or a new initial state should be estimated). However, in some instances, a new model should be created or selected for use.

Note that the activities shown in grey, in Figure 2.5, are offline or development activities, while those in white are online or run-time activities. We also note that the semantics of this model might be slightly different compared to the semantics of Figure 3.1. While the digital twin is operational (and thus providing services), the continual validation

techniques can run in parallel. Furthermore, a new model must be created when the current model is no longer valid, while certain DT services are still up and running.

The two workflow models show that the DT workflow builds on top of the M&S workflow. From that perspective, DT can be seen as an evolution of M&S. However, when examined thoroughly, such evolution also brings new requirements for the concept to be truly feasible, i.e., to allow the simulation model to co-evolve with the actual system, capturing and reflecting its modifications. Defining new methods, techniques, and tools to address those requirements could lead to a revolution in the area of M&S.

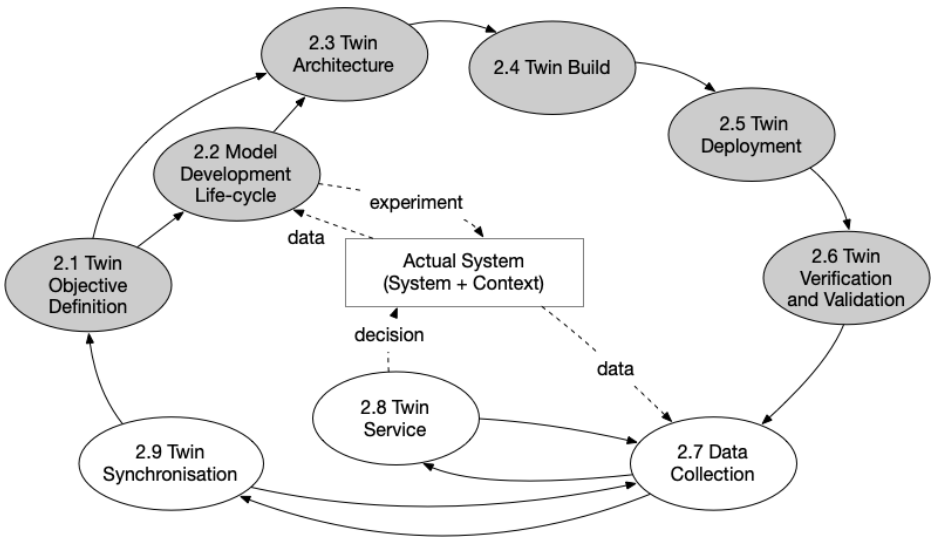

Figure 2.5: Simplified workflow for Digital Twins

2.3 Twinning Paradigm

In the presence of a plethora of definitions and interpretations for digital twin [69], despite its increasing popularity, the concept of digital twin remains inconsistently defined. Rumpe identified 112 distinct definitions, as noted in the most comprehensive DT literature review to date [70]. Although some convergence exists, many definitions are either too narrowly focused on specific domains [71]. overly vague [72], or even contradictory [73]. This lack of clarity extends to related terms, such as Digital Shadow, Digital Model, Digital Avatar, Digital Passport, and Digital Thread, leading to further confusion and inconsistency in the field.

In response to this challenge, Paredis and Vangheluwe [74] propose bringing the Digital twin, Digital Shadow, and Digital Model under the umbrella of the '*Twinning paradigm*'.

Twinning connects a System under Study, also known as the Actual Object, and the corresponding Twin Object, which is a model represented in a suitable formalism along with its simulator or executor [75]. The actual object can be a physical entity, such as a machine, or a digital entity, like software. Correspondingly, the twin object can also be either a digital or a physical representation [74].

DTs have a wide range of applications and purposes [76]:
(a) Potential futures: The provision of strategy and planning support, the execution of 'What if?' scenarios [77], and the implementation of predictive and preventive maintenance programs like state estimation [78], anomaly detection [79], and self-adaptation [80, 81].
(b) Current state: The management of interventions, including operation and maintenance interventions as well as capital investment projects, real-time status monitoring, and control, together with diagnostics and prognostics, which play a significant role in optimising asset performance and safety [82, 83].
(c) History: Keeping records and learning from the past.

In this study, we focus on the second purpose of DTs for applications requiring real-time decision-making and real-time control.

The availability of quantitative data and advanced analytics in real-time via DT enables better-informed and faster decision-making. However, making decisions in particular situations must adhere to real-time constraints. As a result, the DT model should be fast enough, which is in the meantime sufficiently physics-based and accurate as well [11].

2.4 Multi-Paradigm Modelling

In general, a paradigm serves as a pattern for describing an entire class of artefacts that share similar characteristics, or as a framework that encapsulates theories in a scientific field [84].

Multi-Paradigm Modelling (MPM) is widely acknowledged as a strong approach (a paradigm in its own right) that can aid in designing, communicating, and reasoning about CPS. CPS are known for their complexity because of their cross-disciplinary borders and inter-domain interactions [85]. For developing CPS, project managers and engineers must carefully choose the most appropriate development languages, software lifecycles, and interfaces to define various system views, components, and interactions, with as minimise "accidental complexity" [86] as possible. For instance, if system/software requirements are expected to change frequently during the project's lifecycle, adopting an Agile development process can enhance adaptability and responsiveness to evolving needs. Similarly, if the system's behavior depends on event-driven operations triggered by data availability (as seen in reactive systems), Data Flow languages offer a structured approach for specifying critical software behaviors with precision, ensuring they remain well-suited for timing analysis.

The implementation of MPM needs to *model everything explicitly* while using the *most appropriate formalism(s)*, at the *most appropriate level(s) of abstraction* [87]. To find the most appropriate model for a given context, it is important to represent the actual system at different levels of detail. These models have different levels of detail while still representing the same actual system by fulfiling its property(s) of interest.

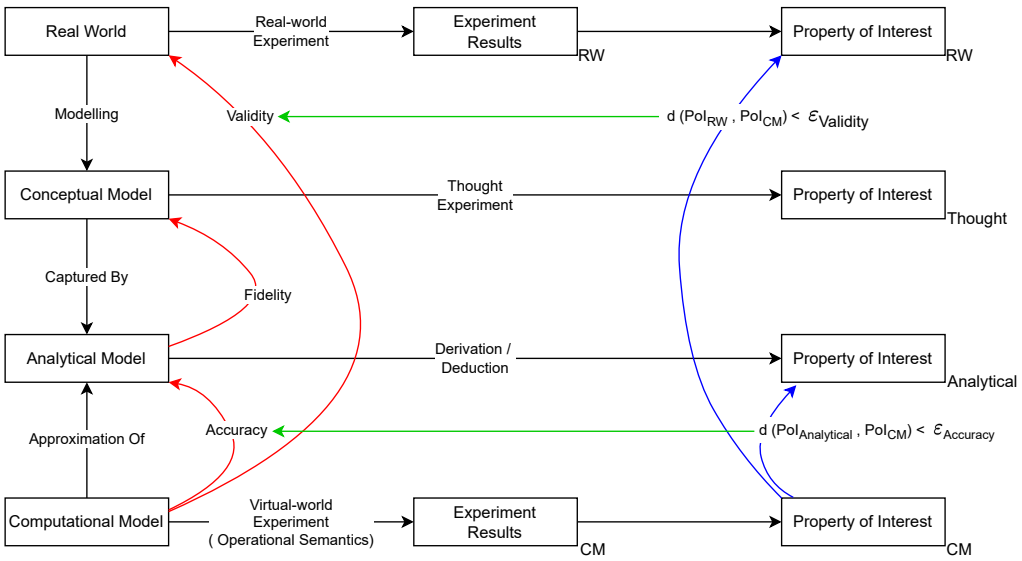

Figure 2.6: Modelling relations *(Inspired by Hans Vangheluwe's presentation from 'Using multi-* modelling to manage complexity in systems engineering' seminar, Shonan, March 2025 [2])*.

2.5 Definitions

When building a model for a specific system and objective, only some models are appropriate, while others may not be. Therefore, ensuring the success of a simulation modeling effort requires maintaining key relationships. Among these, the two most fundamental are the modeling relation, which defines how the model represents the real system, and the simulation relation, which governs how the model behaves under different conditions to generate meaningful insights. Figure 2.6 shows these relationships in detail.

In modelling, clear definitions of core concepts are essential for consistent analysis and communication. In the following chapters, these terms will be used as defined here to ensure clarity and consistency in the discussion of modelling practices.

2.5.1 Model

Stachowiack [88] defines the fundamental characteristics that constitute a *model* as: (a) *Mapping feature:* A model describes some original, real system/entity that either exists or will happen to exist in the future. (b) *Reduction feature:* A model is an abstraction of the original; this means it only reflects a relevant selection of the original's properties. (c)*Pragmatic feature:* A model has a purpose with respect to the original.

Minsky [89] defines a model as "To an observer B, an object A* is a model of an object A to the extent that B can use A* to answer questions that interest him about A."

While Zeigler [18] defines a model as "A model is a representation of a simuland, broadly

grouped into conceptual and executable types. Model is a set of rules for generating behavior, can be represented by a system specification at a structural level." Simuland is the real-world system of interest.

2.5.2 Accuracy

In modelling and simulation context, accuracy is defined as the relation between the computational model and the analytical model. On the other hand, accuracy is the metric distance between the properties of interest derived from virtual-world experiments and those derived from the analytical model, illustrated in Figure 2.6.

$$d(PoI_{Analytical}, PoI_{CM}) < \varepsilon_{Accuracy} \tag{2.1}$$

2.5.3 Validity

Validity is the fundamental modeling relation that defines the relation between a computational model and the real-world system it represents, as confirmed through real-world experiments. On the other hand, validity, as shown in Figure 2.6, is defined as the metric distance between the properties of interest derived from real-world experiments and those obtained from virtual experiments.

$$d(PoI_{RW}, PoI_{CM}) < \varepsilon_{Validity} \tag{2.2}$$

2.5.4 Fidelity

Fidelity in modelling and simulation context means the degree to which a model or simulation accurately represents a real-world system it is intended to simulate. As shown in Figure 2.6, fidelity is the relation between the analytical model and the conceptual model with respect to their properties of interest.

Fidelity encompasses various aspects, including accuracy, resolution, and the level of detail in the model. In the context of DTs and modelling and simulation, fidelity is crucial for ensuring that the model's predictions align closely with actual system behavior. *High-fidelity models* provide detailed and accurate representations, while *low-fidelity models* may sacrifice some detail for computational efficiency. The choice of fidelity level depends on the specific requirements of the application and the trade-offs between accuracy and computational cost.

2.5.5 Abstraction

Abstraction is the process of omitting or reducing the amount of detail that doesn't count as necessary in a model [18]. Abstraction is a specific technique that preserves exact simulation results. It reduces model complexity without reducing accuracy, but with less detail. An abstraction concentrates on a particular aspect of reality and, by

definition, significantly simplifies the complexity of the reality under consideration [90]. In general, abstraction is removing the characteristics of the model, preserving exact simulation results without reducing accuracy.

2.5.6 Approximation

Although there are different definitions of approximation, for example, the one from Frantz and Ellor [90], which talks about approximation as an approach to eliminate exogenous inputs to the model that do not influence the overall conclusions of the simulation. In this research, we define approximation as a process that approximates the characteristics of a model, resulting in a simpler model with less accuracy within the tolerance required to answer a given query about the system.

2.5.7 Abstraction and Approximation Techniques

There are different techniques to drive a simplified model from a high-validity model while preserving model accuracy concerning properties of interest, such as model abstraction, model approximation, and surrogate modelling, etc. Approximated and abstracted techniques are known as simplifying transformation techniques.

2.5.7.1 Model Approximation

We have identified several approaches in the literature, including the approach based on structural approximation, including Causal Approximation [91], and approximation based on sensitivity analysis of the accuracy of a model defined by Weld [92]. The other well-known technique is metamodelling, which approximates complex models to simplify complex system simulations by focusing on specific constraints or regions of interest [93].

However, surrogate models have become popular in engineering design due to their capability to approximate computationally expensive engineering systems. A surrogate model is a model that is less computationally expensive and faster to run [11]. Selecting the appropriate surrogate model structure is critical in practice because of the multiple variants and variances in their hyper-parameter configurations. The process of designing complex engineered systems generally entails the exploration of the whole design space, which requires many runs of expensive simulations. In order to reduce the computational cost associated with these simulations, researchers have developed surrogate modelling techniques to approximate these simulations and significantly reduce computational costs.

The selection of a surrogate model is frequently approached in one of two ways [94]: (1) Manual comparison-based surrogate model selection. (2) Evolutionary algorithm-based surrogate model selection. Selecting the right surrogate model is important, as it directly affects the accuracy, efficiency, and reliability of the model. An inappropriate model can lead to inaccurate predictions, poor decisions, or increased computational costs. Factors

such as the properties of interest, design space complexity, available resources, and the balance between accuracy and cost must guide model selection.

An appropriate surrogate model minimises computational cost while keeping sufficient accuracy. Careful evaluation of the model's characteristics and compatibility with the properties of interest and goals are essential to ensure optimal performance.

2.5.7.2 Model Abstraction

Model abstraction is an approach for reducing model complexity by focusing on a particular set of properties of interest. A model is built specifically with the purpose of addressing and resolving a certain problem. Therefore, it can be observed that a system has an endless number of models, each of which has the potential to be the most suitable for what it does. Numerous formalisms exist for expressing abstracted models of system behavior, which might vary in terms of their level of abstraction and complexity. The choice of specific formalism and abstraction level is based on the modeler's expertise and the specific features of the system being modeled [95]. A wide range of model abstraction techniques has been explored in the literature [90].

To elucidate the concept of a model abstraction, the following is a straightforward illustration of a grocery store model, which has been adapted from Zeigler [96]. There is a grocery store with a single cashier lane in the real-world system. Customers enter the store, do shopping for a while, and then proceed to the checkout line. They may wait until they reach the front of the line, at which point they check out and leave the store. When a customer enters the store, the model input set includes the customer's identity and time. A probability distribution function assigns a shopping time duration to each entering customer. At the end of the shopping time duration, a checkout duration time is calculated from a probability distribution function if the checkout line is empty. The customer leaves the store at the end of the checkout duration time. If the checkout line is not empty, the customer is assigned to a first-in, first-out queue. When a customer reaches the front of the queue, they are assigned a checkout duration time based on a probability distribution. Once the checkout time is up, they exit the store.

This simulation model could be used to calculate the average duration a consumer spends in the store, as well as the average waiting time that they spend in the checkout line. Assume, however, that the simulation requirement is to calculate the average duration for which the checker was busy, as well as the average length of time the checker has to work without a break. Certain aspects of the model are irrelevant to the present problem. We need not focus on specific customers; indeed, we need not indicate individual customers in the queue. We abstract the model by initially removing the identities of individual customers. Upon the completion of a customer's shopping duration, the checkout time is promptly calculated and included into a model variable that indicates the duration till the cashier is no longer busy.

The sole required model outputs are the times at which the checker's state (Busy or Idle) changes. The queue and the order of customers in the queue are no longer relevant and can be ignored. The results' validity is maintained due to the fact that the model output of the checker state changes is identical for both models.

2.6 Validation

Abstracted or approximated models are not necessarily applicable to the entire range of contexts that a system may encounter. It is necessary to identify a frame within which the model remains applicable.

The validation domain, illustrated in Figure 2.7, represents the region of parameter space where validation experiments confirm model accuracy. The relationship between the application domain and the validation domain shown in this figure is part of the class of relationships between them, so the two domains may partially overlap or be entirely disjoint. We refer to the work by Oberkampf and Roy for possible relationships of the validation domain to the application domain [3].

Validation experiments involve testing the model under specific conditions within the validation domain to evaluate its predictive accuracy. Candidate (C_i, $i = 1, 2, ..., 5$) conditions are selected from the application domain, typically at the boundaries or corners of the operating envelope, $(\alpha_i, \beta_i), i = 1, 2, ..., 5$, where inaccuracies are most likely to occur. These experiments prioritise conditions where model accuracy is critical to project requirements.

Using the concept illustrated in Figure 2.7 and validity frame information helps form the basis for adaptive abstraction and approximation by facilitating the knowledge creation.

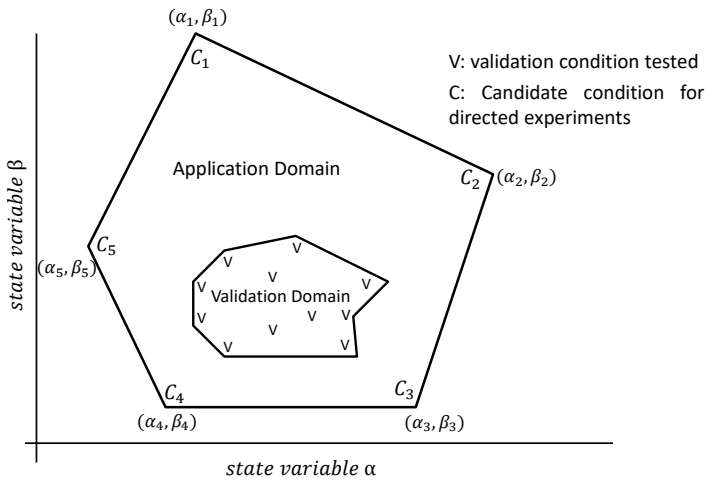

Figure 2.7: Application domain vs validation domain, adapted from [3, 4].

2.6.1 Experimental Frame

The concept of 'frames' in modelling and simulation originates from the early 1980s, introduced as "experimental frames" by Zeigler [97]. An experimental frame represents a set of conditions under which a system is observed or experimented with by real

or virtual experiments. These experimental frames consist of three main components: a) Generator – produces input traces for the model, b) Acceptor – ensures that the experimental conditions are fulfilled, and c) Transducer – examines and interprets the output signals.

2.6.2 Validity Frame

Building on the experimental frame idea, validity frames explicitly encode the contexts in which a model provides valid results for specific properties relative to a real-world system [98, 99]. The concept of validity frame formalises the contexts in which a model provides reliable results, and it is closely related to the methods for verification and validation in simulation modelling defined by Sargent [100]. Furthermore, Klikovits et al. [101] observed that a model's frame is shaped by the specific activity being performed, highlighting why different activities require different frames. In response, the VF concept was introduced to define the particular context in which the model consistently produces predictable results. This concept is further categorised into abstract and concrete validity frame [102].

- ***Concrete Validity Frame:*** The finite set of performed experiments in which a model is valid is called a concrete validity frame [102].
- ***Abstract Validity Frame:*** Abstract validity frame is the possibly infinite Set of experiments e for which the distance d between the obtained (computed) Properties of Interest (PoI) from e carried out in the real world and e carried out in the virtual world is (larger)smaller than a treshold Tr [102].

Validity frame is a conceptual structure which built on top of model boundaries, including Specific assumptions, constraints, and conditions under which a model is regarded as an accurate and reliable representation of the real world.

Even though recent studies highlight the utility of validity frames, to date, there has been no precise and general definition of the concept. In this study, we define the validity frame of a model as its experimental frame, plus a process that allows us to change the experimental frame. Such a process must include minimising the overlap between the experimental frame of the model with respect to the other experimental frames in a multi-model setup.

Let EF denote the experimental frame. We then define the VF as:

$$VF = EF + P(EF) \tag{2.3}$$

Here, the "+" is an abstract operator indicating the aggregation (or augmentation) of the original experimental frame with the modifications provided by the process P.

The use of validity frames is beneficial for several reasons, as they explicitly represent constraints and conditions that have always existed but were often implicit. By formally defining these boundaries, validity frames provide the following advantages:

- Accuracy: They ensure that models are only employed in contexts where they provide reliable results.
- Model Selection: Validity frames serve as a structured mechanism for determining the appropriate model for a given system state.
- Reliability: They guarantee that the system operates within validated conditions, minimising the risk of errors.
- Efficiency: By enabling the safe use of computationally less expensive models when appropriate, validity frames optimise execution time while maintaining reliability and model accuracy.

2.7 Self-Adaptation

Weyns[7] defines a self-adaptive system based on two main principles. First, an external principle: a self-adaptive system is a system that can handle changes and uncertainties in its environment. Second, an internal principle: a self-adaptive system contains two parts: a part that interacts with the environment, the managed system, and a part that handles adaptation logic, the managing system [7]. Self-adaptation pertains to the ability of a system to change parts of its operating algorithm over time [103]. The DT enables the implementation of self-adaptation and processing large amounts of data within a specified timeframe [81].

The most widely accepted reference model for self-adaptive systems is the conceptual MAPE-K model. IBM introduced MAPE-K in 2003. MAPE-K is a closed high-level feedback control loop shown in Figure 2.8 [5]. The managed system, the domain-specific system, and the system manager are all standalone parts of the MAPE-K architecture shown in Figure 2.8. The managed system is a system or component that can be managed. The managing system is divided into four phases that use common knowledge: **M**onitor the managing system and its context. **A**nalyze the issue and determine whether any adaptations are necessary. **P**lan the adaptation to a new configuration and **E**xecute a mode change protocol to carry out the transition to this new configuration.

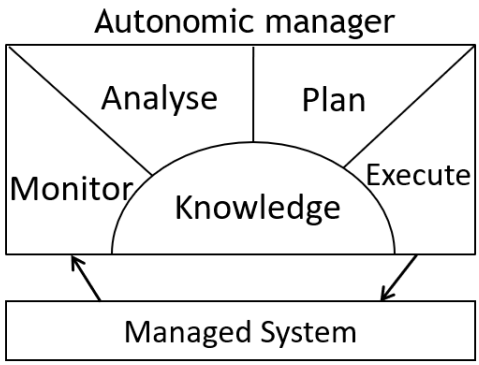

Figure 2.8: MAPE-K architecture from IBM [5].

2.8 Real-Time Systems: Execution Constraints and Scheduling

Real-time systems play a crucial role in applications where the correctness of operations is determined not only by their logical outcome but also by their timing behavior. To ensure timely and predictable task execution, these systems rely on accurate timing analysis and efficient scheduling strategies. This section provides an overview of the foundational concepts relevant to real-time computing, including system classifications, Worst Case Execution Time (WCET) analysis, and scheduling algorithms. In the following, we discuss how real-time constraints matter in DTs.

2.8.1 Real-Time Systems

Real-time systems are computing systems where the correctness of an operation depends not only on the logical result of the computation but also on the time at which the results are produced. These systems are often embedded within physical systems and used in domains where delayed or untimely responses can lead to performance degradation or critical failure [104, 105]. The primary distinction between a real-time and a non-real-time task at the process level is that a real-time task is defined by a deadline, which is the maximum time allowed for its execution to be completed. A real-time task can be classified into three categories based on the potential consequences of missing a deadline [105]:

- **Hard:** A real-time task is considered to be ***hard*** if the results are produced after the deadline and the system under control may suffer catastrophic consequences.
- **Firm:** If completing a real-time task after its deadline makes the results useless to the system while causing no harm, we call the task ***firm***.
- **Soft:** A real-time task is considered ***soft*** if the system still derives some benefit from the results produced after the deadline, despite the performance degradation.

The system must be capable of predicting and potentially avoiding resource conflicts to meet timing constraints, while also remaining flexible enough to accommodate a highly dynamic and adaptive environment [106].

2.8.2 Worst Case Execution Time (WCET)

If the worst-case input for the task were known, it would be relatively simple to provide a solid guarantee based on the worst case execution time of a task. Unfortunately, the input that would result in the worst case input is typically unknown and hard to figure out.

We suppose that a real-time system is composed of several tasks that realise the needed functionality. Figure 2.9 illustrates key aspects of a real-time task. Typically, the execution times of a task show a certain degree of variability, which is dependent on the input data

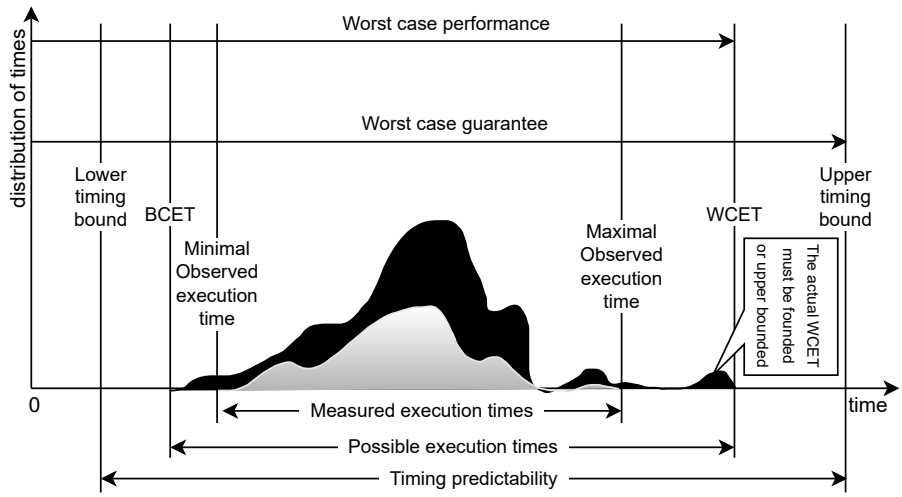

Figure 2.9: Basic concepts of timing analysis. The lower curve shows a subset of measured executions, with its minimum and maximum representing the minimal and maximal observed execution times. The darker curve, which envelopes the first, represents the times of all executions. Its minimum and maximum define the best case execution time (BCET) and worst case execution time (WCET), respectively [6].

or the environment's behavior. The upper curve represents all execution times. The best case execution time (BCET) is the shortest execution time, while the WCET is the longest [6].

Since the early days of embedded systems, WCET has been a crucial concept in computer systems, especially in real-time and embedded systems. WCET is particularly important in safety-critical systems, where missing a deadline could lead to catastrophic failures, such as in avionics or automotive control systems.

Historically, WCET analysis has evolved alongside the increasing complexity of software and hardware. In the early days of embedded computing, developers relied on manual static analysis and end-to-end measurements to estimate execution time. These methods had limitations, such as requiring extensive testing to capture the longest execution path and being prone to errors due to hardware unpredictability.

As real-time systems became more sophisticated, especially in automotive, aerospace, and industrial applications, the need for more reliable WCET analysis grew. Modern approaches now use advanced static analysis techniques and measurement-based methods to provide more accurate execution time estimates.

Wilhelm et al. [6] highlight the complexity of WCET estimation due to modern hardware features like caches, pipelines, and branch predictors, which introduce timing variability. WCET can be analysed using:

- Static analysis: Tools like aiT from AbsInt use abstract interpretation to safely over-approximate WCET.

- Measurement-based analysis: Involves running the code with diverse input scenarios and measuring execution time. However, it may underestimate WCET
- Hybrid methods: Combine measurements and static techniques to balance precision and safety.

Underestimated WCET leads to deadline misses, while overestimation results in resource underutilisation, making accurate analysis crucial [105].

As hardware complexity continues to grow, WCET analysis remains a challenging but essential field, ensuring reliability in systems where timing correctness is as critical as functional correctness.

2.8.3 The Shift Toward Statistical WCET Analysis

WCET analysis has shifted towards statistical approaches due to the increasing complexity of modern computing architectures and the need for more realistic execution time estimates. Traditional WCET analysis often relied on highly pessimistic worst-case scenarios, which could lead to excessive resource allocation and inefficiencies.

Statistical methods, such as Extreme Value Theory and concentration inequalities, help reduce this pessimism by considering the probability distribution of execution times rather than just the absolute worst case [107]. These approaches allow for more accurate and practical WCET estimates, making them particularly useful in real-time systems where precise timing predictions are crucial.

Additionally, probabilistic WCET models have gained traction because they better capture dependencies between execution times and provide more flexible analysis frameworks [108]. This shift is especially relevant in industries like avionics and automotive systems, where safety-critical applications require reliable timing guarantees without unnecessary conservatism [109].

2.8.4 Scheduling Algorithms in Real-Time Systems

Scheduling is the core mechanism for determining task execution order in real-time systems. Its goal is to ensure that all tasks meet their deadlines, given their WCET, period, and priority. The following primary classifications can be distinguished among the number of algorithms that have been proposed for the scheduling of real-time tasks [105]:

- ***Preemptive vs. Non-preemptive:*** Preemptive scheduling allows higher-priority tasks to interrupt lower-priority ones, while in non-preemptive algorithms, once a task starts, it executes until completion.
- ***Static vs. Dynamic:*** Static scheduling assigns fixed priorities, e.g., Fixed-priority preemptive scheduling. The optimal way to assign these fixed priorities (under the conditions that the tasks are non-related, and the deadlines equal the periods of the task) is Rate Monotonic Scheduling (RMS). while dynamic (e.g., EDF) adjusts priorities at runtime.

- ***Off-line vs. Online:*** Scheduling algorithm is executed on the entire task set before tasks activation, while in Online scheduling, scheduling decisions are taken at runtime when a new task arrives or a running task terminates.
- ***Optimal vs. Heuristic:*** Optimal algorithm minimises some given cost function defined for the task set, and a heuristic algorithm tends toward the optimal schedule, but does not guarantee that.

Further information and in-depth analysis of these scheduling algorithms are beyond the scope of this section and are not fully covered in this work. For a more in-depth debate, the reader is referred to [105].

2.8.5 Relationship Between WCET and Scheduling

Scheduling analysis relies on WCET estimates. For instance, schedulability tests for RMS and EDF require knowing each task's WCET to calculate total utilisation and determine if all deadlines can be met.

Inaccurate WCET leads to flawed scheduling decisions:

- Underestimation: missed deadlines and potential system failure.
- Overestimation: low CPU utilisation and inefficient resource use.

Thus, WCET estimation and scheduling are tightly coupled in the design of real-time systems [6].

2.8.6 Real-time Constraints in DTs

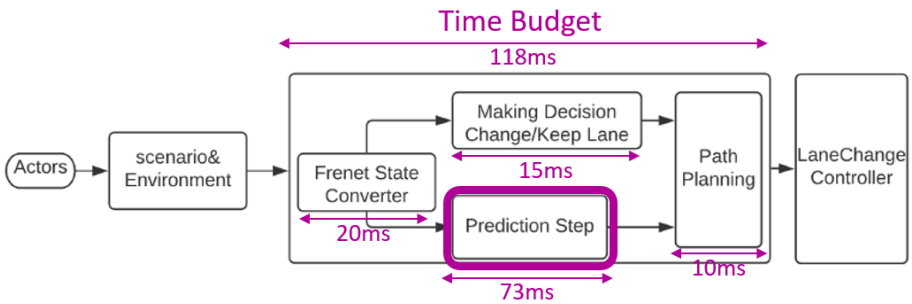

Figure 2.10: Causal block diagram of lane change maneuver case study.

In the context of embedded systems and DTs, real-time constraints are critical. Embedded systems often operate with limited computational resources, requiring models to be optimised for efficiency. Likewise, DTs must continually synchronise with actual systems and execute simulations in a real-time context, resulting in computationally expensive models and real-time constraints.

Model execution on embedded systems and within DTs necessitates balancing accuracy and computational cost. High-validity models, while providing more details of the actual system, are computationally expensive and may exceed the resource limitations of embedded hardware or fail to meet the real-time requirements of DTs. This is where MRM and adaptive abstraction and approximation techniques, such as the SACube framework, become essential. By dynamically switching to a simpler model valid within specific contexts, SACube reduces computational overhead while maintaining the reliability of decisions.

Furthermore, understanding and predicting the computational cost of a model is critical for ensuring real-time performance. Computational cost prediction involves analysing factors such as the complexity of the model, the number of operations required, and the available processing power.

Figure 2.10 is the high-level architecture of the lane change maneuver system shows that in this case study, the focus is on the prediction. In DT systems with real-time constraints, each computational task must complete within a specified time budget to ensure system responsiveness. The sequence of tasks in Figure 2.10 is periodically scheduled. Perhaps the entire timing chain needs to be executed every 250 milliseconds. During the system engineering, the time budget for the predictive step of the algorithm was defined to be a maximum of 118 ms. The other part of the budget is divided between the lane change controller and the sensing part.

In the scenario shown in Figure 2.11, there are three cars, including the ego car and two cars surrounding the ego car. In this example, there is an arbitrary time budget of 73 ms for the prediction step, denoted as $T_{\text{budget}} = 73\,\text{ms}$. The prediction execution time is denoted as T_{pred} is defined as the wall-clock time taken to compute a single forward prediction of the vehicle's position for the next simulation timestep. The system satisfies

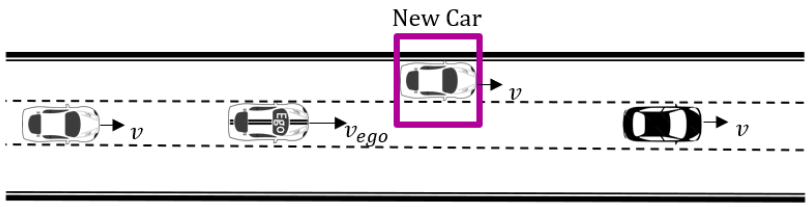

Figure 2.11: Scenario illustrating the ego car and surrounding cars. Introducing a fourth vehicle (highlighted) results in prediction time exceeding the 73 ms constraint, requiring a model switch to fulfill real-time requirements.

the real-time requirement if:

$$T_{\text{pred}} \leq T_{\text{budget}}$$

Initially, with three vehicles, the prediction time satisfies this condition. However, when a fourth vehicle is introduced, highlighted with a bounding box, the computational load increases, resulting in $T_{\text{pred}} > T_{\text{budget}}$, and the system misses its deadline.

Once the deadline is missed, the high-validity model becomes infeasible under the current constraints. To maintain real-time operation, one of the solutions is that the system switches to another model with the narrowest validity frame and reduced computational complexity, ensuring that T_{pred} again satisfies the inequality.

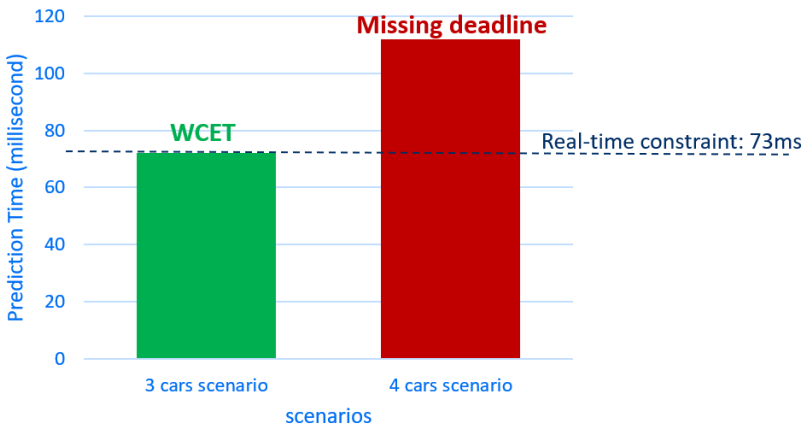

Figure 2.12: Impact of increased vehicle count on real-time constraint: With four vehicles, the WCET approximation exceeds the 73 ms approximation of the 3-car scenario, necessitating a switch to a valid model to meet the real-time constraint.

Figure 2.12 illustrates this behavior: The increase in vehicle count impacts the real-time constraint. With four vehicles, the approximated WCET surpasses the approximated 73 ms WCET of the three-vehicle scenario, necessitating a switch to a valid model to meet the real-time constraint.

2.8.7 Synchronous communication in Real-time System

Regular objects communicate through messages, while synchronous objects rely exclusively on signals. This fundamental difference necessitates the creation of communication protocols, both for interactions among synchronous objects and for their integration with regular objects [110]. In the context of our lane-change maneuver case study, signals are employed for interactions; however, a comprehensive understanding of the underlying principles remains essential.

Synchronous communication means that objects share the same sense of time when they interact. The execution machine manages this using the Clock class, which allows multiple "clocks" to exist in a program to handle events at different time scales. Each clock organises how objects respond based on their connections and dependencies, ensuring they react in the right order. All objects controlled by the same clock form a clock domain [110].

Synchronous Data Flow (SDF) is a pivotal model for structuring real-time embedded systems, particularly in applications. According to Lee and Messerschmitt [111], SDF is a specialised form of the more general data flow paradigm. In an SDF graph, computations are expressed as directed graphs where nodes represent functions and arcs delineate signal paths. A key feature is that each node produces or consumes a fixed number of data tokens per firing, a property that can be specified a priori. This constraint allows for static scheduling at compile time, minimising runtime overhead and making it possible to determine exact execution timings and buffer allocations in advance.

Moreover, SDF naturally accommodates multiple sample rates within a single system. By balancing the predetermined token production and consumption rates, the model supports even complex, heterogeneous signal processing tasks, ensuring that data dependencies are maintained and memory usage is optimised. Static buffering, one of the techniques derived from this approach, precomputes the locations of data samples in memory so that each node accesses the same locations across iterations, thereby reducing runtime management costs.

These characteristics—deterministic execution, efficient memory management, and inherent support for parallelism—make SDF a powerful tool for real-time applications, such as automotive control systems and other embedded implementations that require predictable behavior and high concurrency.

Chapter 3

Challenges

This chapter is based on the following publication:

Z. Ali, R. Biglari, J. Denil, J. Mertens, M. Poursoltan, and M. K. Traoré, "From modeling and simulation to digital twin: evolution or revolution?" *SIMULATION*, vol. 100, no. 7, pp. 751–769, 2024.

The adoption of DT technology has transformed how complex systems are modeled, analysed, and managed. By creating a virtual representation of the actual systems, DTs enable real-time monitoring, simulation, and decision-making. However, as promising as this paradigm is, it introduces a number of challenges that must be addressed to realise its full potential. These challenges arise from the complex interaction of data acquisition, system complexity, real-time synchronisation, and the scalability of digital models.

This chapter explores the key challenges encountered in developing and deploying DTs. These include the difficulties of ensuring reliable and timely data pipelines, achieving seamless interoperability across heterogeneous systems, and maintaining synchronisation between DT's components. The issue of high-validity models, balancing the trade-off between model accuracy and computational efficiency, also presents an important challenge in applications with real-time requirements.

Further, DTs demand continual validation to adapt to the evolving state of the actual system they represent. This requires robust techniques for verification, recalibration, and co-evolution of the models. Challenges in scalability and extensibility compound these difficulties, as DTs must operate across varying temporal, spatial, and dimensional scales while accommodating new functionalities.

Real-time decision-making, a core promise of DTs, introduces additional constraints, requiring computationally efficient solutions to meet strict timing requirements. Finally, as the environmental impact of computational technologies becomes a growing concern, the sustainability of DT systems has emerged as a critical challenge.

Through a detailed analysis of these challenges, this chapter highlights the critical research gaps and technical barriers that shape the DT domain. These challenges not only underline the complexity of implementing DTs but also set the stage for the novel solutions proposed in this thesis.

3.1 Digital Twin: A Revolution?

The adoption of DT technology brings drastic changes both in the engineering and the practice of simulation methods and infrastructures. For example, at the engineering level, the usability of DT is most pronounced when a real system or object undergoes modifications over time, making the initial model of the object obsolete [11]. While in traditional M&S a new model has to be built, the DT model rather captures and reflects these modifications. At the practical level, DT simulation experiments are not based on assumptions on the initial conditions like in traditional M&S, but on current information available from the system [112]. Consequently, the space of possible initial conditions to explore is larger with a traditional M&S model than with the DT model.

In this section, we first revisit major requirements for DT engineering. Then we discuss some of the disruptive challenges that these requirements bring in the M&S field.

3.1.1 Requirements for Digital Twin Engineering

To create a successful DT, it must fulfill a set of requirements. These include both reinforced forms of M&S requirements and novel DT context-specific requirements. In what follows, we discuss the most prevalent requirements mentioned in the literature. These requirements are typically interdependent, and each one can include one or several sub-requirements.

3.1.1.1 Data value chain

Data value chain covers the critical need for the DT to access appropriate data pertaining to its real-world counterpart at the appropriate moment. Digital models collect and analyse huge amounts of data throughout the entire life cycle [59]. There is a need for a well-defined data hierarchy because, at each abstraction level, DT provides data. Thus, the determination of handy information and data with considerable accuracy is vital [113]. Oliver identified an issue; for a DT, there is still a need for adequate and reliable data sources, and this problem has been the same for decades [60]. The identifiable information sources must retain information history and must be trustworthy, valuable, optimised, and available at any time for evaluating deterministic behavior, audit, and analytical purposes [114].

The International Telecommunication Union proposed a three-layered architecture of a digital twin network with a primary focus on three key subsystems: (1) unified data repository, (2) unified data models, and (3) digital twin entity management. The ITU also set out the requirements of a unified data repository as to be trustful and fast, it should provide a variety of data timely and accurately, be able to exchange real-time data within acceptable time delay, be easy to maintain, and be available at all times [115]. Reliability, integrity, and speed of data are crucial for system performance and for representing the state of a physical entity in real-time with an acceptable time delay because a system can be described based on its logic and input data [116].

Data value chain is the foundation of an end-to-end process responsible for collecting,

processing, and supplying data from various sources, which could be better utilised for decision-making and optimisation. Interoperability facilitates effective and efficient integration of the data across the different components of a DT. The next section highlights the interoperability requirements for a DT.

3.1.1.2 Interoperability

is the ability of two or more DTs to exchange information and mutually use the information that has been exchanged [117]. Interoperability allows DT to communicate and exchange data between different DTs, simulation models, software, and platforms such that the same data can be used for different purposes, which can be achieved by establishing equivalence between various model representations [118, 119].

Platform Interoperability is a DT's extension by using value-added services, such as AI, simulation, visualisation, etc. [120]. However, *System Interoperability* is communication and interaction between DTs of different physical entities [120].

In a practical approach, a few DT elements may already exist. In this case, there is no need to develop them as part of the new DT, but a need for a suitable interface to integrate the already existing DTs [113]. Integrating multiple simulations could be challenging. Open architectures and relevant interoperability standards can help to integrate different simulations at different fidelity levels [121]. The Digital Twin Consortium presented a complete DT interoperability framework based on seven key components: the system-centric approach, model-based interactions, holistic flow of information, state-based synchronisation, heterogeneously distributed federated repository integration, actionable information exchange, and scalable mechanisms to streamline connectivity and collaboration in DT ecosystems [114]. Interoperability standards such as ISO 23247-4 [57] and IEEE 1516 [122] can be used for the mutual interaction and integration of heterogeneous DTs. Table 3.1 shows the review of different literature and their main focus in the area of DT interoperability.

Interoperability sets out the foundations for effective collaboration and communication among the different DT components. However, this communication and collaboration need to be synchronised at a certain frequency and fidelity as defined by the Digital Twin consortium [54]. The next section addresses the synchronisation aspects of DT, and the subsequent section explores the fidelity requirements of DT.

3.1.1.3 Synchronisation

means that the state of the real system and its DT are kept consistent and up-to-date with each other's state using appropriate event-based or time-based methods [125, 57]. Depending on the data flow in the DT, synchronisation can be seen in both directions, from virtual to physical and physical to virtual [125]. In the former case, the synchronisation concerns the tracking of the physical world by the virtual one, in the latter it concerns the control enacted by the virtual world upon the physical one. The level of synchronisation (e.g., rate, quality, and volume) can vary and depends on the intended purpose [126]. Table 3.2 shows an overview of relevant literature and which synchronisation aspects are mentioned in their reviews/research. Most of the literature considers the bidirectional

Ref.	System Interoperability	Platform Interoperability	Integration
D.T.C. [114]	x	x	x
Lehner et al. [120]	x	x	x
Schleich et al. [118]	x		x
Durao et al. [119]	x		x
Wagner et al. [113]	x		
Shao et al. [121]	x		
Rasheed et al. [123]	x	x	x
Niaki et al. [124]		x	x

Table 3.1: DT interoperability literature considerations

dataflows, such as in Grieves' definition, but fewer also consider the data rate and data volume considerations and their effect on the synchronisation.

Synchronisation ensures that the real-time representation of the virtual entity is closely aligned with the physical entity, while fidelity ensures the accurate representation of the complexities of the physical entity. The following section highlights the DT fidelity requirements.

Ref.	Physical to Virtual Sync.	Virtual to Physical	Data Rate	Data Volume
ITU [115]	x	x	x	
Jones et al. [125]	x	x	x	
Liu et al. [127]	x	x	x	x
Moyne et al. [126]	x	x	x	
Rasheed et al. [123]	x	x	x	x
Sjarov et al. [128]	x	x		
Meng et al. [28]	x	x	x	x
Talkhestani et al. [129]	x			
Wagner et al. [113]	x	x		
Wang et al.[130]	x			
Zhang et al. [131]	x	x		

Table 3.2: Synchronisation considerations.

3.1.1.4 High validity

is vital for a DT to represent the current state of its physical counterpart as accurately as possible [119]. So, it is the high validity of DT that determines nearness to the real counterpart.

From the perspective of M&S, it can be argued that DT needs high-validity simulation modelling technology, where this validity is not only related to the validity of the model construction but also to the data-related issues, e.g., accuracy and frequency [131]. While

certain scholars discuss fully mirroring [129], ultra-realistic [128] and ultra-high-fidelity simulation [132, 133], others seek to establish fidelity levels to optimise the DT's advantage to existing challenges [134, 57, 61]. Academic definitions commonly imply that a high-fidelity model is a crucial requirement for a DT. However, practical use does not always require high fidelity, as it can be costlier. Therefore, it is crucial to determine the appropriate level of fidelity for DT [135]. Jones et al, 2020 [125] argue that DT fidelity levels comprise multiple dimensions, including the number of parameters, the precision of those parameters, and the degree of abstraction in the reciprocal exchange between the virtual and physical twins. The appropriate fidelity level is not necessarily the highest level of model fidelity feasible [61], and is dependent on the use-case. It seems that While DT fidelity pertains specifically to the suitable precision of the model's representation, DT validity encompasses the broader notion of whether the DT serves as a fitting and efficient tool for its intended application. The subsequent section discusses the Verification and validation aspects of DT.

3.1.1.5 Verification and validation (V&V)

of DTs pertain to the assurance of constructing a DT in alignment with its objectives (validation) and ensuring its accurate implementation (verification) [136]. DTs typically include different models, components, sub-components, and processes, which necessitates V&V on an individual basis as well as for the entire system [61, 118]. To maintain the model's validity throughout the entire life cycle; clear and well-defined guidelines are needed for V&V [121]. A DT must have a validated specification of what to simulate and what to predict, with which input, and which approach [113]. The system's output should be continuously compared and monitored with a reference point to detect errors and anomalies [123].

Primarily DT concept applies to highly automated systems. When there is human involvement and decision-making, then due to the high degree of randomness of human actions, the system cannot be perfectly shadowed virtually [60], thus directly affecting the V&V of DT.

Hua et al. [137] argue that we may need a two-layer approach for V&V of DT, one at the system level and the other at the constituent system level. In general, due to the dynamic nature of DTs, V&V should be a continuous process (either online or offline) that needs to be performed periodically (or on-demand) [59]. Sargent stated that V&V of a basic simulation model comprises four pillars: (1) Conceptual Model Verification, (2) Computerised Model Validation, (3) Operational Validation, and (4) Data Validation [138]. Lugaresi et al. [59] presented the DT V&V approach based on the following four levels: (1) Logic-level Validation (Digital Model), (2) Input-level Validation (Input Data), (3) Event-level validation (System Events), and (4) Performance-level Validation (KPIs). Table 3.3 shows the comparison between different literature and their considerations for the V&V of DT.

For a well-established and accurate system, there could be a need for system improvement, reduction, or extension depending on the requirements. These modifications could be at the core architectural level of the system, or either at the output or input level of the system. These are categorised as extensibility and scalability; the successive sections address these issues.

Ref.	Online DT V&V	Offline DT V&V	Data V&V	General System V&V
ITU [115]			x	
Lehner et al. [120]			x	x
Sargent [138]				x
Lugaresi et al. [59]	x			
Khan et al. [139]	x	x		x
Peter et al. [60]				x
Dahmen et al. [140]				x
Hua et al. [137]	x	x		x
Shao et al. [121]				x
Wagner et al. [113]			x	x
Locklin et al. [141]	x	x	x	x
Rasheed et al. [123]			x	
Schleich et al. [118]				x

Table 3.3: DT V&V literature considerations.

3.1.1.6 Extensibility

refers to the DT's capability to integrate, add, or replace models [118, 119], that allows a DT to expand or enhance easily. Extensibility allows DT to accommodate new applications and functionalities without significant effort. The evolution of DT must be aligned with its physical counterpart [120], while maintaining its backward compatibility [123, 115]. The DT functionalities should smoothly extend their capabilities with no effect on existing functions [126, 115]. Extensibility requirements can vary and may encompass subordinate requirements, notably modularity and standardisation.

Extensibility can vary from small sensor integration to a whole new model integration within the already existing DT ecosystem. Thus, it is an important aspect to focus on at the time of designing a DT architecture. It could directly affect the cost and other related aspects. When a system evolves over time, it has a direct effect on its parameters and data, which gives rise to the problems of scalability. The DT scalability is described in the following section.

3.1.1.7 Scalability

refers to a DT's capability to show the state of its real counterpart at different dimensional, temporal and spatial scales (microscopic scale - fine detail, mesoscopic scale - medium detail, and macroscopic scale) [118, 119]. Processing data at different levels of granularity contributes a lot to the holistic understanding of the modeled entity. Multiscale simulation has been recognised as one of the most important visions of DTs [127]. To enhance the scalability, models should support different dimensions, spatial, and time scales [131]. DT must be capable of automatically adjusting the scale of the virtual twin regarding the growth or shrink of its physical entity [115]. Like extensibility, scalability could also be sometimes a constraint and could limit the reduction, modification, or extension of a system to a step above or below. Some systems change constantly, few

grow rapidly, and others evolve over time, thus, they have fuzzy borders [142].

In addition to the aforementioned requirements, DT needs to be explainable to the user. Explainability enhances user trust. It helps to describe a system and how it processes data and makes conclusion and predictions.

3.1.1.8 Explainability

is an ability that aims to provide insight into how a DT can be understood by the user entity. According to ISO 23247 [143], the user can be human, applications, or other systems that use the DT. When a human being is a DT user, visual representation is essential to provide them with comprehensible outputs [144]. The International Telecommunication Union (ITU) emphasises that all the elements of DT, such as data and models, should be developed by means of visualisation to provide better access for involved humans [115]. Commonly, the M&S community focuses more on model explainability, which facilitates the collaboration and interaction between models and users to provide a consistent understanding [130]. Table 3.4 provides a brief overview of the relevant literature, highlighting the explainability considerations in terms of interaction [A], comprehensiveness [B], semantics [C], intelligence [D], and abnormal data [E] discussed in their respective studies.

Ref.	A	B	C	D	E
Zhu et al. [144]	x	x		x	
Shao et al. [60]		x			
ITU [115]	x				
Zhang et al. [131]	x			x	x
Sjarov [128]					
Wang [130]		x			
Wagner et al. [113]		x	x		
Rasheed et al. [123]	x				
Sjarov et al. [128]		x	x	x	

Table 3.4: Explainability Considerations.

3.1.2 Disruptive challenges

The requirements for DT engineering previously presented bring with them a set of open research challenges. We discuss those that from an M&S perspective appear as potentially disruptive.

3.1.2.1 Dynamic State Estimation

State estimation is the challenge of determining the actual state of the system in operation. *Dynamic state estimation* refers to the process of determining how a system is operating when its state is changing dynamically based on real-time observation data. Dynamic

state estimation is required because, in most circumstances, noisy and incomplete observation data from dynamic systems make it impossible to derive the system state directly from the observation data.

One of the key features of the DT is the integration of real-time data with a digital model to support real-time prediction/analysis of the system under study (similarly to RtS [37, 41, 145, 42, 43, 44]). Most of the DTs model the dynamic behavior of the corresponding physical systems. Therefore, they need the dynamic state estimation.

To enable simulation-based real-time prediction/analysis, Hu proposes a framework of data assimilation for dynamic systems in operation [146]. The goal of the framework is to support real-time decision-making for the system under study. To accomplish this, simulation-based future behavior prediction and analysis of the system are required. The simulation-based prediction and analysis depends on a precise evaluation of the system under study's current state in real-time, which asks for dynamic state estimation, therefore, a simulation-based prediction can be used. Moreover, to accurately characterise the system in operation through simulation-based prediction and analysis, the simulation model becomes essential. This requirement calls for online model calibration of the model parameters based on real-time data gathered from the system. The data assimilation approach addresses both dynamic state estimation and online model calibration activities by merging information from real-time data and the simulation model.

Data assimilation has been recently used for discrete-event and discrete-time systems, including agent-based models. The particle filter (PF) approach is frequently a viable choice for stochastic simulation models of discrete systems due to its non-linearity and non-Gaussianity [147]. But it is computationally expensive because of the probability distributions of model runs. In their findings, they observe that the choice of time intervals, rather than the number of particles, more strongly influences the estimation accuracy of such a system utilising PF. When measurement errors are underestimated, state estimates are poorer than when measurement errors are overestimated. Better state estimations are not a given just because one has a proper understanding of the measurement errors. In addition, over-estimation of errors yields better state estimation and is more sensitive to rapid system changes.

3.1.2.2 Online and Continual Validation

Continual validation is the process of continually ensuring a DT, or more concretely the model(s) in the DT, remains a valid representation of their real-world counterpart. When the model becomes invalid, e.g., due to changes in the real-world system, a recalibration is needed to match the DT to the real world once again, as was shown in stage 2.9 in Figure 2.5. This is conceptually different from model validation in M&S where, typically, a calibration attempt is performed first, after which the model is subjected to the validation procedure. Once positively validated, the model is generally also assumed "finished". Besides this conceptual difference, there is a high level of similarity between traditional model validation and the validation of models in DTs. As such, the model validation techniques from M&S can largely be carried over [148]. However, one faces several challenges when continually attempting to apply those model validation techniques. They stem from the fact that only the runtime data of the system in operation is available. This leads to the following challenges:

- In the traditional model validation, a validation experiment is carefully defined. With DTs, however, data is streamed continuously from the physical system. Therefore, one must define which part of this data stream can be considered an experiment for validation. Stated differently, you need to delineate the experiment in the data stream.
- In traditional model validation, experiment replications are controlled by the experimenter. With DT data, we are relegated to grouping or batching data from equal "experiments" (which have been delineated as stated previously). In this batching procedure, it is important to choose a proper time horizon, e.g., when using data from the last month, there is a risk of averaging out any of the changes that we would want to observe and check against.
- In a traditional validation experiment, the bounds/ranges are carefully controlled. With a DT, we are limited to the bounds that occur naturally from the system's routine/regular operation. A problem with these bounds is that they are usually only a subset of the entire range for which the utilised simulation models were validated at design time. As such, the range you can continuously validate against is limited. A potential workaround for this problem is that of "experimental runs" where we instruct the system to perform an experimental execution, which must still achieve the regular goal but in such a way that it yields additional information content [149].

Continually validating a model has its benefits; the model can be continually checked for correctness. However, the available data is not as information-rich as those gained from a specifically crafted model validation experiment. For physics-based models, traditional literature on this topic, therefore, ought to be reviewed [3, 150, 63].

In the case of DTs of production facilities/smart manufacturing, the used models are often discrete-event queuing models, with arrival times and processing times characterised by stochastic distributions. In such cases, perhaps new validation techniques are needed. In literature, different metrics are calculated on periods in the data streams, combined with thresholding to trigger model updating [151, 59, 152]. The use of stochastic simulation also adds another caveat, which is that both the arrival/input distribution and the logic/model itself could become invalid, and a good validation ought to be able to pinpoint exactly which of these two has changed [151].

This idea of pinpointing errors at runtime leads us to another mature field where methods could be found to aid in this challenge of continual validation: the field of fault detection and diagnosis. Recall that the goal is to detect divergence between the model in the DT and the twinned system, with a recalibration in case of divergence. This is conceptually not that different from fault detection and diagnosis, where the goal is to detect faults in a physical system, diagnose/isolate them and take corrective action based on their identification [153], the main difference being that with the DT, we assume the real-world system is the ground truth and any faults occurring in it should propagate back to the digital model. Not all techniques from this field carry over, e.g., physical redundancy techniques cannot be applied to DTs. Still, model-free techniques such as trace inspection with limit checking [154] or model-based techniques do carry over. Model-free techniques operate on the traces of the model and combine the filtering of those traces with limit checking to detect faults, similar to Lugaresi et al.'s work [151].

Model-based methods use a digital model that produces traces in parallel to the physical system, and through trace comparison, faults can be detected.

Furthermore, these techniques find their application for continuous [153], hybrid [155] and discrete-event [156, 157] systems. Normally, these techniques were also designed to operate at system runtime; as such, they are generally not particularly heavy on the computational side, making them useable for real-time applications.

The previously discussed validation techniques rely on the transmission of data from the physical twin to the virtual space. It's therefore also paramount that this data is flawless. With this in mind, we can state that not only should we validate the models in the digital twin, we ought to also validate the data as it arrives at the digital twin. The initial culprit to blame for faulty data would be the sensors on the physical system that collect the data. Therefore, any digital twin system could benefit from sensor fault detection and isolation [158, 159, 160]. In fault detection and isolation, the sensor faults are generally classified as incipient or abrupt failures. In an incipient failure, the sensor is working in an abnormal or deteriorated way. In an abrupt failure, the sensor suddenly stops working. Various diagnosis methods can be applied to detect and isolate the fault, such as model-based, knowledge-based, and deep learning based approaches. In [160], a set of machine learning techniques are applied for the sensor fault detection in a digital twin specifically. While a likely culprit for faults, the sensor is not the only place where faults can be introduced, so is the communication network, and any layers of software that perform processing/packing/unpacking of data [159]. As such, perhaps the data validation should be performed right before it is fed into the digital twin. We see this idea applied in the field of machine learning, where data is aggregated from multiple sources before being fed into a machine learner [161]. These techniques should be integrated into the digital twin's data pipeline, for the digital twin to work optimally, but also for the continual validation to work correctly.

In summary, more work is needed to get to continual validation, but because of the similarity to stage 1.5 in the M&S workflow of Figure 3.1, there exist techniques that can help us along. We also make one note regarding the use of continual over the generally accepted continuous (such as in continuous testing, integration, and deployment). Nowadays, continuous has the implication that it is an everlasting process, whereas continual implies some periodicity in the process but with pauses. We think that this is a more correct representation of how the process is to be implemented, which is why we opt for continual.

3.1.2.3 Automated Recalibration and Co-evolution

When a model no longer accurately represents the behavior of the real-world system it models, changes must be made to that model. Three scenarios are possible: (a) either the parameters of the model no longer accurately reflect the system and its environment, in which case a parameter calibration within the model's range of validity suffices; (b) the system is no longer within the valid context of the model, in which case a new model must be selected/developed that again fits the system's current context; (c) in cases where inaccuracies stem from a lack of precise initial conditions, the issue resembles a state estimation problem. Rather than recalibrating parameters or replacing the model entirely, estimation techniques such as Bayesian inference or Kalman filtering can be

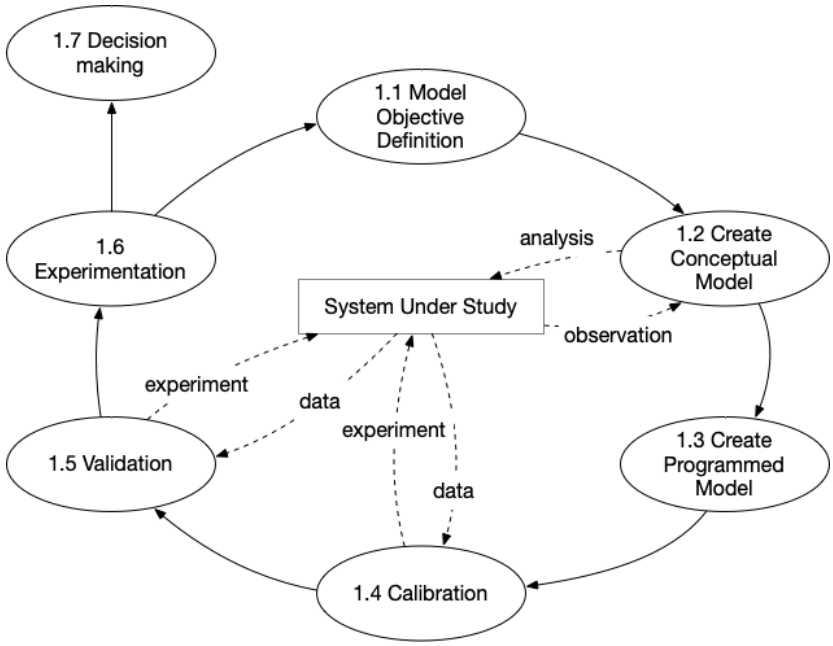

Figure 3.1: Simplified M&S workflow

used to infer the missing initial state, ensuring synchronisation without unnecessary reinitialisation. In either case, reinitialisation of the model is necessary. In [162], this problem is described along with an accompanying workflow, but no solution is given for the reinitialisation step that brings the model back in synchronisation with the real system.

One way to deal with the second scenario is to use meta-information about the validity of a model in a certain context. In literature, Zeigler [64] defined the concept of the experimental frame as *"The conditions under which the system is observed and experimented with"*. The idea of the experimental frame is to make the contextual information about the simulation model explicit. In doing so, it gains a dual purpose: it implements meta-data that is needed to specify the range of validity, and it defines an operational view of the experiment using a generator, transducer, and acceptor. The concept was further refined by Traore and Muzy [163]. The experimental frame could be used to check if the context of the model is still valid. Denil et al. [164] looked at the uses of such an experimental frame: checking for a new context, calibration, searching for a model in a library of simulation models, reproducibility, etc., and concluded that it might not be defined well enough for these purposes. In the context of DTs, no processes are defined to allow for automated calibration experiments. Validity frames [164, 99, 165] evolve the concepts defined in the experimental frame (the experimental frame is embedded within). It has the meta-data needed to reason over the model and run simulations (such as initial conditions, parameter ranges, model architecture and rationale, etc.) and the operational view where signal monitors are generated to check the model's bounds at run-time. It also adds workflows for different activities within the M&S process, such as calibration and validation. This could be used as a starting point for checking if the model could be recalibrated, if a new model should be selected from the model library,

or if a new model should be developed.

Similar methods and tools have been created in the co-simulation community that can be used as a starting point: Otter et al. [166] propose to annotate the parameters of a Modelica model with traceability, uncertainty, and calibration information to improve model quality, thus increasing correct use of models. Instead of relying on external data formats, they insert this machine-readable metadata within the models. The Modelica association also developed a standard for creating co-simulation packages: System Structure and Parameterisation (SSP) [167]. The SSP could be extended to allow for structure verification, parameter verification and boundary adequacy testing.

Another source of inspiration is the control community. For example, the MAPE-K loop which is a high-level feedback control loop from IBM [5] for self-adaptive systems, has been integrated in DTs [168]. This approach is based on the principle of changing the DT model when an anomaly is detected. The MAPE-K architecture makes a distinction between the domain-specific system, the managed system, and the system manager. The managing system contains four phases that use common knowledge to: (a) monitor the managing system and its context, (b) analyse the situation and decide if adaptations are required, (c) plan the adaption to this new configuration and (d) execute the transition to this new configuration using a mode-changing protocol. In this situation, the change of model happens when an anomaly is detected; the new model requires calibration and the controller needs re-optimisation.

However, automatically creating a new model seems to be a difficult problem. One way to deal with this is by searching for alternative models based on the current model. David et al. use reinforcement learning techniques for this specific purpose [169]. Another approach is that of the control community, which has long worked on the system identification problem. In system identification, statistical methods are used to create a black-box model of the system [170]. These techniques can be integrated into the DT if a new white box model needs to be created.

3.1.2.4 Real-time adaptive operations

Real-time aspects pertain to the fact that there may be some factor of timeliness required from the data used in the DT. The stringency of this requirement depends on the goals of the DT. When the DT acts in some form of process control, soft or hard real-time constraints may be required. Otherwise, the non-strict, human, notion of real-time suffices [14].

Some real-time aspects require investigation in the DT concept:

The availability of quantitative data and advanced analytics in real-time via DT enables better-informed and faster decision-making. In particular situations, decision-making processes must adhere to real-time constraints. As a result, the DT model should be sufficiently fast to make decisions within the specified timescale while also accurate enough [11]. The computational cost of using a more complex model is often excessively expensive, and the system might fail to meet the deadline. There are several approaches to dealing with computationally expensive models:

- Multi-resolution modelling (MRM) is the process of creating a single model, a family of models, or both to represent the same phenomenon at several resolution levels while allowing users to enter parameters at each level according to their needs [171]. MRM is also known as variable- or selectable-resolution modelling. Sometimes the word fidelity is used instead of resolution. MRM is closely related to model abstraction, which is a way of simplifying models while keeping the essence of a phenomenon concerning the application at hand [172].
- Franceschini et al. [173] present an adaptive abstraction approach. They utilise a specified trigger to determine when to transition between abstraction levels. A dynamic abstraction simulation that alternates between an agent-based formalism and a discrete event formalism is presented by the author in prior work. The statistical analysis of the observed emergent behavior serves as the basis for the decision to change abstraction levels. A more rigorous framework [174] extends the adaptive abstraction technique to decide when and where to switch between abstraction levels.
- Some research recommends employing an abstracted and/or approximated model instead of a more detailed model. The *self-Adaptive Abstraction and Approximation* technique [24] is based on the MAPE-K loop previously presented to adapt a real-time system under study by changing the model and using an approximated and/or abstracted model instead of the more detailed model. However, the validation of the substitute model is an essential issue that needs more investigation. One approach is to look at the model behavior, calculate the deviation, and find tolerances [175, 25]. Another approach is using the ESS (EMF-Based Simulation Specification) technique [176].

Real-time communication is another challenging part of the real-time aspect. The communication rate between the real world and the system is something to consider, as it is not feasible to communicate at every microsecond.

3.1.2.5 Sustainability

This relates to the observation that DTs use large amounts of computational resources to provide their different services. Computing as an industry is currently responsible for 2% [177] to 6% [178] of the emissions of greenhouse gasses globally, with a predicted share of 6% [179] (22% [178]) in 2040 (2030), therefore we must require future DTs to be sustainable in regard to energy consumption. Reasoning over energy and power consumption and their associated models can include several levels of impact [180]:

- First-order impacts: Impact via the design and operation of the DT.
- Second-order impacts: Secondary impact related to the effect of DTs on, e.g., production and product usage. For example, the decrease in energy consumption of a device because of the optimisation possible by the digital twin.
- Third-order impacts: indirect effects caused by DTs, e.g., impacting an industry's structure or the lifestyle of persons.

To create sustainable DTs, all the above aspects should be considered. Bellis et al. propose an additive model where the consumption of the energy occurs [181]:

$$E_{total} = E_{design} + E_{local} + E_{network} + E_{cloud} + E_{update} \tag{3.1}$$

where:

- E_{design} is the energy consumed for creating the digital twin. Building a simulation model of a digital twin might not greatly impact this factor. However, this term might have a significant impact when using data-driven methods.
- E_{local} is the energy consumption at the analogue side of the system (e.g., by storing the data, pre-processing the data, and executing a part of the digital twin model locally).
- $E_{network}$ is the system's energy consumption by sending and receiving messages on the network.
- E_{cloud} is the energy consumption by executing the digital twin in the cloud environment.
- E_{update} is the energy necessary to redesign and update the model during the system's life cycle.

For the construction of DTs, each of the different sources of energy loss should be further examined. modelling and simulating the full infrastructure and the design process seems a logical first step.

3.2 Conclusion

In this chapter, we have explored the number of challenges that arise when developing and deploying DT systems. From ensuring reliable and timely data pipelines to achieving seamless interoperability across diverse domains, these challenges underscore the complexity of integrating DTs into real-world applications. The need for continuous synchronisation between physical and digital entities, combined with the trade-offs between model validity and computational efficiency, further highlights the technical challenges that must be overcome.

Additionally, preserving the validity of DT models through continual validation and automated recalibration presents critical challenges as systems evolve. Scalability and extensibility complicate matters by requiring DTs to adapt to changing contexts and enhance their functionality. Real-time constraints, essential for timely decision-making, require novel solutions to manage computational constraints while preserving responsiveness. Finally, the sustainability of DT systems remains a remaining concern, emphasising the need for energy-efficient and environmentally conscious design. These challenges not only illustrate the technical complexity of DTs but also establish the foundation for the solutions presented in the subsequent chapters, highlighting their importance in progressing the state of the art.

Chapter 4

Decision-Centric Technique

This chapter is based on the following publications:

R. Biglari and J. Denil, "Self-adaptive abstraction and approximation (sacube) framework for digital twins with real-time requirements," *SIMULATION*, 2025, under peer review.

R. Biglari, J. Mertens, and J. Denil, "Towards Real-time Adaptive Approximation," in *2022 11th European Congress Embedded Real Time Systems - ERTS 2022*, Toulouse, France, Jun. 2022.

R. Biglari and J. Denil, "Model validity and tolerance quantification for real-time adaptive approximation," in *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, ser. MODELS '22, 2022.

A critical aspect of using a model effectively is understanding the regions where it remains valid. A model is a representation of a real-world system, designed to operate within specific boundaries and under certain assumptions. These boundaries define the region within which the model can reliably predict outcomes or simulate behaviors. Even when a model performs well within its defined region, applying it beyond those regions introduces significant risks, as its accuracy and reliability are no longer guaranteed. This misapplication can invalidate the model and lead to incorrect results or decisions. For example, a physical model may assume linear behavior within a limited range of forces or temperatures, while a machine learning model may depend on the statistical properties of the training data. When these constraints are violated—for example, by employing the model with extreme input values or in a different context—the model is likely to fail, either by producing incorrect results or becoming entirely unreliable.

Sometimes, this information is already available in the form of validity frames [98, 99]. Validity frames provide predefined constraints, often obtained from domain knowledge, actual facts, or theoretical derivations.

However, in some cases, the boundaries of a model's validity are not well-documented or fully understood, leaving users without a clear guide for its appropriate application. This lack of information can lead to significant challenges. Without clearly defined

validity frames, users risk using the model in unsuitable contexts, leading to reduced performance and potentially critical errors in decision-making processes.

Addressing the challenge of undefined model validity and an unavailable validity frame is a critical issue in model development. In this chapter, we present a new technique for overcoming this issue.

4.1 Conceptual Framework

We start our conceptualisation inspired by the conceptual framework proposed by Barroca et al. [182] depicted in Figure 4.1. In our proposed conceptual framework in Figure 4.1, we extended the conceptualisation with two different models with different approximations and the layer of the decision-making algorithms. We look into the impact of model approximation and/or abstraction on cost reduction, which forms the foundation of the framework that facilitates system runtime adaptation.

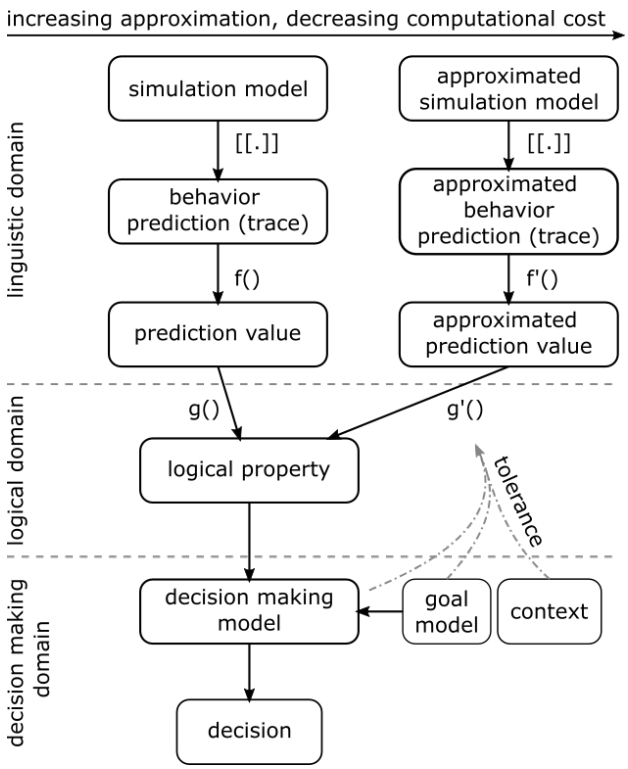

Figure 4.1: Conceptual framework.

The semantics of a simulation model are determined by simulating the model and using a semantic mapping function ([[.]]) that simulates the model and links it with its intended meaning. The simulation results in a collection of traces. While there are a multitude of simulation traces, not all of the provided information is the quantity of interest of

the decision-making algorithm or for reasoning over the logical behavior of the system. We need a function $f()$ to extract properties of interest. Then function $g()$ is used to transform between the quantity of interest and the logical property. A logical property is a Boolean value and is on the ontological level where it gets a real-world meaning. The framework is similar for an approximated/abstracted model and reasons on the same logical property. This means that the function $g'()$ should give an equivalent result to the more detailed model, although with more uncertainty [24]. As shown in Figure 4.1, simulation models produce behaviour traces that influence decision-making through a chain of transformations. When using multiple models of different approximations, our key insight is that model validity can be determined based on decision consistency rather than output similarity.

Traditional validation approaches [3] compare model outputs directly, which may be unnecessarily restrictive. Instead, we propose focusing on whether different models lead to equivalent decisions, even if their detailed outputs differ. This aligns with our practical goal: maintaining reliable decision-making while reducing computational costs.

4.2 Decision-Centric Technique

In this section, we propose decision-centric technique for establishing model validity when pre-defined validity boundaries are unavailable.

Let m_h represent the model with the most detail, and m_s the surrogate model under consideration. Let D be the decision maker that maps model outputs to decisions $y \in \mathcal{Y}$, where $\mathcal{Y}$ is the decision space. We define a distance metric d_Y on the decision space:

$$d_Y : \mathcal{Y} \times \mathcal{Y} \to [0, \infty) \tag{4.1}$$

Since the decision space could be a numerical or categorical (binary/multiple categories/etc.) space, for a given tolerance $\varepsilon > 0$, we define the validity region $\mathcal{V}_\varepsilon$ for numerical decision spaces as:

$$\mathcal{V}_\varepsilon = \{x \in \mathcal{F} \mid d_Y(D(m_h(x)), D(m_s(x))) < \varepsilon\} \tag{4.2}$$

And for categorical decision spaces, we define the validity region as:

$$\mathcal{V}_\varepsilon = \{x \in \mathcal{F} \mid D(m_h(x)) = D(m_s(x))\} \tag{4.3}$$

Under the assumption that $\mathcal{V}_\varepsilon$ is continuous, its boundary $\mathcal{B}$ can be characterised as:

$$\mathcal{B} = \{x \in \mathcal{X} \mid \forall \delta > 0, \exists x_1, x_2 \in N(x, \delta) \text{ where } x_1 \in \mathcal{V}_\varepsilon \text{ and } x_2 \notin \mathcal{V}_\varepsilon\} \tag{4.4}$$

where $N(x, \delta)$ represents a ball of radius δ around x. In the rest of the study, we will remove $_\epsilon$ from the notation and implicitly assume that there can be a tolerance associated with the validity region.

This continuity assumption enables efficient search strategies for identifying the validity boundary. One such algorithm is a simple binary search algorithm that searches for the boundary with a single parameter. However, as most problems are multi-dimensional, this approach does not scale well, and other more feasible algorithms should be employed. Although this topic is related, it is not central to the focus of this thesis and, therefore, lies beyond the scope of this research.

Algorithm 1 FindBoundary

```
function FindBoundary(p1, p2, tolerance)                ▷ p1 ∈ V_ε, p2 ∉ V_ε
    while ||p1 − p2|| > tolerance do
        mid ← (p1 + p2)/2
        if mid ∈ V_ε then
            p1 ← mid
        else
            p2 ← mid
        end if
    end while
return p1
end function
```

Symbolic Reasoning enables us to leverage domain knowledge and previous experimental results to constrain the search space for valid model regions, e.g., knowledge about physical laws allows us to infer validity regions from individual experimental points. For example, Cederbladh et al. introduce the following example on the braking of a machine: When a test shows that a machine cannot brake safely under certain conditions (e.g., 15220kg at 19 degrees incline), we can deduce that heavier machines or steeper inclines will also fail, without requiring additional experiments. Similarly, successful test cases (like 22330kg at 6 degrees) let us infer validity for lighter machines at lower inclines [183]. This systematic approach to knowledge reuse aligns with case-based reasoning principles while incorporating domain-specific constraints to efficiently bound the feasible region. This technique can easily be integrated into our approach. Furthermore, if a partial validity frame with extra constraints is already available we can integrate these constraints as well.

The search space can be constrained using domain knowledge. Let $C = \{c_1, c_2, ..., c_n\}$ be a set of domain constraints, where each $c_i : \mathcal{X} \to \{\text{true}, \text{false}\}$. The feasible region $\mathcal{F}$ is then:

$$\mathcal{F} = \{x \in \mathcal{X} \mid \forall c \in C, c(x) = \text{true}\} \tag{4.5}$$

where $\mathcal{X}$ is the state space. After defining the feasible region $\mathcal{F}$ based on domain constraints, all validity frame experiments and boundary searching are restricted to points within $\mathcal{F}$. This ensures we only explore physically meaningful states and avoid wasting

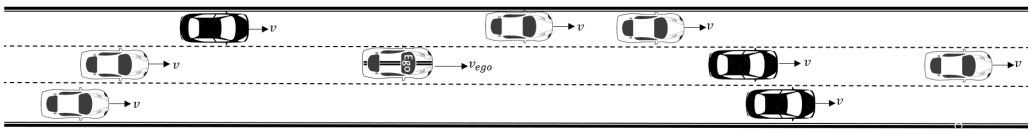

Figure 4.2: Lane changing scenario.

computational resources on testing invalid configurations. This approach's key advantage is enabling efficient discovery of validity boundaries through search algorithms guided by domain knowledge constraints.

4.3 Motivating Example

To represent the challenges in our research, we use a lane change control algorithm as our motivating example. Highway lane change maneuver system models control the longitudinal and lateral direction of the ego car (target car).

Figure 4.2 shows a scenario where the front car decelerates slightly, causing the ego car to change lanes. We utilise a Simulink model to simulate this scenario.

To predict the next position of the front car, we look at the front car's next position. In this experiment, we use two different models.

- High-validity model, the most computationally expensive model in this experiment, predicts the trajectories of the current ego vehicle in response to the surrounding vehicles. By incorporating a controller for each vehicle, the model enables a more granular and detailed evaluation of individual car trajectories.

  To account for the impact of position changes on trajectory predictions, the model operates under a fixed-point approach, ensuring the emergence of a stable system state. Importantly, the validity of the high-validity model is constrained to the validation domain of the system, aligning its validity frame with the validation domain of the system.

- Constant Acceleration model (C.A model) is the kinematic equation $x(t) = 1/2 * a * t^2 + v * t + x_0$. In this formula, a is the vehicle's acceleration in m/s^2, v, the velocity of the vehicle in m/s, and x, the vehicle's position in meters.

In this experiment, $\mathcal{M}$ is our set of models, where m_h is the high-validity model, m_{ca} is the C.A model. $\mathcal{M} = (m_h, m_{ca})$

The validity of m_h is equal to its validation domain shown in Figure 4.3. However, the model validity is not pre-defined and available for m_{ca}. Therefore, we use the decision-centric technique to find out the model validity.

To determine the decision-based model validity, we need to consider all possible situations. Therefore, in the highway lane change system with three lanes, we define the

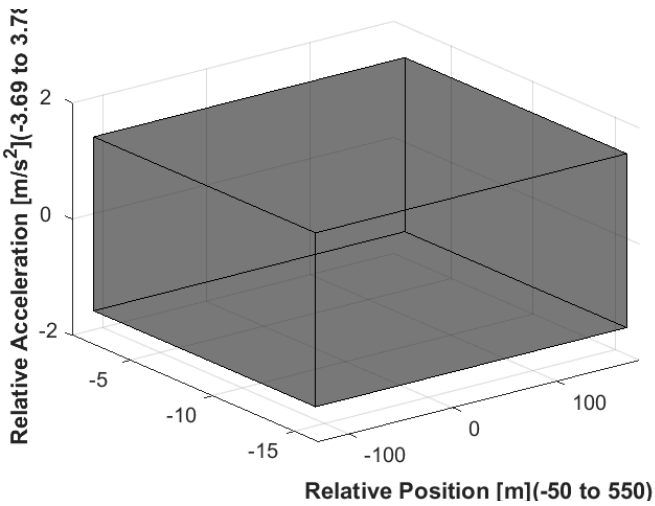

Figure 4.3: Validity region of the high-validity model.

scenario depicted in Figure 4.4. It is possible to achieve this by putting six cars surrounding the ego car. The number of lanes, which in this instance is three, is directly proportional to the number of cars that are in the area around them.

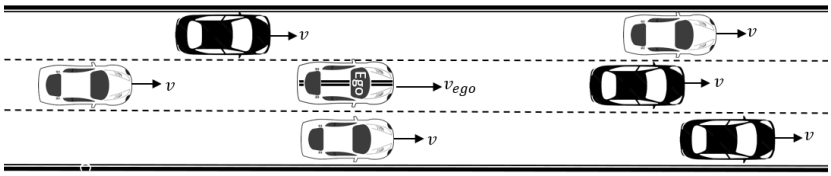

Figure 4.4: Finding model validity scenario.

To find the validity region $\mathcal{V}$ of m_{ca}, we use algorithm 2.

We check domain constraints C as the following set:
c_1: Vehicles maintain deterministic behaviors during the simulation. And in reality, each car might have unknown situations and different behaviour.
c_2: The minimum car speed has to be 6 m/s.
c_3: Each target vehicle, after acceleration or deceleration, continues with the same velocity.
c_4: The front gap, the gap between a car and the front car, has to be at least 30 m. And the rear Safety Gap must also be 30 m.
c_5: The ego car has no acceleration, and the speed is constant.
c_6: When the car is in front of the ego car, the model is valid in positions further away.
c_7: When the car is behind the ego car , the model is valid at positions farther away.

Additionally, for each surrounding vehicle, we determine the boundaries, denoted as $\mathcal{B}$, through a binary search process. This involves evaluating various combinations of positions, velocities, and accelerations using the following method:

For each surrounding vehicle, we begin by varying its relative position (with respect to

Algorithm 2 ValidityRegionSearch

Require:
$bounds_p = [p_{min}, p_{max}]$ ▷ position bounds
$bounds_v = [v_{min}, v_{max}]$ ▷ velocity bounds
$bounds_a = [a_{min}, a_{max}]$ ▷ acceleration bounds
Ensure: ValidityRegion VR
procedure FindValidityBoundaries
  $VR \leftarrow \emptyset$
  for all car c in surrounding_cars **do**
    $p_{boundary} \leftarrow$ BinarySearch($bounds_p$, matchesDecision)
    if $c.position$ is in front **then**
      $valid\_p \leftarrow [p_{boundary}, p_{max}]$ ▷ Valid when further away
    else
      $valid\_p \leftarrow [p_{min}, p_{boundary}]$ ▷ Valid when further away
    end if
    for all p in discretise($valid\_p$) **do**
      $v_{boundary} \leftarrow$ BinarySearch($bounds_v$, matchesDecision(p))
      if $c.velocity$ is positive relative **then**
        $valid\_v \leftarrow [v_{boundary}, v_{max}]$ ▷ Valid at higher relative speeds
      else
        $valid\_v \leftarrow [v_{min}, v_{boundary}]$
      end if
      for all v in discretise($valid\_v$) **do**
        $a_{boundary} \leftarrow$ BinarySearch($bounds_a$, matchesDecision(p, v))
        if $c.acceleration$ is positive **then**
          $valid\_a \leftarrow [a_{boundary}, a_{max}]$ ▷ Valid at higher accelerations
        else
          $valid\_a \leftarrow [a_{min}, a_{boundary}]$
        end if
        $VR \leftarrow VR \cup \{(p, v, a) | a \in valid\_a\}$
      end for
    end for
  end for
  return VR
end procedure
procedure BinarySearch(bounds, decisionCheck)
  $left \leftarrow bounds.min$
  $right \leftarrow bounds.max$
  while $right - left > \delta$ **do** ▷ Small delta for numerical stability
    $mid \leftarrow (left + right)/2$
    if decisionCheck(mid) **then**
      $right \leftarrow mid$
    else
      $left \leftarrow mid$
    end if
  end while
  return mid
end procedure

the ego car) using a binary search in Algoritm 1. We then run simulations using both m_{ca} and m_h, comparing their outputs to evaluate the predicted trajectories of the ego car. The trajectories do not need to align point-for-point; instead, we define a distance metric d_Y to determine divergence. Specifically, we consider two trajectories as different if they result in a lane change by the ego car. If the trajectories are deemed identical according to d_Y, we identify that relative position as the position boundary for the vehicle.

However, identifying the position boundary is not sufficient to establish a valid point within the model. We must also account for whether the vehicle is in front of or behind the ego car. If the vehicle is ahead of the ego car, the feasible region $\mathcal{F}$ extends from the position boundary to the sensor's maximum range (e.g., the end of the highway). Conversely, if the vehicle is behind the ego car, $\mathcal{F}$ is defined as the region between the position boundary and zero.

Next, we determine valid velocities. For each discretised position within $\mathcal{F}$, we explore different velocities using binary search. If the decisions from m_{ca} and m_h align, the point is added to $\mathcal{B}$. This process mirrors the steps used to determine position boundaries. Similarly, we apply the same approach to identify valid accelerations by iterating through all combinations of positions and velocities.

Therefore, we come up with the following results. The resulting validity region for m_{ca} is illustrated in Figure 4.5.

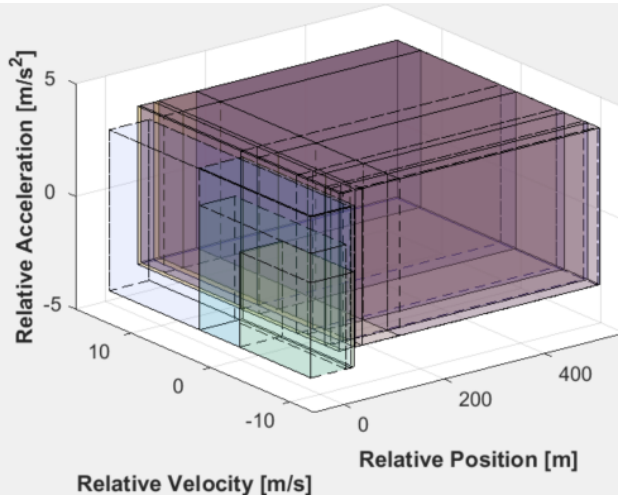

Figure 4.5: Constant Acceleration (C.A) model validity.

4.4 Conclusion

In this chapter, we look at the significance of the model's validity, which is equally as crucial as the model itself. To allow this, we propose a decision-centric technique to find the validity of a model in the context of a decision-maker.

Decision-centric is a technique to assess a model's validity. Rather than focusing merely

on output similarity, this technique assesses a model's applicability within the context of the decision-making process.

We acknowledge that a single case study may not provide sufficient validation for broader generalisation, despite the fact that the proposed technique demonstrates promising results in the presented case study. Nevertheless, we are in the process of researching a follow-up study as part of an upcoming paper [184]. The objective of this study is to apply the technique to a distinct context, specifically an autoclave system case study. It was not possible to incorporate this additional validation into the scope of this thesis due to time constraints.

Chapter 5

SACube Framework

This chapter is based on the following publication:

R. Biglari and J. Denil, "Self-adaptive abstraction and approximation (sacube) framework for digital twins with real-time requirements," *SIMULATION*, 2025, under peer review.

As DT systems become increasingly integral to real-time decision-making and system optimisation, their computational demands continue to rise. High-resolution models provide accurate representations of dynamic environments but often come at the cost of significant computational overhead, making real-time execution challenging. Addressing this issue requires a balance between model resolution and computational efficiency, ensuring that decision-making remains both timely and reliable.

This chapter introduces SACube (Self Adaptive Abstraction and Approximation), a novel framework designed to mitigate these computational challenges through adaptive abstraction and approximation technique. By enabling dynamic model switching, SACube ensures that computational resources are optimally allocated based on contextual requirements, maintaining both scalability and predictive accuracy. The framework emphasises a decision-centric approach, recognising that the validity region map, which defines when and where surrogate models can be employed, is as critical as the models themselves.

To evaluate its effectiveness, we apply SACube to a highway lane change maneuver case study, demonstrating its ability to significantly reduce computational overhead while preserving accurate decision-making. Furthermore, we extend the investigation by exploring SACube's applicability in a DT representation of a traffic system, assessing its adaptability in a complex, multi-agent environment.

By bridging the gap between computational efficiency and decision reliability, SACube offers a scalable and context-aware approach to DT modelling with real-time constraints. This chapter outlines its core principles, implementation, and experimental validation, highlighting its potential as a transformative tool for DT applications.

5.1 SACube Framework

In this section, we introduce the concept of Adaptive Abstraction and Approximation and proceed to present the SACube framework.

5.1.1 Foundations of SACube

The concept of adaptive abstraction and approximation establishes a new type of self-adaptive system that addresses computational challenges by dynamically selecting and utilising less complex models during runtime.

This approach enables the system to replace computationally expensive, high-resolution models with approximated or abstracted alternatives, depending on the real-time constraints and the system's knowledge, including goals, decisions, context, and the models' validity frame. These lower-complexity models are designed to preserve key behavioral properties of interest of the actual system, ensuring that decision-making remains valid within the application domain.

The SACube framework uses the MAPE (Monitor, Analyse, Plan, Execute) loop as its self-adaptation foundation because of its proven utility in managing complex, self-adaptive systems. MAPE's modular structure enables a clear separation of concerns, facilitating the monitoring of system states, analyzing runtime contexts, planning model switching, and executing adaptations in a seamless, structured manner. This cyclic process aligns closely with the requirements of SACube, which must continually adapt the managed system in response to real-time constraints.

Key principles of the SACube framework include:

- Dynamic Models Switching: The managing system contains a library of models with varying levels of detail and switches to a less-detailed model when the less-detailed model is valid for the actual system's current context.
- Validity Region Mapping: The framework emphasises the creation and use of a validity region map, which defines the conditions under which each model is valid. This map ensures that model switchings remain accurate and reliable.
- Decision-Centric Technique: SACube uses a decision-centric technique to assess a model's validity. Rather than focusing merely on output similarity, this technique assesses a model's applicability within the context of the decision-making process.
- Integration in Real-Time Digital Twins: The framework is designed to operate within Digital Twin systems with real-time constraints.
- Structured Workflows: SACube defines explicit workflows based on the MAPE-K (Monitor, Analyse, Plan, Execute, and Knowledge) loop to guide the framework's operations

In order to clarify this concept, consider the example of an autonomous vehicle with varying prediction models for surrounding vehicles, as depicted in Figure 5.1. The

figure demonstrates how adaptivity can balance performance with computational cost. Vehicles closest to the ego vehicle are the most critical for decision-making and require high precision in modelling. Vehicles further away demand less precision, allowing for greater tolerance of uncertainties in position, direction, and velocity.

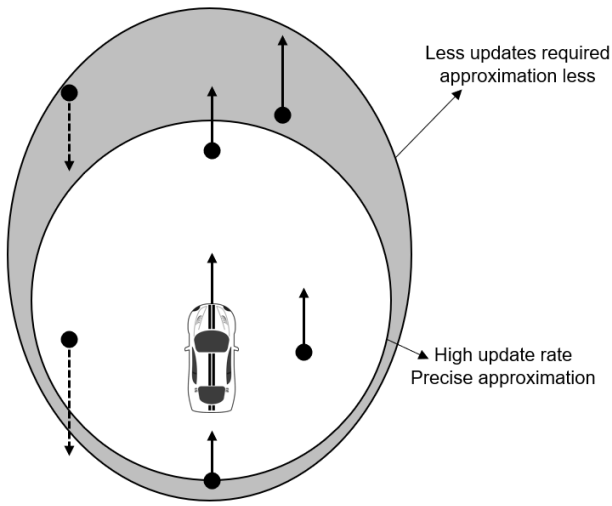

Figure 5.1: Autonomous vehicle approximation based on domain knowledge.

As depicted in Figure 5.2, this approach uses the contextual information from the decision-making process, system goals, and runtime context to dynamically adapt models. However, for such systems to remain safe and reliable, it is crucial to explicitly reason about the trade-offs between model approximation and computational time within the real-time constraints.

Techniques such as surrogate modelling offer a practical foundation for reducing computational cost while preserving validity frames and addressing properties of interest. Additionally, domain knowledge can enhance this adaptivity, enabling the system to apply appropriate levels of abstraction at runtime. By employing this strategy, the SACube framework ensures efficient utilisation of computational resources by optimising the modelling process while meeting real-time constraints.

The semantics of a simulation model are determined by simulating the model and using a semantic mapping function ([[.]]) that simulates the model and links it with its intended meaning. The simulation results in a collection of traces. While there are a multitude of simulation traces, not all of the provided information is the quantity of interest of the decision-making algorithm or or serves to reason the logical behavior of the system. We need a function $f()$ to extract properties of interest. Then the function $g()$ is used to transform between the quantity of interest and the logical property. A logical property is a Boolean value and is on the ontological level, where it gets a real-world meaning. The framework is similar for an approximated/abstracted model and reasons on the same logical property. This means that the function $g'()$ should give an equivalent result to the more detailed model, although with more uncertainty. The more detailed model can

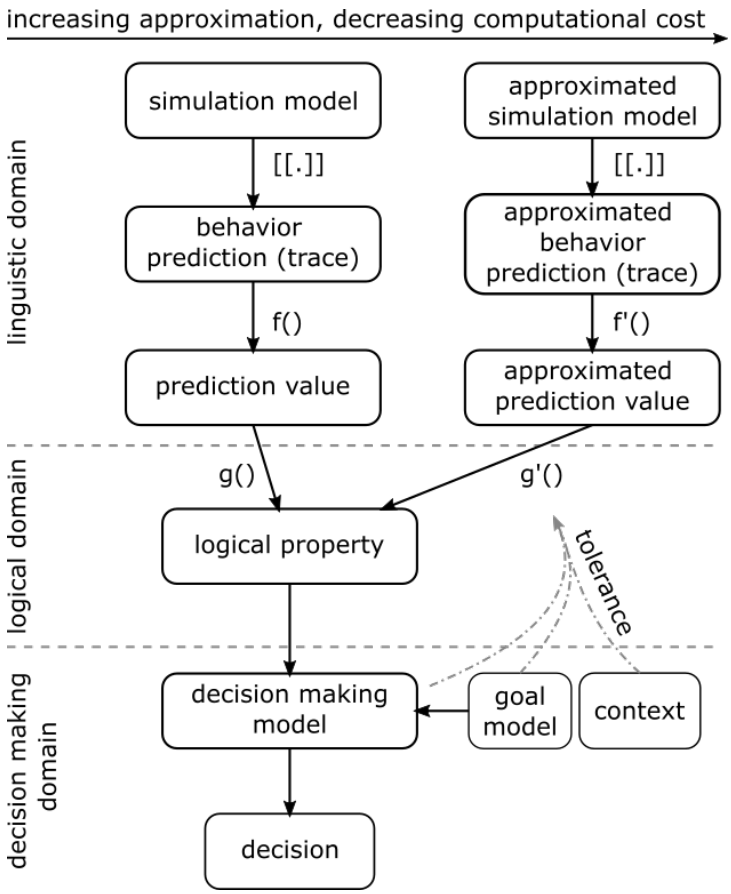

Figure 5.2: Conceptual framework.

be substituted with the approximated one if the uncertainty is within tolerance [24].

To apply this approach, it is necessary to utilise languages and frameworks that facilitate self-adaptation by domain experts, as well as self-adaptation within the real-time twining paradigm. In the following, we describe the Adaptive Abstraction and Approximation (SAcube) framework.

5.1.2 Overview of SACube Framework

The proposed architecture in Figure 5.3 is based on the insights provided by the conceptual framework. This architecture is based on the MAPE-K loop, which is a high-level control loop for self-adaptive systems from IBM. This architecture consists of four phases: Monitor, Analyse, Plan, and Execute. We discuss these phases in detail in the following sections.

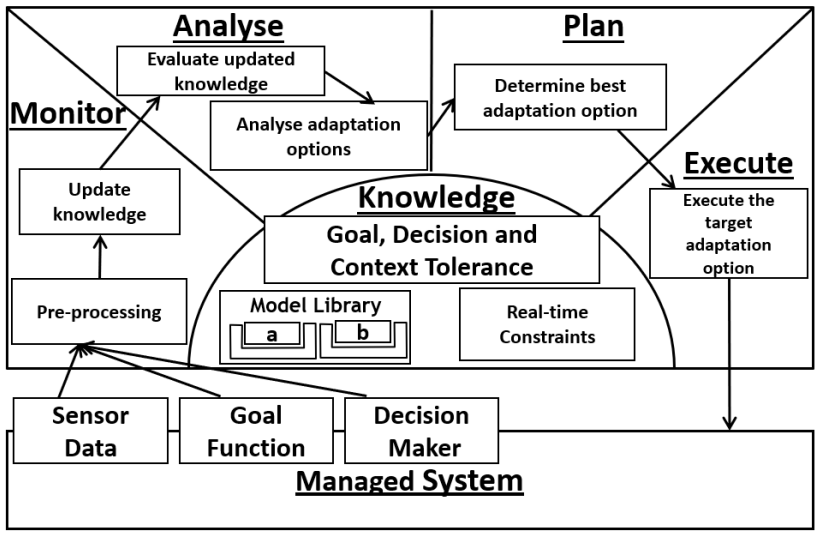

Figure 5.3: Self-adaptive abstraction and approximation architecture.

5.1.3 Knowledge Creation

The knowledge function provides the knowledge and mechanisms needed for the MAPE components to cooperate and achieve self-adaptation as a whole.

Creating the knowledge needed for switching models occurs at design time, as shown in Figure 5.4. However, managed sensor data, including environmental factors, properties of interest, and real-time constraints, is part of the knowledge used during run-time. Knowledge relies on different types of data related to the managed system, environment, properties of interest, adaptation goals, limitations, etc. The meta-information [185] can also be part of the knowledge.

To choose the appropriate model, we need to know which models are valid in which regions, so we can swap them out for others and get the correct results. This information is mostly available in the form of validity frames. If not, we use Decision-Centric technique introduced in Chapter 4 to find the required information.

5.1.3.1 Available Model Validity

The model's validity may have been established upfront, defined by preset constraints, domain knowledge, or other fixed criteria, for example, in a validity frame. This validity frame is determined at the start of the modelling process and explicitly presented for reference throughout the model's application [3]. This is sufficient information to create the needed knowledge for the SACube framework.

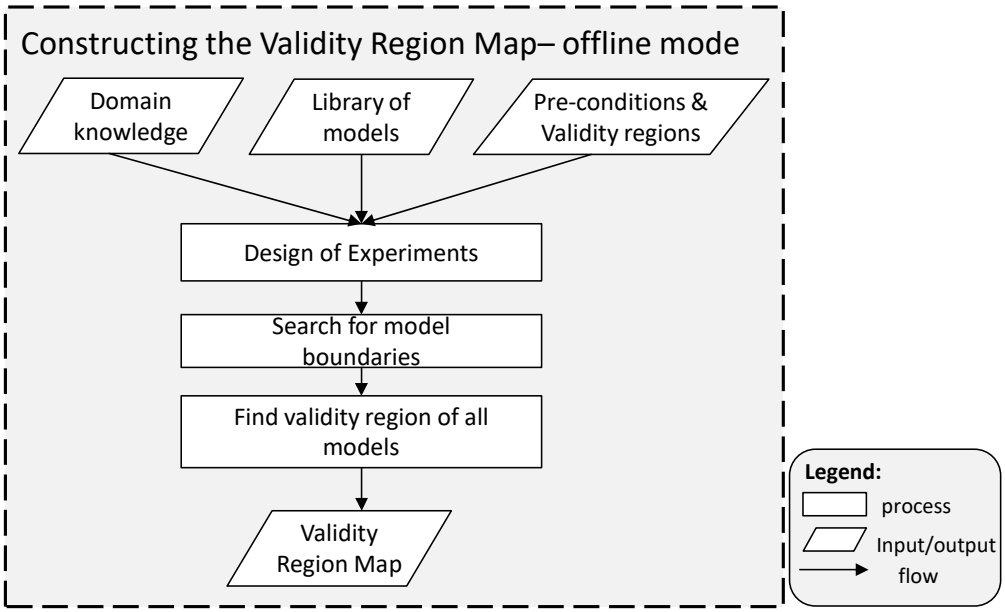

Figure 5.4: Constructing the validity region map.

5.1.3.2 Decision-Based Model Validity

If model validity is not available, it needs to be created. Model validity is determined based on the decision-centric technique described in Chapter 4. Rather than comparing outputs, SACube evaluates if decisions derived from different models are equal and align within acceptable tolerances. This innovative method reduces computational costs while maintaining reliable decision-making.

All validity frame experiments and boundary searches are limited to points within the feasible region $\mathcal{F}$, as determined by domain constraints. This ensures that we avoid wasting computational resources on assessing erroneous configurations and instead focus on physically relevant states.

5.1.3.3 Constructing the Validity Region Map

Once validity regions are established for individual models, we need an efficient way to represent and query this information at runtime. We organise the validity information into a structured map that enables quick model selection based on the current system state.

Let $\mathcal{M}$ be our set of models and let $\mathcal{V}_m$ denote the validity region of model $m \in \mathcal{M}$. We can define a partial order $\sqsubseteq$ on models based on their validity regions:

$$m_1 \sqsubseteq m_2 \iff \mathcal{V}_{m_1} \subseteq \mathcal{V}_{m_2} \tag{5.1}$$

This forms a partially ordered set (poset) $\mathcal{M}$. However, not all models are comparable - there may exist $m_1, m_2 \in \mathcal{M}$ where neither $m_1 \sqsubseteq m_2$ nor $m_2 \sqsubseteq m_1$ holds.

The validity region map R then maps points in the state space to sets of valid models:

$$\begin{aligned} R &: \mathcal{X} \to \mathcal{P}(\mathcal{M}) \\ R(x) &= \{m \in \mathcal{M} \mid x \in \mathcal{V}_m\} \end{aligned} \tag{5.2}$$

where $\mathcal{P}(\mathcal{M})$ is the power set of $\mathcal{M}$.

The validity regions can be represented in several ways depending on their complexity and the dimensionality of the state space. However, to enable real-time model selection, the map must support efficient querying. We implement this through pre-computed lookup tables.

When working with composed models, knowledge creation becomes more complex. The validity regions of composed models may not simply be the intersection of individual model validity regions, as compositional effects can introduce new constraints or behaviors. The validity region map needs to consider these composition effects, ensuring that the combined models not only individually satisfy their validity constraints but also work correctly together. This requires additional validation experiments to verify the boundaries of composed model validity regions and understand any emergent behaviors that arise from model interactions.

5.1.3.4 General Workflow

Figure 5.5 illustrates the general workflow of the proposed framework in the form of a flowchart. The whole MAPE-K control loop operates at runtime. We now focus on different functional elements of the MAPE-K loop, drawing on insights from Weyns' research [7].

5.1.4 Workflows

The Monitor phase maintains the managing system's understanding of the current situation. This includes both the state of the managed system itself and its environment. Figure 5.6 shows the workflow of this phase. This workflow can be activated in different ways. The workflow may be triggered by a probe that has finished a sensing cycle, or the workflow may process sensor data in continuous cycles. The monitor gathers sensor data after it is triggered. A variety of sensors can be employed to monitor the environment and the managed system, which is domain-specific.

5.1.4.1 Monitor workflow

The monitor then updates its knowledge base with the current system state. Examples include how well our current model is performing, what resources (like computation

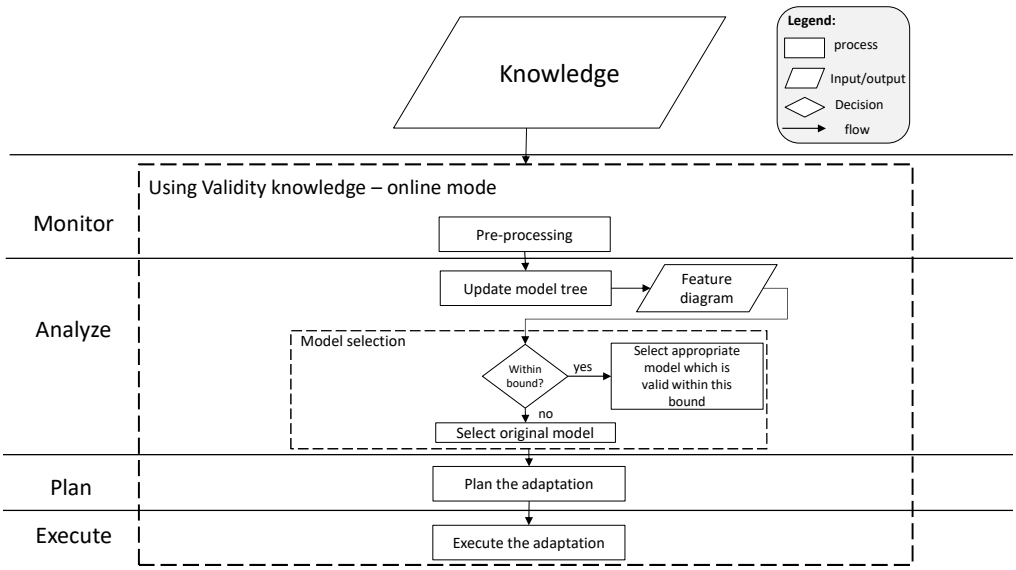

Figure 5.5: General workflow of SACube approach.

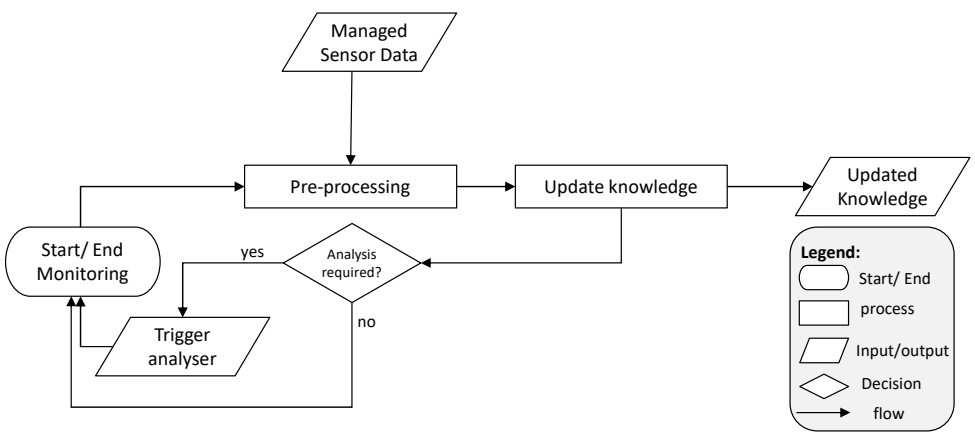

Figure 5.6: Workflow of monitor phase adapted from [7].

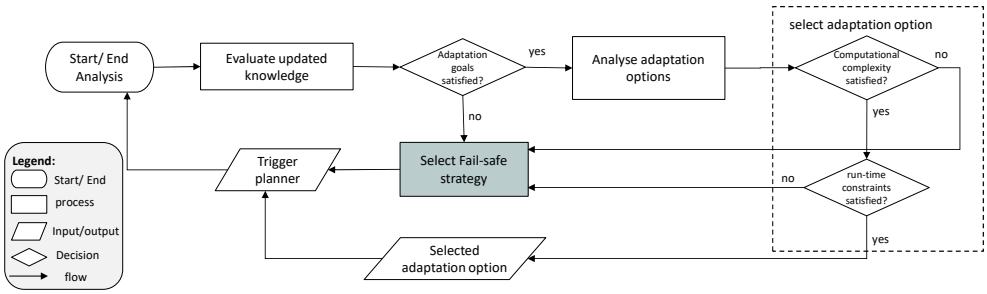

Figure 5.7: Workflow of analyse phase adapted from [7].

time and memory) we are using, and what is happening in the environment around us

It is important to note that this knowledge update is about maintaining our current understanding of the situation. It is different from updating our fundamental knowledge, like validity frames or the model library—those kinds of updates happen in a separate, slower development cycle.

The monitor needs to trigger the analyser in several situations:

- When we might violate runtime constraints—for instance, if our current model is taking too long to compute: $\exists rc \in \mathcal{RC} : r(m_{current})$ approaches false, where $\mathcal{RC}$ is a set of run-time constraints. These runtime constraints rc represent system limitations such as Worst Case Execution Time (WCET), available memory bounds, and resource availability.
- When the system and environment state gets close to the boundaries where our current model might stop being valid: $d(x, \mathcal{B}) < \epsilon$. Where $d(x, \mathcal{B})$ measures how close we are to the validity boundary and ϵ is our safety margin for triggering adaptation.

5.1.4.2 Analysis workflow

The main role of the analyser is to evaluate the up-to-date knowledge and determine if the system meets the adaptation goals. If it does not, the analyser will analyse possible configurations for adaptation. Figure 5.7 adapted from [7] shows the basic workflow of the analyse phase.

Analysis workflows are similar to monitor workflows in that they can be externally active (by the monitor, for example), periodically triggered (based on a predetermined time range), or restarted automatically (after the previous cycle is finished). The analyser analyses the adaptation options. The adaptation options define a set of configurations that can be reached from the current configuration by adapting the managed system, considering the knowledge. Once the analysis of the adaptation options is completed successfully, the planner function will be triggered to create a plan for adapting the managed system using the analysis results. We formalise this as follows:

With R defining the map from state to valid models, we need a selection function S to choose which model to use when multiple models are valid. Let:

$$\begin{aligned} &S : \mathcal{P}(M) \rightarrow \mathcal{M} \\ &S(\mathcal{M}') = m^* \\ &\text{where } m^* = \underset{m \in \mathcal{M}' | \forall rc \in \mathcal{RC}, rc(m)=true}{\arg\min} \text{cost}(m) \\ &\text{where cost} : \mathcal{M} \rightarrow \mathbb{R}^+ \\ &\text{and } rc : \mathcal{M} \rightarrow \{\text{true}, \text{false}\} \end{aligned} \tag{5.3}$$

The selection function S uses a cost function that could consider aspects like computational complexity, memory requirements, numerical precision, and model fidelity. Each model must also satisfy a set of runtime constraints $\mathcal{RC}$. Only models satisfying all runtime constraints are considered in the optimisation process. Note that this can easily be expanded to the composition of models, where the entire composition needs to adhere to the constraints. For example, while individual models might each satisfy WCET constraints in isolation, their composition might exceed timing bounds due to communication overhead or resource contention. Similarly, memory constraints must consider the total memory requirement of all composed models.

Then $(S \circ R)$ gives us a function that maps states to preferred models:

$$\begin{aligned} S \circ R &: \mathcal{X} \rightarrow \mathcal{M} \\ (S \circ R)(x) &= S(R(x)) \end{aligned} \tag{5.4}$$

When multiple models have equal cost under S, the framework needs clear policies. Typically, we default to the model with the broader validity frame in such cases to ensure safety, though this trades off computational efficiency.

Nevertheless, the analysis may encounter that the $(S \circ R)(x)$ is empty and thus prevent its successful completion. One possible cause is that none of the analysed adaptation options meet the minimal requirements to be considered for adapting the managed system. The second cause is the circumstances that do not exist based on current knowledge; for example, in the highway lane change system, the bicycle object is not defined. Another possible cause is that the time window to perform the analysis of the adaptation options passes without any useful results. Under such circumstances, the analyser will take a **fail-safe strategy** before triggering the planner.

The selection of a fail-safe strategy is a domain-specific task that depends on several factors, including the criticality of the system and its current situation, the cause of the unsuccessful analysis, and the available partial analysis results. Possible strategies may include avoiding adapting the system, adapting the system to a pre-defined configuration, or bringing the system to a safe stop. We need a fail-safe strategy to prevent the system from crashing and bring it to a safe stop.

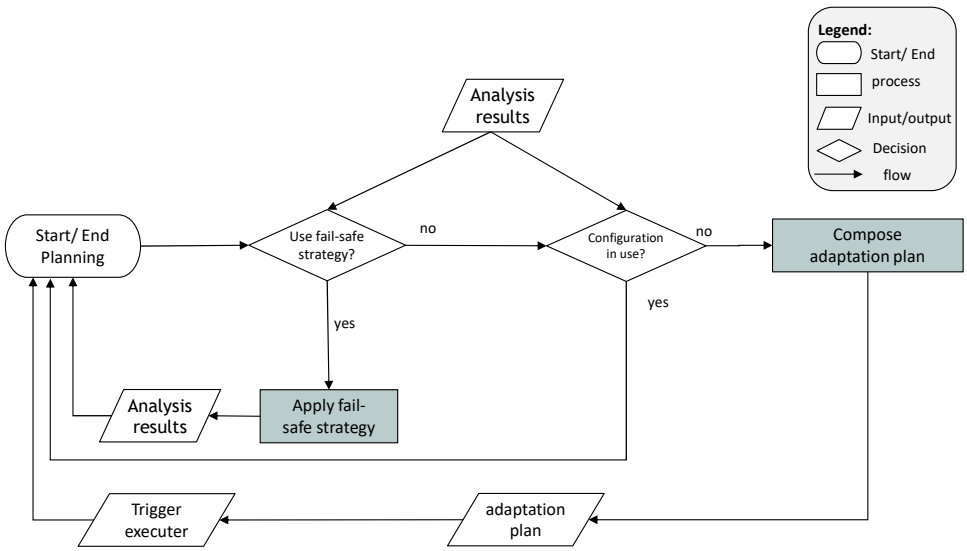

Figure 5.8: Workflow of plan phase adapted from [7]. *"Apply fail-safe strategy" and "Compose adaptation plan" are domain-specific processes*.

5.1.4.3 Plan workflow

The primary role of the planner is to plan the adaptation process that transitions the managed system from its current configuration to the new one determined by the analyser. Figure 5.8, adapted from [7], shows the basic workflow of the planner function, which is generally activated externally by the analyser.

After triggering, the planner uses the analysis results to determine whether a fail-safe strategy must be executed. Should such a strategy be considered essential, the planner formulates an adaptation plan to direct the system towards a safe state. In this case, the planner function terminates. Otherwise, the planner composes a plan to adapt the managed system.

A plan provides the optimal sequence of steps required, i.e., the actions that need to be performed to adapt the managed system from its current configuration to the desired new configuration. An adaptation action is a particular operation executed on the managed system to enable adaptation. It often involves a collection of domain-specific instructions that are interpretable and executable by the managed system.

The selection of planning mechanisms is inherently specific and influenced by the adaptation problem's nature. In self-adaptive systems, two principal elements influence the choice of an appropriate planning mechanism: the available knowledge and specific limitations like real-time constraints. Two types of problems can be encountered; we focus on the first case in SACube. The first one is when the model is used in an experiment-based fashion. The model is often restarted from known initial conditions to predict some property of interest (e.g., when optimising). As the interface of the models is the same, there is only the question of when the change can happen. The most natural point would be

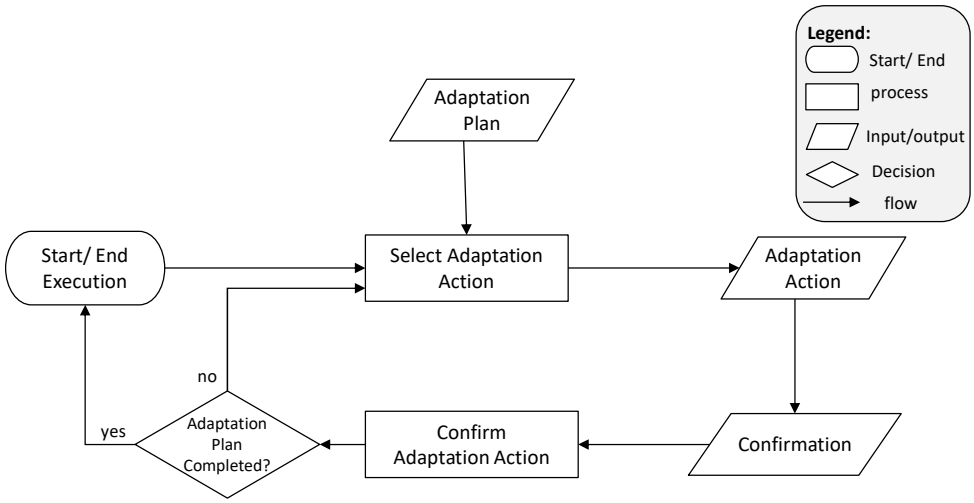

Figure 5.9: Workflow of execute phase adapted from [7].

to switch between the restarts of the model. The second case is much more complicated; it involves switching a running model. This brings in many challenges that should be solved domain-specifically, for example, (a) the model should be initialised with the same state. Switching from a more detailed model to a more abstract/approximated model is feasible; however, the other way around requires that state estimation techniques be used to fill in the gaps of the state. This can be computationally expensive. (b) Once the model is correctly initialised, when to switch the model must be decided. There are some strategies, such as parallel running. At the managed system level, infrastructure might need to be available for this.

For composed models, the composition needs to be created. For example, when using FMI, a new master algorithm might need to be created to ensure the individual models' requirements are met.

After the generation of the adaptation plan, the planner triggers the execute function.

5.1.4.4 Execute workflow

Executing the adaptation plan by reconfiguring the managed system from its current state to the desired configuration is the responsibility of the Execute phase.

As shown in Figure 5.9 adapted from [7], the workflow of the Execute phase is initiated by the planner, which triggers the executor to perform the necessary adaptation actions. The executor chooses an adaptation action from the adaptation plan and implements it in the managed system via an effector. If the plan is defined as an ordered sequence of adaptation actions, the first action of the sequence is selected for execution. In cases where no specific order is established, any action may be chosen for execution.

Once an adaptation action is applied, the executor verifies whether the adaptation plan

has been fully executed. If the plan is incomplete, the subsequent adaptation action is selected and implemented in the managed system. The executor will subsequently await a trigger from the planner to proceed with the next adaptation step. This iterative process guarantees the managed system is continually reconfigured until the adaptation plan is completely implemented.

5.2 Experimental Validation of SACube

In this section, we address the following experimental questions: (a) Does the framework properly detect and adapt to real-time constraints? (b) Is the created validity information valid? Is the configuration valid?

5.2.1 Highway Lane Change Maneuver Case Study

This study uses a lane change control system to demonstrate the challenges of integrating adaptive abstraction and approximation in real-time constrained environments. It is important to clarify that this work does not focus on control engineering or developing specific control techniques. Instead, the highway lane change case study facilitates visual and intuitive reasoning to show the principles and application of our framework.

5.2.1.1 Problem Description

The automated Lane Change Maneuver (LCM) system automatically allows the ego vehicle to autonomously shift from one lane to another. This system models both longitudinal and lateral dynamics, enabling the ego vehicle to adapt its trajectory based on environmental conditions. The Highway Lane Change example demonstrates how to use ground truth data to construct and run a system-level model for lane change. By employing this use case, we effectively show how applying the SACube framework addresses computationally expensive models and accurate decision-making.

5.2.1.2 Experimental Setup

We employ a simulation platform based on MATLAB/Simulink. The sample time is $0.1s$, which indicates when, during the simulation, the block produces outputs. Sample time is the time interval between scenario simulation steps. The simulation can utilise many models with varying levels of abstraction and approximation to predict the next positions of the various cars. In this experiment, we use three different models, listed from the most detailed model to the model with the narrowest validity frame–the least expensive model:

- High-validity Model is the most detailed model, which is computationally the most expensive model in this experiment. This model predicts the trajectories for the current ego vehicle based on surrounding vehicles. By using a controller in

each vehicle, a more detailed evaluation of the trajectories of the cars is possible. However, because position changes affect the trajectories, the model is fixed-pointed such that a stable situation arises. The high-validity model has a validity frame equal to the validation domain of the system.

- Constant Acceleration model (C.A model) is the kinematic equation $x(t) = 1/2 * a * t^2 + v * t + x_0$. In this formula, a is the vehicle's acceleration in m/s^2, v, the velocity of the vehicle in m/s, and x, the vehicle's position in meters.
- There is, however, the possibility of utilising an approximated version of the high-validity model, the Constant Velocity model (C.V model) with the following kinematic equation: $x(t) = v * t + x_0$.

In this experiment, the model composition consists of integrating three models, with each lane treated independently. In this context, "each lane is independent" refers to the vehicles in each lane being modeled separately, with no direct contact or interdependence between the lanes until specifically indicated. As a result, each car can select its own appropriate model based on its current position, velocity, and acceleration; otherwise, it needs to use the high-validity model.

5.2.1.3 Results

In this experiment, $\mathcal{M}$ is our set of models, where m_h is the high-validity model, m_{ca} is the C.A model and m_{cv} is the C.V model.

$\mathcal{M} = (m_h, m_{ca}, m_{cv})$

Creating knowledge starts at design time by building knowledge. Therefore, we create knowledge for switching models to know which of our three models are valid in which boundaries of the combination of position, velocity, and acceleration, and give us the correct results. This helps us select models at run-time.

For each model, we check whether model validity is available. For m_h, we use the available model validity. For this experiment, a valid relative position is between -50m and 550m, as the ego car starts at 50m, and the maximum sensor range is 600m.
The valid relative velocity range is from -12m/s to 15m/s, considering the ego car's initial velocity is 18m/s. The maximum vehicle acceleration depends on factors such as tires and horsepower. Top-of-the-line cars can go from 0 to 60 mph in 5 seconds with an average acceleration equal to 5.4 m/s^2 [186].
Additional research in this field, as shown in [187], concentrated on the acceleration and braking intensity of various vehicle types, including motorcycles and trucks. The study found that the maximum acceleration falls within the range of $0.45m/s^2$ to $2.87m/s^2$, and the maximum deceleration values range from $0.59m/s^2$ to $5m/s^2$. Therefore, for this experiment, we choose -$4.6m/s^2$ for the maximum deceleration and $2.87m/s^2$ for the maximum acceleration. So the valid relative acceleration is between -$3.69m/s^2$ to $3.78m/s^2$ while the ego car acceleration is -$0.91m/s^2$

Based on the available knowledge, the graph in Figure 5.10 illustrates the validity region of m_h. However, for m_{ca} and m_{cv}, the model validity is not available. So, we need to

choose the decision-based model validity instead and use the proposed decision-centric technique to establish model validity.

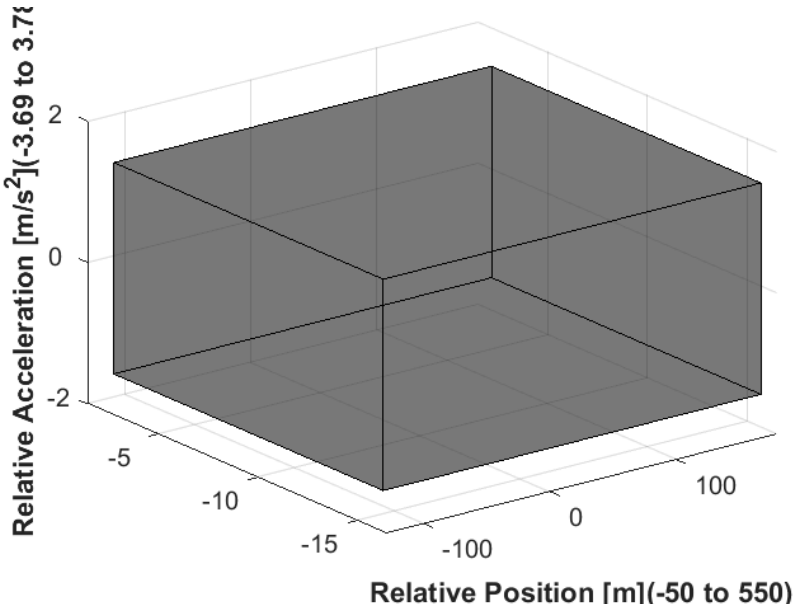

Figure 5.10: Validity region of high-validity model.

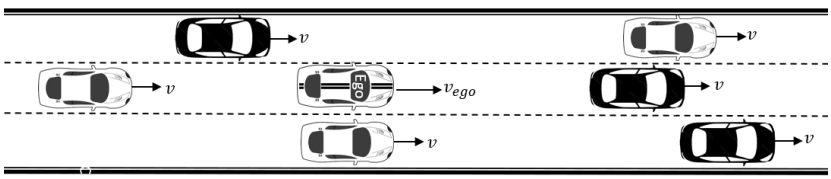

Figure 5.11: Building knowledge scenario.

In this experiment, we define the scenario shown in Figure 5.11 to determine the decision-based model validity, considering all possible situations. This is possible by positioning six cars surrounding the ego car. The number of surrounding cars depends on the number of lanes, which, in this case, is three.

To find the validity region $\mathcal{V}$ for each of m_{ca} and m_{cv}, we use algorithm 2 in Chapter 4.

With this algorithm, we follow the decision-centric technique. We define a set of domain constraints C as the following set:
c_1: Vehicles maintain deterministic behaviors during the simulation. And in reality, each car might have unknown situations and different behaviour.
c_2: The minimum car speed has to be 6 m/s.
c_3: Each target vehicle, after acceleration or deceleration, continues with the same velocity.
c_4: The front gap, the gap between a car and the front car, has to be at least 30 m. And the rear Safety Gap must also be 30 m.

c_5: The ego car has no acceleration, and the speed is constant.
c_6: When the car is in front of the ego car, the model is valid in positions further away.
c_7: When the car is behind the ego car , the model is valid at positions farther away.

Next, we search for boundaries $\mathcal{B}$ for each surrounding car using binary search. We evaluate different combinations of positions, velocities, and accelerations using the following method:
For each surrounding car, we first vary its relative position (relative to the ego car) using binary search, 0then run the simulation once with the m_{cv} and once with m_h. Then we compare the intended results, which are the predicted trajectories for the ego car. The predicted trajectories do not need to be point-to-point the same; we define a distance metric d_Y that we consider two trajectories as different if the car starts changing lanes. So if the trajectories are the same, we consider that relative position as the position boundary for the car.

This point is not yet our intended valid point. Therefore, we check whether the car is in front of the ego car. In this case, the feasible region $\mathcal{F}$ is the region between the boundary position and the sensor's maximum range (the end of the highway). On the contrary, if the car is at the rear, $\mathcal{F}$ is the region between the boundary position and zero.

Then, we determine valid velocities. For all discretised positions, we check different velocities via binary search. If the decisions match, then we add the point to the $\mathcal{B}$ and continue the same process that we did for the position. And we follow the same method for valid accelerations. The result is the validity region for m_{cv} illustrated in Figure 5.12. To find the validity region for m_{ca}, we follow the algorithm 2. The matched decision

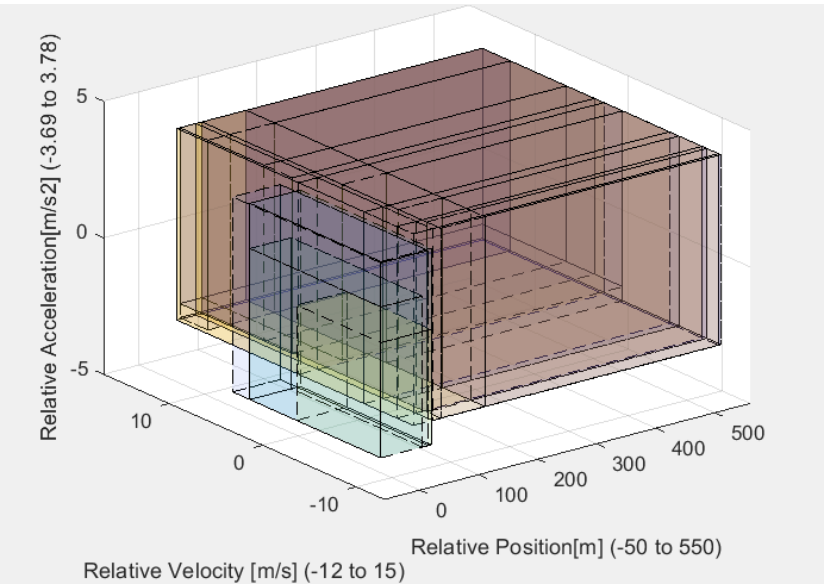

Figure 5.12: Validity region of C.V model.

is achieved by comparing the simulation results from the m_{ca} and the m_h depicted in Figure 5.13. So we have the validity region for individual models, and we construct the validity region map. This process results in a hypercube graph that maps the validity regions of each model based on key properties of interest—in this experiment, position,

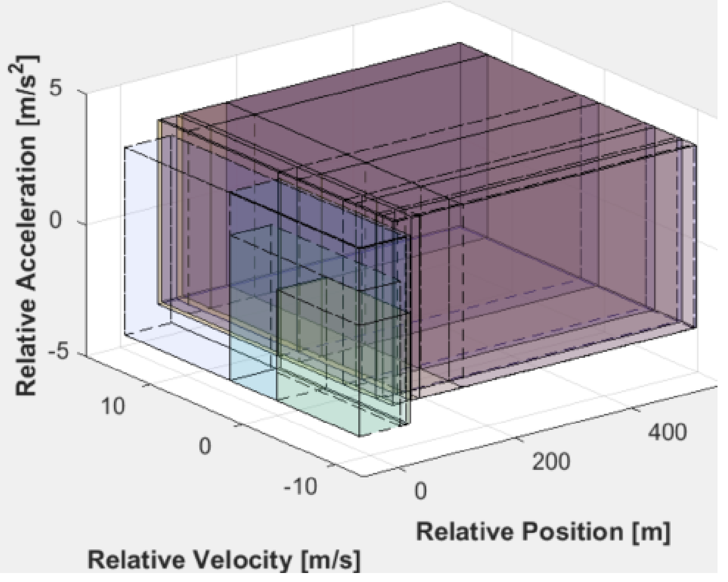

Figure 5.13: Validity region of C.A model.

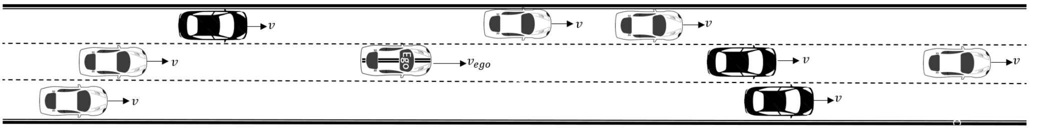

Figure 5.14: Lane changing scenario.

velocity, and acceleration. The validity region map R is shown as a hypercube graph in Figure 5.16.

Finally, we store the validity map in a lookup table for use at runtime.

At runtime, we use a sample scenario shown in Figure 5.14 where the managing system uses the validity region map of the three defined models: m_h, m_{c_v}, and m_{c_a}. The process begins by monitoring the highway lane change system (managing system). Each time the system predicts the next position of cars, the managing system checks whether m_{c_v} is valid for each car. We get this information from the lookup table we provided at design time.

If the car's characteristics—such as position, velocity, and acceleration—fall within the range $\mathcal{V}_{m_{cv}}$, the system selects m_{c_v}. If not, m_{c_v} is evaluated next. When neither of these models is valid, the system defaults to m_h, which is the high-validity model, ensuring accurate predictions and decision-making in its validity region.

To validate the technique, we compare the simulation results with the baseline by running simulations that exclusively use the high-validity model. The outcomes shown in Figure 5.15, including lane changing decisions, are consistent between the SACube framework and the baseline. This confirms the validity of the technique.

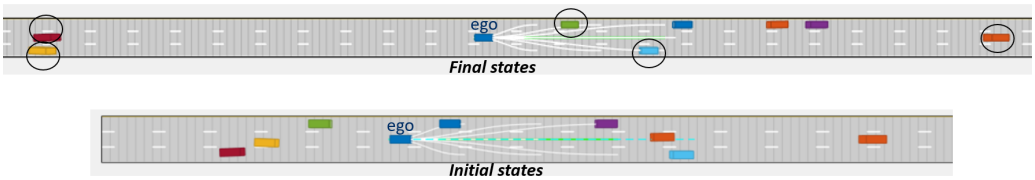

Figure 5.15: Comparison of SACube framework with baseline for lane change decisions.

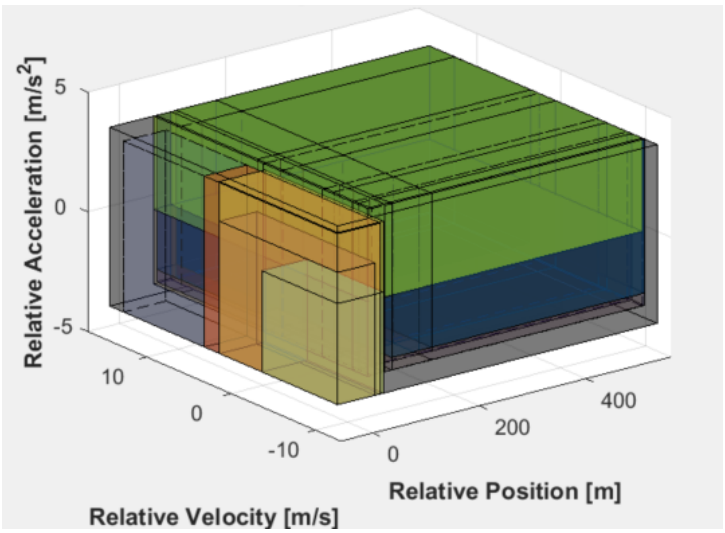

Figure 5.16: Validity region map.

By doing this experiment, we addressed RQ1, determining when, where, and how to switch between models of different complexity during system operation. The cars marked with a circle around them are those that switched models.

We determined the validity regions of the models and used this information to make Figure 5.16. This process directly addresses RQ2 by demonstrating the accuracy and applicability of the validity region map information generated through the steps defined in the framework.

Addressing RQ3, we evaluate the Highway Lane Change System as a DT, where it

scenarioName	**without SACube**(s)	**with SACube**(s)
$scenario_1$	141s	13.4s
$scenario_2$	127s	13.23s
$scenario_3$	204s	19.46s
$scenario_4$	196s	22.97s
$scenario_5$	183.9s	16.98s
$scenario_6$	377s	33s

Table 5.1: Comparison of execution time($seconds(s)$)

continually updates the current state of the vehicle and its surroundings. This virtual representation supports accurate predictions and facilitates real-time decision-making. We run a series of scenarios to evaluate whether the average execution time is reduced when utilising the SACube method. This involves running a diverse set of scenarios and measuring their execution times to assess the framework's ability to meet real-time constraints effectively. For each scenario, we measure the execution time targeting the time for predicting the next car positions by applying the SACube method and comparing it with the baseline, which involves no model switching, abstraction, or approximation, by executing the high-validity model.

We run different scenarios, including $scenario_1$ to $scenario_6$, considering different cars inside and outside the validity region of our three models. Results depicted in Table 5.1 show a big reduction in execution time while using the SACube framework, switching to less-detailed models. Execution time is the total accumulated wall-clock time required to predict the next positions of all vehicles throughout a 5-second simulation window. However, it is worth noting that each run of the same scenario might not have the same execution time due to factors such as background programs running on the PC or system resource availability, which can introduce variability in the timing measurements. Improvement in execution time is influenced by the complexity of the high-validity model as well as the design of less-detailed models.

5.3 Related work

Single models are not always enough to capture the depth of real-world processes; however, simulations that employ multiple models can offer a more comprehensive understanding of the problem at hand [188]. Managing computationally expensive models requires careful consideration, particularly under real-time constraints where high computational demands can affect system performance and deadline adherence. The use of more complex models often incurs significant computational overhead, increasing the risk of missed deadlines. There are multiple approaches for addressing computationally expensive models.

Aggregation techniques commonly used in combat modelling reduce complexity by grouping entities and interactions into higher-level abstractions [189]. Similarly, Multi-Resolution modelling (MRM) involves creating a single model, a family of models, or both that represent the same phenomenon at several resolution levels while allowing users to enter parameters at each level according to their needs [171]. MRM is also known as variable or selectable resolution modelling. Sometimes the word fidelity is used instead of resolution. MRM is closely related to model abstraction, which is a way of simplifying models while keeping the essence of a phenomenon concerning the application at hand [172].

Franceschini et al. [173] present an adaptive abstraction technique. They utilise a specified trigger to determine when to transition between abstraction levels. In prior work, the author presented a dynamic abstraction simulation that switches between an agent-based formalism and a discrete event formalism. The statistical analysis of the observed emergent behavior serves as the basis for the decision to change abstraction levels. A more rigorous framework [174] extends the adaptive abstraction technique to decide

when and where to switch between abstraction levels.

Some research recommends employing an abstracted and/or approximated model instead of a more detailed model. The *self-Adaptive Abstraction and Approximation* technique [24] is based on the MAPE-K loop previously presented to adapt a real-time system under study by changing the model and using an approximated and/or abstracted model instead of the more detailed model. However, the validation of the substitute model is an essential issue that needs more investigation. One approach is to look at the model behavior, calculate the deviation, and find tolerances [175, 25]. Another approach is using the ESS (EMF-Based Simulation Specification) technique [176].

Another research proposes Variable-Resolution modelling (VRM), which was stimulated by the RAND institution in 1990-1992 [190]. VRM is building models or families of models to operate at different resolutions, allowing users to seamlessly adjust the resolution based on the analytical needs. VRM applies to designing a new model or family of models.

Hybrid system identification focuses on identifying models for complex systems using a class of dynamical models, which is called a hybrid system. In the literature, several methods have been proposed to perform hybrid system identification. Lauer et al. proposed a method utilising kernel functions to estimate nonlinearities [191, 192]. Feng et al. studied the use of sparse polynomial optimisation for hybrid system identification. By recasting the problem into a polynomial optimisation form and exploiting the sparse structure to keep a relatively low computational cost [193].

Another line of work explores switching between different model types. For example, Serena et al. consider three types of models: equation-based models (EBM), agent-based models (ABM), and cellular automata models (CA). EBMs are usually continuous space and continuous time as they are often based on a set of differential equations. They are usually efficient and most suited for aggregates over large populations. ABMs are usually continuous space but discrete time, as they consist of autonomous interacting agents situated in space at specific time steps. They can be computationally demanding and are most suited for systems where the interactions of agents with each other or with the environment are important. C.A is usually a discrete space and time stepped. They represent the domain as a lattice of cells with simple rules to update the state of the cell based on the state of a subset of neighbours [194].

The SACube framework builds on the strengths of these prior approaches while providing a unique combination of features that address several critical gaps. Unlike static multi-resolution modelling or predefined abstraction techniques, SACube takes a dynamic approach to adaptive abstraction and approximation. By using the validity frame concept, the framework ensures that model switching is accurate, explicitly validating the conditions under which models with the narrowest validity frame can replace models with broader validity.

5.4 Discussion

In this research domain, selecting the most complex model is not necessarily the best option. Utilising the concept of hysteresis allows for the examination of a more computationally efficient model with less detail, potentially surpassing the high-resolution

option under specific circumstances. When hysteresis is taken into account, the selected model cannot be regarded as the least effective alternative. It is indeed feasible that the boundaries of the validation domain, as depicted in Figure 5.17, may change. As a result, points once considered valid may now lie beyond the validation domain, whereas others previously outside its reach may now satisfy the validity criteria.

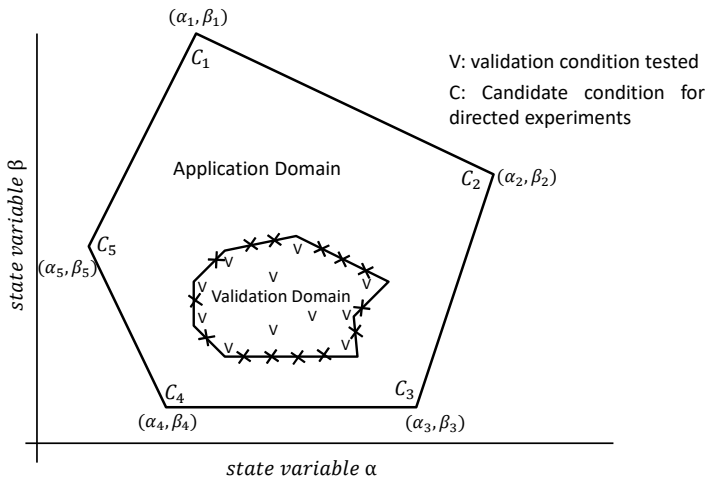

Figure 5.17: Boundaries of the validation domain.

Additionally, model updates and system evolution are essential in responding to changes during the DT lifecycle. Erkoyuncu et al. presented a design framework that encapsulates these lifecycle alterations within a digital twin, specifically from a data architecture perspective [195]. Throughout the DT lifecycle, there are possibilities to enhance models, including the refinement of the reference model. For example, in our experiment in Section 5.2, if a car activates its blinker, the contextual scenario changes, requiring an update to the knowledge. This alters the validity region map.
Nonetheless, these kinds of updates now occur outside of the traditional MAPE-K loop, instead occurring within an external "developer loop" that supervises the MAPE-K process. This shift requires carefully defined strategies and regulations within the twinning paradigm. Furthermore, updating a model necessitates corresponding updates to the library of models and their associated validity frames, ensuring that the revised models remain within the evolving validation boundaries established by the hysteresis concept.

5.4.1 Composability, Model Composition and validity

The approach we advocate relies on the use of a model library (family) consisting of different models that all reason over the same property of interest. However, to scale this approach, many problems can also be decomposed into multiple sub-problems, where for each of these sub-problems, a different model is used. *Composability* is the capability to select and assemble simulation components in various combinations into valid simulation systems [196].

5.4.1.1 Model Composition Mechanisms

In the literature, many ways are described for composing different simulation models, even when modelled in different simulation formalisms. One well-known composition mechanism is co-simulation. A widely adopted standard for model composition is the Functional Mock-up Interface (FMI) for co-simulation [197]. FMI provides a standardised interface for coupling different simulation tools, where simulators can be packaged as Functional Mock-up Units (FMUs) with clear input/output interfaces. Each FMU encapsulates its own solver and advances its state independently, while a master algorithm coordinates data exchange and synchronisation between units at discrete communication points.

Sarjoughian [198] describes model composability as essential for integrating heterogeneous simulation models. He categorises approaches into mono, super, meta, and poly modelling formalisms, each enabling different levels of interoperability. Poly modelling, using a broker like the Knowledge Interchange Broker (KIB), ensures structured data exchange and synchronisation between models with distinct formalisms, supporting scalable and flexible simulation systems.

5.4.1.2 Validity Challenges in Composition

However, a fundamental challenge in model composition is that validity does not necessarily compose ,that is, even when individual models are valid for their specific purposes, their composition may not be valid for the integrated system. This is known as the composability problem [199].

Traditional approaches to validation focus on each model in isolation, but this fails to capture interaction effects between models. There is no general rule for determining the validity of composed models, as the interactions can be complex and domain-specific. This presents a significant challenge for systems that need to dynamically compose and switch between models.

Our framework has already implemented a basic form of composition in the lane change scenario, where we compose models for vehicles in different lanes. This implementation benefits from the property that left and right lanes operate independently of each other, with no direct coupling between their state variables. This independence significantly simplifies the composition problem, as validity in one lane doesn't affect validity in other lanes. However, model composition in more complex, interdependent systems requires deeper exploration.

5.4.1.3 Categories of Model Composition

Various types of model composition exist, each introducing unique challenges for validity:

- **Chained / Sequential Composition:**
  Models are connected in sequence, where the outputs of one model become inputs

to another. For example, Weather prediction feeding into a crop yield model, which feeds into a market price model, or a Simulink model with components.

Validity implication:

- Validity typically narrows (intersection of individual validity regions).
- Errors and uncertainty propagate and often amplify through the chain.
- Earlier models in the chain often have a disproportionate impact on final results.

Capability of Our Technique:
The validity region map can represent the narrowed validity region of the composition. So each composition ($n \times m$) times have to be looked into (unless there are orthogonal parts). The selection function can evaluate entire chains rather than just individual models. Cost calculation needs to account for the entire chain of models.

- **Multi-Perspective Composition**
Different models represent different "views" or aspects of the same system. For example, Electrical, thermal, and mechanical models of an electronic device.

  Validity implication:

  - Complex validity relationships based on interaction between perspectives should always be evaluated together.
  - Shared variables between perspectives create interdependencies
  - Validity may be neither a simple union nor an intersection of individual validity regions.

  Capability of Our Technique:
  Cartesian product of individual model sets ($M_a \times M_b$) creates the composition space. Decision-centric technique applies to evaluate validity against the most detailed combination. The selection function remains unchanged, operating on the space of all possible combinations.

- **Regional / Spatial Composition**
Different regions of the system use different modelling approaches. Example: Traffic simulation with detailed models for intersections, simpler models for highways.
Validity implication:

  - Validity has a spatial dimension - it depends on location within the system.
  - Boundary consistency becomes critical for overall validity.
  - Models may operate at different scales, but must maintain consistent physics.

  Capability of Our Technique:
  Validity regions incorporate spatial dimensions. Selection depends on state and spatial location. Boundary conditions need special attention in validity assessment.

- **Hierarchical / Multi-level Composition**
Models at different levels of abstraction or scale interact with each other. Information flows up (aggregation) and down (disaggregation) between levels. For example, a system-level model interacting with detailed component-level models, biological models of cellular processes interacting with tissue-level models. **Validity implication:**

- Cross-scale consistency becomes critical for validity.
- Aggregation and disaggregation operations must preserve essential properties.
- Different levels may operate at different temporal and spatial scales.

Capability of Our Technique:
Validity regions need to account for cross-scale consistency requirements. Selection must consider which level of detail is appropriate for which parts of the system. Cost functions need to balance detail at critical components with efficiency at the system level.

Building on these categories of model composition, our approach also emphasises the role of adaptive composition mechanisms that adjust dynamically based on system requirements and evolving scenarios. This adaptability is crucial in ensuring that composed models maintain relevance and validity in complex, changing environments. Future research should focus on refining selection functions to optimise model interoperability, leveraging advanced techniques such as machine learning to assess composition validity. By integrating these advancements, we can enhance scalability and robustness in model composition frameworks, making them more effective across diverse applications.

5.4.2 Validity Region Representation

There are different ways to represent the validity regions based on how complicated they are and how many dimensions there are in the state space. But for real-time model selection to function properly, the map needs to be able to handle queries efficiently. In this study, we employed pre-computed lookup table, which is a fast solution and offers efficient retrieval. However, this approach may lack granularity and can limit the number of models that can be handled. To address these limitations, alternative methods such as spatial partitioning with quad-trees [200] can be used. Although slightly more computationally expensive, this method offers greater flexibility and scalability. In particular, query complexity grows logarithmically with the number of models-$O(\log n)$-where n is the number of models [200].

5.5 SACube in Digital Twin Traffic System

In this section, we investigate employing the SACube framework in the DT of a traffic system. DT models are complex, and the computational cost of using such high-resolution models is often excessively expensive, resulting in the system failing to meet the deadline. Figure 5.18 provides a visual representation of the implementation of the SACube approach within the context of the real-time twining paradigm. In Figure 5.18, the actual system is the system under study. In order to transform into a DT, it is necessary to enable bidirectional communication from the actual system to the DT and vice versa, such as data and networking from the actual system to the DT and information process from the DT to the actual system. To implement self-adaptation in DT systems, the MAPE-K feedback control loop applies to the digital representation of the actual system.

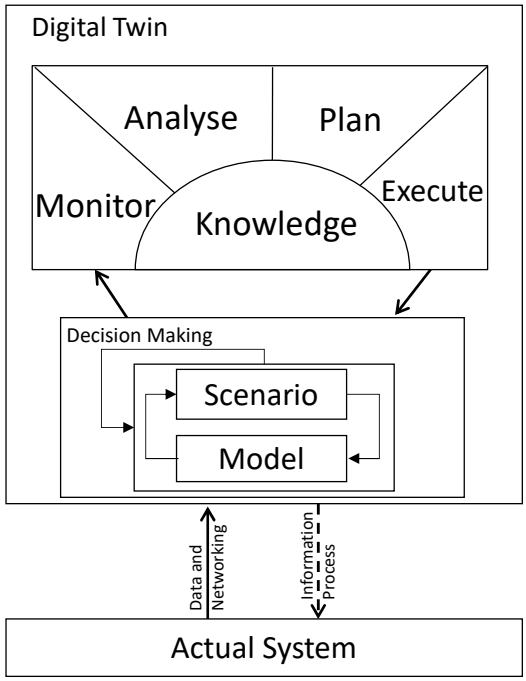

Figure 5.18: Schematic overview of using SACube in DT.

As compared to the frequency of operation of the DT, the frequency of operation for the feedback control loop is significantly higher.

While the highway lane change system can be classified as a DT, we apply our proposed techniques and approaches to the Digital Twin Traffic System, which features different applications and requirements. This section addresses RQ3, answering the question: How can the Self-Adaptive Abstraction and Approximation approach contribute to meeting real-time decision-making requirements within the context of the digital twin paradigm?

Traffic systems are fundamentally large-scale, dynamic, and complex, characterised by the interactions of autonomous agents (vehicles) that adapt to their surroundings and each other. Simulating and optimising these systems necessitates models that accurately represent their complexities at scale. Although agent-based simulation offers high accuracy, it has scalability limitations [201, 202]. These restrictions become most evident when optimisation must occur in real-time. Delays in running the simulation can result in missed deadlines, rendering the optimisation ineffective.

This means our optimisation problem is also a real-time problem. The timing of the decision-making process is just as important as the decision itself [24]. Using the high-resolution model to find optimisation results will likely lead to missed deadlines.

A potential solution to this challenge is adaptive abstraction and approximation. Using this approach can improve computation time. This approach maintains a collection of models, each tailored to different levels of abstraction or approximation. Since these models perform optimally under specific conditions, the optimiser dynamically selects

the most appropriate model for the current scenario [25].

For example, consider two abstracted models of a traffic system: one designed for high accuracy in dense traffic conditions and another for scenarios with lighter traffic. The optimiser can select one of these models based on the current traffic situation. Because these models are computationally less expensive than the high-resolution simulation model, we can generate control commands in a more reasonable time.

In this section, we design a digital twin to optimise a traffic simulation in real-time by implementing adaptive abstraction and approximation.

5.5.1 Experimental Set up

Our setup is composed of three interconnected components. The first component is the actual system, a traffic system that serves as the subject for optimisation. The second component of our setup is a DT integrated with the actual system as a virtual representation of the actual system that captures and reflects its properties, conditions, and behaviors. By collecting data from the actual system and processing it using simulation models, the digital twin provides a platform for analysis and decision-making [203]. We use this DT to generate the optimisation commands for our actual system. The final component of the setup is the library of abstracted and approximated models, which forms an integral part of the digital twin. These models are used to perform the optimisation step. A block diagram of the full setup can be seen in Figure 5.19.

5.5.1.1 The Actual System

During this study, we optimise a system represented by a SUMO model. This model is considered the actual system. SUMO is a micro-level traffic simulator, which means that each vehicle and its dynamics are modeled individually. It has several main advantages. First, it allows us to generate a large-scale map of a certain area. By importing a map from OpenStreetMap, we don't have to spend time building a large-scale network from scratch [204]. Next, using TraCI, we can connect to an ongoing simulation. This allows us to collect simulation data and update the simulation as it runs. TraCI allows us to update the traffic light configuration of our simulation during runtime. Our control commands are essentially a set of updated traffic light configurations [205]. SUMO also allows us to keep a clear record of the simulation results. We can see the total average waiting time of all agents in the system. We define the optimisation of the traffic simulation as minimising waiting time.

The map we optimise covers a large part of Wilrijk, a district in Antwerp, Belgium, as seen in Figure 5.20. When importing an OSM map in SUMO using Netconvert, all lanes, roads, and traffic lights are generated automatically. However, the exact configuration of these traffic lights is automatically assigned. We generate random traffic using a script given by SUMO. This traffic is defined as a large set of vehicles taking random routes [204]. While this often leads to a poorly optimised system with high congestion even in low-density situations, it is an invaluable tool for the rest of this experiment. Specifically, we use this tool to generate a large number of distinct scenarios, which are required for testing and

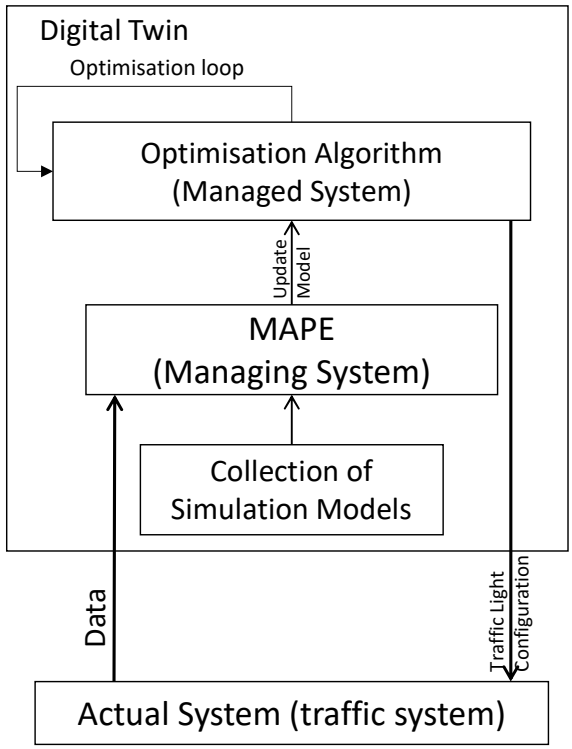

Figure 5.19: Block diagram of the full architecture for the traffic system DT.

validating our approach throughout the rest of the study.

To compare the results of different simulation runs, we create one main scenario to be used in varying conditions. First, we run the main scenario to determine a baseline for the total waiting time without optimisation. Next, we run the scenario and pass the state to our digital twin for optimisation. Every 500 time steps, a state is generated, with a time step in SUMO representing one second. As a result, our optimiser will generate a set of traffic light configurations every 500 seconds. To determine our optimiser's performance, we compare the total waiting time of the optimised scenario to the results without optimisation.

We start the simulation using a Python script, which we also use to initialise the digital twin. Every 500 time steps, the script calls the digital twin to start the optimiser. We notify that a new state is ready in a shared folder. The script also keeps track of a shared data structure to check for a new configuration at every time step. When a new configuration is received, the SUMO simulation is updated accordingly.

Figure 5.20: Covered traffic system in SUMO.

5.5.1.2 Digital Twin

Our digital twin consists of a set of predictive models, an optimisation algorithm, and the most recent data from the actual system. In this experiment, both the real system and its digital twin run on the same machine. To increase the realism of the system, the digital twin is called to run in a separate thread from the main simulation. It has access to the same location where SUMO stores save states. The optimised parameters are returned to the simulation via a synchronised Python queue.

The digital twin is a self-adaptive system. Therefore, it consists of two parts: a managed system and a managing system [206, 207]. The managed system is the part that interacts with the environment, whereas the managing system performs the adaptation logic. In this setup, the managed system is an optimisation algorithm that generates and exchanges control commands with the traffic system. The managing system is a MAPE control loop. The control loop is the DT component that is called when the simulation requests a new configuration set. It selects a model to use and initiates the optimisation algorithm.

The digital twin uses adaptive abstraction and approximation and implements the MAPE control loop to switch between models dynamically [207, 25]. First, the function verifies if there is a saved state in the shared storage location. This is the monitor step. We analyse the state file by extracting relevant data. We store the list of active vehicles per lane. We use a decision-maker that selects a model by comparing the number of active vehicles to a static value. The static value is chosen as the expected point where one model begins to outperform the other. During the planning phase, the objective function of our optimiser is updated to use the correct model. Finally, in the Execute phase, we run the optimiser to generate a new traffic light configuration for the traffic system. Figure 5.21 shows a block diagram for the implementation of the MAPE control loop.

The digital twin accesses two models. One is more accurate in low-density situations, whilst the other performs better in high-density situations. We expect the high-density model to be more accurate when the number of vehicles exceeds a particular threshold.

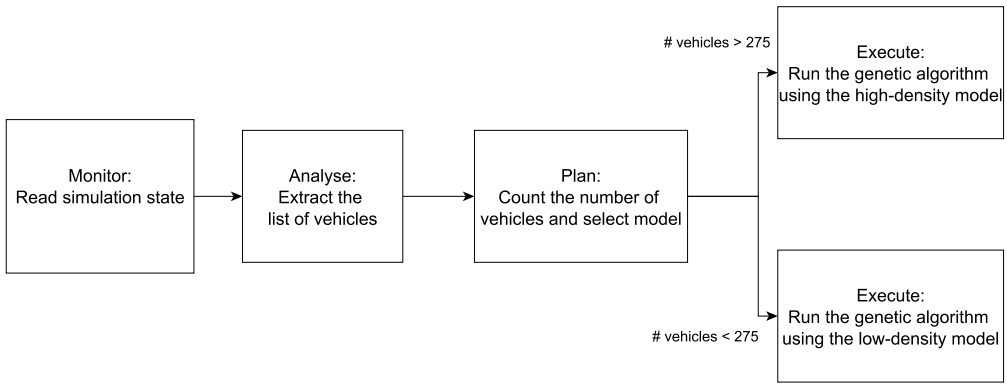

Figure 5.21: Block diagram showing MAPE decision model.

Once the number of vehicles in the system surpasses 275, the high-density model is used.

Finally, the digital twin contains an optimisation algorithm. For this function, we used a genetic algorithm. The algorithm starts by generating a set of random configurations. One of our predictive neural networks serves as the objective function. The algorithm will run for 50 generations, containing 15 individuals each.

5.5.1.3 Models

The abstracted and approximated models complete the setup. These models should be computationally less expensive than the high-resolution simulation model. However, combined, they should be reasonably accurate over the entire domain of the actual system. In our setup, we use two neural networks as predictive models. A neural network can serve as a surrogate model in an agent-based simulation. A neural network can approximate the general outcomes of a simulation with a model that is complex enough and with enough training data [208]. While these models take time to train, they are eventually significantly faster than running the high-resolution model. Building a neural network takes significantly longer at design-time. However, we make a trade-off to ensure that the model performs well during runtime.

The input vector for both networks will contain the current configuration of traffic lights and the number of vehicles per road. The output will be a single value representing the predicted total waiting time for the next 500 seconds.

5.5.2 Training Data

The first step to building a neural network is to have a set of training data. Because data sets for our purposes are not readily available, we generate two sets containing a large number of scenarios with random traffic and random traffic light timings. We run all these simulations to obtain a set of input-output combinations for our training data. Since we want both neural networks to be accurate in different contexts, we generate

	Low-density	High-density
Minimum	0	200
Maximum	354	722
Average	181.9295	414.3440

Table 5.2: Number of active vehicle distribution in training datasets

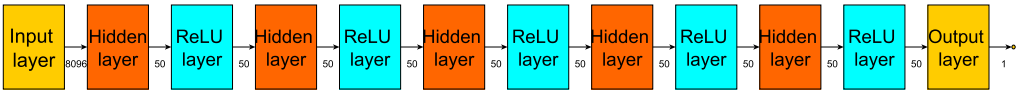

Figure 5.22: Schematic of the neural network configuration

two distinct datasets. The first set includes scenarios with little traffic, while the second contains scenarios with significant traffic density. We define density as the total number of cars in the system. Table 1 shows the total number of vehicles in the system across both datasets. The transition point for the digital twin is chosen to be in the middle of the overlapping area between both networks.

The datasets are generated by creating a scenario with a constant traffic light configuration and traffic demand. For each scenario, we saved 15 states. These states serve as input values for our dataset. Each of these states is then simulated for 500 time steps. The waiting times after the simulation are the output values in the dataset. The dataset for training the low-density neural network has 6000 data points. The dataset for the high-density network has 6985 points.

5.5.3 Neural Network Configuration and Training

In this experiment, the neural networks are implemented using d2l multilayer perceptrons in Pytorch [209] with several parameters based on the neural network used by Angione et al. [208] in their research. Both networks contain 11 layers in total. Five of these layers are hidden layers with 50 nodes each. For every hidden layer, there is a ReLu activation layer present. Finally, there is an input layer with 8096 nodes and an output layer with one node. A schematic of the full network can be seen in Figure 5.22.

During training, the learning rate is set at 0.0003. Both networks are trained for 1000 epochs with a batch size of 32. We use the mean absolute error (MAE) as a loss function to determine the accuracy of our models. After training, the neural networks give an average error on the validation set of about 30-50.

5.5.4 Results

Using the full setup, we first run the simulation without optimisation. This provides us with a baseline from which we can begin to analyse the optimisation achieved with the digital twin. We also run several simulations with only one neural network. Finally, we look at the results when simulating with both neural networks present. This way, we can

	Low-density	High-density	Both
Minimum	607.29	589.73	575.36
Maximum	704.77	701.78	689.14
Average	659.12	632.94	643.51
Improvement	11.7%	15.2%	13.8%

Table 5.3: Resulting total waiting time after optimisation

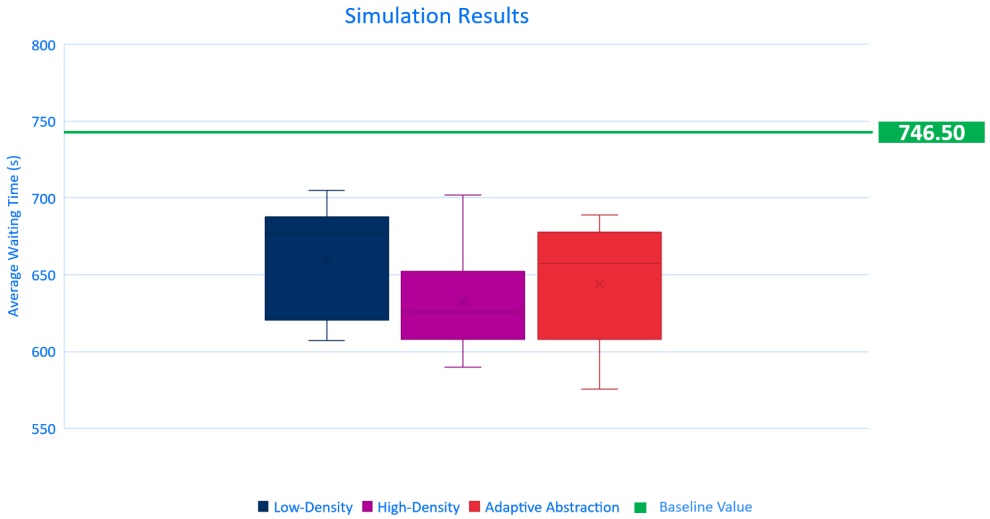

Figure 5.23: Comparison of running different simulation models

get an idea of the improvement we get with just one neural network, and compare it to the situation where we perform adaptive abstraction. To ensure accurate comparisons, all simulations begin with the same scenario.

Figure 5.23 shows the results of running our simulation 10 times using different conditions. First, we run only the low-density network, then only the high-density network, and finally, using both networks for optimisation. It shows the best, worst, and average results over all runs. Table 5.3 shows the average improvement percentage compared to the baseline.

5.5.5 Discussion and Limitations

In this section, we demonstrated the feasibility of a digital twin using adaptive abstraction and approximation in real-time optimisation. While our approach consistently outperformed the baseline simulation, the results deviated from our initial expectations. Specifically, we anticipated that optimisation would be more effective when utilising two NN models simultaneously. The rationale behind this hypothesis was that combining both models would yield more accurate predictions, ultimately leading to improved and more stable optimisation. However, our findings did not reveal a clear advantage of using both models over a single one. Several factors could explain this outcome.

One potential explanation lies in the accuracy of the Neural Network models themselves. It is possible that the results of our models are not distinct enough to produce significantly different results. As mentioned before, the exact prediction of our model is not as important to get a good result. As long as the model can identify which configuration would perform better, the genetic algorithm will return this as a new configuration. Both models may give similar configurations a lower prediction even though one is technically more accurate.

Another possibility is statistical variance. Given the considerable spread in our results, the number of simulations conducted may not be sufficient to capture the true average performance. A larger set of simulations could still align with our initial hypothesis over time.

Finally, the results could be caused by the unexpected performance of our models. As mentioned earlier, the high-density model might outperform the low-density model in situations where a lot of vehicles are concentrated in only a few lanes. Because the total number of vehicles is relatively low in these scenarios, our digital twin uses the low-density model. This could lead to the simulation with both models performing worse than the simulation with only the high-density model and better than the simulation with only the low-density model. We can see this possibility reflected in the results.

5.6 Conclusion

This chapter introduces the SACube framework, a novel approach that addresses the computational challenges of DT systems operating under real-time constraints. By employing adaptive abstraction and approximation, the framework enables dynamic model switching to balance computational efficiency with decision accuracy. Ensuring reliable model switching based on contextual requirements. The framework's effectiveness was evaluated using a highway lane change case study, which showed considerable reductions in computational overhead while maintaining accurate decision-making.

The findings underscore the potential of context-aware surrogate modelling as a key enabler for scalable, real-time DT applications. Further, we extended the investigation by applying SACube to a DT representation of a traffic system, exploring its adaptability in a broader, more dynamic environment.

Ultimately, SACube represents a significant step forward in scalable digital twin architectures, offering a promising pathway toward computationally efficient modelling in complex, dynamic systems with real-time constraints.

Chapter 6

Towards a Validity Frame of Multi-Modal Surrogate Models for Traffic Simulation

This chapter is based on the following publication:

R. Biglari, C. Gomes, and J. Denil, "Towards a validity frame of multi-modal surrogate models for traffic simulation," in *2025 Annual Modeling and Simulation Conference (ANNSIM)*, 2025, in press

Complex systems such as traffic systems, climate simulations, or multi-physics industrial processes include different system and environment modes, each requiring specific modelling considerations. For example, the traffic system of a city needs to consider multiple modes, such as peak rush-hour congestion and off-peak flow patterns, to ensure a complete representation of real-world dynamics. To address this, surrogate modelling, particularly machine learning-based surrogate modelling [208] serves as a practical method to develop simplified models with reduced computational cost for different modes, but switching between modes requires careful management of the models' experimental frames. ***The experimental frame of a model*** *is the set of inputs for which the model can provide valid predictions*. One of the challenges is to ensure that the experimental frames of the models are non-overlapping. This is crucial to avoid the extra complexity in the models of each mode, and to make it clearer which model to use at any given time.

The validity frame of a model *is defined as its experimental frame, plus a process that allows us to change the experimental frame*. Such process must include minimising the overlap between the experimental frame of the model with respect to the other experimental frames in a multi-model setup.

In this chapter, we show one possible approach to define such a validity frame. To illustrate this, we create multiple experimental frames, each belonging to a surrogate model. Each surrogate model is specifically designed to operate within a distinct validity region, enabling efficient and accurate predictions across different system and environment modes. In addition, we describe the process used to define the experimental frame of each model.

As part of our methodology, we also address the key challenge of building complete and non-redundant experimental frames. That is, the union of our models' experimental frames needs to form a complete system operating domain. Achieving this requires access to a complete dataset, which is derived by ensuring the proper distribution of data across the system's operational and environmental modes. This partitioning approach ensures complete coverage of model validity while reducing computational costs and improving scalability.

We choose the traffic system as a case study using SUMO (Simulation of Urban MObility) [210], a renowned open-source micro-traffic simulator, suitable for optimising traffic light configurations. Additionally, available domain knowledge about the two main systems' operating modes enables us to easily propose two surrogate models.

6.1 Approach

This section introduces an approach to define the validity frame of the surrogate models using the asset of a machine learning-based surrogate modelling through a traffic simulation use case. It also details the experimental setup, data generation process, and model development, addressing key challenges such as ensuring dataset diversity, defining surrogate models' experimental frames, and minimising redundancy in training data.

6.1.1 Problem Description

Traffic systems are complex systems, and their simulations necessitate computationally expensive models, constraining their scalability for practical implementations. We aim to predict the average total waiting time of all cars in the traffic system by simulating a large part of Wilrijk, a district in Antwerp, Belgium, as seen in Figure 6.1, with deep neural network models trained on SUMO-generated data.

Figure 6.1: Covered traffic system in SUMO.

6.1.2 Experimental Setup

We model the traffic system for our intended map by using SUMO. SUMO is a micro-level traffic simulator, meaning each vehicle and its dynamics are modeled individually. It has several main advantages. First, it allows us to generate a large-scale map of a certain area. The road network is imported directly from OpenStreetMap (OSM) [211], eliminating the need to build a network from scratch. Next, using the Traffic Control Interface (TraCI), we can connect to an ongoing simulation. This allows us to collect information about the simulation and to update the ongoing simulation. Traffic scenarios are generated using SUMO's randomTrips.py, which generates a set of random trips for a given network. In the absence of real-world data; therefore, we generate data for our traffic system and run the simulation. SUMO allows us to keep a clear record of the simulation results. We can view the total average waiting time of all vehicles in the system.

6.1.3 Surrogate Models

Drawing from domain knowledge, we focus on two distinct traffic modes: rush-hour mode, characterised by congestion and high traffic density, and off-peak mode, with lower traffic density. For each mode, we develop a dedicated surrogate model: the low-density model, trained on low-density data, and the high-density model, trained on high-density data. While the low-density model performs better during off-peak periods, the high-density model is expected to be more accurate when the number of cars surpasses a certain threshold, making it more suitable for rush-hour conditions.

The input vector for both networks will contain the combination of the traffic light configurations and the number of cars per road. The output will be a single value representing the total waiting time for the next 500 seconds derived from running the simulation. By dividing the data into separate density categories, each model can specialise in a specific region of the problem.

6.1.4 Data Generation

Since the training data is not readily available for our experiment, we generate training data for the surrogate models by simulating several traffic scenarios with differing traffic densities and traffic light configurations. To generate low-density scenarios, we use SUMO's randomTrips.py and *--insertion-density* argument = 15. This argument is the number of vehicles per hour per kilometer of road that the user expects. And to generate high-density scenarios, we use SUMO's randomTrips.py and *--insertion-density* argument = 25. These values are selected based on calculations of the number of cars at high-demand intersections such as shown in Figure 6.2. Through testing different settings, we observed that a value of 25 resulted in significantly denser traffic, whereas a value of 15 produced lighter traffic flow. Although there may be alternative methods for identifying high-demand regions that could further enhance this research. This is, however, outside the scope of this research.

For each scenario, we saved 15 states. These states serve as the input values in our dataset. Each of these states is then simulated for 500 time steps. The waiting times after

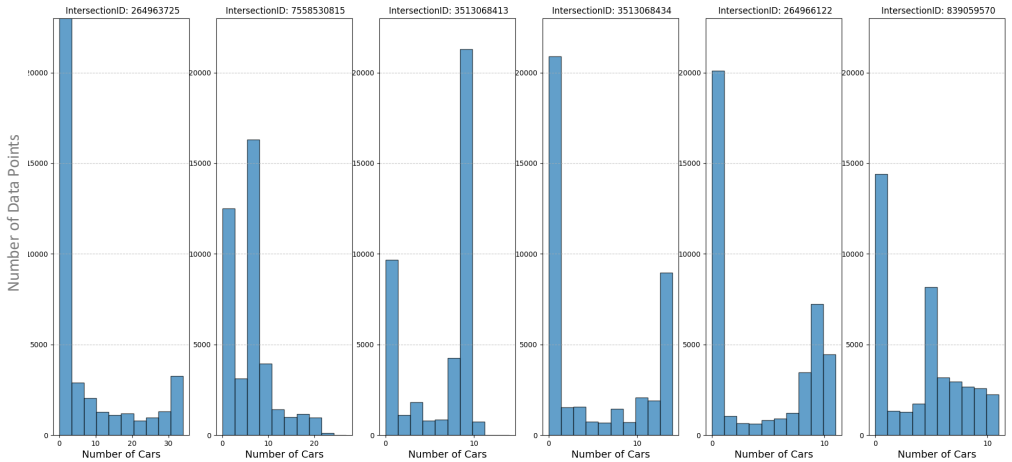

Figure 6.2: Distribution of vehicle counts across high-traffic intersections over 600 simulation runs demonstrating balanced and diverse data coverage.

the simulation are the output values in the dataset. Separate datasets are prepared for the low-density and high-density models, ensuring each model is trained on data relevant to its intended use case.

6.1.5 A Complete and Balanced Experimental Frame

For a model to be accurate, it needs to operate within a clearly defined experimental frame. These boundaries outline the range of conditions and scenarios it is designed to handle accurately.

An essential element of establishing an experimental frame is ensuring it includes the complete range of scenarios the model is expected to face in its intended use. This does not imply that the model must include every potential scenario, but rather, it should reflect the range of actual conditions it is likely to manage. In machine learning-based surrogate modelling, if the model's training data or calibration does not cover crucial scenarios, such as unusual or extreme events, it may fail when these situations occur. For example, a traffic simulation model considered exclusively normal traffic hours may have difficulty predicting traffic during a concert in the city center or a football match. A complete experimental frame needs:

Understanding the system: Clearly identifying the system's goals, properties of interest, conditions, and modes that describe the system under study.

Balanced Data: A complete experimental frame needs balanced data, which does not mean all possible situations but rather representative and diversified data. Therefore, there is a need for uniformity in data, which means the dataset needs to be uniformly distributed over situations, scenarios, and features depending on the specific context and application of the system under study.

In this experiment, the dataset must accurately represent varied traffic configurations and include the most pertinent and commonly observed traffic light configurations. Moreover, careful consideration is devoted to preserving consistency in data distribution to prevent biases. This approach guarantees the model stays robust and reliable across various expected operational scenarios. A diverse dataset helps accurately reflect the system's behavior, avoiding the overrepresentation of specific subsets and promoting a balanced understanding of traffic dynamics.

To ensure that we have balanced data in this use case, we compare high-demand intersections with heavy traffic. We visually inspect the distribution with a histogram shown in Figure 6.2, which verifies data distribution aligns with the required balance and diversity. We could focus on all possible traffic light configurations, but in practice, it's not computationally possible since there are 186 traffic lights in this map, each having at least 3 phases, and for each phase, there are many duration possibilities. Therefore, we chose to focus on high-demand intersections.

Furthermore, to build a complete experimental frame, the combined experimental frames must cover the complete operational domain of the system under study. This ensures the surrogate models collectively keep the accuracy and robustness necessary for reliable system representation across all operational scenarios.

6.1.6 Minimally overlapping experimental Frames

In the context of surrogate models, it is essential to define distinct experimental frames for each model. To ensure clear model selection and eliminate ambiguity, these experimental frames must be minimally overlapping. This guarantees that each model matches perfectly to a unique subset of the problem space, consequently avoiding prediction conflicts.

Achieving minimally overlapping experimental frames requires creating corresponding minimally overlapping datasets for training the models, since we are using surrogate models. This means the data used to train one model should not overlap with the data used for another. In this experiment, we divided the dataset into two categories, shown in Figure 6.3.

Additionally, avoiding redundancy helps ensure that the surrogate models are trained on the most informative data, allowing them to perform accurately and efficiently within their respective experimental frames. To achieve low redundancy in the dataset, there are two main approaches: 1) Randomly removing data points from the dataset. 2) Removing data points with high similarity to others. In this research, we use the second approach, which prioritises the removal of similar data points to ensure diversity in the dataset. To determine similarity, we define a notion of distance between data points. We use the most familiar distance metric, n-dimensional Euclidean distance. The number of roads in our network plus the number of phases for all traffic lights in our map is n, and the calculation of distance takes $O(n)$.

In this experiment, each data point represents a combination of the traffic light configurations and the number of cars per road. The Euclidean distance between two points is calculated based on these attributes. If the distance between two points is small, they are

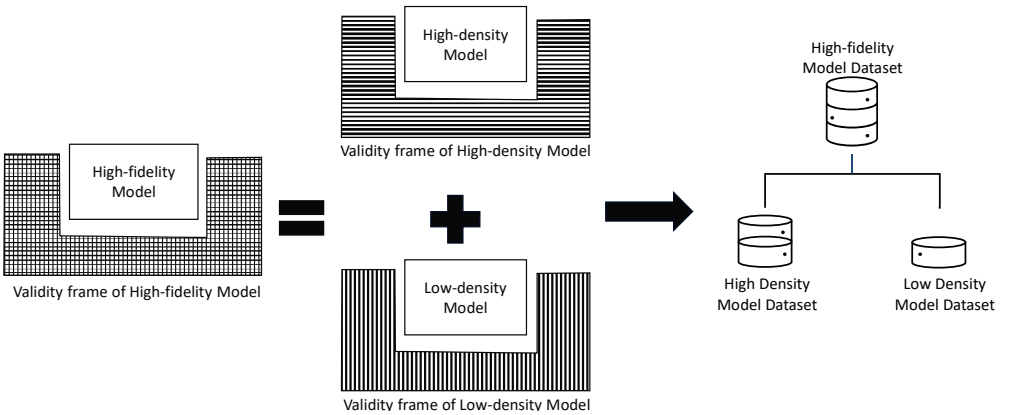

Figure 6.3: Minimally overlapping experimental frame and minimally overlapping datasets.

considered similar, and one of them is removed to reduce redundancy. In this example, we eliminate 8,046 out of 8,999 samples identified as redundant, based on observations from the heatmap graph in Figure 6.4, where a distance of 50 appears to indicate data points that are too close to each other. A similar approach is applied to the low-density dataset, where we remove 7,710 out of 8,999 data points. It is important to note that the distance metric is domain-specific and varies depending on the particular use case. The calculation of these distances between P data points is $O(P^2)$.

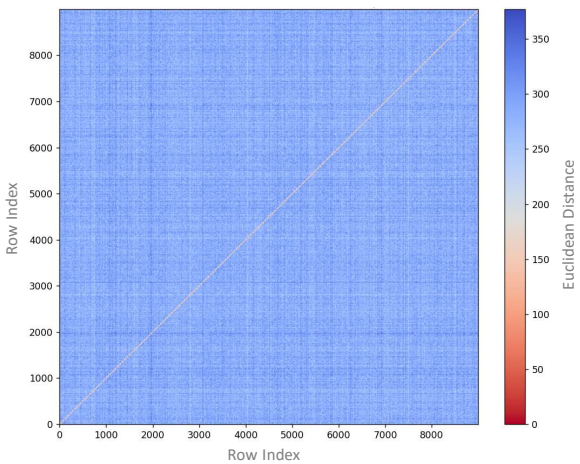

Figure 6.4: Pairwise distance heatmap for the high-density dataset.

After removing redundant data points, we retrain the model depicted in Figure 6.5 and Figure 6.6, showing the retrained model compared to the previous model for the high-density and low-density models, respectively. The figures show the model predictions before and after removing the redundancy.

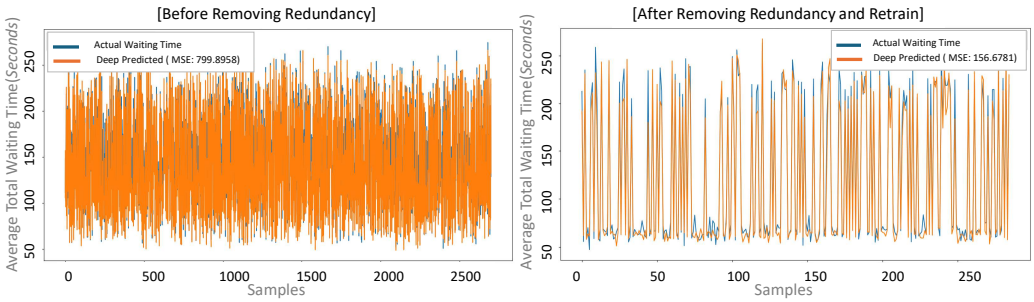

Figure 6.5: Removing redundancy and retraining high-density model (in the right figure, about eight times fewer samples are used in the training).

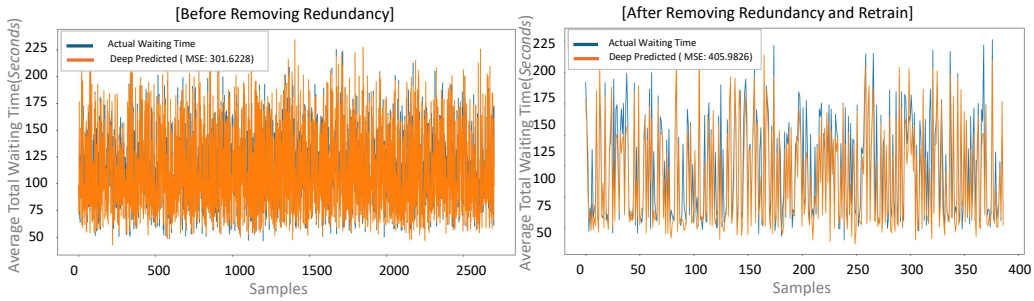

Figure 6.6: Removing redundancy and retraining low-density model (in the right figure, about seven times fewer samples are used in the training).

6.1.7 Predictive Accuracy for Interpolation and Extrapolation

We perform a complete evaluation to validate the model's predictive capabilities. We check the interpolation (predictions within the range of training data), results shown in Figure 6.7 for the high-density model and in Figure 6.8 for the low-density model. These results demonstrate that–even after partitioning the dataset and removing redundant data–the model keeps making accurate predictions. However, when we check extrapolation (predictions outside the range of training data), we do not get correct results. This outcome aligns with what we focus on: that predictions outside the experimental frame of the model are unreliable. Extrapolation is scenarios with *--insertion-density* argument = 50, which creates a very dense traffic situation and falls outside the validity region of the high-fidelity model.

6.2 Related Work

The increasing complexity of simulations and the need for efficient computational methods have driven significant advancements in techniques for surrogate modelling, adaptive abstraction, and runtime model integration. These approaches aim to balance precision, computational cost, and scalability while maintaining the validity of simulation results

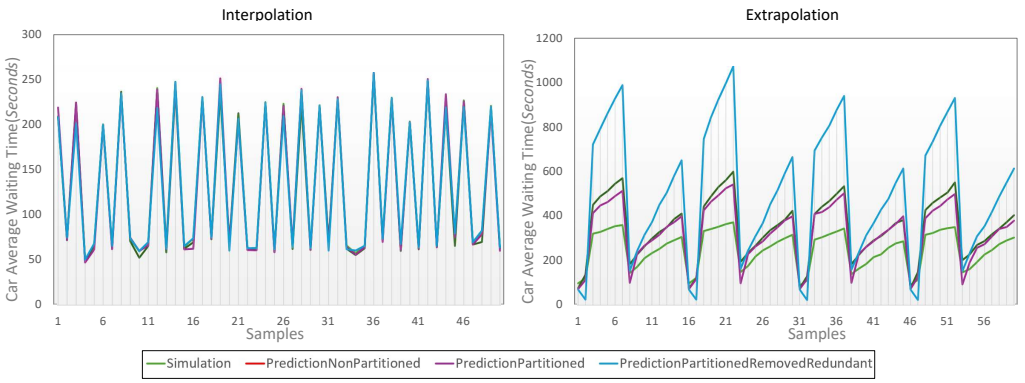

Figure 6.7: Comparison of interpolation and extrapolation predictions for the high-density model.

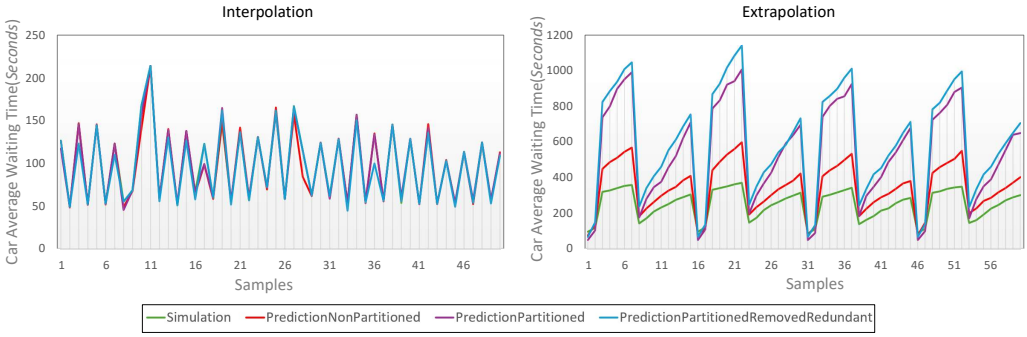

Figure 6.8: Comparison of interpolation and extrapolation for the low-density model.

across diverse domains such as agent-based modelling, traffic systems, and building energy simulations. We briefly summarise relevant work in machine learning for surrogate modelling and adaptive abstraction techniques, emphasising their relevance to our focus on ensuring the completeness and reliability of experimental frames in dynamic simulation environments. The problem of dynamically switching surrogate models can be seen as a hybrid system identification problem, and we refer the reader to [170] for an introduction to the topic.

6.2.1 Machine Learning for Surrogate Modelling

The work in [208] investigates the use of machine learning methods as surrogate models for agent-based models (ABMs), focusing on reducing computational costs in sensitivity analysis and parameter calibration. Surrogate models replace computationally expensive ABMs with statistical models that replicate their behavior, enabling faster evaluations. The study compares various ML methods, including neural networks, gradient-boosted trees, and Gaussian processes, and finds that artificial neural networks (ANNs) and

gradient-boosted trees outperform traditional Gaussian process surrogates, especially in capturing non-linear or chaotic behaviors. It advocates for further development of ML-based surrogate models, particularly for complex, policy-relevant ABMs, and suggests that these techniques could democratise access to detailed model analyses in resource-constrained environments. Moreover, Ribeiro et al. introduce the local surrogate model, which is a simplified model trained to approximate a complex model's behavior around a specific input or region [212].

For an extensive review of constructing neural network-based models for simulating dynamical systems, we highlight the surveys in [213]. These surveys provide a comprehensive discussion of data-driven modelling techniques and challenges, particularly those related to dynamical systems.

Similar to our work on quantifying and characterising validity frame from a methodological point of view, we highlight the work [214, 215, 25] where validity frames are computed/quantified for different models. With a focus on neural networks, the work of [215] highlights the importance of distinguishing interpolation (reliable predictions within the training data range) from extrapolation (unreliable predictions outside the data range). To tackle this, the study proposes a methodology to calibrate novelty detection algorithms specifically for extrapolation detection in ML models. This approach focuses on ensuring reliable model predictions within the validity region while highlighting the risks of extrapolation in building energy systems.

6.2.2 Adaptive Abstraction and Approximation

The work of Bosmans et al. [201] explores a strategy for reducing computational costs in large-scale simulations, particularly those modelling complex systems like traffic networks or IoT systems. It introduces an opportunistic model approximation technique leveraging information theory to identify and abstract areas of the simulation that contribute minimally to global behavior. These low-information regions are dynamically transformed into less detailed representations, effectively reducing computational demand while preserving the validity of the emergent behaviors at the global scale. The technique involves entropy-based transformations, where entropy measures the informational complexity of simulation areas. Regions with lower entropy—indicating less influence on overall system behavior—are approximated to save computational resources.

The paper in [216] discusses adaptive abstraction techniques in agent-based simulations, emphasising their potential to enhance computational efficiency without significantly compromising result accuracy. Adaptive abstraction dynamically switches between levels of detail in simulations, depending on context, to strike a balance between execution speed and model precision. Using a traffic simulation case study, the authors explore two major patterns—PUPPETEER and ZOOM—for aggregating and disaggregating agent data. While PUPPETEER preserves detailed individual states, ZOOM relies on statistical aggregation, sacrificing some detail for computational efficiency. Results demonstrate that adaptive abstraction can effectively reduce computational costs while maintaining valid emergent behaviors, such as traffic jams, though domain-specific design decisions significantly influence outcomes.

Bosmans et al. [174] serves as motivation to the importance of our own work. It presents a

framework for adaptive multi-level traffic simulation, combining micro-level (individual vehicles) and meso-level (aggregated groups) models to balance computational cost and simulation accuracy dynamically. By leveraging the MAPE-K feedback loop and experimental frames, the framework assesses the validity of meso-level models in real-time and switches to micro-level models when higher fidelity is required. Using a custom-built simulation environment, the authors demonstrate the approach in urban traffic scenarios, showcasing significant improvements in computational efficiency while maintaining simulation validity. Results indicate that the dynamic approach outperforms static hybrid simulations in balancing performance and error rates.

With a focus on run-time model swapping, the work in [217] explores a mechanism for dynamically integrating and swapping models during simulation time without disrupting the simulation. It introduces a framework that facilitates the replacement of individual models or structural changes in coupled-models, enabling updates based on evolving system requirements. While the proposed method can be used for adaptively varying the abstraction level at runtime, as done in [216] and [174], our work is more focused on identifying the triggering conditions for model swapping.

Our work builds on prior research in surrogate modelling and adaptive abstraction by shifting the focus toward constructing non-overlapping experimental frames and formalising the concept of a validity frame. Unlike existing methods that often rely on runtime model switching or overlapping data regions, we propose a data-driven approach to partition the validity region upfront. This enables cleaner model boundaries, reduces redundancy, and supports more interpretable and efficient surrogate modelling in multi-modal systems, as demonstrated in our traffic simulation case study.

6.3 Discussion and Limitations

Choosing Between Linear Regression and Deep Neural Networks: When designing a predictive model, the choice between using Linear Regression (LR) or Deep Neural Network (DNN) depends on the complexity and characteristics of the problem. In this challenge, we discuss selecting the appropriate approach based on the structure of the dataset and the predictive requirements of the traffic system model.

For this experiment, we developed both an LR model and a DNN model to evaluate their performance. Figure 6.9 illustrates the results of each model. The data structure and requirements align more closely with DNN's capabilities, making it the most appropriate option for this experiment. Since the training duration of the DNN model is longer than that of the LR model, we run the training on an NVIDIA GeForce MX450 GPU instead of a CPU using CUDA, which is a parallel computing platform and programming model developed by NVIDIA.

Balanced Data: The distribution of randomly generated datasets often depends on the specific process used for data generation, which can lead to imbalanced data and consequently limit the model's abilities to achieve optimal generalisation. Therefore, we changed the dataset by generating different traffic light configurations to keep the duration phase of 90 seconds for each traffic light and the yellow phase between 3 and 6 seconds according to ADOT Traffic Engineering Guidelines and Processes, section

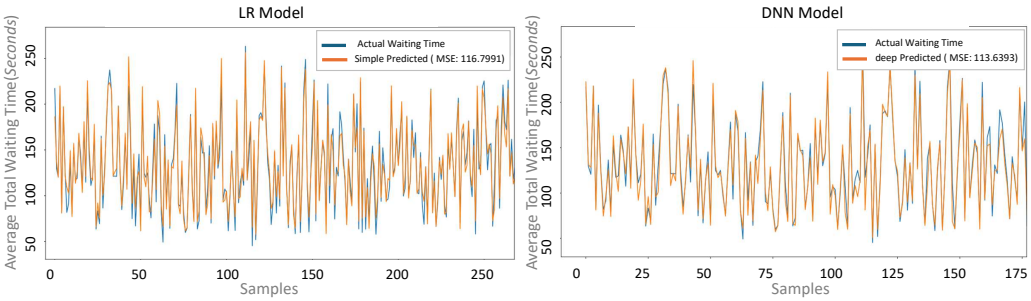

Figure 6.9: Zoomed-in view of linear regression and deep neural network models.

600 [218].

Choice of Metric: In this use case, we chose Euclidean distance as the metric space. Euclidean distance measures the straight-line distance between two points in space. However, this metric is not universally suitable for all use cases. Its effectiveness depends on the domain and data characteristics. For example, in high-dimensional spaces or non-linear data, alternative metrics (e.g., cosine similarity or Manhattan distance) might be more appropriate. The choice of metric should align with the specific requirements and structure of the data in the application.

Computational Considerations in Pairwise Distance Calculation: Calculating pairwise distances between every data point in the dataset can be computationally expensive and may pose scalability challenges. To mitigate this issue, an alternative approach is to randomly sample data points and compute their distances rather than performing a full pairwise comparison. This strategy ensures that redundancy and overlap in the dataset can still be effectively identified while reducing computational overhead.

Boundary and overlapping Regions: The focus is on minimising the overlap between experimental frames as much as possible; however, some overlap remains. Moreover, there are some boundaries. These areas introduce ambiguity in model selection and highlight the need for further investigation to improve decision-making in these regions.

6.4 Conclusion

This chapter presented an approach to surrogate modelling for multi-modal systems, focusing on traffic simulations. We defined the concept of the validity frame as an experimental frame plus a process that allows us to change the experimental frame. In this thesis, we introduced a method to define this validity frame by partitioning the system operating domain into distinct experimental frames and training separate surrogate models on minimally overlapping datasets, enabling more efficient simulation while maintaining predictive accuracy within each model's intended scope.

In this research, we showed how to construct surrogate models for low and high-density traffic scenarios using data generated in SUMO, and we demonstrated how balancing

the dataset and reducing redundancy improves model performance. Our approach ensures that the union of the experimental frames forms complete and non-redundant experimental frames.

While this method improves scalability and clarity in surrogate modelling, overlap at the boundaries between modes poses the challenge of uncertainty in choosing the appropriate model. Future work will focus on refining model-switching mechanisms, exploring adaptive and runtime approaches for validity frame management, and expanding this method to more complex or real-time traffic systems.

Chapter 7

Conclusion

DTs are complex systems that require high-validity, computationally expensive models. These models, while necessary for capturing complex system behaviors, use input from sensors but also from other supporting processes, e.g., predictions over the state of its context, to come to a control decision. The decision processes are implemented in software that runs on embedded hardware and is commonly real-time constrained, meaning that the time at which the decision is taken is as important as the decision itself.

7.1 Summary

In this thesis, we deal with the contradiction of better performance and reduced cost in DT by allowing adaptivity at runtime and changing the underlying decision and estimation models such that they are sufficient for computing the control actions, but at the same time computationally less expensive. To achieve this balance, we use techniques that abstract or approximate models, which are practical and effective in reducing complexity without compromising the system's functional requirements. Chapter 2 provided detailed background, and in Chapter 3, we focus on identifying and analyzing challenges for this thesis.

The remainder of the thesis focused on addressing research questions listed in Chapter 1 to complete the development of self-adaptive abstraction and approximation.

In the following, we provide a concise summary of our findings. Furthermore, we outline potential directions for future research, providing insights into how our findings might serve as a foundation for further investigation and innovation.

7.1.1 Decision-Centric Technique (Chapter 4)

In Chapter 4, we propose a novel technique called Decision-Centric technique to find the validity frame of a model. SACube employs this technique to evaluate a model's validity. This method focuses on assessing the model's applicability instead of solely emphasising output similarity within the context of the decision-making process. Further, we use this technique to find the validity region map and address RQ2.

Our approach operates under a set of assumptions: We considered deterministic models and decision-makers and decisions are the primary concern. Moreover, we assumed that the clear validity regions exist.

However, while the approach demonstrates significant utility, it is not without limitations. The method assumes deterministic and state-independent decision-making, which may limit its application in systems with temporal or stochastic dependencies. Chaotic systems or systems with intricate path dependencies in decision-making processes present challenges that are not addressed in this research.

7.1.2 SACube Framework (Chapter 5)

Chapter 5 introduces our proposed framework and method using MAPE-K control loop while providing an in-depth analysis of the technique employed in this study to answer RQ1. Moreover, we have designed a digital twin to optimise traffic simulation in real-time by implementing adaptive abstraction and approximation, addressing RQ3, and see how the Self-Adaptive Abstraction and Approximation (SACube) approach can contribute to meeting real-time decision-making requirements within the context of the digital twin paradigm.

The framework operates under several key assumptions: the MAPE-K (Monitor, analyse, Plan, Execute, and Knowledge) loop's overhead is predictable and manageable; clear transition points exist between models, facilitating smooth operational changes; and models can be seamlessly substituted by setting initial conditions and parameters appropriately.

However, certain challenges remain, which underscore areas requiring further investigation. While this research touches on clean model transitions in long-running systems, the practical implementation of these transitions has not been fully worked out. Additionally, the analysis of stability during model transitions remains limited, raising questions about potential instabilities that may arise. Finally, integrating SACube into existing systems presents a layer of complexity.

7.1.3 Towards a Validity Frame of Multi-Modal Surrogate Models for Traffic Simulation (Chapter 6)

In Chapter 6, we introduce a novel approach that partitions the validity region of a high-validity model into minimally overlapping subregions, each represented by specific surrogate models. By addressing critical challenges such as complete and minimally overlapping validity frames, the introduced method demonstrates improved scalability, reduced redundancy, and reliable performance across various operational modes. The proposed ensemble of surrogate models, coupled with adaptive techniques, represents a significant step forward in the efficient modelling of dynamic and computationally expensive models. In this chapter, we address RQ4 to create a good set of approximated models by focusing on traffic simulation to demonstrate our contributions.

This approach is built on several key assumptions: the existence of natural system regimes that allow for meaningful partitioning; the ability to identify clear boundaries between

these partitions; and the capacity for models to specialise effectively within their valid regions. These assumptions form the foundation for implementing partitioned surrogate modelling, ensuring both computational feasibility and accuracy in predictions.

Despite its advantages, the partitioned approach is not without challenges. Significant effort is required to create training datasets for each partition and to identify or design optimal models for each region. This effort can be resource-intensive and may limit the scalability of the method in systems with highly dynamic or complex state spaces.

7.2 In Preparation

As part of our ongoing efforts to share the insights gained from this research, we are preparing several manuscripts for publication. Although final acceptance of these papers is not guaranteed, I would like to emphasise that the years of study documented in this thesis have helped make these follow-up studies possible.

A combination of Chapter 4 introducing decision-centric technique as one of the top findings of this research, with a use case of an autoclave system, is being marked up for a conference and would benefit the scientific community by solving the problem of models with unavailable validity [184].

Another paper under development is an integrated study that merges the insights and methodologies from Chapter 5 and Chapter 6. This study is under development and is being marked up for a journal introducing the validity of multi-modal surrogate models [219].

Moreover, as we found it crucial to focus on the quality of DT to help the scientific community benefit from it, we are working on DT quality, which is being marked up for the 2nd International Conference on Engineering Digital Twins [220]. This study emphasises the quality aspect of digital models, alongside the decision-maker, to determine which quality aspect is crucial for digital twins serving different purposes. Furthermore, we aim to identify any quality aspects that are absent in model quality and data quality but are present in DT systems.

7.3 Future Work

In this thesis, we made several contributions to facilitating modelling and simulation by using approximated and abstracted models instead of the high-validity, computationally expensive models. However, despite these gains, additional challenges remain, providing plenty of opportunities for future research.

One of the future research directions to broaden the applicability of the Decision-Centric Technique is to create techniques for validating path-dependent decisions where historical events or decisions constrain later events or decisions.

The other direction will focus on identifying and characterising optimal transition points using advanced techniques such as attractor analysis. Understanding stability during

transitions will also be critical, particularly in systems with complex or nonlinear dynamics. Furthermore, developing strategies for handling transitions in different regions of the state space will be essential to ensure robust and reliable performance in diverse operational contexts.

These developments will strengthen the SACube framework's potential as a practical and reliable approach for CPS and DTs with real-time constraints.

The other aspect of future work will focus on enhancing the practicality and adaptability of the partitioned modelling approach. Key directions include the development of automated methods for partition identification, such as using behavioral heatmaps or frameworks inspired by the Hybrid System Identification methods [170]. Additionally, designing adaptive model management systems capable of dynamically adjusting to changing system regimes will be critical. Applying optimal partition selection strategies, including trade-offs between partition granularity and computational overhead, will further refine the methodology.

Bibliography

[1] M. Grieves and J. Vickers, *Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems*. Cham: Springer International Publishing, 2017, pp. 85–113. [Online]. Available: https://doi.org/10.1007/978-3-319-38756-7_4 13, 34, 35, 36

[2] H. Vangheluwe and J. Denil, "'relating (multi-paradigm) modelling concepts' presentation, msdl research day 2025," 2025, antwerp, Belgium. [Online]. Available: http://msdl.uantwerpen.be/presentations/25.03.25.MSDLResearchDay/slides/HV_JD.pdf 13, 42

[3] W. L. Oberkampf and C. J. Roy, *Verification and validation in scientific computing*. Cambridge university press, 2010. 13, 37, 46, 65, 73, 85

[4] T. G. Trucano, M. Pilch, and W. L. Oberkampf, "General concepts for experimental validation of asci code applications," Sandia National Lab.(SNL-NM), Albuquerque, NM (United States), Tech. Rep., 2002. 13, 46

[5] J. Kephart and D. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003. 13, 48, 68

[6] R. Wilhelm, J. Engblom, A. Ermedahl, N. Holsti, S. Thesing, D. Whalley, G. Bernat, C. Ferdinand, R. Heckmann, T. Mitra *et al.*, "The worst-case execution-time problem—overview of methods and survey of tools," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 7, no. 3, pp. 1–53, 2008. 13, 50, 52

[7] D. Weyns, *An introduction to self-adaptive systems: A contemporary software engineering perspective*. John Wiley & Sons, 2020. 14, 48, 87, 88, 89, 91, 92

[8] Z. Ali, R. Biglari, J. Denil, J. Mertens, M. Poursoltan, and M. K. Traoré, "From modeling and simulation to digital twin: evolution or revolution?" *SIMULATION*, vol. 100, no. 7, pp. 751–769, 2024. 21, 25

[9] R. Eramo, F. Bordeleau, B. Combemale, M. v. d. Brand, M. Wimmer, and A. Wortmann, "Conceptualizing digital twins," *IEEE Software*, vol. 39, no. 2, pp. 39–46, 2022. 21

[10] F. Bordeleau, B. Combemale, R. Eramo, M. van den Brand, and M. Wimmer, "Towards model-driven digital twin engineering: Current opportunities and future challenges," in *Systems Modelling and Management: First International Conference, ICSMM 2020, Bergen, Norway, June 25–26, 2020, Proceedings 1*. Springer, 2020, pp. 43–54. 21

[11] L. Wright and S. Davidson, "How to tell the difference between a model and a digital twin," *Advanced Modeling and Simulation in Engineering Sciences*, vol. 7, no. 1, pp. 1–13, 2020. 21, 36, 41, 44, 58, 68

[12] K. Kušić, R. Schumann, and E. Ivanjko, "A digital twin in transportation: Real-time synergy of traffic data streams and simulation for virtualizing motorway dynamics," *Advanced Engineering Informatics*, vol. 55, p. 101858, 2023. 21

[13] F. Psarommatis, "A generic methodology and a digital twin for zero defect manufacturing (zdm) performance mapping towards design for zdm," *Journal of Manufacturing Systems*, vol. 59, pp. 507–521, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0278612521000765 22

[14] M. Singh, E. Fuenmayor, E. P. Hinchy, Y. Qiao, N. Murray, and D. Devine, "Digital twin: Origin to future," *Applied System Innovation*, vol. 4, no. 2, 2021. [Online]. Available: https://www.mdpi.com/2571-5577/4/2/36 22, 68

[15] F. Chinesta, E. Cueto, E. Abisset-Chavanne, J. L. Duval, and F. E. Khaldi, "Virtual, digital and hybrid twins: a new paradigm in data-based engineering and engineered data," *Archives of computational methods in engineering*, vol. 27, pp. 105–134, 2020. 22

[16] G. M. Amdahl, "Validity of the single processor approach to achieving large scale computing capabilities," in *Proceedings of the April 18-20, 1967, spring joint computer conference*, 1967, pp. 483–485. 22

[17] J. L. Gustafson, "Reevaluating amdahl's law," *Commun. ACM*, vol. 31, no. 5, p. 532–533, may 1988. [Online]. Available: https://doi.org/10.1145/42411.42415 22

[18] B. P. Zeigler, A. Muzy, and E. Kofman, *Theory of modeling and simulation: discrete event & iterative system computational foundations*. Academic press, 2018. 22, 42, 43

[19] S. Peitz and M. Dellnitz, "A survey of recent trends in multiobjective optimal control—surrogate models, feedback control and objective reduction," *Mathematical and Computational Applications*, vol. 23, no. 2, p. 30, 2018. 22

[20] N. Goldenson, L. R. Leung, L. O. Mearns, D. W. Pierce, K. A. Reed, I. R. Simpson, P. Ullrich, W. Krantz, A. Hall, A. Jones *et al.*, "Use-inspired, process-oriented gcm selection: Prioritizing models for regional dynamical downscaling," *Bulletin of the American Meteorological Society*, vol. 104, no. 9, pp. E1619–E1629, 2023. 22

[21] M. Schiller, M. Dupius, D. Krajzewicz, A. Kern, and A. Knoll, "Multi-resolution traffic simulation for large-scale high-fidelity evaluation of vanet applications," in *Simulating Urban Traffic Scenarios: 3rd SUMO Conference 2015 Berlin, Germany*. Springer, 2019, pp. 17–36. 22

[22] R. Biglari and J. Denil, "Self-adaptive abstraction and approximation (sacube) framework for digital twins with real-time requirements," *SIMULATION*, 2025, under peer review. 25, 26

[23] R. Biglari, C. Gomes, and J. Denil, "Towards a validity frame of multi-modal surrogate models for traffic simulation," in *2025 Annual Modeling and Simulation Conference (ANNSIM)*, 2025, in press. 25, 26

[24] R. Biglari, J. Mertens, and J. Denil, "Towards Real-time Adaptive Approximation," in *2022 11th European Congress Embedded Real Time Systems - ERTS 2022*, Toulouse, France, Jun. 2022. 26, 69, 73, 84, 100, 105

[25] R. Biglari and J. Denil, "Model validity and tolerance quantification for real-time adaptive approximation," in *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, ser. MODELS '22, 2022. 26, 69, 100, 106, 108, 121

[26] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. WieBner, "Microscopic Traffic Simulation using SUMO," *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, vol. 00, pp. 2575–2582, 2018. 27

[27] C. Greer, M. Burns, D. Wollman, and E. Griffor, "Cyber-physical systems and internet of things," 2019. 32

[28] W. Meng, Y. Yang, J. Zang, H. Li, and R. Lu, "Dtuav: a novel cloud–based digital twin system for unmanned aerial vehicles," *Simulation*, vol. 99, no. 1, pp. 69–87, 2023. 32, 60

[29] W. Kritzinger, M. Karner, G. Traar, J. Henjes, and W. Sihn, "Digital twin in manufacturing: A categorical literature review and classification," *Ifac-PapersOnline*, vol. 51, no. 11, pp. 1016–1022, 2018. 32, 36

[30] A. Vallée, "Digital twin for healthcare systems," *Frontiers in Digital Health*, vol. 5, p. 1253050, 2023. 32

[31] J. Bao, D. Guo, J. Li, and J. Zhang, "The modelling and operations for the digital twin in the context of manufacturing," *Enterprise Information Systems*, vol. 13, no. 4, pp. 534–556, 2019. [Online]. Available: https://doi.org/10.1080/17517575.2018.1526324 32, 36

[32] M. Farsi, A. Daneshkhah, A. Hosseinian-Far, H. Jahankhani *et al.*, *Digital twin technologies and smart cities*. Springer, 2020. 32

[33] M. J. Walker, "Hype cycle for emerging technologies, 2017," *Gartner*, pp. 1–68, 2017. 32

[34] M. J. Walker, B. Burton, and M. Cantara, "Hype cycle for emerging technologies," *Gartner, Stamford, USA*, pp. 1–73, 2018. 32

[35] B. D. Allen, "Digital twins and living models at NASA," in *2021 Digital Twin Summit*, 2021. [Online]. Available: https://ntrs.nasa.gov/citations/20210023699 33

[36] S. D. Roberts and D. Pegden, "The history of simulation modeling," in *2017 Winter Simulation Conference (WSC)*. IEEE, 2017, pp. 308–323. 33

[37] J. Tarnawski and T. Karla, "Real-time simulation in non real-time environment," in *2016 21st International Conference on Methods and Models in Automation and Robotics (MMAR)*. IEEE, 2016, pp. 577–582. 33, 64

[38] M. Rubinoff, "Digital computers for real-time simulation," *Journal of the ACM (JACM)*, vol. 2, no. 3, pp. 186–204, 1955. 33

[39] J. W. Forrester and R. R. Everett, "Project whirlwind collection." [Online]. Available: https://archivesspace.mit.edu/repositories/2/resources/1157 33

[40] R. R. Everett, "The whirlwind i computer," in *Papers and discussions presented at the Dec. 10-12, 1951, joint AIEE-IRE computer conference: Review of electronic digital computers*, 1951, pp. 70–74. 33

[41] K. Popovici and P. J. Mosterman, *Real-time simulation technologies: principles, methodologies, and applications*. CRC Press, 2017. 34, 64

[42] G. Lugaresi and A. Matta, "Real-time simulation in manufacturing systems: Challenges and research directions," in *2018 Winter Simulation Conference (WSC)*. IEEE, 2018, pp. 3319–3330. 34, 64

[43] P. Rogers and R. J. Gordon, "Simulation for real-time decision making in manufacturing systems," in *Proceedings of the 25th conference on winter simulation*, 1993, pp. 866–874. 34, 64

[44] N. M. Alison Harper and M. Pitt, "Increasing situation awareness in healthcare through real-time simulation," *Journal of the Operational Research Society*, vol. 74, no. 11, pp. 2339–2349, 2023. [Online]. Available: https://doi.org/10.1080/01605682.2022.2147030 34, 64

[45] D. Gelernter, *Mirror worlds: Or the day software puts the universe in a shoebox... How it will happen and what it will mean*. Oxford University Press, 1993. [Online]. Available: https://www.google.fr/books/edition/Mirror_Worlds_Or_The_Day_Software_Puts_t/PmCoPwAACAAJ?hl=en 34

[46] W. J. Davis, "On-line simulation: Need and evolving research requirements," *Handbook of simulation*, vol. 465, p. 516, 1998. 34

[47] R. Fujimoto, W. H. Lunceford Jr., E. H. Page, and A. Uhrmacher, "Grand Challenges for Modelling and Simulation (Dagstuhl Seminar 02351)," Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, Dagstuhl Seminar Report 350, 2002. [Online]. Available: https://drops.dagstuhl.de/entities/document/10.4230/DagSemRep.350 34

[48] H. Aydt, S. J. Turner, W. Cai, and M. Y. H. Low, "Symbiotic simulation systems: An extended definition motivated by symbiosis in biology," in *2008 22nd Workshop on Principles of Advanced and Distributed Simulation*, 2008, pp. 109–116. 34

[49] B. S. Onggo, *Symbiotic Simulation System (S3) for Industry 4.0*. Cham: Springer International Publishing, 2019, pp. 153–165. [Online]. Available: https://doi.org/10.1007/978-3-030-04137-3_10 34

[50] Y. Cao, C. Currie, B. S. Onggo, and M. Higgins, "Simulation optimization for a digital twin using a multi-fidelity framework," in *2021 Winter Simulation Conference (WSC)*, 2021, pp. 1–12. 34

[51] M. Shafto, M. Conroy, R. Doyle, E. Glaessgen, C. Kemp, J. LeMoigne, and L. Wang, "Modeling, simulation, information technology & processing roadmap," *National Aeronautics and Space Administration*, vol. 32, no. 2012, pp. 1–38, 2012. [Online]. Available: https://www.nasa.gov/sites/default/files/501321main_TA11-ID_rev4_NRC-wTASR.pdf 35

[52] E. Glaessgen and D. Stargel, *The Digital Twin Paradigm for Future NASA and U.S. Air Force Vehicles*, 2012. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/6.2012-1818 35, 36

[53] Alliance Industrie du futur. (2023) Digital twin: Leveraging the digital transformation of the industry. [Online]. Available: http://www.industrie-dufutur.org/content/uploads/2023/05/AIF_JumeauNumeriqueEN_US.pdf 35

[54] "Digital twin definition," *Digital Twin Consortium*. [Online]. Available: https://www.digitaltwinconsortium.org/initiatives/the-definition-of-a-digital-twin/ 35, 59

[55] C. Lo, C. Chen, and R. Y. Zhong, "A review of digital twin in product design and development," *Advanced Engineering Informatics*, vol. 48, p. 101297, 2021. [Online]. Available: https://doi.org/10.1016/j.aei.2021.101297 36

[56] General Electric. (2023) What is a digital twin? [Online]. Available: https://www.ge.com/digital/blog/what-digital-twin 36

[57] "ISO 23247-1:2021 Automation systems and integration- Digital twin framework for manufacturing - Part 1: Overview and general principles," International Organization for Standardization, Geneva, CH, Standard, Mar. 2021. 36, 59, 61

[58] B. R. Babic, "Digital twins in smart manufacturing," in *Soft Computing in Smart Manufacturing*. Walter de Gruyter GmbH, 2021. 36

[59] G. Lugaresi, S. Gangemi, G. Gazzoni, and A. Matta, "Online validation of digital twins for manufacturing systems," *Computers in Industry*, vol. 150, p. 103942, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0166361523000921 36, 58, 61, 62, 65

[60] G. Shao, S. Jain, C. Laroque, L. H. Lee, P. Lendermann, and O. Rose, "Digital twin for smart manufacturing: the simulation aspect," in *2019 Winter Simulation Conference (WSC)*. IEEE, 2019, pp. 2085–2098. 36, 58, 61, 62, 63

[61] E. VanDerHorn and S. Mahadevan, "Digital twin: Generalization, characterization and implementation," *Decision support systems*, vol. 145, p. 113524, 2021. 37, 61

[62] O. Balci, "A life cycle for modeling and simulation," *Simulation*, vol. 88, no. 7, pp. 870–883, 2012. 37, 38

[63] A. M. Law and D. Kelton, *Simulation modeling and analysis*. Mcgraw-hill New York, 2007, vol. 3. 37, 65

[64] B. P. Zeigler, H. Praehofer, and T. G. Kim, *Theory of modeling and simulation*. Academic press, 2000. 37, 67

[65] S. Schlesinger, R. E. Crosbie, R. E. Gagné, G. S. Innis, C. Lalwani, J. Loch, R. J. Sylvester, R. D. Wright, N. Kheir, and D. Bartos, "Terminology for model credibility," *SIMULATION*, vol. 32, no. 3, pp. 103–104, 1979. [Online]. Available: https://doi.org/10.1177/003754977903200304 37

[66] G. Kim, J. Humble, P. Debois, J. Willis, and N. Forsgren, *The DevOps handbook: How to create world-class agility, reliability, & security in technology organizations*. IT Revolution, 2021. 38

[67] J. Pushpa and S. Kalyani, "Using fog computing/edge computing to leverage digital twin," in *Advances in Computers*. Elsevier, 2020, vol. 117, no. 1, pp. 51–77. 39

[68] A. Matta and G. Lugaresi, "Digital twins: Features, models, and services," in *2023 Winter Simulation Conference*, 2023. 39

[69] A. Wortmann. (2023) Digital twin definitions. [Online]. Available: https://awortmann.github.io/research/digital-twin-definitions/ 40

[70] M. Dalibor, N. Jansen, B. Rumpe, D. Schmalzing, L. Wachtmeister, M. Wimmer, and A. Wortmann, "A cross-domain systematic mapping study on software engineering for digital twins," *Journal of Systems and Software*, vol. 193, p. 111361, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121222000917 40

[71] C. Steinmetz, A. Rettberg, F. G. C. Ribeiro, G. Schroeder, and C. E. Pereira, "Internet of things ontology for digital twin in cyber physical systems," in *2018 VIII Brazilian Symposium on Computing Systems Engineering (SBESC)*, 2018, pp. 154–159. 40

[72] F. Yacob, D. Semere, and E. Nordgren, "Anomaly detection in skin model shapes using machine learning classifiers," *The International Journal of Advanced Manufacturing Technology*, vol. 105, no. 9, pp. 3677–3689, 2019. 40

[73] J. Autiosalo, J. Vepsäläinen, R. Viitala, and K. Tammi, "A feature-based framework for structuring industrial digital twins," *IEEE Access*, vol. 8, pp. 1193–1208, 2020. 40

[74] R. Paredis and H. Vangheluwe, "Modelling and simulation-based evaluation of twinning architectures and their deployment," in *Proceedings of the 14th International Conference on Simulation and Modeling Methodologies, Technologies and Applications - SIMULTECH*, INSTICC. SciTePress, 2024, pp. 170–182. 40

[75] M. Grieves, "Digital twin: manufacturing excellence through virtual factory replication," *White paper*, vol. 1, no. 2014, pp. 1–7, 2014. 40

[76] H. Richard, "The gemini principles. centre for digital built britain." 2023, accessed: 2023-11-12. [Online]. Available: https://www.cdbb.cam.ac.uk/Resources/ResoucePublications/TheGeminiPrinciples.pdf 41

[77] S. Rizzi, "What-if analysis. encyclopedia of database systems," 2009. 41

[78] K. Dehghanpour, Z. Wang, J. Wang, Y. Yuan, and F. Bu, "A survey on state estimation techniques and challenges in smart distribution systems," *IEEE Transactions on Smart Grid*, vol. 10, no. 2, pp. 2312–2322, 2018. 41

[79] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, pp. 1–58, 2009. 41

[80] P. Zhou, D. Zuo, K. M. Hou, Z. Zhang, J. Dong, J. Li, and H. Zhou, "A comprehensive technological survey on the dependable self-management cps: From self-adaptive architecture to self-management strategies," *Sensors*, vol. 19, no. 5, p. 1033, 2019. 41

[81] H. Feng, C. Gomes, C. Thule, K. Lausdahl, A. Iosifidis, and P. G. Larsen, "Introduction to digital twin engineering," in *2021 Annual Modeling and Simulation Conference (ANNSIM)*. IEEE, 2021, pp. 1–12. 41, 48

[82] M. O. Ward, G. Grinstein, and D. Keim, *Interactive data visualization: foundations, techniques, and applications*. AK Peters/CRC Press, 2010. 41

[83] R. Agrawal, A. Kadadi, X. Dai, and F. Andres, "Challenges and opportunities with big data visualization," in *Proceedings of the 7th International Conference on Management of computational and collective intElligence in Digital EcoSystems*, 2015, pp. 169–173. 41

[84] M. Amrani, D. Blouin, R. Heinrich, A. Rensink, H. Vangheluwe, and A. Wortmann, "Multi-paradigm modelling for cyber–physical systems: a descriptive framework," *Software and Systems Modeling*, vol. 20, no. 3, pp. 611–639, 2021. 41

[85] P. Carreira, V. Amaral, and H. Vangheluwe, "Multi-paradigm modelling for cyber-physical systems: Foundations," *Foundations of Multi-Paradigm Modelling for Cyber-Physical Systems*, pp. 1–14, 2020. 41

[86] F. P. Brooks and N. S. Bullet, "Essence and accidents of software engineering," *IEEE computer*, vol. 20, no. 4, pp. 10–19, 1987. 41

[87] Y. Van Tendeloo, "A foundation for multi-paradigm modelling," Ph.D. dissertation, University of Antwerp, 2018. 41

[88] H. Stachowiak, *Allgemeine Modelltheorie*. Wien, New York: Springer, 1973. 42

[89] M. Minsky, "Matter, mind and models," 1965. 42

[90] F. K. Frantz and A. J. Ellor, *Model abstraction techniques*. Rome Laboratory, Air Force Materiel Command, 1996. 44, 45

[91] P. P. Nayak, "Causal approximations," *Artificial Intelligence*, vol. 70, no. 1-2, pp. 277–334, 1994. 44

[92] D. S. Weld, "Reasoning about model accuracy," *Artificial Intelligence*, vol. 56, no. 2-3, pp. 255–300, 1992. 44

[93] D. Caughlin, "Model abstraction via solution of a general inverse problem to define a metamodel," *METAMODELING TECHNIQUES AND APPLICATIONS*, p. 0, 2000. 44

[94] L. Jia, R. Alizadeh, J. Hao, G. Wang, J. K. Allen, and F. Mistree, "A rule-based method for automated surrogate model selection," *Advanced Engineering Informatics*, vol. 45, p. 101123, 2020. 44

[95] P. J. Mosterman and H. Vangheluwe, "Computer automated multi-paradigm modeling: An introduction," *Simulation*, vol. 80, no. 9, pp. 433–450, 2004. 45

[96] B. P. Zeigler, *Theory of Modeling and Simulation*. John Wiley, 1976. 45

[97] B. Zeigler, *Multifacetted modelling and discrete event simulation*. Academic Press Professional, Inc., 1984. 46

[98] B. Van Acker, P. De Meulenaere, H. Vangheluwe, and J. Denil, "Validity frame–enabled model-based engineering processes," *Simulation*, vol. 100, no. 2, pp. 185–226, 2024. 47, 71

[99] S. Van Mierlo, B. J. Oakes, B. Van Acker, R. Eslampanah, J. Denil, and H. Vangheluwe, "Exploring validity frames in practice," in *International Conference on Systems Modelling and Management*. Springer, 2020, pp. 131–148. 47, 67, 71

[100] R. G. Sargent, "An introductory tutorial on verification and validation of simulation models," in *2015 winter simulation conference (WSC)*. IEEE, 2015, pp. 1729–1740. 47

[101] S. Klikovits, J. Denil, A. Muzy, and R. Salay, "Modeling frames," in *CEUR workshop proceedings*, 2017, pp. 315–320. 47

[102] R. Mittal, R. Eslampanah, L. Lima, H. Vangheluwe, and D. Blouin, "Towards an ontological framework for validity frames," in *2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, 2023, pp. 801–805. 47

[103] A. Sari and J. C. Sopuru, "Bayesian model for evaluating real-world adaptation progress of a cyber-physical system," in *Artificial Intelligence Paradigms for Smart Cyber-Physical Systems*. IGI Global, 2021, pp. 324–343. 48

[104] J. A. Stankovic, "Misconceptions about real-time computing: A serious problem for next-generation systems," *Computer*, vol. 21, no. 10, pp. 10–19, 1988. 49

[105] G. C. Buttazzo, *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications*, 3rd ed. Springer Publishing Company, Incorporated, 2011. 49, 51, 52

[106] J. A. Stankovic, "Real-time and embedded systems," *ACM Comput. Surv.*, vol. 28, no. 1, p. 205–208, Mar. 1996. [Online]. Available: https://doi.org/10.1145/234313.234400 49

[107] L. Cucu-Grosjean, "Invited Paper: Statistical, Stochastic or Probabilistic (Worst-Case Execution) Execution Time? - What Impact on the Multicore Composability," in *22nd International Workshop on Worst-Case Execution Time Analysis (WCET 2024)*, ser. Open Access Series in Informatics (OASIcs), T. Carle, Ed., vol. 121. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024, pp. 6:1–6:10. [Online]. Available: https://drops.dagstuhl.de/entities/document/10.4230/OASIcs.WCET.2024.6 51

[108] F. Reghenzani, *Beyond the Traditional Analyses and Resource Management in Real-Time Systems*. Cham: Springer International Publishing, 2022, pp. 67–77. [Online]. Available: https://doi.org/10.1007/978-3-030-85918-3_6 51

[109] P. Gliwa, *Embedded Software Timing: Methodology, Analysis and Practical Tips with a Focus on Automotive*. Springer Nature, 2021. 51

[110] C. André, F. Boulanger, M.-A. Péraldi, J.-P. Rigault, and G. Vidal-Naquet, "Objects and synchronous programming," *RAIRO-APII-JESA-Journal Europeen des Systemes Automatises*, vol. 31, no. 3, pp. 417–432, 1997. 54

[111] E. Lee and D. Messerschmitt, "Synchronous data flow," *Proceedings of the IEEE*, vol. 75, no. 9, pp. 1235–1245, 1987. 54

[112] R. Rosen, G. Von Wichert, G. Lo, and K. D. Bettenhausen, "About the importance of autonomy and digital twins for the future of manufacturing," *Ifac-papersonline*, vol. 48, no. 3, pp. 567–572, 2015. 58

[113] S. Wagner, M. Milde, F. Barhebwa-Mushamuka, and G. Reinhart, "Digital twin design in production," in *2022 Towards Sustainable Customization: Bridging Smart Products and Manufacturing Systems: Proceedings of the 8th Changeable, Agile, Reconfigurable and Virtual Production Conference (CARV2021) and the 10th World Mass Customization & Personalization Conference (MCPC2021), Aalborg, Denmark, October/November 2021 8*. Springer, 2022, pp. 339–346. 58, 59, 60, 61, 62, 63

[114] A. Budiardjo and D. Migliori, "Digital twin system interoperability framework," *Digital Twin Consortium*, 2021. 58, 59, 60

[115] "ITU-TY.3090 - Digital twin network – Requirements and architecture - Y series: Global information infrastructure, Internet protocol aspects, next-generation networks, Internet of Things and smart cities, Digital twin network – Requirements and architecture," International Telecommunication Union (ITU), Standard, Feb. 2022. 58, 60, 62, 63

[116] B. Nelson *et al.*, *Foundations and methods of stochastic simulation*. Springer, 2021. 58

[117] "ISO/IEC 22123-1:2023: Information technology—Cloud computing—Part 1: Vocabulary," International Organization for Standardization, Geneva, CH, Standard, Mar. 2023. 59

[118] B. Schleich, N. Anwer, L. Mathieu, and S. Wartzack, "Shaping the digital twin for design and production engineering," *CIRP annals*, vol. 66, no. 1, pp. 141–144, 2017. 59, 60, 61, 62

[119] L. F. C. Durão, S. Haag, R. Anderl, K. Schützer, and E. Zancul, "Digital twin requirements in the context of industry 4.0," in *2018 Product Lifecycle Management to Support Industry 4.0: 15th IFIP WG 5.1 International Conference, PLM 2018, Turin, Italy, July 2-4, 2018, Proceedings 15*. Springer, 2018, pp. 204–214. 59, 60, 62

[120] D. Lehner, J. Pfeiffer, E.-F. Tinsel, M. M. Strljic, S. Sint, M. Vierhauser, A. Wortmann, and M. Wimmer, "Digital twin platforms: requirements, capabilities, and future prospects," *IEEE Software*, vol. 39, no. 2, pp. 53–61, 2021. 59, 60, 62

[121] G. Shao and D. Kibira, "Digital manufacturing: Requirements and challenges for implementing digital surrogates," in *2018 Winter Simulation Conference (WSC)*. IEEE, 2018, pp. 1226–1237. 59, 60, 61, 62

[122] H. W. Group *et al.*, "IEEE 1516-2010—IEEE standard for modeling and simulation (M&S) high level architecture (hla)—framework and rules," *IEEE Computer Society: Washington, DC, USA*, 2010. 59

[123] A. Rasheed, O. San, and T. Kvamsdal, "Digital twin: Values, challenges and enablers from a modeling perspective," *IEEE Access*, vol. 8, pp. 21 980–22 012, 2020. 60, 61, 62, 63

[124] S.-H. Attarzadeh-Niaki and I. Sander, "An extensible modeling methodology for embedded and cyber-physical system design," *Simulation*, vol. 92, no. 8, pp. 771–794, 2016. 60

[125] D. Jones, C. Snider, A. Nassehi, J. Yon, and B. Hicks, "Characterising the digital twin: A systematic literature review," *CIRP journal of manufacturing science and technology*, vol. 29, pp. 36–52, 2020. 59, 60, 61

[126] J. Moyne, Y. Qamsane, E. C. Balta, I. Kovalenko, J. Faris, K. Barton, and D. M. Tilbury, "A requirements driven digital twin framework: Specification and opportunities," *IEEE Access*, vol. 8, pp. 107 781–107 801, 2020. 59, 60, 62

[127] M. Liu, S. Fang, H. Dong, and C. Xu, "Review of digital twin about concepts, technologies, and industrial applications," *Journal of Manufacturing Systems*, vol. 58, pp. 346–361, 2021. 60, 62

[128] M. Sjarov, T. Lechler, J. Fuchs, M. Brossog, A. Selmaier, F. Faltus, T. Donhauser, and J. Franke, "The digital twin concept in industry–a review and systematization," in *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1. IEEE, 2020, pp. 1789–1796. 60, 61, 63

[129] B. A. Talkhestani, N. Jazdi, W. Schloegl, and M. Weyrich, "Consistency check to synchronize the digital twin of manufacturing automation based on anchor points," *Procedia Cirp*, vol. 72, pp. 159–164, 2018. 60, 61

[130] L. Wang, T. Deng, Z. Zheng, and Z.-J. M. Shen, "Explainable modeling in digital twin," in *2021 Winter Simulation Conference (WSC)*. IEEE, 2021, pp. 1–12. 60, 63

[131] R. Zhang, F. Wang, J. Cai, Y. Wang, H. Guo, and J. Zheng, "Digital twin and its applications: A survey," *The International Journal of Advanced Manufacturing Technology*, vol. 123, no. 11-12, pp. 4123–4136, 2022. 60, 62, 63

[132] M. S. Sapkota, "Adaptive simulation modelling using the digital twin paradigm," Ph.D. dissertation, Bournemouth University, 2023. 61

[133] K. Reifsnider and P. Majumdar, "Multiphysics stimulated simulation digital twin methods for fleet management," in *2013 54th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2013, p. 1578. 61

[134] C. Kober, V. Adomat, M. Ahanpanjeh, M. Fette, and J. P. Wulfsberg, "Digital twin fidelity requirements model for manufacturing," in *2022 Proceedings of the Conference on Production Systems and Logistics: CPSL 2022*. Hannover: publish-Ing., 2022, pp. 595–611. 61

[135] C. Kober, B. N. Algan, M. Fette, and J. P. Wulfsberg, "Relations of digital twin fidelity and benefits: A design-to-value approach," *Procedia CIRP*, vol. 119, pp. 809–815, 2023. 61

[136] T. Yue, S. Ali, P. Arcaini, and F. Ishikawa, "Towards requirements engineering for digital twins of cyber-physical systems," in *2022 International Symposium on Leveraging Applications of Formal Methods*. Springer, 2022, pp. 9–21. 61

[137] E. Y. Hua, S. Lazarova-Molnar, and D. P. Francis, "Validation of digital twins: Challenges and opportunities," in *2022 Winter Simulation Conference (WSC)*. IEEE, 2022, pp. 2900–2911. 61, 62

[138] R. G. Sargent, "Verification and validation of simulation models," in *2010 Proceedings of the 2010 Winter Simulation Conference*. IEEE, 2010, pp. 166–183. 61, 62

[139] A. Khan, M. Dahl, P. Falkman, and M. Fabian, "Digital twin for legacy systems: Simulation model testing and validation," in *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2018, pp. 421–426. 62

[140] U. Dahmen, T. Osterloh, and J. Roßmann, "Verification and validation of digital twins and virtual testbeds," *Int J Adv Appl Sci*, vol. 11, no. 1, pp. 47–64, 2022. 62

[141] A. Löcklin, M. Müller, T. Jung, N. Jazdi, D. White, and M. Weyrich, "Digital twin for verification and validation of industrial automation systems–a survey," in *2020 25th IEEE international conference on emerging technologies and factory automation (ETFA)*, vol. 1. IEEE, 2020, pp. 851–858. 62

[142] A. Cortés *et al.*, "Deep air–a smart city AI synthetic data digital twin solving the scalability data problems," in *2022 Artificial Intelligence Research and Development: Proceedings of the 24th International Conference of the Catalan Association for Artificial Intelligence*, vol. 356. IOS Press, 2022, p. 83. 63

[143] "ISO 23247-2:2021 Automation systems and integration — Digital twin framework for manufacturing — Part 2: Reference architecture," International Organization for Standardization, Geneva, CH, Standard, Mar. 2021. 63

[144] Z. Zhu, C. Liu, and X. Xu, "Visualisation of the digital twin data in manufacturing by using augmented reality," *Procedia Cirp*, vol. 81, pp. 898–903, 2019. 63

[145] J. Bélanger, P. Venne, J.-N. Paquin *et al.*, "The what, where and why of real-time simulation," *Planet Rt*, vol. 1, no. 1, pp. 25–29, 2010. 64

[146] X. Hu, "Data assimilation for simulation-based real-time prediction/analysis," in *2022 Annual Modeling and Simulation Conference (ANNSIM)*, 2022, pp. 404–415. 64

[147] Y. Huang, X. Xie, Y. Cho, and A. Verbraeck, "Particle filter–based data assimilation in dynamic data-driven simulation: sensitivity analysis of three critical experimental conditions," *Simulation*, vol. 99, no. 4, pp. 403–415, 2023. 64

[148] M. K. Traore, S. Gorecki, and Y. Ducq, "A simulation based approach to digital twin's interoperability verification & validation," in *2022 Workshop" Interoperability challenges and solutions within industrial networks" co-located with 11th International Conference on Interoperability for Enterprise Systems and Applications (I-ESA 2022)*, vol. 3214, 2022. 64

[149] J. Mertens and J. Denil, "Digital-twin co-evolution using continuous validation," in *2023 Proceedings of the ACM/IEEE 14th International Conference on Cyber-Physical Systems (with CPS-IoT Week 2023)*, ser. ICCPS '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 266–267. [Online]. Available: https://doi.org/10.1145/3576841.3589628 65

[150] C. Beisbart and N. J. Saam, *Computer simulation validation*, C. Beisbart and N. J. Saam, Eds. Switzerland: Springer Nature Switzerland AG, 2019. 65

[151] G. Lugaresi, S. Gangemi, G. Gazzoni, and A. Matta, "Online validation of simulation-based digital twins exploiting time series analysis," in *2022 Winter Simulation Conference (WSC)*, 2022, pp. 2912–2923. 65

[152] L. Overbeck, A. Le Louarn, O. Brützel, N. Stricker, and G. Lanza, "Continuous validation and updating for high accuracy of digital twins of production systems," in *2021 Simulation in Produktion und Logistik 2021, Erlangen, 15.-17.September 2021. Hrsg.: J. Franke*, ser. Simulation in Produktion und Logistik 2021, vol. 0. Cuvillier Verlag, 2021, p. 609–617. 65

[153] J. Gertler, *Fault detection and diagnosis in engineering systems*. Boca Raton: CRC press, 1998. 65, 66

[154] Y. Harada, Y. Yamagata, O. Mizuno, and E.-H. Choi, "Log-based anomaly detection of CPS using a statistical method," in *2017 8th International Workshop on Empirical Software Engineering in Practice (IWESEP)*, 2017, pp. 1–6. 65

[155] S. Narasimhan and G. Biswas, "Model-based diagnosis of hybrid systems," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 37, no. 3, pp. 348–361, 2007. 66

[156] S. Hashtrudi Zad, R. Kwong, and W. Wonham, "Fault diagnosis in discrete-event systems: framework and model reduction," *IEEE Transactions on Automatic Control*, vol. 48, no. 7, pp. 1199–1212, 2003. 66

[157] A. Ramirez-Trevino, E. Ruiz-Beltran, I. Rivera-Rangel, and E. Lopez-Mellado, "Online fault diagnosis of discrete event systems. a petri net-based approach," *IEEE Transactions on Automation Science and Engineering*, vol. 4, no. 1, pp. 31–39, 2007. 66

[158] D. Li, Y. Wang, J. Wang, C. Wang, and Y. Duan, "Recent advances in sensor fault diagnosis: A review," *Sensors and Actuators A: Physical*, vol. 309, p. 111990, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0924424719308635 66

[159] A. Abid, M. T. Khan, and J. Iqbal, "A review on fault detection and diagnosis techniques: basics and beyond," *Artificial Intelligence Review*, vol. 54, no. 5, pp. 3639–3664, Jun 2021. [Online]. Available: https://doi.org/10.1007/s10462-020-09934-2 66

[160] H. Darvishi, D. Ciuonzo, and P. S. Rossi, "A machine-learning architecture for sensor fault detection, isolation, and accommodation in digital twins," *IEEE Sensors Journal*, vol. 23, no. 3, pp. 2522–2538, 2023. 66

[161] N. Polyzotis, M. Zinkevich, S. Roy, E. Breck, and S. Whang, "Data validation for machine learning," in *2019 Proceedings of Machine Learning and Systems*, A. Talwalkar, V. Smith, and M. Zaharia, Eds., vol. 1, 2019, pp. 334–347. [Online]. Available: https://proceedings.mlsys.org/paper_files/paper/2019/file/928f1160e52192e3e0017fb63ab65391-Paper.pdf 66

[162] J. Mertens and J. Denil, "The digital twin as a common knowledge base in devops to support continuous system evolution," in *2021 Computer Safety, Reliability, and Security. SAFECOMP 2021 Workshops*, I. Habli, M. Sujan, S. Gerasimou, E. Schoitsch, and F. Bitsch, Eds. Cham: Springer International Publishing, 2021, pp. 158–170. 67

[163] M. K. Traoré and A. Muzy, "Capturing the dual relationship between simulation models and their context," *Simulation Modelling Practice and Theory*, vol. 14, no. 2, pp. 126–142, 2006. 67

[164] J. Denil, S. Klikovits, P. J. Mosterman, A. Vallecillo, and H. Vangheluwe, "The experiment model and validity frame in m&s," in *2017 Proceedings of the Symposium on Theory of Modeling & Simulation*, 2017, pp. 1–12. 67

[165] B. Van Acker, P. De Meulenaere, J. Denil, Y. Durodie, A. Van Bellinghen, and K. Vanstechelman, "Valid (re-) use of models-of-the-physics in cyber-physical systems using validity frames," in *2019 Spring Simulation Conference (SpringSim)*. IEEE, 2019, pp. 1–12. 67

[166] M. Otter, M. Reiner, J. Tobolář, L. Gall, and M. Schäfer, "Towards modelica models with credibility information," *Electronics*, vol. 11, no. 17, p. 2728, 2022. 68

[167] J. Köhler, H.-M. Heinkel, P. Mai, J. Krasser, M. Deppe, and M. Nagasawa, "Modelica-association-project "system structure and parameterization"–early insights," in *2016 The First Japanese Modelica Conferences, May 23-24, Tokyo, Japan*, no. 124. Linköping University Electronic Press, 2016, pp. 35–42. 68

[168] H. Feng, C. Gomes, S. Gil, P. H. Mikkelsen, D. Tola, P. G. Larsen, and M. Sandberg, "Integration of the mape-k loop in digital twins," in *2022 Annual Modeling and Simulation Conference (ANNSIM)*. IEEE, 2022, pp. 102–113. 68

[169] I. David and E. Syriani, "DEVS model construction as a reinforcement learning problem," in *2022 Annual Modeling and Simulation Conference (ANNSIM)*. IEEE, 2022, pp. 30–41. 68

[170] F. Lauer, G. Bloch, F. Lauer, and G. Bloch, *Hybrid system identification*. Springer, 2019. 68, 120, 128

[171] P. K. Davis, "Exploratory analysis enabled by multiresolution, multiperspective modeling," in *2000 Winter Simulation Conference Proceedings (Cat. No. 00CH37165)*, vol. 1. IEEE, 2000, pp. 293–302. 69, 99

[172] P. K. Davis and J. H. Bigelow, *Experiments in multiresolution modeling (MRM)*. Rand Santa Monica, CA, USA, 1998. 69, 99

[173] R. Franceschini, S. Van Mierlo, and H. Vangheluwe, "Towards adaptive abstraction in agent based simulation," in *2019 Winter Simulation Conference (WSC)*. IEEE, 2019, pp. 2725–2736. 69, 99

[174] S. Bosmans, T. Bogaerts, W. Casteels, S. Mercelis, J. Denil, and P. Hellinckx, "Adaptivity in multi-level traffic simulation using experimental frames," *Simulation Modelling Practice and Theory*, vol. 114, p. 102395, Jan. 2022. 69, 99, 121, 122

[175] P. Muñoz, M. Wimmer, J. Troya, and A. Vallecillo, "Using trace alignments for measuring the similarity between a physical and its digital twin," in *2022 Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, ser. MODELS '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 503–510. [Online]. Available: https://doi.org/10.1145/3550356.3563135 69, 100

[176] J. Mertens and J. Denil, "ESS: EMF-based simulation specification, a domain-specific language for model validation experiments," in *2022 Annual Modeling and Simulation Conference (ANNSIM)*. IEEE, 2022, pp. 416–427. 69, 100

[177] J. Malmodin and D. Lundén, "The energy and carbon footprint of the global ICT and E&M sectors 2010–2015," *Sustainability*, vol. 10, no. 9, 2018. [Online]. Available: https://www.mdpi.com/2071-1050/10/9/3027 69

[178] S. G. Anders and T. Edler, "On global electricity usage of communication technology: Trends to 2030," *Challenges*, vol. 6, no. 1, pp. 117–157, 2015. [Online]. Available: https://www.mdpi.com/2078-1547/6/1/117 69

[179] L. Belkhir and A. Elmeligi, "Assessing ICT global emissions footprint: Trends to 2040 recommendations," *Journal of Cleaner Production*, vol. 177, pp. 448–463, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S095965261733233X 69

[180] F. Berkhout and J. Hertin, "Impacts of information and communication technologies on environmental sustainability: speculations and evidence. report to the oecd," *Science and Technology Policy Research Unit, University of Sussex, Brighton*, 2001. 69

[181] S. Bellis and J. Denil, "Challenges and possible approaches for sustainable digital twinning," in *2022 Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, ser. MODELS '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 643–648. [Online]. Available: https://doi.org/10.1145/3550356.3561551 70

[182] B. Barroca, T. Kühne, and H. Vangheluwe, "Integrating language and ontology engineering." in *MPM@ MoDELS*, 2014, pp. 77–86. 72

[183] J. Cederbladh, L. Cleophas, E. Kamburjan, L. Lima, and H. Vangheluwe, "Symbolic reasoning for early decision-making in model-based systems engineering," in *2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, 2023, pp. 721–725. 74

[184] R. Biglari, S. Bellis, and J. Denil, "Decision-centric: A method for digital twins demonstrated through an autoclave use case," 2025, manuscript in preparation. 79, 127

[185] I. Gilboa, "Information and meta information," in *Proceedings of the 2nd Conference on Theoretical Aspects of Reasoning about Knowledge*, 1988, pp. 227–243. 85

[186] Copradar. (2024) Acceleration reference. Accessed: 2024-12-11. [Online]. Available: https://copradar.com/chapts/references/acceleration.html 94

[187] P. Bokare and A. Maurya, "Acceleration-deceleration behaviour of various vehicle types," *Transportation Research Procedia*, vol. 25, pp. 4733–4749, 2017, world Conference on Transport Research - WCTR 2016 Shanghai. 10-15 July 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352146517307937 94

[188] L. Yilmaz and T. I. Ören, "Discrete-event multimodels and their agent-supported activation and update," in *Proceedings of The Agent-Directed Simulation Symposium of the Spring Simulation Multiconference (SMC'05)*, 2005, pp. 63–72. 99

[189] M. L. Loper and C. Turnitsa, *History of Combat Modeling and Distributed Simulation*. John Wiley and Sons, Ltd, 2012, ch. 16, pp. 329–355. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118180310.ch16 99

[190] P. K. Davis, "An introduction to variable-resolution modeling," *Naval Research Logistics (NRL)*, vol. 42, no. 2, pp. 151–181, 1995. 100

[191] F. Lauer, G. Bloch, and R. Vidal, "Nonlinear hybrid system identification with kernel models," in *49th IEEE Conference on Decision and Control (CDC)*, 2010, pp. 696–701. 100

[192] F. Lauer and G. Bloch, "Hybrid system identification," *Lecture Notes in Control and Information Sciences*, 2018. 100

[193] C. Feng, C. M. Lagoa, and M. Sznaier, "Hybrid system identification via sparse polynomial optimization," in *Proceedings of the 2010 American Control Conference*, 2010, pp. 160–165. 100

[194] L. Serena, M. Marzolla, G. D'Angelo, and S. Ferretti, "Design patterns for multi-level modeling and simulation," in *2023 IEEE/ACM 27th International Symposium on Distributed Simulation and Real Time Applications (DS-RT)*, 2023, pp. 48–55. 100

[195] J. A. Erkoyuncu, I. F. del Amo, D. Ariansyah, D. Bulka, R. Roy *et al.*, "A design framework for adaptive digital twins," *CIRP annals*, vol. 69, no. 1, pp. 145–148, 2020. 101

[196] E. W. Weisel, M. D. Petty, and R. R. Mielke, "Validity of models and classes of models in semantic composability," in *Proceedings of the Fall 2003 Simulation Interoperability Workshop*, vol. 9, no. 68. Citeseer, 2003. 101

[197] A. Junghanns, C. Gomes, C. Schulze, K. Schuch, R. Pierre, M. Blaesken, I. Zacharias, A. Pillekeit, K. Wernersson, T. Sommer *et al.*, "The functional mock-up interface 3.0-new features enabling new applications," in *Modelica conferences*, 2021, pp. 17–26. 102

[198] H. S. Sarjoughian, "Model composability," in *Proceedings of the 2006 Winter Simulation Conference*, 2006, pp. 149–158. 102

[199] M. D. Petty and E. W. Weisel, "Chapter 4 - model composition and reuse," in *Model Engineering for Simulation*, L. Zhang, B. P. Zeigler, and Y. laili, Eds. Academic Press, 2019, pp. 57–85. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128135433000044 102

[200] C. A. Shaffer, *Data structures & algorithm analysis in Java*. Courier Corporation, 2011. 104

[201] S. Bosmans, S. Mercelis, P. Hellinckx, and J. Denil, "Reducing computational cost of large-scale simulations using opportunistic model approximation," in *2019 Spring Simulation Conference (SpringSim)*. IEEE, Apr. 2019, pp. 1–12. 105, 121

[202] S. Bosmans, S. Mercelis, J. Denil, and P. Hellinckx, "Challenges of modeling and simulating internet of things systems," in *Advances on P2P, Parallel, Grid, Cloud and Internet Computing*, F. Xhafa, F.-Y. Leu, M. Ficco, and C.-T. Yang, Eds. Cham: Springer International Publishing, 2019, pp. 457–466. 105

[203] J. A. Erkoyuncu, I. F. del Amo, D. Ariansyah, D. Bulka, R. Vrabič, and R. Roy, "A design framework for adaptive digital twins," *CIRP Annals*, vol. 69, no. 1, pp. 145–148, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0007850620301086 106

[204] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018. [Online]. Available: https://elib.dlr.de/124092/ 106

[205] A. Wegener, H. Hellbruck, C. Wewetzer, and A. Lubke, "Vanet simulation environment with feedback loop and its application to traffic light assistance," in *2008 IEEE Globecom Workshops*, 2008, pp. 1–7. 106

[206] D. Weyns, "Engineering self-adaptive software systems–an organized tour," in *2018 ieee 3rd international workshops on foundations and applications of self* systems (fas* w)*. IEEE, 2018, pp. 1–2. 108

[207] A. Petrovska, T. Hutzelmann, and S. Kugele, "A theoretical framework for self-adaptive systems: Specifications, formalisation, and architectural implications," in *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, ser. SAC '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 1440–1449. [Online]. Available: https://doi.org/10.1145/3555776.3577665 108

[208] C. Angione, E. Silverman, and E. Yaneske, "Using machine learning as a surrogate model for agent-based simulations," *PLOS ONE*, vol. 17, no. 2, p. e0263150, Feb. 2022. 109, 110, 113, 120

[209] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, *Dive into Deep Learning*. Cambridge University Press, 2023, https://D2L.ai. 110

[210] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wiessner, "Microscopic traffic simulation using sumo," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 2575–2582. 114

[211] OpenStreetMap contributors, "Openstreetmap," 2025, accessed: 2025-03-27. [Online]. Available: https://www.openstreetmap.org/ 115

[212] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 1135–1144. [Online]. Available: https://doi.org/10.1145/2939672.2939778 121

[213] C. Legaard, T. Schranz, G. Schweiger, J. Drgoňa, B. Falay, C. Gomes, A. Iosifidis, M. Abkar, and P. Larsen, "Constructing neural network based models for simulating dynamical systems," *ACM Computing Surveys*, vol. 55, no. 11, pp. 1–34, Feb. 2023. 121

[214] B. Sanguino, "Parameter estimation and analysis of wind impact on ship maneuvering model," Master's thesis, NTNU, 2024. 121

[215] M. Rätz, P. Henkel, P. Stoffel, R. Streblow, and D. Müller, "Identifying the validity domain of machine learning models in building energy systems," *Energy and AI*, vol. 15, p. 100324, Jan. 2024. 121

[216] R. Franceschini, S. V. Mierlo, and H. Vangheluwe, "Towards adaptive abstraction in agent based simulation," in *2019 Winter Simulation Conference (WSC)*, 2019, pp. 2725–2736. 121, 122

[217] H. Ejersbo, K. Lausdahl, M. Frasheri, and L. Esterle, "Dynamic runtime integration of new models in digital twins," in *2023 IEEE/ACM 18th Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, May 2023, pp. 44–55. 122

[218] Arizona Department of Transportation, "Traffic group photolog summary report: Tgp0621," https://azdot.gov/sites/default/files/2019/05/tgp0621-2018-01.pdf, 2018, accessed: 2025-04-10. 123

[219] R. Biglari, C. Gomes, and J. Denil, "A validity frame for multi-modal surrogate models," 2025, manuscript in preparation for submission to SIMPAT. 127

[220] R. Biglari, J. Michael, J. Denil, and H. Vangheluwe, "On quality of digital twin," 2025, manuscript in preparation for submission to the EDT Conference. 127

Appendix A

List of Symbols

Abbreviations & Acronyms

ABM	Agent Based Model
ANN	Artificial Neural Network
AV	Autonomous Vehicle
BCET	Best Case Execution Time
C.A	Constant Acceleration
CMIP	Coupled Model Intercomparison Project
CPS	Cyber-Physical System
C.V	Constant Velocity
DNN	Deep Neural Network
DSL	Domain-Specific Language
DT	Digital Twin
EBM	Equation Based Model
ESS	EMF-Based Simulation Specification
FMI	Functional Mock-up Interface
FMU	Functional Mock-up Unit
GCM	Global Climate Models
GE	General Electric
ISO	International Organisation for Standardisation
ITU	International Telecommunication Union
KIB	Knowledge Interchange Broker
LCM	Lane Change Maneuver
LR	Linear Regression
M&S	Modelling and Simulation
MAE	Mean Absolute Error
MIO	Most Important Object

ML	Machine Learning
MPM	Multi Paradigm Modelling
MRM	Multi Resolution Modelling
OSM	OpenStreetMap
PF	Particle Filter
PLM	Product Lifecycle Management
PoI	Properties of Interest
poset	partially ordered set
RMA	Rate Monotonic Scheduling
RtS	Real-Time Simulation
SACube	Self Adaptive Abstraction and Approximation
SDF	Synchronous Data Flow
SSP	System Structure and Parameterisation
SUMO	Simulation of Urban MObility
TraCI	Traffic Control Interface
VRM	Variable-Resolution Modelling
VS	Virtual Space
V&V	Verification and Validation
WCET	Worst Case Execution Time

Appendix B

Data and Code Availability

To support reproducibility and further exploration, all necessary datasets, source code, and experimental configurations developed for this thesis are available via the author's personal webpage:

`https://rahelehbiglari.github.io/`

Appendix C

List of Publications

The following articles, which I (co-)authored serve as direct input for this dissertation: following publication:

- R. Biglari, J. Mertens, and J. Denil, "Towards Real-time Adaptive Approximation," in *2022 11th European Congress Embedded Real Time Systems - ERTS 2022*, Toulouse, France, Jun. 2022.
- R. Biglari and J. Denil, "Model validity and tolerance quantification for real-time adaptive approximation," in *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, ser. MODELS '22, 2022.
- Z. Ali, R. Biglari, J. Denil, J. Mertens, M. Poursoltan, and M. K. Traoré, "From modeling and simulation to digital twin: evolution or revolution?" *SIMULATION*, vol. 100, no. 7, pp. 751–769, 2024.
- R. Biglari and J. Denil, "Self-adaptive abstraction and approximation (sacube) framework for digital twins with real-time requirements," *SIMULATION*, 2025, under peer review.
- R. Biglari, C. Gomes, and J. Denil, "Towards a validity frame of multi-modal surrogate models for traffic simulation," in *2025 Annual Modeling and Simulation Conference (ANNSIM)*, 2025, in press.

Moreover, there are also some journal and conference papers currently in preparation. Note that there is no guarantee that these will be accepted. Below, the working titles are listed.

- R. Biglari, J. Michael, J. Denil, and H. Vangheluwe, "On quality of digital twin," 2025, manuscript in preparation for submission to the EDT Conference.
- R. Biglari, S. Bellis, and J. Denil, "Decision-centric: A method for digital twins demonstrated through an autoclave use case," 2025, manuscript in preparation.
- R. Biglari, C. Gomes, and J. Denil, "A validity frame for multi-modal surrogate models," 2025, manuscript in preparation for submission to SIMPAT.