Adaptive hybrid reasoning for agent-based digital twins of distributed multi-robot systems

Adaptive Hybrid Reasoning for Agent-based Digital Twins of Distributed Multi-robot Systems

Hussein Marah[1,2*] and Moharram Challenger[1,2]

[1*]Department of Computer Science, University of Antwerp, Middelheimlaan 1, 2020 Antwerpen, Flanders, Belgium.
[2*]AnSyMo/CoSys Core-lab, Flanders Make Strategic Research Center, Leuven, 3001, Flanders, Belgium.

*Corresponding author(s). E-mail(s): hussein.marah@uantwerpen.be;
Contributing authors: moharram.challenger@uantwerpen.be;

Abstract

Digital Twin (DT) mainly acts as a virtual exemplification of a real-world entity, system, or process via multi-physical and logical models, allowing the capture and synchronisation of its functions and attributes. The bridge between the actual system and the digital realm can be utilized to optimize the system's performance, forecast and predict its behaviour. Incorporating intelligent and adaptive reasoning mechanisms into DTs is crucial to enable them to reason, adapt, and take efficacious actions in complex and dynamic environments. To this end, we introduce an approach for deploying agent-based digital twins for cyber-physical systems. The foundation pillars of this approach are (1) integrating the concepts, entities, and relations of Zeigler's modelling and simulation framework from the perspective of agent-based digital twins; (2) utilizing an expandable and scalable architecture for designing and materializing these twins, which handily enables extending and scaling physical and digital assets of the system; finally, (3) a two-tier reasoning strategy; reactive and rational models are conceptually redefined in the context of modelling and simulation framework of agent-based digital twins and technically deployed to boost adaptive reasoning and decision-making function of DTs. As a result, an implemented simulation and control platform for a multi-robot system demonstrates the approach's applicability and feasibility, manifesting its usability and efficiency. The platform represents physical entities as autonomous agents, creates their DTs, and assigns adequate reasoning capability to promote adaptive planning, autonomous resource management, and flexible logical decision-making to handle different situations and scenarios.

Keywords: Agent-based Modelling, Multi-agent Systems, Digital Twins, Reasoning, Multi-robot Systems

1 Introduction

Over the past decade, significant and rapid technological progress has been made in various fields, leading to a new era in industry and manufacturing. This shift in technology, driven by digitization, leads to replacing traditional systems with smart, self-organized, and adaptive systems as requirements and the complexity of systems significantly increase.[49] Besides achieving interoperability and easy integration with other solutions, new systems promise to provide better efficiency, accuracy, speed, and reliability.

Digital Twin (DT)[18] is becoming a momentous technological concept and primary tool to achieve digitization and digital transformation in different domains, such as smart manufacturing.[65] With the emergence of DT technology, previously challenging and tedious tasks can now be performed virtually while connecting with the real-world environment. A dependable way of designing, testing, and studying a system is by using computer simulation. With respect to that and not exclusively, computer simulation is used to understand a system and develop an operational solution.[40] Consequently, exploiting and integrating DT as a simulation tool has countless opportunities and advantages.

Cyber-physical systems (CPS) and the Internet of Things (IoT)[22, 42] play a crucial role in fulfilling the requirements for building complex physical systems. However, CPS/IoT contains multiple interconnected parts and components. Their complexity increases as more subsystems and sub-components are included and connected. Exploiting DT capabilities for these systems is crucial in tackling the complexity and enhancing various aspects of the system's life cycle. However, creating DT for complex, large-scale, distributed, and heterogeneous systems such as CPS is a non-trivial task.

From this perspective, agent technology and multi-agent systems (MAS) have several advantages to seamlessly design and build complex industrial CPSs[32] and create/integrate/deploy their DTs.[36] Some of their significant merits: (1) modularity of agents enables easy scaling up/down of the designed system depending on its complexity and requirements; (2) distributed agent architecture supports parallel control/processing and decentralized decision-making; (3) the high degree of flexibility and adaptability make agents well-suited for designing, modelling, and deploying complex and dynamic systems that contain a massive number of components whose requirements are subject to updates, changes, and modifications; (4) intelligent and collaborative decision-making of agents is ideal for representing systems where multiple actors must collaborate to accomplish a mutual objective. Overall, using agents and the MAS paradigm can make a leap forward and support the design and assembly of flexible, adaptable, scalable, sustainable, and robust DTs for CPS.

Multi-robot system (MRS)[53] is a type of robotic CPS that involves multiple robots working together. These robots may be physically connected or completely separate and have different capabilities and roles within the system. MRS is a complex and dynamic system that requires coordination capabilities, allowing the robots to communicate and collaborate effectively to achieve their tasks accurately. According to the current statistics and a forecast prepared by the International Federation of Robotics (IFR)[1], the number of operational robots, especially in the industry has increased during the last few years and will continue to do so, as reported in Fig. 1. Besides industry, robotic systems such as MRS can be leveraged in various applications, such as logistics, medical, and environmental domains.

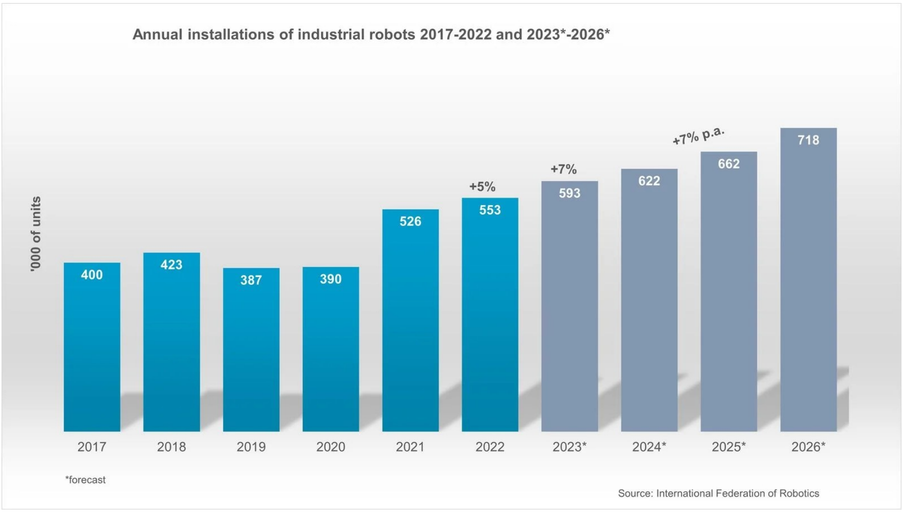


Fig. 1 Annual installation of industrial robots.

Leveraging the advantages of agents and MAS technology, a distributed, flexible, adaptable, and scalable agent-based[31] cognitive DTs[1, 14, 3] can be designed and implemented for any CPS such as MRS. Multiple heterogeneous elements and components of a system can be associated with agents to support decision-making and problem-solving procedures during the operation of

[1]International Federation of Robotics - https://ifr.org/

agents in a dynamically changing environment. To realize this conceptualization, different types of autonomous agents that encompass simple and complex cognitive reasoning capabilities should be incorporated into the DTs of the system components (e.g., robots) to enable them to solve any problem they face efficiently and make the most appropriate actions according to the situation and the scenario.

However, supporting DTs with simple and advanced cognitive decision-making capabilities requires understanding the requirements of the corresponding physical twins and how they behave/operate in the real environment. In addition, incorporating different operational characteristics, behaviours, and actions inside an individual agent may cause a conflict in their decisions and result in contradicting solutions and actions for a specific problem. Consequently, it is of paramount importance to provide a framework amalgamated and empowered by modelling and simulation capabilities for managing and orchestrating different types of agents within the DT of the system and making them available seamlessly on the fly so the system operator can painlessly assign the proper type of agent reasoning for the suitable task.

For instance, in a scenario where real-time stimuli/action is needed, an agent with reactive capabilities can be assigned to this situation where the response/action of the agent is based on perception, and actions are taken in a real-time manner. On the other hand, rational agents are designed to handle dynamic and uncertain environments and provide goal-oriented reasoning and long-term cognitive thinking.[3]

The two primary types of agents, reactive and cognitive, present diametrically opposing solutions, but in reality, they can be seen as complementary. Hence, it is feasible to develop hybrid architectures that integrate both approaches to construct more adaptable and flexible solutions (agents) for problem-solving by considering factors like response time, precision, efficiency, and long-term benefits.[41]

Aligning with what was previously mentioned, the major contributions of this paper are enumerated in the following points:

1. It strives to introduce a conceptual framework that takes advantage of agent technology and uses a well-founded architecture for designing DTs for complex CPS according to the modelling and simulation concepts and relations.
2. It expounds the twinning concept in the proposed framework and materializes it between physical and digital worlds via agents; besides, it incorporates a reactive and rational decision-making capacity into DTs of the actual system and redefines how these cognition abilities operate and communicate inside the physical and digital representations of a DT.
3. It implements a simulation and control platform that deploys agents representing the physical system of the MRS and its DTs. The platform offers multi-perspective reasoning and cognitive mechanisms (e.g., reactive and rational). It can assign reasoning methods for the robot's DTs on the fly, and it enables real-time control and simulation of the physical robots by means

of their DTs, which analyze and examine the available knowledge and environment dynamics that the actual systems convey. Then, they extrapolate and take actions and decisions based on their reasoning capabilities.

2 Background

This section presents the background and the basic concepts of intelligent agents and multi-agent systems, their application in the domain of CPS and multi-robotic systems, the DT concept, and finally, the positioning technology used for navigation in MRS.

2.1 Intelligent Agents and Multi-Agent Systems

The term software agent has roots in the early times of computer science and artificial intelligence (AI) discipline. Obviously, software or a computer system operating in an environment and taking action to achieve its objective is an agent. Following this definition, agents that function stably and rigidly in dynamic, unpredictable, rapidly, and constantly changing environments are considered intelligent or autonomous agents[60] due to their abilities to adapt, react, and take proactive procedures. In alignment with this definition, a MAS is a system that consists of multiple agents deployed in an environment to influence its perimeter. The primary objective of a MAS is to solve complex distributed problems autonomously and collaboratively, which cannot be solved by individual agents.

Several systems require more advanced, adaptive, and autonomous decision-making mechanisms[58, 13] that can be efficient, reliable, and agile as some systems are time-sensitive and must take immediate actions in specific situations. Thus, a dedicated intelligent and autonomous software agent should be in charge of executing such missions. However, in general, and at a high level of the agent theory, an agent can embody various forms of entities, such as a machine, part or software.[2]

Conventionally, to call software as *intelligent* is a philosophical and open argument. But at least minimum requirements should be met to have intelligent characteristics in software. In this paper, we define intelligent agent software as one that operates in its environment robustly and flexibly, avoids failures, and eventually is capable of achieving its designed objectives.[60, 2] In the following points, we summarise some properties and characteristics of ***Intelligent Agents***.[60]

- **Reactivity**: in order to understand the environment and its surroundings, an intelligent agent should be able to perceive and sense the environment, respond and act accordingly.
- **Pro-activity**: agent pro-activity is the ability to initiate and operate based on goal-oriented behaviour that makes the agent able to reach its goals and objectives without waiting for external triggers

- **Sociability**: intelligent agents should communicate and interact with other agents in the same or an external environment; this interaction could be as a source of information required to complete a specific task internally or to reach a mutual objective on the level of a whole system.

Beyond the abstract definition of agents, an agent can be classified based on its internal structure and how it functions and operates in the environment.[60] For instance, there are two common architectures of agents: ***reactive*** model and ***rational*** model, such as the belief-desire–intention (BDI).[10, 52]

2.2 CPS and Multi-Robotic Systems

CPS is a type of system that integrates both physical and computational components. CPS usually requires interaction between physical and virtual computational entities to observe and perceive the environment using physical components, such as sensors, or control and regulate the environment using actuators.

MRS is a system that comprises multiple robots (i.e., CPSs or Cobot) that work together to achieve their internal goals and the system's common objective.[53] These robots can be either physically linked or separated, as they have different roles, capabilities, and features within the system's functionality. MRS can be classified into two types: (1) homogeneous, which is composed of robots that have the same physical characteristics and capabilities, and (2) heterogeneous, which is comprised of robots that have distinct physical characteristics and components.

Integrating the software agent into CPS, such as MRS, may facilitate the programming of these systems at a higher level of abstraction than conventional methods. When a software agent deployed into the cyber part acquires control of the somatic components of a CPS instance, it can influence its environment via behavioural definition. However, deploying these agents onto an existing system may not be feasible considering the cost, safety, and size that emerged from the physical constraints. The target system can be scaled down as an alternative, preserving the main functions and requirements and considering composability and modifiability. The LEGO may be utilized as a technology[44, 8] to mimic a CPS instance as it has (de)-composable and reusable concrete materials, including sensors and actuators. Consequently, LEGO was preferred for providing miniaturization and instrumentation for actual robotic CPS implementations for the purpose of research and experiments.

2.3 Digital Twin (DT)

Life-cycle management of complex systems requires new approaches to reduce resource use, cost, and time, from development and deployment phases to run-time sustainability. Modelling and simulating these physical systems in the cyber world (i.e., virtual space) can be a solution to achieve the mentioned aims and provide better interpretation and monitoring of the system's behaviours. DT, which inherits the modelling and simulation methods, is an advancing concept to deal with the various behavioural problems that emerge in the complex system of systems.[19]

The DT can be defined as a technology that digitalizes a physical entity, object, or process based on the feature of interest in those entities. This twin can be constructed in the cyber domain from the flow of digital information between itself and the system it represents through its life cycle. A real-time data flow between the physical and digital parts is essential to represent the twinned entity digitally. The continuous data gathered on the cyber side can be used to develop a new product version, improve the operational system, manage and analyze its behaviour, reason about its current status, and predict its future attitude.

Generally, a complex system and its functionalities may require a deep understanding of its heterogeneous correlated sub-functions and components. Therefore, the twins' data play a vital role, and it can be visualized to provide insights for the human actors, raising awareness and allowing better interpretation of these complex systems. The past states of the system can also be preserved as historical data and evaluated irrespective of the current state of their physical counterparts.

Moreover, the collected data can be perused to predict the physical system's global status or any centre of attention property based on its twin. This way, unpredictable and undesirable emerging behaviours/scenarios can be minimized or even mitigated. In our study, the system's DTs (atomic twins) control the agent-based CPS instances, collect the data, analyze the status of each instance, and send action and feedback to the corresponding cyber twin according to their states.

2.4 Positioning System

The technology used in this study for positioning and localization of CPS elements is the Ultra-Wide-Band (UWB). It is a radio technology that operates in a short range and consumes low energy. Its high bandwidth makes it a suitable choice for indoor environments as it is more interference-resistant than Wi-Fi technologies. Therefore, it works in a concentrated material environment with high accuracy.[30] This way, mobile MRSs utilize this technology to achieve indoor localization. In addition, UWB signals can operate in an environment where other short/large-range wireless communications signals exist. This enables us to use Wi-Fi-based embedded boards to achieve wireless communication and deploy agent software on mobile robots.

3 Related Work

Due to the wide range of DT applications, the research community and industry have put much effort into realizing and implementing dependable DT systems. Thus, multiple literature review studies have been conducted to explore the potential of the twinning paradigm, enabling technologies, challenges, and road blockers, to name but a few.[61, 22, 23, 56]

Building a DT for a CPS system typically involves mimicking the behaviour and collecting information about physical components to form virtual representations. DTs' distributed and heterogeneous nature, as they compose physical and cyber components and sub-processes, raises several challenges that hinder DTs' easy design and deployment. Thus, well-grounded technologies and design approaches are necessary to address and handle these challenges. In our study, we aim to leverage the agent-based paradigm as an enabling technology to address several challenges, such as representing heterogeneous physical and digital actors, implementing intelligent reasoning mechanisms, and managing the distributed nature of CPS systems as we deploy our solution in a distributed multi-robotic system.

This section categorizes related works into four categories: intelligent agents and MAS, agent-based CPS and robotic systems, DTs for multi-robotic systems, and agent-based DTs. A comparison table is then provided to show the gap we aim to fill with this study.

3.1 Agents and Multi-Agent Systems

Nowadays, MAS is widely regarded as a practical approach and efficient programming paradigm for modelling, designing, and implementing different distributed system applications and understanding the complex interactions between individual entities.[40] Different systems are deployed using the MAS approach in grid energy management[54], energy optimization[17], oil and gas industry[20], manufacturing and control[50], supply chain and logistics[26], robotics[48] and numerous other domains[50].

This broad utilization of MAS and agents in various domains makes them an interesting topic for academic research and industrial applications. Thus, several surveys and reviews have been conducted to expose the potential, advantages, challenges, and current trends of MAS in different topics.[43, 33, 41, 62, 34, 25] In essence, the core scope of MAS mainly focuses on designing, simulating, and implementing systems where many components interact with each other, and they have complex dynamics, significant variability, and different constraints.[41]

Now, more than ever, MAS relates to the realms of industrial digital transformation and achieving sustainability in industrial solutions.[25] Nevertheless, the advent of CPS and its implementation in different fields has opened up opportunities for agent-based methodologies. The rationale behind this lies in the seamless alignment of agents' inherent attributes and abilities, which can be harnessed to tackle the CPS challenges effectively.[24]

Agent technology could powerfully support dealing with and managing CPS from different aspects and be one of the key enablers for realizing Industry 4.0-compliant CPS solutions. For this reason, this paper delves into the theoretical and technical implementation of MAS and intelligent agents. It promotes deploying DTs for CPS integrated with advanced reasoning mechanisms powered by MAS and agent paradigms.

3.2 Multi Agent CPS and Robotic Systems

Over the years since the agent-based theory was founded, more attention has focused on providing agent-based conceptual architectures, middle-ware structures, and implementations for real-world applications.

Soriano et al.[57] proposed a multi-agent platform to address collision avoidance in mobile robot systems using a systematic approach. They utilize the advantages and cooperative capabilities of MAS. Their proposed method has four stages: detecting collisions, identifying obstacles, negotiating, and avoiding collisions.

Lee et al.[28] introduced a five-layer architecture to implement CPS for Industry 4.0-oriented manufacturing systems. Their proposed architecture consists of a smart connection level for acquiring data from physical components, a data-to-information conversion, a cyber level as a central information hub, a cognition layer for generating knowledge of the component, and lastly, a configuration level for providing feedback from the cyber domain to the physical world.

Introducing an architecture for agent-based CPS, Sanislav et al. [55] considered a cloud layer where simple-reflex agents developed using the JAVA Agent DEvelopment Framework (JADE) are located. The middle networking layer delivers the control actions to the bottom sensing and actuation layer. The information exchange between the CPS components is done in the networking layer. The approach provides benefits such as modular composition, efficient resource utilization, and adaptation. Modularisation can enable flexible and scalable system extension in a combinational manner. The collected data and captured events can be used for optimization. Adapting to environmental changes can be achieved by providing self-adaptive capacity based on the data collected from the operation perimeter.

Qi et al.[51] inspected the CPSs' evolution through the DT paradigm. After defining the cyber and physical world spaces, the DT enhances the CPS interaction and integration, considering real-time data flow. Moreover, they examine the harmony of the DT and CPS under three levels: unit level, system level, and system of systems (SoS) level.

Yahouni et al.[63] presented a three-layer conceptual architecture similar to what Sanislav et al.[55] demonstrated in a previous study. Agents are employed to program CPSs in the manufacturing domain. The agents are deployed with JADE and are utilized at the management layer to gather data and extract the KPIs of the system for decision-making, while the application layer retains databases, algorithms, and system status. Lastly, the physical layer is defined for sensing and actuating purposes.

Onyedinma et al.[47] explained connecting the Robot Operating System (ROS) with the BDI reasoning mechanism. This was demonstrated in a simulated environment. ROS manages the links to low-level sensors and actuators, while the BDI reasoning process manages high-level reasoning and decision-making. Sensory information is sent to the reasoning system as perceptions through ROS. The perceptions are analyzed, and an action string is returned to ROS for interpretation, which makes the required actuator act accordingly.

3.3 Agent-Based Digital Twins for CPS

Agent-based digital twin extends the capabilities and the value of the agent-based CPS by creating digital representatives of the agentive versions of the physical instances. This section highlights prior works on DTs, focusing on those incorporating the agent paradigm.

Focusing on cloud technology, Alam and El Saddik[4] propounded an architecture for a cloud-based CPS DT (C2PS) using a model that helps to identify the level of interaction between basic and hybrid modes of computation, communication, and control. The implementation connects physical components with cloud-based counterparts and uses a Bayesian belief network and fuzzy rules for self-reconfiguration.

In another DT application, Braglia et al.[9] presented an agent-based simulation model for a paper products warehouse that uses UHF RFID technology and DT to optimize routes and handle congestion.

Ambra and MacHaris[6] demonstrated a proof-of-concept of constructing a DT by integrating physical and virtual spaces, using ABM and MAS to provide self-awareness to agents.

Considering the IoT technologies, Clemen et al.[11] discussed implementing a DT in smart cities, using IoT tools and modelling and simulation approaches with the MARS framework for large-scale MAS.

Zheng et al.[65] suggested a DT modelling approach based on a MAS architecture. The method concentrates on quality control during the manufacturing processes and offers solutions to gather relevant information and assess the effects on product quality. The model is based on five parts: physical entities, virtual models, DT data, services, and finally, communication channels between components.

Lee et al. [28] provided guidelines for implementing CPS based on a unified 5-level architecture. Later on, Latsou et al.[27] adopted that architecture and provided a 5-layer architecture for the deployment of DTs and agent-based CPS by integrating a multi-agent cyber-physical manufacturing system (CPMS) and RFID technology to enhance the traceability and trackability of complicated manufacturing processes. The implementation considers interactions within a single complex manufacturing system and between different locations within a supply chain.

In another industrial application, Ocker et al.[46] described utilizing DTs, specifically the Asset Administration Shell (AAS), to implement MASs in the manufacturing industry. A parser gathers information automatically from the

DTs and initializes agents in a MAS, such as the Python agent development framework (PADE).

Recently, in 2021, Erkoyuncu et al.[15] discussed an intelligent agent-based architecture for DTs. The architecture is utilized to enhance the DT's robustness (including the accuracy of representation) and resilience (including prompt updates). The approach is demonstrated using a case study of cryogenic secondary manufacturing.

3.4 Digital Twins for Robotic Systems

In the extension of a previous conceptual design and architecture.[28], Lee et al.[29] considered integrating DT and learning approaches to enable the shift towards intelligent manufacturing and Industry 4.0 as the implementation has been utilized in a shop floor case study.

Zong et al.[66] presented a method for a multi-robot monitoring system using DT technology. The system gathers information through the data exchange standard (OPC UA) to simulate the motion of a six-degree-of-freedom robot (6-DOF) and provide collision detection during multi-robot collaboration.

Table 1 compares related works reported previously. From the investigated literature, we can conclude that providing solutions to CPS based on agents and/or integrated with DTs has received considerable attention in recent years. Even though DT implementations cover multiple domains, we observed a shortage of research in the domain of agent-based digital twins for MRS.

For instance, Alzetta and Giorgini[5] presented a BDI agent-based solution for multi-robotics, but a DT integration was not considered. In contrast, other researchers presented solutions targeting agent-based CPS driven by DT [65, 15] in the manufacturing domain. In addition, Zong et al.[66] introduced a DT for a multi-robot system for manufacturing. Meng, Wei, et al.[39] focused on the initial exploration of combining DT, 5G, cloud platforms, and virtual reality (VR) technologies to advance the development of autonomy in unmanned aerial vehicles (UAVs). Regardless, the agent paradigm was not utilized to design and build the system in the two latter. Besides, no reasoning mechanisms were provided in all the aforementioned DTs.

In a nutshell, this paper attacks the gap of lacking agent-based digital twins for CPS supported by modelling and simulation capabilities, besides the non-availability of implementations that support adaptive multi-logical reasoning mechanisms for DTs. It applies the principles and concepts of the agent paradigm for conceiving and building DTs customized to the field of mobile MRS. Also, it provides a novel approach for defining and simulating agents during run-time and determining their reasoning model that exemplifies and supervises the internal agent behaviour, making agents more context-aware inside the dynamic environment.

Table 1 Comparison table of the related work

Paper	Agent Integration	DT Deployment	Multi-Reasoning	Application Domain
Lee et al. (2015)[28]	✓	✗	✗	Manufacturing and Supply Chain
Erkoyuncu et al. (2021)[15]	✓	✓	✗	Manufacturing
Soriano et al. (2013)[57]	✓	✗	✗	Multi-robotic System
Zong et al. (2021)[66]	✗	✓	✗	Smart Manufacturing and Multi-robotic System
Clemen et al. (2021)[11]	✓	✓	✗	Smart Cities
Alzetta and Giorgini (2019)[5]	✓	✗	✗	Multi-robotic System
Zheng et al. (2020)[65]	✓	✓	✗	Manufacturing
Meng, Wei, et al. (2023)[39]	✗	✓	✗	Robotics
This paper	✓	✓	✓	CPS & Robotics

4 The Proposed Approach

In this section, we delve into our overall approach, starting with introducing the Modelling and Simulation Framework (MSF) for agent-based digital twins, then discussing the fundamental and conceptual reasoning methods; after that, we describe the adopted architecture for deploying agent-based digital twins, and finally, we conclude with the implementation-specific details of the system deployment.

4.1 Modelling and Simulation Framework (MSF)

From a foundation and conceptual point of view, Zeigler et al.[64] developed an MSF that outlines specifications, basic entities, and relations of modelling and simulation (M&S). Fundamentally, the Zeigler MSF is composed of five primary elements:

1. **Source System:** represents a source of data, which can be from a virtual or real system.
2. **Behaviour Database:** the collected data from the system and the environment.
3. **Experimental Frame:** specifies under which conditions the experiments and observations are conducted.
4. **Model:** is a specification and an abstract representation of the system.

5. **Simulator:** the computational tool that generates the model's behaviours.

However, Zeigler's work on MSF did not discuss the notion of twinning or the integration of agents into the framework. From this perspective, our developed MSF builds upon these presented concepts, adopts and extends the fundamental entities, and adheres to the defined relations to be valid in the field of modelling and simulation of agent-based digital twins. Therefore, eight main entities are proposed that constitute the novel MSF specific for agent-based digital twins, as illustrated in Fig. 2.

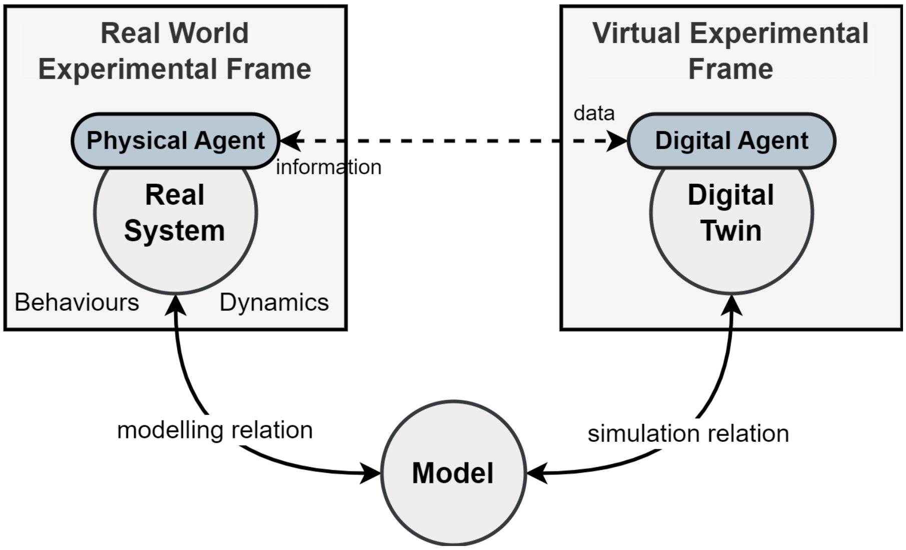


Fig. 2 The MSF of agent-based digital twins.

The essential entities of the agent-based digital twin MSF are clarified in the following points:

1. **Real System:** represents the actual (e.g., physical or virtual) twinned system that is the origin and main source of data in the framework.
2. **Digital Twin:** is a multi-model and frequently updated digital representation of the real system.
3. **Digital Agent (DA):** is an agent representation of the Digital Twin in a virtual environment.
4. **Physical Agent (PA):** is the agent that encompasses the real system's behaviours, capabilities, and features and interacts with the physical environment.
5. **Behaviours and Dynamics:** are a general representation space of the events, actions, and reactions in a dynamic environment responding to stimuli or specific situations.

6. **Virtual Experimental Frame:** specifies the parameters and conditions of the simulated experiments.
7. **Real World Experimental Frame:** determines the controllable parameters and conditions of the experiments in the physical environment to answer the raised questions.
8. **Model:** is a formal specification/representation (physical, mathematical, or logical) of the mapping between virtual and physical entities at different levels of abstraction.

All these entities and concepts are either modelled and included explicitly or implicitly in our approach and inside the implemented simulation and control platform.

4.2 Agent Reasoning Strategies

The MSF of agent-based digital twins describes the twinning relation of the `PA` representing the actual (real) system and its `DA` resembling the DT. Making this bare-bone MSF of agent-based digital twins more intelligent, capable, and effective in dealing with real-world problems, another layer is needed to provide the agents of DTs with different decision-making processes. For that reason, combining reactive and cognitive (rational) reasoning strategies[16, 52] in the context of DTs can offer significant and practical benefits in the decision-making process in real-world scenarios and situations. Hence, incorporating reactive and rational agents, such as BDI, into the MSF of agent-based digital twins provides several advantages, such as adaptability, real-time responsiveness, long-term planning and decision-making, and handling uncertainty in DTs.

For example, reactive agents excel at real-time responsiveness, enabling a DT to react quickly to sudden changes or events in the physical world. In contrast, BDI agents enable a DT to utilize efficient planning and consider long-term and future-oriented goals to make informed decisions based on reasoning about beliefs, desires, and intentions. BDI agents engage in reasoning based on knowledge expressed using a symbolic formalism. This knowledge explicitly captures information about their environment, including states, properties, and dynamics of objects within the environment and information about other agents involved.[41]

In real-world systems, uncertainties are inevitable. Reactive agents might struggle to handle uncertain situations that deviate from their predefined rules. However, BDI agents can utilize their flexible reasoning mechanism to reason about uncertainties, making the DT more robust and reliable in uncertain or unpredictable scenarios and adapting to dynamic and complex conditions. This flexibility of having more than one type of reasoning in the MSF of the agent-based digital twins boosts the performance of a physical system that might encounter different situations that require different problem-solving techniques. Table. 2 compares the planning and the behaviour of the two reasoning methods that are exploited in our approach.

4.2.1 Reactive Agent:

This part provides a description of the abstract reactive agent model using a pseudo-mathematical notation where the function and the relation of the PA and the DA are represented as the following. Let:

- $Agent_{reactive}$ denotes the reactive agent's behaviour.
- I_{PA} be the set of collected sensory inputs (perceptions) in the PA that determines its current state S_{PA}.
- A_{DA} be the set of possible action/s that are executed by the DA according to the satisfied rule/s from the R_{DA}.

$$Agent_{reactive}(I_{PA}, S_{PA}) \to (R_{DA}, A_{DA})$$

A set of predefined reactive rules describe the agent's behaviour as the following:

$$\text{Rule}_i : \text{if condition}_i \text{ then action}_i, i \in 1, ..., n$$

The condition$_i$ is a particular situation evaluated by rules based on sensory inputs I_{PA} and the state S_{PA} of the real system indicated in the PA, while the action$_i$ is the selected action/s from the A_{DA} determined by the fulfilled rule/s from the R_{DA} in the DA. If that condition is satisfied, the latter executes the action in the simulation environment; simultaneously, the action is sent to the PA to be executed and influence the real environment.

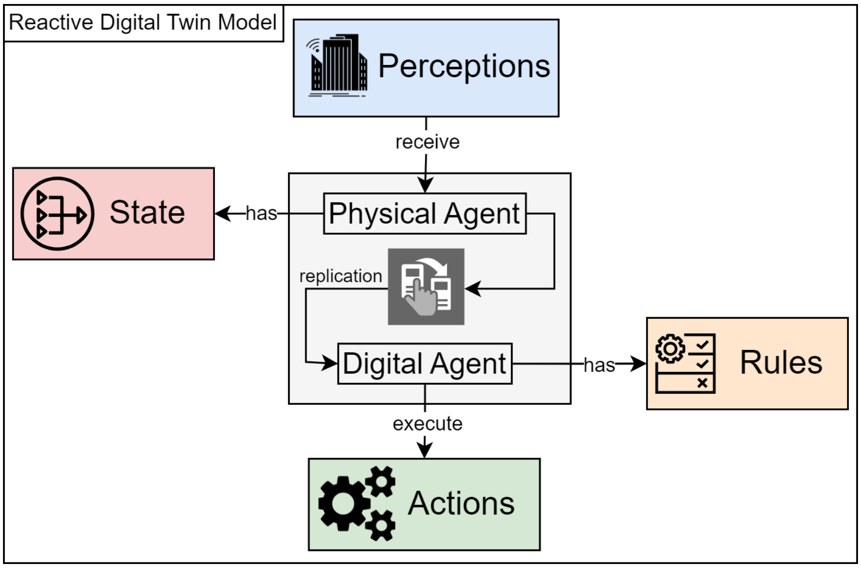


Fig. 3 Reactive reasoning model in a digital twin.

For adapting reactive behaviour in our approach, Fig. 3 conveys the procedures of making decisions based on the perceptions handled by the PA. In the *replication* process, the DA is created to represent an up-to-date instance of the PA and its current status in a digital environment. In fact, the PA updates its state and propagates these updates to the DA frequently. In addition, the PA receives the action/s from the DA to influence the real world; concurrently, the latter affects the digital world.

4.2.2 Rational Agent:

In contrast to reactive agents, rational agents such as BDI exploit their capability to select a proper plan based on their collected information about the world and the current goals they want to achieve. This behaviour makes them more autonomous and flexible in different circumstances. The logic of the goal-oriented behaviour of BDI agent is described using a more complex formalism based on beliefs, desires, and intentions.[12, 52] In addition, the inclusion of PAs and DAs and their coordination in the deliberation cycle is depicted in Fig. 4. Also, the abstract model is described using the following notation. Let:

- $Agent_{bdi}$ denotes the BDI agent's behaviour.
- B_{PA} be the set of beliefs (dynamic internal knowledge base about the real world) of the PA.
- D_{PA} be the set of designated desires or goals of the PA.
- I_{DA} be the set of the defined intentions (representing the agent's plans to achieve goals) of the DA.

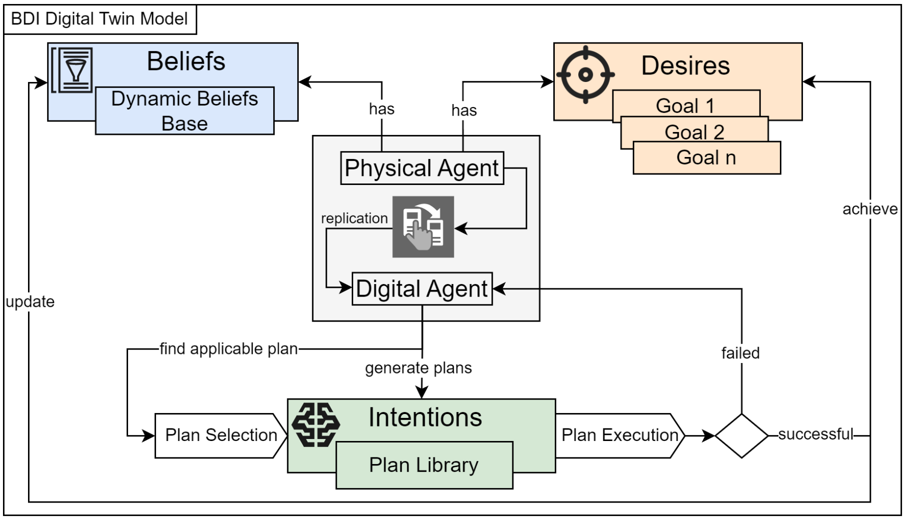


Fig. 4 BDI reasoning model in a digital twin.

The general structure of the BDI model maps the belief set and the desires of the PA to generate possible intentions that will be translated into actions by the DA.

$$Agent_{bdi}(B_{PA}, D_{PA}) \rightarrow (I_{DA})$$

Overall, the BDI model's decision-making process and its behaviour cycle are described by the interaction between the PA and DA. This step entails updating the belief set of the PA within a dynamic environment, appointing its preferred desire, forming an intention in the DA, and executing an action according to the formed intention. The details are described in the following steps:

1. **Belief Update**: The agent receives sensory inputs S_{PA} and updates its beliefs accordingly.

$$B'_{PA} = UpdateBeliefs(S_{PA}, B_{PA})$$

2. **Desire Selection**: The agent evaluates the set of desires D_{PA}, and then selects a desire PA_d wants to achieve based on the updated beliefs and its internal motivation.

$$PA_d = SelectDesire(B'_{PA}, D_{PA})$$

3. **Intention Formation**: The agent forms an intention DA_i that represents a plan to achieve the selected desire PA_d.

$$DA_i = FormIntention(I_{DA}, PA_d)$$

4. **Plan and Action Execution**: The agent takes action/s DA_a based on the plan execution of the formed intention DA_i in the DA.

$$DA_a = ExecutePlan(DA_i)$$

The PA regularly updates its current status and thus refreshes its knowledge and beliefs using ***UpdateBeliefs*** function. Then, it delegates a goal as requested/needed using the ***SelectDesire*** function. Finally, it sends all the relevant and required data/information and selected goals to its replica DA once it is created in the *replication* process. The DA then handles the decision-making capability via the two functions, ***FormIntention*** and ***ExecutePlan***, which are pursued sequentially to form and execute the best plan, turning intentions into actions.

4.3 System Architecture

The architecture[37] of the agent-based digital twin utilized in this paper is based on a previous work, where it introduced the main components and the workflow[36] for designing and building intelligent agent-driven digital twins for CPS.[35] So, this section covers the fundamental and essential parts of the architecture used to develop this solution. Fig. 5 illustrates the generic high-level representation of the agent-based digital twin MSF for an MRS.

Table 2 Comparison between characteristics of Reactive and BDI agents.

Criteria	**Reactive Agent(s)**	**BDI Agent(s)**
Design Paradigm	Follow a reactive design paradigm. A Set of predefined rules or mappings from sensory inputs to actions primarily determine their behaviours.	Follow a more sophisticated design paradigm. They execute reasoning procedures based on their beliefs, desires (goals), and intentions (plans).
Knowledge Representation	Use simple reactive information of the immediate sensory inputs for knowledge representation.	Maintain the knowledge about their current state and the world as beliefs.
Goal-Directed Behaviour	They have stimulus-driven behaviour, responding directly to environmental triggers without long-term planning.	They have high-level goals (desires) they want to achieve. They evaluate the most appropriate available plan (intentions) to reach those goals utilizing their beliefs.
Flexibility and Adaptability	Less flexible in handling complex and dynamic environments, and well-suited for tasks that require quick and efficient responses to specific stimuli.	More adaptable and flexible in dealing with complex, dynamic, and uncertain environments due to their ability to reason about beliefs and intentions to adjust their behaviour based on changing circumstances.
Complexity	Simple in design and implementation, as they map inputs to actions.	More complex to design and implement since they are used in applications requiring long-term planning and decision-making.

The agent-based digital twin mainly comprises two interconnected major worlds/regions, the cyber and the physical worlds. The first region is the `Digital Assets` layer powered by MAS and accommodates `DAs`. This layer encompasses two types of agents: virtual representatives of the `PAs`, which are situated inside `Digital Agents Organization`, and the second type, abstract agents that have a distinct function and capability, such as a `Reasoning Agent (RA)` for making context-aware decisions and inferring about various situations, a `Simulation Agent (SA)` for conducting simulations, a `Visualization Agent (VA)` for visualizing the obtained and processed information, and a `Learning agent (LA)` for improving the system through learning iterations from past experiences. The number of agents in the `Digital Assets` layer is relative to the requirements and complexity of the desired features in the DT of the system under study.

The second pivotal part in the architecture is the `Physical Assets` layer, which is also powered by MAS technology and includes the physical system and related components represented as `PAs` organized inside `Physical Agents`

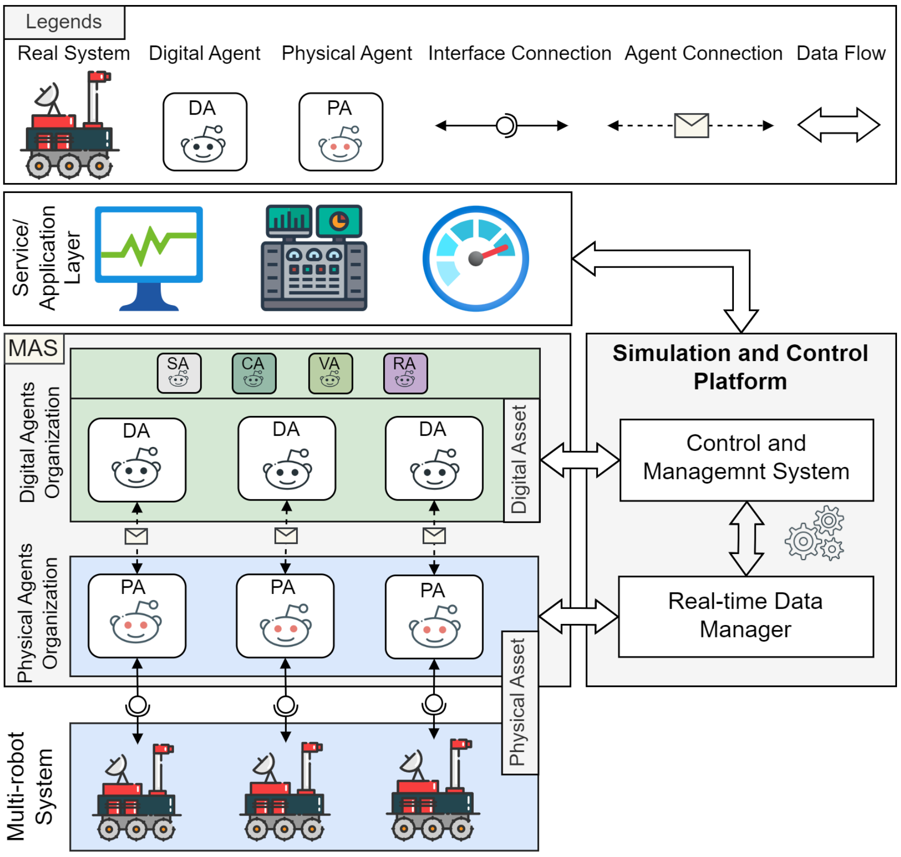


Fig. 5 A high-level overview of the MSF agent-based digital twin deployment for the MRS.

`Organization` to control and operate physical components and communicate with their corresponding `DAs` in the `Digital Agents Organization`.

The adopted architecture advocates following a hybrid architectural design, where a decentralized MAS is followed for designing `Physical Assets` and their digital representatives in the `Digital Assets`. This allows for avoiding bottlenecks or a single point of failure in the system, as the components are not tightly dependent on a central unit. This option can flexibly scale up the system and handle enormous numbers of agents and complex interactions.

In contrast, in the service and application layer of the MAS-integrated DTs, the centralized architectural choice was preferred to ensure effective collaboration and coordination between agents, manage possible conflicts, allocate resources efficiently, and provide some services that cannot be acquired from the individual agents. Accordingly, they can be exclusively obtained from the DTs' application layer, where a comprehensive system overview is maintained. As a factual example, obstacle detection in robot agents can be realized internally by their `PAs`. In contrast, collision avoidance between multiple robots should be managed and coordinated via a service where an overview of the entire system (the environment and the relevant robots) is available.

4.4 Simulation and Control Platform

In essence, the simulation and control platform for the target CPS (MRS) is designed and implemented based on the main concepts, entities, and relations introduced in the MSF of the agent-based digital twin and also according to the previously proposed general architecture. Agents are used to design and deploy the two layers; the `Digital Assets` and `Physical Assets` implicitly represent the two environments `Virtual Experimental Frame` and `Real World Experimental Frame` respectively, where the `DAs` and `PAs` operate.

The two experimental frames encompass the relevant elements of each. The `Digital Twin` deployed as a `DA` inside the `Digital Assets`. Moreover, the `Real System` incorporated within a `PA` in the `Physical Assets`. The `PA` and `DA` are designed by taking into account the specifications and the objectives defined by the actual system `Model` that preserves a `Simulation Relation` between the `PA` and `DA`, and it should guarantee that it correctly simulates the model and generates a valid output within the `Virtual Experimental Frame`, while, the `Modelling Relation` is precisely capturing the system behaviour just to the extent defined by the system's objectives. Agents allow binding the two worlds continually/continuously based on the characteristics and requirements of the twinning.

The rest of this sub-section discusses the technical details of each component and the technologies (starting from introducing JADE and concluding with the implemented agent reasoning models) utilized to deploy the simulation and control platform. Fig. 6 exemplifies the full implementation and deployment details. Besides agent technology, other technologies and tools are leveraged to enable and support executing tasks, such as simulation, visualization, and data processing.

4.4.1 JADE:

Specifically, to realize the implementation of the agent-based digital twin simulation and control platform, we have utilized one of the most widespread and well-known agent-oriented middle-ware called Java Agent DEvelopment framework (JADE[2]). JADE is an open-source platform that has been widely utilized in academic research and some industrial applications for building intelligent agent systems. It offers several features[7] for developing MASs:

1. Agent Interaction: It provides a robust communication infrastructure that enables agents to exchange messages, collaborate, negotiate, and coordinate their activities.
2. Agent Management: It offers tools and services for agent lifecycle management, such as agent creation, termination, migration, and monitoring.

[2] https://jade.tilab.com

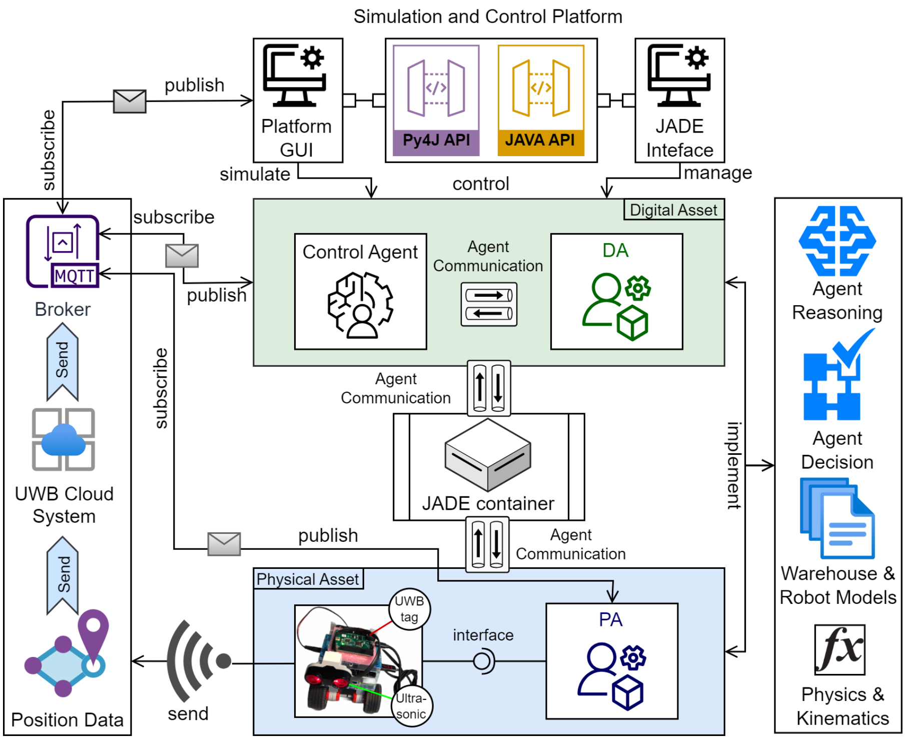


Fig. 6 A detailed structure of the implemented simulation and control platform.

3. Agent Behavior: It allows developers to define agent behaviours, determining how agents respond and execute their tasks.
4. FIPA Compliance: It adheres to FIPA standards, ensuring interoperability and compatibility with other FIPA-compliant MAS.
5. Agent Mobility: It supports agent mobility, enabling agents to move between platforms or devices during runtime.

In addition, it provides a structured framework that is composed of the main constituents: agents, their behaviours, and the environment where agents live, and they are called containers. This environment can boost the programming and deployment of MAS-based applications.

4.4.2 Communication:

In alignment with using the JADE platform. The Agent Communication Language (ACL) is JADE's main communication language. ACL is a standard for agent communication proposed by the Foundation for Intelligent Physical Agents (FIPA)[3]. This standard facilitates message exchange and interactions between agents. Following a specific setup, an agent with ACL-based defined information can establish a communication channel and exchange messages

[3]http://www.fipa.org

with the intended agent, which may be located at a separate machine or in a different geographical area. Therefore, various agents in our DT platform (i.e., `PAs` and `DAs`) can interact and send information. However, ACL messaging is not very efficient for high-volume real-time data communication. For this reason, another alternative option from the IoT technologies (MQTT) was utilized for this purpose.

4.4.3 UWB Technology:

UWB is a localization technology, and it was discussed in the background section. Fig. 6 shows how UWB technology is deployed in our platform. Essentially, UWB main anchors have been installed in a room to provide the required coverage to have an operational positioning system to mimic a warehouse environment. Then, another tag called the *master tag* is connected to a central server to collect the sensory data from each tag attached to any robot operating in the warehouse. Then, the localization data for all tags are retrieved from the cloud (i.e., UWB server). Afterwards, data is published through the MQTT broker, and the subscribed agents of the robots or other applications receive the position updates. As MQTT technology is highly dependable in IoT applications, it can provide reliable data exchange appropriate for the real-time positioning of moving robots.

4.4.4 Physical Asset:

Physical systems are usually composed of hardware and software parts. Thus, `Physical Assets` combine the physical components and their software implementations, which are represented as agents in our implementation. Every individual physical system (robot) has been designed and programmed with agents as a digital entity (`PA`) and then has been coupled and mapped into its digital counterpart (`DA`) as illustrated in Fig. 6.

`Physical Robot` is the operational and actual materialization, including different models (e.g., kinematics, architectural, and logical processes) of the actual robot system. Robots are basically implemented using LEGO MINDSTORMS EV3 and Raspberry Pi-powered BrickPi[4]. The ev3dev[5] operating system, which is based on Debian Linux, is utilized for programming and deploying `PAs'` code programmed in JADE on the robots. Besides, a UWB tag is mounted on the robot used for navigation and localization. The concrete robot prototype has been demonstrated in our previous study.[35]

The `PA` controls and instruments the CPS (robot) hardware and uses its different models to perform tasks as designed. The physical system consists of sensors that vary from one system to another (e.g., ultrasonic sensor, speed sensor, position sensor) and actuators (e.g., mechanical actuator). Outputs are generated from those sensors, resulting in environmental perceptions. In contrast, actuators manipulate, influence, and eventually affect the environment through their actions.

[4]https://www.dexterindustries.com/brickpi
[5]https://www.ev3dev.org/

The robot's components, including the sensors and actuators, are embodied inside a singular agent in our robot design. The role of the `PA` is to orchestrate all the processes, functions, and procedures of the robot so it receives specific input from its sensors and then issues commands and actions that control the actuators.

Ultimately, in the `Physical Assets`, the `PA` represents the physical system according to a particular perspective and properties of interest defined by the DT's requirements and objectives. In conformity with what we mentioned, Fig. 6 exemplifies the integration of the physical robot and its `PA` within the simulation and control platform.

4.4.5 Digital Asset:

This layer is critically important to provide functions in the simulation and control platform, such as agent reasoning and decision-making strategies. In essence, a DT requires virtual representations of `Physical Assets` in a virtual space with meaningful information flow between them. This means a digital instance is required for each physical component needed to be included in the DT.

Thus, DTs should represent the essential properties and processes of the physical components, such as sensors and actuators. The crucial aspect of this mapping process is ensuring that the virtual entities are kept in sync with their physical counterparts. By creating digital counterparts of `Physical Assets`, we can add new intelligent features to these physical entities in a modular manner. For example, a physical sensor in the physical layer only senses and sends data to its digital instance, which can perform more complex processes in the digital layer by taking advantage of the system's flexibility and computation capabilities.

The `DA` is located in the digital space, representing the `PA`, which in order represents the `Physical Assets` in the physical space as highlighted in Fig. 6. These agents operate in a digital environment (cloud) and communicate with their physical counterparts. In the introduced platform, the `DA` operates in the digital domain where most of the processing is done; thus, it plays the instructor role and guides the `PA` in different situations.

In this context, twinning the properties of the interest of the physical system components in the DT is a flexible and adjustable configuration that is up to the designer to decide the level of granularity of every twin (i.e., twinning every sensor, actuator, and sub-process in the DT or having a single twin that encompasses all these components and their sub-processes). In our approach, we have followed the peer-to-peer design. Thus, a single agent represents the entire robot system, including all its elements, features, and behaviours. This agent will have its twin in the `Digital Assets`. By doing so, the communication overhead between `DAs` and `DAs` is significantly decreased since not every individual component of the robot is represented as an agent. Above that, the complexity of designing and programming the robot can be straightforward, and scaling the system is more manageable and flexible.

An agent named `Control Agent (CA)` functions as an orchestrator and middleware that manages the other agents' operations. It operates in two main parts: the platform's graphical user interface (GUI) and the JADE container host. The GUI is connected to the Java JADE through Py4J to send GUI events to the `CA`, which, in accordance, sends commands to other agents, the `PAs` and `DAs`. Essentially, Py4J allows Python scripts executed within a Python interpreter to access the Java objects dynamically in a Java Virtual Machine. This enables Java methods to be invoked from Python applications. Using such a connection, we can send messages to the `PAs` and `DAs` in the JADE container from the Python side of the GUI through the `CA`. Furthermore, `CA` receives messages from the `DA` and passes them to the GUI.

4.4.6 Simulation and Control GUI:

The GUI is part of the presented simulation and control platform, as shown in Fig. 6. It was designed and developed using Python and the PyQt6 library.

Mainly, the main menu interface, as depicted in Fig. 7, is intended to enable the users of the agent-based digital twin of the MRS to perform several tasks: (1) create `DAs` for physical robots, (2) define the reasoning mechanism of each agent, (3) define paths, destinations, charging station in the environment, (4) run simulations and perform similarity measure calculations.

The three dots with distinct colours appearing in Fig. 7 represent the position of each robot, which is localized in real-time by a UWB tag and updated regularly. Different configurations can be defined in the agent menu, such as the reasoning type and unique identifier names of `PAs` and `DAs`.

4.4.7 Agent Reasoning Model:

The introduced approach takes advantage of the reasoning models provided in the DT to deliver more intelligent feedback and instructions to the actual system. In the following parts, we explain and clarify the deployment of the two reasoning mechanisms.

Reactive Reasoning Deployment. As mentioned before, JADE provides an environment for programming and deploying MASs. The behavioural model of JADE agents is based on a reactive model that makes agents act in a stimulus-response fashion. In this reasoning mechanism, the agent perceives the environment and surroundings and acts based on its programmed behaviours toward those perceptions. Therefore, the reactive model for `DA` is designed so that it reacts to messages sent from the `PA`, as illustrated in Fig. 3.

Examples of agent perceptions include a low-battery notification, mission-completed acknowledgement, or obstacle detection alert. These events change the agent's state in the environment and are communicated between JADE agents using ACL messages. In the situation when the physical robot runs out of battery, it sends a low-battery message to the `DA` indicating the robot's battery has reached a threshold level, so the `DA` acts according to the reactive reasoning mechanism and the predefined rules (e.g., pause the current mission and move immediately to the closest charging station, then continue).

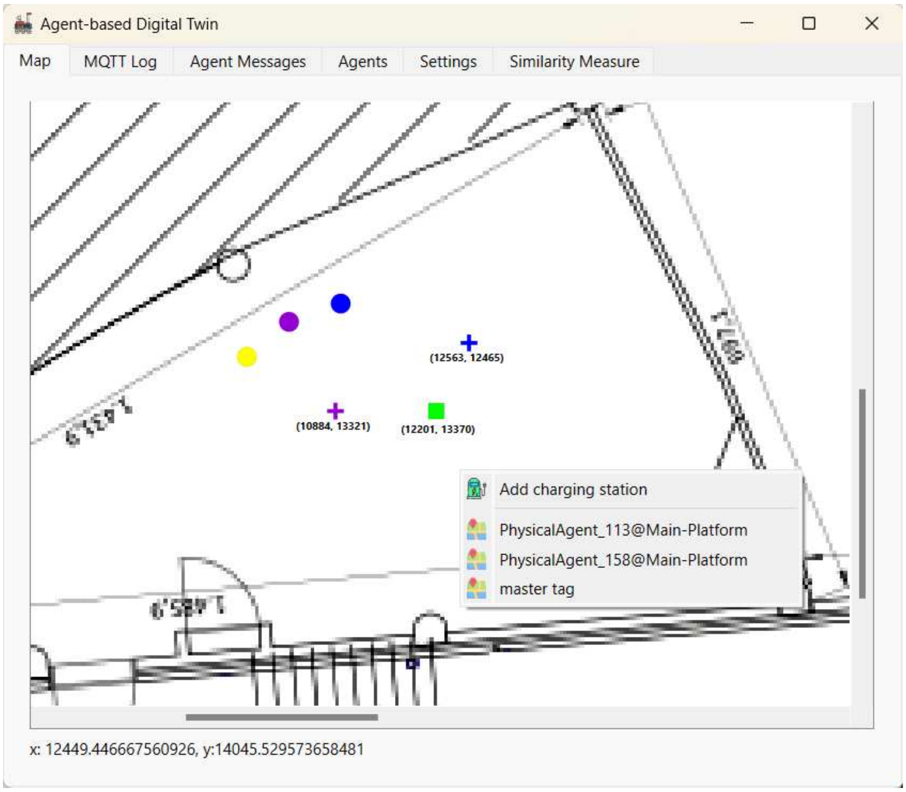


Fig. 7 The main interface menu in the simulation and control platform.

BDI Reasoning Deployment. The JADE platform does not support the BDI architecture. Therefore, to deploy BDI agents within JADE, a compatible implementation with JADE is required. Hence, we used a BDI extension named *BDI4JADE*[6]. It is provided as a layer on top of the JADE platform. This extension offers an infrastructure to implement agents using the JADE platform, utilizing the Java language to develop BDI-based agents.[45]

BDI agent consists of a dynamic beliefs base, a plan library, and a set of defined goals. Plans are composed of rules that define specific procedures that dictate when a plan should be executed to achieve a goal or update the agent's beliefs. Utilizing the BDI4JADE, features provided by JADE are reused and leveraged as much as possible to model the beliefs, plans, and goals of the BDI agents. Therefore, using the JADE platform and BDI4JADE, the BDI and Reactive reasoning mechanisms are designed and programmed in our platform.

As indicated, a `DA` implements two reasoning capabilities (BDI and Reactive). Any reasoning mechanism can be selected and specified for a particular `DA`. Hence, heterogeneous agents with different reasoning mechanisms can operate simultaneously in the environment.

[6]https://github.com/ingridnunes/bdi4jade

5 Case Study

To drill down further and gain a better understanding, we show a comprehensive case study to shed light on the features and capabilities of our approach and the realized platform. These details are elaborated on using an example of a smart logistics warehouse. Warehouse management demands employing intelligent and automated methods to reduce the load on workers and achieve efficiency, accuracy, and low cost. In this regard, the robots in MRS mimic a semi-real-world context in a warehouse. MRS is integrated with the simulation and control platform for creating DTs for the robots to provide efficient and intelligent reasoning methods to achieve their tasks.

In fact, achieving this requires considering several requirements and constraints, such as accuracy, time, and energy efficiency. Accordingly, our case study example demonstrates how the proposed approach is applied and utilized to develop the platform and finally provides results regarding the performance of the robots using the reasoning mechanisms offered in the platform.

In essence, we can have as many robots as we want in the case study, depending on the availability of the physical hardware and components. However, in the next two subsections, we explain how to configure DTs, define missions for robots, and set the environment. Later on, Section 6 provides a detailed examination of the conducted experiments in the case study.

5.1 Defining the Cases

In Fig. 7, the main menu interface of the simulation and control platform is viewed. This interface comprises several tabs. Every tap provides a function in the platform. **Map** tap shows the map of the warehouse (floorplan) used in our application, and this map can be changed and calibrated with the UWB anchors. **MQTT Log** tap lists all the messages retrieved from the UWB broker, which contain the information of the UWB tags. The **Agent** tap in the platform is used to create `DAs` and define the relevant reasoning capability.

The coloured points in Fig. 7 represent the active physical robots in the system (except the yellow one that represents the master tag, which serves as a hub to collect data for other tags). UWB tags continuously propagate the information (i.e., location) for every robot. A new agent window appears by clicking on a particular robot. In this agent menu, DTs (i.e., `DAs`) can be defined for every physical robot with a unique name *GUID* in the JADE container. From that menu, the type of agent **bdi** or **reactive** can be selected, and by doing so, it defines the reasoning mechanism of the `DA` and creates a DT (replication), particularly for that robot. The `DAs` and `PAs` have specific colours that can be changed during initiation or run-time to discriminate them from each other in the environment. The main activities and the use cases for using the simulation and control platform of the agent-based digital twins of the MRS are explicitly illustrated in Fig. 8.

The initialization of the MRS simulation and control platform, then the process of creating DTs for the robots in the replication process, are highlighted

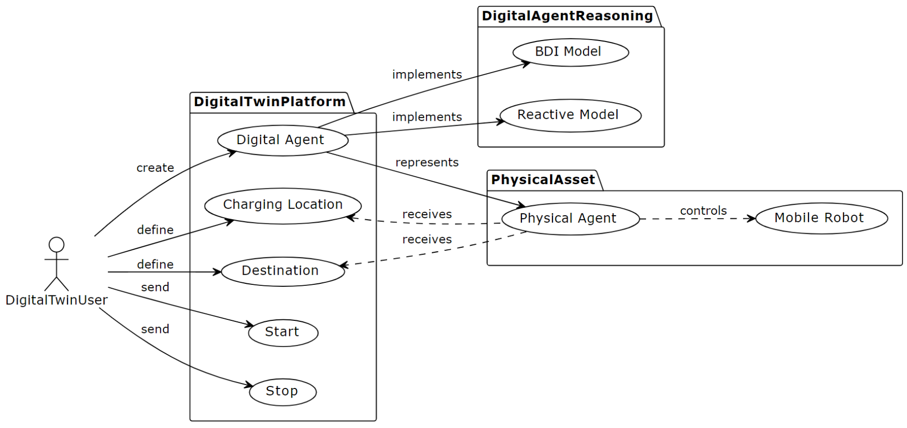


Fig. 8 The use case diagram of using the simulation and control platform for the MRS.

in Fig 9. The user selects the target robot to create a DT. After that, the platform executes the replication function that creates an up-to-date instance with all features of interest, which are cloned from the `PA` into the `DA`.

The procedures of defining and scheduling a mission that might include more than one destination, the place of charging stations, and executing these missions for a specific robot agent are depicted in the Fig 10 sequence diagram.

6 Experiments and Analysis

This section discusses the experiments conducted to emphasize the advantages of the proposed approach. Then, it evaluates the implemented simulation and control platform for the MRS and, more specifically, assesses the performance of the agents' reasoning mechanisms and monitors the difference between the physical and virtual instances.

The first experiment deployed two robots; one robot used reactive reasoning, while the other used BDI reasoning. An identical setup and conditions in the experiment environment are defined for both robots to compare the performance of their `DAs` based on particular criteria. In the second experiment, we compared a `DA` with a `PA` of an operational physical robot to observe and determine the simulation to reality gap.

Practically, we followed the processes explained in the previous section of defining a particular reasoning model for an agent (reactive or BDI), then we created `DAs` and assigned destinations and defined charging stations, and finally, we executed the mission.

It is important to emphasize that both experiments assume that a `PA` follows the behaviour of a `DA`. The latter is responsible for reasoning and making decisions that ultimately guide the `PA` to execute the mission. This assumption is made because robots represented as `PAs` tend to be affected by environmental conditions, especially if the robots are just prototypes used in a proof of concept. Consequently, many requirements, such as friction force, weight, and

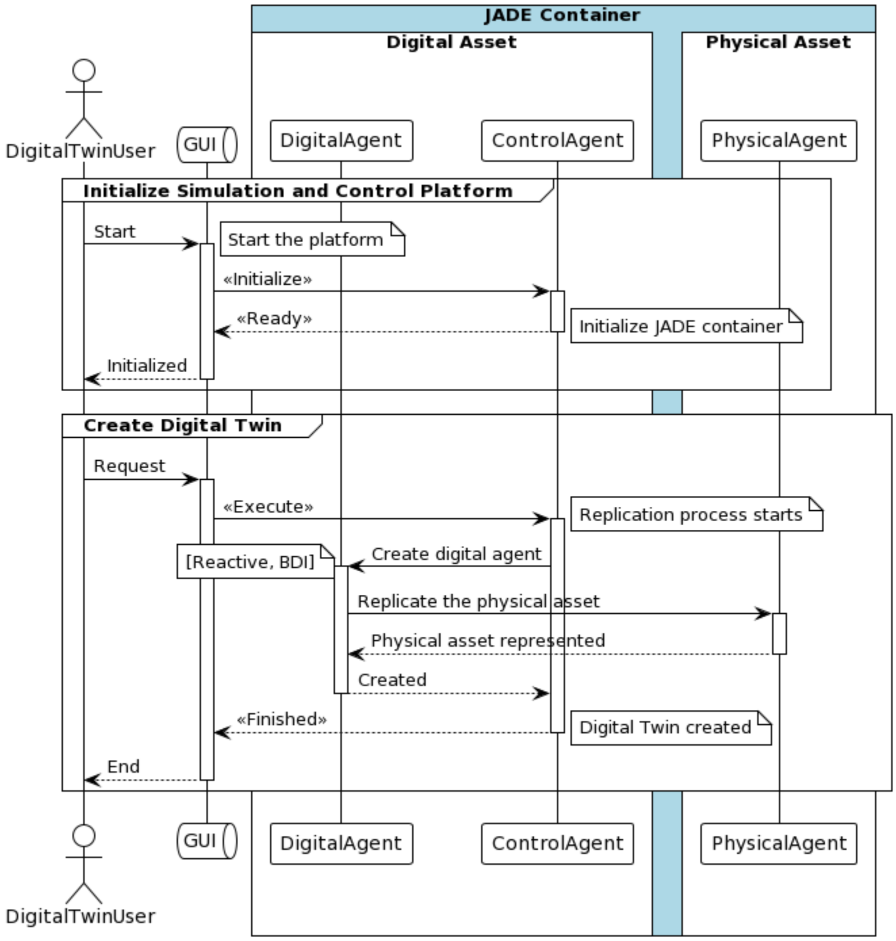


Fig. 9 The sequence diagram of creating digital twins (DAs).

other motion dynamics that can affect the robot's behaviour, are unsatisfied. In many cases, anomaly and jittering in UWB sensors, as we experienced, can make the robot's motion and behaviour abnormal.

Accordingly, in the first experiment, we considered comparing and analyzing the behaviour of `DAs` in the simulation and control environment, where the primary objective is to evaluate and assess their reasoning and internal behaviour. On the other hand, the second experiment monitors and observes how the `PA` behaves in contrast to the behaviour of its `DA`.

6.1 Reactive and Rational Agents Comparison

Four destinations and four charging stations have been defined in this experiment. Destinations are global because they are loaded from a file; therefore, all destinations are accessible to all active robots in the environment that already

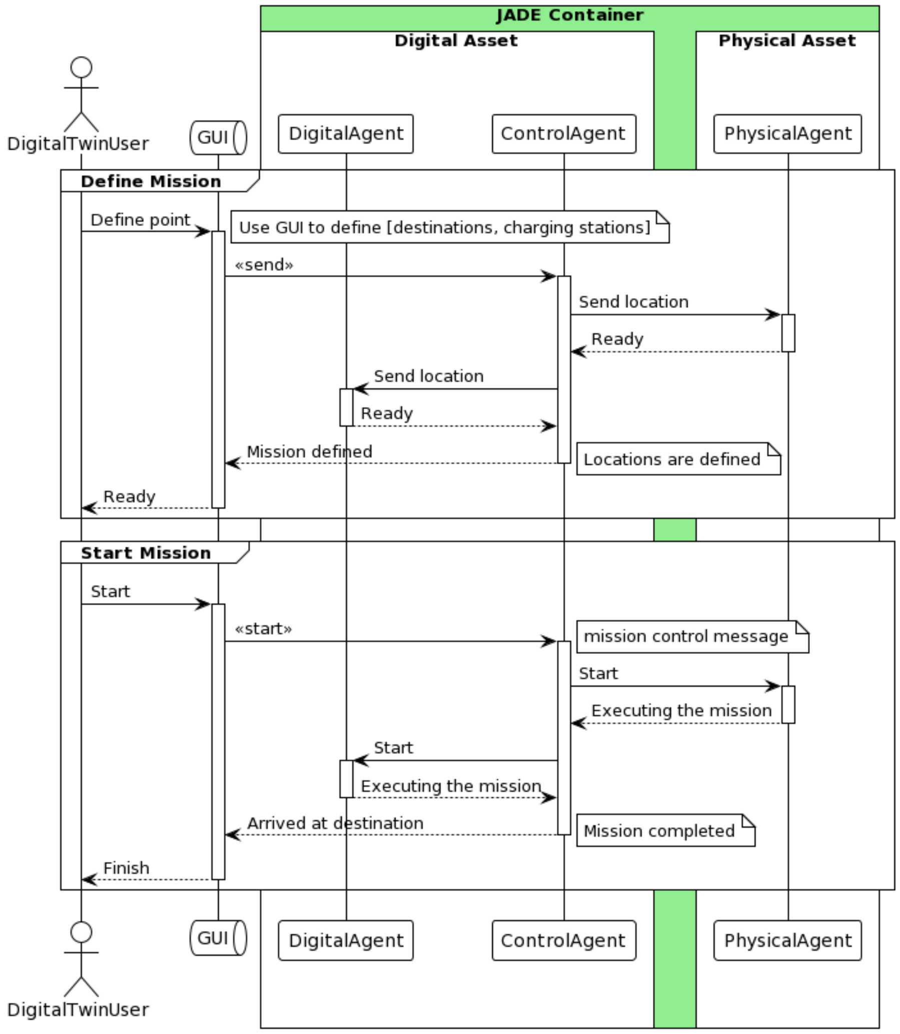


Fig. 10 The sequence diagram of defining and executing a mission.

have DAs. Above that, the user can still define exclusive destinations using the dynamic option. Nonetheless, charging stations are always global destinations. However, in this experiment, the scenario should be identical for both agents to evaluate and observe the similarities and differences between them.

This experiment has been conducted to compare the behaviour and performance of the two types of reasoning mechanisms supported by DAs (Reactive and Rational). As given before, the constraints and the scenario are precisely the same in the two agents. Both agents should execute the mission from the start point to the final destination. At a certain and exact point in the trajectory, a low-battery message was sent to the two agents to observe their behaviour and reaction.

6.1.1 Reactive Agent:

The reactive agent reacts to stimuli and takes action accordingly, as described in the context of DTs in section 4. Its operation mechanism is given in Fig. 3. Applying reactive reasoning in the considered case of MRS, the robot agent perceives the surroundings from its sensors, such as an ultrasonic sensor, to detect obstacles, a battery sensor to determine the available battery capacity and a UWB sensor to localize and identify its position in the environment to avoid collisions.

In this experiment, the focus is on the DT's reactive reasoning efficiency to (1) follow the most suitable path, (2) consume less energy, and (3) reduce the charging times as much as possible. In simple terms, the algorithm of the reactive agent monitors the state of the `PA`, then updates and communicates with the `DA`, which reasons in the digital environment and can communicate with other agents to get a broader context of the systems state, after that, it takes an action based on a set of predefined rules that affects the physical world via the `PA`.

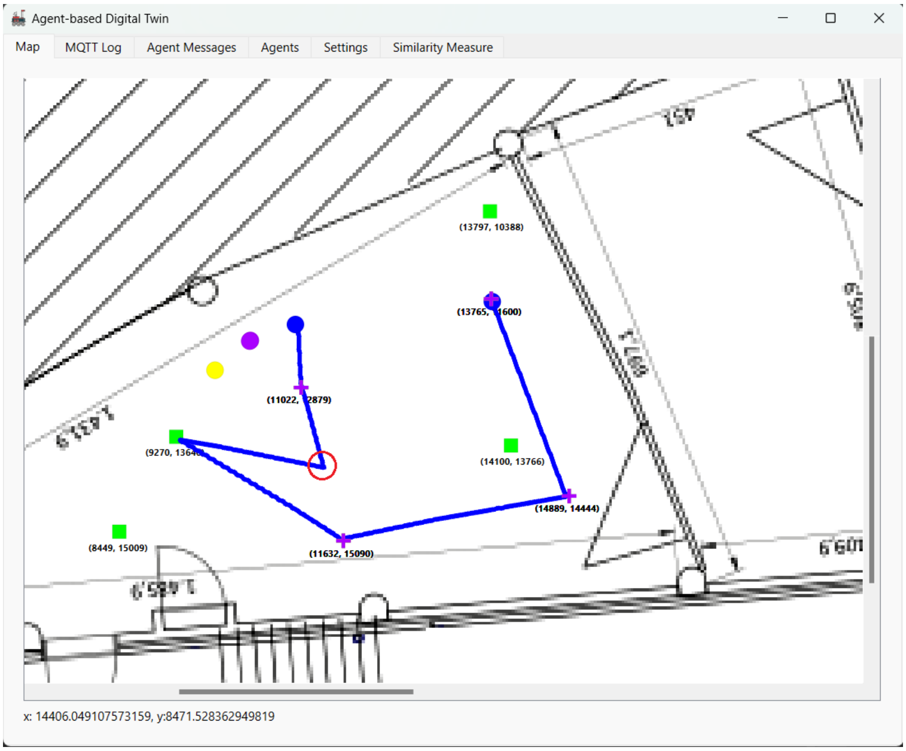
Fig. 11 shows a drawn solid blue line that originated from the blue dot. Basically, It is the trace of the reactive agent of that robot executing the specified mission. Clearly, the reactive agent visited all the defined destinations. First, it receives a list of all destinations and charging stations; then, it calculates the distance of all destinations and selects the one with the shortest path according to its current location determined by the UWB tag.

In reality, a low-battery message is received from the `PA` when the battery level is low. Immediately, the reactive agent reacts, pausing the current mission and then moving to the closest charging station to its current location. It is essential to mention that, in our implementation, messages from `PA` to `DA`, such as low-battery messages, are sent only if both are deployed in the JADE container. Still, to make debugging relatively easier, the simulation platform provides simulation messages such as a low-battery message that can be sent to the `DA`, which is identical to the actual message usually sent from the `PA`. So, according to the defined scenario, the red circle in Fig. 11 points to the instant when the reactive agent received a low-battery message that resulted in chanting its direction to the closest charging station.

6.1.2 Rational Agent:

The same procedures were followed to conduct the experiment with the BDI agent. The reasoning algorithm of the BDI is more complex than the reactive.

Conceptually, the reasoning cycle of the BDI agent is initiated once the `PA` is twinned into a `DA` with the BDI reasoning capability. Accordingly, the agent sets and updates the beliefs and desires (goals) based on the status of `PA`, while the plans are determined solely in the `DA`. The system's motivational state is embodied by goals, which are the desires the `PA` wants to achieve or reach, such as reaching a destination or going to a charging station in the case of insufficient battery level. Meanwhile, intentions represent the deliberative



x: 14406.049107573159, y:8471.528362949819

Fig. 11 An experiment showing a digital twin executing a mission as a reactive agent in the blue trajectory.

component of the system. When an agent has an intention, it chooses the most appropriate plan to attain that goal until it is reached, no longer desired, or tagged as unachievable. The plans in our implementation are a message plan, movement plan, and charge plan. On the other hand, beliefs capture environmental characteristics that are updated regularly after detecting changes in the robot's status. They can be viewed as the informative component of the whole system and the dynamic environment. In the BDI agent, beliefs such as the battery level and position information are updated within every robot.

However, the essence here is to show the primary reasoning and cogitation features of the BDI agent. Once the agent executes its reasoning, it moves towards the defined destinations, following the shortest path. As in the reactive agent scenario, a low-battery message is sent from the simulation platform at the exact same point.

The BDI agent visited all the assigned destinations. In contrast to the reactive agent, it continued its current mission after receiving a low-battery message. Still, the distinction is that the former had reasoned about the situation and chose to continue its mission and then go to a charging station, which is considered the *optimal* charging station along the followed path to the final destination. Fig. 12 elaborates on the difference between the two agents'

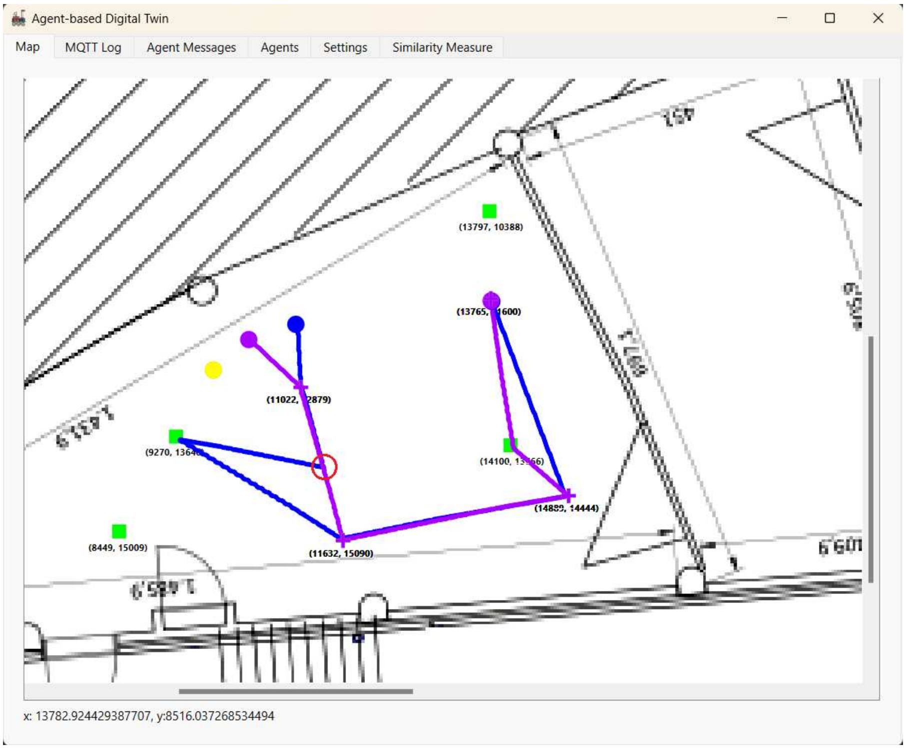


Fig. 12 An experiment showing a digital twin executing a mission as a BDI agent in the purple trajectory.

behaviour and executed paths. The figure shows the BDI agent as the purple dot drawing a purple line representing its trajectory while executing its mission, visiting all destinations, and ultimately arriving at the final one.

6.1.3 Evaluation and Results:

The efficiency and performance of the two agents are evaluated by calculating the elapsed time of the two agents during executing the same mission and the total cost of each agent's path from the start of the mission until the end.

Elapsed Time and the Total Path Cost. These values identify the more efficient and optimized agent as in Fig. 13 while executing the defined scenario. Straightforwardly, we compared the time needed for each agent to complete its mission. In addition, the total path cost is calculated during the agents' run-time. The total distance units needed to reach the final destination goal denotes the total path cost.

The graph in Fig. 14 contains two axes. The horizontal x-axis represents the measurement units of the elapsed time for the agent to reach the last destination. The vertical y-axis represents the distance measured in centimetres (cm) to reach every destination within the entire trajectory.

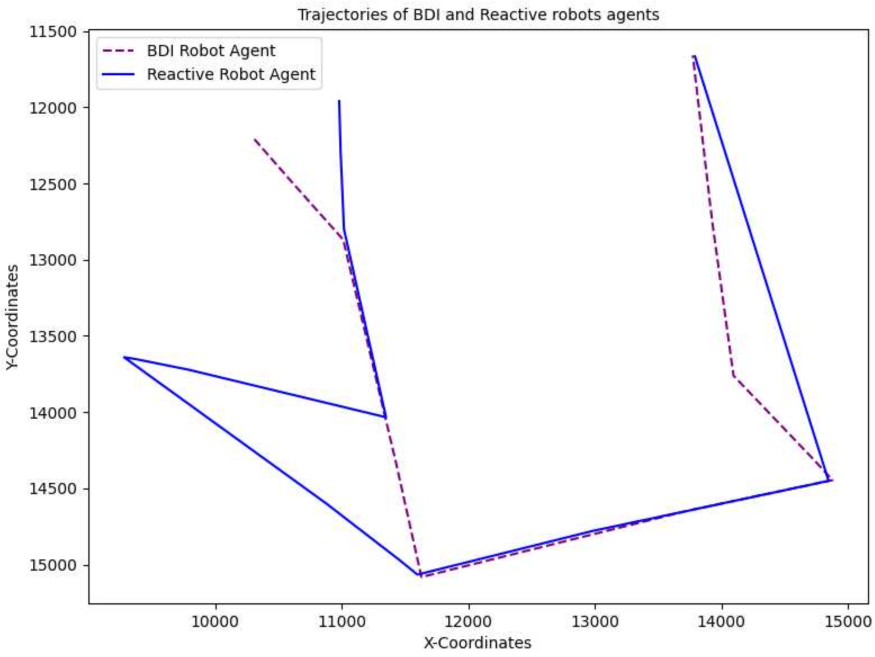


Fig. 13 Comparison of the total path length between the two trajectories of Reactive and BDI.

The graph contains sharp increases (peaks) and declines (valleys). Every valley point represents the agent's arrival at a particular destination along the path where the distance between the robot and that destination must be around zero on arrival. On the other hand, peaks describe the start of a new movement to a new destination, where the robot should travel the calculated distance of that destination, decreasing gradually until it reaches that goal. This is repeated until the robot reaches its final destination. From Fig. 14, we can observe the performance and efficiency of the two agents, where we can see that the Reactive agent has longer peaks than the comparison BDI agent, which has shorter peaks leading to travelling a shorter path and taking less time thanks to its efficient long-term reasoning algorithm.

Comparing the two agents while executing the same mission shows that BDI and reactive agents required about (44 seconds, 9900 distance units) and (57 seconds, 13380 distance units) respectively, to reach the final destination. These insights are significant as they demonstrate which agent is more optimized to execute this exact mission. This powerful approach enables the exploitation of DTs in a simulation environment connected to the real world, employing different decision-making processes to obtain the optimal solution for different scenarios. This experiment can be repeated in different complex situations to observe the more efficient agent that takes a well-planned route.

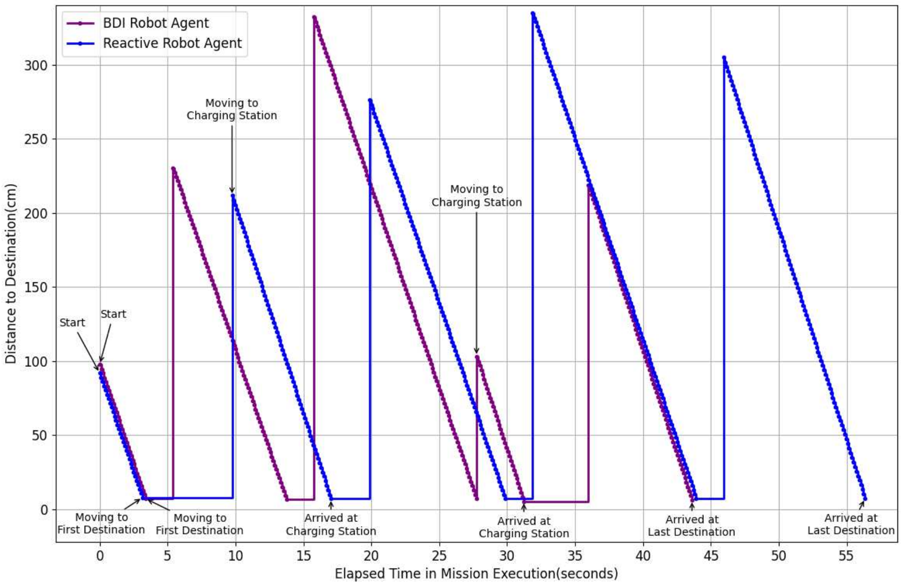


Fig. 14 Time comparison between Reactive and BDI agent.

6.2 Digital and Physical Agents Comparison

This experiment focuses on comparing a DA with the PA. It aims to provide a concrete overview of the simulation-to-reality gap. Therefore, the dissimilarity between the behaviour of either a DA and its corresponding PA is demonstrated.

6.2.1 Physical Agent:

Implementing a PA in the platform enables a deep examination of how the PA for a real system operates in a real environment constrained by laws of physics, environmental conditions, and uncertainties. Hence, in this experiment, a PA is deployed on the robot, which should receive instructions from its DA to execute the mission based on the selected decision-making model.

6.2.2 Digital Agent:

The DA operates in a more ideal environment than the PA as it functions in a simulation where most external factors and unpredictable behaviours are mitigated. The agent can utilize one of the available reasoning models to execute the defined mission.

The DA representation of the PA robot is specified as a reactive agent. Via the platform, a path containing five destinations is defined as a mission for this robot. The DA executes the mission by selecting the shortest path considering all destinations and sending the commands to the PA to do the same as depicted in Fig. 15.

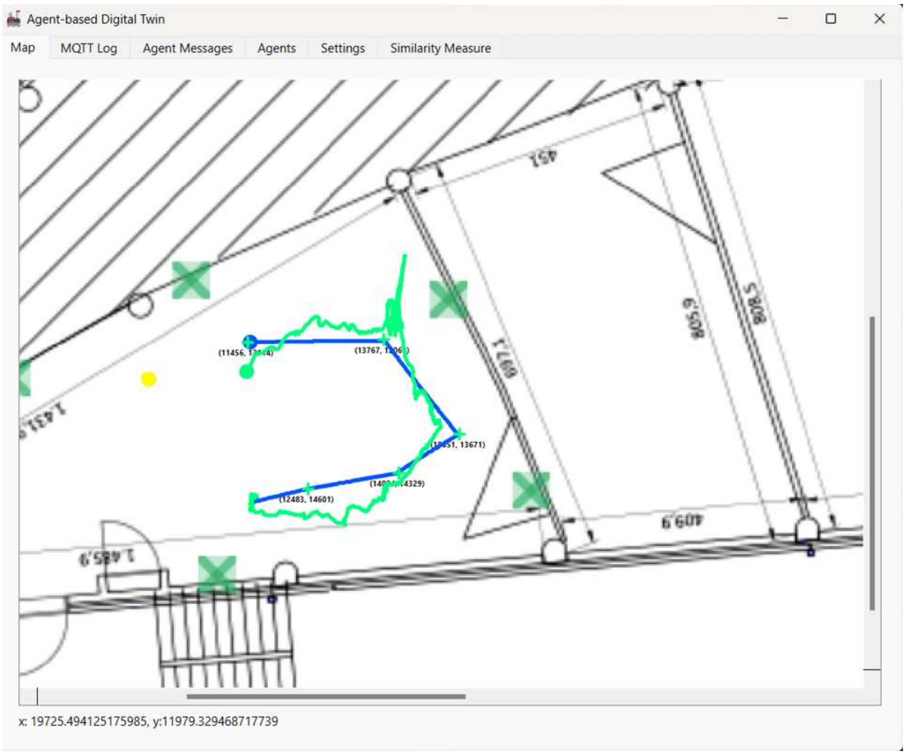


Fig. 15 An experiment showing the trajectories of the PA in green and the DA in blue.

6.2.3 Evaluation and Results:

In the case of the DA, the trajectory of this agent is explicitly drawn in the simulation environment by following the shortest path to accomplish the assigned mission. The DA solely mirrors the current status of the PA (i.e., its current position and battery level, etc.). However, during execution, it is detached from the actual environmental conditions as they exclusively affect the PA. As a consequence, the DA is not impacted directly by environmental dynamics and uncertainties, and it operates ideally in the simulated world, as elucidated in Fig. 15. On the other hand, once the PA (robot) receives the instruction from the DA, it starts executing the mission. However, the circumstances in a physical environment are often unexpected, and the behaviour of the robot is significantly influenced by physics, errors, malfunctions, instabilities, and the fluctuations that might occur in one of its components, such as sensors (UWB) or actuators (servo motor) and other parts (tires) as in our case.

Fig. 15 reveals a clear picture of the PA behaviour. Due to some anomalies in the positioning sensor readings and some slippery movements caused by the motors and robot's wheels with the environment floor (this is quite evident when the robot rotates at the fourth destination), the robot trajectory was rather oscillatory while moving and executing the mission.

DTs are utilized for several purposes. Utilization of this technology allows for performing multiple tasks, including monitoring, operating, anomaly detection, optimization, and improving the system under study, to mention a few. In this case study, DTs control, monitor, and provide intelligent reasoning capabilities for the actual systems (the robots). Thus, leveraging the platform's features, monitoring and detecting abnormal behaviour of the robots can be observed in this experiment. This gives information about the simulation-reality gap and performance of the actual robot. In addition, a similarity measure such as Fréchet Distance is used to identify this gap, providing feedback about the needed improvement.

Fréchet Distance. Measuring the similarity of trajectories is a crucial activity when analyzing moving objects. Measures such as Fréchet Distance are used to detect the variation and similarity between any two trajectories.

Given two sequences of poly-points in $\mathbb{R}^d$, where $p = \{p1, p2, p3...pn\}$ and $q = \{q1, q2, q3...qm\}$, then, the Fréchet distance, denoted by $f(x, y)$, is determined by finding the maximum value among the minimum distances between corresponding points pi and qi in two given trajectories p and q. Mathematically, the Fréchet distance [21] between two curves $f(x, y)$ is defined as:

$$max\Big(||p_{i(t)} - q_{j(t))}||, min\Big(f(x-1, y), f(x, y-1)\Big)\Big)$$

Intuitively, the parameter (t) can be taught as the instant of "time" when the snapshot of that point is captured from a time series.

Using Fréchet distance to measure the similarity or dissimilarity between two trajectories/curves has two main benefits: (1) Trajectory Analysis, which involves analyzing trajectories and comparing the traces of moving objects; (2) Path Planning, particularly in robotics, this can aid in path planning and motion planning. By comparing a planned trajectory with the actual executed trajectory, like in this experiment, the system can assess how well the planned path matches the real movement, which is crucial in robot navigation.

Accordingly, the Fréchet distance between the `PA` and `DA` is calculated, and both paths are recorded. As illustrated in Fig. 15, the actual path of the `PA` contains vividly and noticeable noisy data caused by the UWB sensor's instability. This noise was excluded while comparing both paths since it occurred just while the robot was rotating. The final clean version of the paths is provided in Fig. 16.

The results show that the Fréchet distance is about 2.889 m. In this case, the smaller is the better as the gap between the `PA` and `DA` becomes less. However, these results could definitely be improved by addressing the UWB sensors' jittering problem, considering surfaces with enough friction with the robot's wheels, and using robots that are more stable/reliable than LEGO MINDSTORMS. However, the ultimate goal of doing such experiments is to show the applicability and feasibility of the introduced framework and the developed and implemented platform to manage complex CPS such as MRS and provide services such as controlling, managing, and reasoning for these systems.

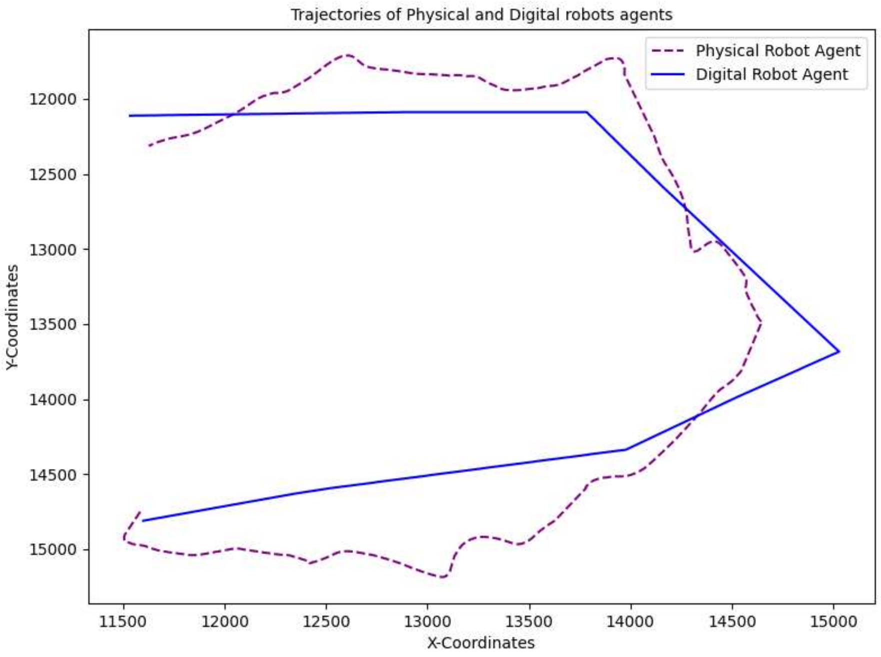


Fig. 16 Trajectory comparison between physical and digital agents.

7 Discussion

A DT is a technology that integrates physical and digital entities. Designing and building DTs for complex systems such as CPSs, which can be decentralized and distributed and can comprise a vast number of heterogeneous elements, makes this task very complicated. Therefore, a well-found framework that establishes and provides the essential tools for implementing a DT for such systems is needed. This framework should provide a flexible, modular, and adaptable architecture that enables the construction of a DT and allows the incorporation of advanced features to enhance the intelligence and robustness of the system.

To address and overcome these challenges, our study introduces a novel approach integrated with MSF that utilizes the capabilities, potential, and intelligence of the agent-based approach and MAS to deploy and build a simulation and control platform of a distributed and autonomous DT for MRS. The introduced platform is built and designed by considering the concepts, entities and relations of the Zeiglers' MSF, which are combined and blended into the context of agent-based and MAS-driven DTs. Specific reasoning techniques are modelled inside `DAs`; these decision-making capabilities can be assigned on the fly to the physical systems according to the situation or the condition in which the physical system will operate.

From the implementation point of view, a MAS toolchain called JADE and BDI4JADE plugin are utilized to develop and deploy the agent-oriented DTs for an intelligent MRS integrated inside a simulation and control platform and boosted by two versions of reasoning models (reactive and BDI). The platform is supported with simulation capabilities where `DAs` can be created and initiated with their physical counterparts and simultaneously can be monitored in the simulation and the real world.

The materialization of reasoning techniques in agents such as the reactive and BDI models provided the `DAs` with different but complementary and powerful reasoning capabilities. The two reasoning techniques operate with different cognitive computing mechanisms, where (1) BDI agents utilize their internal planning and reasoning cycle by considering parameters represented in beliefs and desires updated in the `PAs`, and intentions processed in the `DAs`, which enable them to make decisions in different situations by deploying different plans. Furthermore, the BDI agent reasoning approach provides flexible and adaptable components that respond to changes and operate efficiently in a dynamic environment. This flexibility is crucial in complex systems where events occur frequently and unpredictably. In the reactive reasoning approach, (2) the agent responds to environmental events in `PAs` quickly and directly without using a complicated reasoning cycle and planning. Reactive agents are designed to react to the current state of the environment and produce actions based on a predefined set of rules and behaviours in `DAs`. Reactive agents are often used when quick and straightforward responses are required and preferred, such as in robotics or sensitive real-time control systems. Their main advantages are speed and efficiency, as they can respond quickly to environmental changes without requiring intensive computation, planning, or deliberation.

A case study for a warehouse where an MRS is integrated into the simulation and control platform demonstrates the proposed approach's potential, applicability and capability. In addition, an experiment was conducted to observe and analyze the behaviour of the two `DAs` while another one was performed to identify the gap between the `PAs` and `DAs`.

The scenario in the first one shows a significant advantage of using a BDI agent over the reactive one, especially for long-term planning, where a situation is more complex, including many choices. Yet, in several situations, there is a need to use a more straightforward approach for taking action/s and decision/s immediately without delays. Therefore, both agents can be utilized hybridly according to the system requirements, constraints, environment conditions, and the level of reasoning required in CPS systems.

In the second experiment, the gap between the real world and the simulated environment is compared. The results show that this gap is relatively associated with the physical system's performance and is also affected by real-world physics. Thus, high-fidelity modelling of the physical system and the environment should always be considered; a physical system with high-quality components is more accurate and reliable. In our case, the low performance of some parts of the robots (fluctuation of UWB sensors' readings, motors' inaccurate rotations, slippery tires, etc.) increased this gap.

Although the two conducted experiments are relatively simple, their implication could be reflected in more complex systems and scenarios. Nevertheless, the main goal at this stage is to prove the applicability and showcase the promising results of the proposed approach and the developed platform. Later on, more experiments with more industrial relevance applications and use cases will be targeted.

The proposed platform supports two different reasoning methods for `DAs`, which can be initiated on the fly. Still, in many cases, they have to be redesigned and evolved as the requirements of the physical system are modified or extended. Agent technology and MAS are the key elements of the platform. Thus, enhancing the existing components by adding new features or adding new components to the system can be done distributively in a modular way. This would make providing more reasoning mechanisms to the platform relatively manageable and uncomplicated.

Overall, this work covers significant research area that addresses exploiting MAS tools and the agent paradigm to construct distributed, autonomous, and intelligent DTs for CPS designed according to M&S concepts. However, providing multiple and hybrid reasoning methods to these DTs has extended the applicability and flexibility of the introduced approach to support operating CPS (e.g., robots) in different situations and circumstances. The latter part has not been tackled in previous research studies. Hence, our work is leading the way in contributing to this direction.

8 Conclusion & Future Work

8.1 Conclusion

DT is a key technology in the digitalization headway. It has been exploited enormously in a wide range of applications and several contexts. The task of constructing dependable and sustainable DTs for CPS is intricate due to the complexity of mapping digital and physical components and providing autonomy, modularity, flexibility, and appropriate reasoning and decision-making capacity that makes DTs intelligent enough to provide deep insights and enough information to boost the performance and function of CPS.

In this regard, this paper discusses an approach for developing DTs enabled by agent paradigm and integrates different types of reasoning in DTs' agents. The pillars of the suggested approach are the proposed MSF combined with multi-reasoning mechanisms and extensible architecture for agent-based digital twins, which are utilized for designing and deploying a simulation and control platform for a mobile MRS case study.

Utilizing the capabilities of agents (autonomy, reactive, proactive, heterogeneity modelling, communication, etc.) provides an extra edge in developing flexible, modular, and scalable DTs. In addition, the reasoning techniques employed in the agent-based platform for MRS boost the performance of the robots' DTs to reason about simple cases, such as what-if analysis, or a multitude of complex scenarios within dynamic/uncertain environments such as a warehouse.

8.2 Future Work

Despite the practical and functional usage of the implemented platform for managing, controlling, and simulating CPS, such as robots through their DTs, and the substantial importance of including different reasoning behaviours in these twins that allowed the physical systems to have various attitudes and employ different decision-making strategies, there are still multiple tracks of improvements and research directions that should be investigated to enhance the proposed approach.

An example of these directions is the difficulty of managing high-frequency generated data from the physical system during operation and how it can be processed, analyzed, and fed to `DAs` to support the planning and reasoning cycle. In addition, anomalies and inconsistencies in this data can result in undesirable behaviour in the physical system, as the DTs are incapable of making the correct decisions. To tackle these implications, some suggested solutions aim to integrate dependable mechanisms such as time-series technologies in agents for handling big data and using complex-event processing methods to detect deviations.

Although using agents in DT modelling has considerable advantages, most agent-based and multi-agent system programming languages and frameworks are not trivial. Using them efficiently requires effort, time, and adequate programming skills. Besides, different DT implementations are almost entirely implemented following an ad-hoc approach.

For this reason, we plan to incorporate model-driven engineering concepts and techniques to add more services such as monitoring[59], simplifying, and expediting the process of building and updating the design models[38] of DTs. This would provide a layer of abstraction that facilitates the creation of multi-purpose models that can be modified and used for other DT implementations that target different systems and domains. Doing such will reduce the gap between high-level design and actual code implementation. So, different reasoning methods can be defined in high-level models, deployed in the target systems, and also might be reused in other systems.

Learning from previous experiences can potentially enable the development of intelligent and adaptive systems. Therefore, we aim to exploit multi-agent reinforcement learning (MARL). In MARL, each agent learns to make decisions based on the feedback received from the environment and other agents; such technology can make the system more intelligent and adaptive.

References

[1] Sailesh Abburu, Arne J Berre, Michael Jacoby, Dumitru Roman, Ljiljana Stojanovic, and Nenad Stojanovic. Cognitwin–hybrid and cognitive

digital twins for the process industry. In *2020 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, pages 1–8. IEEE, 2020.

[2] Rashi Agarwal, Supriya Khaitan, and Shashank Sahu. Intelligent agents. *Distributed Artificial Intelligence: A Modern Approach*, page 19, 2020.

[3] Mohammad Abdullah Al Faruque, Deepan Muthirayan, Shih-Yuan Yu, and Pramod P Khargonekar. Cognitive digital twin for manufacturing systems. In *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 440–445. IEEE, 2021.

[4] Kazi Masudul Alam and Abdulmotaleb El Saddik. C2ps: A digital twin architecture reference model for the cloud-based cyber-physical systems. *IEEE access*, 5:2050–2062, 2017.

[5] Francesco Alzetta and Paolo Giorgini. Towards a real-time bdi model for ros 2. In *WOA*, pages 1–7, 2019.

[6] Tomas Ambra and Cathy MacHaris. Agent-based digital twins (abm-dt) in synchromodal transport and logistics: The fusion of virtual and pysical spaces. *Proceedings - Winter Simulation Conference*, 2020-December:159–169, 12 2020. ISSN 08917736. doi: 10.1109/WSC48552.2020.9383955.

[7] Fabio Luigi Bellifemine, Giovanni Caire, and Dominic Greenwood. *Developing multi-agent systems with JADE*, volume 7. John Wiley & Sons, USA, 2007.

[8] Alexey A Bobtsov, Anton A Pyrkin, Sergey A Kolyubin, Sergey V Shavetov, Sergey A Chepinskiy, Yuriy A Kapitanyuk, Alexander A Kapitonov, Vladimir M Bardov, Anton V Titov, and Maxim O Surov. Using of lego mindstorms nxt technology for teaching of basics of adaptive control theory. *IFAC Proceedings Volumes*, 44(1):9818–9823, 2011.

[9] M. Braglia, R. Gabbrielli, M. Frosolini, L. Marrazzini, and L. Padellini. Using rfid technology and discrete-events, agent-based simulation tools to build digital-twins of large warehouses. *2019 IEEE International Conference on RFID Technology and Applications, RFID-TA 2019*, pages 464–469, 9 2019. doi: 10.1109/RFID-TA.2019.8892254.

[10] Michael Bratman. *Intention, Plans, and Practical Reason.* Cambridge, MA: Harvard University Press, Cambridge, 1987.

[11] Thomas Clemen, Nima Ahmady-Moghaddam, Ulfia A. Lenfers, Florian Ocker, Daniel Osterholz, Jonathan Ströbele, and Daniel Glake. Multi-agent systems and digital twins for smarter cities. In *Proceedings of the*

2021 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation, pages 45–55. Association for Computing Machinery, Inc, 5 2021. ISBN 9781450382960. doi: 10.1145/3437959.3459254.

[12] Philip R Cohen, Hector J Levesque, et al. *Rational interaction as the basis for communication.* CSLI Stanford, 1987.

[13] Louise A Dennis, Jonathan M Aitken, Joe Collenette, Elisa Cucco, Maryam Kamali, Owen McAree, Affan Shaukat, Katie Atkinson, Yang Gao, Sandor M Veres, et al. Agent-based autonomous systems and abstraction engines: Theory meets practice. In *Towards Autonomous Robotic Systems: 17th Annual Conference, TAROS 2016, Sheffield, UK, June 26–July 1, 2016, Proceedings 17*, pages 75–86. Springer, 2016.

[14] Pavlos Eirinakis, Kostas Kalaboukas, Stavros Lounis, Ioannis Mourtos, Jože M Rožanec, Nenad Stojanovic, and Georgios Zois. Enhancing cognition for digital twins. In *2020 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, pages 1–7. IEEE, 2020.

[15] John Ahmet Erkoyuncu, Maryam Farsi, Dedy Ariansyah, et al. An intelligent agent-based architecture for resilient digital twins in manufacturing. *CIRP annals*, 70(1):349–352, 2021.

[16] Jacques Ferber and Alexis Drogoul. Using reactive multi-agent systems in simulation and problem solving. *Distributed artificial intelligence: Theory and praxis*, 5:53–80, 1992.

[17] Alfonso González-Briones, Fernando De La Prieta, Mohd Saberi Mohamad, Sigeru Omatu, and Juan M Corchado. Multi-agent systems applications in energy optimization problems: A state-of-the-art review. *Energies*, 11(8):1928, 2018.

[18] Michael Grieves. Digital twin: manufacturing excellence through virtual factory replication. *White paper*, 1:1–7, 2014.

[19] Michael Grieves and John Vickers. Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems. In *Transdisciplinary perspectives on complex systems*, pages 85–113. Springer, 2017.

[20] Khadijah M Hanga and Yevgeniya Kovalchuk. Machine learning and multi-agent systems in oil and gas industry applications: A survey. *Computer Science Review*, 34:100191, 2019.

[21] Sariel Har-Peled. Fréchet distance: How to walk your dog. *Geometric Approximation Algorithms*, 2011.

[22] Weifei Hu, Tongzhou Zhang, Xiaoyu Deng, Zhenyu Liu, and Jianrong Tan. Digital twin: a state-of-the-art review of its enabling technologies, applications and challenges. *Journal of Intelligent Manufacturing and Special Equipment*, 2:1–34, 8 2021. ISSN 2633-660X. doi: 10.1108/JIMSE-12-2020-010.

[23] David Jones, Chris Snider, Aydin Nassehi, Jason Yon, and Ben Hicks. Characterising the digital twin: A systematic literature review. *CIRP Journal of Manufacturing Science and Technology*, 29:36–52, 5 2020. ISSN 1755-5817. doi: 10.1016/J.CIRPJ.2020.02.002.

[24] Stamatis Karnouskos and Paulo Leitao. Key contributing factors to the acceptance of agents in industrial environments. *IEEE Transactions on Industrial Informatics*, 13(2):696–703, 2016.

[25] Stamatis Karnouskos, Paulo Leitao, Luis Ribeiro, and Armando Walter Colombo. Industrial agents as a key enabler for realizing industrial cyber-physical systems: Multiagent systems entering industry 4.0. *IEEE Industrial Electronics Magazine*, 14(3):18–32, 2020.

[26] Manal Khayyat and Anjali Awasthi. An intelligent multi-agent based model for collaborative logistics systems. *Transportation research procedia*, 12:325–338, 2016.

[27] Christina Latsou, Maryam Farsi, John Ahmet Erkoyuncu, and Geoffrey Morris. Digital twin integration in multi-agent cyber physical manufacturing systems. *IFAC-PapersOnLine*, 54(1):811–816, 2021.

[28] Jay Lee, Behrad Bagheri, and Hung-An Kao. A cyber-physical systems architecture for industry 4.0-based manufacturing systems. *Manufacturing letters*, 3:18–23, 2015.

[29] Jay Lee, Moslem Azamfar, Jaskaran Singh, and Shahin Siahpour. Integration of digital twin and deep learning in cyber-physical systems: towards smart manufacturing. *IET Collaborative Intelligent Manufacturing*, 2(1): 34–36, 2020.

[30] Jin-Shyan Lee, Yu-Wei Su, and Chung-Chou Shen. A comparative study of wireless protocols: Bluetooth, uwb, zigbee, and wi-fi. In *IECON 2007 - 33rd Annual Conference of the IEEE Industrial Electronics Society*, pages 46–51, 2007. doi: 10.1109/IECON.2007.4460126.

[31] Paulo Leitão, Vladimír Mařík, and Pavel Vrba. Past, present, and future of industrial agent applications. *IEEE Transactions on Industrial Informatics*, 9(4):2360–2372, 2012.

[32] Paulo Leitao, Stamatis Karnouskos, Luis Ribeiro, Jay Lee, Thomas Strasser, and Armando W Colombo. Smart agents in industrial cyber–physical systems. *Proceedings of the IEEE*, 104(5):1086–1101, 2016.

[33] Paulo Leitão and Stamatis Karnouskos. A survey on factors that impact industrial agent acceptance. *Industrial Agents: Emerging Applications of Software Agents in Industry*, pages 401–429, 1 2015. doi: 10.1016/B978-0-12-800341-1.00022-X.

[34] Zheng Ma, Mette Jessen Schultz, Kristoffer Christensen, Magnus Værbak, Yves Demazeau, and Bo Nørregaard Jørgensen. The application of ontologies in multi-agent systems in the energy sector: A scoping review. *Energies*, 12(16):3200, 2019.

[35] Hussein Marah and Moharram Challenger. Intelligent agents and multi agent systems for modeling smart digital twins. In *10th International Workshop on Engineering Multi-Agent Systems (EMAS), 9-10 May 2022, Auckland, New Zealand, pp. 1–18.*, pages 1–18, 2022.

[36] Hussein Marah and Moharram Challenger. Madtwin: a framework for multi-agent digital twin development: smart warehouse case study. *Annals of Mathematics and Artificial Intelligence 2023*, pages 1–31, 7 2023. ISSN 1573-7470. doi: 10.1007/S10472-023-09872-Z.

[37] Hussein Marah and Moharram Challenger. An architecture for intelligent agent-based digital twin for cyber-physical systems. In *Digital Twin Driven Intelligent Systems and Emerging Metaverse*, pages 65–99. Springer, 2023.

[38] Hussein Marah, Geylani Kardas, and Moharram Challenger. Model-driven round-trip engineering for tinyos-based wsn applications. *Journal of Computer Languages*, 65:101051, 2021. ISSN 2590-1184. doi: https://doi.org/10.1016/j.cola.2021.101051.

[39] Wei Meng, Yuanlin Yang, Jiayao Zang, Hongyi Li, and Renquan Lu. Dtuav: a novel cloud–based digital twin system for unmanned aerial vehicles. *Simulation*, 99(1):69–87, 2023.

[40] Fabien Michel, Jacques Ferber, and Alexis Drogoul. Multi-agent systems and simulation: A survey from the agent commu-nity's perspective. In *Multi-Agent Systems*, pages 17–66. CRC Press, 2018.

[41] Jörg P Müller and Klaus Fischer. Application impact of multi-agent systems and technologies: A survey. *Agent-Oriented Software Engineering: Reflections on Architectures, Methodologies, Languages, and Frameworks*, pages 27–53, 2014.

[42] Elisa Negri, Luca Fumagalli, and Marco Macchi. A review of the roles of digital twin in cps-based production systems. *Procedia Manufacturing*, 11: 939–948, 1 2017. ISSN 2351-9789. doi: 10.1016/J.PROMFG.2017.07.198.

[43] Muaz Niazi and Amir Hussain. Agent-based computing from multi-agent systems to agent-based models: a visual survey. *Scientometrics*, 89(2): 479–499, 2011.

[44] Stuart Nolan. Physical metaphorical modelling with lego as a technology for collaborative personalised learning. In *Technology-supported environments for personalized learning: Methods and case studies*, pages 364–385. IGI Global, 2010.

[45] Ingrid Nunes, Carlos JP De Lucena, and Michael Luck. Bdi4jade: a bdi layer on top of jade. In *Ninth International Workshop on Programming Multi-Agent Systems:(ProMAS 2011), Taipei, Taiwan*, pages 88–103, 2011.

[46] Felix Ocker, Chris Urban, Birgit Vogel-Heuser, and Christian Diedrich. Leveraging the asset administration shell for agent-based production systems. *IFAC-PapersOnLine*, 54(1):837–844, 2021.

[47] Chidiebere Onyedinma, Patrick Gavigan, and Babak Esfandiari. Toward campus mail delivery using bdi. *Journal of Sensor and Actuator Networks*, 9(4):56, 2020.

[48] Jun Ota. Multi-agent robot systems as distributed autonomous systems. *Advanced engineering informatics*, 20(1):59–70, 2006.

[49] Diego GS Pivoto, Luiz FF de Almeida, Rodrigo da Rosa Righi, Joel JPC Rodrigues, Alexandre Baratella Lugli, and Antonio M Alberti. Cyber-physical systems architectures for industrial internet of things applications in industry 4.0: A literature review. *Journal of manufacturing systems*, 58:176–192, 2021.

[50] Terrin Pulikottil, Luis Alberto Estrada-Jimenez, Hamood Ur Rehman, Jose Barata, Sanaz Nikghadam-Hojjati, and Leszek Zarzycki. Multi-agent based manufacturing: current trends and challenges. In *2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–7. IEEE, 2021.

[51] Qinglin Qi, Dongming Zhao, T Warren Liao, and Fei Tao. Modeling of cyber-physical systems and digital twin based on edge computing, fog computing and cloud computing towards smart manufacturing. In *International Manufacturing Science and Engineering Conference*, volume 51357, page V001T05A018. American Society of Mechanical Engineers, 2018.

[52] Anand S Rao, Michael P Georgeff, et al. Bdi agents: from theory to practice. In *Icmas*, volume 95, pages 312–319, 1995.

[53] Yara Rizk, Mariette Awad, and Edward W Tunstel. Cooperative heterogeneous multi-robot systems: A survey. *ACM Computing Surveys (CSUR)*, 52(2):1–31, 2019.

[54] Robin Roche, Benjamin Blunier, Abdellatif Miraoui, Vincent Hilaire, and Abder Koukam. Multi-agent systems for grid energy management: A short review. In *IECON 2010-36th Annual Conference on IEEE Industrial Electronics Society*, pages 3341–3346. IEEE, 2010.

[55] Teodora Sanislav, Sherali Zeadally, and George Dan Mois. A cloud-integrated, multilayered, agent-based cyber-physical system architecture. *Computer*, 50(4):27–37, 2017.

[56] Concetta Semeraro, Mario Lezoche, Hervé Panetto, and Michele Dassisti. Digital twin paradigm: A systematic literature review. *Computers in Industry*, 130:103469, 9 2021. ISSN 0166-3615. doi: 10.1016/J.COMPIND.2021.103469.

[57] Angel Soriano, Enrique J. Bernabeu, Angel Valera, and Marina Vallés. Multi-agent systems platform for mobile robots collision avoidance. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7879 LNAI: 320–323, 2013. ISSN 16113349. doi: 10.1007/978-3-642-38073-0_37.

[58] Okan Topçu. Adaptive decision making in agent-based simulation. *Simulation*, 90(7):815–832, 2014.

[59] Michael Vierhauser, Hussein Marah, Antonio Garmendia, Jane Cleland-Huang, and Manuel Wimmer. Towards a model-integrated runtime monitoring infrastructure for cyber-physical systems. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, pages 96–100, 2021. doi: 10.1109/ICSE-NIER52604.2021.00028.

[60] Michael Wooldridge. Intelligent agents. *Multiagent systems: A modern approach to distributed artificial intelligence*, 1:27–73, 1999.

[61] Jiaju Wu, Yonghui Yang, X. U.N. Cheng, Hongfu Zuo, and Zheng Cheng. The development of digital twin technology review. *Proceedings - 2020 Chinese Automation Congress, CAC 2020*, pages 4901–4906, 11 2020. doi: 10.1109/CAC51589.2020.9327756.

[62] Jing Xie and Chen-Ching Liu. Multi-agent systems and their applications. *Journal of International Council on Electrical Engineering*, 7(1):188–197,

2017.

[63] Zakaria Yahouni, Asma Ladj, Farouk Belkadi, Oussama Meski, and Mathieu Ritou. A smart reporting framework as an application of multi-agent system in machining industry. *International Journal of Computer Integrated Manufacturing*, 34(5):470–486, 2021.

[64] Bernard P Zeigler, Herbert Praehofer, and Tag Gon Kim. *Theory of modeling and simulation.* Academic press, 2000.

[65] Xiaochen Zheng, Foivos Psarommatis, Pierluigi Petrali, Claudio Turrin, Jinzhi Lu, and Dimitris Kiritsis. A quality-oriented digital twin modelling method for manufacturing processes based on a multi-agent architecture. *Procedia Manufacturing*, 51:309–315, 2020.

[66] Xueyan Zong, Yan Luan, Hongliang Wang, and Shu Li. A multi-robot monitoring system based on digital twin. *Procedia Computer Science*, 183:94–99, 2021.

Author biographies

Hussein Marah is currently pursuing a PhD at the Department of Computer Science at the University of Antwerp. His research interests include the modelling and simulation of intelligent complex systems, Agent-based Modelling, Cyber-physical Systems, Internet of Things, and Robotics.

Moharram Challenger received his PhD in IT from the International Computer Institute at Ege University (Turkey) in Feb 2016. From 2010 to 2013, he was a researcher and team leader of a bilateral project between Slovenia and Turkey (TUBITAK). From 2012 to 2016, he was the R&D director of UNIT IT Ltd, leading one national project funded by TUBITAK and two European projects. He was also an external post-doc researcher at the IT group of the Wageningen University of Research from 2016 to 2017. In 2017 and 2018, he has been a faculty member as an assistant professor at Ege University. From Jan 2019 to July 2020, he was a post-doc researcher at the University of Antwerp, working on Flanders Make projects. He is currently a tenure-track assistant professor in the Department of Computer Science at the University of Antwerp. His research interests include domain-specific modelling languages, multi-agent systems, Cyber-physical Systems, and the Internet of Things.