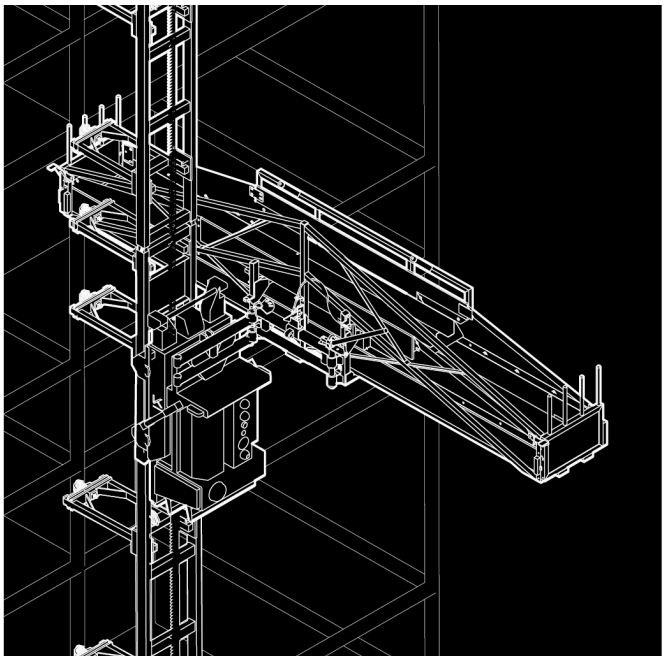
TECHNISCHE HOCHSCHULE
OSTWESTFALEN-LIPPE
UNIVERSITY OF
APPLIED SCIENCES
AND ARTS

KEWAZO



Digital twin for on-site construction robot:

from 3D real time monitoring to performance prediction

Master Thesis

Muny-Roth Chev

Master of Engineering in Integrated Design
Specialization Computational Design

TH-OWL supervisor :	Prof. Dipl. Ing. Jens-Uwe Schulz
KEWAZO supervisors :	Dipl.-ing. Uy Ha Dipl.-Ing. Arch. Alexander Liu Cheng

Detmold, June 2023

STATUTORY DECLARATION

I hereby declare that I have authored this master thesis «Digital twin for on-site construction robot: from 3D real time monitoring to performance prediction» independently, that I have not used there than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Muny-Roth Chev

Detmold, 20.06.2023

Abstract

In the field of construction work, human cognition plays a significant role in task planning. While robots are capable of performing repetitive tasks with precise motion control and carrying heavy loads, the unstructured and complex nature of the construction field hinders us from fully leveraging their potential for on-site applications. To address this issue, monitoring on-site robots emerges as a viable approach. A highly effective solution lies in the creation of a Digital twin, which can be defined as a realistic digital representation of assets, processes, or systems in the built or natural environment. It involves synchronizing data from the physical realm to the digital realm, enabling communication between the physical and virtual spaces. Digital twin technology presents a novel approach in the architecture, engineering, and construction (AEC) sector. While it can be used as a visual representation of buildings, commonly known as BIM (Building Information Modeling), new advancements have expanded its scope by incorporating real-time data, statistical and probabilistic models, and even artificial intelligence techniques to enable predictive capabilities. Furthermore, it can also incorporate virtual reality and augmented reality, enhancing the overall experience and functionality.

Although prototypes of such digital twins are developed in an academic context, very few examples can be found in an industrial context. There is a lack of standardization in the process as each robot is different and may require a completely different workflow to achieve a solid prototype. This thesis explores how an adaptable process digital twin can be developed and implemented in the construction industry, considering the diverse range of possible workflows and the current limited adoption of process digital twins in real-world applications. The first objective of the thesis was to gather a comprehensive step-by-step process, showcasing all the different possibilities one might consider in the development of a twin. This includes the choice of the software platform, data management, workflow, and extrapolation for future performance. Finally, the thesis focuses on the development of a digital twin for a specific case study, namely the robot LIFTBOT developed by the firm KEWAZO, with the aim of providing a tangible solution.

Keywords: Digital twin, on-site robotic construction, Real-time monitoring, data management, Unity, SQlite, performance prediction.

Acknowledgement

I would like to express my sincere gratitude to my thesis supervisor, Uy Ha, at KEWAZO for his thorough supervision. His door was always open whenever I had a question, and his expertise and insights were indispensable in the development of this work. Additionally, I would like to extend my thanks to Alexander Liu Cheng from KEWAZO, who suggested me the subject and without whom this work would not have been possible. I would also like to express my gratitude to the KEWAZO team for granting me the opportunity to turn this vision into a reality.

I would also like to acknowledge Prof. Schulz for his supervision and valuable advice throughout the thesis development process. His guidance helped me understand the academic requirements and expectations. Furthermore, I want to express my gratitude to my friends from TH-OWL, Vatsapol Nanta and Jason Daniel, for their continuous support and advice during this journey.

Lastly, I would like to express my deepest appreciation to my partner, Yusuf Bozkus, for his unwavering support and motivation throughout the research, writing, and development of the thesis. His presence has played a crucial role in keeping me focused and driven.

Author

Muny-Roth Chev

Content

1 Introduction

In the construction work field, human cognition plays an important role in the context of task planning. While robots can nowadays be used to perform repetitive tasks with precise motion control, and/or carry important loads, the unstructured and complex nature of the construction field, prevent us to take full advantage of them for on-site application. (1) Indeed, to the contrary to the industrial environment, where preprogrammed robot thrives at executing quasi-repetitive tasks, on-site work has proven to be highly impractical for them, due to relatively loose work tolerances and deviations of as-built work from the project design. (2)

In order to tackle this issue, monitoring on-site robots would be one viable approach. The most effective solution would be the creation of a digital twin, which can also be defined as «a realistic digital representation of assets, processes, or systems in the built or natural environment» where data are synchronized from the physical to the digital, therefore it is seen as a technology that enables the physical and virtual space to communicate. (3) Digital twin (DT) presents a new approach in the AEC sector: it is not only a building visual representation but can be enhanced with real-time data to diagnose the asset state and with the integration of statistic, probabilistic or AI models to allow predictive skills. Twins can be used for the following applications: real time monitoring, simulation, diagnosis, and performance prediction. (3)

By integrating real-time data from sensors and IoT devices, this tool enable facility managers to monitor operations, and analysis of robot performance, enabling proactive maintenance and continuous improvement. Robotic construction can save time, improve geometrical accuracy, reduce construction cost and improve site safety and human well-being. It is also a good way for construction workers to visualize the project and get feedback to be in a visual format to better understand and respond efficiently to the issue at hand. (4)

---

(1) Wang, Xi, Ci-Jyun Liang, Carol C. Menassa, and Vineet R. Kamat. "Interactive and Immersive Process-Level Digital Twin for Collaborative Human–Robot Construction Work." Journal of Computing in Civil Engineering 35, no. 6 (November 1, 2021): 04021023.

(2) Dawod, Mohamed, and Sean Hanna. "BIM-Assisted Object Recognition for the on-Site Autonomous Robotic Assembly of Discrete Structures." Construction Robotics 3, no. 1 (December 1, 2019): 69–81.

(3) Bruno Daniotti. "Digital Transformation in the Construction Sector: From BIM to Digital Twin." In Digital Transformation - Towards New Frontiers and Business Opportunities, edited by Alberto Pavan, translated by Claudio Mirarchi, Ch. 6. Rijeka: IntechOpen, 2022.

(4) Ravi, Kaushik Selva Dhanush, Ming Shan Ng, Jesús Ibáñez, and Daniel Hall. Real-Time Digital Twin of On-Site Robotic Construction Processes in Mixed Reality, 2021.

Figure 1 : *Conventional feedback loop*

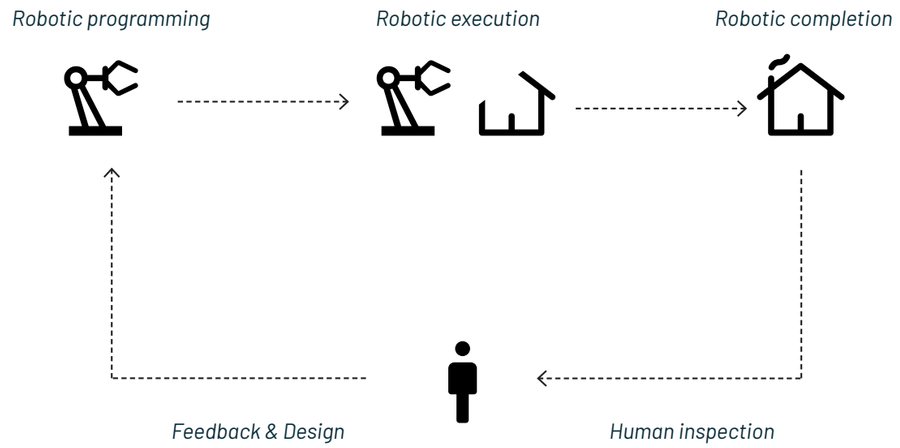
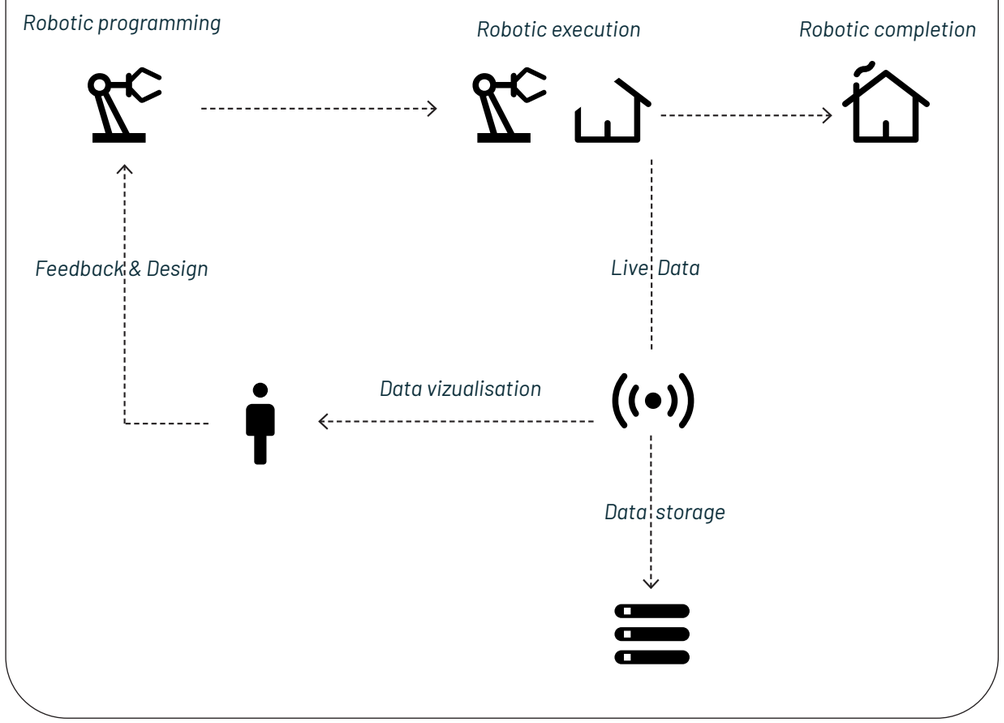


Figure 2 : *New feedback loop with DT*



Working with a digital twin implies an alternative feedback loop to the work for on-site construction robotic. Instead of getting the information at the end the whole system integrates information throughout the processes using sensors and IoT devices. The data produced in the process is stored in a database to use for future purposes such as progress monitoring or process automation. The proposed feedback loop shown in *Figure 2* allows users to communicate with the robot in a much more efficient way.

1.1 Definition and origins

NASA's Apollo space program was the first program to use the «twin» concept. The program built two identical space vehicles so that the space vehicle on earth can mirror, simulate, and predict the conditions of the other one in space. The vehicle remained on earth was the twin of the vehicle that executed mission in the space. The first use of the «digital twin» terminology appeared in Hernández and Hernández's work. Digital twin was used for iterative modifications in the design of urban road networks. However, it is widely acknowledged that the terminology was first introduced as «digital equivalent to a physical product» by Michael Grieves at University of Michigan in 2003. Nowadays the definition has extended to additional characteristics that includes the notion of properties, behavior and interactions. (5)

A clear definition of a DT can, however, be found in Grieves and Vickers (2016) (6) as:
«a digital informational construct about a physical system ... created as an entity on its own. This digital information would be a «twin» of the information that was embedded within the physical system itself and be linked with that physical system through the entire lifecycle of the system.»
A DT could then be said to take the level of integration further, in that information regarding the state of the physical and digital components can be interrogated simultaneously and compared.

In modern times, the concept of a digital twin encompasses the following key characteristics: (7)
1. It serves as a virtual representation of a physical object, system, or process.
2. It functions as a digital replica that mirrors its real-world counterpart, capturing its properties, behavior, and interactions within a virtual environment.
3. The concept also enables the synchronization and integration of data and information between the physical and digital realms.

---

(5) Liu, Mengnan, Fang Shuiliang, Huiyue Dong, and Cunzhi Xu. "Review of Digital Twin about Concepts, Technologies, and Industrial Applications." Journal of Manufacturing Systems 58 (July 1, 2020).

(6) Grieves, Michael, and John Vickers. "Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems," 85–113, 2017.

(7) "What Is a Digital Twin? | IBM." Accessed June 10, 2023. https://www.ibm.com/topics/what-is-a-digital-twin.

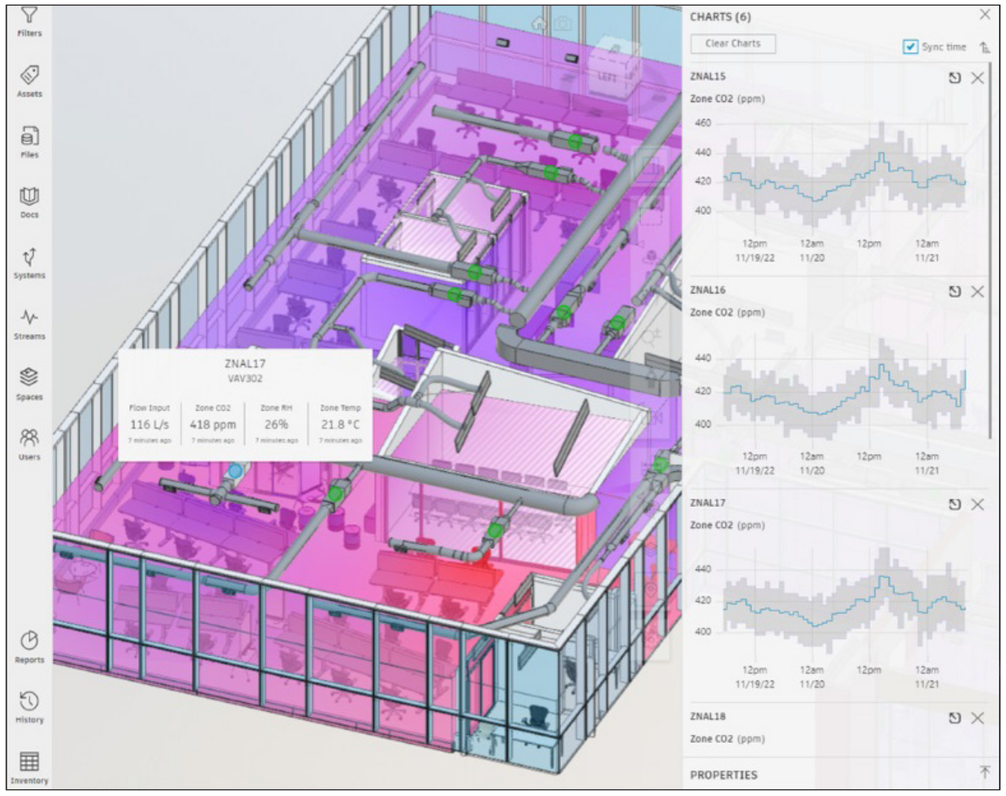


Figure 3 : *Example of Product Digital Twin in AEC industry. Autodesk Tandem Sceenshot*

Digital twins can be classified into two types (4). The first type is Product digital twin, which in the construction industry primarily refers to a digital replica of the building itself rather than the construction process. The second type is Process digital twins, which capture real-time or historical data related to the behavior, performance, and condition of the physical asset. Additionally, we will also delve into the concept of Spatial Computing in Mixed Reality (MR), which enhances the former DT concept by introducing virtual object augmentation and adding another dimension to its capabilities.

1.2 State of the art

1.2.1 Product digital twin

A Product digital twin is a digital representation of a physical asset. It captures the properties and behavior of the asset through modeling, analysis, and simulation. The key distinction between a digital model and a digital twin is the link between the physical and digital versions. In the construction industry, a product DT is commonly known as the digital asset of a building during its operational phase. It includes the building's BIM model, along with real-time integrated data modeled with semantic relationships. (4)

BIM models contain comprehensive information about the design, construction, and operation of a building, serving a multitude of purposes including design coordination, construction planning, and facility management. These models are utilized across the entire lifecycle of a building, starting from the design phase, continuing through the construction phase, and extending to the operation and maintenance phase. (8)

1.2.2 Process digital twin

A Process DT involves the real-time synchronization of digital and physical operations to achieve dynamic behavior. It is used in manufacturing environments for monitoring, control, diagnostics, prediction, and efficient collaboration between humans and robots. Process DT can also be applied to product lifecycle management and includes features like robotic process simulation, two-way communication with robots, real-time data monitoring, and data storage for future use. (4)

---

(4) Ravi, Kaushik Selva Dhanush, Ming Shan Ng, Jesús Ibáñez, and Daniel Hall. Real-Time Digital Twin of On-Site Robotic Construction Processes in Mixed Reality, 2021.

(8) Sepasgozar, Samad M. E., Ayaz Ahmad Khan, Kai Smith, Juan Garzon Romero, Xiaohan Shen, Sara Shirowzhan, Heng Li, and Faham Tahmasebinia. "BIM and Digital Twin for Developing Convergence Technologies as Future of Digital Construction." Buildings 13, no. 2 (February 2023): 441.

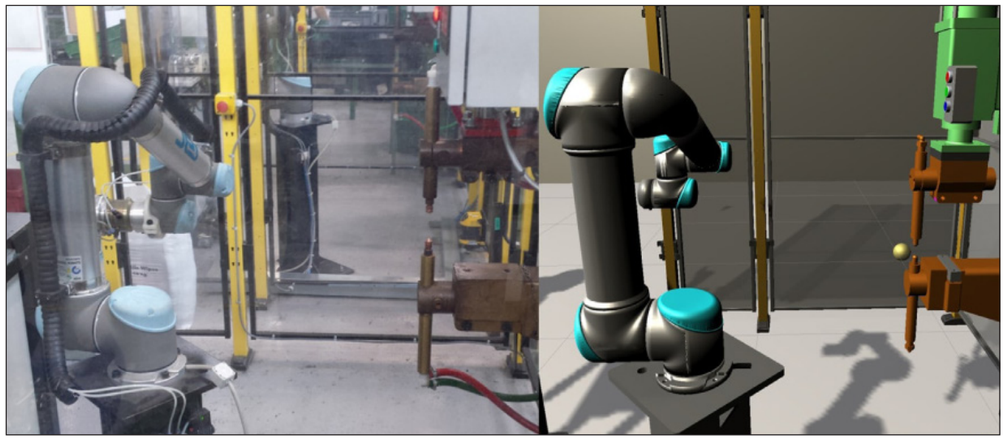


Figure 4 :*A side by side view of the* ***physical*** *and* ***digital twins*** *in real-world industrial welding process.*

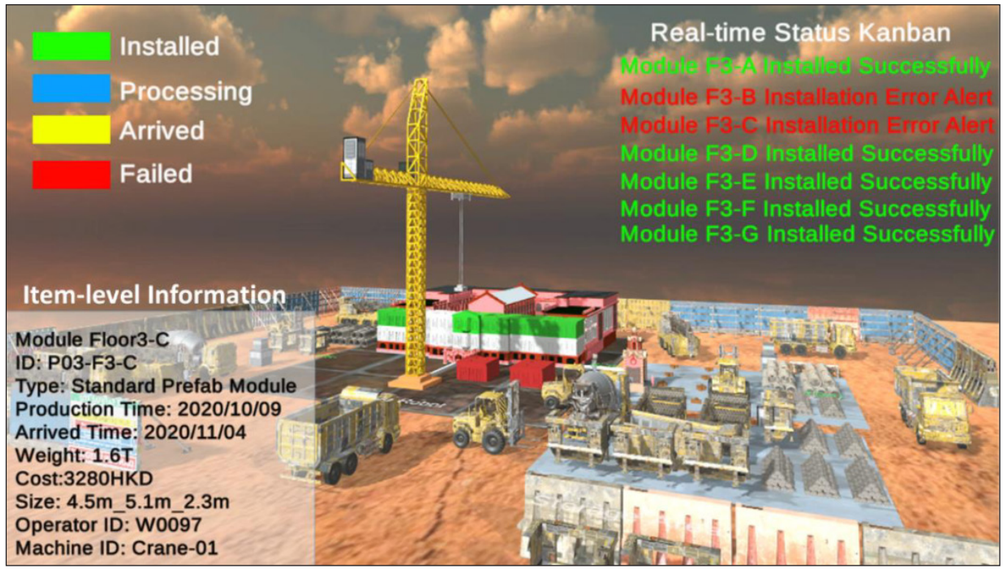


Figure 5: *Digital twin of the construction site developed by the university of Hong Kong*

The main difference there is between Product DT and Process DT is that the Process twin primarily deals with the dynamic behavior and synchronization of operations in real-time while a Product twin focuses on creating a digital representation of a physical asset, capturing its properties and behavior, commonly used in industries like construction for design and management throughout the product's lifecycle.

In this section, we present a case study of a real-world industrial manufacturing process in the UK (9). It is a first good approach to the notion of Process twin in the industry as the researchers created a DT capturing in real time the movement of an industrial robot. (*Figure 4*) The case study focuses on collaborative robotics, where multiple digital twins work together within a modular framework. This framework allows different components to be combined to create a complete scenario, enabling the identification and management of safety concerns in new situations. Indeed, this project addresses the challenges of safety assurance in industrial collaborative robotics. The researchers have developed a modular DT framework that enables online and offline development, testing, deployment, and validation of safety tools.

Specifically, the case study involves the exchange of a work-piece between a human operator and a collaborative robot for spot-welding. To ensure safety, the robot uses a shared handover table to transfer assembled/welded components between the operator and itself. Safety measures, such as a safety LIDAR (laser technology) at the welding machine base and a light barrier at the handover table, monitor and provide feedback on any potential safety issues.

A team from the university of Hong Kong has developed a Process Digital twin for the industry with a focus on performance. (*Figure 5*) While not specifically focused on robotic, this example is particularly relevant in the realm of on-site construction, also showing all the challenges that comes with creating a twin for the building site. On-site resources are converted into smart Objects attaching with wireless devices to collect and integrate real-time data, such as identity, location, cost, and construction progress. Through smart mobile gateway, various on-site resources and activities could be real-timely interoperated with their corresponding digital twins. Cloud-based services facilitate monitoring through virtual models and offer remote control capabilities with automatic navigations. This example also shows us the number of parameters that is necessary to take into account in the conception of a twin for the building site. (10)

---

(9) Douthwaite, James, B. Lesage, Mario Gleirscher, Radu Calinescu, Jonathan Aitken, Rob Alexander, and James Law. "A Modular Digital Twinning Framework for Safety Assurance of Collaborative Robotics." Frontiers in Robotics and AI 8 (December 1, 2021)

(10) Jiang, Yishuo, Ming Li, Daqiang Guo, Brave Wu, Ray Zhong, and George Q. Huang. "Digital Twin-Enabled Smart Modular Integrated Construction System for on-Site Assembly." Computers in Industry 136 (April 1, 2022): 103594.

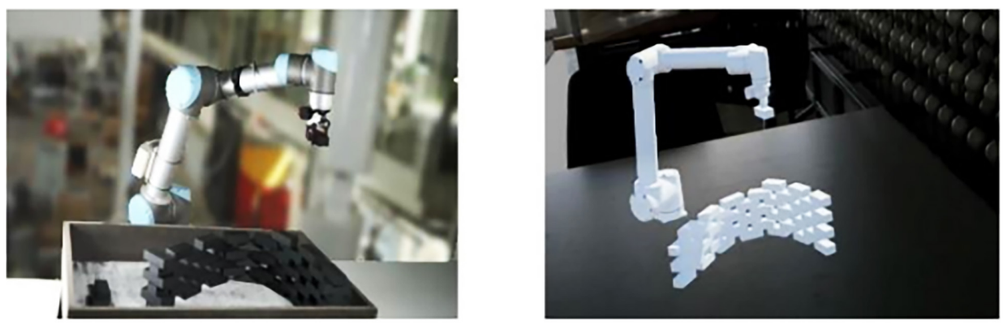


Figure 6: *Remote monitoring*

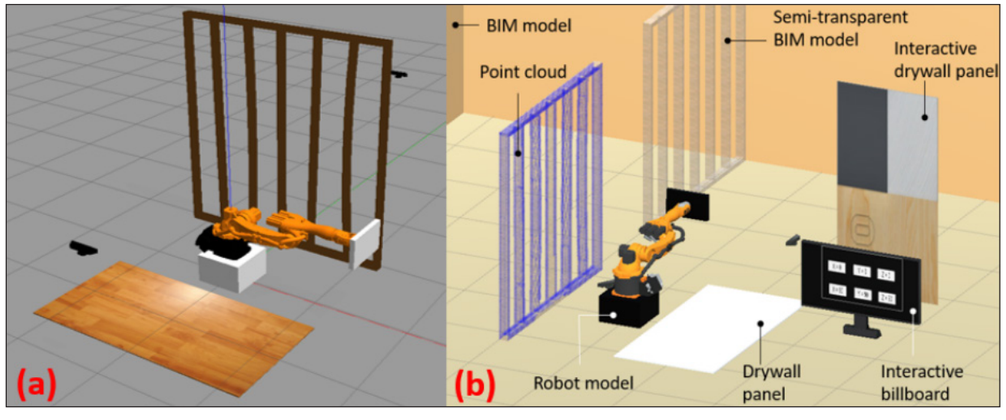


Figure 7: *(a) Robot operation environment (b) VR environment*

1.2.2 Spatial computing in Mixed-reality

Spatial computing in mixed reality (MR) involves overlaying virtual objects onto the real world through a head-mounted device, allowing users to see both the real world and the virtual objects within it. This concept facilitates real-time interaction between virtual objects and the physical environment, enabling applications such as controlling robots. In the construction industry, MR can be utilized for various purposes including safety collaboration, site survey, prefabrication, remote design, worker training, and facility management. However, there is currently no defined application of MR in robotic construction processes.

A good example of Spatial computing in mixed reality was developed by the team of the university of researchers at the ETH Zurich, Switzerland (4) Their work focuses on exploring effective human-robot collaboration in automated construction processes using real-time digital twin (DT) technology in a mixed reality (MR) construction environment. The study develops a DT prototype for a robotic construction process, establishing two-way communication between the physical robot and its virtual model. Real-time process data is collected from the robot and transmitted to visualization and database platforms. The workflow enables the transfer of real-time data to MR headsets, providing direct visual feedback and interaction with the robot arm. The prototype is validated through a case study demonstration of robotic masonry construction. (*Figure 6*) In the future, this work can be applied in a scenario where a digital fabrication engineer uses virtual commissioning in the actual construction site as a pre-check before implementing the robotic process. With the help of mixed reality (MR), the robotic process can be assessed on-site. This approach can prevent collisions in a complex construction site and overcome delays caused by uncertainties in execution.

Another interesting example of Spatial computing in Mixed-reality would be the framework developed by a research team at Michigan University (USA) allowing human-robot interaction and collaboration within a real-time, immersive virtual reality (VR) digital twin that is created by combining the as-designed BIM model and the evolving as-built workspace geometry obtained from on-site sensors. (11)

In this example, Immersive virtual reality (VR) offers advantages by providing users with realistic experiences and overcoming limitations of the real world. For example, users can access additional information not available in the real world, like comparing designs with actual structures, and can surpass real-world constraints such as gravity. The digital twin system incorporates interactive VR elements, including a versatile billboard. (*Figure 7*)

(4) Ravi, Kaushik Selva Dhanush, Ming Shan Ng, Jesús Ibáñez, and Daniel Hall. Real-Time Digital Twin of On-Site Robotic Construction Processes in Mixed Reality, 2021.

(11) Kamat, Vineet. "Real-Time Process-Level Digital Twin for Collaborative Human-Robot Construction Work." edited by Hisashi» «Osumi "Furuya, Hiroshi", "Tateyama, Kazuyoshi," 1528-35. International Association for Automation and Robotics in Construction (IAARC), 2020.

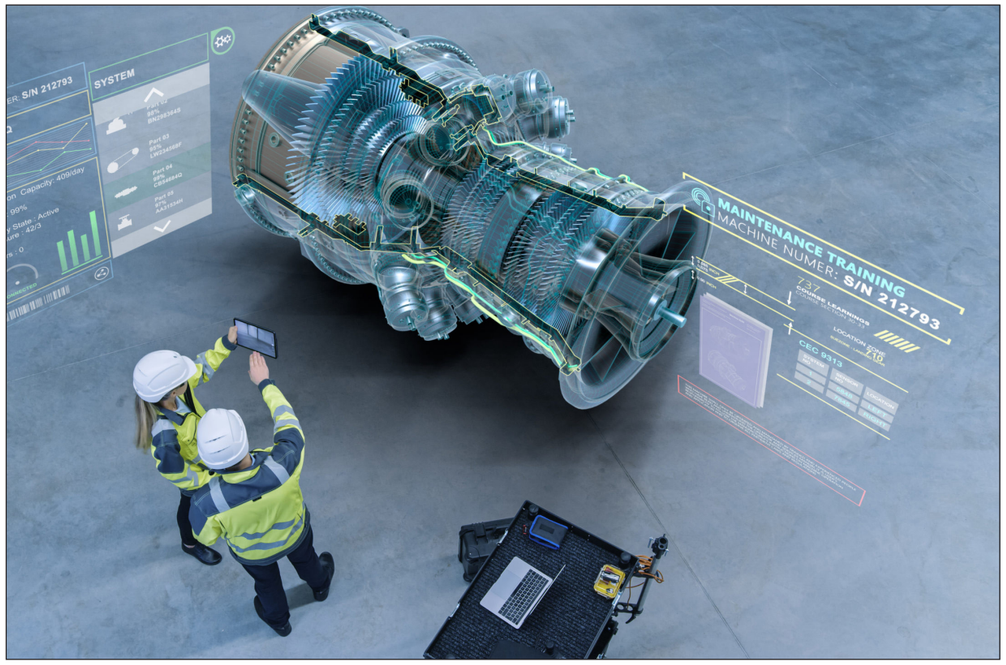


Figure 8: *Creative vision, with the future of industry actively using digital twins*

The billboard serves two purposes: displaying system messages that are not directly accessible in the physical construction environment (such as warnings) and functioning as an input device where users can interact with buttons on the screen to give commands. To accommodate complex construction environments, users can manipulate the billboard's position and orientation using VR controllers. Additionally, interactive construction materials have been created for tasks like pick-and-place, allowing users to grab and suspend objects in the air for high-level task planning.

1.3 Research question

Although digital twins offer numerous opportunities for on-site robotics, their adoption is still limited. One primary reason is the complex and unstructured nature of construction environments, which makes it challenging to accurately capture and replicate the physical conditions in a digital twin. Each construction project is unique, with different site conditions, designs, and requirements. In a specific example provided by the University of Hong Kong, the team used various sensors and systems, resulting in a potentially overwhelming display of data in the digital twin. (10)

Another hindrance to their development is the lack of standardization in the process. The field of digital twins in robotics for on-site construction is still evolving, and there is a lack of standardized processes and frameworks. Each robot has its own unique characteristics, requiring a different workflow to create a digital twin for each one. In the previous case studies we have examined, each digital twin utilizes a unique workflow distinct from the others.

Furthermore, while process digital twins are extensively researched and prototypes are created in academic settings, their implementation in the industry remains limited. Each robot's deployment necessitates a distinct workflow. Additionally, research indicates that important features like performance prediction are often not implemented or adequately researched. The integration of AI technologies is essential to establish a comprehensive construction digital twin that is self-reliant, self-learning, and capable of adapting to the project's needs.

How can an adaptable digital twin workflow be developed and implemented in the construction industry, considering the diverse range of possible workflows and the current limited adoption of process digital twins in real-world applications?

(10) Jiang, Yishuo, Ming Li, Daqiang Guo, Brave Wu, Ray Zhong, and George Q. Huang. "Digital Twin-Enabled Smart Modular Integrated Construction System for on-Site Assembly." Computers in Industry 136 (April 1, 2022): 103594.

Figure 9: *LIFTBOT in action, view of the robotic module and the platform*

2. Case study

2 . 1 Presentation of LIFTBOT by KEWAZO

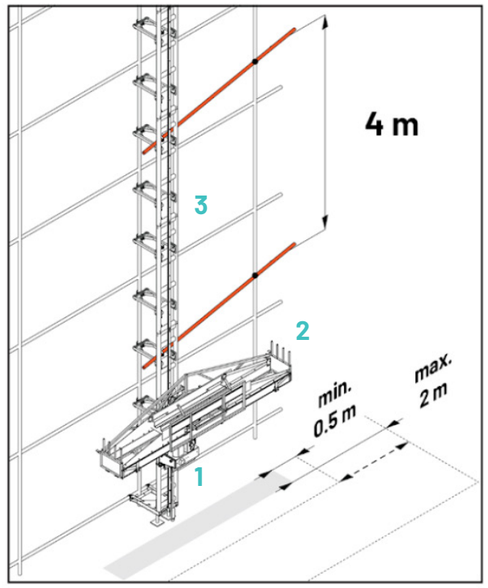
LIFTBOT is an advanced robotic hoist developed by KEWAZO, a German company, with the aim of enhancing safety and efficiency in scaffolding processes during assembly and disassembly tasks. This intelligent equipment can be remotely operated and leverages the power of a cloud-based data analytics platform.

The LIFTBOT system consists of three primary components: the Robotic Module (RM), the Transportation Platform (TP), and Rails. The RM and TP work in tandem, moving vertically along the rails to transport scaffolds and construction materials to different levels, reaching heights of up to 50 meters. The platform offers additional flexibility by allowing rotation and opening, facilitating workers' tasks and enabling efficient loading and unloading. In addition to the physical robotic system, KEWAZO provides access to the ONSITE platform, an online tool for analyzing the sensor data collected by the LIFTBOT system. This platform empowers clients to monitor the performance of their robot and optimize its operations on-site, ensuring maximum productivity and safety.

The system captures a vast amount of sensor data, comprising more than 400 different data streams. This data is continuously transmitted to KEWAZO's ONSITE platform for analysis. Through this process, raw data is transformed into meaningful information, which is then condensed into actionable insights. The raw data would act as the fundamental building block for creating a Digital Twin (DT) of the system. In this DT, every action, state of function, and operational phenomenon of the robot can be mirrored and replicated in near-real-time, enabling comprehensive monitoring, predictive maintenance, and performance optimization. The DT would serve as a virtual representation of the physical LIFBOT system, providing valuable insights and facilitating informed decision-making integrated into the existing ONSITE platform. By leveraging its capabilities and the Digital Twin concept, LIFTBOT contributes to improved safety, efficiency, and productivity in construction activities.

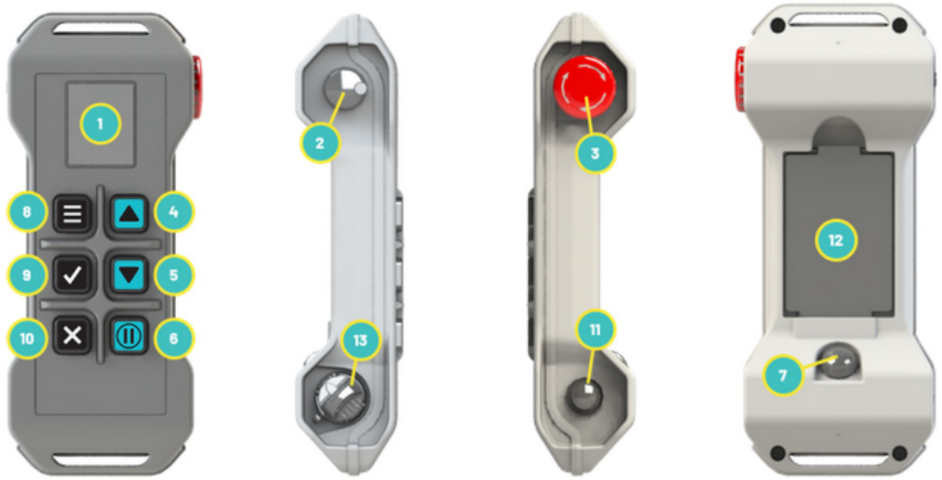
2.2 Objectives of the research

Our research aims to address the following objectives based on the initial research question: «How can an adaptable digital twin workflow be developed and implemented in the construction industry, considering the diverse range of possible workflows and the current limited adoption of process digital twins in real-world applications?»



Max. Head / Rail length	50 m
Safety distances around Transportation Platform in the lowest loading point	at least 0.5 m
Nominal load / loading capacity of the transport platform	100 kg
Position of the load handling attachment out of service	Robot in the lowest loading point

Figure 10: *The LIFTBOT system with a robotic module (1) a Transportation Platform (2) and a Rail system (3)*



1. LCD Screen
2. Rotary Switch "Level/Auto/Manual/ Offset"
3. EMERGENCY STOP Button
4. "Move UP" Button
5. "Move DOWN" Button

Figure 11: *Remote controller*

In order to achieve these objectives, preliminary investigations are necessary in areas such as software platforms, data management, digital twin development, and performance extrapolation specific to the case of LIFTBOT.

The second goal of this thesis is to create a visual representation of the Digital Twin for LIFTBOT that accurately reflects its state at different stages of the project over time. This visualization goes beyond a mere didactic representation or abstraction. It entails accurately representing every component of the Robotic Module (RM) and Transportation Platform (TP) for which sensor data is collected. (*Figure 10*) Furthermore, the visualization should depict the loading and unloading of transported loads at each level, providing the client with information on the amount transported from one level to another.

The third goal of the thesis is to develop a mathematical or computational model that utilizes performance indicators obtained from input data to simulate and extrapolate hypothetical scenarios for LIFTBOT beyond its existing data record. This simulation would offer insights into how LIFTBOT is expected to perform based on certain predefined variables, projecting its behavior from a specific time period (*t1*) to future times (*tn*), building on the existing data from the past (*t0* to *t1*).

2.3 Potential challenges

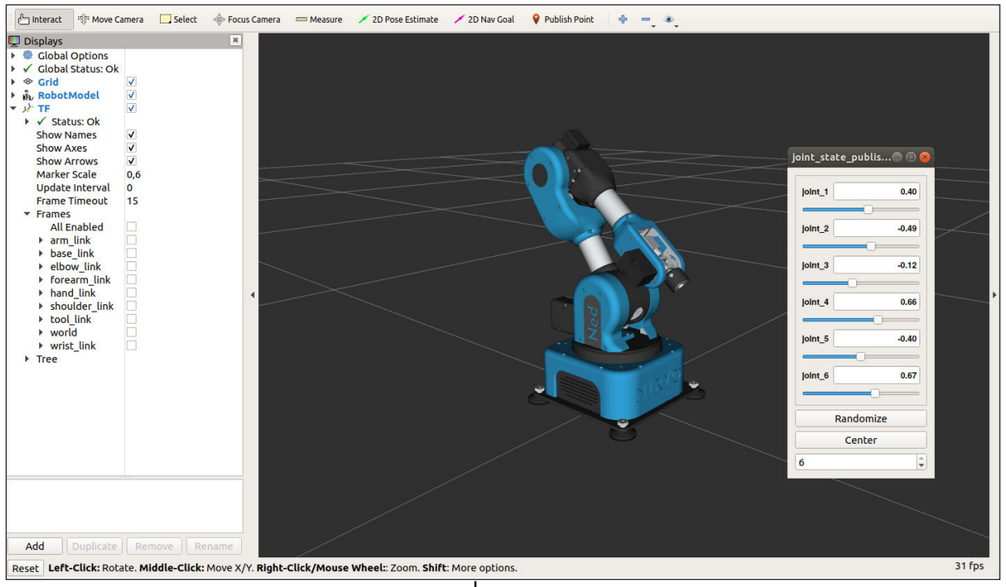
There are several potential challenges that may arise during the research process:

1. Time limitation: The given timeframe for the thesis may be insufficient to develop a comprehensive model encompassing real-time data rendering in 3D and future performance extrapolation. Due to the limited time, the focus of the thesis may primarily be on the visual aspects of the Digital Twin.

2. Uncertainty about the workflow: The uniqueness of the robot being developed poses challenges in determining the appropriate workflow for creating a digital twin for an on-site robot. There is limited available resources and guidance online specifically addressing the development of a digital twin for such robots.

3. Creating a user-friendly interface: Designing an interface that is user-friendly and intuitive becomes a challenge considering the vast amount of data involved and the complexity of the project. Ensuring that the interface is accessible and easily navigable for users is crucial for effective utilization of the digital twin system.

ROS interface



Gazebo

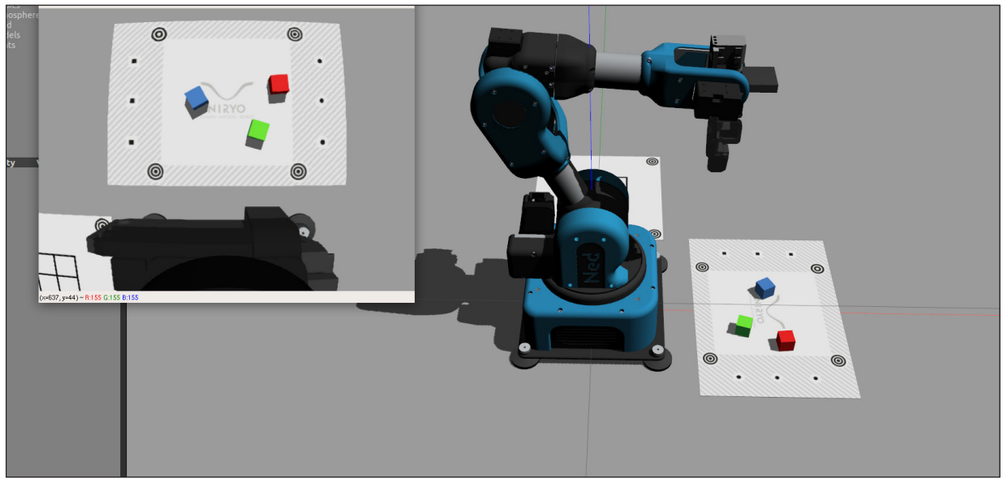


Figure 12: *Example of Workflow linking ROS for joint adjustement, then Gazebo for simultation*

3 Methodology

In this chapter, we delve into the process of creating a digital twin from scratch. The steps outlined below form the fundamental framework that drives the functionality of the digital twin. Although each robot digital twin in the construction field serves a unique purpose, there are common steps involved in their conception that we will explore here. By providing a step-by-step explanation of the technical process, we can gain a comprehensive understanding of the various methods and evaluate their relevance in our specific case. This breakdown enables us to assess the significance of each choice made during the development process and make informed decisions tailored to our particular needs.

3.1 Platform Selection for the Digital Twin

The initial stage in creating a digital twin involves choosing a suitable platform to host it. This decision is influenced by factors such as the type of robot and the specific objectives of the twin. Various platform software options are available for developing a digital twin of a robot. The following text will present some of the most utilized platforms and provide application examples.

ROS is a flexible framework for writing robot software. It offers libraries, tools, and communication infrastructure to develop robot digital twins and facilitate interaction with physical robots. It is an open-source system that integrates various tools and software libraries for robot operation. While ROS itself is not specifically designed for running simulations, it can serve as a framework for developing digital twins that are deployable and executable on specific hardware. Therefore, it is often combined with Gazebo, an open-source robot simulation platform. Gazebo enables users to design, model, and simulate robot behavior, making it an ideal choice for creating digital twins and testing robot functionalities. (12)

It is important to note that Gazebo primarily functions as a simulation platform and is not intended for creating standalone versions of digital twins or applications. For system-level design purposes, other popular platforms like MATLAB/Simulink, which offer simulation-based digital twins for virtual commissioning, are more suitable and interesting. These platforms provide a comprehensive environment for designing and testing systems at a higher level. (13)

(12) Qian, Wei, Zeyang Xia, Jing Xiong, Yangzhou Gan, Yangchao Guo, Shaokui Weng, Hao Deng, Ying Hu, and Jianwei Zhang. "Manipulation Task Simulation Using ROS and Gazebo," 2014.

(13) Phanden, Rakesh, Priavrat Sharma, and Anubhav Dubey. "A Review on Simulation in Digital Twin for Aerospace, Manufacturing and Robotics." Materials Today: Proceedings 38 (July 1, 2020).

Figure 13: *Example of stystem combining ROS to Unity3D for Virtual Reality and remote control*

On the other hand, game engines such as Unity or Unreal Engine are powerful real-time 3D development platforms commonly employed for creating virtual environments and simulations. They can be effectively utilized to construct digital twins of robots by incorporating realistic graphics, physics, and interactive capabilities. Particularly in the domain of on-site robotics and digital twin development, game engines find extensive use, as they enable users to leverage virtual reality and augmented reality technologies, taking the digital twin experience to the next level. (11)

Another advantage of game engines is their deployement possibilities. For exemple, Unity provides the capability to create standalone applications from digital twins. The engine empowers developers to build applications that can be executed on different platforms (such as Windows, macOS, Android, iOS) independently of the Unity editor. These standalone applications can be distributed to others for offline usage. Additionally, Unity supports online publishing, enabling the deployment of digital twins as web-based applications or hosting them on platforms like Unity Connect for online accessibility.

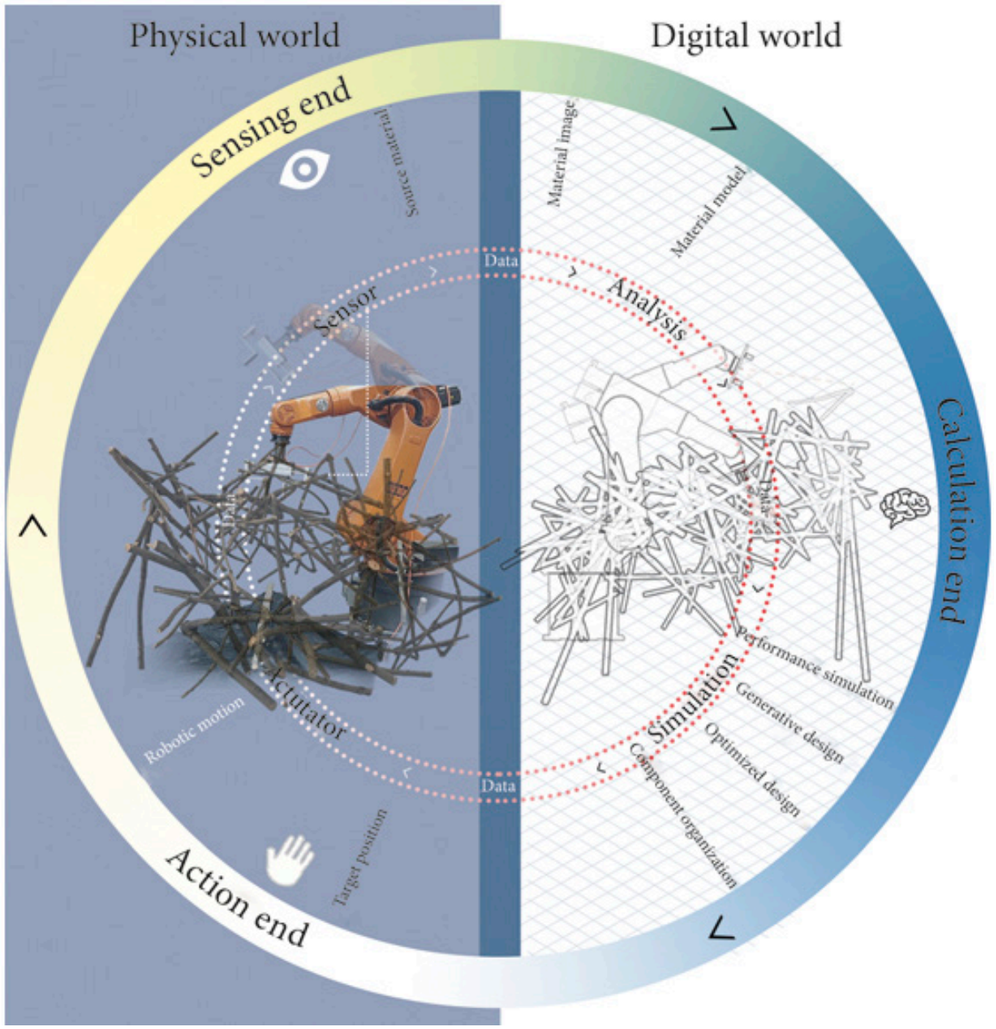
Many researchers have compared the workflows of ROS-Unity3D and the popular ROS-Gazebo robotics simulation environment in terms of their architecture, environment creation process, resource utilization, and accuracy when simulating real-time autonomous ground robots. ROS-Unity3D proves to be a promising alternative to ROS-Gazebo due to its extensive file format support and a robust scripting interface, enabling the creation of customized functionalities. Tests demonstrate that ROS-Unity3D excels in handling larger environments, delivering superior shadow quality, achieving comparable or even better visual-based SLAM performance, (SLAM performance refers to the effectiveness and accuracy of a system or algorithm in simultaneously localizing the robot's position within an environment and mapping the surroundings.) and offering advanced real-time LiDAR simulation capabilities in comparison to ROS-Gazebo. LiDAR simulation refering to the process of emulating or reproducing the behavior and characteristics of a LiDAR (Light Detection and Ranging) sensor within a virtual or simulated environment.

Nonetheless, ROS-Gazebo possesses its own strengths over ROS-Unity3D. It boasts a more streamlined interface between ROS and Gazebo, provides a wider array of pre-existing sensor plugins, and exhibits better resource efficiency when simulating smaller environments. However, considering the versatility and capabilities offered by ROS-Unity3D, especially in simulating larger environments and incorporating realistic LiDAR simulation, it emerges as an enticing choice for simulation tasks. (14)

---

(11) Kamat, Vineet. "Real-Time Process-Level Digital Twin for Collaborative Human-Robot Construction Work." edited by Hisashi» «Osumi "Furuya, Hiroshi", "Tateyama, Kazuyoshi," 1528–35. International Association for Automation and Robotics in Construction (IAARC), 2020.

(14) Platt, Jonathan, and Kenneth Ricks. "Comparative Analysis of ROS-Unity3D and ROS-Gazebo for Mobile Ground Robot Simulation." Journal of Intelligent & Robotic Systems 106 (December 20, 2022).



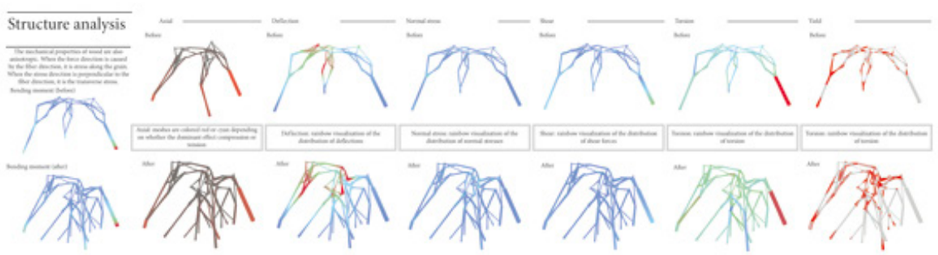


Figure 14: *Usage of Rhino and Grasshopper for design iteration and structural analysis*

Another commonly used workflow in academic and research contexts within the field of computational design involves the utilization of Rhinoceros 3D and Grasshopper software. These powerful design and modeling tools are widely employed in architecture, engineering, and construction (AEC) industries. While their primary purpose is not focused on digital twin creation, they can be integrated into the overall process and prove particularly interesting when developing digital twins of design processes.

For instance, a team of researchers from ETH Zurich in Switzerland, previously mentioned for their work on effective human-robot exploration, utilized a Rhino + Grasshopper process as part of their project. (4) In this process, the masonry wall was initially modeled using computational design techniques, representing the bricks as mesh objects within Rhino. Relevant data for robotic execution, such as the midpoints of each brick, the orientation plane, and the brick width, were extracted from the modeling phase.

Within the Grasshopper script, the planes representing the midpoints of the blocks were stored using the plane component. These planes were then connected to the robotic execution component, enabling the precise placement of bricks at their designated locations. The robot's path was visually coded using the Robots plugin in Grasshopper, which facilitated the repetitive process of picking and placing bricks. Once the path was simulated and ready for execution, the data could be transmitted to the robot via a LAN connection, allowing it to pick up bricks from a specified location.

The simulation and live data obtained from Rhino Grasshopper are visualized in mixed reality (MR). The mesh data is transmitted as a game object from Grasshopper to Unity, leveraging the Rhino integration. Inside Unity, the data is further visualized using MR headsets with the assistance of Zero Iteration. Additionally, the real-time positions of the robot are captured every second from Grasshopper and transmitted to DynamoDB using Amazon Web Services.

Another advantage of utilizing the Rhino/Grasshopper toolset is its versatility in not only enabling robotic simulation but also allowing users to explore multiple design possibilities. (15) However, when it comes to deployment, it should be noted that Rhino and Grasshopper have limitations in creating standalone versions of the digital twin. Although there are some plugins like Speckle that facilitate showcasing parametric models online, the options for demonstrating simulations or real-time data are somewhat limited.

---

(4) Ravi, Kaushik Selva Dhanush, Ming Shan Ng, Jesús Ibáñez, and Daniel Hall. Real-Time Digital Twin of On-Site Robotic Construction Processes in Mixed Reality, 2021.

(15) Zhang, Ye, A. Meina, Xuhao Lin, Kun Zhang, and Zhen Xu. "Digital Twin in Computational Design and Robotic Construction of Wooden Architecture." Advances in Civil Engineering 2021 (April 2, 2021): 1–14.

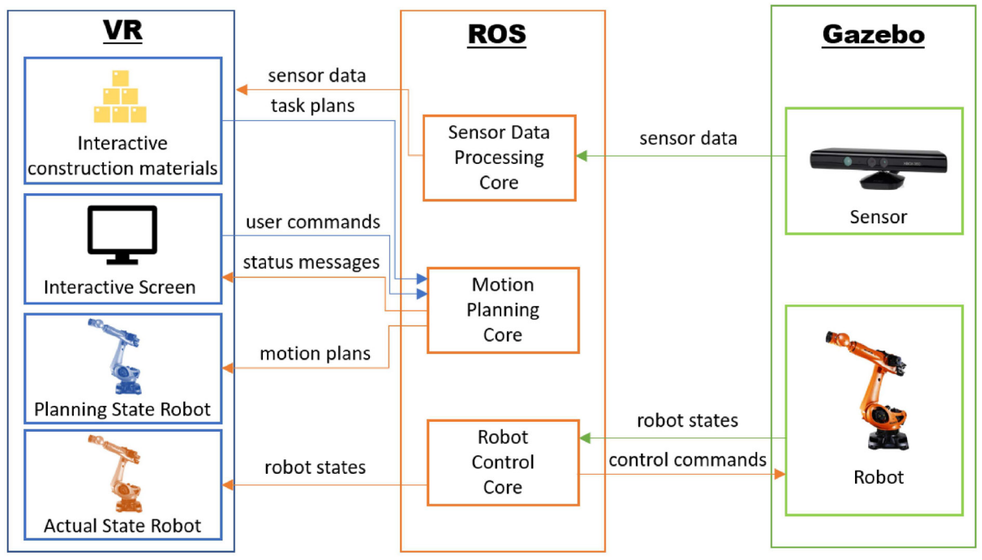


Figure 15: *A possible system communication framework using a combination of ROS as the core system in synergy with Gazebo for detection and Unity for simulation*

Sometimes, achieving the desired goals involves combining multiple workflows. This approach was employed by a research team in Michigan, where ROS served as the core computational system. ROS facilitated communication with Unity3D, Gazebo, and the physical robot.(11) Apart from communication, ROS was responsible for sensor data processing, motion planning, and robot control within the system. (*Figure 15*)

Upon program initiation, Gazebo initiated the continuous publication of sensor data and robot states to ROS. Simultaneously, ROS processed the sensor data and transmitted the processed data along with robot states to Unity3D. These data were then visualized in Unity3D as a point cloud and the state of the «execution» robot in virtual reality (VR). Based on the point cloud, the user developed a task plan, which, after confirmation, was sent to ROS. Subsequently, ROS generated a collision-free motion plan based on the task plan. The motion plan was sent back to Unity3D, allowing the user to visualize it on the «planning» robot. If the user found the motion plan unsatisfactory, they had the option to adjust their task plan or request a new motion plan from ROS. ROS would then generate a revised motion plan in response. Upon user approval, a message was sent to ROS, which converted the motion plan into execution commands to control the physical robot. As the physical robot executed the assigned tasks, updated robot state messages were received by both ROS and Unity3D, resulting in corresponding movements of the «execution» robot in VR.

3 . 2 Data management

Digital processes, particularly in the context of digital twins for construction processes, generate vast amounts of data. This real-time data, acquired from sensors, holds significant potential for process tracking and project monitoring.

Several factors need to be considered, starting with the identification and integration of multiple data sources. It is crucial to recognize and incorporate data from diverse sensing technologies to gather comprehensive information about the construction robot and its environment. With the advancements in information technologies, there has been a surge in the availability of real-time data collected by sensors, IoT devices, mobile devices, and wearable devices. This wealth of data can be harnessed to enhance understanding and facilitate analysis in construction processes. (16)

(11) Kamat, Vineet. "Real-Time Process-Level Digital Twin for Collaborative Human-Robot Construction Work." edited by Hisashi» «Osumi "Furuya, Hiroshi", "Tateyama, Kazuyoshi," 1528–35. International Association for Automation and Robotics in Construction (IAARC), 2020.

(16) Zhang M, Tao F, Huang B et al. Digital twin data: methods and key technologies [version 2; peer review: 4 approved]. Digital Twin 2022, 1:2 (https://doi.org/10.12688/digitaltwin.17467.2)

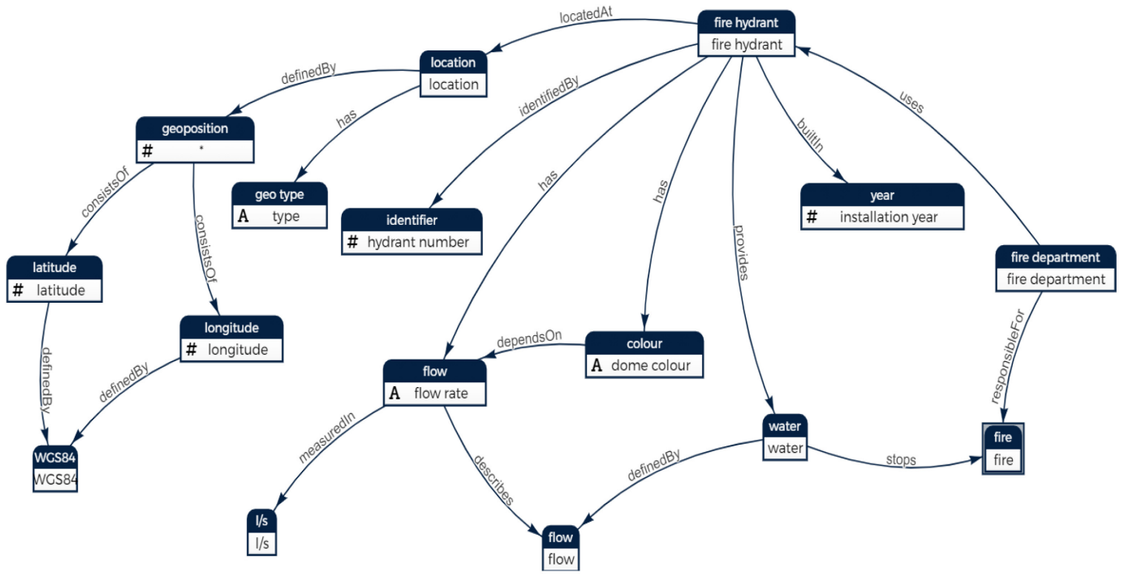


Figure 16: Example semantic model. This model describes water hydrants in Edmonton, Canada. Each node consists of a concept and, if directly mapped to a data attribute, the attribute name accompanied by its data type.

Additionally, to enhance the accuracy, efficiency, and adaptability of DT-based services, comprehensive data selection is crucial. This encompasses a broad range of information, including both normal and abnormal states, common and rare events, and certain and uncertain scenarios. By incorporating such diverse data, DT-based services can offer performance prediction, process optimization, and quality assurance capabilities, among others.(16)

In terms of data utilization, a significant step is defining specific purposes and objectives for leveraging the data obtained from the digital twin. This could involve optimizing processes or enabling intelligent semantic platforms. Semantic data relationships refer to meaningful connections and associations between data elements based on their contextual or conceptual similarities, allowing for enhanced understanding and analysis of the data. These relationships help create a framework that enables the seamless integration and interpretation of data within the digital twin ecosystem. (17)

Finally, one major aspect to take into account in terms of data management for the development of a digital twin is standardized Data Transfer, which involves the use of ontology-based communication methods to ensure standardized and interoperable data exchange between systems and platforms. This approach utilizes standardized data structures and definitions, known as ontologies, to facilitate meaningful data transfer and interoperability across different stages of the construction process.

Indeed, one challenge in DT applications is the lack of data universality. Transferring DT across diverse application scenarios with varying data acquisition requirements and constraints poses difficulties. These challenges include differences in physical entities (e.g., robots, machine tools, and autonomous vehicles), data interfaces, and communication protocols. Achieving seamless data integration and sharing across different application scenarios such as design, production, and maintenance is hindered by varying data formats. To address these challenges, it becomes necessary to unify data transformation to achieve high data universality.

(16) Zhang M, Tao F, Huang B et al. Digital twin data: methods and key technologies [version 2; peer review: 4 approved]. Digital Twin 2022, 1:2 (https://doi.org/10.12688/digitaltwin.17467.2)

(17) Boje, Calin, Annie Guerriero, Sylvain Kubicki, and Yacine Rezgui. “Towards a Semantic Construction Digital Twin: Directions for Future Research.” Automation in Construction 114 (June 1, 2020): 103179.

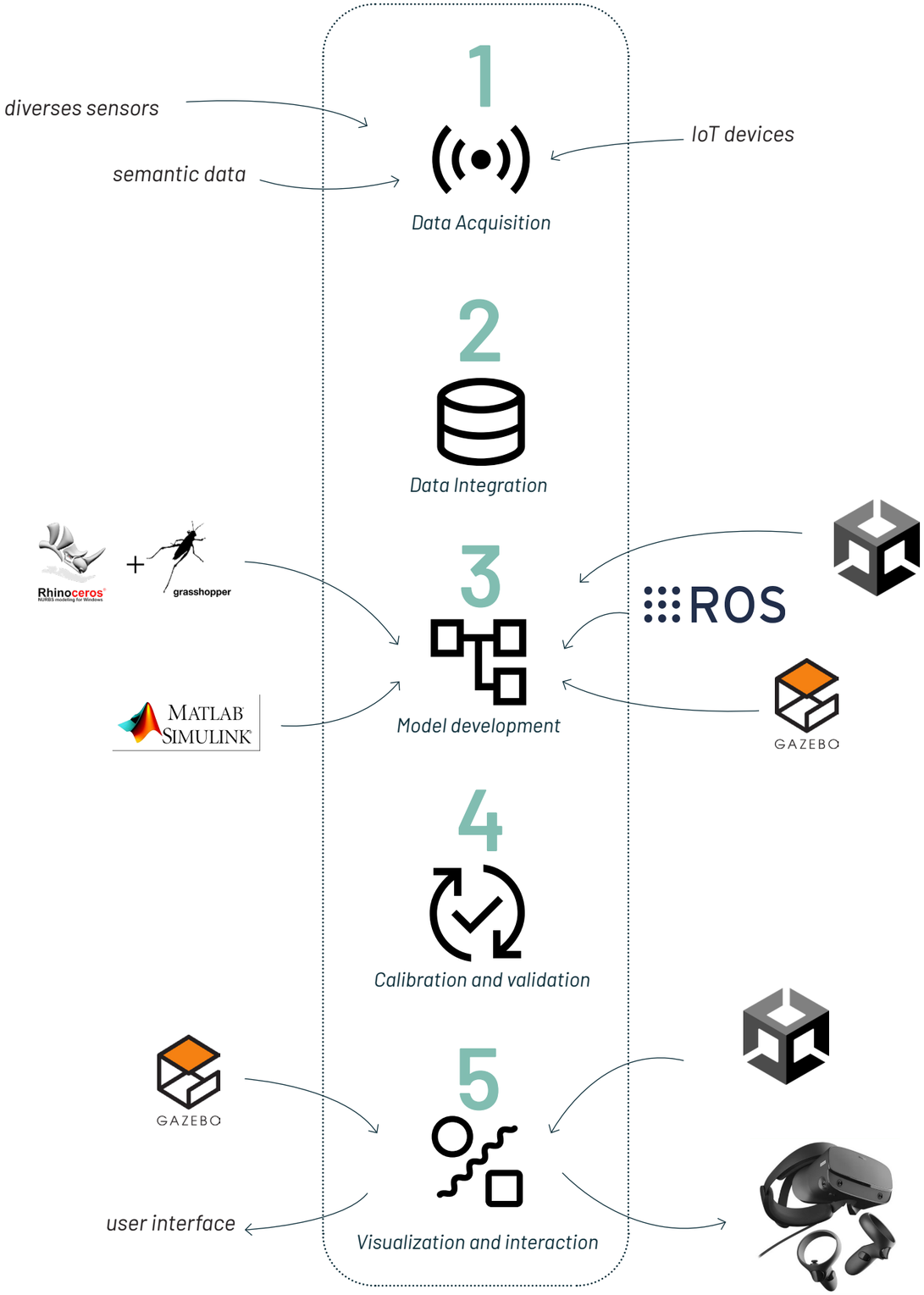


Figure 17: *The Different Stages of Creating a Digital Twin*

3 . 3 Digital twin development

Considering the previous findings, it is now possible to establish a step-by-step workflow for creating the digital twin of a construction robot.

Step 1: Data Acquisition
The initial step involves identifying and acquiring relevant data from diverse sources, including sensors and IoT devices. As mentioned earlier, the developer should focus on integrating data from multiple sensing technologies to gather comprehensive information about the construction robot and its environment. By incorporating data from various sources, semantic data relationships can be established, which have the potential to enhance the model. Additionally, integrating both real and virtual data from normal and abnormal states can contribute to the creation of a more accurate model.

Step 2: Data Integration
The next step is to integrate the acquired data into a unified platform or database to facilitate standardized data transfer. For example, KEWAZO primarily uses the relational database SQLite to store information obtained from the sensors. In certain cases, it may be crucial to implement data preprocessing and cleaning techniques to ensure data consistency and quality.

Step 3: Model Development
After collecting the data, the next step involves developing the virtual model. As we have previously discussed, the approach may vary depending on factors such as the robot's nature, the desired simulation type, and the need for online deployment or standalone functionality. Depending on the desired outcome, the developer may prefer a specific workflow or a combination of multiple platforms such as ROS, Unity, Gazebo, Rhino, and others. Ultimately, the final model represents the behavior and characteristics of the physical system or process. This can be achieved through mathematical models, physics-based simulations, or machine learning algorithms.

Step 4: Calibration and Validation
An important aspect of model development is the calibration of digital twin models using real-world data to ensure accuracy and reliability. This involves an iterative process of validating the models against the behavior and performance of the physical system, requiring thorough testing and adjustment.

Step 5: Visualization and Interaction
Certain platforms, such as game engines, facilitate the development of user interfaces or visualization platforms that enable users to interact with and monitor the digital twin. This can include 2D/3D visualizations, augmented reality (AR)/virtual reality (VR) experiences, dashboards, and real-time data displays.

3 . 4 Extrapolation for future performance

In addition to the previous steps, we would like to explore the realm of performance prediction in this thesis and understand the available methods. While there are limited examples of digital twins in construction robotics that integrate these steps, this particular aspect holds great relevance in developing a comprehensive and robust model.

Predicting the future performance of a robot based on its past performance can be approached using various methods. These methods can be categorized into groups that heavily rely on data-driven analysis, those that focus more on the physics of the robot, and others that utilize a combination of both techniques. (18)

3.4.1 Performance prediction data based

Statistical time series analysis offers an interesting approach to performance prediction based on data. Time series data consists of sequential observations recorded over time, providing valuable insights into the dynamics and interdependencies among variables. By analyzing patterns, trends, and dependencies within these data sequences, we can gain a deeper understanding of how variables influence each other. This particular model becomes particularly relevant when a substantial amount of historical performance data is available, enabling accurate short-term forecast predictions. (19)

On the other hand, regression algorithms offer a powerful method for establishing relationships between input variables (historical performance data) and an output variable (future performance). Regression serves two primary purposes. Firstly, it is commonly employed for forecasting and prediction. Secondly, it can be utilized to determine causal relationships between independent and dependent variables in certain cases. This technique proves especially valuable when dealing with a large and diverse dataset of historical performance data, aiming to capture intricate relationships between input variables and future performance. Moreover, it is particularly useful when the robot's performance is influenced by multiple factors and there is a desire to identify the most significant predictors. (20)

(18) https://jpt.spe.org/twa/author/hector-klie. "A Tale of Two Approaches: Physics-Based vs. Data-Driven Models." JPT, May 3, 2021.

(19) Arafet, K.; Berlanga, R. Digital Twins in Solar Farms: An Approach through Time Series and Deep Learning. Algorithms 2021, 14, 156.

(20) Maulud, Dastan, and Adnan Mohsin Abdulazeez. "A Review on Linear Regression Comprehensive in Machine Learning." Journal of Applied Science and Technology Trends 1 (December 31, 2020): 140–47.

3.4.2 Performance prediction physic based

Physics-based models use mathematical equations based on physical laws to simulate how a robot behaves. By understanding the robot's mechanics, sensors, and environment, we can predict how it will perform in different situations. These models are built using fundamental principles like Newton's second law and include details like friction, damping, and boundary conditions. Each parameter in the model has a clear physical meaning. Although these models can be complex and time-consuming to create, they provide accurate results and can be used to analyze different scenarios. In summary, physics-based models are highly accurate and understandable, making them useful for analyzing various situations. (21)

Hybrid approaches, combining different methods, are also possible. One approach is to use a physics-based model as a foundation and incorporate machine learning techniques to enhance predictions using historical data. This combination is particularly useful when there is a need to leverage the benefits of both physics-based modeling and machine learning. It is effective in refining predictions by incorporating historical performance data and addressing complex interactions that may not be captured by the physics-based model alone.

Data-driven models alone have many advantages as they do not rely on physics laws and can provide accurate predictions when sufficient data is available. They also offer fast performance, making them suitable for real-time applications. However, they are often limited to the training domain and may struggle to extrapolate results confidently. Finally, they typically require more data for training and calibration due to the absence of an initial mathematical structure. Integrating physics-based models with machine learning approaches appears to be a logical approach to leverage their respective strengths and mitigate their limitations.(22)

3.4.3 Performance prediction strongest models

Ensemble methods are utilized to enhance prediction accuracy and robustness by combining multiple models or algorithms. By leveraging the strengths of different approaches, ensemble methods provide more reliable predictions. Alternatively, reinforcement learning algorithms can be applied to train a digital twin that interacts with its environment, aiming to optimize its performance. By utilizing past experiences, the digital twin learns from its actions and feedback, allowing it to adapt and improve its future performance. This approach is particularly beneficial when the digital twin needs to continuously optimize its performance over time and adapt to changing conditions.(22)

(21) Ritto, T. G., and F. A. Rochinha. "Digital Twin, Physics-Based Model, and Machine Learning Applied to Damage Detection in Structures." Mechanical Systems and Signal Processing 155 (June 2021): 107614.

(22) Pawar, Suraj, Shady E. Ahmed, Omer San, and Adil Rasheed. "Hybrid Analysis and Modeling for next Generation of Digital Twins." Journal of Physics: Conference Series 2018, no. 1 (September 2021): 012031.

1	**Motion Parameters**			
2	**bit name**	**Byte Name**	**Units**	**Description**
3	-	"actual_position"	counts (245760 counts = 1m)	This Object represents the actual value of the position sensor (motor 1 encoder)
4	-	"actual_velocity"	counts pers second (245760 c/s = 1m/s)	This Object represents the actual value of the velocity sensor (motor 1 encoder)
5	-	"target_position"	counts (245760 counts = 1m)	The target position is the set-point position to which the drive should move
6	-	"target_velocity"	counts pers second (245760 c/s = 1m/s)	The target velocity is the input for the trajectory generator. The value is given in velocity units.
7	-	"profile_velocity"	counts pers second (245760 c/s = 1m/s)	Write the profile velocity to the SD. This object is the velocity normally attained at the end of the acceleration ramp during a profiled position move and is valid for both directions of motion. The profile velocity is given in user-defined speed units. It is converted to position increments per second using the velocity encoder factor.
8	-	"actual_position"	counts (245760 counts = 1m)	This Object represents the actual value of the position sensor (motor 2 encoder)
31	**Payload Related Parameters**			
32	**bit name**	**Byte Name**	**Units**	**Description**
33	-	"actual_torque"	‰ (to convert to Nm use the following equation) X/1000*4.8 (X is the raw value in ‰ of the rated torque of the motors)	This value returns the actual value of the torque of Motor 1
34	-	"actual_current"	A	This value returns the actual value of the current of Motor 1
35	-	"target_torque"	‰ (to convert to Nm use the following equation) X/1000*4.8 (X is the raw value in ‰ of the rated torque of the motors)	This value returns the target value of the torque of SD1
36	-	"actual_torque"	‰ (to convert to Nm use the following equation) X/1000*4.8 (X is the raw value in ‰ of the rated torque of the motors)	This value returns the actual value of the torque of Motor 2
37	-	"actual_current"	A	This value returns the actual value of the current of Motor 2
38	-	"target_torque"	‰ (to convert to Nm use the following equation) X/1000*4.8 (X is the raw value in ‰ of the rated torque of the motors)	This value returns the target value of the torque of SD2
39	-	"payload"	integer (to convert to kg use the following equation) (X/2500 - 8.7)*13.0835443 (X is the raw integer value of the payload)	Payload
40	-			
99	**E-Stop Signals**			
100	**bit name**	**Byte Name**	**Units**	**Description**
101	Platform Doors Sensors.	"xtio1_input"	Boolean	Raw signal that represent if any of the doors of the platform is open. "0" means that at least one of the doors is open OR the cable is not connected. "1" means that the cable is connected and all doors are closed.
102	Hatox E-Stop.		Boolean	Raw signal that represent if the E-Stop button on any of the Remote Controllers is pressed or if any of the Remote controllers has lost connection or has been turned off (e.g. batteries of the remote are empty). "0" means that one of the decribed events is present.
103	TOP Proximity.		Boolean	Raw signal that represent if the TOP end of rail sensor does not see the rail. "0" means the the sensor does not see the rail.
104	Bottom Terminal Switch.		Boolean	Raw signal that represent if the Bottom Terminal Switch is triggered. "0" means the the sensor is triggeres (pressed).
105	Robot E-Stop.		Boolean	Raw signal that represent if the E-Stop Button on the robot is pressed. "0" means pressed.
106	Safety Edges.		Boolean	Raw signal that represent if the Safety Strips on the bottom of robot are pressed. "0" means pressed.
107	Gripper Sensor Chain.		Boolean	Raw signal that represent if any of the grippers are unlocked. "0" means that at least one of the grippers is unlocked. "1" means that all grippers are locked.
108	Platform Rotation Sensor.	"xtio2_input"	Boolean	Raw signal that represent if the Platfomr rotation sensor is triggered. "0" means the the sensor is triggered (Platform is Rotated).

Figure 18: *Documentation*

4 Development of a DT for LIFTBOT

4.1 Choice of platform

The thesis aims to develop an accessible digital twin that showcases the robot's movement over a specific time period by utilizing the existing data collected from the robot's sensors instead of creating a new database.

For the deployment options, whether it's an online interface or a standalone file, game engines like Unity or Unreal Engine are the ideal choices to create the initial prototype of the digital twin for KEWAZO. The data from LIFTBOT sensors is stored and organized in SQLite databases, which are compatible with Unity. Unity with C# was selected for the thesis due to the supervisor's familiarity with the language.

Furthermore, Unity offers significant advantages with its ability to create standalone applications from digital twins and support for online publishing, enabling deployment as web-based applications. Previous research has demonstrated improved user experiences with such systems, and there is potential for future operations using HoloLens or virtual reality (VR) for interacting with the digital twin.

4.2 Data extraction from SQlite

The next step involves extracting the necessary data from SQlite databases. It is important to consider the specific information we want to include in the final product. This includes the basic movement of the LIFTBOT, such as going up and down the rail, rotating to deliver materials, and opening and closing the door. Additionally, real-time information such as payload, time, and height should be displayed.

Figure 18 illustrates the process of reviewing the documentation to determine which data needs to be extracted and where it can be found in the database. The «actual_position» category provides information about the system's absolute position at a given moment. The received number needs to be divided by 245,760 to convert it into meters. The «payload» category directly provides the load being transported, and the received number X can be converted to kilograms using the formula: (X/2,500 - 8.7) * 13,0835443.

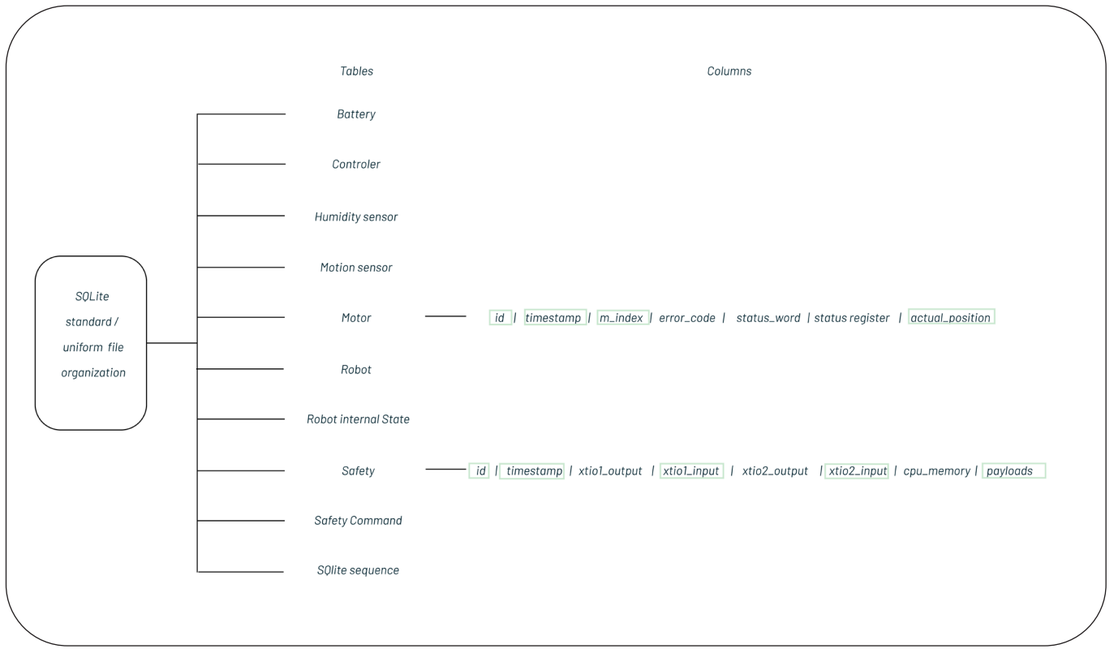


Figure 19: *Simplified structure of database*

```
public void import_database_positions(string sqliteFilePath)
{
    string connectionString = $"Data Source={sqliteFilePath};";

    try
    {
        using (IDbConnection dbConnection = new SqliteConnection(connectionString))
        {
            dbConnection.Open();

            using (IDbCommand dbCommand = dbConnection.CreateCommand())
            {
                dbCommand.CommandText = @"SELECT strftime('%Y-%m-%d %H:%M:%S', Robot.timestamp) AS second, AVG(actual_position) AS average_position
                                          FROM motor
                                          JOIN robot ON motor.id = robot.motor1_id
                                          JOIN Safety ON Safety.id = robot.safety_id
                                          WHERE m_index = 1
                                          GROUP BY second";

                using (IDataReader reader = dbCommand.ExecuteReader())
                {
                    while (reader.Read())
                    {
                        string second = reader.GetString(0);
                        float position = reader.GetFloat(1);
                        positionsList.Add(position);
                        //print(second + "POSITION: " + position);
                    }
                }
```

	second	average_position
1	2022-11-07 22:47:17	935.25
2	2022-11-07 22:47:18	1246.6
3	2022-11-07 22:47:19	1246.88235294118
4	2022-11-07 22:47:20	1247.0
5	2022-11-07 22:47:21	1246.875
6	2022-11-07 22:47:22	1247.0
7	2022-11-07 22:47:23	1246.89473684211
8	2022-11-07 22:47:24	1246.72222222222
9	2022-11-07 22:47:25	1246.86363636364
10	2022-11-07 22:47:26	1246.83333333333
11	2022-11-07 22:47:27	1246.88
12	2022-11-07 22:47:28	1246.68

Figure 20: *Query example*

The «xtio1_input» category contains multiple useful information, including the platform door sensors. If the Boolean value is 1, it indicates that the cable is connected and all doors are closed. If the Boolean value is 0, it means that either one of the doors is open or the cable is not connected.

The «xtio2_input» category provides information about the platform rotation. If the Boolean value is 1, it indicates that the raw sensor is not triggered, meaning the platform is not rotating. Conversely, if the Boolean value is 0, it means that the sensor is triggered and the platform is rotating.

When examining the provided SQlite databases from KEWAZO, we have gained an understanding of their standard organization. (*Figure 19*) In SQLite, databases are structured using tables, which serve as the primary units of organization. These tables contain structured collections of data arranged in rows and columns. The tables that are of particular interest to us are «Motor» and «Safety.» In all of the tables, there are common columns, namely «id» and «Timestamp» as the robot retrieves data at frequent intervals. In the «Motor» table, the column «m_index» is important, as it corresponds to different motors. Specifically, we will focus on «m_index == 1,» which corresponds to motor 1 and provides the «actual_positions» values. In the «Safety» table, we will extract information from the columns «xtio1_input,» «xtio2_input,» and the «payloads» column.

The code snippet provided in *Figure 20* demonstrates how to access the database:

The method takes a «sqliteFilePath» parameter, which represents the path to the SQLite database file. Then, the «connectionString» variable is created, specifying the path to the SQLite database file. The script establishes a connection to the SQLite database using the «SqliteConnection» class and opens the connection. Within the connection block, a SQL query is defined using the SELECT statement. Since timestamps may vary between tables, the query retrieves data from multiple tables (motor, robot, and Safety) by joining them based on specified conditions (id and motor 1 selection). The «AVG» function is used to calculate the average position, and the «strftime» function is used to format the timestamp value.

The query is executed using the «ExecuteReader» method of the «IDbCommand» object, which returns a data reader object (reader) containing the result set. The script then enters a loop that iterates over the rows of the result set using the «Read» method of the «IDataReader» object. Within the loop, the script retrieves the values from the columns of the current row using the «GetString» and «GetFloat» methods of the «IDataReader» object. The values are stored in the «positionsList» and «TimestampsList» lists, respectively. Finally, as the loop completes, the database connection is closed, and any encountered exceptions are caught and logged as error messages.

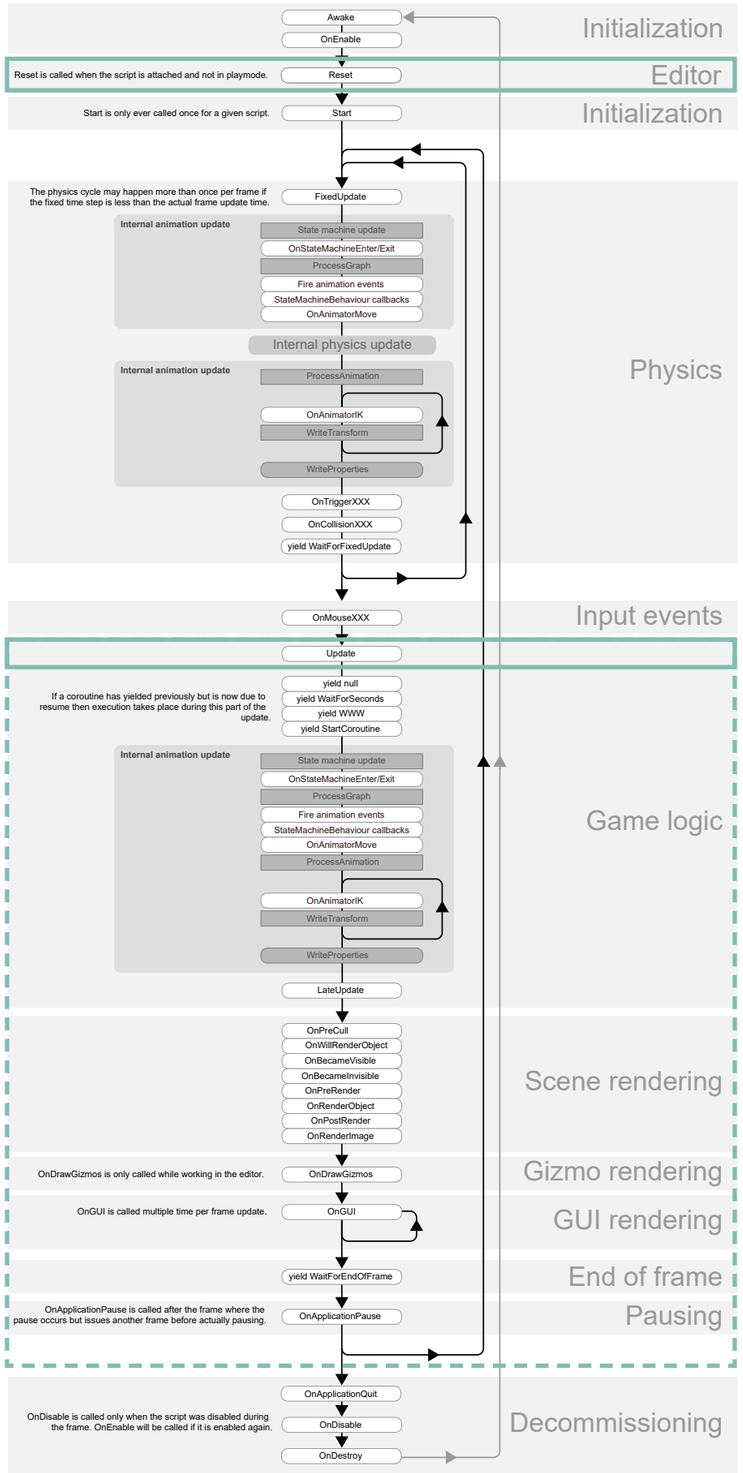


Figure 21: *Unity order of event*

4 . 3 Time frame management

To understand how timing is managed in terms of frames, it is important to first understand how a script is written in Unity. When we create a new script in Unity, we already have some code written by default. The important feature to first take into account is the «class» which serves as a container for data and methods, providing functionality to a program. Every line of code in C# must be inside a class.

```
using UnityEngine;
using System.Collections;

public class NewBehaviourScript : MonoBehaviour {

    // Use this for initialization
    void Start () {

    }

    // Update is called once per frame
    void Update () {

    }
}
```

In our example, we have named the class «NewBehaviorScript». In a public class, the name should match the code defined in Unity. A public class means that it is accessible by other classes or scripts in the game engine.

In Unity, the «void Start ()» and «void Update ()» functions are called from the «UnityEngine» library. On one hand, the Start function is called by Unity before gameplay begins (i.e., before the Update function is called for the first time), and it is an ideal place to perform any initialization. For objects that are part of a scene asset, the Start function is called on all scripts before other functions are executed. (23)

On the other hand, the Update function allows us to monitor inputs and other events regularly from a script and take appropriate actions. This means that the Update function will run periodically while our game is running. If our game runs at 60 FPS (frames per second), the Update function will run 60 times per second. However, as we mentioned earlier, we are extracting data from the database every second, so we need to implement a method such as Time.deltaTime to synchronize our data extraction with the game's frame rate.

---

(23) Technologies, Unity. "Unity - Manual: Order of Execution for Event Functions." Accessed June 20, 2023. https://docs.unity3d.com/Manual/ExecutionOrder.html.

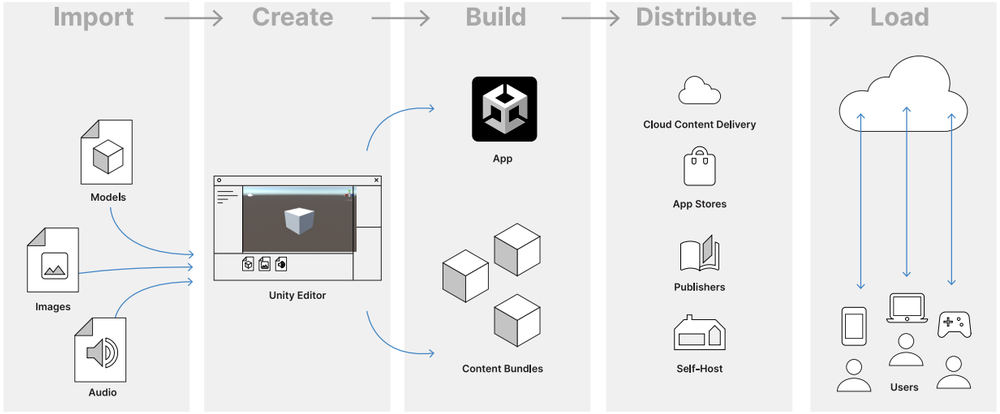


Figure 22: *Unity workflow*

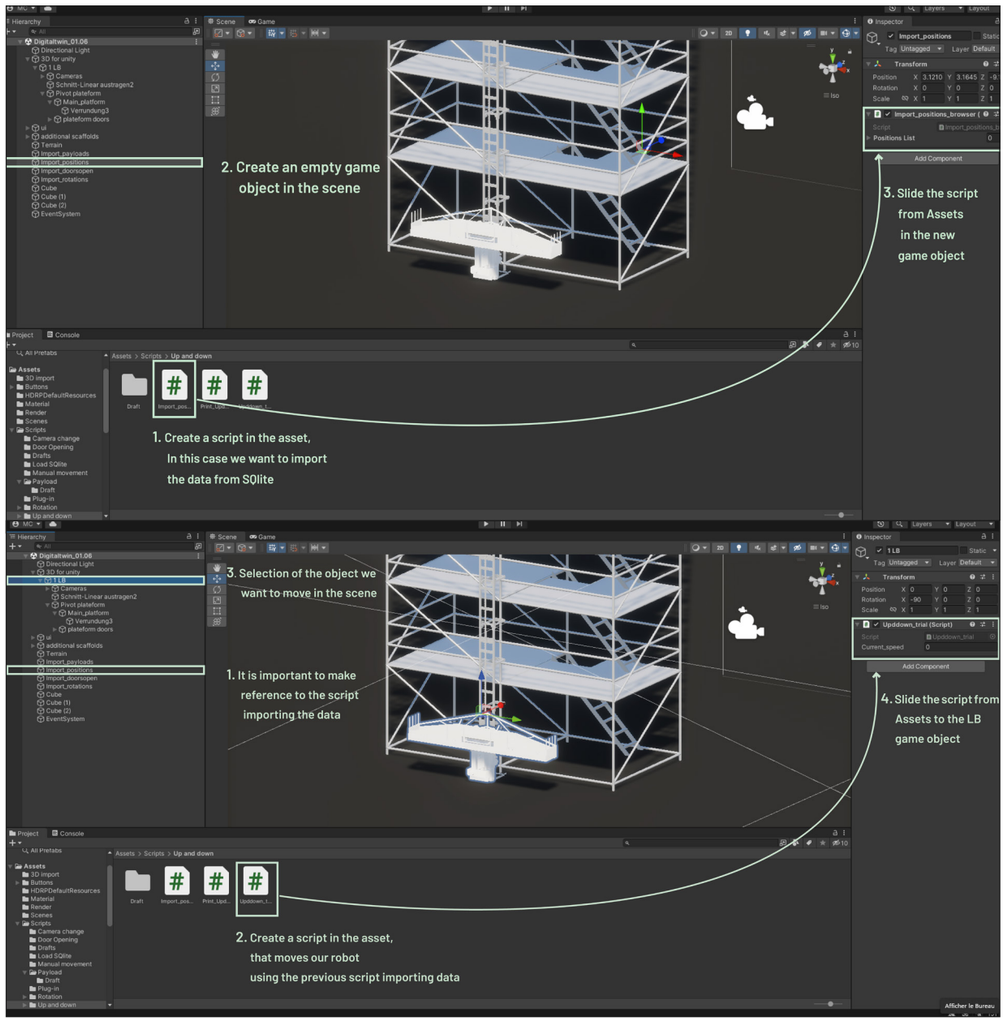


Figure 23: *Associating scripts to Game Objects*


```
private void Update()
{
    PrintPayload();
}


private void PrintPayload()
{
    currentTime += Time.deltaTime;

    ...

}
```

Time.deltaTime is defined as the time elapsed in seconds since the last frame. It utilizes the computer's internal clock, enabling us to create a frame-independent digital twin. This means that regardless of the frames per second (fps), the twin will run at a consistent speed. By adding Time.deltaTime to the currentTime variable every frame, we can accumulate the total time that has passed since the simulation began running. (24)

4 . 4 From data to movement : Workflow

In this section, we will explore the steps involved in creating a digital twin in Unity. We will explain how to connect «Scripts» implemented in C# to «Game Objects» in the 3D scene and provide a comprehensive overview of our file organization.

The first step is importing, which involves bringing source files into the Unity Editor for working with. By saving a file into the project's Assets folder, Unity imports the file and allows us to work with it in the Editor. Once the necessary assets have been imported, we can begin creating the twin. This typically involves placing assets into one or more Scenes as Game Objects and adding scripts that control user interactions. As the project development progresses, it becomes practical to organize assets into separate groups. Finally, when the project meets our satisfaction, we can proceed with building it. Building refers to the process of exporting the completed project into binary files that can be distributed and run. For example, when building for Windows, Unity generates an .EXE file along with accompanying data files. (25)

---

(24) Elnur. "Understanding Time.DeltaTime." Star Gazers (blog), March 26, 2023. https://medium.com/star-gazers/understanding-time-deltatime-6528a8c2b5c8.

(25) Technologies, Unity. "Unity - Manual: Asset Workflow." Accessed June 20, 2023. https://docs.unity3d.com/Manual/AssetWorkflow.html.

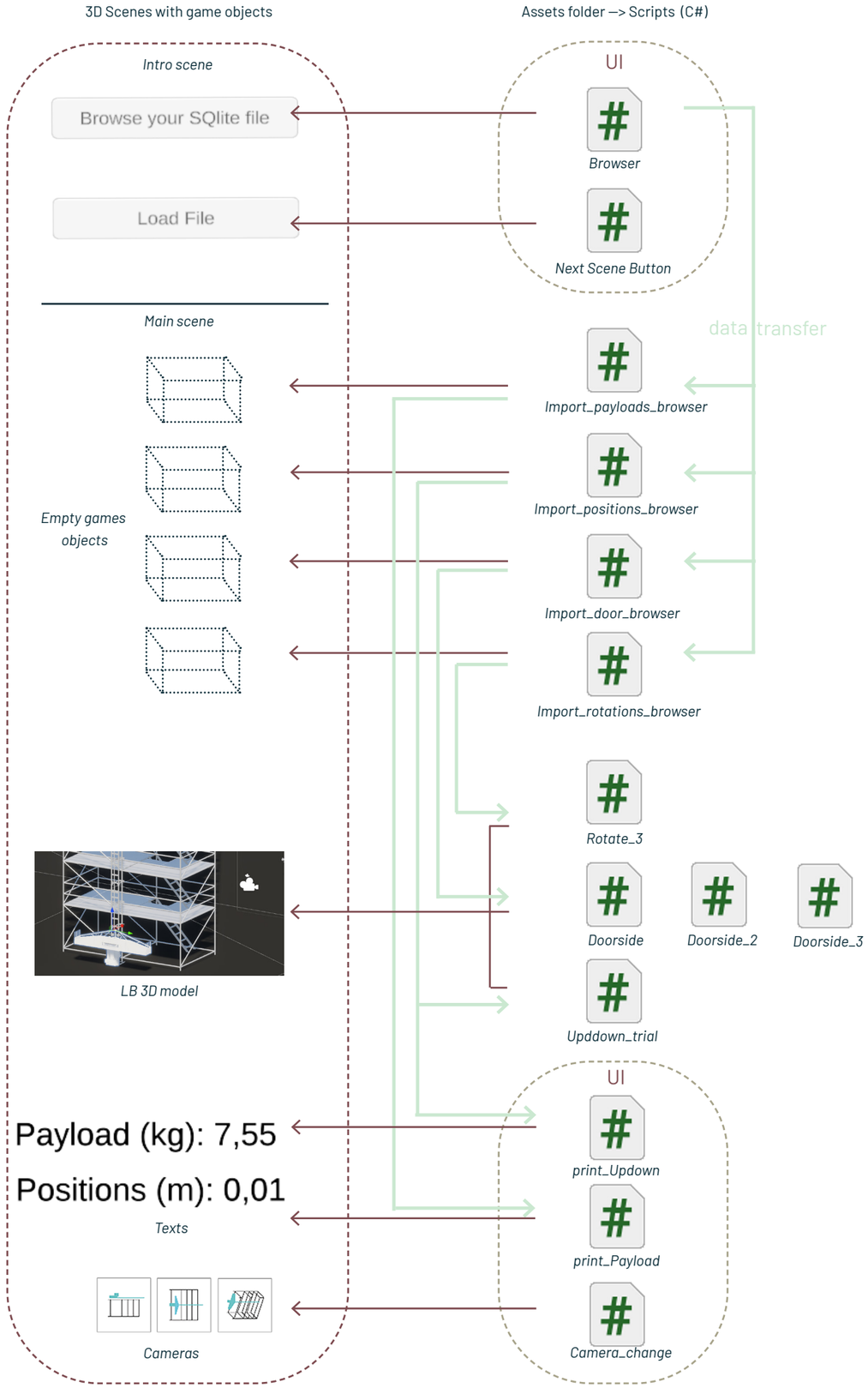


Figure 24: *Simplified organization _ scripts and game objects*

In *Figure 24*, we observe the association of C# scripts with Game Objects in different scenes of our projects. In the introduction scene, where users select their SQLite file and load it, two buttons are displayed: «Browse your SQLite file» associated with the *Browser* script, and «Load» associated with the *Next Scene Button* script. Detailed information about the structure and construction of these scripts will be provided in subsequent chapters. The loaded file remains persistent across scenes and is retrieved and loaded by other scripts.

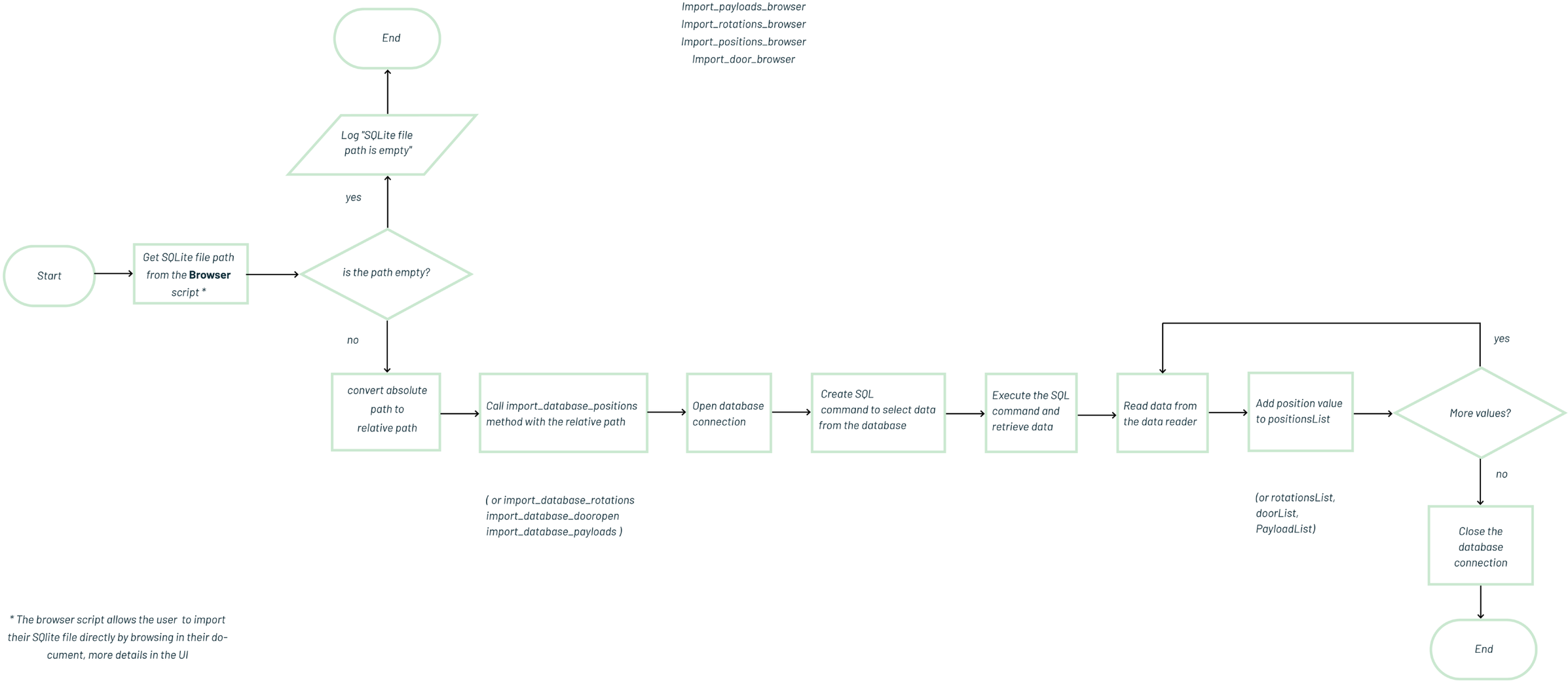
In the main scene, where we witness the movement of LIFTBOT, several scripts assist in extracting various data from the same database. After loading the file obtained from the *Browser* script, the *Import_payloads_browser* script retrieves data from the *payloads* column in the Safety table, while the *Import_positions_browser script* retrieves data from the *actual_positions* column in the *Motor* table. The *Import_door_browser* and *Import_rotations_browser* scripts retrieve data from the *xtio1_input* and *xtio2_input* column in the *Safety* table, respectively. Each of these scripts needs to be attached to a Game Object in the scene to be utilized later.

There are also scripts which directly control the movement of Game Objects, which is only possible if the data scripts are already present in the scene. For instance, the *Rotate_3* script, which controls platform rotation, needs to load data from *Import_rotations_browser* and to be attached to the platform game 3D object in the main scene. The *Upddown_trial* script, which control up and down movement of the whole system needs to load data from *Import_positions_browser* and to be attached to the 3D LIFTBOT system in the scene. Same process goes for the doors of the platform, each script controlling door movement need to be attached to each door separately and to load data from *Import_door_browser*.

The subsequent pages detail the process of writing scripts, starting from data import and extending to movement. Flowcharts are employed to enhance the reader's comprehension of the process, providing a visual representation of the steps involved.

Breaking down a project into smaller scripts in Unity offers several advantages. It promotes modularity and maintainability by focusing on specific functionalities or components, making code easier to understand, debug, and maintain. This approach also enables scalability, allowing for the addition or removal of features without impacting the entire project. Smaller scripts promote code reusability, saving development time and effort, and facilitate performance optimization by targeting specific areas for improvements. Overall, breaking down a project into smaller scripts enhances development efficiency and manageability in Unity.

Figure 25: *Flowchart, data import*



Import_payloads_browser
Import_rotations_browser
Import_positions_browser
Import_door_browser

** The browser script allows the user to import their SQlite file directly by browsing in their document, more details in the UI*

Figure 26: *Flowchart, movement up and down of the 3D object*

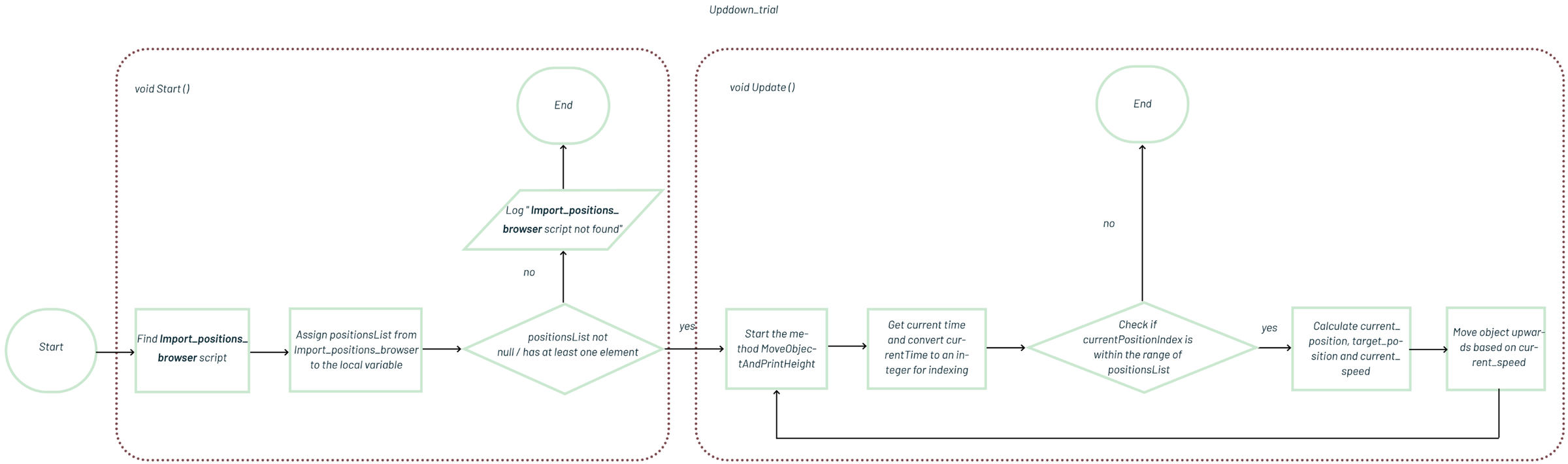


Figure 27: *Flowchart, rotation of the 3D object*

Rotate_3
similar principle for scripts: Doorside, Doorside_2, Doorside_3

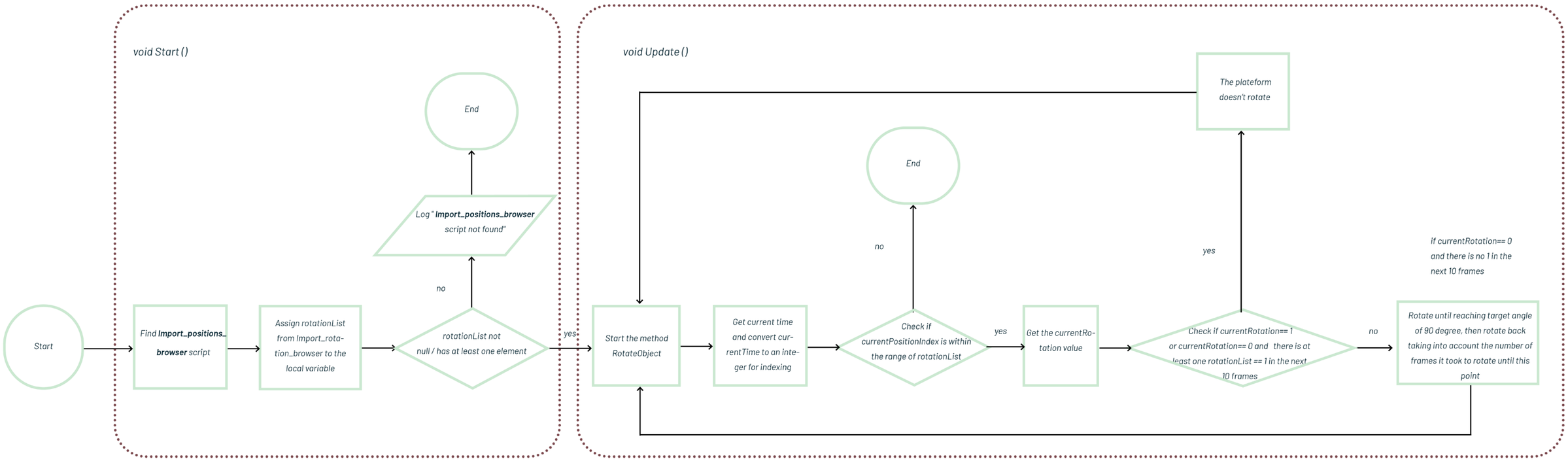


Figure 28: *Introduction Scene*

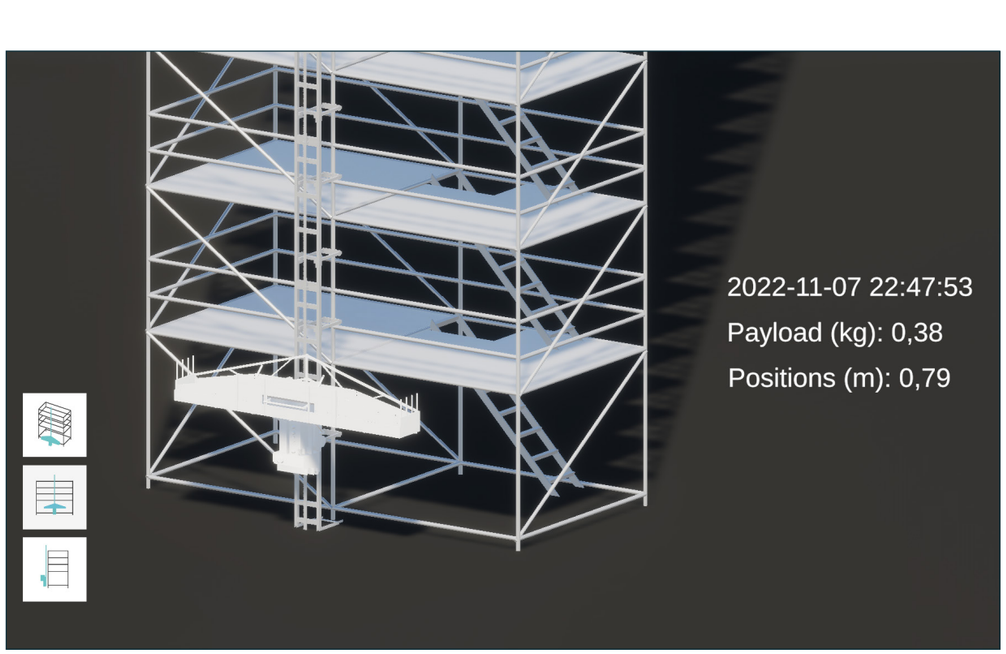


Figure 29: *Main Scene*

4.5 UI Development

Regarding the user interface (UI), specific requirements were established by the KEWAZO team. The primary objective was to display real-time information about loads and system height during the simulation. To achieve this, scripts were implemented to continuously update and modify the visible «texts» in the main scene. Additionally, a timestamp was included for reference, ensuring a comprehensive understanding of the timeline. Furthermore, the UI was designed to align with the camera trajectory, enhancing the overall user experience.

In response to feedback received, it was identified as necessary to load any SQLite file. To prevent overloading the main scene, a separate introduction scene (*Figure 28*) was created. This allowed users to comprehend the importance of loading an SQLite file before proceeding to the next step. Moreover, based on suggestions, a versatile system was developed to enable users to change the perspective of viewing the robot. This system offers multiple options such as axonometry, face-on, and side-on angles, providing flexibility and customization.

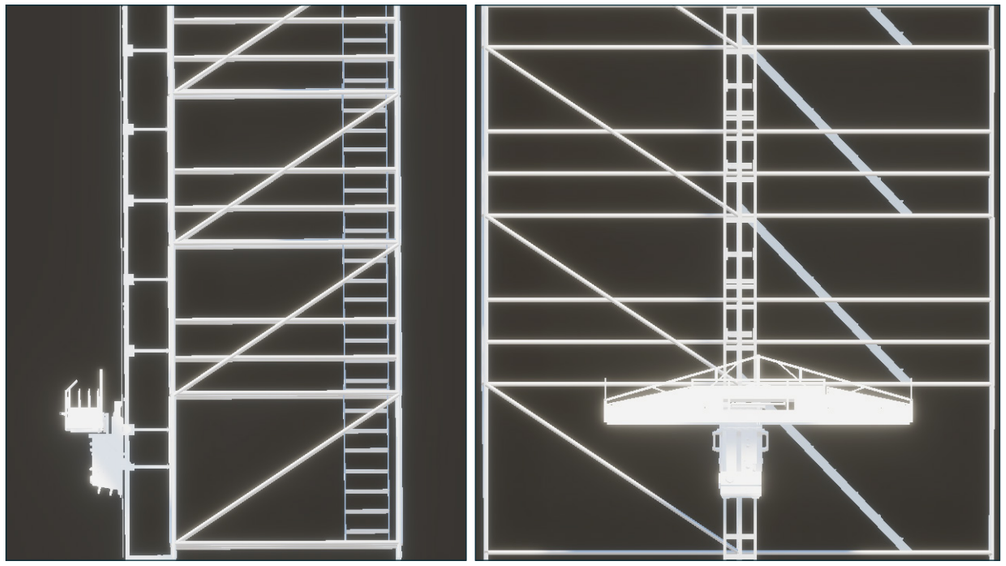


Figure 30: *Other possible views*

The subsequent pages provide a description of the script development process for loading the SQLite file and implementing the UI display.

Figure 31: *Flowchart, loading a file*

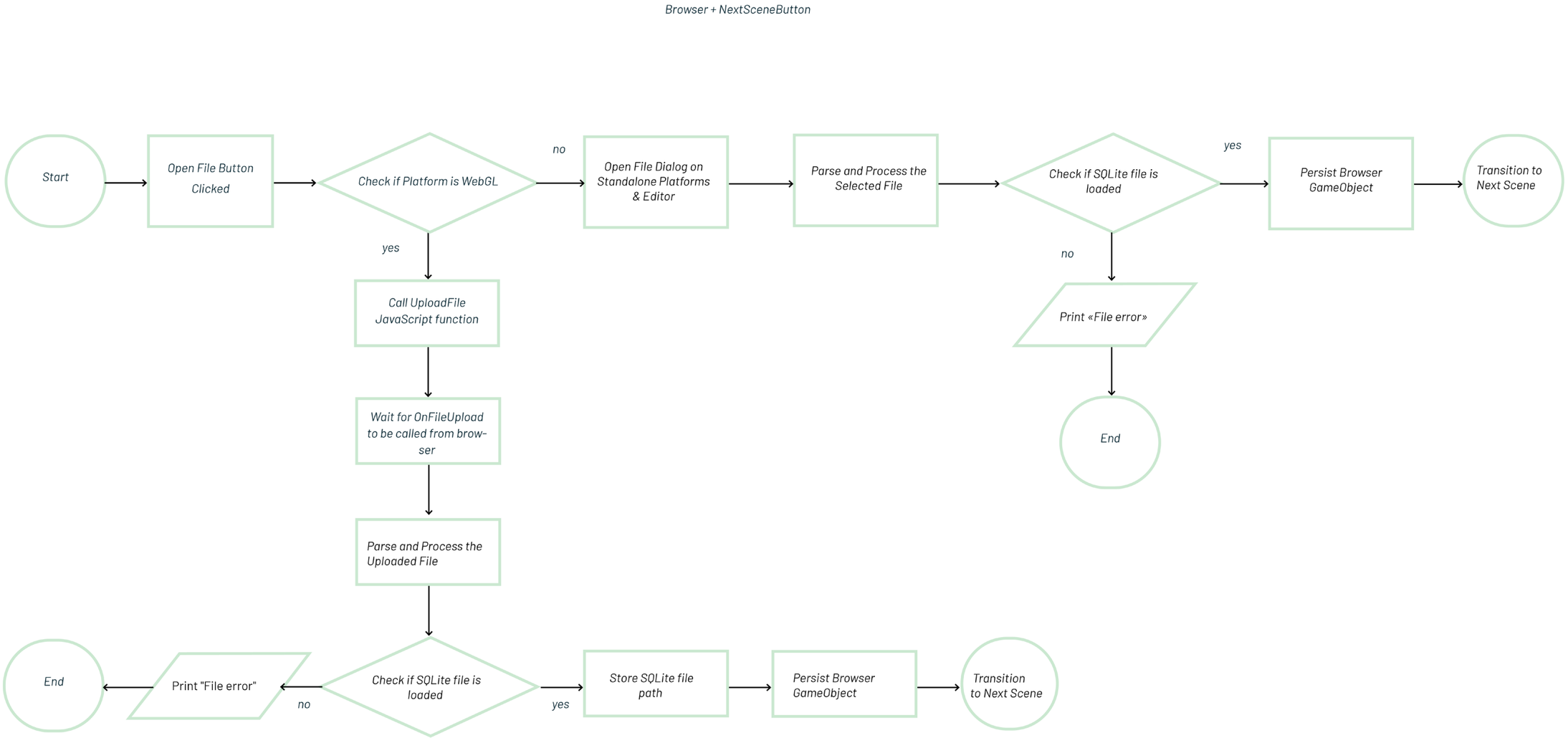
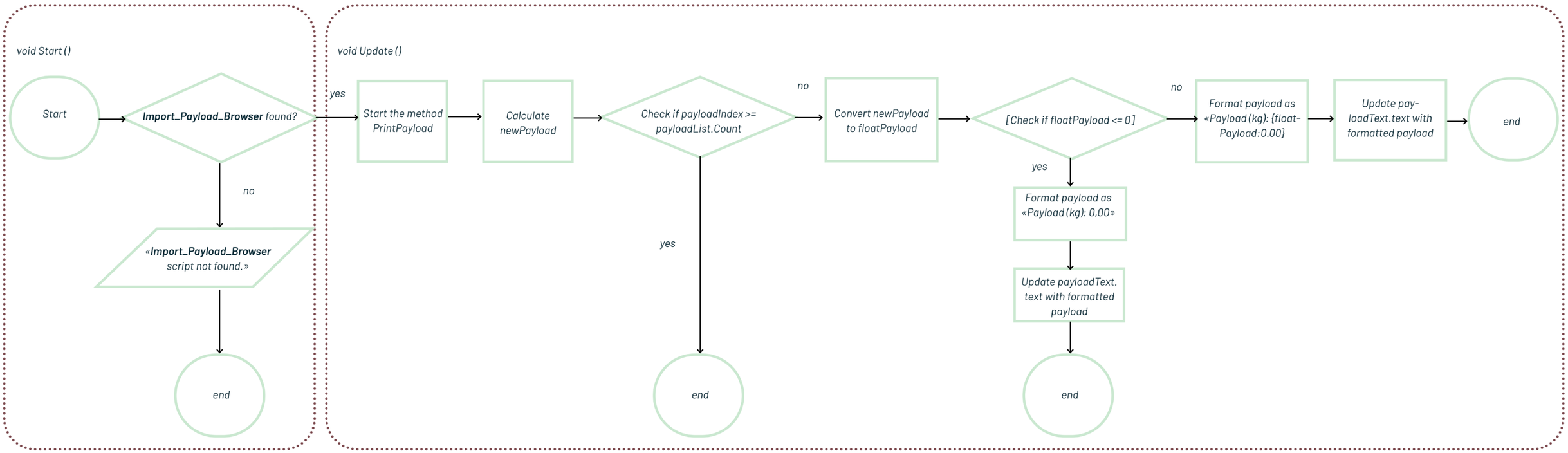


Figure 32: *Flowchart, UI, printing information*

print_Payload
print_Updown
Timer



4 . 6 Attempt at Performance prediction : multivariate linear regression

Due to the limited time available to complete the thesis, it was not possible to develop a robust computational model for predicting future performance. However, this section presents an attempt to propose a hypothetical multivariate regression model for LIFTBOT. To accomplish this, a dummy database was created in advance as part of the optimization class with Prof. Schulz.

To introduce the concept, regression analysis is a statistical method used to determine the relationships between variables and make predictions based on those relationships. It is a versatile tool that is used in various fields such as economics, finance, social science and engineering and allows to analyze and model complex relationships between variables. (26)

In the context of the improvement of the functioning of the robot Lifbot, a construction hoist allowing to lift weight up and down a rail attached to scaffolds, we are trying to predict the value of time based on other multiples other parameters. Considering the robot have a constant speed for each run, (one run being the time the robot takes to go up, then down) we decided to focus on the total time of the project. The other parameters that would allow us to predict this time would be given by the user : the total weight lifted, the height of the scaffold and the lenght of the scaffold.

The main reason why it is interesting for us to use linear regression is because we have a large dataset provided by the continuous usage of the robot, containing both values for the independent and dependent variables. (27) It is recommended to have a minimum of 10 observations or cases for each independent variable included in a linear regression model. For instance, if a model has four predictors, a sample size of at least 40 cases would be needed to ensure sufficient statistical power and reliability of the estimates. This guideline is based on the idea that having an adequate sample size helps to reduce the risk of overfitting and increases the generalizability of the model to new data. (28) The second reason for using linear regression, being the underlying linearity between the independent and dependent variables, as the time taken to transport the material is proportionally bigger as the structure gets heigher and wider, and the material transported gets heavier.

---

(26) Ng, Set Foong, Yee Chew, Pei Chng, and Kok Shien Ng. "An Insight of Linear Regression Analysis." Scientific Research Journal 15 (December 31, 2018): 1.

(27) Montgomery, Douglas C., Elizabeth A. Peck, and G. Geoffrey Vining. Introduction to Linear Regression Analysis. 5. edition. Hoboken, NJ: Wiley, 2012.

(28) El Aissaoui, Ouafae, Yasser El Alami El Madani, Lahcen Oughdir, Ahmed Dakkak, and Youssouf El Allioui. "A Multiple Linear Regression-Based Approach to Predict Student Performance." In Advanced Intelligent Systems for Sustainable Development (AI2SD'2019), edited by Mostafa Ezziyyani, 1102:9–23. Advances in Intelligent Systems and Computing. Cham: Springer International Publishing, 2020.

Theory

Univariate regression analysis uses one independent variable to analyze the relationship between that variable and a dependent variable. The result is an equation that represents the linear relationship between the two variables:

$$y= \beta_0 +\beta_1 x_1+ \varepsilon$$

where y is the dependent variable; β_0 is the value of y when x_1 equals to 0, also called an intercept, x_j is one dependent variable, β_1 is the regression coefficient and ε is a random error term that represents the difference in the linear model and a particular observed value for y.

Multivariate regression analysis, on the other hand, uses more than one independent variable to account for variations in the dependent variable. (29) The model for multivariate regression analysis is formulated :

$$y= \beta_0 +\beta_1 x_1+ ...+ \beta_n x_n+ \varepsilon$$

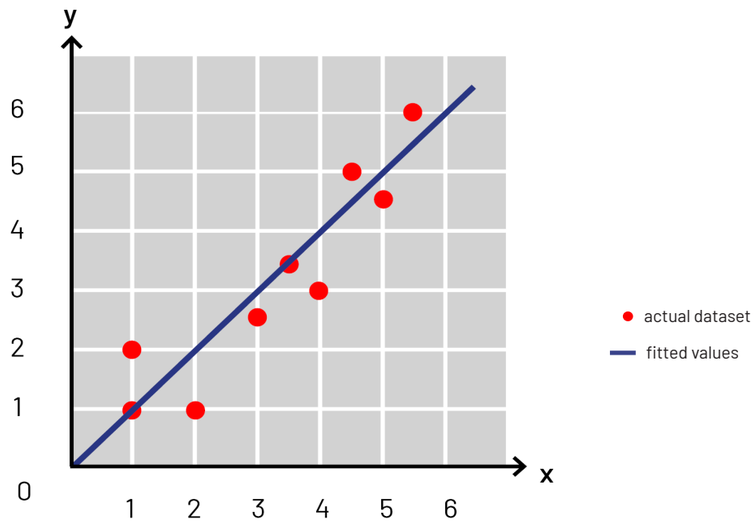
Here we can see we have multiple dependent variables and their regression coefficients. In linear regression, the dependent variable is the variable we want to predict, while the independent variable(s) is the variable(s) we use to make that prediction. Because the independent variable(s) can help us predict the value of the dependent variable, it is often referred to as the predictor variable. (26) In our previous example, our predictor variable is the time we are trying to predict, while the parameters of height, lenght and weight transported are the one helping us reaching that goal.

The purpose is to identify the linear relationship between the independent variables and the dependent variable, and to estimate the coefficients of the linear model. This can be done using the least squares method, which minimizes the sum of the squared errors between the predicted values and the actual values in the dataset. Once the coefficients of the linear model have been estimated, they can be used to make predictions on new data points.

---

(26) Ng, Set Foong, Yee Chew, Pei Chng, and Kok Shien Ng. "An Insight of Linear Regression Analysis." Scientific Research Journal 15 (December 31, 2018): 1.

(29) Kaya Uyanık, Gülden, and Neşe Güler. "A Study on Multiple Linear Regression Analysis." Procedia - Social and Behavioral Sciences 106 (December 1, 2013): 234–40.



y	x	$\overline{y}$	$\overline{y}$ - y
1	1	1	1-1 = 0
1	2	2	2-1 =0
2	1	1	-1
2.5	3	3	0.5
3	4	4	1
4	3.5	3.5	-0.5
4.5	5	5	0.5
5	4.5	4.5	-0.5
6	5.5	5.5	-0.5

Figure 33: *Graphical representation of linear regression data.*

Cost function

The red points in this graphic example represents data points from the dataset, and the blue curve represents the prediction or the best fitted line, which can be written, as previously mentioned:

$$y = \beta x + \beta_0$$

To check how good of a fit is the blue curve, it is necessary to measure the error which can be defined by $y - \hat{y}$ with is the difference between the real value of y provided by the dataset and the $\hat{y}$ which is predicted value given by our fitted curve. As it is necessary to sum out all the values without negatives, the sum of the total error is then squared. The result is the cost function which can be written as:

$$\text{Cost function} = (y - \hat{y})^2$$

We can then replace the formula of $\hat{y}$ in the cost / loss function:

$$L = ((\beta x + \beta_0) - y)^2$$
$$L = f(\beta, \beta_0)$$

Considering the size of the dataset and the number of points have also an influence, we can rewrite the cost function:

$$L_{total} = \frac{1}{N}\sum_{i=1}^{N}((\beta x + \beta_0) - y_i)^2$$

The goodness of fit of the curve is measured by this loss function. The lower this value is, the better is the curve fitted. (30) The process of gradient descend will then allow to minimize the loss function, progressively update the each regression coefficients and get the best fitted curve.

(30) Udemy. "Deep Learning Foundation : Linear Regression and Statistics." Accessed March 20, 2023. https://www.udemy.com/course/linear-regression-in-python-statistics-and-coding/.

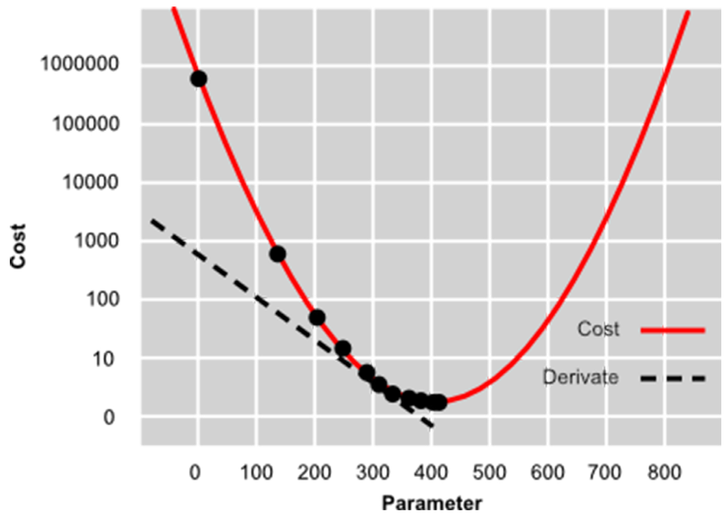


***Figure 34**: Graphical representation of the processus of gradient descend*
Source: quicktomaster.com

Gradient descent

The gradient descent is an iterative solution that incrementally steps towards an optimal solution and is used in varieties of situations. This process has to be implemented for each parameter of the loss function. It usually starts with an initial guess and then improves it one at the time, to reach an optimal solution. In order to reach the minima of a function, we use the derivative as its value indicate how far we are to the optimized solution. (31) The magnitude of the derivative is proportional to how big of a step we should take towards the minimim. A relatively large value for the derivative corresponds to a steep slope and suggest that we are relatively far from the bottom of the curve, while a small value suggests we are relatively close to the bottom of the curve. The sign (+/-) indicates us which direction: a negative derivative tells us, it is necessary to take a step to the right to get to the minima, while a positive derivative indicates we need to take a step to the left. (31)

The derivative in relationship to the intercept β_0 can be written in the following way:

$$\frac{\partial L}{\partial \beta_0} = \frac{\partial}{\partial \beta_0}(\sum_{i=1}^{N}((\beta x + \beta_0) - y_i)^2)$$

Using the chain rule, we can then rewrite it in the following way:

$$\frac{\partial}{\partial \beta_0} = 2\sum_{i=1}^{N}((\beta x + \beta_0) - y_i)$$

The derivative in relationship to the parameter β can also be written:

$$\frac{\partial L}{\partial \beta} = \frac{\partial}{\partial \beta}(\sum_{i=1}^{N}((\beta x + \beta_0) - y_i)^2)$$

$$\frac{\partial L}{\partial \beta} = 2\sum_{i=1}^{N} x_i((\beta x + \beta_0) - y_i)$$

After getting the the derivative equations, it is now possible to calculate the step size from one point to another on the curve, which is written:

$$\text{Step size} = \text{Derivative} \times \text{Learning rate}$$

The learning rate being a value that prevents us taking steps that are too big and skiping past the lowest point in the curve. Typically for Gradient descent, the learning rate is determined automatically: it starts relatively large and gets smaller with every steps taken. (31)

(31) PhD, Josh Starmer. The StatQuest Illustrated Guide To Machine Learning. Independently published, 2022.

As we want to get closer to the optimal value for the intercept (β_0 in our case), the formula will be written:

$$\text{New intercept} = \text{Current intercept} - \text{Step size}$$

Then, we repeat those different steps updating the intercept after each iteration until the step size is close to 0, or we take the maximum number of steps, which is often set to 1000 iterations. Those steps are also repeated for β.

Goodness of fit

R-squared (R^2): measures the squared correlation between the actual outcome values and the values predicted by the model. The higher the adjusted R^2, the better the model. An R^2 value of 1 indicates a perfect fit, while values closer to 0 indicate a poor fit. (28)

$$R^2 = \frac{Model\ sum\ of\ squared}{Total\ sum\ of\ squared} = \frac{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Where $\hat{y}_i$ is the predicted outcome, y_i is the observed outcome and $\bar{y}$ is the average of observed outcomes.

Other measures of error include the Root Mean Squared Error (RMSE), which measures the average magnitude error made by the model while predicting an observation's outcome. It's the square root of the average of squared residuals. Residuals are the difference between the actual values and the predicted values. The lower the RMSE, the better the model. Mean Absolute Error (MAE) is a measure of average absolute differences between observed and predicted outcomes. The lower the MAE, the better the model. (28)

Method

The research data presented in the next page, under the csv format is composed of a the columns Height, Lenght which corresponds to the height and lenght of the structure in meters, the column Time, which corresponds to the total time of the project in hours, as well as the column Weight which corresponds to the total weight transported in tons.

(28) El Aissaoui, Ouafae, Yasser El Alami El Madani, Lahcen Oughdir, Ahmed Dakkak, and Youssouf El Allioui. "A Multiple Linear Regression-Based Approach to Predict Student Performance." In Advanced Intelligent Systems for Sustainable Development (AI2SD'2019), edited by Mostafa Ezziyyani, 1102:9–23. Advances in Intelligent Systems and Computing. Cham: Springer International Publishing, 2020.

Sample

```
Project,Height,Lenght,Time,Weight
1, 32, 15, 42, 36.000
2, 55, 22, 137, 40.572
3, 25, 7.5, 32, 16.250
4, 38, 12, 41, 25.240
5, 39, 14, 50, 27.536
6, 20, 6, 15, 12.660
7, 35, 12, 44,32.922
8, 49, 15, 89, 37.750
9, 24, 9, 22, 18.205
10, 42, 14, 65, 33.750
11, 48, 12, 66, 28.750
12, 30, 12.5, 48, 33.428
13, 25, 8.5, 36, 20.510
14, 15, 6, 25, 15.620
15, 52, 18.5, 120, 42.600
16, 45, 14, 74, 35.400
17, 42, 13, 58, 31.250
18, 20, 7, 34, 17.520
19, 24, 11, 33, 18.600
```

b) Programming

Based on the theory previously presented, we are going to go through the coding of the algorithm, underline the most important points and provide an analysis of the results. The first step of the coding involves importing the different libraries necessary for the process:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib as mpl
mpl.use('qt5agg')
```

We use Pandas for data analysis and extracting our data from the csv file, NumPy to deal with array-like objects and finally Matplotlib to plot our values. We are then importing both the time column as our Y, which is our dependent variable and the columns height, lenght and weight, as our X, which are our independent variables.

```
#Multivariate regression analysis

X = df [['Height', 'Lenght', 'Weight']]
Y = df.Time
```

The second step is to initialize the values of regression coefficients with random numbers, but also the values of Y current, «pre-cost» function that is later on going to be updated with the cost function, and learning rate for the Gradient descent.

```
# Y = mX+c
# Initializing m and c with random numbers
k = X.shape[1] # return the total number of parameters in X
m_current = np.zeros(k)
c_current=0
N = X.shape [0] # return the total number of values
Y_current = 0
print (X)

# cost function
pre_cost = sum ((Y-Y_current)**2) /2

# learning_rate
learning_rate = (1 / pre_cost)
```

As an additional step, we ask the user to input values for height, lenght and weight, that are going to be plotted afterwards, in order to find the time values for those specific input. np.array (user_weight) creates a numpy array from the user_weight variable, and reshape(1, -1) changes the shape of the numpy array to have one row and as many columns as necessary (-1). This is done to ensure that the array has the right shape to perform matrix multiplication with the m_current variable, which has a shape of (3,)(3 features in X: Height,Weight and Lenght). The shape of user_weight_np after this line is (1, 1), which is compatible with the shape of m_current.

The next step is the start of the Gradient descent, we use a for loop and give a certain number of iterations that can then be adjusted afterwards.We set the formula for the partial derivatives in regards to m and c (since Y= mX + c in the script) and through the process of Gradient descent, updating those values with the learning rate we previously defined. With the new values of m and c, Y and the value of the cost function are updated.

```
for i in range (25):

      m_grad = ((2/N) * (np.dot(X.T,(Y_current- Y))))

      c_grad = (2/N) * sum (Y_current-Y)


      # Multiply the gradient with the learning rate
      # Update m and c by substracting the derivative from initial numbers

      m_update = m_current - m_grad * learning_rate
      c_update = c_current - c_grad * learning_rate

      #We need to sum all the X column values
      Y_update = (m_update * X). sum (axis = 1) + c_update



      #Cost function
      cost = sum ((Y_update - Y)**2) / N

      print ('cost', i, cost)
```


We set a new condition, where if the new cost value is lower than the previous cost, we then update the values of m, c, the cost and plot the values for each iterations.

```
#Verify that if the cost value gets lower, then we are on the right track

if pre_cost > cost:
      m_current = m_update
      c_current = c_update
      pre_cost = cost

      # Trying to plot our values for only cost (to see how the cost influence the performance

      fig = plt.figure(figsize=(12,12))

      fig.subplots_adjust(left=0.1, bottom=0.1, right=1, top=0.9, wspace=0.4, hspace=0.4)

      # creating subplots
      ax0 = plt.subplot2grid((3,6), (0,0), rowspan=1, colspan=2)
      ax1 = plt.subplot2grid((3,6), (1,0), rowspan=1, colspan=2)
      ax2 = plt.subplot2grid((3,6), (2,0), rowspan=1, colspan=2)
      ax3 = plt.subplot2grid((3,6), (0,2), rowspan=2, colspan=4, projection='3d')
```

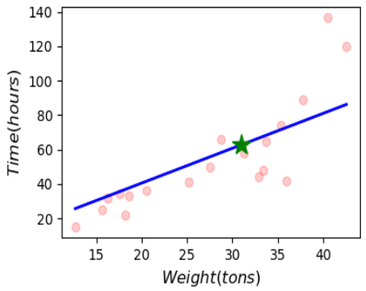
In order to get a linear curve, it is also necessary to sort the values before plotting:

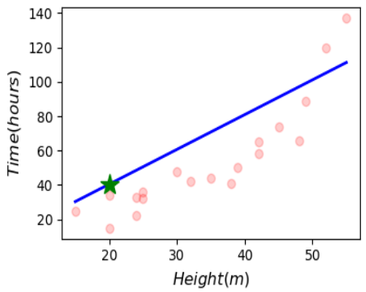
```
#sorting the values for weight before plotting
sorted_x0 = np.sort(df.Weight, axis=None)
sorted_x0 = sorted_x0[:, np.newaxis] # add a new axis
sorted_y0 = (m_update * sorted_x0).sum(axis=1) + c_update

#sorting the values for height before plotting
sorted_x1 = np.sort(df.Height, axis=None)
sorted_x1 = sorted_x1[:, np.newaxis] # add a new axis
sorted_y1 = (m_update * sorted_x1).sum(axis=1) + c_update

#sorting the values for height before plotting
sorted_x2 = np.sort(df.Lenght, axis=None)
sorted_x2 = sorted_x2[:, np.newaxis] # add a new axis
sorted_y2 = (m_update * sorted_x2).sum(axis=1) + c_update
```

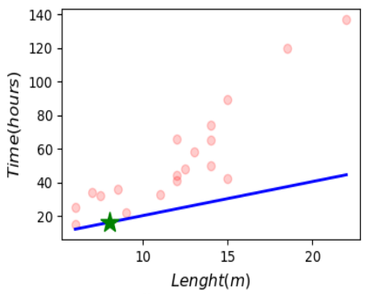
Finally, to show the results we plot each of the independent variables in relationship to the dependent variable of time, to show how each one of them affect the total time globally.
We also plotted the results in 3D to see the results in regards to Time, Weight and Height.

After the end of the iterations and the process of Gradient descent, the coefficients of regression as well as the optimal time for the user inputs are printed.The final R2 is also calculated, as a way for us to analyze how good of a model we have.

```
# calculate R2 (coefficient of determination) in order to find out how solid our model is
def r2(y_true, y_pred):
    y_bar = np.mean(y_true)
    ss_tot = np.sum((y_true - y_bar)**2)
    ss_res = np.sum((y_true - y_pred)**2)
    r2 = 1 - (ss_res/ss_tot)
    return r2
```







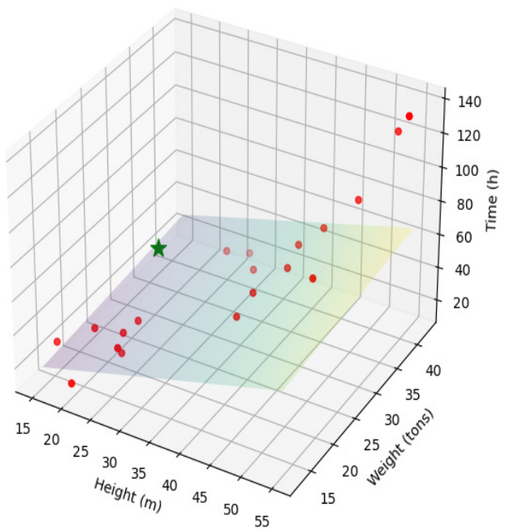


***Figure 35**: Results of the multivarite regression process*

Interpretation of the results

The graph shows us correctly the dataset and the fitted values for each variable. We can get a sense of how each of these variable impact the final Time. We also can see the input value of the user.

The printed values shows us how the progression of the iterations and the progression of the cost error, which allows us to re-adjust the learning rate to reach better final values.
The final R^2 of 0.7058 indicates that the model explains 71% of the variability in the data around its mean. In other words, there is still 29% of the variability that is not explained by the model. R^2 can be improved with various techniques such as normalization of the data or inscrease of the dataset. (27) Below is an example of the final results.

```
please enter the value of the total load you want to carry (in tons): 31
please enter the height of the structure(m): 20
please enter the lenght of the structure(m): 8
cost 0 3072.150234233784
cost 1 2332.318205601131
cost 2 1706.5565456809873
cost 3 1194.8652544733536
cost 4 797.2443319782294
cost 5 513.6937781956158
cost 6 344.213593125512
cost 7 288.8037767679182
cost 8 347.46432912283416
cost 9 289.5366654113468
cost 10 288.8257339663405
cost 11 288.8054591711011
cost 12 288.8039398750698
cost 13 288.80379302730177
cost 14 288.8037783933431
cost 15 288.80377693045546
cost 16 288.8037767841719
cost 17 288.80377676954356
cost 18 288.8037767680807
cost 19 288.8037767679346
cost 20 288.8037767679196
cost 21 288.8037767679184
cost 22 288.80377676791824
cost 23 288.8037767679181
m_current = [0.94843665 0.32954939 0.74460114]
c_current = 0.023277981803493254
optimal time (h) = [44.71104122]
R2 score: 0.7058
```

(27) Montgomery, Douglas C., Elizabeth A. Peck, and G. Geoffrey Vining. Introduction to Linear Regression Analysis. 5. edition. Hobo-

5 Conclusion

At the beginning of our research, we asked ourselves how to develop and implement an adaptable digital twin workflow in the construction industry. We acknowledged that there are multiple workflows and no standardized method, considering the diverse range of robots and unpredictable nature of construction sites.

To materialize this model, our first task was to comprehend the various workflows available on different platforms and determine which one would be relevant in our case. We delved into the concept of data management and recognized its significance, particularly in handling the complexity of construction sites. Important concepts such as multiple sources of information/sensors on-site, semantic relationships, and standardized transfer were emphasized in developing a robust model. We also outlined the common steps applicable to all digital twin developments for robots, positioning my own development work within the overall framework. Furthermore, we explored different models of performance prediction, including data-driven, physics-driven, and hybrid models.

With this theoretical framework established, we proceeded to develop a model for our specific case study. Leveraging the existing data from the company, our work involved choosing a platform and creating the 3D twin. Considering the company's requirements, Unity 3D emerged as the most suitable choice for creating an accessible platform, whether online or standalone, with a user-friendly interface. We successfully extracted the necessary information from the SQlite database and translated it into a real-time 3D representation by combining scripts with game objects. The flexibility of Unity's architecture, composed of interconnected scripts, allowed for a versatile final model. Additionally, we proposed a simple yet intuitive user interface, fulfilling the company's requirements. We also made an attempt at performance prediction that could serve as a foundation for future development.

5.1 Limitation and future improvements

Some difficulties were encountered in developing a more comprehensive digital twin within the given timeframe. As mentioned earlier, the feature of performance prediction was researched but not added to the final DT. Another issue arose from technical difficulties related to platform compatibility, as we were unable to resolve the problem of online deployment due to the lack of support for the SQlite plug-in in WebGL.

Another aspect that could be improved is the precision of data received from the sensors. To create a more accurate twin for LIFTBOT, it would be beneficial to integrate multiple additional sources of data from various sensing technologies to capture comprehensive information about the construction robot and its environment. For example, in terms of rotation, the current data indicates that the platform is rotating, but it was not possible to determine the extent of the rotation. The same issue applies to the door opening, where it was not possible to identify which door is open and to what extent.

In the context of machine learning, having a more diverse range of sensor sources can help establish meaningful connections between different concepts, such as climate conditions, which would be highly valuable for predicting future performance.

The user interface (UI) can also be significantly improved. Currently, all levels have the same height, which is not representative of reality. It would be beneficial to allow users to enter the number of levels and their respective heights, which would then be generated in the main scene.

5.2 Contribution to knowledge

In the end, visualizing a large amount of data allows for easy comprehension by anyone observing it. The benefits of such a visualized Digital Twin are numerous. Firstly, it enables clients to virtually and remotely monitor the operation of LIFTBOT in high fidelity. Site managers, supervisors, or their delegates can analyze project execution, identify bottlenecks, improper usage, safety guideline violations, and more. This information informs subsequent projects, leading to continuous improvements in the operational process.

Additionally, this DT for robots in construction contributes to refining the system and finding solutions by identifying anomalies within an operational context. It helps engineers intuitively understand the problem and determine the causes or factors contributing to it in various real and projected scenarios.

In summary, this master's thesis managed to offer a thorough exploration of the varied workflows encompassing the creation of digital twins for on-site construction robots and beyond. With limited online guidance and resources on this subject, a key objective was to elucidate the distinct steps involved. At the end we also managed to deliver a industrial digital twin, answering the requirement of the company to a certain extent.

References

(1) Wang, Xi, Ci-Jyun Liang, Carol C. Menassa, and Vineet R. Kamat. "Interactive and Immersive Process-Level Digital Twin for Collaborative Human–Robot Construction Work." Journal of Computing in Civil Engineering 35, no. 6 (November 1, 2021): 04021023.

(2) Dawod, Mohamed, and Sean Hanna. "BIM-Assisted Object Recognition for the on-Site Autonomous Robotic Assembly of Discrete Structures." Construction Robotics 3, no. 1 (December 1, 2019): 69–81.

(3) Bruno Daniotti. "Digital Transformation in the Construction Sector: From BIM to Digital Twin." In Digital Transformation - Towards New Frontiers and Business Opportunities, edited by Alberto Pavan, translated by Claudio Mirarchi, Ch. 6. Rijeka: IntechOpen, 2022.

(4) Ravi, Kaushik Selva Dhanush, Ming Shan Ng, Jesús Ibáñez, and Daniel Hall. Real-Time Digital Twin of On-Site Robotic Construction Processes in Mixed Reality, 2021.

(5) Liu, Mengnan, Fang Shuiliang, Huiyue Dong, and Cunzhi Xu. "Review of Digital Twin about Concepts, Technologies, and Industrial Applications." Journal of Manufacturing Systems 58 (July 1, 2020).

(6) Grieves, Michael, and John Vickers. "Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems," 85–113, 2017.

(7) "What Is a Digital Twin? | IBM." Accessed June 10, 2023. https://www.ibm.com/topics/what-is-a-digital-twin.

(8) Sepasgozar, Samad M. E., Ayaz Ahmad Khan, Kai Smith, Juan Garzon Romero, Xiaohan Shen, Sara Shirowzhan, Heng Li, and Faham Tahmasebinia. "BIM and Digital Twin for Developing Convergence Technologies as Future of Digital Construction." Buildings 13, no. 2 (February 2023): 441.

(9) Douthwaite, James, B. Lesage, Mario Gleirscher, Radu Calinescu, Jonathan Aitken, Rob Alexander, and James Law. "A Modular Digital Twinning Framework for Safety Assurance of Collaborative Robotics." Frontiers in Robotics and AI 8 (December 1, 2021)

(10) Jiang, Yishuo, Ming Li, Daqiang Guo, Brave Wu, Ray Zhong, and George Q. Huang. "Digital Twin-Enabled Smart Modular Integrated Construction System for on-Site Assembly." Computers in Industry 136 (April 1, 2022): 103594.

(11) Kamat, Vineet. "Real-Time Process-Level Digital Twin for Collaborative Human-Robot Construction Work." edited by Hisashi» «Osumi "Furuya, Hiroshi", "Tateyama, Kazuyoshi," 1528–35. International Association for Automation and Robotics in Construction (IAARC), 2020.

(12) Qian, Wei, Zeyang Xia, Jing Xiong, Yangzhou Gan, Yangchao Guo, Shaokui Weng, Hao Deng, Ying Hu, and Jianwei Zhang. "Manipulation Task Simulation Using ROS and Gazebo," 2014.

(13) Phanden, Rakesh, Priavrat Sharma, and Anubhav Dubey. "A Review on Simulation in Digital Twin for Aerospace, Manufacturing and Robotics." Materials Today: Proceedings 38 (July 1, 2020).

(14) Platt, Jonathan, and Kenneth Ricks. "Comparative Analysis of ROS-Unity3D and ROS-Gazebo for Mobile Ground Robot Simulation." Journal of Intelligent & Robotic Systems 106 (December 20, 2022).

(15) Zhang, Ye, A. Meina, Xuhao Lin, Kun Zhang, and Zhen Xu. "Digital Twin in Computational Design and Robotic Construction of Wooden Architecture." Advances in Civil Engineering 2021 (April 2, 2021): 1–14.

(16) Zhang M, Tao F, Huang B et al. Digital twin data: methods and key technologies [version 2; peer review: 4 approved]. Digital Twin 2022, 1:2 (https://doi.org/10.12688/digitaltwin.17467.2)

(17) Boje, Calin, Annie Guerriero, Sylvain Kubicki, and Yacine Rezgui. "Towards a Semantic Construction Digital Twin: Directions for Future Research." Automation in Construction 114 (June 1, 2020): 103179.

(18) https://jpt.spe.org/twa/author/hector-klie. "A Tale of Two Approaches: Physics-Based vs. Data-Driven Models." JPT, May 3, 2021.

(19) Arafet, K.; Berlanga, R. Digital Twins in Solar Farms: An Approach through Time Series and Deep Learning. Algorithms 2021, 14, 156.

(20) Maulud, Dastan, and Adnan Mohsin Abdulazeez. "A Review on Linear Regression Comprehensive in Machine Learning." Journal of Applied Science and Technology Trends 1 (December 31, 2020): 140–47.

(21) Ritto, T. G., and F. A. Rochinha. "Digital Twin, Physics-Based Model, and Machine Learning Applied to Damage Detection in Structures." Mechanical Systems and Signal Processing 155 (June 2021): 107614.

(22) Pawar, Suraj, Shady E. Ahmed, Omer San, and Adil Rasheed. "Hybrid Analysis and Modeling for next Generation of Digital Twins." Journal of Physics: Conference Series 2018, no. 1 (September 2021): 012031.

(23) Technologies, Unity. "Unity - Manual: Order of Execution for Event Functions." Accessed June 20, 2023. https://docs.unity3d.com/Manual/ExecutionOrder.html.

(24) Elnur. "Understanding Time.DeltaTime." Star Gazers (blog), March 26, 2023. https://medium.com/star-gazers/understanding-time-deltatime-6528a8c2b5c8.

(25) Technologies, Unity. "Unity - Manual: Asset Workflow." Accessed June 20, 2023. https://docs.unity3d.com/Manual/AssetWorkflow.html.

(26) Ng, Set Foong, Yee Chew, Pei Chng, and Kok Shien Ng. "An Insight of Linear Regression Analysis." Scientific Research Journal 15 (December 31, 2018): 1.

(27) Montgomery, Douglas C., Elizabeth A. Peck, and G. Geoffrey Vining. Introduction to Linear Regression Analysis. 5. edition. Hoboken, NJ: Wiley, 2012.

(28) El Aissaoui, Ouafae, Yasser El Alami El Madani, Lahcen Oughdir, Ahmed Dakkak, and Youssouf El Allioui. "A Multiple Linear Regression-Based Approach to Predict Student Performance." In Advanced Intelligent Systems for Sustainable Development (AI2SD'2019), edited by Mostafa Ezziyyani, 1102:9–23. Advances in Intelligent Systems and Computing. Cham: Springer International Publishing, 2020.

(29) Kaya Uyanık, Gülden, and Neşe Güler. "A Study on Multiple Linear Regression Analysis." Procedia - Social and Behavioral Sciences 106 (December 1, 2013): 234–40.

(30) Udemy. "Deep Learning Foundation : Linear Regression and Statistics." Accessed March 20, 2023. https://www.udemy.com/course/linear-regression-in-python-statistics-and-coding/.

(31) PhD, Josh Starmer. The StatQuest Illustrated Guide To Machine Learning. Independently published, 2022.

List of figures