Creation of digital twin for industry 4.0 plant using MATLAB and Simulink

Bachelor Thesis

Supervisor:
Prof. Dr. Ing. Frank Schrödel
Co-Supervisor:
M.Eng. Jayabadhrinath Krushnan

German university in Cairo
Faculty of Engineering and material science
Mechatronics department
Hochschule Schmalkalden
University of Applied Sciences

Thesis Submitted by
Omar Montaser Ahmed Mohamed Hussein
omar.hussein@student.guc.edu.eg
omarmontasser51@gmail.com
52-14724
318404

August 2024

Abstract

The rapid progression of digital technologies such as the Industrial Internet of Things (IIoT), cloud computing, and data analytics has ushered in the era of Industry 4.0 in manufacturing. Central to this transformation is the digital twin concept, a virtual model of a physical asset or system that simulates behavior, predicts performance, and optimizes operations. This thesis focuses on developing a comprehensive digital twin for an Industry 4.0 manufacturing plant, which includes key components like a conveyor system with proximity sensors, a filling plant with two tanks, a pump, pipes, a flow meter, and a 4-DOF ASRS robotic arm for material handling. To achieve seamless integration between the digital twin and the plant's operational systems, the OPC UA (Open Platform Communications Unified Architecture) communication protocol was implemented for real-time data exchange and synchronization between the virtual and physical environments. The digital twin for the filling plant was created by modeling and simulating the pump and filling process in Simulink. Additionally, a digital twin for the 4-DOF ASRS robotic arm was developed by modeling its kinematics, including trajectory, direct, and inverse kinematics, using Simulink and the Robotics Toolbox. The thesis concludes with the integration of these individual digital twins into a unified model of the entire manufacturing system. This integrated digital twin allows for the observation of the physical plant's behavior in a virtual environment, facilitating further system optimization. The successful development and validation of these digital twins illustrate the potential of this technology to enhance the monitoring, control, and optimization of complex Industry 4.0 manufacturing systems.

Contents

List of Figures

List of Abbreviations

Abbreviation	Description
ASRS	Automated Storage and Retrieval System
DOF	Degrees of Freedom
IGUS	Integrated Gantry Unit System
I/O	Input/Output
OPC UA	Open Platform Communications Unified Architecture
PID	Proportional-Integral-Derivative
PLC	Programmable Logic Controller
RFID	Radio Frequency Identification
SCADA	Supervisory Control and Data Acquisition
TIA	Totally Integrated Automation

1 Introduction

The Industrial Revolution, a significant period in modern history, marked the shift from an agrarian and craft-based economy to one dominated by industrialization and machine-driven production. These technological advancements brought about new ways of working and living, leading to trans-formative changes in society. The initial phase, known as the First Industrial Revolution, began in England around 1760 and subsequently spread to other parts of the world. During this phase, manual craftsmanship was gradually replaced by water and steam-powered machines. With the invention of electricity, the Second Industrial Revolution (Industry 2.0) emerged, characterized by the use of electricity to power machines. This revolution saw the introduction of mass production techniques in the late 19th century, which were further improved by the implementation of assembly lines in the early 20th century. The Third Industrial Revolution (Industry 3.0), which took place in the 1970s, was centered around automation and was made possible by advancements in electrical engineering and information technology. Finally, the Fourth Industrial Revolution (Industry 4.0) is characterized by the rise of intelligent factories that focus on manufacturing customized products. This revolution involves the development of flexible and highly complex systems (Schwab, 2016).

1.1 Motivation

The rise of Industry 4.0 has ushered in a new era of smart manufacturing, characterized by the increased use of digital technologies, automation, and data-driven decision making. A key component of Industry 4.0 is the concept of the digital twin - a virtual model of a physical asset, process, or system that can be used to monitor, analyze, and optimize performance. Digital twins offer numerous benefits for industrial plants operating in the Industry 4.0 landscape (i-Scoop, 2020). By creating a digital representation of the physical plant, engineers and operators can gain deeper insights into plant performance, identify bottlenecks, test new configurations, and make informed decisions to improve efficiency, productivity, and quality. Additionally, digital twins can enable predictive maintenance, allowing problems to be detected and addressed before they lead to costly downtime (IBM, 2020). Developing a digital twin for an Industry 4.0 plant would be a highly relevant and impactful project. This would involve:

- Gathering data from sensors and systems within the plant to create a comprehensive digital model of the physical assets, processes, and workflows
- Integrating the digital twin with the plant's control systems, production management software, and other relevant IT infrastructure to enable real-time monitoring and data exchange
- Implementing advanced analytics and simulation capabilities within the digital twin to support decision-making, optimize plant operations, and enable predictive maintenance
- Validating the accuracy and usefulness of the digital twin through testing and pilot deployments, and identifying areas for further improvement (Siemens, 2021)

By completing such a project, A valuable hands-on experience with the latest Industry 4.0 technologies, data integration, and digital twin development will be gained. The resulting digital twin could also provide tangible benefits to the plant, helping to drive improvements in efficiency, productivity, and competitiveness. Overall, a bachelor's thesis focused on creating a digital twin for an Industry 4.0 plant would be an excellent opportunity to tackle a cutting-edge challenge with real-world relevance and impact. A digital twin process is illustrated in the figure 1 below.

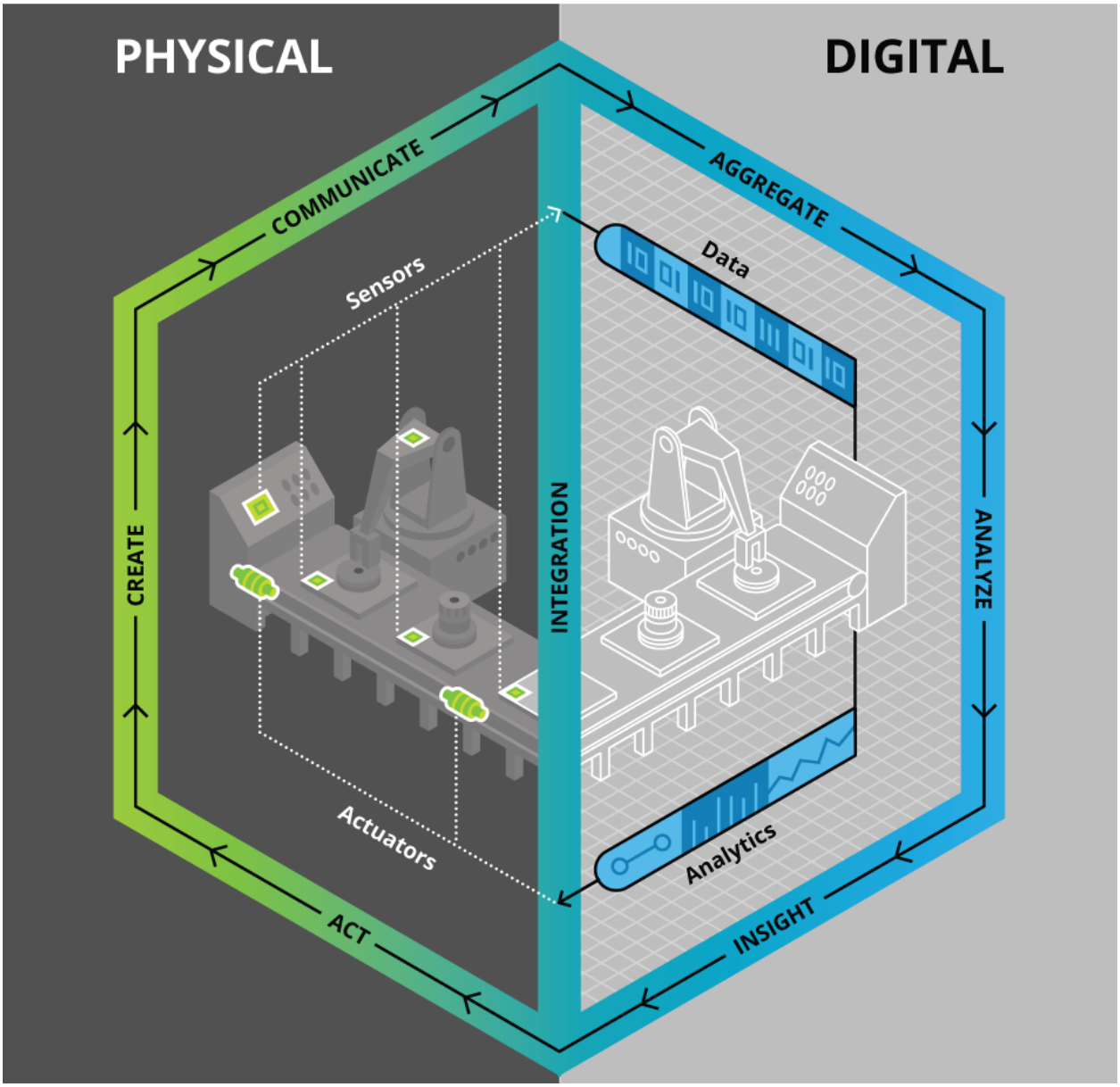


Figure 1: Digital Twin [5]

The development of a digital twin for an Industry 4.0 plant also presents an excellent opportunity to showcase innovative thinking and technical expertise. As the concept of digital twins continues to evolve, there is significant scope for exploring novel approaches and capabilities that can set the plant apart from its competitors. For example, the integration of advanced data analytics, machine learning, and predictive modeling within the digital twin could enable more sophisticated optimization and decision-making capabilities. By leveraging artificial intelligence to identify patterns, predict failures, and recommend improvements, the digital twin could become a strategic asset that drives sustainable competitive advantage. Demonstrating the ability to design and implement a cutting-edge digital twin solution that delivers tangible business value would be a significant accomplishment for the plant. Not only would it showcase the technical proficiency, but it would also highlight the capacity for innovation, problem-solving, and strategic thinking. By emphasizing the opportunities for innovation and competitive advantage.

1.2 Problem Statement

The manufacturing industry is currently undergoing a significant transformation, driven by the advent of Industry 4.0 technologies. This technological revolution has led to the increased adoption of cyber-physical systems, which integrate physical production processes with digital data processing and communication.

One of the key enablers of this transformation is the concept of the digital twin:

- A digital twin is a virtual representation of a physical system or process that is continuously updated with real-time data from the physical counterpart.
- This digital model allows for the simulation, analysis, and optimization of the physical system, leading to improved efficiency, reduced downtime, and enhanced decision-making capabilities.

Despite the recognized benefits of digital twins, their implementation in complex manufacturing environments poses several challenges:

- The development of a comprehensive digital twin that accurately captures the behavior of all the interdependent components within the manufacturing plant can be a complex and time-consuming task.
- The integration and synchronization of the individual digital twins into a cohesive model of the entire system requires careful design and coordination.
- The validation of the digital twin's accuracy and its ability to reliably predict the behavior of the physical plant is crucial for building trust and enabling effective decision-making.

This thesis aims to address these challenges by developing a comprehensive digital twin for an Industry 4.0 manufacturing plant, with a focus on modeling, simulation, and validation of the key components, as well as the integration of the individual digital twins into a unified system-level model. The successful implementation of this digital twin framework can serve as a blueprint for the adoption of this technology in similar complex manufacturing environments.

1.3 Structure

The purpose is to develop a complete digital twin for an Industry 4.0 plant, which includes a Festo filling plant and a 4-DOF robotic arm. This is achieved by establishing communication between the physical system (the Festo plant) and the software (MATLAB and Simulink) using the OPC Unified Architecture (OPC UA) communication protocol. The 4-DOF robotic arm is modeled using Simulink tools such as the Simscape and Robotics System toolboxes. The thesis covers the basics and fundamentals of the processes involved in the project, including PLCs, communication protocols, MATLAB and Simulink, control engineering, and the concept of the digital twin. The second part of the thesis details the implementation of the digital twin, including the connection between the PLC and MATLAB/Simulink using the OPC UA protocol. It also includes the modeling of the Festo filling plant, which features a pump model in Simulink with a PID block to optimize the filling process. Further development of the Simulink model is done by adding functions to enhance the filling process. The modeling of the 4-DOF robotic arm is also described, using Simulink tools. The digital twin of the Festo filling plant and the robotic arm model are validated through simulations and graphs from the software, which are compared with the actual behavior of the physical system. The thesis concludes with a summary of the findings and results.

2 Fundamentals

At the core of this project is the development of a comprehensive digital twin that accurately represents the physical assets, processes, and operations of an Industry 4.0 plant. This will require a thorough understanding of the plant's architecture, sensors, control systems, and data infrastructure. The digital twin must be built upon a robust data model that can integrate information from various sources. Establishing secure and reliable data connectivity between the physical plant and the digital twin will be crucial, as will the implementation of data preprocessing, cleaning, and transformation techniques to ensure the integrity and quality of the digital data. Different types of PLCs are used in the plant that are communicating with the other parts of the system using communication protocols such as OPC Unified Archeticture (OPC UA), MODBUS and Ethernet. These communications are between the PLCs (The physical system) and the MATLAB or Simulink software. Some software tools are utilized to model the physical system such as simscape toolbox and robotics systems toolbox. These tools are mainly used to model the robotic arms in the plant.

2.1 PLC

Programmable Logic Controllers (PLCs) are industrial digital computers that are widely used for automation and control of various industrial processes. They are a crucial component in modern manufacturing, robotics, and process control systems. PLCs consist of a central processing unit (CPU), memory, and input/output (I/O) modules that are programmed to monitor sensors, make decisions based on a control program, and send signals to actuators or other devices to control the process. The control program is typically written using ladder logic, a graphical programming language that resembles electrical diagrams, but other programming languages such as structured text, function block diagrams, and sequential function charts are also commonly used. PLCs are known for their ruggedness, reliability, and flexibility, making them ideal for harsh industrial environments. They can be programmed to perform a wide range of tasks, from simple on/off control to complex motion control and data processing. As technology continues to evolve, modern PLCs incorporate advanced features like networking, remote access, and integration with SCADA (Supervisory Control and Data Acquisition) systems, further enhancing their capabilities in industrial automation and control.The scan cycle is the fundamental operational loop of a PLC, where the controller repeatedly executes a series of tasks in a specific order. During each scan cycle, the PLC reads the current state of all input devices, executes the user-programmed logic, and updates the state of all output devices. The duration of this scan cycle, known as the scan time, is a critical factor in determining the overall responsiveness and performance of the PLC-controlled system. Factors such as the complexity of the programmed logic, the number and type of I/O devices, and the processing power of the PLC hardware can all impact the scan cycle time (Programmable Logic Controllers (PLCs), n.d.).

Figure 2: Siemens PLC [7]

There are various types of PLCs depending on the maunfacturing company, dealing with Siemens PLC in this project.Siemens PLCs are renowned for their reliability, performance, and ease of use, making them a preferred choice for manufacturing, process control, and infrastructure applications.

TIA Portal For Siemens PLC Programming

The Totally Integrated Automation (TIA) Portal is the engineering framework developed by Siemens for configuring and programming their Programmable Logic Controllers (PLCs), Human-Machine Interfaces (HMIs), and other automation components. TIA Portal provides a unified software environment that allows engineers to seamlessly integrate various Siemens automation devices and create complete automation solutions [1].
TIA Portal is primarily used to program Siemens PLCs, which are the core of many industrial automation systems. Some key features of TIA Portal for PLC programming include:

- Integrated programming of Siemens PLCs, HMIs, and other devices
- Intuitive user interface with drag-and-drop functionality
- Support for a wide range of Siemens automation hardware Powerful programming languages such as Ladder Logic, Function Block Diagram, and Structured Text
- Comprehensive project management and version control capabilities
- integrated simulation and testing tools for validating automation programs.

For more detailed information on TIA Portal and its use in Siemens PLC programming, refer to the Siemens TIA Portal documentation and user guides[Siemens. (2023)].

2.2 Communication Protocols

Effective data exchange and integration between the physical plant and the digital twin is a critical component of this bachelor thesis. A robust communication infrastructure that can seamlessly link the various systems, devices, and data sources within the plant to the digital twin platform should be implemented. This will require a deep understanding of the common industrial communication protocols used in modern manufacturing environments, such as Modbus, OPC UA, EtherNet/IP, and PROFINET. Evaluation of the capabilities and limitations of these protocols should be done, as well as their compatibility with the plant's existing automation and control systems (Schulse, 2018). Establishing secure and reliable data connectivity will be of paramount importance, and the student may need to explore the use of industrial internet of things (IIoT) gateways, edge computing devices, or cloud-based integration platforms to facilitate the seamless flow of data between the physical and virtual domains. Additionally, the student should investigate methods for handling real-time data streams, historical data, and event-based information to ensure the digital twin accurately reflects the current state of the plant. Mastering the intricacies of industrial communication protocols and data integration strategies will be essential for developing a digital twin that can effectively monitor, analyze, and optimize the performance of the Industry 4.0 plant. The student's ability to navigate these technical complexities and implement a scalable, secure, and interoperable communication infrastructure will be a key aspect of the successful completion of this bachelor thesis.

2.2.1 Modbus Protocol

Modbus is a widely adopted industrial communication protocol that is particularly well-suited for integrating the digital twin with the physical assets and control systems of an Industry 4.0 plant. Some key information about Modbus and its role in digital twins:

- Modbus is one of the earliest and most widely used protocols in the automation industry, providing a simple and robust mechanism for exchanging data between the digital twin and the plant's various devices, such as programmable logic controllers (PLCs), sensors, and actuators
- The Modbus protocol operates on a master-slave architecture
- This allows the digital twin to access real-time process data, monitoring information, and control parameters directly from the plant's operational systems
- Modbus supports both serial (Modbus RTU) and Ethernet (Modbus TCP/IP) communication, providing flexibility in how the digital twin can be integrated with the plant's existing infrastructure
- By utilizing the Modbus protocol, the establishment of reliable and secure data links between the digital twin and the plant is made, enabling the virtual model to accurately reflect the current state of the physical assets and support advanced analytics, simulation, and optimization capabilities
- Mastering the implementation of Modbus communication within the digital twin will be a crucial aspect, as it underpins the ability to seamlessly integrate the virtual and physical domains of the Industry 4.0 plant

(Modbus Organization, 2021)

2.2.2 Ethernet

Ethernet, the ubiquitous local area network (LAN) technology, plays a pivotal role in enabling the seamless integration of the digital twin with the various systems and devices within an Industry 4.0 plant. Some key information about the importance of Ethernet in digital twin implementation:

- Ethernet has become the preferred communication backbone as the plant's operational technology (OT) and information technology (IT) domains converge, providing a reliable, high-speed, and widely adopted means of data exchange
- By utilizing Ethernet, the digital twin can access real-time data, perform remote monitoring and control, and facilitate the exchange of information between the virtual and physical domains
- Mastering the implementation of Ethernet-based communication within the digital twin architecture will be crucial, as it enables the virtual model to seamlessly interact with the plant's operational systems, ultimately supporting the digital twin's role in optimizing plant performance and driving Industry 4.0 initiatives

2.2.3 OPC UA Communication

OPC UA is a flexible and secure communication standard that has become increasingly prominent in the world of industrial automation and control systems. With some properties such as:

- Client-Server Architecture:
  OPC UA communication is based on a client-server model, where clients (PC in our case) can access data and services provided by servers (Siemens PLC)
- Standardized Data Models:
  OPC UA utilizes standardized data models that define the structure and semantics of the information being exchanged between systems such as object model, address space model and data type model.

  These data models can be extended to accommodate specific industry or application requirements, ensuring seamless integration of diverse systems
- Robust Security:
  OPC UA includes comprehensive security mechanisms to protect data integrity and ensure authorized access.
  Features like strong encryption, user authentication, and access control help safeguard sensitive information and prevent unauthorized access or manipulation.
- Flexible Connectivity:
  OPC UA supports various transport protocols, including TCP/IP and HTTPS, enabling communication across a wide range of network topologies and infrastructure.

  This flexibility allows for the deployment of OPC UA in a variety of industrial settings, from small-scale automation systems to large-scale, geographically distributed operations.

- Seamless Integration:
  The adoption of OPC UA has revolutionized industrial communication by providing a standardized, secure, and scalable platform for the efficient exchange of data and the integration of diverse automation and control systems

(OPC Foundation, 2020).

OPC UA communication protocol is illustrated in the figure 3 below.

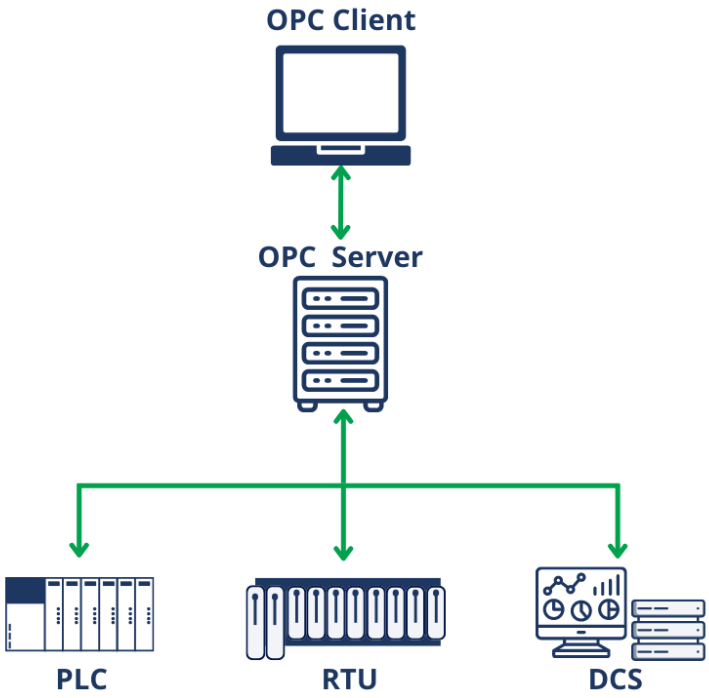


Figure 3: OPC UA protocol [13]

2.3 Modelling

Modeling is a fundamental process in various fields, from science and engineering to economics and social sciences. It involves the creation of representations or simplified versions of real-world systems, phenomena, or processes. The purpose of modeling is to gain a deeper understanding of the subject, make predictions, and test hypotheses without the need for direct experimentation or observation of the actual system. Effective modeling requires the identification of the key variables, parameters, and relationships that define the system of interest. Depending on the complexity of the system, models can range from simple conceptual diagrams to sophisticated mathematical or computational simulations. The choice of modeling approach depends on the specific goals, the available data, and the level of detail required. Iterative refinement and validation of the model are crucial to ensure its accuracy and relevance. Ultimately, modeling provides a powerful tool for problem-solving, decision-making, and advancing knowledge in various disciplines.

2.4 Degrees of Freedom in Robotics

Building upon the principles of modeling, robotics is a rapidly advancing field that has transformed numerous industries, from manufacturing to healthcare. At the core of robotic systems lies a crucial concept: degrees of freedom (DOF) (Craig, 2017). Understanding this concept is essential for designing, controlling, and programming these sophisticated machines.
Degrees of Freedom (DOF):

- DOF refers to the independent movements or axes of rotation that a robotic system can perform
- Each DOF represents a specific type of motion, such as linear translation along an axis or rotation about an axis
- The number of DOFs directly determines the robot's ability to manipulate objects, navigate through complex environments, and execute intricate tasks

Varying degrees of freedom in robotic systems:

- Basic industrial robots typically have 3 to 6 DOFs, suitable for repetitive tasks like pick-and-place operations or welding
- Increased DOFs provide greater dexterity, allowing these robots to mimic the range of motion and flexibility of the human body

Importance of understanding degrees of freedom:

- Knowledge of DOFs is crucial in the design, control, and programming of robotic systems.
- By considering the required DOFs for a specific application, optimized robotic solutions for efficiency, accuracy, and adaptability can be developed

2.5 MATLAB and Simulink

MATLAB and Simulink are powerful software tools that can play a pivotal role in the creation and deployment of the digital twin for an Industry 4.0 plant. As part of this bachelor thesis, the student will need to explore the capabilities of MATLAB and Simulink in modeling, simulating, and optimizing the various systems and processes within the plant. MATLAB's robust mathematical and analytical capabilities can be utilized to develop advanced models and algorithms that accurately represent the plant's assets, production workflows, and control systems. Simulink, on the other hand, provides a visual, block-based programming environment that can facilitate the integration of these models into a comprehensive digital twin architecture. The student may leverage Simulink's extensive library of pre-built blocks, as well as its ability to interface with a wide range of industrial protocols, such as OPC UA, EtherNet/IP, and Modbus, to seamlessly connect the digital twin with the plant's operational systems. Furthermore, MATLAB and Simulink offer a range of optimization and control design tools that can be incorporated into the digital twin to drive process improvements and enhance overall plant performance. By harnessing the capabilities of MATLAB and Simulink, the student can develop a digital twin that is not only highly accurate and realistic but also equipped with powerful analytical and optimization capabilities to support the plant's transition to Industry 4.0.

2.5.1 Time Clock and Stop Simulation Blocks

The Time Clock block in MATLAB's Simulink is used to provide the current simulation time to other blocks in the model. This block is essential for synchronizing the various components of a Simulink model and ensuring that the simulation progresses at the appropriate pace. The Time Clock block generates an output signal that represents the current simulation time, which can be used by other blocks in the model to timestamp events or coordinate the behavior of different subsystems.
The stop block in MATLAB's Simulink is used to terminate the simulation of a model. It is a special block that does not have any inputs or outputs, and its sole purpose is to end the simulation run. When the stop block is reached during the simulation, MATLAB will stop the simulation and display the final simulation results. This can be useful in a variety of situations, such as:

- Conditional Termination: You can use the stop block to terminate the simulation based on certain conditions, such as when a specific variable reaches a certain value or when a certain amount of time has elapsed
- Debugging and Testing: The stop block can be used for debugging purposes, allowing you to pause the simulation and examine the state of the model at a specific point in time
- Optimization and Parameter Tuning: When you are trying to optimize the parameters of your model or explore the effects of different parameter values, the stop block can be used to terminate the simulation once the desired conditions are met

(Mathworks, 2023).

2.5.2 MATLAB Function & MATLAB Interpreted Function

The MATLAB Function block is essentially a Simulink block that can execute MATLAB code during the simulation. This block provides a way to incorporate user-defined functionality, complex algorithms, or specialized computations into your Simulink models.

Here are some key features and benefits of the MATLAB Function block:

- Custom Functionality: The MATLAB Function block enables you to encapsulate your own MATLAB code and utilize it within your Simulink model. This allows you to extend the capabilities of Simulink by adding specialized algorithms, complex mathematical operations, or any other custom functionality
- Flexible Inputs and Outputs: The MATLAB Function block can have multiple input and output ports, allowing you to pass data between the block and the rest of your Simulink model. This flexibility enables you to integrate the custom MATLAB code with the rest of your system model
- Code Reuse: By using the MATLAB Function block, you can easily reuse your existing MATLAB code within your Simulink models, leveraging your investments in MATLAB development
- Rapid Prototyping: The MATLAB Function block facilitates rapid prototyping and iterative development. You can quickly modify and test your custom MATLAB code within the Simulink environment, without the need to constantly recompile your entire model
- Integration with MATLAB Toolboxes: If your custom MATLAB code utilizes functions or features from MATLAB Toolboxes, you can seamlessly integrate them into your Simulink model using the MATLAB Function block

MATLAB Interpreted Functions offer similar features and benefits to standard Simulink blocks, but with a key distinction. In the case of an Interpreted Function, the underlying function is pre-saved and pre-built within the MATLAB environment as an .m file.

This allows the same function to be utilized across multiple Simulink models simultaneously.
In contrast, a MATLAB Function is constructed solely within the Simulink model itself, and is not accessible outside of the specific .slx file where it is defined.
The choice between using an Interpreted Function or a MATLAB Function depends on the specific requirements of the project. If the function needs to be reused across multiple Simulink models, the Interpreted Function approach is generally preferred, as it provides a centralized and easily shareable implementation.
On the other hand, if the function is unique to a particular Simulink model and will not be used elsewhere, the MATLAB Function may be a more convenient option. MATLAB Functions can also be advantageous when the function has a large number of inputs and outputs, as the MATLAB Function block can provide a more intuitive and manageable interface within the Simulink model.
In summary, both Interpreted Functions and MATLAB Functions have their respective use cases, and the decision to use one over the other should be guided by the specific requirements of the project and the desired level of code reuse and model portability.(Mathworks, 2023).

2.5.3 Simscape Toolbox

The Simscape toolbox within Simulink is a powerful addition to the digital twin development process for an Industry 4.0 plant. Simscape allows to create physics-based models of the plant's physical systems, including mechanical, electrical, hydraulic, and thermal components, and seamlessly integrate them into the overall digital twin architecture. By leveraging Simscape, Detailed, multi-domain models that accurately simulate the real-world behavior of the plant's assets, processes, and interconnections can be constructed. This level of physical modeling is crucial for the digital twin, as it enables the virtual representation to mimic the intricate dynamics and interactions that occur within the physical plant. The Simscape toolbox provides a comprehensive library of pre-built components, which can be customized and combined to build complex models, reducing the time and effort required for model development. Additionally, Simscape's tight integration with Simulink allows for the seamless co-simulation of the digital twin, where the physical models interact with the control systems, manufacturing workflows, and optimization algorithms. By incorporating Simscape into the digital twin, the student can create a more robust, accurate, and versatile virtual representation that can be used for predictive maintenance, process optimization, and real-time decision support in the Industry 4.0 environment.(Mathworks, 2023).

2.5.4 Robotics Systems Toolbox

The Robotics Systems Toolbox within Simulink can be a valuable asset in the creation of a comprehensive digital twin for an Industry 4.0 plant, particularly in scenarios where robotic automation plays a significant role. This toolbox provides the student with a wide range of capabilities to model, simulate, and analyze robotic systems, which can be seamlessly integrated into the overall digital twin architecture. By leveraging the Robotics Systems Toolbox, the student can develop detailed, physics-based models of industrial robots, grippers, and other automation equipment used within the plant. These models can accurately capture the kinematics, dynamics, and control systems of the robotic components, allowing the digital twin to simulate their precise movements and interactions with the plant's production workflows. Furthermore, the toolbox's ability to interface with real-world robot controllers and communication protocols, such as ROS and KUKA Robot Language, enables the student to establish a bi-directional link between the digital twin and the actual robotic systems on the plant floor. This integration allows the digital twin to monitor and respond to real-time changes in the robotic automation, enhancing the virtual model's ability to predict and optimize production performance. By incorporating the Robotics Systems Toolbox into the digital twin development, A more comprehensive, integrated, and accurate virtual representation of the Industry 4.0 plant can be created. Robotics systems toolbox in Simulink library is captured in figure 4 below.(Mathworks, 2023).

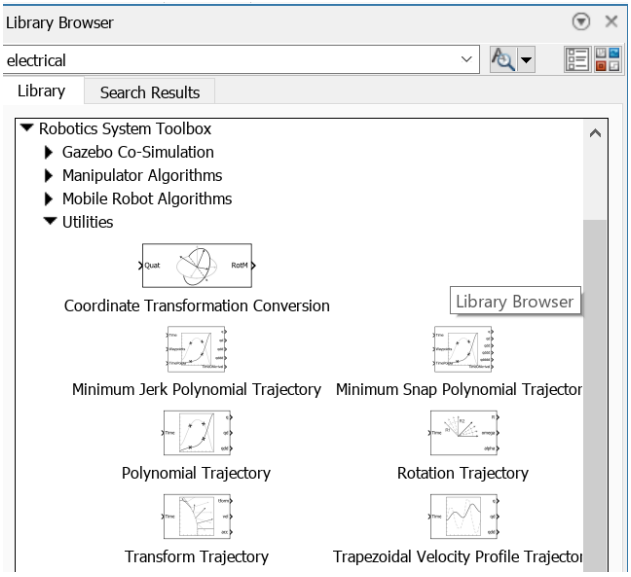


Figure 4: Robotics toolbox in MATLAB environment

2.6 Digital Twin

A digital twin is a virtual representation of a physical object, system, or process, which could be referred to as D.T. It is created by integrating real-time data from sensors, historical data, and computational models to simulate the behavior and characteristics of the physical counterpart. Digital twins are used to monitor, analyze, and optimize the performance of physical assets and processes.

To implement a digital twin for an Industry 4.0 plant, the typical steps involve data acquisition, data integration and synchronization, modeling, visualization and monitoring, analysis and optimization, and communication and control. This allows the digital twin to accurately reflect the current state of the physical system, provide real-time insights, and enable advanced analysis and decision-making.

The utilization of digital twins in production systems offers several benefits, including simulation, optimization, and prediction capabilities. Digital twins provide an accurate representation of the production system, allowing for the identification of areas for improvement and the optimization of operations. They can be used to test scenarios, monitor manufacturing processes, and verify the effectiveness of processes to minimize downtime. Digital twins are also employed for status monitoring, simulation, and visualization, as well as applications in predictive maintenance, virtual commissioning, and research and development.

Illustrated in the figure below is a visualization of how a digital twin can be conceptualized as a parallel system alongside the physical world system. The digital twin and the physical system are connected through communication channels that allow for the exchange of information regarding inputs, outputs, and real-time variables. This setup enables the two systems to operate concurrently, with the virtual digital twin model replicating the conditions of the real-world physical system. This approach allows for the exploration and analysis of the system's behavior in the digital world, while the physical system continues to function in the real world.

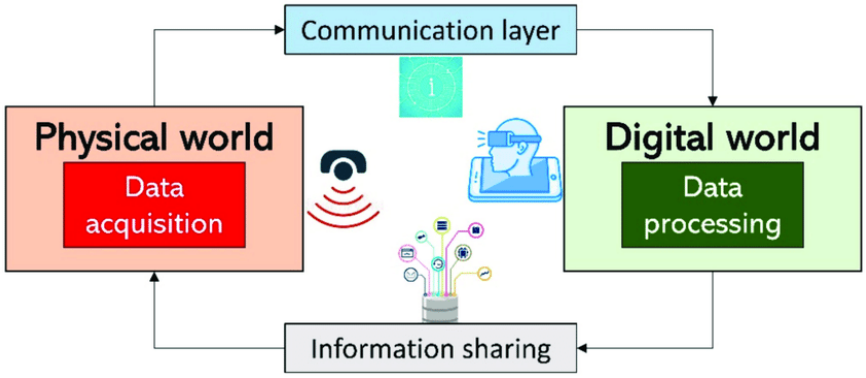


Figure 5: Visualisation of Digital Twin [15]

The integration of programmable logic controllers (PLCs), microcontrollers, or other industrial controllers with a digital twin of the entire system offers significant benefits for industrial applications. Studies have shown that this approach can lead to improvements in system performance, reliability, and efficiency(Boschert, 2016; Qi, 2018). The key advantage lies in the ability to leverage the vast amount of data collected from the physical system to enhance the fidelity of the digital twin model (Negri, 2017). This data can be used for model validation, parameter estimation, and the development of more accurate predictive algorithms (Tao, 2018). The validated digital twin can then be used to explore optimization scenarios and implement predictive control strategies, leading to improved overall system performance (Soderberg, 2017). Furthermore, the integration of digital twins can enable remote monitoring, diagnostics, and preventive maintenance, resulting in reduced downtime and increased system reliability (Qi, 2019). This convergence of the digital and physical worlds paves the way for more intelligent and adaptive industrial systems.
To sum up, the integration of programmable Logic Controllers (PLCs) , micro-controllers or any other controller with a digital twin of the whole system provides another horizon of precise control and data collection in the physical world, through utilizing the great amount of data for the sake of modeling and validation which is further used for optimization and predictive control.
The key benefits of integrating digital twins with control systems and maintenance strategies include:

- Condition Monitoring and Problem Detection: The digital twin continuously monitors real-time data to identify anomalies and predict potential failures
- Predictive Maintenance: Maintenance schedules are optimized based on the digital twin's insights, enabling preventive actions before failures occur This reduces unplanned downtime and extends asset lifespans.
- Automated Maintenance Planning: The digital twin integrates with maintenance management systems to automatically schedule and optimize maintenance activities
- Continuous Improvement: Performance data and maintenance insights from the digital twin are used to iterative refine the physical system's design, controls, and maintenance practices
- Simulation-based Optimization: The digital twin's simulation capabilities allow testing and validation of different maintenance strategies and operational scenarios

By closely coupling digital twins with control systems and maintenance management, manufacturers can move to proactive, data-driven maintenance approaches. This enables smart manufacturing that improves overall equipment effectiveness, reduces downtime, and enhances product quality. The seamless integration of digital twins and controllers is a key enabler of efficient, predictive maintenance and the transition towards smart, data-driven manufacturing.

As illustrated in the figure 6 below the final image of digital twin utilization with the real system.

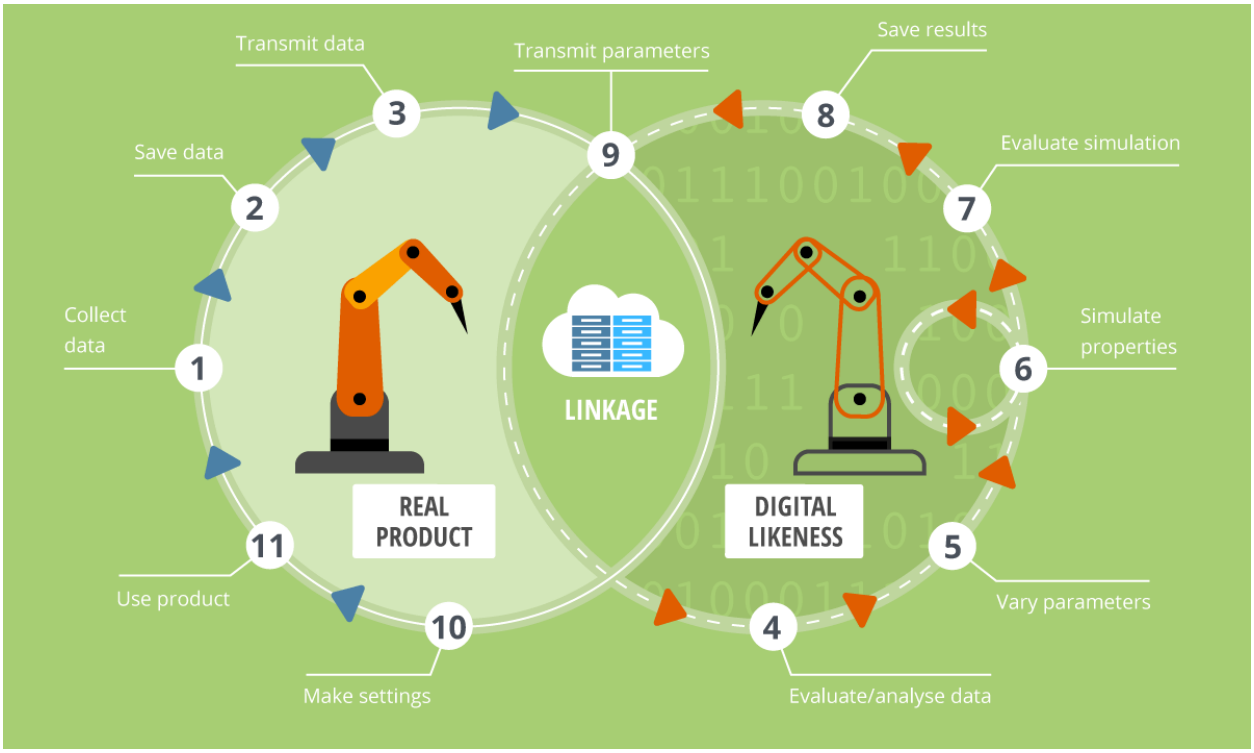


Figure 6: Utilization of Digital Twin [25]

2.7 Control Engineering

Control engineering is a multifaceted discipline that plays a crucial role in the design, analysis, and optimization of complex systems across a wide range of industries. From manufacturing processes to transportation networks, control engineering principles and techniques are fundamental to ensuring the reliable, efficient, and safe operation of modern technological systems. Here are some key aspects of control engineering and its importance:

- At its core, control engineering involves the development of systems and algorithms that can monitor, regulate, and manipulate the behavior of physical processes and devices to achieve desired performance objectives
- Control engineers leverage mathematical modeling, feedback control theory, and advanced computational methods to design control systems capable of precisely regulating variables such as speed, position, temperature, pressure, flow, and other critical parameters
- Control engineers are responsible for developing sophisticated algorithms and control architectures, such as PID (Proportional-Integral-Derivative) controllers, state-space models, and adaptive control techniques, to address the unique requirements and constraints of different systems
- In the context of Industry 4.0 and the rise of cyber-physical systems, control engineering plays a pivotal role in bridging the physical and digital domains, enabling the seamless integration of real-time data, machine learning, and autonomous decision-making capabilities (Nise, 2019)

2.7.1 Open Loop Control

Open-loop control systems represent a fundamental approach in control engineering, where the control action is determined solely based on the input or reference signal, without any feedback from the system's output or the controlled process. In an open-loop system, the controller does not have the ability to monitor the actual behavior of the system and make adjustments to correct any deviations from the desired response. The control signal is calculated and applied to the system in a predetermined manner, relying on the assumption that the system's behavior is well-understood and predictable. This simplicity makes open-loop control systems relatively straightforward to design and implement, as they do not require complex feedback mechanisms or sensors to measure the system's output. However, the lack of feedback also means that open-loop systems are susceptible to disturbances, parameter variations, and other uncontrolled factors that can cause the actual system response to deviate from the intended behavior. As a result, open-loop control is often used in applications where the system's dynamics are well-defined, the environmental conditions are stable, and the performance requirements are not particularly stringent. Examples of open-loop control applications include simple timing-based mechanisms, preset position or speed control systems, and certain types of home appliances or industrial processes where the desired output can be reliably achieved without the need for feedback-based adjustment. The open loop control is shown in figure 7.

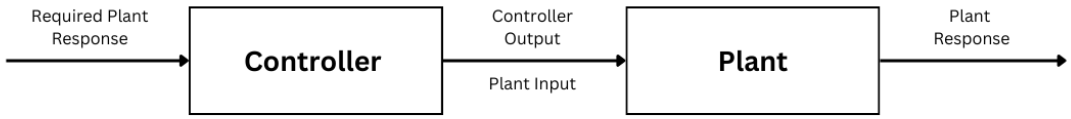


Figure 7: Open loop control

2.7.2 Closed Loop Control

Closed-loop control systems, also known as feedback control systems, represent a more sophisticated and versatile approach in control engineering. Unlike open-loop systems, closed-loop control continuously monitors the output of the system and uses this feedback information to adjust the control inputs in real-time, with the goal of maintaining the desired system performance. The fundamental mechanism of a closed-loop system involves comparing the actual output with the desired reference or setpoint, and then using an error signal to drive the control actions. This feedback loop allows the system to automatically compensate for disturbances, parameter variations, and other external influences that may affect the system's behavior. Closed-loop control systems are characterized by their ability to achieve higher accuracy, better stability, and enhanced robustness compared to open-loop systems. By constantly adjusting the control signals based on the measured output, closed-loop controllers can adapt to changes in the system dynamics and maintain the desired response, even in the presence of unpredictable or time-varying conditions. The flexibility and versatility of closed-loop control have made it the predominant approach in a wide range of applications, including industrial processes, servomechanisms, robotics, automotive systems, and various consumer electronics and appliances, where precise and reliable control is of critical importance. The closed loop control is shown in figure 8.

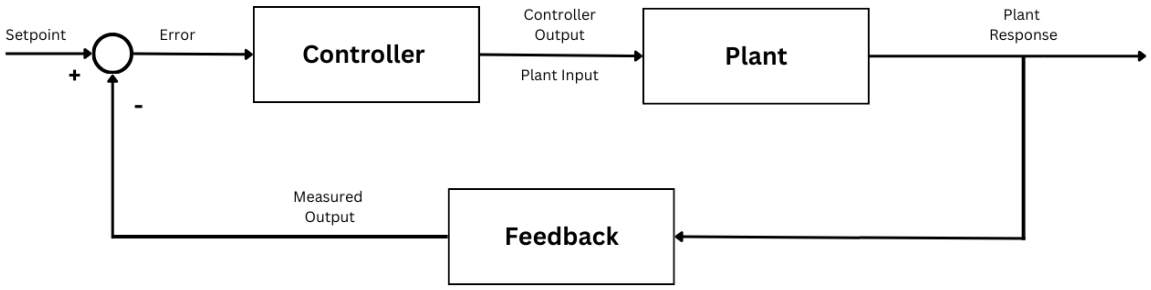


Figure 8: Closed loop control

2.7.3 PID Control

PID (Proportional-Integral-Derivative) control is a fundamental and widely-used control algorithm in the field of closed-loop control systems. The PID controller combines three distinct control actions - proportional, integral, and derivative - to generate a control signal that drives the system towards the desired setpoint or reference value. The proportional term responds to the current error between the output and the setpoint, the integral term addresses any steady-state errors by accumulating the error over time, and the derivative term anticipates future errors by responding to the rate of change in the error. By tuning the relative contributions of these three components, known as the PID gains, the controller can be optimized to achieve the desired system performance in terms of response time, stability, and robustness. PID controllers are renowned for their simplicity, effectiveness, and ease of implementation, making them a popular choice across a wide range of industries and applications, including process control, motion control, temperature regulation, and many others. The PID algorithm's ability to provide precise, reliable, and responsive control has solidified its status as a fundamental building block in the field of control engineering, and its widespread use continues to drive advancements in the design and optimization of complex, mission-critical systems.The PID control is shown in figure 9 below.

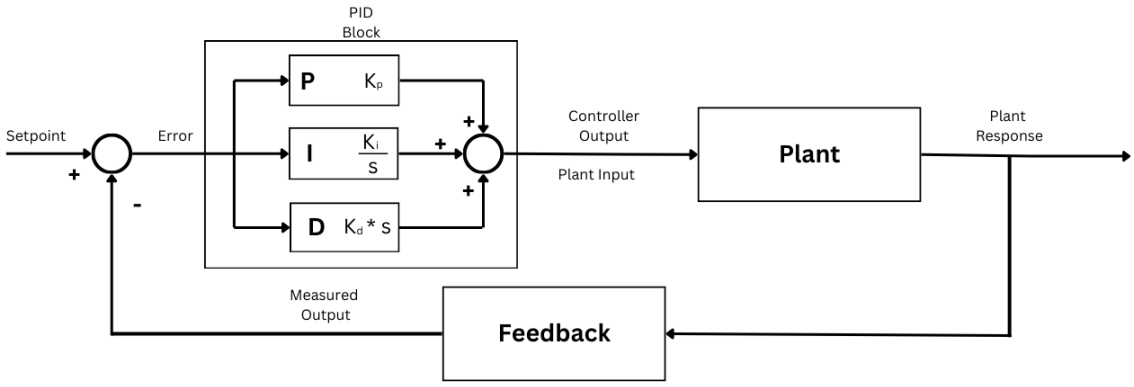


Figure 9: PID control

3 Concept Implementation

As discussed previously the main approach of this thesis is to demonstrate a digital twin replicate for the system targeting mainly the Festo fluid plant along with 4-DOF robotic arm for ASRS system.
In a nutshell the system will be divided into several sections :

- Section 1:
  Filling plant, which represent the heart of the whole system which contains the pump, the tanks and flow meter.
- Section 2:
  Two robotic arms responsible for handling in and out the product, first one is the 4-DOF robotic arm responsible for handling in, the other robotic arm is the 5-DOF IGUS robot for handling out.
- Section 3:
  This system consists of the conveyor , sensors and the capping station, which represents the transitions phases between handing in, filling and finally handing out which mainly depends on other processes.

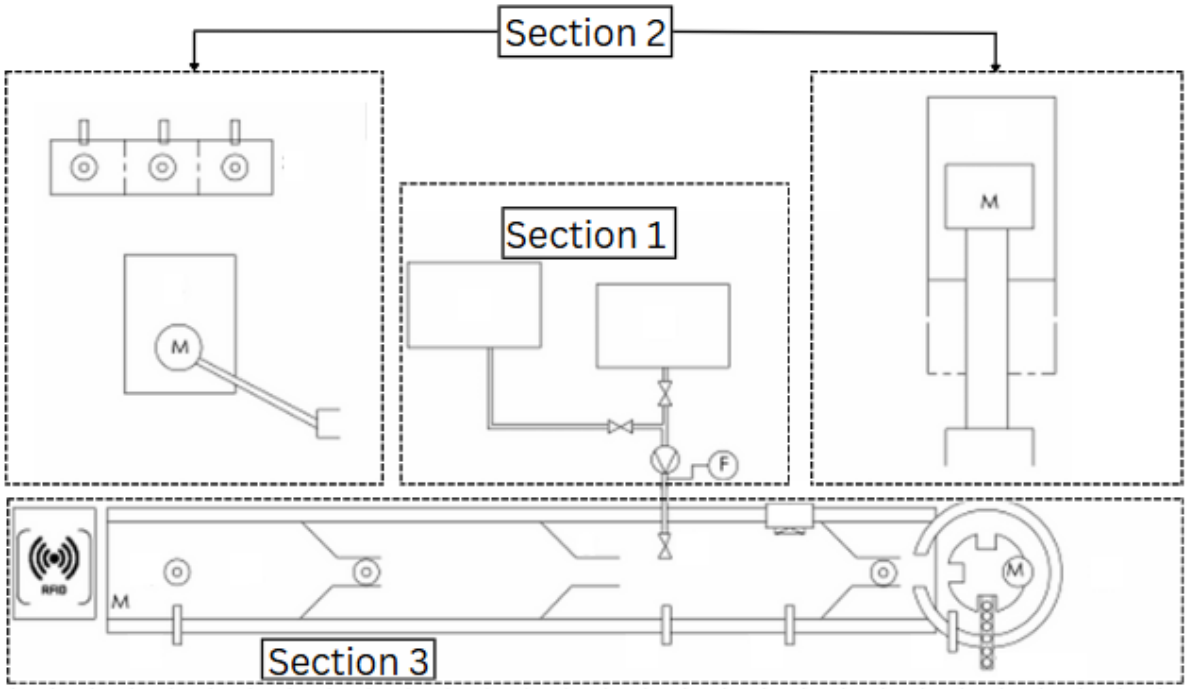


Figure 10: Plant Sections

3.1 Digital Twin using MATLAB and Simulink

In developing the comprehensive digital twin for the Industry 4.0 manufacturing plant, the decision was made to utilize MATLAB and Simulink as the primary modeling and simulation tools. This choice was supported by the capabilities and features offered by these widely-adopted software platforms. Firstly, MATLAB and Simulink provide a robust and flexible platform for modeling a wide range of physical systems and processes. The software's capability to seamlessly integrate various toolboxes, such as the Robotics Toolbox, enabled the detailed modeling of the complex components within the manufacturing plant, including the 4-DOF robotic arm and the filling plant with its pump, tanks, and flow meter (Mathworks, 2023). Moreover, the graphical programming interface of Simulink allowed for intuitive and efficient model development. The ability to create modular, hierarchical models facilitated the integration of the individual digital twins into a comprehensive representation of the entire manufacturing system. This modular approach not only simplified the model building process but also enhanced the overall flexibility and scalability of the digital twin (Simulink, 2023). Another crucial factor in selecting MATLAB and Simulink was the ease of co-simulation and integration with the Siemens PLC controlling the physical Festo filling plant. The OPC UA communication protocol enabled the seamless exchange of data between the digital twin and the real-world system, allowing for the validation of the digital model through direct comparison with the physical plant's performance (Cinar, 2020). Furthermore, the wide range of built-in libraries, toolboxes, and optimization algorithms available in MATLAB and Simulink proved invaluable in the development and fine-tuning of the digital twin. These tools and resources provided the necessary functionality to accurately model the system dynamics, optimize the various processes, and analyze the overall performance of the manufacturing plant (Mathworks, 2023).

3.2 Digital Twin Creation Concept using MATLAB and Simulink

The creation of a digital twin utilizing MATLAB and Simulink involves a series of procedural steps to model, integrate, and deploy the virtual representation of a physical system.

1. Modeling the Physical System
    - Develop a mathematical model of the physical system grounded in first principles, empirical data, or a combination thereof.
    - Construct a simulation model of the physical system employing MATLAB and Simulink.
2. Sensor Data Integration
    - Integrate sensor data from the physical system into the Simulink model.
    - utilize MATLAB and Simulink's data acquisition and communication capabilities to connect to real-time data sources, historical data, or simulated sensor data.
3. Model Validation and Calibration
    - Validate the simulation model against the actual physical system's observed behavior.
    - Utilize experimental data, field measurements, or other available information to calibrate and fine-tune the model parameters.
    - Ensure the digital twin accurately represents the physical system's dynamics and performance.
4. Real-time Simulation and Monitoring
    - Implement the digital twin in a real-time simulation environment employing Simulink Real-Time or other real-time simulation tools.

- Connect the digital twin to the physical system's sensors and actuators, facilitating real-time monitoring and data exchange.
- Visualize the digital twin's behavior and compare it to the physical system's performance.

The creation of a comprehensive digital twin is an iterative process, necessitating the revisitation of various steps to refine the model, improve the accuracy, and adapt to changing system requirements. By following this conceptual workflow, a virtual representation of a physical system can be developed, capable of supporting predictive maintenance, optimization, and integration into larger enterprise systems.

3.3 OPC UA Communication

A key aspect of the digital twin development process was the integration between the virtual models created in MATLAB and Simulink and the physical control system of the Festo filling plant, which was managed by a Siemens PLC. To enable this seamless communication and data exchange, the decision was made to use the OPC Unified Architecture (OPC UA) protocol.
OPC UA was chosen as the communication protocol for several compelling reasons. Firstly, OPC UA is a widely-adopted standard in the industrial automation and control systems domain, particularly in the context of Industry 4.0 and the integration of cyber-physical systems (Cinar, 2020). This widespread acceptance and support ensured that the communication between the digital twin and the Siemens PLC would be reliable and well-supported. Moreover, OPC UA provides a robust and scalable framework for data exchange, allowing for the bidirectional flow of information between the virtual and physical systems. This two-way communication was crucial for the validation of the digital twin, as it enabled the comparison of the simulated performance with the actual behavior of the Festo filling plant (Mahnke, Leitner, & Damm, 2009). The OPC UA protocol also offered a high degree of flexibility and extensibility, which was essential in the context of this project. The ability to define custom data models and object types allowed for the seamless integration of the various components and subsystems within the digital twin, ensuring a comprehensive and coherent representation of the entire manufacturing plant (Cavalieri & Salafia, 2019). Furthermore, the security features inherent to OPC UA, such as authentication, authorization, and encryption, provided a robust and reliable means of protecting the communication between the virtual and physical systems. This was particularly important in the context of Industry 4.0, where the integration of cyber and physical domains necessitates a heightened focus on cybersecurity (Cavalieri & Salafia, 2019). By using the capabilities of OPC UA, the integration of the digital twin with the Siemens PLC-controlled Festo filling plant was accomplished in a seamless and efficient manner. This integration enabled the validation of the virtual models against the physical system, ultimately contributing to the development of a comprehensive and reliable digital twin for the Industry 4.0 manufacturing plant.
In order to have an OPC UA communication built and fully functional, there are some technical requirements needed.These involve hardware establishment, matlab commands and finally PLC itself which also will need to be prepared to send or receive from or in a certain data memory.

1. Hardware establishment :

   Simply Ethernet cable from communication port of the PLC down to the router, then back from the router directly to the PC. See Appendix A.1.

2. PLC Activation and interfaces :

   - Communication Activation :
     Device configuration -> Properties -> General ,then choose OPC UA -> server -> General here all needed is to check activation and copy the address which would be needed by the other client. See Appendix A.2.

- Server Interface :
  At this step you determine the accesible data by creating a new server interface where all the accessible data will have special address, this data can be simply PLC tags dragged and dropped in this interface. See Appendix A.3.
- Last step would be to download the program to the PLC and go online to be ready send and receive.

3. Matlab Commands :

- First enter address and establish connection. See Appendix A.4.

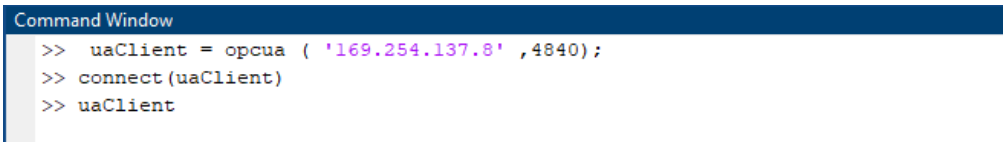


Figure 11: OPC address in Matlab

- Next step would be to choose one of the data variables in the server interface this depend on which data you need to access.
  For example here we target variable 'Matlab_to_PLC' found in position 47 in server interface of OPC back in the PLC program. See Appendix A.4.

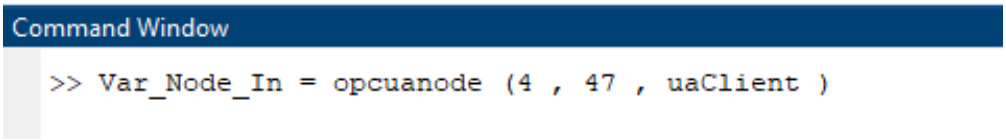


Figure 12: variable address in Matlab

3.4 Fluid Filling Plant

In this part of the thesis, the focus will be on representing the filling process of the Festo fluid plant, where the main parameters are the input voltage and the output flow rate. The key component in this process is the pump, which is surrounded by supplementary components such as the flow meter, valves, and tanks. This part of the system was previously explored in a master thesis Shah (2023). However, in this part of the thesis, the aim is to provide a clarification and more visualization of the digital twin in order to link it more closely with the hardware and the PID control already implemented on the plant. While the mentioned thesis focused on the mathematical modeling and basic steps of the digital twin. Instead, the great focus at this part of the thesis will be on discussing the steps of evolving the model on Simulink and the degree of limitations compared to the real hardware. The goal is to establish a strong connection between the Simulink model and the actual hardware.

3.4.1 Modeling Of The Pump

- Model of the motor:

  The most important component in the whole plant is the pump, which is responsible for the filling process. The pump's motor operates at a certain speed, measured in rotations per minute (RPM), with an input voltage ranging from 0 to 10V. The following equations model the behavior of the pump's motor:

  Voltage Equation:

$$V_a(s) - E_b(s) = I_a(s)(R_a + sL_a) \tag{1}$$

  where the variables are defined as follows:

  $V_a(s)$ is the applied armature voltage $E_b(s)$ is the back-emf voltage $I_a(s)$ is the armature current R_a is the armature resistance L_a is the armature inductance Substituting $E_b(s) = (K_e\phi)\omega_m(s)$, it becomes:

$$I_a(s) = \frac{V_a(s) - (K_e\phi)\omega_m(s)}{R_a(1 + s\tau_a)} \tag{2}$$

  where $\tau_a = \frac{L_a}{R_a}$ is the armature circuit constant.

  The Torque Equation is derived as follows:

  $\omega_m(s)$ is the motor angular velocity $K_e\phi$ is the motor constant $T_L(s)$ is the load torque B is the viscous friction coefficient $\tau_m = \frac{J}{B}$ is the mechanical constant of the motor, where J is the motor inertia.

$$\omega_m(s) = \frac{(K_e\phi)I_a(s) - T_L(s)}{B(1 + s\tau_m)} \tag{3}$$

  Figure 13 shows the complete mathematical model of the pump's motor in Simulink, through combining both equation 2 and equation 3 using common variable $I_a(s)$, to have at the end a model with voltage as the input and rotational speed as the output.

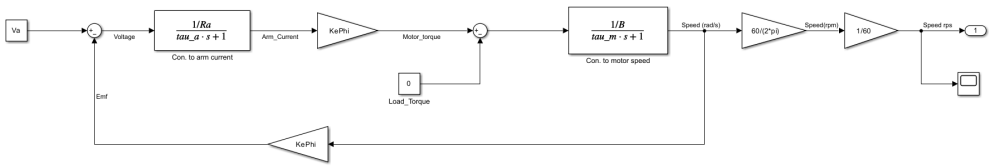


Figure 13: Motor Model

Ra	0.5 Ω
La	0.1 H
J	5 Kg-m^2
B	0.01 N-m-s/rad
LT	0 H
τ_a	L_a/R_a
$K_e\phi$	1.6 V-s/rad
Va	8.6 V

Table 1: Table of Variables

For the variables in the model, some were constants provided with the pump, while others were derived through tests. The value of V_a was a random number ranging from 0 to 10V used to test the model.

- Model of the whole pump :

  Where the pump composes of mainly a motor but the main output variable would be the flow rate rather than just speed of rotation. This relation to relate given speed to an actual flow rate was encountered through many tests done in the thesis Shah (2023) to get average constant value in order to have final model of the pump. This constant was 2.01 to relate rps with the flow rate.

  As shown in the next figure the model after building the relation between the voltage and the flow rate.

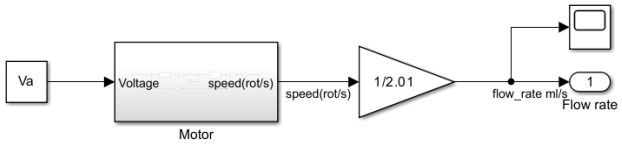


Figure 14: Pump Model

At this point the model was a digital twin for the pump and it's motor where it takes a voltage signal as input ranged from 0-10V and gives flow rate ml/s as an output.
This voltage signal can be constant input for example: 8.6V or it could be signal built which represents percentage of voltage to be supplied which could be used for testing purposes under various voltage levels.

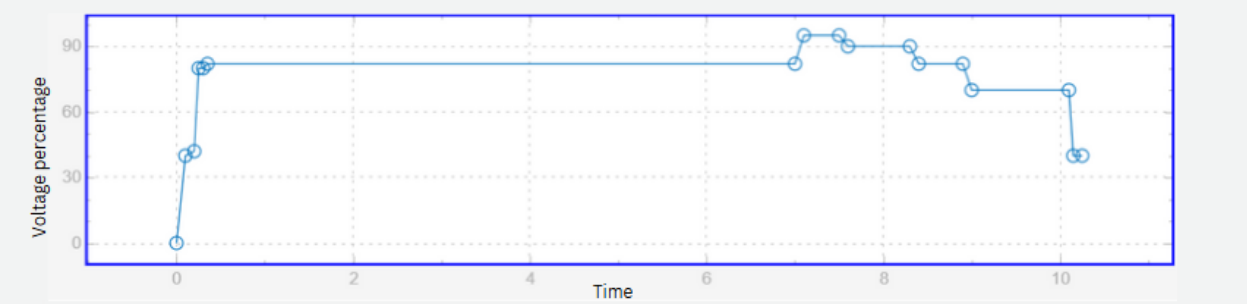


Figure 15: Input Voltage signal built

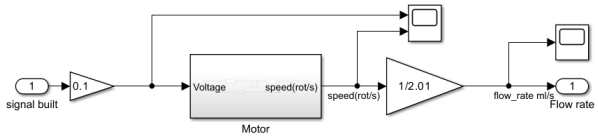
In order to take the model to a higher level of control and optimization there will be two aspects:
One way is to add PID control block in order to have sort of feedback control on the voltage supplied not only constant value or a predefined built signal in order to be more adaptive depending on the readings of the flow. Other way would be trying to relate the model more to the hardware in order to have similar response in the behavioral way not only the readings value, this way is mainly targeted in this thesis only in order to optimize the final model behaviour.

3.4.2 PID Closed Loop

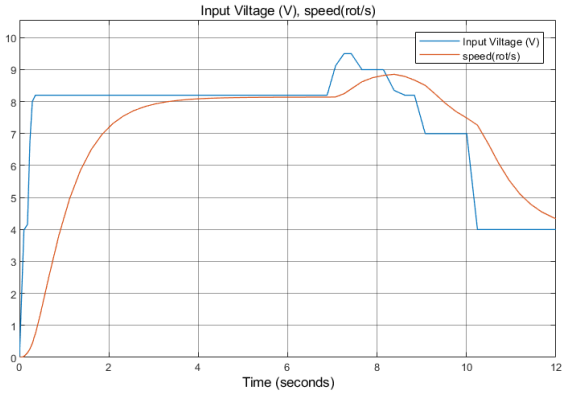
To go with the first way, a PID block should be added in order to have feedback control on the input voltage. This could be done through taking a feedback signal from a well recognized output from the model in order to compare it with the reference input supposed to be given. The error signal would then be processed with the PID gains K_p, K_i, and K_d to generate the control input voltage. The PID transfer function can be expressed as:

$$G_{PID}(s) = K_p + \frac{K_i}{s} + K_d s \tag{4}$$

This output signal was chosen to be the rotation speed per second (RPS),the reason for choosing this output specifically is that through testing the model with constant voltage input the rps was nearly the same value of the input voltage in the chosen range of 0-10V, this means that feedback signal could be taken from RPS signal without any in between gains to compare it with input voltage signal.
This is obvious in the following figures.



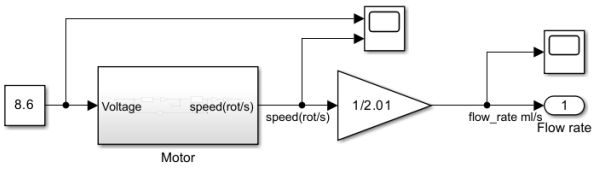
(a) Pump model with signal built as input



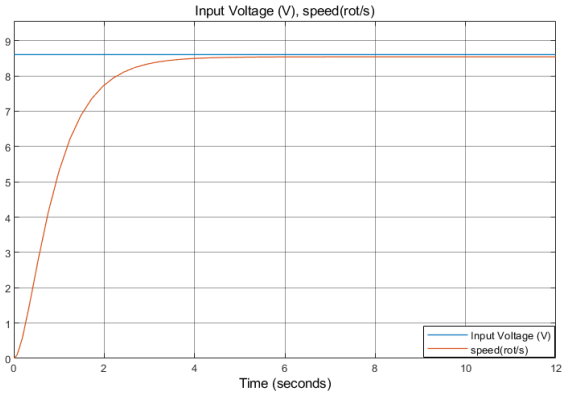
(b) RPS graph vs time

Figure 16: Rps response with signal built

The signal built which represents voltage percentage was multiplied by gain of 0.1 to scale from 0%-100% down to 0-10V, and it was obvious that it's typically the same. The only difference is the response which is the main target of the PID closed loop.
This response can be easily noticed when a constant input (step input) is supplied to the voltage.



(a) Pump model with step input



(b) RPS graph vs time

Figure 17: Rps response with step input

After putting hands on the required output to take the feedback, what next should be done is the closed loop itself.

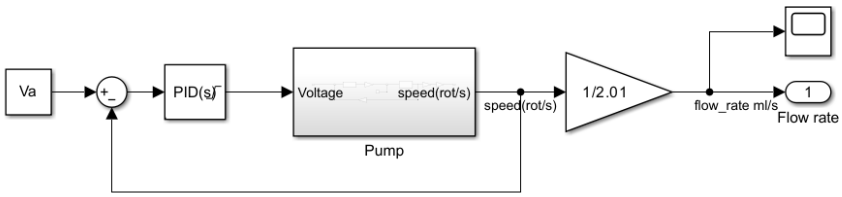
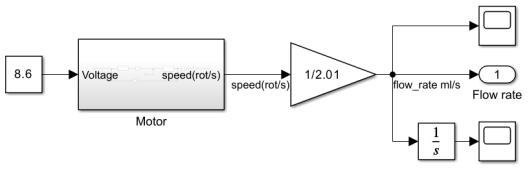


Figure 18: Model with PID

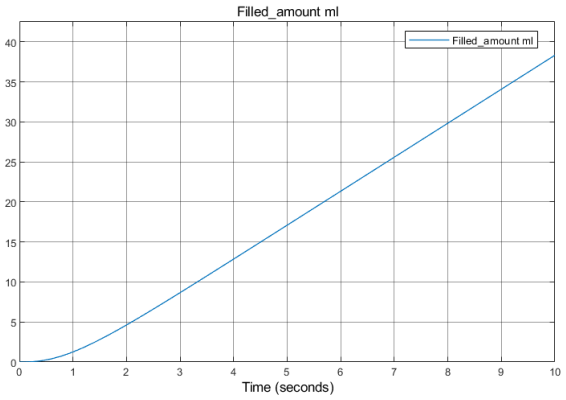
For the PID block variables : Kp : 4 , Ki : 5 , Kd : 1, with upper limit: 10 and lower limit: 0.
This values of Kp, Ki and Kd were tuned through iterative experiments for this stage of the model, also the upper limits and lower limits of output are used to limit the control signal of voltage according to the plant limits. SeeB.2.
After integrating the PID block with the system, it is important to highlight then the reference voltage role that will be still used. This reference will be mainly chosen to be a preferable number for the pump to work on, nevertheless sometimes this value will not be supplied due to the PID effect but for quite long filling process this will be the input most of the time away from the transition at the beginning.

3.4.3 Model Development

Developing the current model has mainly triggered by the desire of both utilizing the PID control for better response along with having closer model to the real plant in the behavioral way in order to finally have a complete model for the whole filling process from the start till the end signals.
By this mindset work flow will be first by modifying the model to be more generalised, then tune the PID factors and other factors of the model to finally reach a new version of the previous model.
As a step zero simple block was added in order to calculate the amount of flow from the flow rate in order to just have a sort of monitoring on the amount of flow, simply integrator block gets the quantity of flow by accumulating the flow rate with respect to time.



(a) Model with flow amount calculator



(b) Filled amount graph

Figure 19: Flow amount calculator

The values in the graph in figure 19b represent the total amount filled from start till the point of time itself.

New methods:

One main difference between the digital twin model and the real plant is the effect of pipes path length which have sort of delay for the flow and require sort of relatively high input voltage to get over piping distance.

- The point of relatively high voltage was targeted through manual test by connecting the pump directly to a voltage supplier and start to test it with multiple voltages start from 0-10V :
  The pump was supposed to start supplying flow from 2V but under the effect of pipes flow start to run at voltage of 7V. See B.1.

  This test have shown the possible range that could be given as reference input voltage.

- The problem of the sort of delay was targeted through a simple idea: pipes have a 2-second delay from being supplied till being read by the flow meter and reaching the bottle required to be filled, as clearly shown in Chapter 5, Section 5.2 of the thesis (Shah, 2023).

  By this way of observing, a block was inserted to the closed loop in order to mimic this behaviour, this block simply ignores the flow rate of the first 2 seconds and behaves as its normally no flow yet, but this is only done inside the closed loop. This means that this flow will be taken in consideration with the total flow amount calculation and this makes sense as however this flow isn't observed by the flow meter but still represents amount of liquid will reach the target bottle at the end.

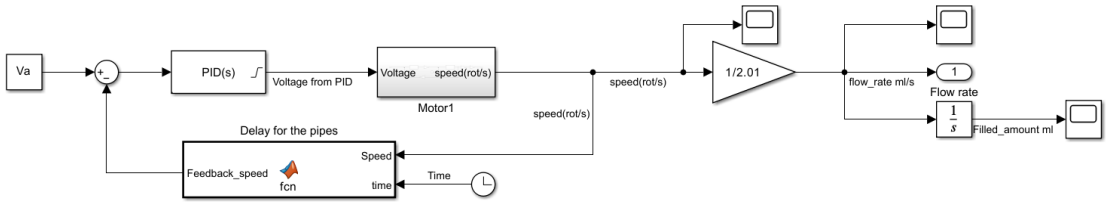


Figure 20: Pipes delay model

```
function Feedback_speed = fcn(Speed, time
    )
    if time < 2
        Speed = 0;
    end
    Feedback_speed = Speed;
end
```

This function gives a zero feedback to the closed loop for the first 2 seconds only. After that, it does nothing. The time in seconds is supplied by the Clock block in Simulink.

One drawback in the previous model was related to problems with the total filled amount precision throughout the entire process. To address this, the idea of altering the reference voltage supplied to the model were used. This is done by comparing the required amount to be filled with what has already been filled, and then providing a reference voltage by predicting when the filling is about to stop, in order to utilize the response of the pump when the voltage is cut to fill the rest amount intsead of having an over flow. This is different from a typical PID controller, which triggers the response through its gains.
The new approach also helps the PID controller predict when the filling is about to end, allowing for a more precise filling process. To do this, a function block needs to be insert that is responsible for the selection process.

This function will take the current amount filled, the required amount to be filled and the previously determined reference, then use this information to adjust the reference voltage.

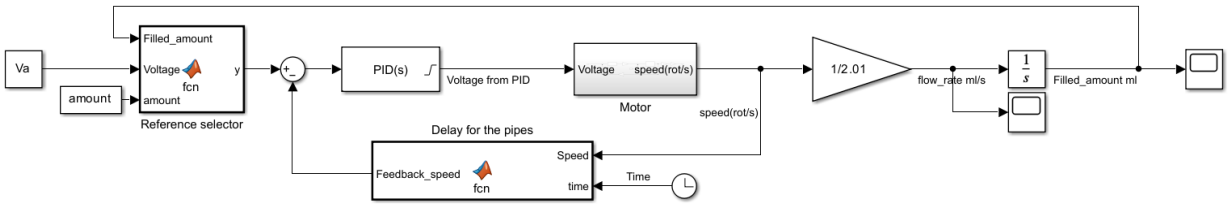


Figure 21: Model with voltage selector

The "Amount" block would be a variable in the workspace that contains the amount to be filled, just like "Va" is the reference voltage. The main function algorithm is shown in figure 22 below, for the function in MATLAB see B.3.

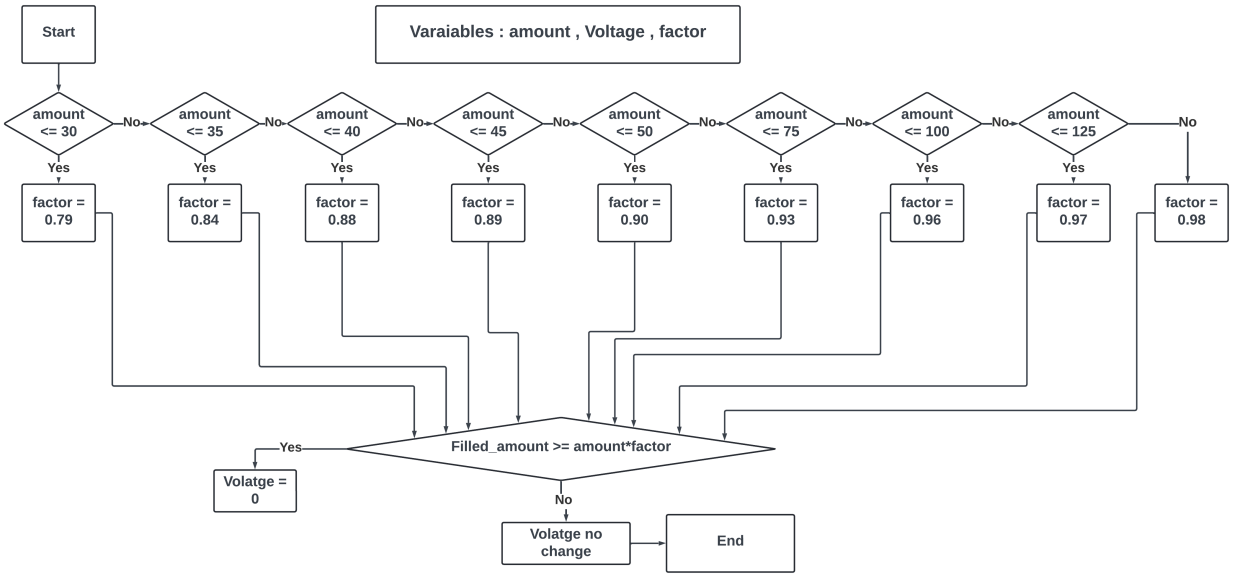
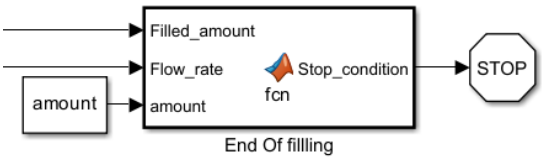


Figure 22: Voltage selector algorithm

The main purpose of this function is to check the required amount against the already filled amount, and then choose an appropriate comparison factor based on the variety of liquid volumes the pump handles. This check is done to see if the filled amount has reached a certain percentage determined by the factor. If so, the reference voltage would be set to zero, allowing the pump to predict the end of the process before it actually finishes, since the flow doesn't immediately stop when the voltage is zero.
The selection process and the ranges/factors used were determined through a series of tests and iterations, in order to achieve the best filled amount for precision, as well as optimized voltage and liquid usage. As an illustration for every amount range the factor selected gives the most accurate filling, if the factor increased this gives over flow as prediction is late, on the other hand if the factor is less then filled amount will be less than the required as input voltage will be set to zero too early to fill the required amount.

Finally in order to have full functional digital twin, here comes the role of the stop factor. It means that when the amount required to be filled is reached and the flow rate becomes zero, this means that the process of filling is done and a stop signal for this model simulation must be sent, which could be also used if all parts integration to be used for processes transitions end of filling and begin of conveyor movement or filling from tank A is done then tank B starts.




```
function Stop_condition = fcn(Filled_amount,
    Flow_rate, amount)
t=0;
if((Flow_rate<=0.01) && (Filled_amount>=amount
    *0.98))
   t=1;
end
Stop_condition = t;
```

Figure 23: End of process block

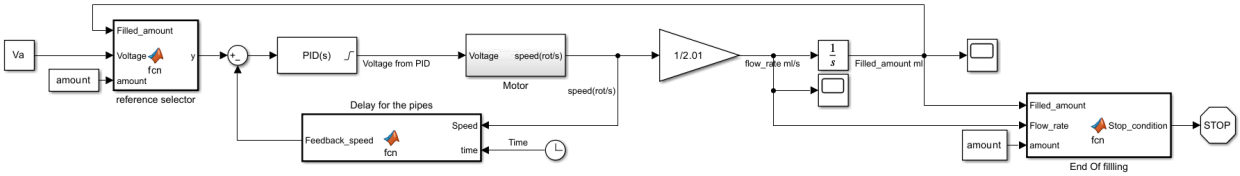


Figure 24: Final Model

After all these amendments in the model, values for the PID block became for sure not suitable so after sort of tuning the values became :

$$\text{Kp} : 12,\ \text{Ki} : 12,\ \text{Kd} : 0.1$$

These values were chosen for best signal response as an example for the small Kd, this due to it's big effect on decreasing the speed of response as it acts as damping factor and at our case the most important factor is the time, but for sure can't be zero or system might go unstable. Big values for Kp, Ki helps go stable faster and more precised with nearly zero steady state error.

3.5 ASRS 4 DOF Robotic Arm

The 4-DOF robotic arm within the ASRS system is responsible for pick-and-place operations, including picking empty bottles from the storage rack, scanning them at the RFID station, and then placing them on the conveyor. This cycle is repeated three times, with the arm accessing three different positions in the storage rack. Robotic arm is shown in figure 25 below.

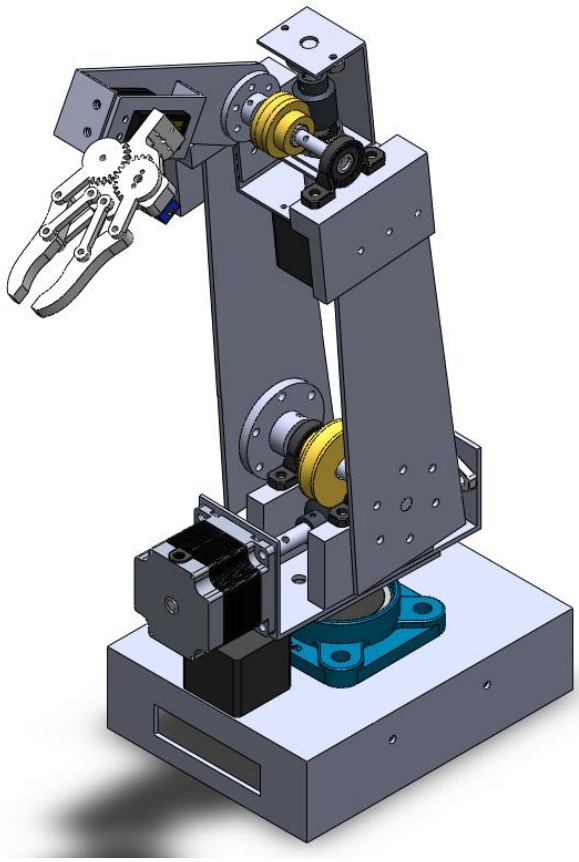


Figure 25: 4 DOF robotic arm [31]

The key specifications of the 4DOF robotic arm include:

- 4 degrees of freedom (DOF) with mechanical links measuring 250 mm, 165 mm, and 64 mm
- A maximum reach of up to 470 mm
- Powered by Nema 23 and Nema 17 stepper motors, and MG995 and SG90 servo motors
- Torque requirements of 70 Kgcm, 20 Kgcm, and 11 Kgcm for joints, with output torques of 130 Kgcm and 66.28 Kgcm
- High precision with contactless angle sensors (AS5600) for accurate positioning
- Modular and versatile design suitable for various applications, driven by an Arduino Mega 2560 microcontroller

(Krushnan, 2023).

Degrees of freedom of ASRS robot: This robotic arm has 4 degrees of freedom, which are distributed as follows:
The robotic arm features a base rotation around the vertical axis, allowing it to pivot and change the orientation of the entire arm. It also has a shoulder rotation and an elbow rotation, both around horizontal axes, which enable the arm to extend, bend, and reach different positions within its workspace. The final degree of freedom is the wrist pitch, which allows the end-effector to tilt or change the orientation of the tool or gripper attached to the arm.
This 4-DOF configuration provides the robot with a good range of motion and flexibility, making it suitable for tasks such as pick-and-place operations, simple assembly work, or material handling applications. The combination of these four rotational degrees of freedom allows the robot to position and orient the end-effector as needed to interact with its environment and carry out the required functions. The degrees of freedom of this robot are shown in figures 26 and 27.

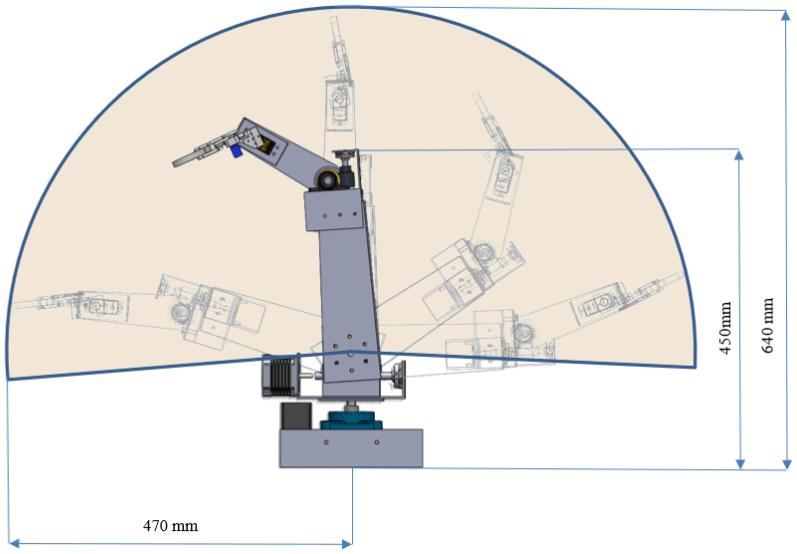
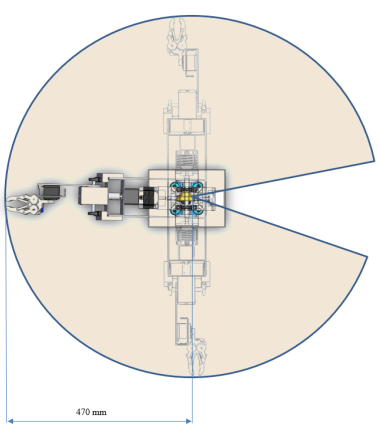
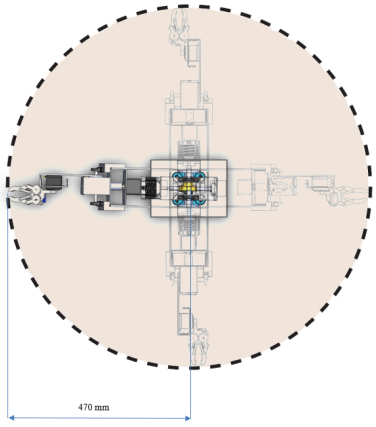


Figure 26: ASRS robotic arm degrees of freedom side view



(a) Degrees of freedom top view



(b) Degrees of freedom top view boundary

Figure 27: ASRS robotic arm degrees of freedom

To model the kinematics and dynamics of the 4-DOF robotic arm, the Denavit-Hartenberg (D-H) parameters are utilized to define the geometric relationship between adjacent links. These parameters, summarized in Table 2, are essential for the development of the robotic arm's mathematical model. Specifically, the D-H parameters provide the necessary information to transform the coordinate frames from one link to the next, which is crucial for both forward and inverse kinematics calculations.

Joint (i)	θ_i **(Joint Angle)**	d_i **(Link Offset)**	a_i **(Link Length)**	α_i **(Link Twist)**
1 (Base)	θ_1 (Variable)	d_1 (Fixed)	250 mm	0°
2 (Shoulder)	θ_2 (Variable)	0 mm	165 mm	90°
3 (Elbow)	θ_3 (Variable)	0 mm	64 mm	0°
4 (Wrist)	θ_4 (Variable)	0 mm	0 mm	0°

Table 2: Denavit-Hartenberg Parameters for the 4-DOF Robotic Arm

These parameters are then utilized within the Robotics System Toolbox in MATLAB, which provides a comprehensive set of tools and functions for modeling, simulating, and analyzing robotic systems. The toolbox allows for the creation of a detailed digital twin of the robotic arm, enabling the study of its motion, trajectory planning, and control(Mathworks, 2023). The process of developing the model for the ASRS 4-DOF robotic arm using Robotics System Toolbox in MATLAB involves the following:

Simulink Model Development: A simulink model is created using necessary blocks and their interconnections to represent the robotic arm. Simscape toolbox blocks were used to build and represent the joints and links of the robotic arm.there are 3 main blocks to be used in any model representing a robot. The first block of them is " Solver configuration" block which determines which ODE solver will be used. The second block is "World frame" which defines the coordinate system of the whole mechanism. The third block of them is "Mechanism configuration" which defines the gravity or any other forces acting on the mechanism. As shown in figure 28 below.

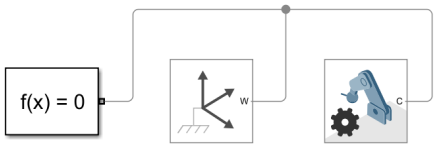


Figure 28: 3 main blocks for robot representation

Continuing with other blocks that makes the model of the robotic arm, The "World to base" block which Defines a fixed 3-D rigid transformation between two frames. A joint block which represents the joint between 2 links depending on the type of motion of the links. And a block representing the link itself and it is made by choosing the body of the link and entering the dimensions of it in the 3 axes. The model of base and link1 is shown in figure 29 below.

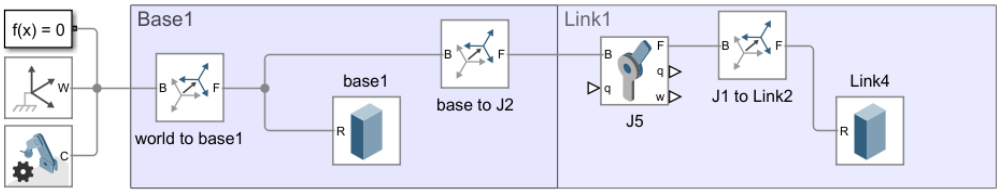


Figure 29: Base and link1 model

Following the same way the whole model for the robotic arm is built as shown in figure 72 using these main components by changing its factors depending on each link and joint.
By running the Simulink model of the robotic arm in figure 72, A model representation of the robot is generated with the dimensions entered before as shown in figure 30. These dimensions were provided from the CAD drawing of the robot. A gripper was added to represent the actual gripper installed in the robot in the plant.

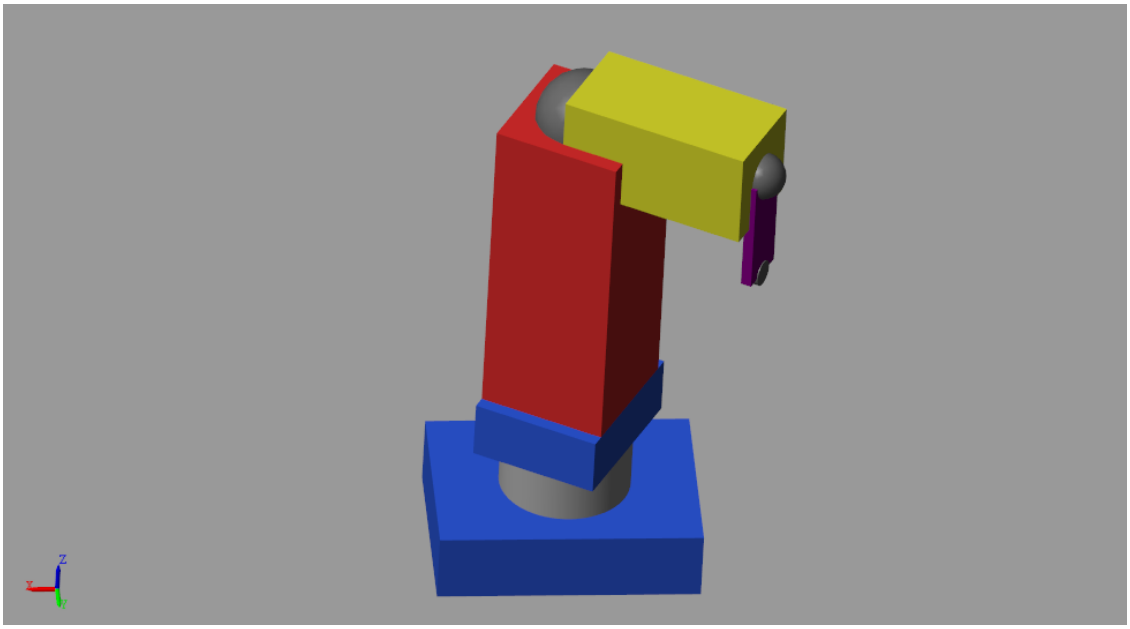


Figure 30: ASRS robotic arm Simulink model

Another block is used to generate the trajectory through which the robot will move. This block is the "Polynomial trajectory," which generates polynomial trajectories through multiple way points. The use of polynomial trajectories is often preferred when representing robot motion in Simulink because of the smoothness and continuity of the generated movement and the computational efficiency of the polynomial functions. Alternative trajectory generation methods can be computationally expensive and less efficient. Some points which specify the trajectory should be generated in an array in a Matlab code and then fed to the polynomial trajectory block. A time interval should also be specified in the code and fed to the block. Every point in the trajectory should have a corresponding time point. This block is shown in figure 31 below.

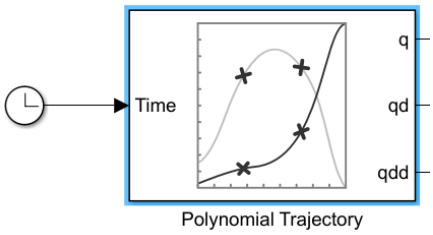


Figure 31: Polynomial trajectory block

A code which generates the trajectory in a specific time interval is below. This code will be illustrated in details in validation section 4.3.

```
Ts = 0.001;
[DOF4_Arm, ArmInfo] = importrobot ('TempARMS');

Hp = [0.2 0 0.21]; % Home position
Rp1 = [-0.3 -0.18 0.12]; % Rack position 1
Rp1a = [-0.3 -0.18 0.14]; % Rack position 1 approach
Rp2 = [-0.3 -0.12 0.12]; % Rack position 2
Rp2a = [-0.3 -0.12 0.14]; % Rack position 2 approach
Rp3 = [-0.3 -0.06 0.12]; % Rack position 3
Rp3a = [-0.3 -0.06 0.14]; % Rack position 3 approach
Rt = [-0.1 -0.06 0.2]; % Rack transition helps in trajectory genration accuracy
Rfid = [0.2 -0.05 0.175]; % RFID position
Bc = [0.36 0 0.14]; % Bottle on conveyor
Bca = [0.36 0 0.16]; % Bottle on conveyor approach

trajectory =
[Hp; Rt; Rp1a; Rp1; Rp1; Rt; Rfid; Rfid; Bca; Bc; Bc; Bca; Hp;
Rt; Rp2a; Rp2; Rp2; Rt; Rfid; Rfid; Bca; Bc; Bc; Bca; Hp;
Rt; Rp3a; Rp3; Rp3; Rt; Rfid; Rfid; Bca; Bc; Bc; Bca; Hp ];
Totaltime =
[0 1.5 2 2.5 3 4 5 6 6.5 7 7.5 8 9
10 11.5 12 12.5 13 14 15 16 16.5 17 17.5 18 19
20 21.5 22 22.5 23 24 25 26 26.5 27 27.5 28 29 ];

wp = transpose(trajectory);
t = Totaltime;
```

Moving on with the the trajectory generated, An inverse kinematics block is used to generate the angles of the joints. This block takes the trajectory as an input and gives the angles of the joints as output. Another 2 inputs to this block are "weights" and "initial guess" where the weights is negligible here so a zero vector is given as input and for the initial guess it was assumed for angles to be zero aswell. The inverse kinematics block is shown in figure 32.

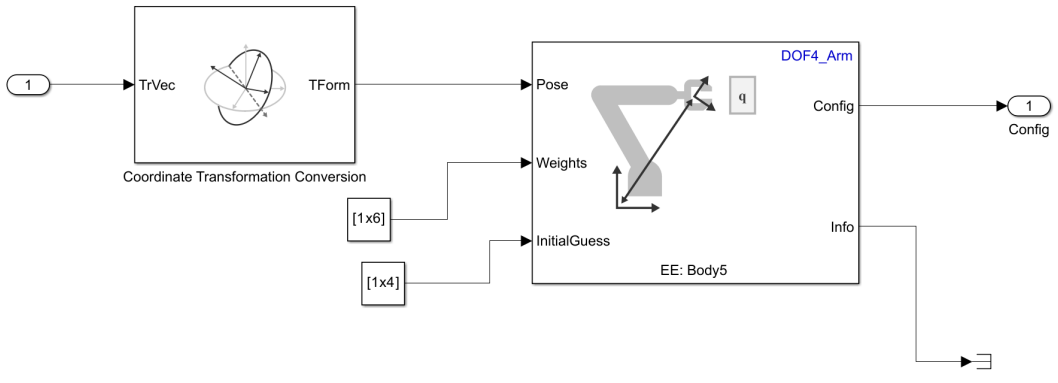


Figure 32: Robotic arm inverse kinematics

Then the angles are given as inputs to the model of the robot. Every joint in the robot takes the theta of the given path and returns the theta of the actual path taken by it. So we can visualize the path of the joint across some period of time and compare it to the given path. We can also visualize the angular velocity of the joint and angular acceleration through a period of time.This is shown in figure 33.

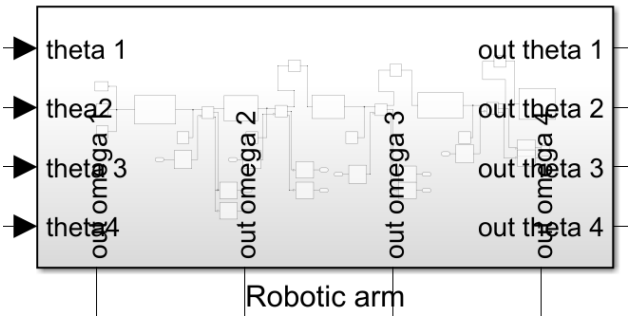


Figure 33: Inputs and outputs of the robotic arm's model

After that the angles of the joints are provided from the model block of the robot and given to the forward kinematics block. The forward kinematics block takes the angles of the joints as inputs and gives the position of the Tool Center Point (TCP) in the XYZ coordinates, where the TCP is the reference base for these coordinates. This is shown in figure 34.

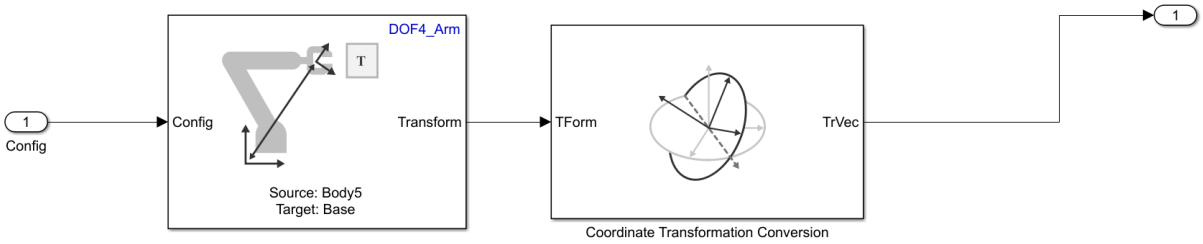


Figure 34: Robotic arm forward kinematics

The angles of the joints are limited to specific range to limit the free movement of the links. Every joint has its own limits to specify its range of movement. The first joint is limited in range from 30 to 330 degrees. The second joint is limited in range from 100 to 170 degrees. The third joint is limited in range from 20 to 140 degrees. The fourth joint is limited in range from -60 to 50 degrees. All these limitation are validated from iterative experiments on the model created as the original limits of the robotic arm were used in this model but the robot did not move in the real way it should be. So these limits were used to achieve the real movements of the robot. See limits of the joints in the appendix C.2.

Finally, the actual path of the robot in the XYZ coordinates are provided and can be visualized and compared to the given path. The whole representation of the movement model of the robot is shown in the figure 35.

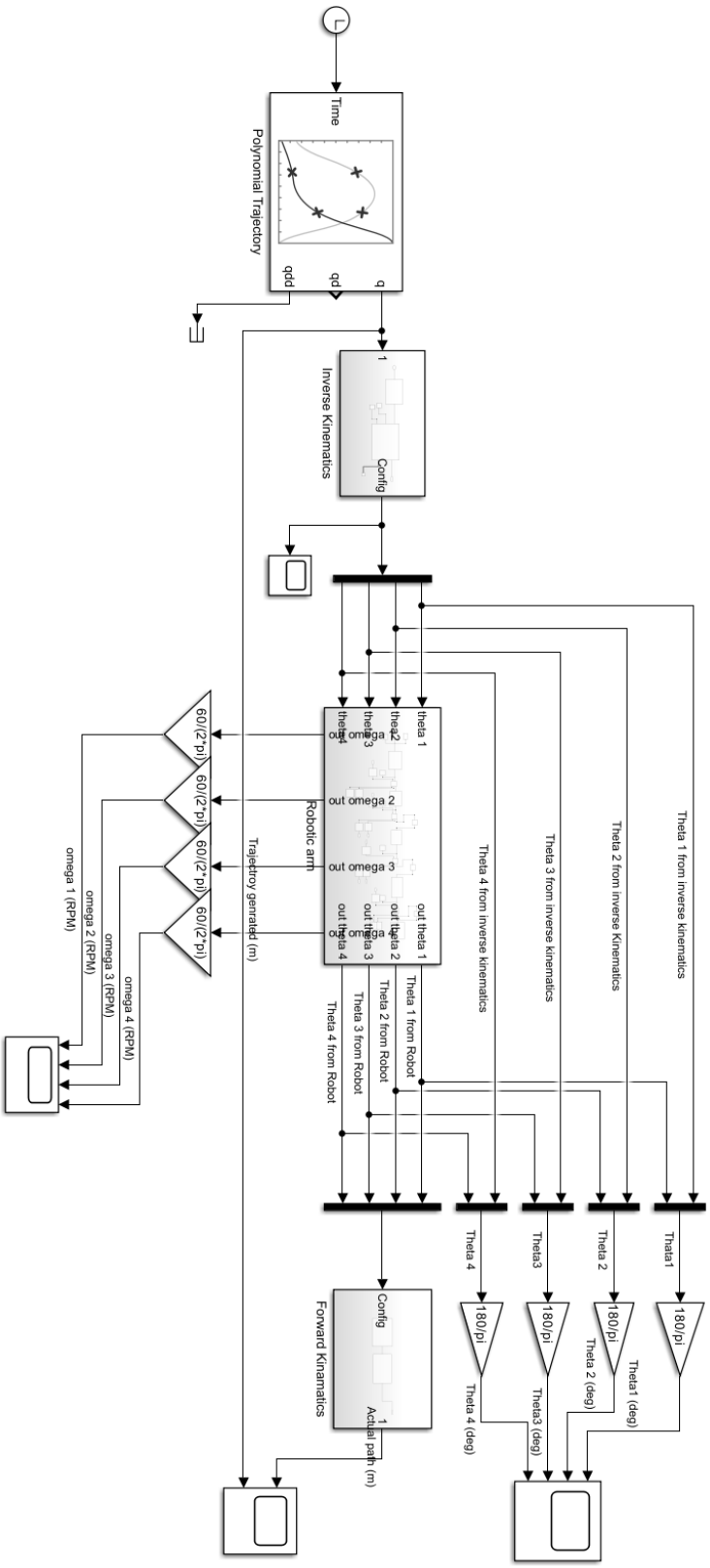


Figure 35: ASRS movement and kinematics

4 Validation

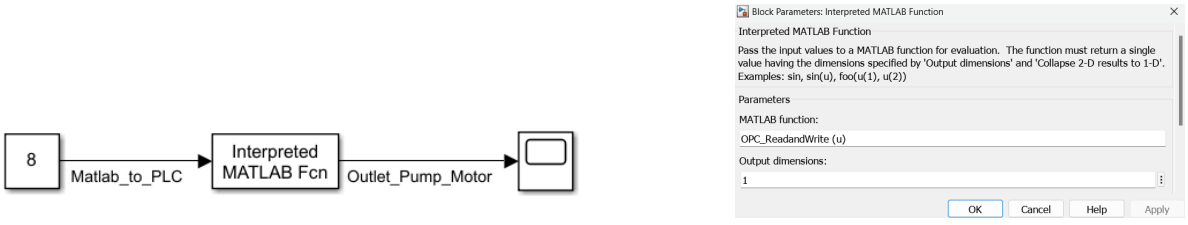
In this chapter the main target is to validate all the concepts introduced in the previous chapter, which is a crucial step in the whole story built for the digital twin developed. First part would be to check the validity of the OPC UA communication established between the Siemens PLC and MATLAB environment, Second part would be for the filling plant model validation and the new methods introduced. The last part is the validation of movement and trajectory for the 4 DOF robotic arm of the ASRS system.

4.1 OPC UA read and write

This section will focus on validating the OPC UA communication between the Siemens PLC and MATLAB environment. The aim is to ensure that the established connection is allowing reliable data exchange through reading and writing operations. Additionally, limitations encountered during this process will be encountered, particularly regarding the data transfer capacity between the PLC and MATLAB.

```
function [x] = OPC_ReadandWrite ( y )
persistent init_Server ;
persistent init_Nodes1 ;
persistent uaClient ;
persistent Var_Node_In ;
persistent Var_Node_Pumpvalve ;
init_Nodes1 = 0 ;
if (isempty(init_Server))
    init_Nodes1 = 0;
    init_Server = 0;
end
if init_Server == 0
    init_Server = 1;
    uaClient = opcua ( '169.254.137.8' ,4840);
    connect ( uaClient );
end
if uaClient.isConnected == 1 && init_Nodes1 == 0
    init_Nodes1 = 1 ;
    Var_Node_In = opcuanode (4 , 47 , uaClient );
    Var_Node_Pumpvalve = opcuanode (4 , 12 , uaClient );
end
if uaClient.isConnected ==1 && init_Nodes1 == 1
    a = readValue ( uaClient , Var_Node_Pumpvalve ) ;
    writeValue ( uaClient , Var_Node_In , y )
end
x = a;
end
```

This function is read and write in order to validate the proper connection and check readings and writings, where server in address 47 is the 'Matlab_to_PLC' as refereed previously which will be written by Matlab to PLC, while server in address 12 is the status of pump whether closed or opened (Logical True or False).

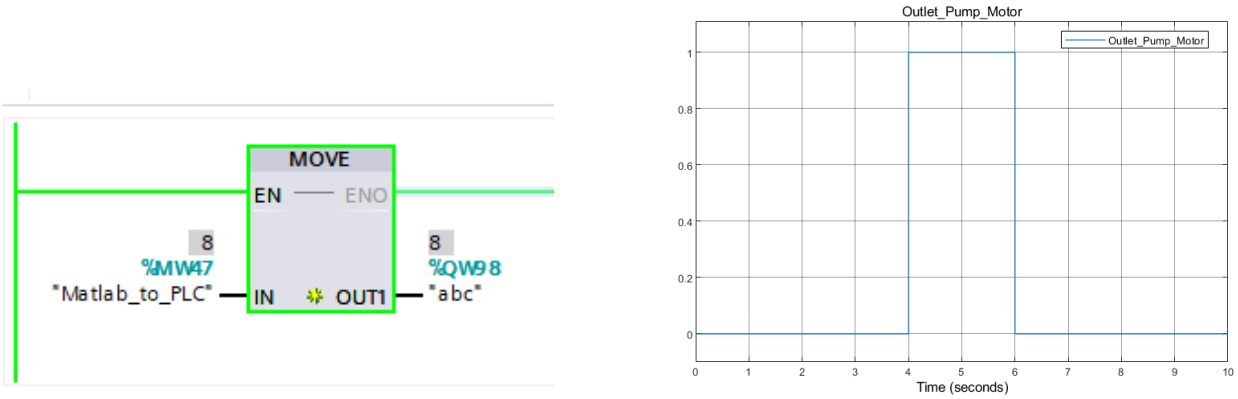


(a) Simulink model

(b) Interpreted MATLAB Function

Figure 36: Simulink model to run the read write function continuously

By running this Simulink model in figure 36a continuously to write 8 as constant to 'Matlab_to_PLC' server and receiving the reading of 'Outlet_Pump_Motor'.
Results can be seen on both the scope of Simulink and Tia of the PLC through block monitoring to see value of any variable inside the blocks.



(a) Server variable in TIA portal

(b) Status of pump on Simulink scope

Figure 37: Validation of read and write

After validating the capability of the OPC protocol to read from and write on MATLAB environment, it is important to highlight the limitation of the OPC protocol with the PLC TIA portal is that it can communicate with multiple clients but the problem is that it can maximum deal with five servers which corresponds to five data variables, which limits the amount of data transfer between the PLC and MATLAB.

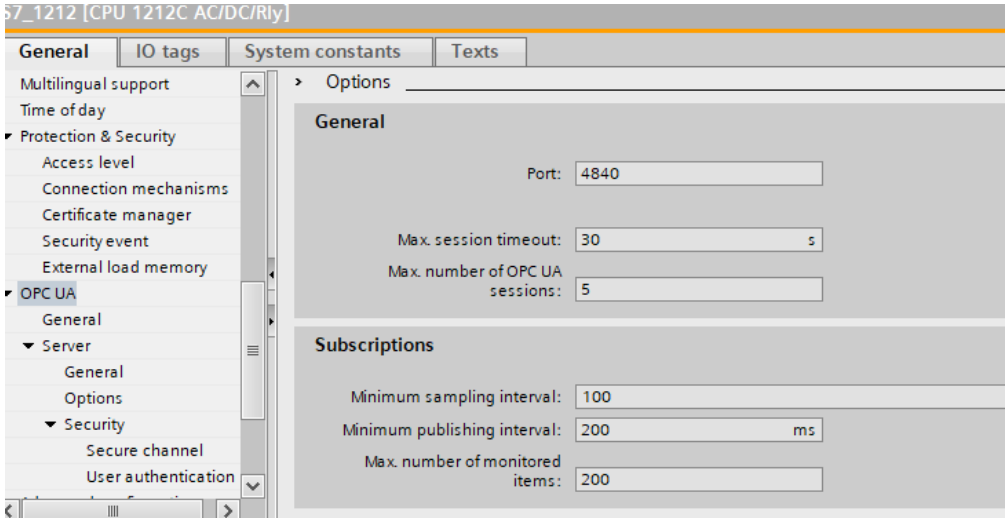


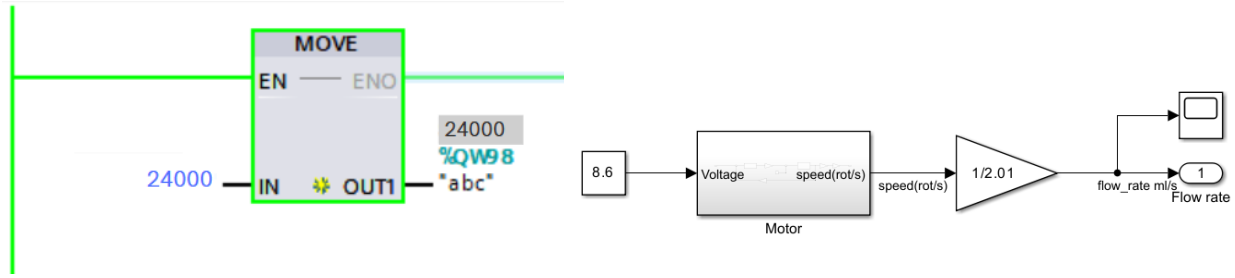
Figure 38: OPC limitations

4.2 Fluid Filling Plant Model

For the validation of the digital twin model for the fluid filling plant controlled by a Siemens PLC, a two-stage approach will be employed.
The first stage will focus on validating the digital twin model itself, comparing its performance and behavior against the actual physical plant. This initial validation will ensure that the digital twin accurately represents the real-world system.
The second stage of the validation process will evaluate the effect of developing the digital twin model with the PID controller and other methods discussed in the previous chapter. This step will build upon the established reliability of the digital twin model, ensuring that the enhancements made to the model truly enhance the representation of the actual hardware.
By following this two-stage validation approach, the aim is to minimize the need for extensive hardware-in-the-loop testing, which can hinder the evolution and refinement of the digital twin model. Instead, the digital twin will be validated first through tests with hardware using OPC UA communication with the physical Siemens PLC, and then the subsequent tests and implementation of new concepts can be carried out directly on the digital twin within the MATLAB environment.
This systematic validation process will help to establish the digital twin as a reliable and trustworthy representation of the filling plant, enabling the efficient development and evaluation of control strategies and optimization methods without the constraints of continuous hardware interactions.

4.2.1 Stage 1: Model Validation

At this stage a step input of 8.6V will be supplied to the pump and the digital twin and the floe rate will be monitored as an output for both. For pump voltage will be supplied from the PLC through TIA portal by direct input to avoid any interference from any other factors, while for the model on Simulink it will be given step input through a constant of 8.6. In TIA portal input will be scaled from 0-10V to 0-27648 to be given as an analog output to te PLC, though 8.6 will be approximately 24000.

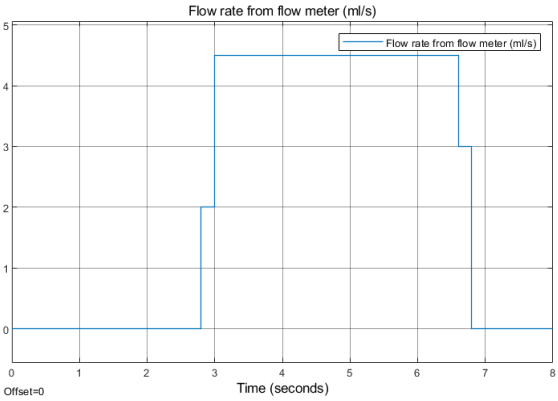


(a) Step input on TIA portal

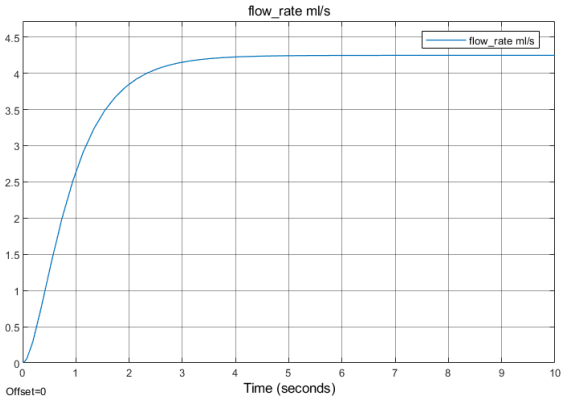
(b) Step input on Simulink model

Figure 39: Step input on both hardware and model

Next step is to receive the readings of the flow meter in order to build the graph of the flow rate, this was carried using OPC to take send the reading if the flow meter stored in the TIA portal to the MATLAB using Interpreted function with OPC read function and recieve the readings through a graph in Simulink scope. See B.4.



(a) Response from flow meter



(b) Response from digital twin

Figure 40: Step input reponse on both hardware and digital model

The graph above shows that the value of the flow rate is nearly the same from the digital twin and from the flow meter, the difference mainly in the response and this is due to the factor of the piping where fluid takes roughly 2 seconds to get out of pipes in addition to the influence of the communication to be discrete with time lag of approximate 0.5 seconds, which prevents the graph to be smoothly continuous.
It is important to note that the flow meter has already been validated through work conducted in a previous master thesis by Shah (2023). In this study, the readings from the flow meter were compared with actual measurements from the filling process, demonstrating high accuracy.

4.2.2 Stage 2: New Methods Validation

At this stage the main target will be to check the validity of the whole plant model in the figure below through two factors, first validity with respect to different input reference Voltage, second validity with respect to different filling amounts.

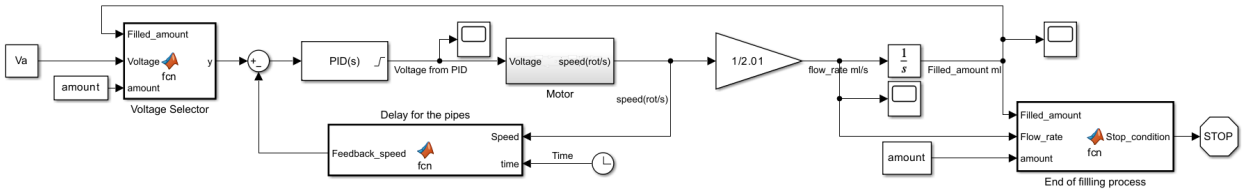


Figure 41: Fluid filling plant final digital model

The whole plant at this stage contains all what were built together starting from the pump model, then the PID block, the Voltage selector function and finally the end function, where all work together to reach the final goal of filling the required amount with the highest precision. However, each component in the model has a unique factor that determines its accuracy. For instance, the success of the voltage selector function hinges on filling the required amount with the highest precision. Meanwhile, the PID's performance is measured by the response of the voltage signal supplied to the plant. Lastly, the end function is considered successful when the simulation stops at the moment the filling is complete, indicating the flow rate has reached zero and the process has finished as intended, which provides also the exact moment the simulation stops and the whole process.

- **Voltage reference range:**
  From the test done in section 3.4.3, it was shown that the pump responds the best starting from 7V. for that case validation of the model will be on range from 7V to 9V, and the reason for choosing 9V not 10V is that it will not be practical to set the pump running at maximum voltage as a reference leaving this high range to be supplied by the PID block depending on the need. In order to test this range 3 values are chosen which are 7V, 8V and 9V, while having same target to fill 200ml.

  – **Case 1:** Using 7V as input reference voltage

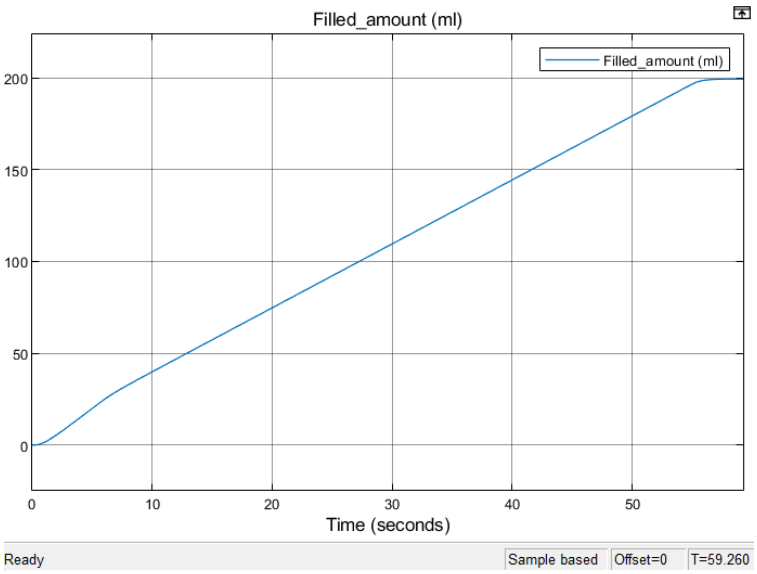
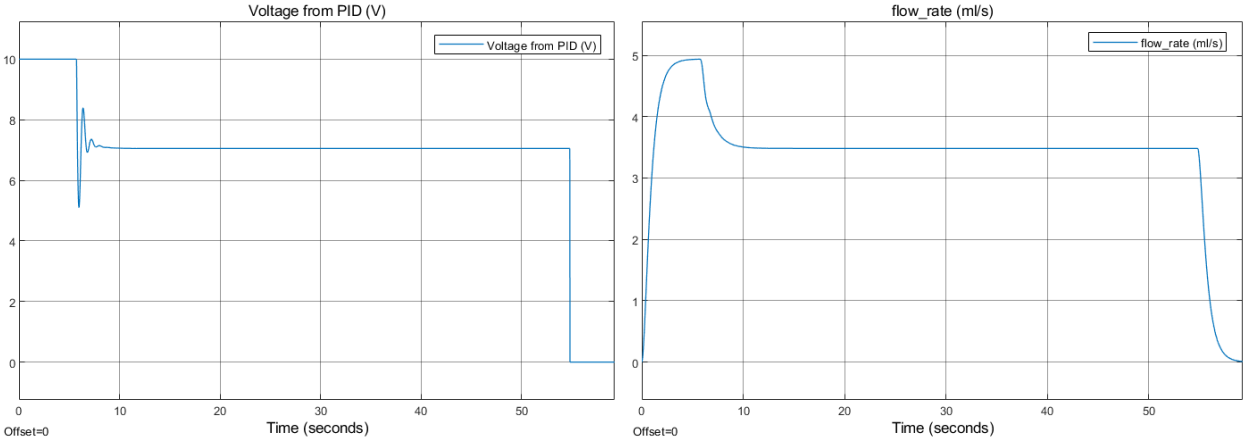


Figure 42: Filled amount graph case 1



(a) Voltage signal from PID block

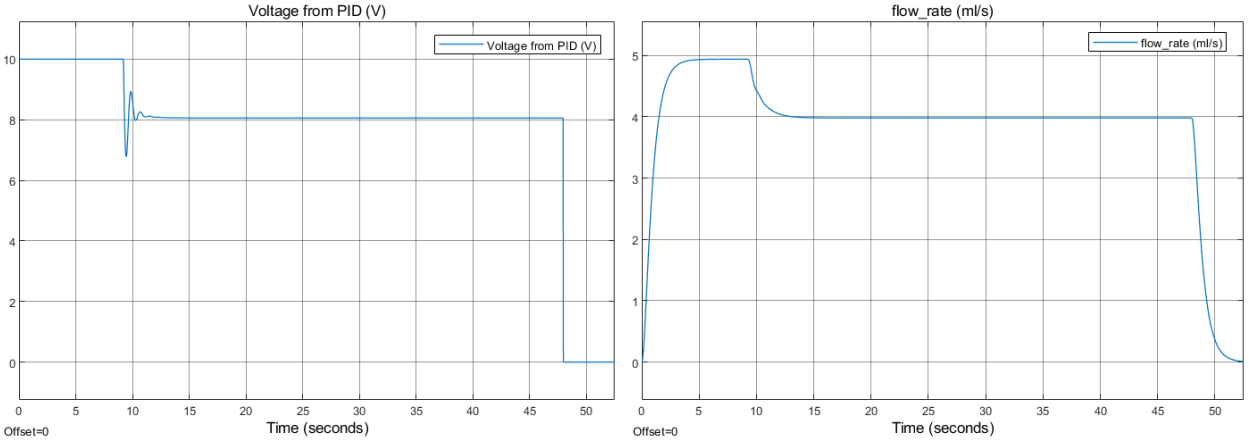
(b) Flow rate graph

Figure 43: Voltage signal and flow rate graphs case 1

From the graph in figure 42, it can be seen the required amount is filled accurately and also the process stops exactly at T = 59.26s, meanwhile the flow rate in in figure 43b reached zero, in addition to the voltage signal n figure 43a coming from the PID block, which gives maximum voltage at the beginning then starts to go down to the reference voltage, which is the case to get over piping and develop constant flow rate.

– **Case 2:** Using 8V as input reference voltage
For that input, difference in the filled amount graph in figure 44b was only in the slope as filling process ends this time exactly at T = 52.2s, with accurate filling as well. See B.5.

While for the voltage graph shown in figure 44a below response didn't differ much settling down from 10V which is supplied at the beginning to 8V at this case, for the flow rate, it reached zero at the end of the process after filling the required amount successfully.

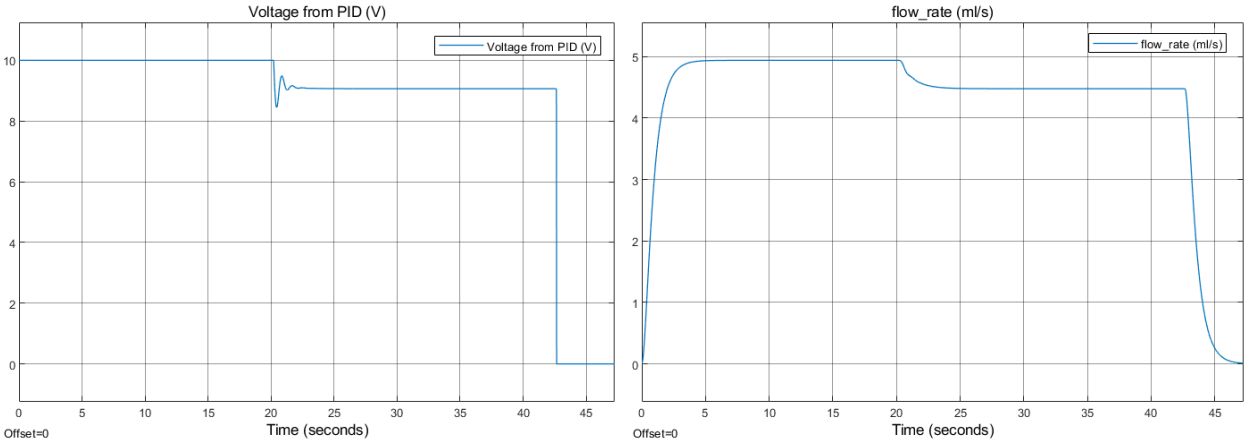


(a) Voltage signal from PID block (b) Flow rate graph

Figure 44: Voltage signal and flow rate graphs case 2

– **Case 3:** Using 9V as input reference voltage
For that input, difference in the filled amount graph in figure 45b was only in the slope as filling process ends this time exactly at T = 47.26s, with accurate filling as well. See B.5.

While for the voltage graph shown in figure 45a below response didn't differ much settling down from 10V which is supplied at the beginning to 9V at this case, for the flow rate, it reached zero at the end of the process after filling the required amount successfully.



(a) Voltage signal from PID block (b) Flow rate graph

Figure 45: Voltage signal and flow rate graphs case 3

The main difference in the 3 previous cases can be highlighted to be the time of filling, which decreases by increasing reference voltage, other than that the PID behaves the same way in all cases and the flow rate responds to the voltage supplied, reaching zero at the end of the process.

- **Filling amount range:**
  For testing with amount range the case will be the opposite as the input reference voltage will be constant 8.6V, while range of filling will be different from 30 till 200ml. For that range values where chosen to be 30ml, 80ml, 140ml and 200ml.

  – **Case 1:** Filling amount to be 200 ml.

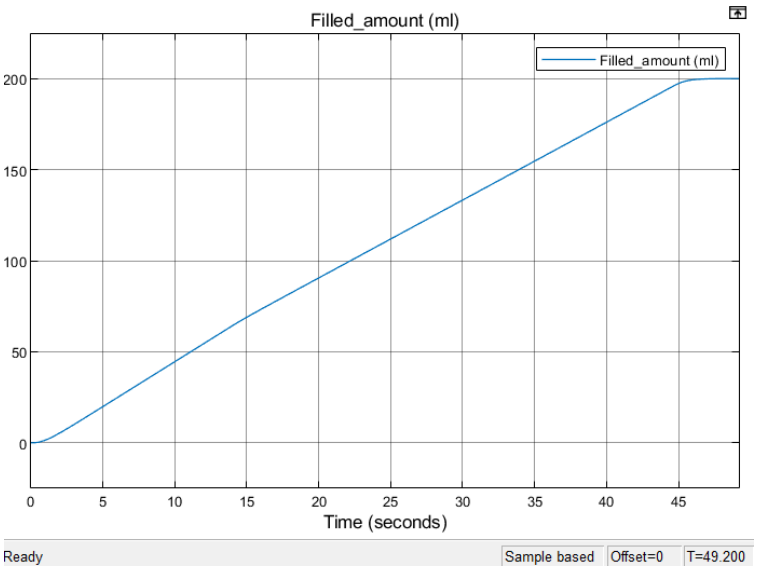
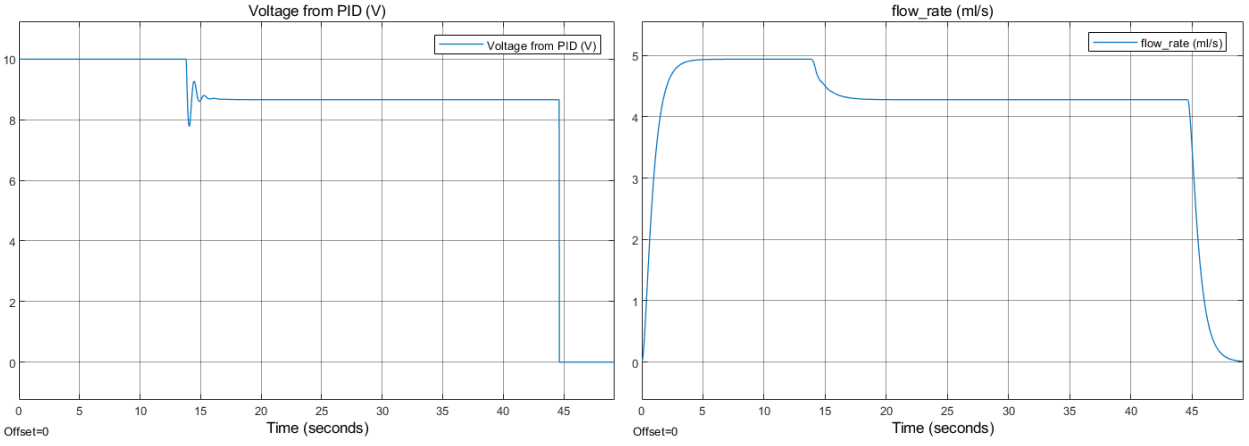


Figure 46: Filled amount graph case 1



(a) Voltage signal from PID block

(b) Flow rate graph

Figure 47: Voltage signal and flow rate graphs case 1

For this case as shown in figures 46 and 47, response is much similar to previous test where filling stops at T = 49.2s and 200 ml required is filled accurately at the end, and the flow rate reaches zero when the process ends.

– **Case 2:** Filling amount to be 140 ml.
For this case process ends at T = 35.46s, with filling this random amount accurately as shown in figure 48. Also for the voltage signal and flow rate graphs there weren't much difference compered to previous tests where the flow stops at the end time, only the shape of the graphs differs due to the difference in the filled amount and filling time. See B.6.

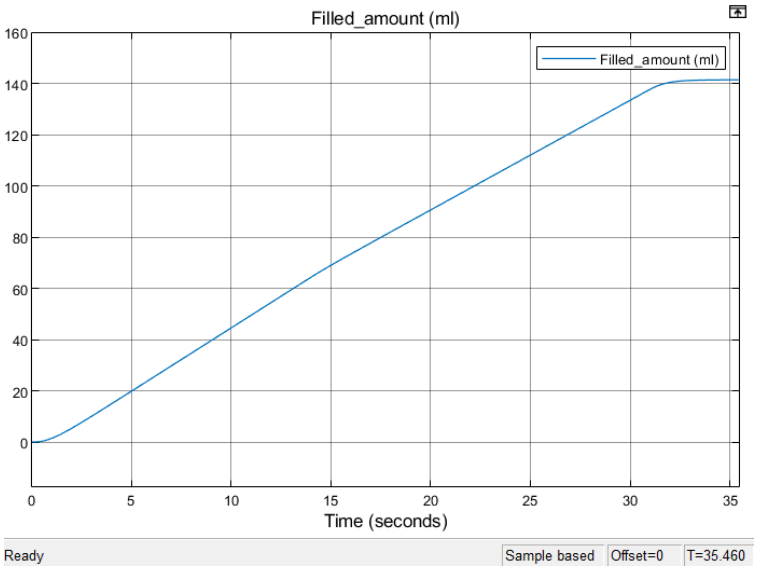


Figure 48: Filled amount graph case 2

– **Case 3:** Filling amount to be 80 ml.
For this case process ends at T = 21.36s, with filling the required amount accurately as in figure 49. For the voltage signal and flow rate graphs, the behaviour was the same as the previous case. See B.6.

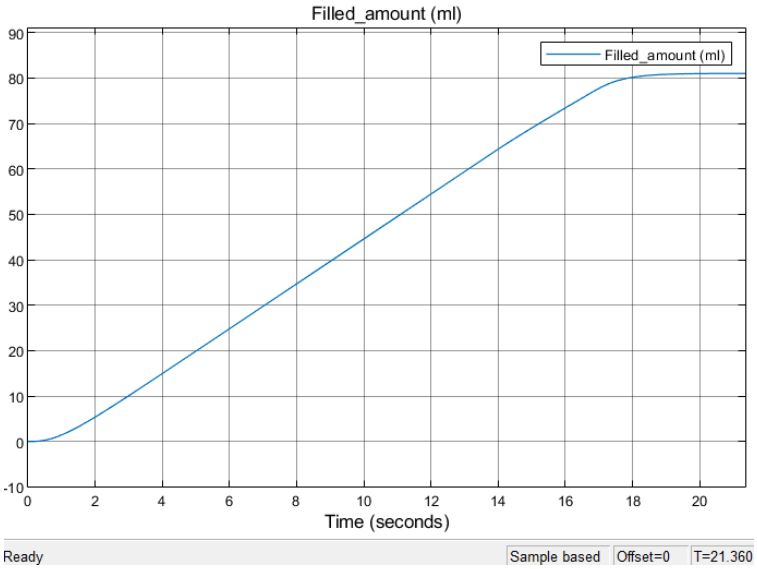


Figure 49: Filled amount graph case 3

– **Case 4:** Filling amount to be 30 ml.

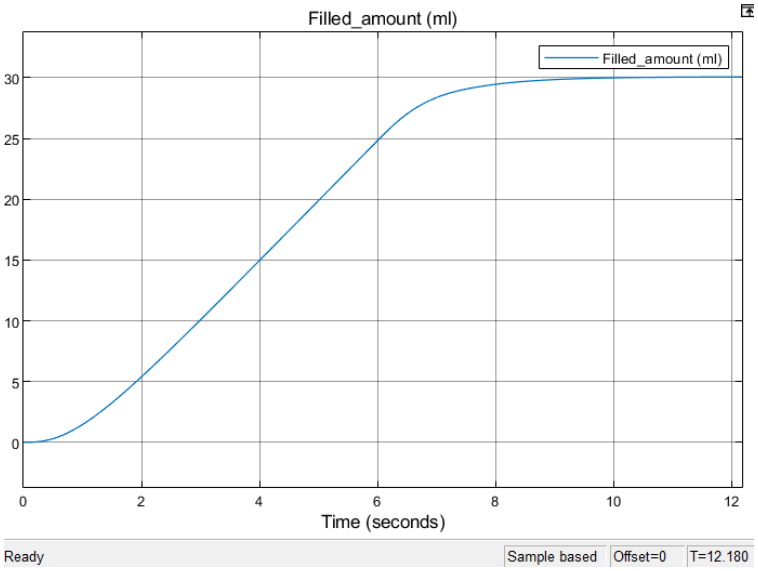
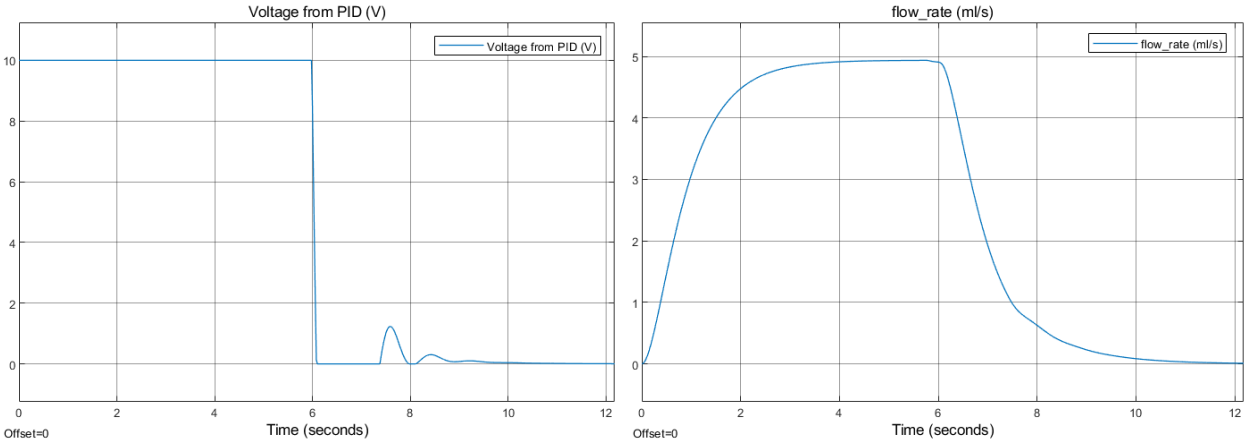


Figure 50: Filled amount graph case 4



(a) Voltage signal from PID block

(b) Flow rate graph

Figure 51: Voltage signal and flow rate graphs case 4

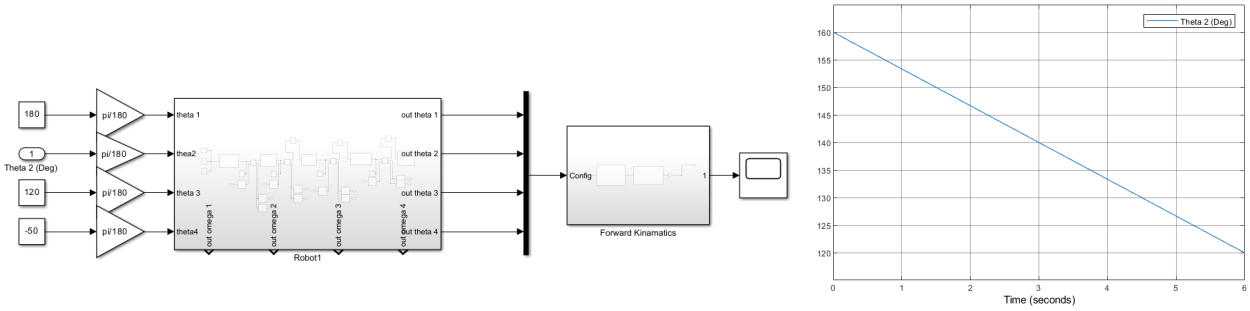
The last case were no much difference in the filled amount behaviour in figure 50 as the required amount is filled accurately at T = 12.18s, while the difference is mainly at the voltage signal in figure 51a this is due to the PID effect and the small amount required to be filled filling approached to the end before the stage of settling the input voltage to the reference, which means that this 30 ml where filled without the influence of the reference voltage, also the small oscillations after settling at zero is due to the PID is having a feedback with small amount of filled but this doesn't effect on the flow rate and as seen in it's graph in figure 51b it reached zero as well at the end of the process.

4.3 ASRS 4 DOF Robotic Arm

The validation of the Simulink model for the 4 DOF robotic arm, depicted in figure 35, involves two key steps: ensuring that the model accurately mirrors the hardware and validating the ability to visualize the entire trajectory performed by the hardware using the trajectory generation and inverse kinematics blocks in the model.

4.3.1 Model and Hardware Performance Comparison

Firstly, to validate that the model performs similarly to the hardware, a simple movement test was conducted on the second joint of the robotic arm. The video referenced in figure 53 shows the hardware performing a movement of the second joint from approximately 160° down to 120° over 6 seconds. This movement was replicated in the Simulink model by providing direct input for fixed positions for all joints except the second, which was given a signal input ranging from 160 to 120 over the same duration, as illustrated in figure 52b. Figure 52a shows the Simulink model setup with direct angle inputs, bypassing the trajectory generation and inverse kinematics blocks.



(a) Model with direct input

(b) Theta 2 provided to the model

Figure 52: Simulink model with direct angles input

The comparison between the hardware and the Simulink model, as shown in the videos in figure 53, indicates that both performed the required motion in an identical manner. This simple test demonstrates the initial validation of the model's performance relative to the hardware.

Figure 53: Robotic arm movement vs Simulink model movement [32]

4.3.2 Whole Trajectory Validation

To validate the model for performing the entire trajectory, a MATLAB code was written to specify the arm's path over a cycle of 29 seconds, covering movements from the home position to various rack positions, the RFID, and the conveyor. This cycle was repeated three times with different pickup positions.

```
Ts = 0.001;
[DOF4_Arm, ArmInfo] = importrobot ('TempARMS');

Hp = [0.2 0 0.21]; % Home position

Rp1 = [-0.3 -0.18 0.12]; % Rack position 1
Rp1a = [-0.3 -0.18 0.14]; % Rack position 1 approach

Rp2 = [-0.3 -0.12 0.12]; % Rack position 2
Rp2a = [-0.3 -0.12 0.14]; % Rack position 2 approach

Rp3 = [-0.3 -0.06 0.12]; % Rack position 3
Rp3a = [-0.3 -0.06 0.14]; % Rack position 3 approach

Rt = [-0.1 -0.06 0.2]; % Rack transition helps in trajectory genration accuracy

Rfid = [0.2 -0.05 0.175]; % RFID position

Bc = [0.36 0 0.14]; % Bottle on conveyor
Bca = [0.36 0 0.16]; % Bottle on conveyor approach

trajectory =
[Hp; Rt; Rp1a; Rp1; Rp1; Rt; Rfid; Rfid; Bca; Bc; Bc; Bca; Hp;
Rt; Rp2a; Rp2; Rp2; Rt; Rfid; Rfid; Bca; Bc; Bc; Bca; Hp;
Rt; Rp3a; Rp3; Rp3; Rt; Rfid; Rfid; Bca; Bc; Bc; Bca; Hp ];
Totaltime =
[0 1.5 2 2.5 3 4 5 6 6.5 7 7.5 8 9
10 11.5 12 12.5 13 14 15 16 16.5 17 17.5 18 19
20 21.5 22 22.5 23 24 25 26 26.5 27 27.5 28 29 ];

wp = transpose(trajectory);
t = Totaltime;
```

The trajectory dimensions were estimated through manual measurements, ensuring the model's suitability for general use within the robotic arm's range. Figure 54 shows the visualization of the entire process in Simulink, demonstrating the model's accuracy in replicating the actual movement.

Figure 54: Full trajectory in Simulink [33]

The video in Figure 54 shows a visualization in Simulink for the whole process with approximate locations for every position encountered by the 4 DOF robotic arm through its process of handling bottles from the rack positions, scanning at the RFID, and finally placing them on the conveyor.

Besides the motion shown in the video in figure 54, the generated trajectory closely matches the actual trajectory, as shown in figure 55, which means that the robotic arm was able to follow the whole generated trajectory without being stuck due to any extra limits or insufficient time period given. This confirms the validity of the trajectory generation, the inverse kinematics block, and their compatibility with the model and the forward kinematics.

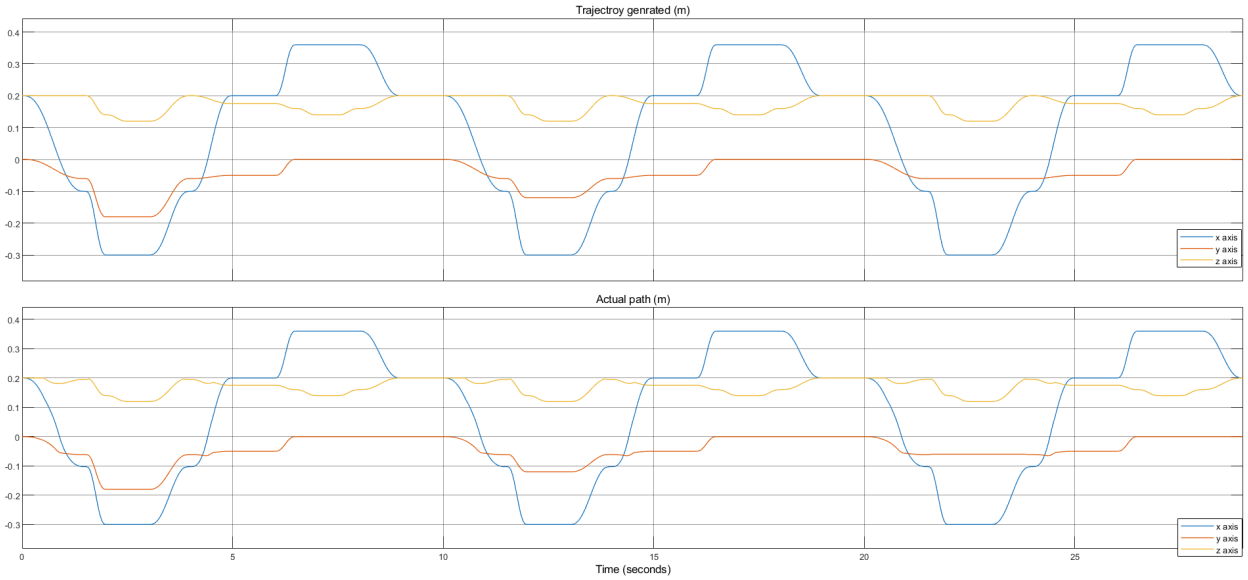


Figure 55: ASRS generated trajectory compared to actual trajectory in Simulink model

The input angle of the first joint matches the output angle, as evidenced by the overlapping graph lines shown in Figure 56. This consistency is observed for all joints, with detailed graphs provided in Appendix C.3.

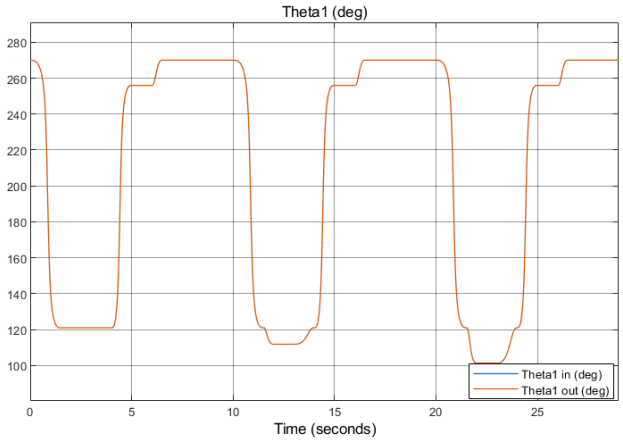


Figure 56: Angle of first joint

These results demonstrate that the Simulink model accurately replicates the behavior of the hardware, validating its effectiveness for simulating the ASRS 4 DOF robotic arm.

5 Conclusion

This thesis has successfully demonstrated the creation and implementation of a comprehensive digital twin for an Industry 4.0 plant. Utilizing advanced modeling, optimization, and simulation techniques, a robust virtual representation was developed to accurately reflect the complex systems, processes, and inter-dependencies within the plant. The use of MATLAB and Simulink has been crucial, providing a powerful platform for building detailed physical models, implementing optimization algorithms, and seamlessly integrating the digital twin with the plant's operational systems. The inclusion of Simscape enabled the creation of physics-based models that closely mimic the real-world behavior of the 4-DOF robotic arm. Additionally, the Robotics Systems Toolbox was essential for incorporating the plant's robotic automation into the digital twin, ensuring a comprehensive and accurate virtual representation through simulating the whole trajectory done by the robotic arm in 29 seconds, which may vary depending on robotic arm limits.

A key aspect of integrating the digital twin with the plant's operational systems was the use of the OPC UA (Open Platform Communications Unified Architecture) communication protocol. Establishing a robust OPC UA connection between the PLC (Programmable Logic Controller) and the MATLAB-based digital twin with a max delay of 0.5 seconds allowed for seamless data exchange and real-time synchronization between the virtual and physical environments. This integration enabled the digital twin to monitor and respond to changes in the plant's operational status, production data, and equipment performance, enhancing the accuracy and responsiveness of the virtual model.

While this thesis has highlighted the immense potential of digital twins in the Industry 4.0 context, it also underscored the importance of ongoing maintenance and updates to the virtual model. As the physical plant evolves and new technologies are implemented, the digital twin must be continuously refined and expanded to maintain its accuracy and relevance. This may involve adding more control blocks such as the PID block introduced with values of Kp : 12, Ki : 12, Kd : 0.1, incorporating new sensors and equipment, and utilizing advanced optimization techniques to enhance the model's predictive capabilities.

Future Scope:

To further the impact and capabilities of the digital twin, several avenues for future research and development can be pursued:

- **Enhanced Predictive Maintenance**: Integrating machine learning algorithms to predict equipment failures before they occur, thereby reducing downtime and maintenance costs.
- **IoT Integration**: Incorporating more IoT devices to gather real-time data from various points within the plant, enhancing the digital twin's ability to monitor and optimize plant operations.
- **Advanced Control Strategies**: Developing and implementing more sophisticated control algorithms, such as model predictive control (MPC), to improve the efficiency and responsiveness of the plant's operations.
- **Scalability**: Expanding the digital twin to cover larger and more complex systems within the plant or across multiple plants, enabling more comprehensive operational insights.
- **User Interface Improvements**: Creating more intuitive and interactive user interfaces for operators and engineers to interact with the digital twin, making it easier to use and interpret data.

To sump up, this thesis has successfully delivered a digital twin for the Industry 4.0 plant, showcasing the trans-formative potential of this technology in driving operational excellence and innovation within the manufacturing sector. The integration of MATLAB, Simulink, and their specialized toolboxes, along with seamless OPC UA communication between the digital twin and the plant's PLC, has been instrumental in this achievement, demonstrating the power of these software tools and communication protocols in developing complex, multi-domain virtual models. Future work will continue to build on this foundation, exploring new technologies and methodologies to further enhance the capabilities and applications of digital twins in industrial settings.

References

[1] Schwab, K. (2016). The Fourth Industrial Revolution. World Economic Forum. [https://www.weforum.org/agenda/2016/01/the-fourth-industrial-revolution-what-it-means-and-how-to-respond/].

[2] i-Scoop. (2020). Industry 4.0: The Fourth Industrial Revolution - Guide to Industrie 4.0. Retrieved from [https://www.i-scoop.eu/industry-4-0/].

[3] IBM. (2020). What is a Digital Twin? Retrieved from [https://developer.ibm.com/articles/what-are-digital-twins/].

[4] Siemens. (2021). Integrating the digital twin with the plant's control systems, production management software, and other relevant IT infrastructure to enable real-time monitoring and data exchange. [https://www.sw.siemens.com/en-US/technology/digital-twin/].

[5] Parrott, A., & Warshaw, L. (2017, May 12). Industry 4.0 and the digital twin: Manufacturing meets its match. Deloitte Insights. [https://www2.deloitte.com/xe/en/insights/focus/industry-4-0/digital-twin-technology-smart-factory.html].

[6] Programmable Logic Controllers (PLCs). Unitronicsplc. Retrieved from [https://www.unitronicsplc.com/what-is-plc-programmable-logic-controller/].

[7] Automation parts website, Siemens PLC. Retrieved from [https://de.automationparts.com/siemens/automation-plc-and-process-control/6ES72121HE310XB0].

[8] Siemens. (2023). Totally Integrated Automation (TIA) Portal [https://www.siemens.com/global/en/home/products/automation/industry-software/automation-software/tia-portal.html].

[9] Yokogawa. (n.d.). OPC UA data communication standard for process control. Yokogawa. Retrieved from [https://www.yokogawa.com/de/solutions/featured-topics/digital-infrastructure-wiki/technologies/opc-ua-data-communication-standard-for-process-control/].

[10] Schluse, M., Priggemeyer, M., Atorf, L., & Rossmann, J. (2018). Experimentable Digital Twins—Streamlining Simulation-Based Systems Engineering for Industry 4.0. IEEE Transactions on Industrial Informatics, 14(4), 1722-1731. [https://doi.org/10.1109/TII.2018.2804917].

[11] Modbus Organization. (2021). Modbus Application Protocol Specification V1.1b3. Retrieved from [https://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf].

[12] OPC Foundation. (2020). OPC Unified Architecture Specification Part 1: Overview and Concepts. Retrieved from [https://opcfoundation.org].

[13] OPC UA: Data Communication Standard for Process Control [https://www.yokogawa.com/de/solutions/featured-topics/digital-infrastructure-wiki/technologies/opc-ua-data-communication-standard-for-process-control/].

[14] Craig, J. J. (2017). Introduction to Robotics: Mechanics and Control (4th ed.). Pearson. ISBN: 978-0133489798. Retrieved from [https://www.pearson.com/store/p/introduction-to-robotics-mechanics-and-control/P100000306349/9780133489798].

[15] Is Digital Twin Technology Supporting Safety Management? ABibliometric and Systematic Review [https://www.researchgate.net/publication/350206568_Is_Digital_Twin_Technology_Supporting_Safety_Management_A_Bibliometric_and_Systematic_Review#fullTextFileContent].

[16] Boschert, S., & Rosen, R. (2016). Digital twin—the simulation aspect. In *Mechatronic Futures* (pp. 59-74). Springer, Cham.

[17] Negri, E., Fumagalli, L., & Macchi, M. (2017). A review of the roles of digital twin in CPS-based production systems. *Procedia Manufacturing*, 11, 939-948.

[18] Qi, Q., & Tao, F. (2018). Digital twin and big data towards smart manufacturing and industry 4.0: 360 degree comparison. *Ieee Access*, 6, 3585-3593.

[19] Qi, Q., Tao, F., Zuo, Y., & Zhao, D. (2019). Digital twin service towards smart manufacturing. *Procedia Cirp*, 72, 237-242.

[20] Söderberg, R., Wärmefjord, K., Carlson, J. S., & Lindkvist, L. (2017). Toward a digital twin for real-time geometry assurance in individualized production. *CIRP Annals*, 66(1), 137-140.

[21] Tao, F., Cheng, J., Qi, Q., Zhang, M., Zhang, H., & Sui, F. (2018). Digital twin-driven product design, manufacturing and service with big data. *The International Journal of Advanced Manufacturing Technology*, 94(9-12), 3563-3576.

[22] Cinar, O. (2020). OPC UA for IoT/Industrie 4.0 - Data Exchange Made Easy. Packt Publishing.

[23] Mathworks. (2023). MATLAB and Simulink. Retrieved from [`https://www.mathworks.com/`].

[24] Simulink. (2023). Simulink - Simulation and Model-Based Design. Retrieved from [`https://www.mathworks.com/products/simulink.html`].

[25] Digital twins – doubling the potential for innovation [`https://www.konicaminolta.eu/eu-en/rethink-work/tools/digital-twins-doubling-the-potential-for-innovation`].

[26] Nise, N. S. (2019). Control Systems Engineering (8th ed.). Wiley. ISBN: 978-1119495985. Retrieved from [`https://www.wiley.com/en-us/Control+Systems+Engineering%2C+8th+Edition-p-9781119495985`].

[27] Cavalieri, S., & Salafia, M. H. (2019). Enhancing security in OPC UA communications. IEEE Transactions on Industrial Informatics, 15(4), 2321-2331.

[28] Cinar, O. (2020). OPC UA for IoT/Industrie 4.0 - Data Exchange Made Easy. Packt Publishing.

[29] Mahnke, W., Leitner, S. H., & Damm, M. (2009). OPC Unified Architecture. Springer-Verlag.

[30] Shah, Kevin. 2023. *Developing a Digital Twin Model of an Industry 4.0 Lab Scale Plant Using the OPC UA Protocol To Communicate With A Siemens PLC For Self-Optimization.* Master thesis, Hochschule Schmalkalden.

[31] Krushnan, Jayabadhrinath. *Design and Fabrication of a 4-DOF Robotic Arm.* Project Work Report, Hochschule Schmalkalden, Mechanical Engineering, Mechatronics and Robotics, Master of Engineering, 2023.

[32] ASRS robotic arm one joint movement hardware vs Simulink model video [`https://drive.google.com/file/d/1qMGhnqhuP4reKOXsiIvz72FzMkuMTsni/view?usp=sharing`].

[33] ASRS robotic arm model full trajectory movement in Simulink video [`https://drive.google.com/file/d/1mi326tN4ljRvmfDGOsL00pr3Ni7tfKL3/view?usp=sharing`].

A Appendix A

A.1 OPC hardware setup

Figure 57: Physical connection between Siemens PLC and PC

A.2 OPC settings setup

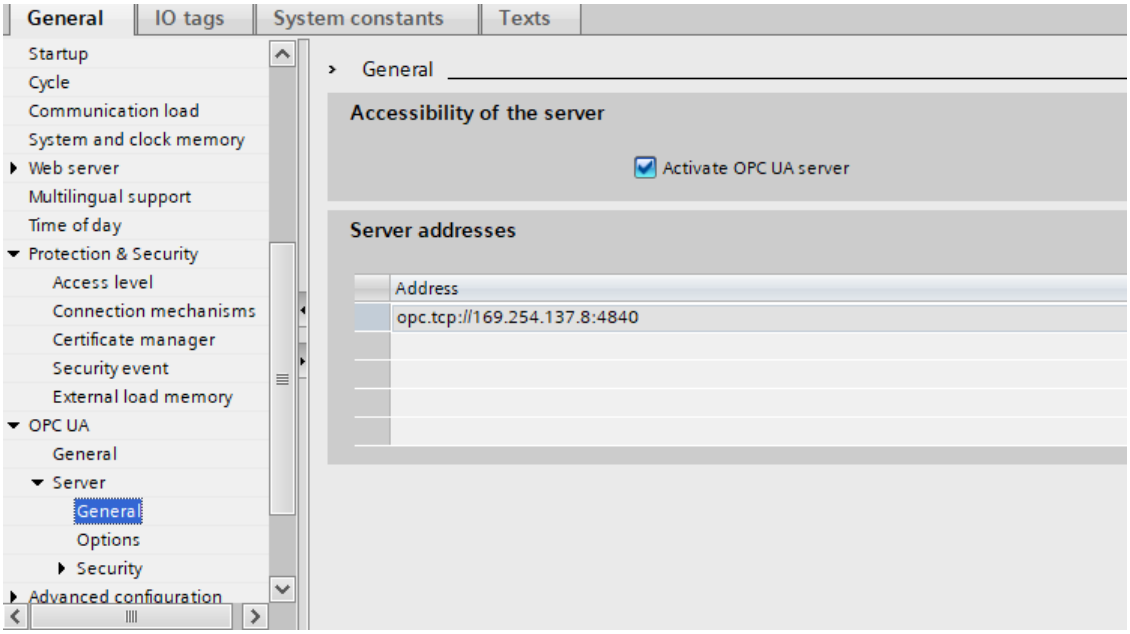


Figure 58: OPC Communication Settings

A.3 OPC interface setup

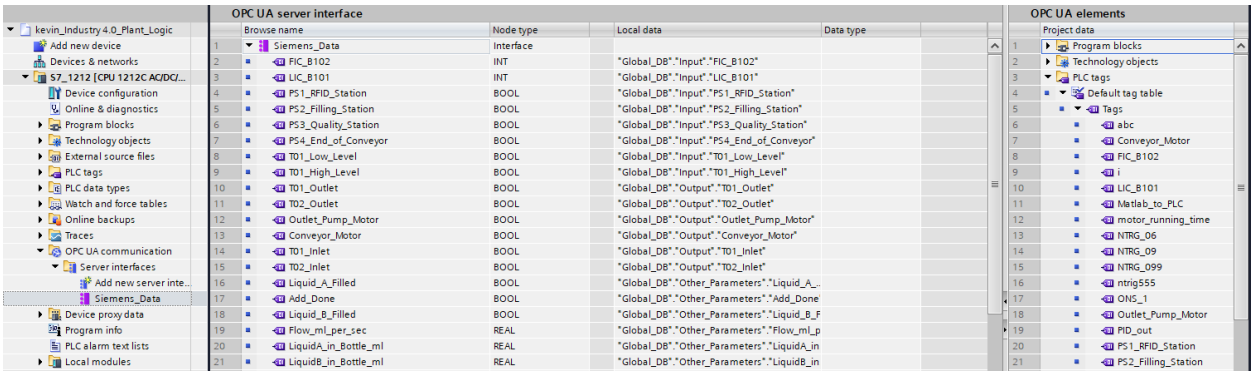


Figure 59: OPC server interface

A.4 OPC on MATLAB setup

```
uaClient =

OPC UA Client:

   Server Information:
                      Name: 'SIMATIC.S7-1200.OPC-UA.Application:S7_1212'
                  Hostname: '169.254.137.8'
                      Port: 4840
               EndpointUrl: 'opc.tcp://169.254.137.8:4840'

   Connection Information:
                   Timeout: 10
                    Status: 'Connected'
               ServerState: 'Running'

   Security Information:
        MessageSecurityMode: None
     ChannelSecurityPolicy: None
                 Endpoints: [1×1 opc.ua.EndpointDescription]

   Server Limits:
              MinSampleRate: 0.0001 sec
               MaxReadNodes: 0
              MaxWriteNodes: 0
        MaxHistoryReadNodes: 0
    MaxHistoryValuesPerNode: 0
```

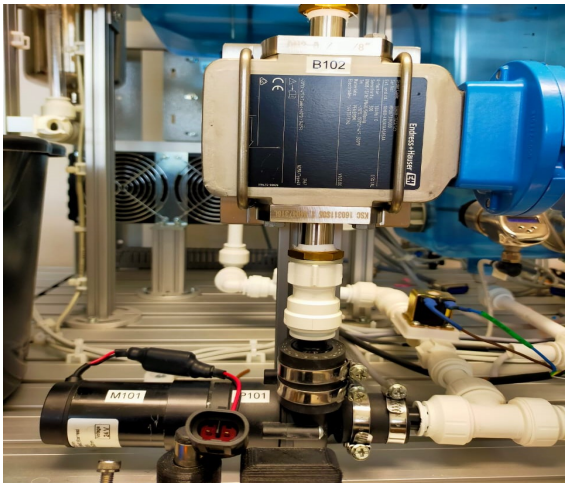
Figure 60: OPC address in Matlab

```
uaClient =

OPC UA Client:

   Server Information:
                        Name: 'SIMATIC.S7-1200.OPC-UA.Application:S7_1212'
                    Hostname: '169.254.137.8'
                        Port: 4840
                 EndpointUrl: 'opc.tcp://169.254.137.8:4840'

   Connection Information:
                     Timeout: 10
                      Status: 'Connected'
                 ServerState: 'Running'

   Security Information:
          MessageSecurityMode: None
       ChannelSecurityPolicy: None
                  Endpoints: [1×1 opc.ua.EndpointDescription]

   Server Limits:
              MinSampleRate: 0.0001 sec
               MaxReadNodes: 0
              MaxWriteNodes: 0
          MaxHistoryReadNodes: 0
    MaxHistoryValuesPerNode: 0
```

Figure 61: OPC address in Matlab

B Appendix B

B.1 Voltage supplier test

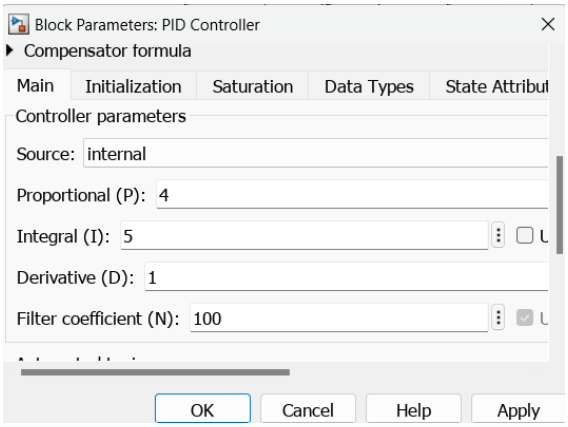


(a) Pump

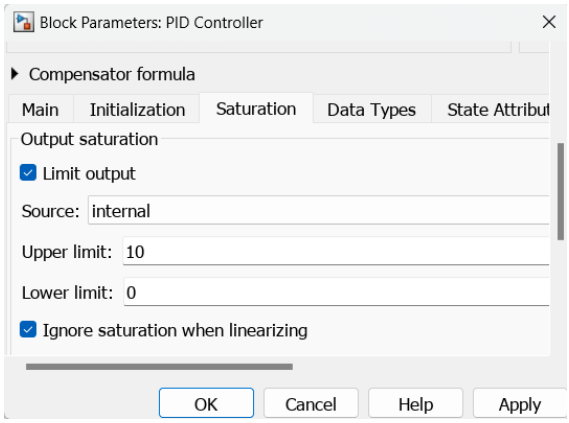
(b) Voltage supplier

Figure 62: pump with voltage supplier test

B.2 PID Configuration



(a) Kp, Ki and Kd values



(b) Voltage limits

Figure 63: PID block configuration

B.3 Voltage selector function

```
function y = fcn( Filled_amount,Voltage, amount)
if(amount <=30)
    factor = 0.79;
elseif(amount<=35)
    factor = 0.84;
elseif(amount<=40)
    factor = 0.88;
elseif(amount<=45)
    factor = 0.89;
elseif(amount<=50)
    factor = 0.9;
elseif(amount<=75)
    factor = 0.93;
elseif(amount<=100)
    factor = 0.96;
elseif(amount<=125)
    factor = 0.97;
else
    factor = 0.98;
end
if(Filled_amount>= amount*factor)
    Voltage = 0 ;
end
y = Voltage;

```

Figure 64: Voltage selector function

B.4 OPC read function

OPC_Read.m

```
function [x] = OPC_Read()
persistent init_Server ;
persistent init_Nodes1 ;
persistent uaClient ;
persistent Flow_rate;

init_Nodes1 = 0 ;
if (isempty(init_Server))
init_Nodes1 = 0;
init_Server = 0;
end

if init_Server == 0
init_Server = 1;
uaClient = opcua ( '169.254.137.8' ,4840);
connect ( uaClient );
end

if uaClient . isConnected == 1 && init_Nodes1 == 0
init_Nodes1 = 1 ;
Flow_rate = opcuanode (4 , 48 , uaClient );

% 48 is the address of the Flow rate variable in OPC data interface used
% can differ depending on the data interface used.
end

if uaClient . isConnected ==1 && init_Nodes1 == 1
    a = readValue ( uaClient , Flow_rate ) ;
end
    x=a;
end
```

Figure 65: MATLAB function reads flow rate through OPC

B.5 Reference Voltage validation

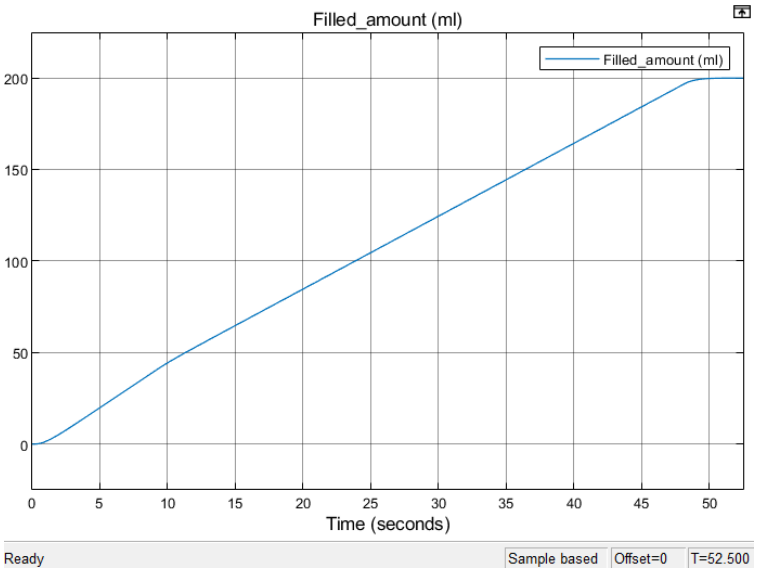


Figure 66: Filled amount with 8V reference voltage

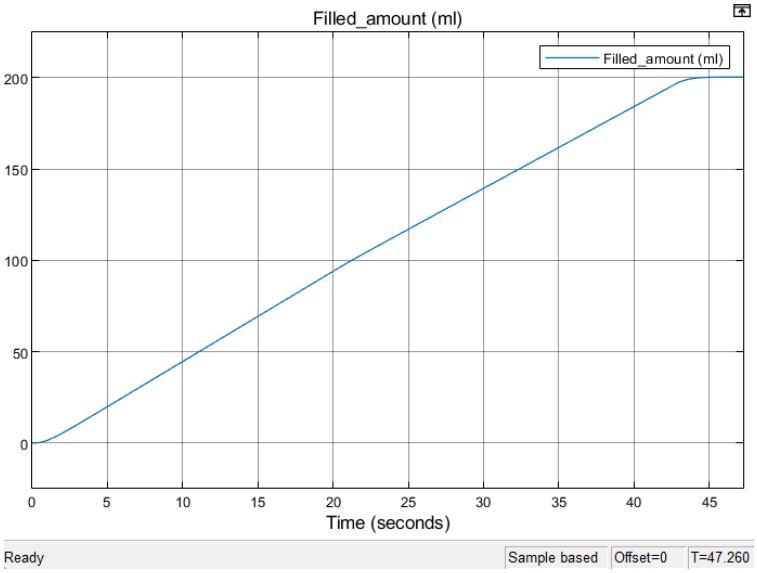


Figure 67: Filled amount with 9V reference voltage

B.6 Filling amount validation

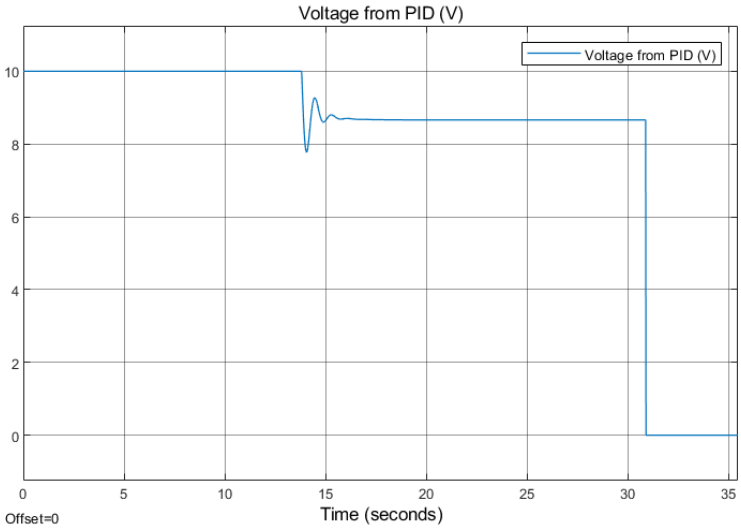


Figure 68: Voltage signal from PID, with filling amount 140 ml and 8.6V reference voltage

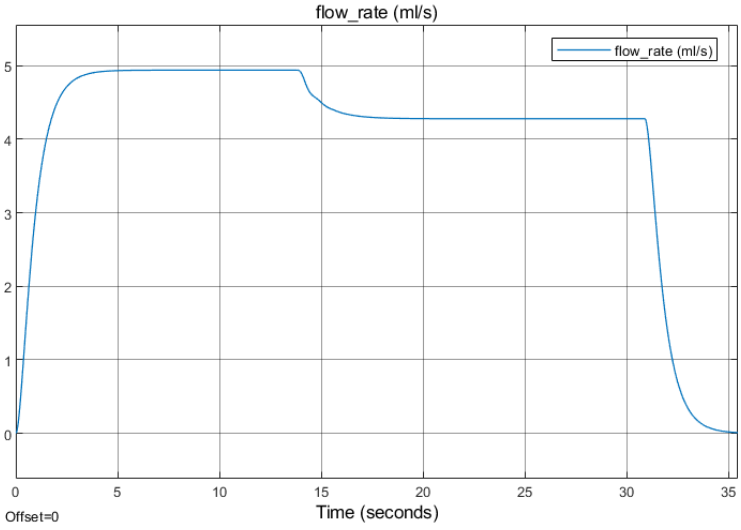


Figure 69: Flow rate with filling amount 140 ml and 8.6V reference voltage

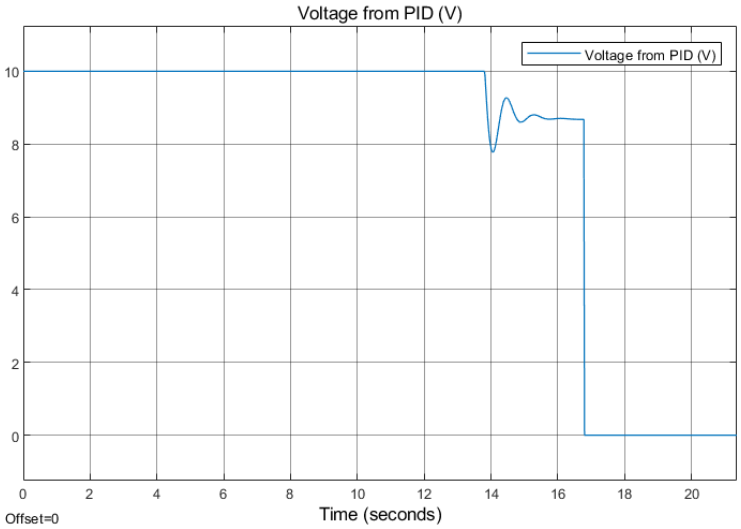


Figure 70: Voltage signal from PID, with filling amount 80 ml and 8.6V reference voltage

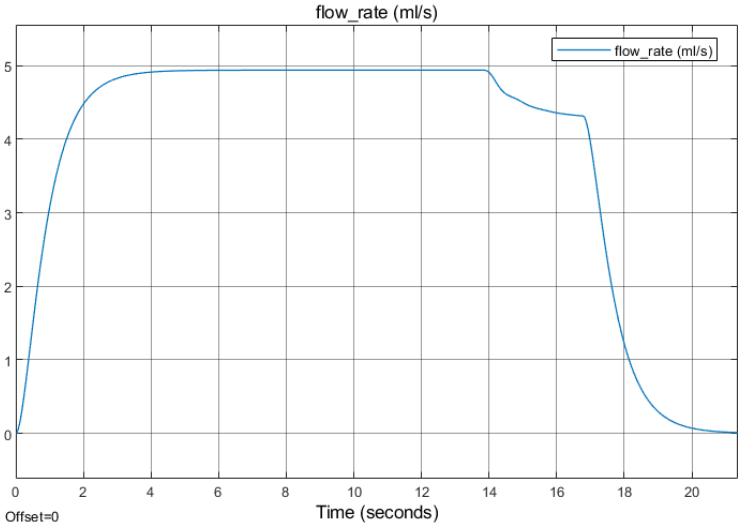


Figure 71: Flow rate with filling amount 80 ml and 8.6V reference voltage

C Appendix C

C.1 ASRS 4 DOF robotic arm model

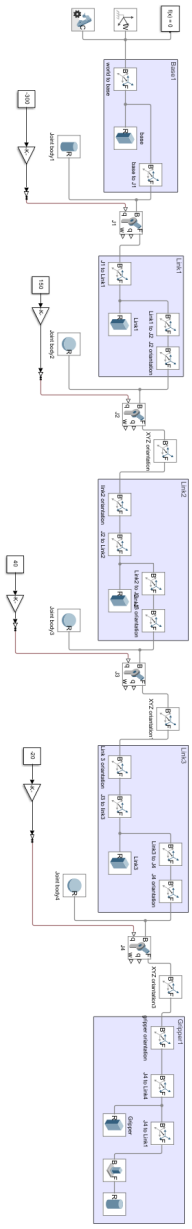


Figure 72: ASRS robotic arm model

C.2 Limitation of Angles of the Joints of ASRS 4 DOF robot

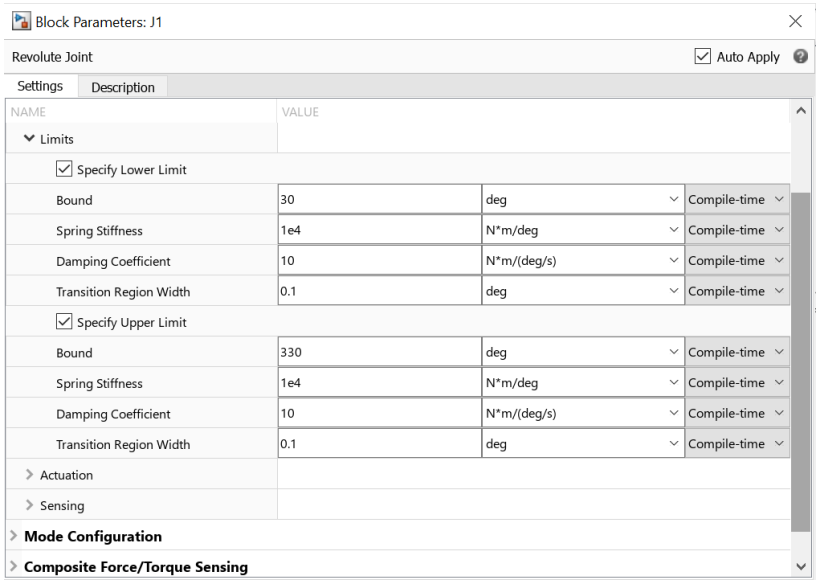


Figure 73: ASRS limits of angle of joint 1

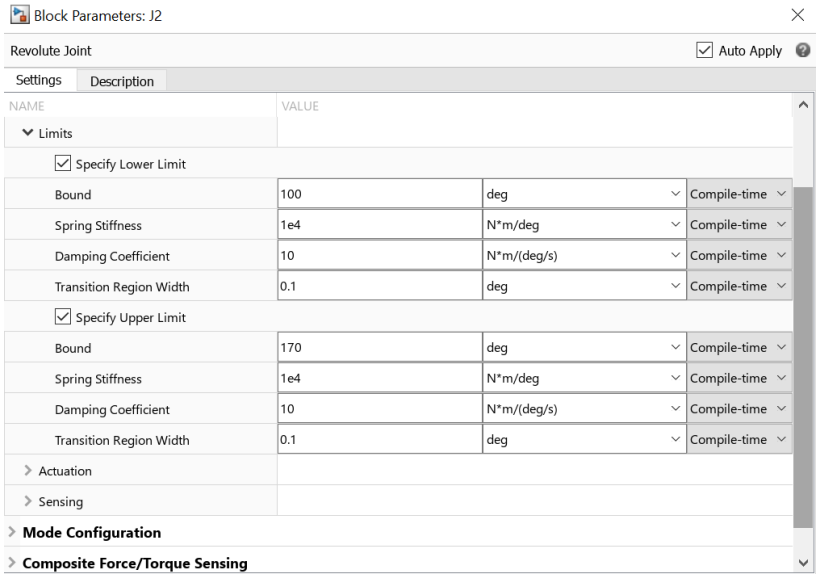


Figure 74: ASRS limits of angle of joint 2

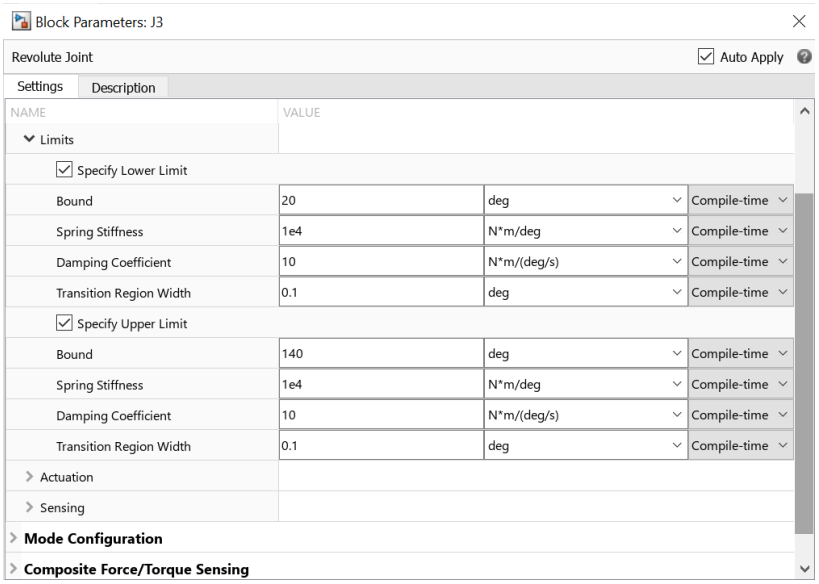


Figure 75: ASRS limits of angle of joint 3

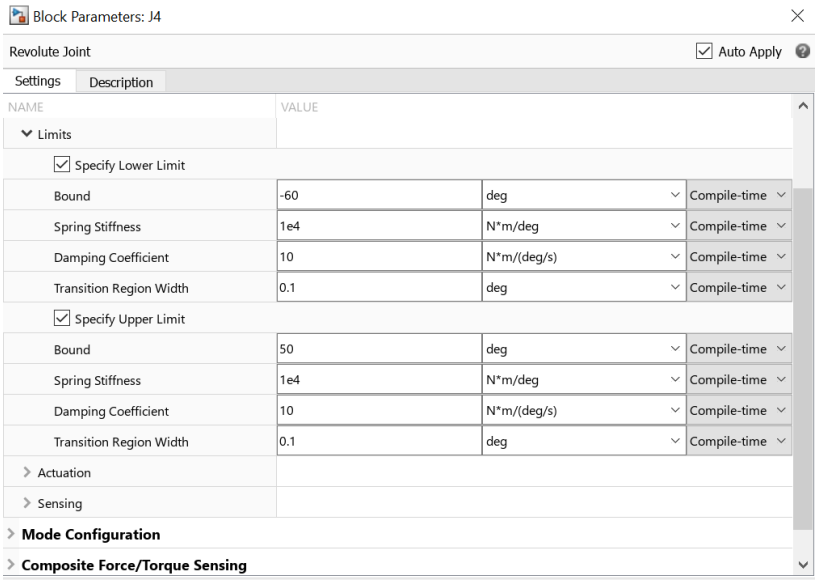


Figure 76: ASRS limits of angle of joint 4

C.3 Angles of the Joints of ASRS 4 DOF robot

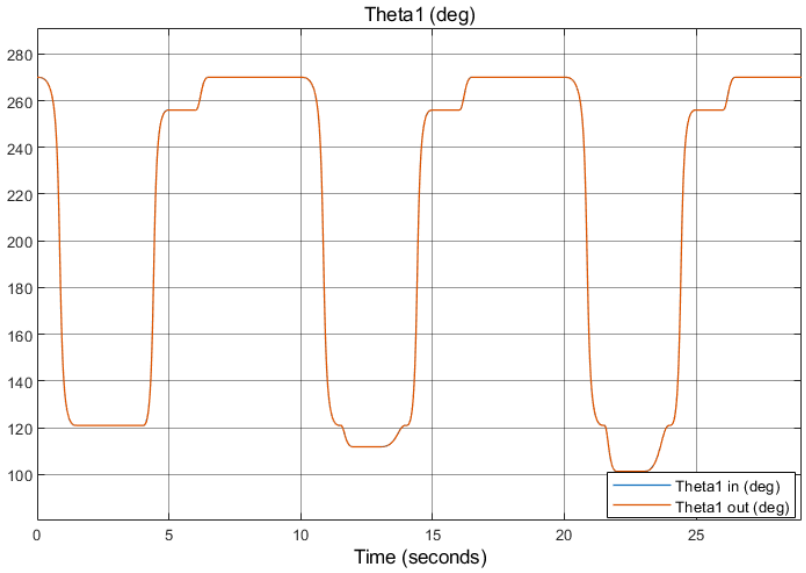


Figure 77: ASRS Angle of joint 1

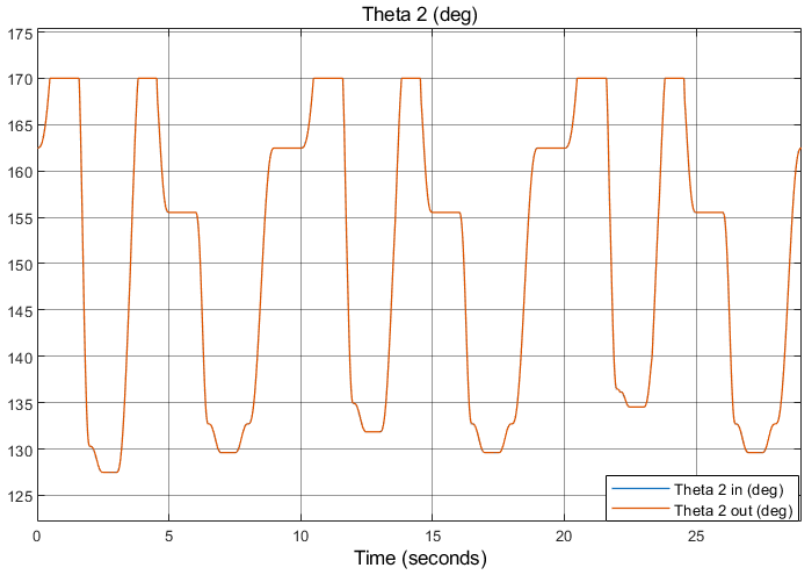


Figure 78: ASRS Angle of joint 2

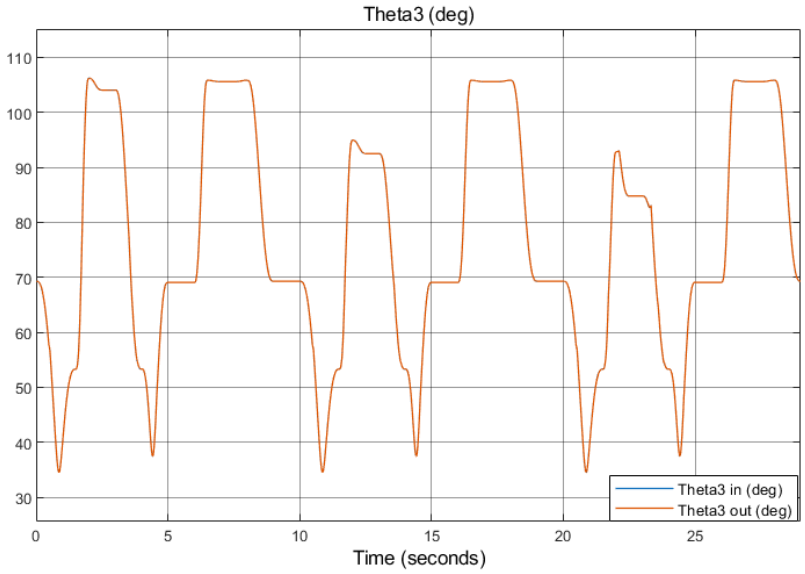


Figure 79: ASRS Angle of joint 3

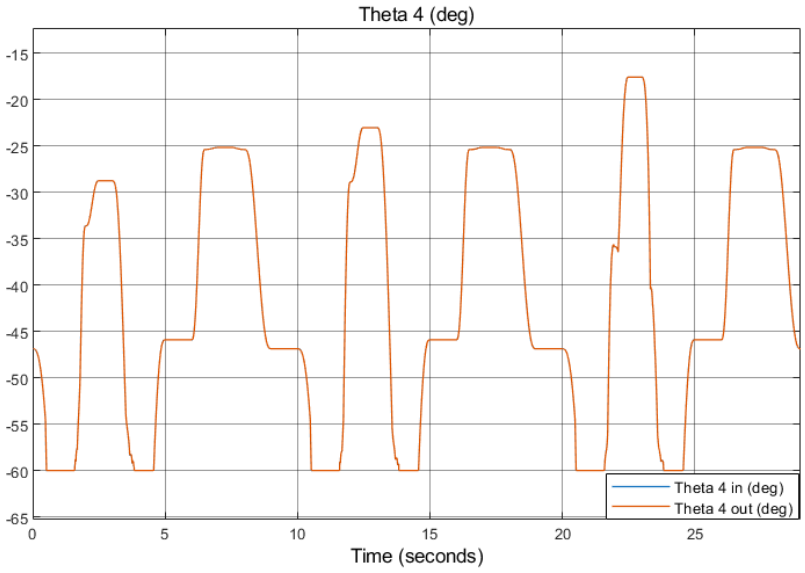


Figure 80: ASRS Angle of joint 4

C.4 Angular velocities of the joints of ASRS 4 DOF robot

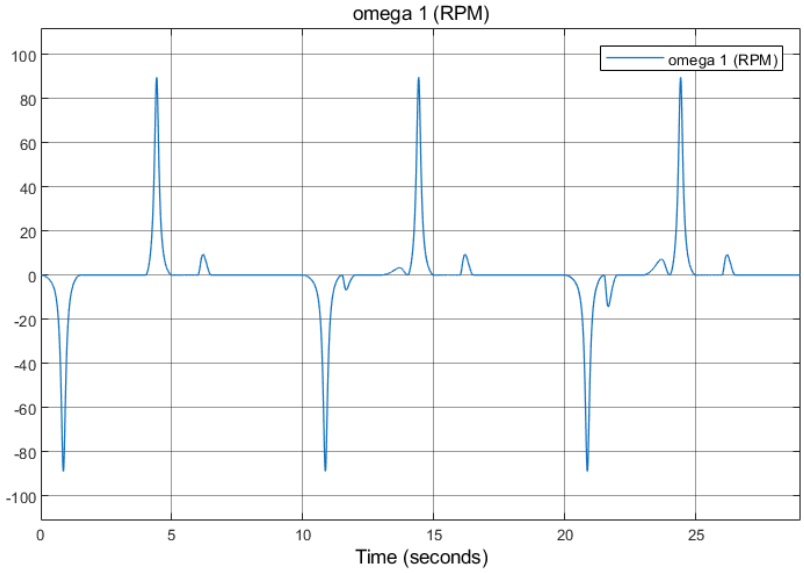


Figure 81: ASRS Angular velocity of joint 1

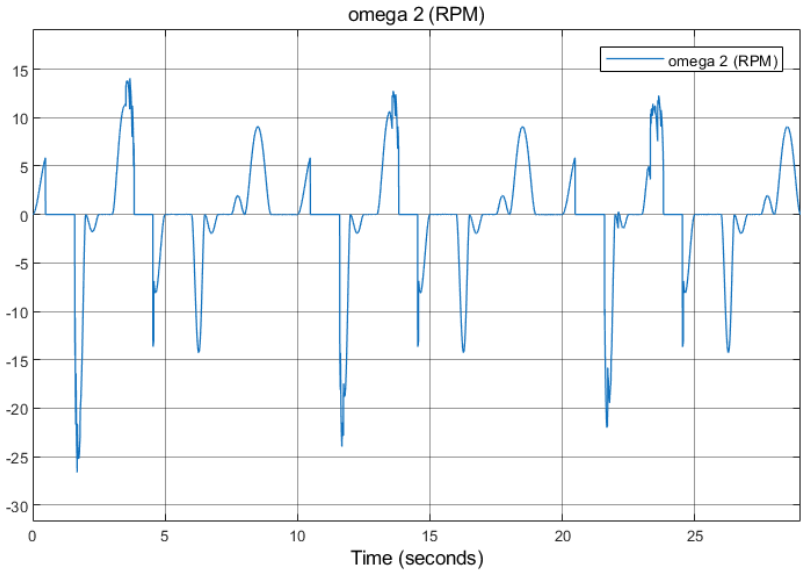


Figure 82: ASRS Angular velocity of joint 2

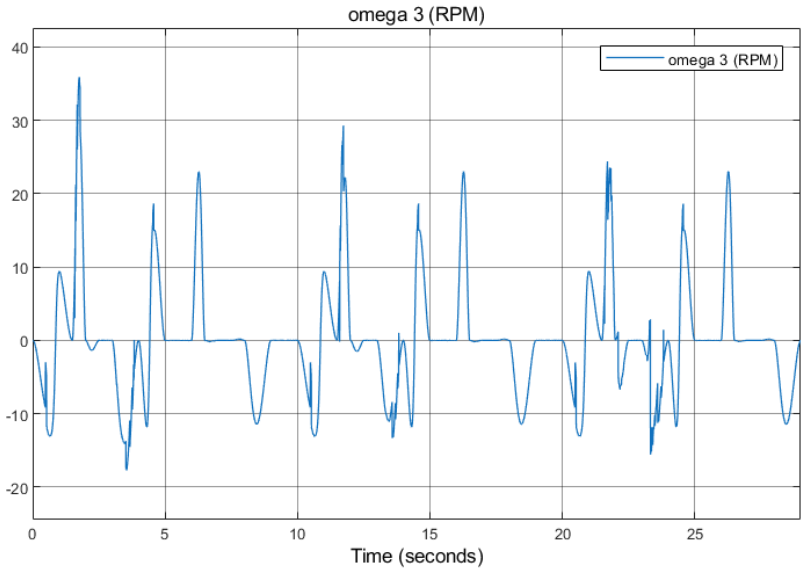


Figure 83: ASRS Angular velocity of joint 3

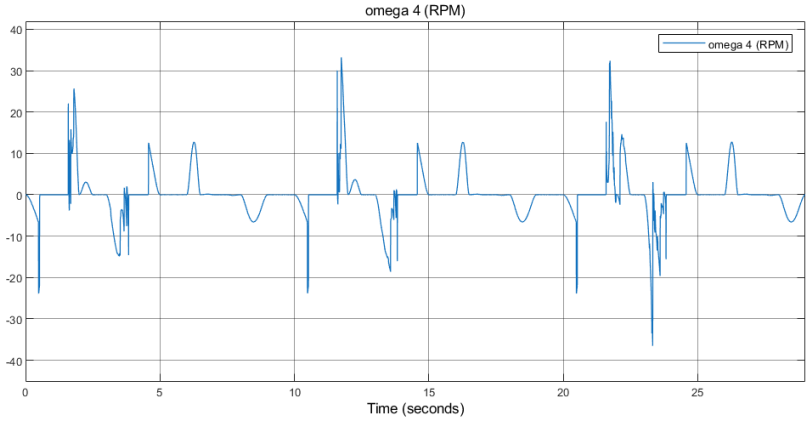


Figure 84: ASRS Angular velocity of joint 4