EAI Endorsed Transactions
on AI and Robotics

Research Article EAI.EU

Simulation and Control of the KUKA KR6 900EX Robot in Unity 3D: Advancing Industrial Automation through Virtual Environments

Anand Ajayakumar Sujatha [1], Amin Kolahdooz [1,*], Mohammadreza Jafari [1] and Alireza Hajfathalian[2]

[1] Faculty of Technology, School of Engineering and Sustainable Development, De Montfort University, Leicester, UK
[2] Mechanical Engineering - Manufacturing and Production, Noshirvani University of Technology, Babol, Iran

Abstract

This study presents the development of a virtual simulation of a KUKA robot within the Unity 3D platform, focusing on its ability to execute pick-and-place operations in an industrial setting. The research emphasizes the importance of digital simulations as cost-effective and safe alternatives to physical prototypes in industrial automation. By replicating robotic tasks in a virtual environment, organizations can mitigate wear and tear on expensive machinery and minimize safety hazards inherent in real-world operations. The simulation process commenced with the creation of a detailed 3D model of the KUKA robot utilizing Creo CAD software. This model was subsequently imported into the Unity 3D environment, where an interactive and realistic simulation environment was constructed. A manual control system was implemented through custom C# scripts, enabling precise joint manipulation via keyboard inputs. While the current control mechanism remains manual, this study provides a foundational framework for the future integration of advanced algorithms for trajectory planning and autonomous control. The simulation successfully demonstrates the feasibility of performing industrial robotic tasks within a virtual environment. It serves as a platform for further research, including the automation of robotic movements and the integration of virtual reality and digital twin technologies. These advancements have the potential to significantly enhance real-time monitoring, operator training, and overall operational efficiency in industrial applications. This work underscores the growing significance of virtual simulation technologies in industrial automation, presenting a scalable and flexible solution for prototyping, testing, and training within complex industrial ecosystems.

Keywords:

Unity 3D, Robotics Simulation, Creo, Automated Systems, Pick-and-Place Operations, Industrial Automation

Received on 02 December 2024, accepted on 25 February 2025, published on 20 March 2025

doi: 10.4108/airo.8026

1. Introduction

The utilization of robotic systems within industrial settings has witnessed substantial growth in recent years, primarily driven by the imperative for enhanced efficiency, precision, and operational flexibility. Robots such as the KUKA KR6 900EX have become indispensable in automating tasks encompassing assembly, welding, and material handling. However, the conventional methodology for developing, testing, and optimizing these robotic systems within physical environments presents significant drawbacks, characterized by both time-consuming and costly procedures. As a viable solution, virtual simulations have emerged as invaluable assets within robotics research and development, empowering engineers to evaluate control strategies, optimize movement trajectories, and simulate diverse operational scenarios without the necessity of physical prototypes [1, 2]. Unity 3D, initially conceived as a game engine, has evolved into a potent tool for simulating real-world physics

*Corresponding author. Email: Amin.kolahdooz@dmu.ac.uk

and interactions [3]. Equipped with the NVIDIA PhysX engine and real-time rendering capabilities, Unity has garnered significant traction within the robotics domain due to its capacity to create interactive and immersive simulation environments. Through the implementation of detailed 3D modelling, robust physics simulation, and flexible scripting functionalities, Unity provides an adaptable platform for simulating intricate robotic systems such as the KUKA KR6 900EX. This review delves into the utilization of Unity for robotic simulation, examining key methodologies, control techniques, and integration approaches while concurrently highlighting the challenges and future avenues for further advancement within this domain.

1.1. Advances in Robotics and Simulation Technologies

Robots have emerged as pivotal actors within the contemporary industrial era, exhibiting the capacity to execute a diverse array of tasks with exceptional precision and efficiency. Their indispensability is evident across a spectrum of industries, including service [4], agriculture [5], pharmaceuticals, electronics, aerospace, and automotive manufacturing, as well as in image processing applications within unmanned aerial vehicles (UAVs) [6, 7]. Within the manufacturing sector, robots are predominantly employed to automate repetitive tasks, such as assembly, welding, and painting. Industrial robots serve as the driving force behind production, logistics, and construction processes in the modern world. These programmable and automated machines are meticulously designed to execute tasks demanding high levels of accuracy, repeatability, and efficiency, encompassing operations such as welding, assembly, painting, and quality inspection [8] . The automotive sector exemplifies the transformative impact of robots, where they spearhead automation initiatives, leading to significant enhancements in both production rates and product quality [9]. In the healthcare domain, robots play a crucial role in assisting with surgical procedures, rehabilitation, and patient care [10]. Furthermore, within the logistics sector, robots are employed in warehouse environments for the efficient sorting, packing, and transportation of goods [11]. To ensure the effective implementation and operation of robots, simulation methodologies play a critical role. Simulation enables the creation of virtual models of robotic systems and their respective environments, facilitating comprehensive testing and optimization without the necessity of developing and testing physical prototypes [2].The development of physical prototypes for testing can be both financially and temporally demanding. By leveraging simulation, developers can iteratively refine designs and control algorithms, resulting in significant reductions in both costs and development timelines [12].
Furthermore, simulation facilitates advancements in trajectory planning, control system development, and rigorous testing within virtual environments. This approach offers unparalleled flexibility, enabling developers to optimize both system performance and stability through iterative design processes. For instance, simulation has proven to be an invaluable tool in optimizing air suspension systems for vehicles, where key parameters such as comfort and stability are effectively enhanced through virtual testing [13]. Analogously, within the realm of robotics, simulation empowers developers to refine their systems efficiently, thereby minimizing risks and costs while concurrently improving overall functionality [14]. In the specific context of robotic system development, simulation emerges as a particularly effective tool for optimizing both designs and control systems. By incorporating robust design principles into the simulation process, developers can ensure smoother system operation and reduce computational costs, drawing parallels to the concept of topology optimization commonly employed in structural engineering. This robust approach to simulation not only enhances system performance but also aids in the proactive identification of potential issues prior to implementation, consequently increasing the reliability of robotic systems in real-world applications [15, 16].
Simulation facilitates the comprehensive testing of diverse configurations and materials, enabling the optimization of the design prior to the commencement of any manufacturing processes. Furthermore, simulation provides a secure platform for evaluating robot functionalities under a wide range of conditions, including hazardous environments [17]. This capability is of paramount importance for applications within industries such as aerospace [18]. The ability to rapidly test and iterate upon robotic designs and algorithms within a virtual environment significantly enhances the overall development process. Simulation tools are frequently equipped with visualization and feedback capabilities, which facilitate highly accurate testing of the robot's performance.

1.2. Trajectory Planning in Robotic Systems

Trajectory planning constitutes a fundamental aspect of robotic systems, encompassing the determination of the robot's path from its initial position to the desired final location. To ensure smooth and efficient operation, this process necessitates the consideration of various constraints, including kinematic, dynamic, and obstacle avoidance factors. Effective trajectory planning methodologies are crucial for enabling the robot to perform its tasks safely, smoothly, and within a timely manner, thereby maximizing operational efficiency and productivity [19]. Traditional trajectory planning methods often involve complex algorithms and necessitate substantial computational resources, thus underscoring the need for more efficient and user-friendly solutions [20].
One of the primary motivations for meticulous trajectory planning arises from the fact that robots frequently operate within dynamic environments where interactions with humans, other robots, and machinery are inevitable. A

well-planned trajectory ensures that the robot effectively avoids collisions and operates within its designated workspace, thereby minimizing the risk of accidents and damage . Safety is of paramount importance in industrial settings where robots handle heavy machinery or equipment or interact with hazardous materials, as any deviations from the planned path could have severe consequences [21]. Furthermore, through the integration of real-time image processing and advanced deep learning algorithms, such as those employed in UAV object detection, the robot's ability to adapt to dynamic obstacles and adjust its trajectory in real-time is significantly enhanced, ensuring greater precision and safety [22]. In numerous robotics applications, including assembly, welding, and surgical procedures, precision and accuracy are of paramount importance. In such instances, trajectory planning empowers the robot to execute movements with high precision, ensuring that tasks are performed accurately and consistently. In the context of robotic surgery, precise trajectory planning is indispensable to guarantee that surgical instruments reach their intended targets without causing damage to surrounding tissues. Within the manufacturing domain, precise trajectory planning ensures proper component assembly, thereby maintaining product quality and minimizing material wastage [23].

Trajectory planning plays a crucial role in optimizing the robot's path to minimize the time and energy required to complete tasks. This optimization leads to faster cycle times and reduced operational costs, factors that are of critical importance for industries such as electronics manufacturing and automotive production. Effective trajectory planning also contributes to extending the lifespan of robotic components by minimizing unnecessary movements and reducing wear and tear [24].

1.3. Virtual Reality for Robotic Simulations

Virtual reality (VR) technology has witnessed a surge in popularity within the modern robotics domain, providing immersive experiences that empower users to interact with and manipulate virtual environments. In the context of robotics, VR significantly enhances training, simulation, and visualization capabilities, enabling users to experiment with complex scenarios within a safe and controlled virtual space. This capability proves particularly valuable in the domains of trajectory planning and robot programming, where real-world testing can be both costly and inherently risky. VR empowers developers to create detailed and dynamic simulations where robots can be developed and rigorously tested within a diverse range of environments without the necessity of constructing physical prototypes. Recent studies have underscored the transformative role of VR in enhancing training and simulation processes, particularly within fields such as engineering education, robotics, and healthcare [25]. These studies provide compelling evidence of how VR facilitates safe experimentation and accelerates skill acquisition in environments where traditional methodologies may be prohibitively expensive or impractical [26, 27]. Traditional training methods can be both costly and time-consuming. However, through the utilization of VR technologies, trainees can effectively learn to operate complex robotic systems, such as surgical robots or industrial robots, through immersive simulations [28, 29]. This approach not only improves the learning curve but also enhances the safety of both the machine and the operator.

The integration of VR into the field of robotics represents a significant advancement, offering novel possibilities for development, training, remote operation, and human-robot interaction. Through the leveraging of this technology, robots can be rendered more intelligent, efficient, and safer for both the environment and human operators [30].

1.4. The Role of the Unity Engine in Robotic Simulations

The Unity game engine [31] serves as a versatile platform for the creation of interactive 2D and 3D experiences and constitutes a valuable tool for simulations across diverse fields, including education, robotics, automotive, and architecture. Its robust feature set renders it particularly well-suited for dynamic, immersive, and interactive simulations. The incorporation of a powerful physics engine, which accurately simulates real-world physical interactions, enables precise modelling of object interactions [32]. This capability is instrumental in robotic simulations, providing insights into how the robot interacts with its surrounding environment. Unity's capacity for creating interactive and immersive simulations is further enhanced by its compatibility with VR hardware, making it an invaluable asset for robotics engineers [33]. Unity's extensive library and user-friendly interface make it accessible to both novice and experienced users. The platform supports multiple programming languages, including C# and JavaScript, affording developers the flexibility to choose the language with which they are most comfortable. Unity's robust scripting capabilities enable the integration of complex algorithms and control systems, a crucial requirement for advanced simulations [34]. Within the realm of robotics, Unity is employed to construct digital twins, or virtual replicas, of robots, which accurately mimic their interactions and behavior in the real world. Unity's ability to simulate a wide range of scenarios and environments allows developers to evaluate robot performance under real-world conditions, thereby reducing the time and cost associated with physical testing. Furthermore, Unity's capacity to simulate sensor data and integrate with robotic control systems makes it a powerful tool for the development of autonomous robots and advanced robotics applications. Beyond development and testing, Unity finds widespread application in training and educational settings [35]. Its immersive and interactive nature makes it an exceptional tool for conveying complex concepts and skills. In robotics education, students can effectively visualize and interact with robotic systems

within the Unity environment, thereby enhancing their understanding of theoretical concepts through hands-on experiences [36]. Unity's advanced features and inherent flexibility establish it as a highly suitable platform for simulations across various domains. Its ability to create realistic environments, coupled with a robust physics engine and sophisticated scripting capabilities, enables the development of detailed and accurate simulations for advanced robotics applications [37].

2. Methodology

2.1. Design and Development of Robotic Arm Using Creo Parametric

The KUKA KR6 900EX robotic arm was meticulously modeled within Creo Parametric 10.0, a widely recognized CAD software renowned for its precision in mechanical design. The modeling process commenced with the base, which provides structural stability to the entire assembly, followed by the sequential design and integration of the shoulder, upper arm, elbow, forearm, and wrist components. Each joint was meticulously designed based on the technical specifications outlined in KUKA's official documentation, ensuring a high degree of accuracy and fidelity to the physical robot.

Particular attention was devoted to the design of the end effector, configured as a pick-and-place gripper, which was optimized to securely handle objects without causing any damage. Upon the completion of the individual components, they were assembled into a unified Creo file, with realistic movement constraints applied to accurately simulate operational behavior. Figure 1 presents the engineering drawings of the robotic arm components, providing a detailed view of the structural design and layout. These drawings elucidate the dimensions and interrelationships between the various parts, serving as the foundational framework for subsequent simulation and analysis.

2.2. Methodology for CAD Model Integration with Unity 3D

Exporting the 3D model from Creo to Unity 3D involves a multi-step process. This section outlines the comprehensive procedure for exporting the 3D model from Creo and subsequently importing it into Unity 3D. The completed assembly model is exported in a format compatible with Unity 3D, specifically the .obj file format. Unity 3D natively supports the .obj format, facilitating the seamless integration of all associated robot components within the Unity environment. Prior to exporting, the model undergoes a thorough inspection to ensure the absence of any errors, such as non-manifold edges, gaps, or overlaps. Finally, a visual inspection of the model is performed within Creo to confirm that all components appear as expected and that all features have been correctly implemented.

3. Simulation Environment Configuration in Unity 3D

3.1. Model Import Process in Unity 3D

Following the export of the 3D model of the KUKA robot from Creo, the next crucial step involved its importation into Unity 3D, a powerful game engine renowned for its robust physics simulation capabilities and real-time rendering performance[38]. Unity 3D was selected as the preferred platform due to its inherent capacity to handle complex simulations and its user-friendly interface for creating interactive environments. Upon importation, the robot's orientation and scale were meticulously calibrated to ensure proper alignment within the simulation space. Furthermore, the robot's joints and components were accurately mapped to Unity's physics engine, enabling realistic movement and interactions.

Subsequent to the successful importation of the robot model (Figure 2A), additional measures were implemented to enhance the simulation's realism. Color was applied to the robot model (Figure 2B), significantly improving its visual clarity and facilitating easier differentiation between various components during the simulation process. To further enhance the realism and functionality of the setup, a linear rail system (sourced from the Unity Asset Store [39]) was integrated with the robot (Figure 2C), expanding its operational range and enabling the simulation of tasks requiring greater mobility.

Finally, the robot was strategically positioned within a realistic industrial environment (Figure 2D). This environment, sourced from the Unity Asset Store[40], was meticulously designed to replicate conditions commonly encountered in industrial automation scenarios. Key elements, such as a conveyor belt system [39] for transporting objects and a designated pallet area for item placement, were included to accurately mimic real-world workflows. Additionally, a linear rail system [41] was customized and integrated into the simulation to enhance operational flexibility. The industrial setting was further enriched with basic features such as walls, floors, and appropriate lighting to enhance realism and accurately reflect operational conditions. These enhancements collectively provided an immersive and functional testing ground for evaluating the robot's performance within practical applications.

3.2. Implementation of Robotic Control Systems

To simulate real-time control of the KUKA robot, a manual control system was implemented utilizing a C# script within the Unity environment. This control system

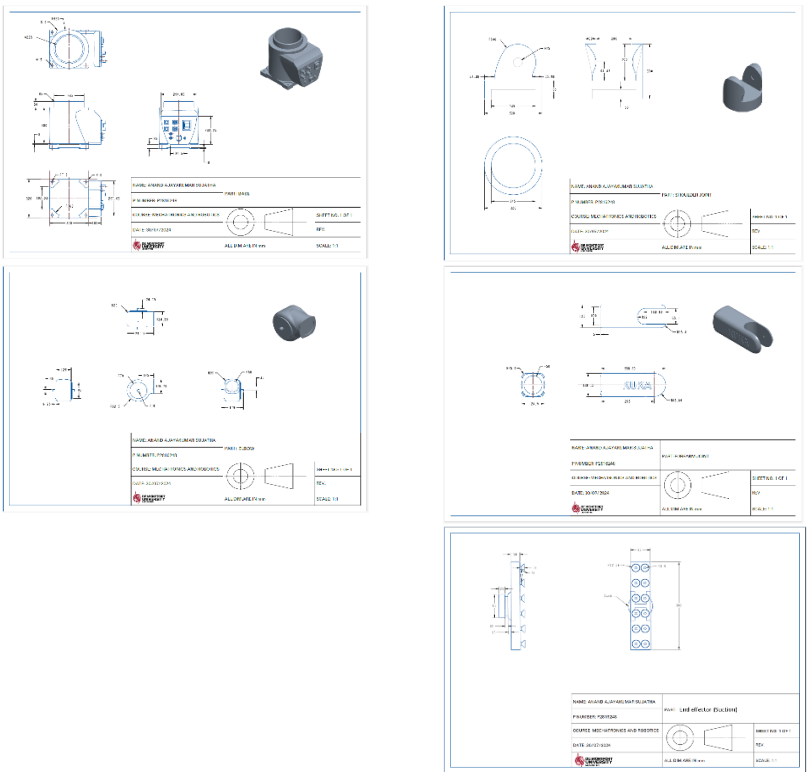

Figure 1: Detailed engineering drawings of the KUKA KR6 900EX robotic arm components

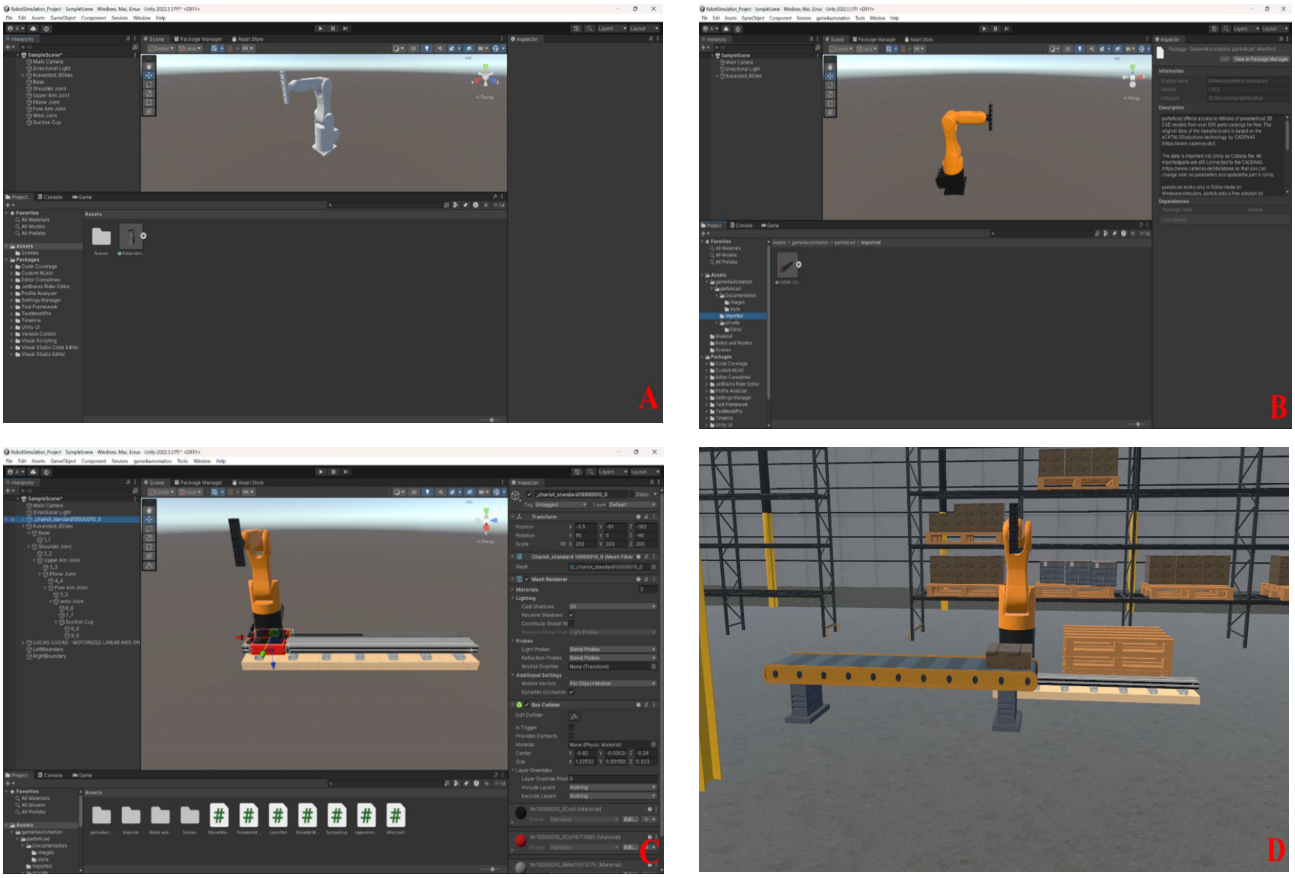

Figure 2: Sequential enhancements in the simulation environment—(A) Imported KUKA robot model, (B) Color application, (C) Integration with linear rail, (D) Placement in a realistic industrial setting

facilitated the manipulation of the robot through keyboard inputs, effectively simulating manual operation as it would occur in an industrial setting. Each joint of the robot, encompassing the shoulder, upper arm, elbow, forearm, wrist, and suction end-effector, along with pick-and-place operations, were programmed to respond to specific key inputs. This approach enabled precise manipulation of the robot's joints in real time. Table 1 presents the specified keys and their corresponding functions. These keyboard inputs empowered the operator to control the robot's trajectory with a high degree of accuracy, simulating the manual control required for pick-and-place operations within real industrial settings. Each joint was rotated around its respective axis by tracking the operator's input in real time.

Table 1. Control Keys and their functions

Joint	Keys	Functions
Shoulder joint	Q (rotate left), A (rotate right)	Shoulder joint rotation
Upper Arm	W (move up), S (move down)	Upper Arm movement
Elbow joint	E (bend), D (straighten)	Elbow joint movement
Forearm joint	R (rotate clockwise), F (rotate anticlockwise)	Forearm movement
Wrist joint	T (rotate left), G (rotate right)	Wrist joint movement
End effector	Y (rotate clockwise), H (rotate anticlockwise)	End-effector rotation
Suction control	P (pick up), Spacebar (release)	Suction controls the Pick and Place operation

3.3. C# Script Development for Robotic Control

To simulate the real-time control of the KUKA robotic arm, a manual control system was developed using Unity's C# scripting. Each joint—from shoulder to wrist—was programmed to respond to specific keyboard inputs, enabling precise control over the robot's trajectory and pick-and-place tasks. Figure 3 illustrates the flowchart of the shoulder joint control logic, visually representing the C# script. It shows how key presses ('Q' and 'A') trigger forward or backward rotation, with limits enforced to mimic the robot's physical constraints. This flowchart translates the scripting logic into a clear, visual format, enhancing the understanding of joint control while ensuring flexibility and accuracy in movement.

3.4. Robotic Kinematics: A Comprehensive Analysis

A comprehensive understanding of robotic arm kinematics is paramount for effectively controlling its motion and ensuring precise end-effector positioning, a critical requirement for the successful execution of complex tasks such as assembly, welding, and material handling. In the context of an industrial robot like the KUKA KR 10 R1100 Sixx, kinematic equations serve to define the intricate relationship between joint movements and the corresponding position and orientation of the end-effector (e.g., a gripper or tool). These equations are conventionally categorized into two distinct branches: forward kinematics, which involves the calculation of the end-effector's position and orientation based on a set of given joint angles, and inverse kinematics, which focuses on determining the necessary joint angles to achieve a desired end-effector position and orientation.

3.5. Forward Kinematics: Calculations and Applications

In the realm of forward kinematics, the objective is to compute the spatial configuration of the end-effector given a known set of joint parameters. For the KUKA KR 10 R1100 Sixx, which possesses six degrees of freedom (DOF), the Denavit-Hartenberg (D-H) convention provides a suitable framework for modelling the forward kinematics. Each transformation between adjacent joints can be characterized using the D-H parameters.
The transformation matrix T_i^{i-1} for each link i is given by Equation 1:

$$T_i^{i-1} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

where:

- θ_i: Joint angle (for revolute joints, variable; for prismatic joints, constant).
- d_i: Offset along the previous z-axis.
- a_i: Link length along the x-axis.
- α_i: Twist angle between the z-axes.

The position of the end effector is calculated by multiplying the individual transformation matrices of each joint, as shown in Equation 2:

$$T_0^6 = T_0^1 . T_1^2 . T_2^3 . T_3^4 . T_4^5 . T_5^6 \quad (2)$$

This matrix T_0^6 represents the homogeneous transformation matrix that defines the position and orientation of the end effector relative to the base frame.

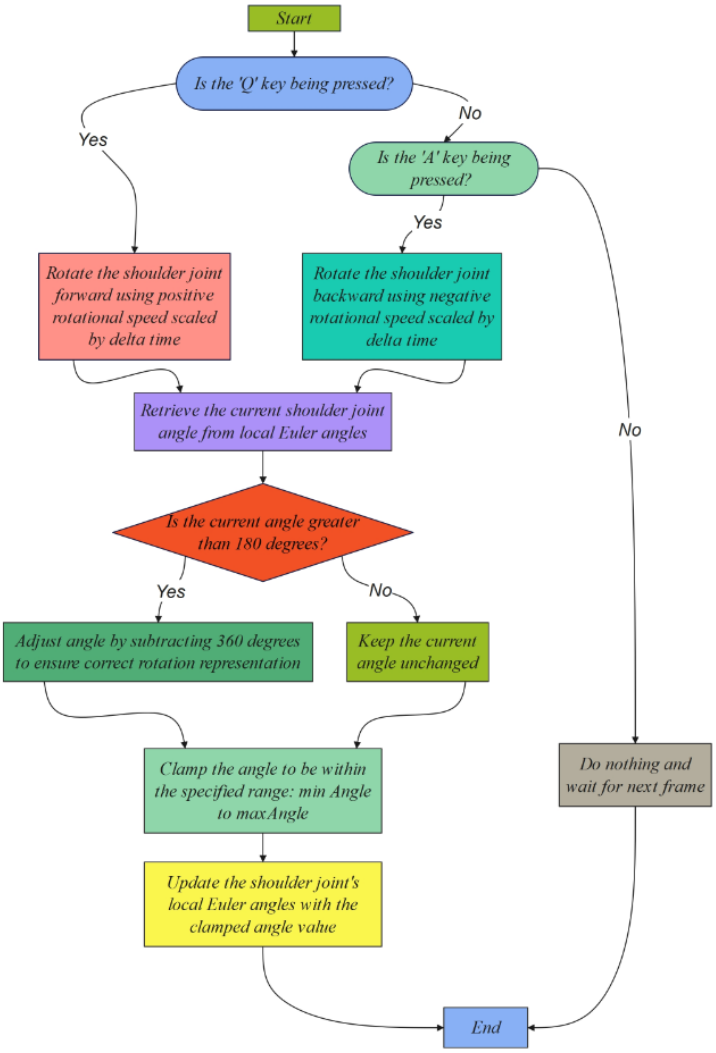

Figure 3: Movement Coding Flowchart

3.6. Inverse Kinematics: Algorithms and Optimization

Inverse kinematics is used to determine the specific joint angles required to achieve a given end-effector position and orientation. Unlike forward kinematics, which has a single solution, inverse kinematics for a 6-DOF robotic arm like the KR 10 R1100 Sixx typically has multiple possible solutions. These equations are generally more complex, involving trigonometric functions and requiring either analytical or numerical methods to solve.

The objective in inverse kinematics is to find the joint angles $\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6$ that satisfy the relationship shown in Equation 3:

$$f(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6) = (x, y, \text{orientation}) \quad (3)$$

Where (x, y, z) represents the desired position and orientation specifies the end effector's required rotation or angular configuration.

3.7. Suction System Design and Efficiency Calculations

The end effector utilizes a vacuum system with 12 suction cups to handle objects. For this design the suction force was calculated to ensure the system could lift a box weighing 10kg. The lifting force F required for the 10 kg box due to gravity was calculated using Equation 4:

$$F = m \times g \quad (4)$$

Where:

m =10Kg

g = 9.81 m/s^2

$$F = 10 \times 9.81 = 98.1N$$

The effective surface area of each suction cup is calculated using Equation 5:

$$A_{cup} = \pi \times r^2 \quad (5)$$

Where:

$$\text{r} = 12.59\text{mm} = 0.01259\text{m}$$
$$A_{cup} = \pi \times (0.01259m)^2$$
$$A_{cup} = 12 \times 4.980 \times 10^{-4}\, m^2$$
$$A_{cup} = 5.976 \times 10^{-3}\, m^2$$
$$A_{cup} = 0.00598\, m^2$$

The force provided by a suction cup is given by Equation 6:

$$F = P \times A_{cup} \quad (6)$$

Where:

- F is the required force
- P is the vacuum pressure
- A is the total surface area

The Vacuum pressure is then calculated as shown in Equation 7:

$$\text{P} = F/A_{cup} \quad (7)$$
$$P = 98.1/0.00589$$
$$P = 16416.72\, Pa$$
$$P = 16.4\, kPa$$

Based on the above calculations, a vacuum pressure of 16.4 kPa is required for each suction cup to effectively lift a 10 kg cardboard box. This ensures the system's capability to handle the designed load efficiently.

4. Results

4.1. Simulation Performance

The Unity-based simulation effectively demonstrated the feasibility of using game engine technology for industrial robotic applications. The KUKA robot performed accurate pick-and-place tasks within the virtual environment, responding reliably to manual control inputs. Movements were smooth and precise, with each joint adhering to predefined rotational limits. Additionally, the conveyor system operated effectively, ensuring that objects transitioned smoothly to the robot's pickup point, allowing for continuous task cycles.

4.2. Performance Analysis

Quantitative analysis of the simulation's performance showed that the robot's joint movements were precise and the pick-and-place operation was successfully replicated within the Unity environment. However, the reliance on manual inputs highlighted the need for further automation to enhance operational efficiency. Additionally, Unity's real-time physics simulation was effective for most tasks, but further optimization is needed for highly dynamic or intricate tasks to match the precision required in real-world applications.

4.3. Limitations

The primary limitation identified in the study was the reliance on a manual control system. While functional, this approach restricted the scalability of the simulation and its applicability to automated industrial workflows. Incorporating sensor-based controls or advanced path-planning algorithms could significantly enhance the system's ability to emulate real-world industrial environments. Additionally, the absence of high-fidelity collision detection for complex geometries in Unity posed challenges in accurately simulating intricate object interactions. Addressing this limitation would improve the simulation's precision, particularly for scenarios involving dynamic environments. Future advancements in integrating real-time data processing with the robotic arm's control system could further enhance accuracy, responsiveness, and adaptability, making the simulation more aligned with the demands of modern industrial automation.

4.4. Virtual Reality Integration for Enhanced Control and Training

Unity's integration with virtual reality (VR) technologies offers significant potential for immersive control and training applications. By utilizing VR gloves or motion controllers, operators can interact with and manipulate the robotic system within a virtual environment, enabling intuitive and precise control over its functions. This VR-driven approach not only enhances the user experience but also provides a highly effective platform for operator training. Personnel can safely practice complex robotic tasks in a simulated setting, reducing the risks and costs associated with on-site training and ensuring better preparedness for working with physical robotic systems.

5. Conclusion

This study successfully developed a comprehensive simulation of a KUKA robotic arm in Unity 3D, effectively replicating a pick-and-place operation within a virtual environment. The simulation allowed the robot to interact with key components, with its movements manually controlled via keyboard inputs. This work underscores the significant benefits of digital simulation in industrial automation, particularly in minimizing the costs and inefficiencies associated with physical prototyping and testing.

Although the current implementation lacks advanced trajectory planning and automation, the established framework provides a robust foundation for future advancements. The next phase of this research will focus on integrating trajectory planning algorithms, enabling the

robot to autonomously navigate its environment. Additionally, Virtual Reality (VR) and digital twin technologies will be incorporated to create immersive training environments and facilitate real-time monitoring, thereby enhancing operational efficiency.

This project highlights the transformative potential of digital simulations as a critical tool for optimizing automation workflows. The integration of VR and digital twin technologies in future iterations promises to revolutionize operator training and significantly enhance the effectiveness of industrial robotic systems.

References

[1] Chen, S., et al., Digital Twin Virtual Welding Approach of Robotic Friction Stir Welding Based on Co-Simulation of FEA Model and Robotic Model. Sensors, 2024. 24(3): p. 1001.

[2] Li, Y., et al., Advances in the Application of AI Robots in Critical Care: Scoping Review. Journal of Medical Internet Research, 2024. 26: p. e54095.

[3] Li, B., et al. Unity 3d-based simulation data driven robotic assembly sequence planning using genetic algorithm. in 2022 14th International Conference on Computer and Automation Engineering (ICCAE). 2022. IEEE.

[4] Moshayedi, A.J., et al., Design and Development of FOODIEBOT Robot: From Simulation to Design. IEEE Access, 2024.

[5] Moshayedi, A.J., et al., Robots in Agriculture: Revolutionizing Farming Practices. EAI Endorsed Transactions on AI and Robotics, 2024. 3.

[6] Zarei, M., et al. Indoor UAV object detection algorithms on three processors: implementation test and comparison. in 2023 3rd International Conference on Consumer Electronics and Computer Engineering (ICCECE). 2023. IEEE.

[7] Osmani, K. and D. Schulz, Comprehensive Investigation of Unmanned Aerial Vehicles (UAVs): An In-Depth Analysis of Avionics Systems. Sensors, 2024. 24(10): p. 3064.

[8] Singh, B., N. Sellappan, and P. Kumaradhas, Evolution of industrial robots and their applications. International Journal of emerging technology and advanced engineering, 2013. 3(5): p. 763-768.

[9] Krüger, J., T.K. Lien, and A. Verl, Cooperation of human and machines in assembly lines. CIRP annals, 2009. 58(2): p. 628-646.

[10] Kyrarini, M., et al., A survey of robots in healthcare. Technologies, 2021. 9(1): p. 8.

[11] Marchuk, V.Y., O. Harmash, and O. Ovdiienko, World trends in warehousing logistics. Intellect. Logist. Supply Chain. Manag, 2020. 2: p. 32-50.

[12] da Silva, N.P., S. Eloy, and R. Resende, Robotic construction analysis: simulation with virtual reality. Heliyon, 2022. 8(10).

[13] Li, Y., G. Li, and X. Wang, Research on Trajectory Planning of Autonomous Vehicles in Constrained Spaces. Sensors, 2024. 24(17): p. 5746.

[14] Armansyah, A., et al., Optimization of air suspension system for improved ride and handling performance in road vehicles dynamic. Mechanical Engineering for Society and Industry, 2024. 4(2).

[15] Latifi Rostami, S.A., et al., Robust topology optimization of continuum structures with smooth boundaries using moving morphable components. Structural and Multidisciplinary Optimization, 2023. 66(6): p. 121.

[16] Devan, P.A.M., et al., A novel hybrid harris hawk-arithmetic optimization algorithm for industrial wireless mesh networks. Sensors, 2023. 23(13): p. 6224.

[17] Žlajpah, L., Simulation in robotics. Mathematics and Computers in Simulation, 2008. 79(4): p. 879-897.

[18] Cyril, X., G. Jaar, and J. St.-Pierre. Advanced space robotics simulation for training and operations. in Modeling and Simulation Technologies Conference. 2000.

[19] Xiong, L., et al., An optimized trajectory planner and motion controller framework for autonomous driving in unstructured environments. Sensors, 2021. 21(13): p. 4409.

[20] Pham, Q.-C., Trajectory planning. Handbook of Manufacturing Engineering and Technology, 2015: p. 1873-1887.

[21] Hou, S., et al., A data-driven approach for motion planning of industrial robots controlled by high-level motion commands. Frontiers in Robotics and AI, 2023. 9: p. 1030668.

[22] Moshayedi, A.J., et al., Deep learning application pros and cons over algorithm deep learning application pros and cons over algorithm. EAI Endorsed Transactions on AI and Robotics, 2022. 1(1).

[23] Boscariol, P., A. Gasparetto, and L. Scalera, Path planning for special robotic operations, in Robot Design: From Theory to Service Applications. 2022, Springer. p. 69-95.

[24] Hwang, P.-J., et al., Vision-Based Learning from Demonstration System for Robot Arms. Sensors, 2022. 22(7): p. 2678.

[25] Savir, S., et al., Virtual reality: The future of invasive procedure training? Journal of cardiothoracic and vascular anesthesia, 2023.

[26] Durojaye, A., et al., Immersive horizons: exploring the transformative power of virtual reality across economic sectors. EAI Endorsed Transactions on AI and Robotics, 2023. 2.

[27] Durojaye, A., A. Kolahdooz, and A. Hajfathalian, Enhancing Virtual Reality Experiences in Architectural Visualization of an Academic Environment. EAI Endorsed Transactions on AI and Robotics, 2025. 4.

[28] Pan, J., et al. Real-time VR simulation of laparoscopic cholecystectomy based on parallel position-based dynamics in GPU. in 2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR). 2020. IEEE.

[29] Sun, F., et al., Digital-Twin-Assisted Skill Learning for 3C Assembly Tasks. IEEE Transactions on Cybernetics, 2024.

[30] Peng, H., N. Shi, and G. Wang, Remote sensing traffic scene retrieval based on learning control algorithm for robot multimodal sensing information fusion and human-machine interaction and collaboration. Frontiers in neurorobotics, 2023. 17: p. 1267231.

[31] Unity Technologies, "Unity 3D". Available from: https://unity.com/.

[32] Ding, L., et al. Numerical Calculation and Optimization Algorithm Based on Unity Physics Engine. in 2024 IEEE 13th International Conference on Communication Systems and Network Technologies (CSNT). 2024. IEEE.

[33] Moshayedi, A.J., A. Kolahdooz, and L. Liao, Unity in Embedded System Design and Robotics: A Step-by-step Guide. 2022: Chapman and Hall/CRC.

[34] Jangra, S., et al. Interactivity Development Using Unity 3D Software and C# Programming. in 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT). 2023. IEEE.

[35] Ortega, A., et al., Composable and executable scenarios for simulation-based testing of mobile robots. Frontiers in Robotics and AI, 2024. 11: p. 1363281.
[36] Pan, X., et al. Virtual Assembly of educational robot parts based on VR technology. in 2019 IEEE 11th International Conference on Engineering Education (ICEED). 2019. IEEE.
[37] Al-Tawil, B., et al., A review of visual SLAM for robotics: evolution, properties, and future applications. Frontiers in Robotics and AI, 2024. 11: p. 1347985.
[38] Moshayedi, A.J., et al., Integrating virtual reality and robotic operation system (ROS) for AGV navigation. EAI Endorsed Transactions on AI and Robotics, 2023. 2.
[39] EK Studio, "Low poly factory machine pack demo". Jun 3 [cited 08/19/2024; Available from: https://assetstore.unity.com/packages/3d/props/industrial/low-poly-factory-machine-pack-demo-272637.
[40] 3.D. Unity, "Unity AssetStore". Available from: https://assetstore.unity.com.
[41] real virtual, "Game4Automation parts4cad". 08/05/2024]; Available from: https://assetstore.unity.com/packages/3d/environments/industrial/game4automation-parts4cad-150998#description.

Appendix A:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ShoulderMovement : MonoBehaviour
{
    public Transform ShoulderJoint;
    public float rotationalSpeed = 50f;
    public float minAngle = -170f;
    public float maxAngle = 170f;


    // Update is called once per frame
    void Update()
    {
        if (Input.GetKey(KeyCode.Q))
        {
            RotateJoint(Vector3.up, rotationalSpeed * Time.deltaTime

        }
        if (Input.GetKey(KeyCode.A))
        {
            RotateJoint(Vector3.up, -rotationalSpeed * Time.deltaTim
        }
    }

    void RotateJoint(Vector3 axis, float angle)
    {
        ShoulderJoint.Rotate(axis,angle);
        Vector3 currentRotation=ShoulderJoint.localEulerAngles;
        float currentAngle=currentRotation.y>180?currentRotation.y-
360:currentRotation.y;
        currentAngle=Mathf.Clamp(currentAngle,minAngle,maxAngle);
        currentRotation.y=currentAngle;
        ShoulderJoint.localEulerAngles=currentRotation;
    }

}
```

Figure 1-A: C# Script for Shoulder Movement