Study on robot simulation and code generation softwares

Milja Sebastian[1], Riya Roy[1], Sandhra Shaji[1], Therese Siby[1], and Divya Sunny[2]

[1]Dept.of Computer Science and Engineering,St.Joseph's College of Engineering and Technology,Palai,Kerala
[2]Assistant Professor,Department of CSE,St.Joseph's College of Engineering and Technology,Palai,Kerala

***Abstract*—The field of robotics has advanced significantly in recent years, with the emergence of sophisticated robot simulation and code generation software tools. These tools have revolutionized the way robots are designed, programmed, and deployed in real-world scenarios. This study provides an overview of the various robot simulation and code generation software tools available today, their key features, and their applications in different domains such as manufacturing, healthcare, and agriculture. The study also explores the benefits and challenges of using these software tools, and discusses future directions in this rapidly evolving field. The study could highlight some of the popular robot simulation and code generation software tools such as ROS, Gazebo, V-REP, MATLAB/Simulink, and RobotStudio. It could discuss how these tools can help reduce the time and cost associated with building and testing physical robots by allowing for virtual testing and prototyping. The study could also examine the role of these tools in facilitating robot-human collaboration, as well as in addressing safety and ethical considerations in robot design. The abstract could mention the potential impact of these software tools on the future of robotics, particularly in terms of enabling the development of more intelligent, autonomous, and adaptable robots. Finally, the study could highlight some of the limitations and future research directions for these tools, such as the need for more accurate and realistic simulations, better integration with physical hardware, and improved code optimization techniques.**

I. Introduction

Robot simulation and code generation software are tools used in the field of robotics to design, test, and deploy robotic systems. They are essential in developing robotic applications for various industries, including manufacturing, healthcare, and entertainment.Robot simulation software allows engineers to design and test robots in a virtual environment before they are built. This can save time and money in the design process by allowing for changes to be made and tested quickly and easily. The software can simulate the behavior of the robot under various conditions, including different types of terrain, lighting, and obstacles.

Code generation software, on the other hand, is used to automate the process of generating code for a robot. It takes the design specifications from the simulation software and generates the code needed to control the robot's movements and behaviors. This can save a significant amount of time and effort in the programming process.There are several different types of robot simulation and code generation software available, each with its own strengths and weaknesses. Some popular examples include ROS (Robot Operating System), Gazebo, MATLAB Robotics System Toolbox, and Simulink. These tools offer various features, such as the ability to simulate different types of sensors and actuators, generate real-time code, and integrate with other software and hardware platforms.

Overall, the use of robot simulation and code generation software is becoming increasingly common in the robotics industry. They provide a powerful set of tools for engineers to design and test robotic systems quickly and accurately, ultimately leading to more efficient and cost-effective robotics development.

II. LITERATURE REVIEW

A. Web-Based Multi-Robot Programming and Simulation Tool[1]

The web-based multi-robot programming and simulation tool called Assembly aims to simplify the development and deployment of robotic systems by providing a user-friendly interface that allows users to program and simulate the behavior of multiple robots. Assembly is based on a distributed architecture that allows the simulation of robots to be executed on multiple machines, which makes it scalable and efficient. The tool supports various programming languages, including Python and JavaScript, and provides a set of pre-built simulation scenarios that can be customized by users. The tool describes the features and architecture of Assembly and presents a case study that demonstrates its use in the development of a multi-robot system for warehouse automation. The results of the case study show that Assembly can significantly reduce the time and effort required to develop and deploy robotic systems, making it a valuable tool for researchers and developers in the field of robotics.

B. IRoSim[2]

The implementation of a new manufacturing system requires the design and simulation of the robotic work-cells in order to validate the chosen system configuration and the robotic cell layout. IRoSim is a generic, extensible and flexible platform that uses CAD and knowledgeware technologies for the design, planning and optimization of industrial robotic work-cells. The platform enables the configuration and validation of the robotic cell in terms of the geometry, the robot kinematics, and the

programming of robot motions. It also allows the optimization of the robot motion planning and simulation through the development and use of optimization algorithms. The platform is developed using a Model Driven Architecture approach and supports the modeling of robot kinematics and the creation of simulation models using a CAD tool. A set of performance indicators is also provided to evaluate the effectiveness of the robot work-cells during simulation. The proposed platform is illustrated using a case study of a work-cell consisting of an industrial robot with a vision sensor and a conveyor. The results show that IRoSim provides a complete and integrated platform for the design, planning, optimization and simulation of industrial robotic work-cells.

C. Automation in Robotics Software Systems[3]

The integration of collaborative robots and automation software systems with smart factories is an essential component of Industry 4.0. There is a need for a systematic design process for developing collaborative robotics and automation software systems that are adaptive and dynamic, using a modular architecture. The design process is based on software quality attributes, including scalability, flexibility, extensibility, and modifiability. The implementation of a case study that integrates collaborative robots with a smart factory is also performed. The implementation demonstrates the effectiveness of the proposed design process by providing a comprehensive, scalable, and adaptive system that can be modified to accommodate different application scenarios. The system is designed to be extensible and provides a platform for future improvements and modifications. The results show that the proposed design process is effective in providing a flexible, scalable, and adaptive system for integrating collaborative robotics and automation software systems with smart factories. The system can be modified to accommodate different application scenarios and is designed to be extensible, providing a platform for future improvements and modifications.

D. WORKSPACE[4]

WORKSPACE is a microcomputer-based industrial robot simulator and off-line programming system. The system is developed for educational and research purposes and offers a cost-effective solution for robot programming and simulation. It provides the user with a graphic user interface for creating robot programs and simulating the robot's movements. The system also offers an extensive library of predefined robot movements and tasks that can be easily customized and combined to create complex programs. The programs can be tested and debugged in the simulator before being executed on the real robot, which minimizes the risk of errors and damages. The system's hardware and software architecture, its features, and its potential applications in industry and education is also described. The system's effectiveness is demonstrated by several examples of robot programs and simulations created using WORKSPACE.

E. Quori[5]

The development of social robots that interact with humans requires that the design process is grounded in people's everyday experiences and cultural expectations. The development of Quori, a socially interactive humanoid robot designed through a community-informed process is studied. The design process consisted of three stages: (1) eliciting user requirements through surveys and interviews, (2) iterative prototyping and participatory design sessions, and (3) user evaluation studies. The resulting robot is designed to be expressive, approachable, and responsive to people's social cues. The robot's capabilities include speech and face recognition, object manipulation, and gaze and posture control. The robot's design is intended to support the development of social skills in children with autism spectrum disorder, as well as to provide a platform for studying social interaction in human-robot interaction research. Overall, the community-informed design approach was found to be effective in creating a robot that is acceptable, useful, and engaging for human users.

F. Code Generator Composition for Model-Driven Engineering[6]

A methodology and a set of tools for Model-Driven Engineering of component connector systems in robotics is defined here. The approach is based on the development of a domain-specific language (DSL) for connector specification and the use of model transformations to generate code that implements connectors in the context of a robotic system. The language, which captures the semantics of connector composition, and the code generator, which produces executable code in a target language is described. The approach is validated through the development of a case study, which involves the integration of components from different robotics platforms using the connector framework. The results of the case study demonstrate the effectiveness of the approach in terms of the reduction of development time and the simplification of the integration process. In conclusion the methodology and tools presented can be used to increase the efficiency and effectiveness of the development of robotics systems.

G. RobCoGen[7]

The development of articulated robots is challenging because it requires specialized knowledge and significant programming expertise. Here RobCoGen, a domain-specific language (DSL) that generates C++ code for the kinematics and dynamics of articulated robots is presented. The DSL allows

for a high level of abstraction and compact code, while still producing efficient code. The generated code can be used on different robot control frameworks, and RobCoGen has been tested on both simple and complex articulated robots. The results of the experiments showed that RobCoGen generates efficient code compared to other code generators. The developed code is consistent and always correct because it is generated by the DSL. Additionally, it was shown that the generated code could be used on various platforms, including on microcontrollers. This contributes to the development of articulated robots by providing an effective and efficient code generator for the kinematics and dynamics of articulated robots

H. Automatic Code Generation for Set-plays Design[8]

Automatic code generation a tool that allows the design of new set plays for team sports, such as soccer. The tool is based on an event-driven architecture and a visual programming language that simplifies the creation of new behaviors for the players. However, the tool required a considerable amount of code to be generated for each new behavior, which increased the complexity of the system. In order to tackle this issue, the authors proposed an automatic code generation module that allows the tool to generate the necessary code for a new behavior. The module is based on a model-driven approach, where the behavior is defined using a set of parameters and the code is generated automatically. The authors evaluated the performance of the module and concluded that it was able to generate correct code for a range of behaviors, with a significant reduction in the time and effort required. The proposed approach has the potential to simplify the development of new behaviors in sports, allowing coaches and players to focus on the tactics and strategies, rather than the technical details of the code.

I. Intelligent Robot Control Simulation Software[9]

A simulation software package for the control of intelligent robots is presented. The software, which is still under development, is designed to provide a simulation environment in which new robot control strategies can be tested and evaluated. The software incorporates several features that are intended to make the simulation environment as realistic as possible, including the ability to simulate a wide range of sensor data, dynamic models of the robot, and a user-friendly interface for controlling the robot. The authors argue that the software will be an important tool for the development of new robot control strategies, and will also be useful for training and education in the field of robotics. In conclusion the limitations of the software, and suggests several directions for future development are discussed.

J. Study on robot simulation and monitoring system based on PC[10]

A new robot simulation and monitoring system has been developed. Running on Pentium II PCs with Windows 95 operation system and being developed with OpenGL, this system has the capacity of real-time simulating the motion of an industrial robot through 3D animation with ray tracing [10]. Connected with the robot controller via network, users can monitor the behavior of the robot dynamically or even directly control the action of the robot if necessary. By comparing with the traditional off-line programming system, the framework and the functions of this system and the hardware and software platform of the system are described. The principle of 3D-motion simulation and both the geometry modeling and kinematics modeling are discussed in more detail.

K. XML and P code for robot motion control[11]

IRL (Lola Industrial Robot Language) is procedural language for industrial robot programming. Offline and real-time programming system are main parts of robot programming system based on L-IRL programming language [11]. System for robot programming is using object code as one of the main communication tools between different elements of the system.Also it presented how object code of the compiler can be generated in the form of P code or XML code. This gives software solution that is compact, portable and easy to use with standardized object code that consist information for robot motion control. After generation of object code on the offline part of the system, it is sent to the real-time part of the system using CORBA protocol.

L. Fault-tolerant computing for robot kinematics using linear arithmetic codes[12]

A fault-tolerant parallel processing method based on the linear arithmetic code is proposed for computation of robot kinematics [12]. This method can detect and correct online a single error in the resultant matrix or vector with low hardware and time redundancy. The method also can be extended to multiple errors in computation. It can be applied to various hardware architectures for robot kinematics computation.

M. Part Position Correction During Assembly According to Force and Torque Sensor Signals[13]

The issues of algorithmic support when using force-torque sensing for robotic assembly of shaft-hole type connections [13]. The assembly system consists of an ABB IRB 140 industrial robot equipped with a Schunk GSM-P-64-E-180

pneumatic gripper and a Gamma SI-65-5 six-component force-torque sensor by ATI Industrial Automation. The robot positioning accuracy is insufficient for precision joints assembly. This can cause misalignments (linear, angular) of the parts axis when performing a technological assembly. This can lead to unwanted contact interactions. If the robot movement strategy remains unchanged, the phenomenon of jamming may occur. Therefore, it is proposed to use force-torque sensing to recognize contact interaction at assembly. The sensor is installed in front of the gripper of the assembly manipulator. The difficulty in solving this problem is that the force-torque sensor signals do not contain direct information about the position or orientation of the connected parts. To correct the robot movement it is necessary to recognize the position of the parts contact point by means of force and torque sensor signals. For this purpose, the equations of the forces - torque relationship were derived. The sensor whose parameters determine the position of the parts contact point measures force and torque values. The article considers the case of a one-point contact of the connected parts, which occurs at an unsuccessful matching. Based on a computer simulation of the process, it became possible to establish some common factors between the signs of force effects and the contact point position. The main purpose of this research is to develop an adaptive algorithm that corrects the movement of an assembly robot. The algorithm is based on the information about the forces and torques from the sensor, as well as the data on the operating element position. The use of a position-force control algorithm for robot movement will make it possible to perform a reliable assembly of precision joints.

N. Robot programming in high speed assembly applications[14]

Robot programming in high speed assembly applications presents work about robot control architecture, assembly planning and task planning for manufacturing robots [14]. The interface between an offline planning unit and control systems is handled with skill primitives. Thus, skill primitives and skill primitive nets are explained in detail. Our long term aim is to combine robot control with task and assembly planning, so that with less human interaction manufacturing costs can be reduced. Even parallel kinematic machines provide enormous opportunities to reduce cycle times and thus the benefit should not be wasted by expensive specialized robot programming. Thus, we give an overview of our system and focus on some aspects to implement such a sophisticated system.

O. Realistic robot simulation: multiple instantiating of robot controller software[15]

To realise the concept of a "Virtual Factory" a robot manufacturer's challenge is to provide a "virtual" model of their controller software which is finally to be embedded in different simulation systems [15]. Within a simulation environment, in addition to the standard controller features, further simulation specific functionalities have to be provided. During the project "Realistic Robot Simulation" the functional requirements and a standardized communication interface of a Virtual Robot Controller (VRC) has been specified. One of the strongest demands for such a Virtual Robot Controller for a robot manufacturer is the capability of running multiple instances of the virtual controller software. The results of our efforts towards multiple instantiating of our dedicated robot controller software are discussed. Thus it succeeded in running two controller instances and one simulation system on a off the shelf personal computer. Thus, we verified the potential of creating robot controller systems for coming e-manufacturing processes.

P. Enhancements in a Social Robotic Simulator for Indoor Environments[16]

Social robotics represents a branch of the Artificial Intelligence (AI) area, which aims to develop robots to work in unstructured environments in partnership with human beings. Research in AI and particularly in cognitive science are catalysts in the development of social robotics as they model the ability to obtain, process, store information and direct actions performed by a robot. A simulator, Robot House Simulator (RHS), is proposed to validate robotic systems controlled by cognitive architectures [16]. The main objective of RHS is to minimize the effort made by the developer of cognitive systems in obtaining information associated with the actuators and sensors of the robot. For this, the simulator makes use of high-level abstraction commands and the standardization of sensory information through the OntPercept ontology, which is being also developed. Some experiments that demonstrate the simulator's ability to provide a validation environment for cognitive architectures directed to social robotics are presented. This environment is a pioneer in the integration of a simulator and a cognitive architecture, as well as providing rich sensory information, highlighting the unprecedented modeling available for the senses of smell and taste.

Q. Visual Servoing of Robotic Manipulators[17]

Position-based visual servoing (PBVS) approach to control the motion of multiple robotic manipulators is presented. The PBVS method uses a visual feedback system to estimate the position and orientation of the target object and generate control signals to drive the manipulators to the desired position. The proposed method is verified in the Gazebo simulator, which allows for the creation of a realistic virtual environment. The results demonstrate that the PBVS method can effectively control the motion of multiple robotic manipulators in a coordinated manner, while ensuring that they converge to the desired position and orientation. The study highlights

the potential of the proposed method for use in real-world applications involving multiple robotic manipulators, such as industrial manufacturing and warehouse automation.

R. Robotic kinematics teaching system[18]

A new teaching system for robotic kinematics that integrates virtual reality, remote control, and on-site laboratory resources is discussed. The system offers an interactive and comprehensive learning environment, which enables students to acquire practical experience in kinematics and improve their programming, simulation, and control skills. By incorporating virtual reality and remote control technologies, the system provides access to a variety of robotic equipment and systems, allowing students to experiment and learn from anywhere. An on-site laboratory is also available, where students can gain hands-on experience with robotic hardware and software. The effectiveness of the system is demonstrated through an evaluation of student performance and feedback. The potential of this approach is to enhance the teaching of robotic kinematics, and suggests that it could be adapted to other areas of engineering education.

III. Conclusion

In conclusion, the study on robot simulation and code generation software has shown that these tools have great potential in facilitating the development and deployment of robotic systems. Through the use of these tools, developers can simulate the behavior of robots in different scenarios and environments, test and refine their algorithms, and generate code that can be easily deployed on real robots. These software tools have been shown to be useful for various types of robotic systems, from industrial robots to mobile robots and even autonomous vehicles. The study on robot simulation and code generation software is an active and rapidly evolving field of robotics research. There are many software packages available that offer varying levels of functionality, usability, and compatibility with different robot platforms. However, there are still some limitations to the current state of these tools. Some of the challenges identified in the study include the complexity of the software, the need for specialized skills to operate them effectively, and the limitations in the accuracy of the simulations. Additionally, the study highlights the need for more research and development to further improve the performance of these tools and address the limitations.

Despite these challenges, the study concludes that the use of robot simulation and code generation software will continue to grow as the demand for robotics in various industries increases. This will lead to the development of more sophisticated and user-friendly tools that will facilitate the development of high-performance robotic systems.

References

[1] T.B. Ionescu (2022), Assembly: A Web-Based Multi-Robot Programming and Simulation Tool, in ScienceDirect, vol. 55, pp. 313-318, doi: 10.1016/j.ifacol.2022.04.212.

[2] K. Baizid et al. (2016) IRoSim: Industrial Robotics Simulation Design Planning and Optimization platform based on CAD and knowledgeware technologies, in ScienceDirect, vol. 42, pp. 121-134, doi: 10.1016/j.rcim.2016.06.003.

[3] Z. Salcic, U. D. Atmojo, H. Park, A. T. -Y. Chen and K. I. -K. Wang, Designing Dynamic and Collaborative Automation and Robotics Software Systems, in IEEE Transactions on Industrial Informatics, vol. 15, no. 1, pp. 540-549, Jan. 2019, doi: 10.1109/TII.2017.2786280.

[4] J. Owens, "WORKSPACE-a microcomputer-based industrial robot simulator and off-line programming system," IEE Colloquium on Next Steps for Industrial Robotics, 1994, pp. 4/1-4/4.

[5] A. Specian et al. (2021) Quori: A community-informed design of a socially interactive humanoid robot, arXiv.org, doi: 10.48550/ARXIV.2109.00662.

[6] A. Roth et al. (2015). Language and Code Generator Composition for Model-Driven Engineering of Robotics Component Connector Systems. Journal of Software Engineering for Robotics (JOSER), vol. 6.

[7] Marco, Frigerio, et al. "RobCoGen: A Code Generator for Efficient Kinematics and Dynamics of Articulated Robots, Based on Domain Specific Languages." RobCoGen: A Code Generator for Efficient Kinematics and Dynamics of Articulated Robots, Based on Domain Specific Languages, 1 Jan. 2016, aisberg.unibg.

[8] SALES, Raoni Magalhães Mascarenhas, Ana Patrícia Simões, Marco Souza, Josemar. (2020). Automatic code generation for new behaviors in a tool for set plays design. 10.29327/118637.1-8.

[9] "Intelligent Robot Control Simulation Software," RO-MAN 2007 - The 16th IEEE International Symposium on Robot and Human Interactive Communication, 2007, pp. 656-656, doi: 10.1109/ROMAN.2007.4415166.

[10] Di Wang, Yimin Chen, Minglun Fang and Yongyi He, "A study on a new robot simulation and monitoring system based on PC," Proceedings of the 3rd World Congress on Intelligent Control and Automation (Cat. No.00EX393), 2000, pp. 1328-1332 vol.2, doi: 10.1109/WCICA.2000.863460.

[11] M. Lutovac, G. Ferenc, J. Vidaković, Z. Dimić and V. Kvrgić, "Usage of XML and P code for robot motion control," 2012 Mediterranean Conference on Embedded Computing (MECO), 2012, pp. 162-165.

[12] J.Y. Han, "Fault-tolerant computing for robot kinematics using linear arithmetic codes," Proceedings., IEEE International Conference on Robotics and Automation, 1990, pp. 285-290 vol.1, doi: 10.1109/ROBOT.1990.125988.

[13] S. V. Kuznetsova and A. L. Simakov, "Part Position Correction During Assembly According to Force and Torque Sensor Signals," 2021 International Russian Automation Conference (RusAutoCon), 2021, pp. 184-189, doi: 10.1109/RusAutoCon52004.2021.9537359.

[14] U. Thomas, F. M. Wahl, J. Maass and J. Hesselbach, "Towards a new concept of robot programming in high speed assembly applications," 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005, pp. 3827-3833, doi: 10.1109/IROS.2005.1545582.

[15] K. Vollmann, "Realistic robot simulation: multiple instantiating of robot controller software," 2002 IEEE International Conference on Industrial Technology, 2002. IEEE ICIT '02., 2002, pp. 1194-1198 vol.2, doi: 10.1109/ICIT.2002.1189343.

[16] J. P. Ribeiro Belo, H. Azevedo and R. A. F. Romero, "Enhancements in a Social Robotic Simulator for Indoor Environments," 2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE), 2018, pp. 457-463, doi: 10.1109/LARS/SBR/WRE.2018.00087.

[17] S. Noh, C. Park and J. Park, "Position-Based Visual Servoing of Multiple Robotic Manipulators: Verification in Gazebo Simulator," 2020 International Conference on Information and Communication Technology Convergence (ICTC), 2020, pp. 843-846, doi: 10.1109/ICTC49870.2020.9289554.

[18] Xu X, Guo P, Zhai J, Zeng X. Robotic kinematics teaching system with virtual reality, remote control and an on–site laboratory. International Journal of Mechanical Engineering Education, vol. 48, pp. 197-220, doi:10.1177/0306419018807376.