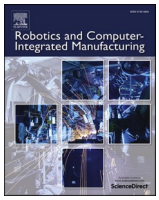
Robotics and Computer–Integrated Manufacturing 92 (2025) 102862



Contents lists available at ScienceDirect

Robotics and Computer-Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim



Rapid and automated configuration of robot manufacturing cells

Seemal Asif [a,*], Mikel Bueno [a], Pedro Ferreira [b], Paul Anandan [b], Ze Zhang [c], Yue Yao [c], Gautham Ragunathan [d], Lloyd Tinkler [d], Masoud Sotoodeh-Bahraini [b], Niels Lohse [b], Phil Webb [a], Windo Hutabarat [c], Ashutosh Tiwari [c]

[a] Centre for Robotics and Assembly, Cranfield University, Cranfield, Bedfordshire, UK
[b] Intelligent Automation Centre, Wolfson School of Mechanical, Electrical and Manufacturing Engineering, Loughborough University, Loughborough, UK
[c] Department of Automatic Control and Systems Engineering, The University of Sheffield, Sheffield, UK
[d] The University of Sheffield Advanced Manufacturing Research Centre (AMRC), Sheffield, UK

ARTICLE INFO

Keywords:
Reconfigurable manufacturing
Modular robotics
Common communication platform
Autonomous systems
Robot perception
Autonomous program generation
Reconfigurable architecture
Manufacturing automation

ABSTRACT

This study presents the Reconfigurable and Responsive Robot Manufacturing (R3M) architecture, a novel framework engineered to autonomously adapt to fluctuating product variants and demands within manufacturing environments. At the heart of R3M lies an integrated architecture that ensures a seamless data flow between critical modules, facilitated by an advanced communication platform. These modules are central to delivering a range of services crucial for operational efficiency. Key to the architecture is the incorporation of Automated Risk Assessment aligned with ISO-12100 standards, utilizing ROS2 Gazebo for the dynamic modification of robot skills in a plug-and-produce manner. The architecture's unique approach to requirements definition employs AutomationML (AML), enabling effective system integration and the consolidation of varied information sources. This is achieved through the innovative use of skill-based concepts and AML Class Libraries, enhancing the system's adaptability and integration within manufacturing settings. The narrative delves into the intricate descriptions of products, equipment, and processes within the AML framework, highlighting the strategic consideration of profitability in the product domain and distinguishing between atomic and composite skills in equipment characterization. The process domain serves as an invaluable knowledge repository, bridging the gap between high-level product demands and specific equipment capabilities via process patterns. The culmination of these elements within the R3M framework provides a versatile and scalable solution poised to revolutionize manufacturing processes. Empirical results underscore the architecture's robust perception abilities, with a particular focus on a real-world application in robotic lamination stacking, elucidating both the inherent challenges and the tangible outcomes of the R3M deployment.

1. Introduction

The rapid and automated configuration of robot manufacturing cells has garnered significant attention from both researchers and industry experts over recent years. This interest is driven by the need for manufacturing systems that are not only flexible but can also quickly adapt to changes in the production environment using modular and reconfigurable components. Among the innovations in this field, the Reconfigurable and Responsive Robot Manufacturing (R3M) architecture stands out by integrating Automated Program Generation (APG). This technology represents a shift in how robotic cells are programmed, merging semantic descriptions with artificial intelligence to refine and enhance the sequencing of robotic actions. This approach closely aligns with high-level product requirements and the specific capabilities of the equipment, reducing reliance on human input and pushing the system toward greater autonomy. Over time, as the system's libraries grow, it is expected that the need for human intervention will decrease, making the system increasingly self-reliant. APG also allows the system to dynamically adapt its operations based on learned behaviors, significantly improving its ability to handle diverse and changing production scenarios efficiently.

This was supported part by the EPSRC in under Grant V051180.
* Corresponding Author.
E-mail address: s.asif@cranfield.ac.uk (S. Asif).

https://doi.org/10.1016/j.rcim.2024.102862
Received 15 May 2024; Received in revised form 2 August 2024; Accepted 27 August 2024
Available online 5 September 2024
0736-5845/© 2024 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

1.1. Scope of the paper

This paper provides a thorough analysis of an integrated architecture for manufacturing, covering modules like SoA integration, risk assessment, requirements definition, program generation, perception systems, and a practical use-case scenario. It highlights the significance of seamless data exchange, resource utilization, and automated programming in modern manufacturing. Through structured sections, the paper aims to highlight the architecture's potential to streamline operations, enhance efficiency, and ensure safety in dynamic industrial settings.

1.2. Reconfigurable manufacturing systems

The integration of Industry 4.0 technologies has brought about a transformative shift in manufacturing systems, particularly in addressing the inherent extremes of scale. Traditionally, these systems encompassed largely manual, highly flexible, bespoke, and small-batch production on one end, contrasting with the highly automated yet inflexible mass production on the other as illustrated in *Fig. 1*. However, the advent of Industry 4.0 is altering this landscape.

The established paradigm contends that highly automated processes, while conducive to efficiency and quality, incur prohibitively high development and deployment costs, making them economically unviable for low production volumes. As a result, industries engaged in the manufacture of large, complex, or high-value products (e.g., aerospace) often find it challenging to leverage the benefits of automation, including improved productivity, quality, and traceability, especially when production volumes are limited.

Manufacturing systems often occupy extremes of scale: largely manual yet highly flexible bespoke and small-batch production at one end; and highly automated inflexible mass production at the other. This is illustrated in *Fig. 1*. Due to the high cost of development and deployment such automation is not economical at low production volumes. Consequently, the manufacture of large, complex or high-value products (e.g. aerospace) which might necessarily only be produced at low volumes cannot capitalize on the benefits of automation such as improved productivity, quality, traceability etc.

The advent of mass customization has increased the flexibility of automation, adding value to otherwise standardized products. This is typically achieved by selecting between preconfigured options during manufacture, resulting in a plethora of product variants. However, in the manufacture of high-value products, it may not be known in advance what variations will be required and this approach may be overly restrictive. Moreover, over-specifying the system to include preconfigured options carries an additional cost which might be recouped if not used.

Traditional Manufacturing Systems such as Dedicated Manufacturing Systems (DMS), Job Shops, Flow Lines, Cellular Manufacturing Systems (CMS), as well as Flexible Manufacturing Systems (FMS) all have their own limitations and difficulties to run as per the expectations of the new business/industry portfolio [1]:

- Dedicated Manufacturing Systems (DMS) produce core products at a high production rate with low flexibility. As the main drawback, product features are supposed to be constant during the system lifetime, customization is costly and difficult to implement, and modifications rarely occur [2].
- Flexible Manufacturing Systems (FMS) consist of automated workstations managed through a central control unit. Their main advantage is the flexibility in managing resources to manufacture a large variety of parts, but in most of the cases, the throughput of these systems is lower than for DMS and the dedicate equipment increases the part full cost.
- Cellular Manufacturing Systems (CMS) overcome some limitations of the previous systems. They involve the use of multiple independent working cells dedicated to product families with similar processing requirements [2]. Nonetheless, CMSs are designed to produce a specific set of products with stable demand level and sufficiently long lifecycle.

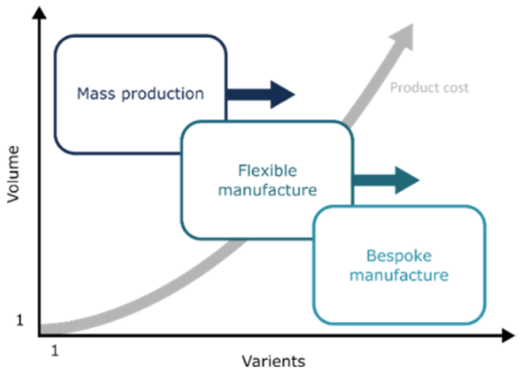


Fig. 1. The Impact of Volume and Variants on Product Cost for Different Manufacturing Processes

Therefore, it would be beneficial to develop flexible or reconfigurable manufacturing systems to occupy the middle-ground between bespoke production and mass production. Such a Reconfigurable Manufacturing System (RMS) is one which can change its functionality through changes in the hardware and software [3]. In contrast to Flexible Manufacturing Systems (FMS), as exemplified by mass customization, the scope of these changes do not need to be specified at the outset. Proponents of RMS argue that the higher initial cost can justified by the lower cost of changes in functionality or increases in capacity, and by the competitive advantage afforded by shorter lead times [4].

RMS support the rapid addition, removal or modification of process controls, functions and operations, through reconfigurable hardware and software, to scale production capability and capacity [5]. As compared to other manufacturing systems, RMSs do provide a solution for the constant alterations to the market and product requirements, and seem to be a solid solution for the future manufacturing and production scenario. Nonetheless, RMSs do have some barriers which still limit industry adoption, such as higher costs, complexity, and lower speeds.

The core characteristics that every single RMS should acquire are [4]:

- Modularity: The ability to design hardware and software components with modular attributes. Optimizing it involves finding as many common modules as possible to be shared among the product families.
- Integrability: The ability to include and integrate new components, modules, resources, or technologies withing the existing system.
- Convertibility: The ability of the system to adjust/evolve functionality over time in response to the dynamic changes.
- Diagnosability: The ability of the system to self-check its current state and diagnose it effectively and rapidly.
- Customization: The variety of the required system modules/components according to the flexibility of processing a product family.
- Scalability: The ability to adapt the production capacity to cope with changes in the system/process.

1.3. Reconfigurable factory demonstrator cell

The Reconfigurable Factory Demonstrator (RECON) cell, shown in *Fig. 2*, was initially developed as a facility for short projects on robotics and automation in an academic research and development environment, and significantly reduces the time and effort required in setting up new processes. The RECON cell features a custom designed flooring system which allows robots, fixtures, and other equipment to be installed in different arrangements. The floor consists of an array of 1 m x 1 m rafts

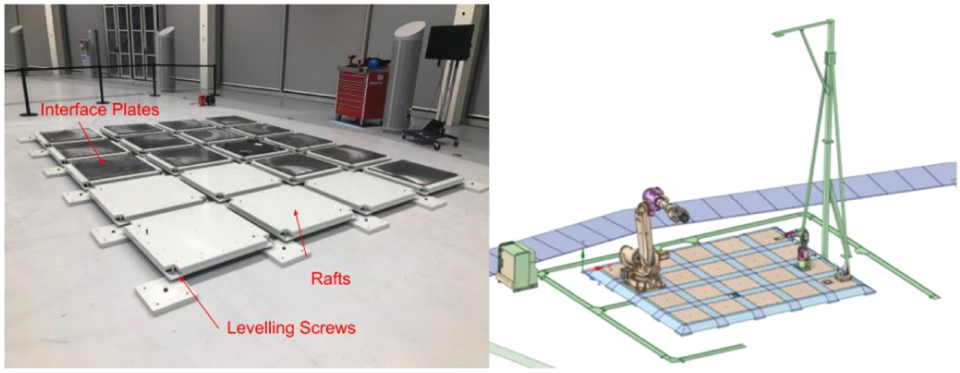


Fig. 2. Reconfigurable Cell Demonstrator. (left) Image of RECON Cell's reconfigurable floor during assembly. (right) CAD model of current RECON Cell containing a large ABB IRB-6640 robot, a small ABB IRB-1200 robot and a tall mounting fixture for sensors.

with steel interface plates mounted on top with regular or specific bolt patterns for equipment. Service trenches run between rafts to allow cables and pipes to be routed through the floor to minimize trip hazards. The accuracy of this floor allows robot programs generated and simulated offline to be deployed with minimal correction to work object coordinate systems.

To aid integrability, the cell uses a distributed I/O (input/output) sensor network using Beckhoff EtherCAT I/O modules which allow rapid integration of new sensors, actuators, and other peripheral devices into the cell. The cell is controlled by a central soft-PLC (software-based programmable logic controller) running on an industrial personal computer (IPC, Beckhoff C6032). This allows bi-directional communication of I/O and safety signals between the soft-PLC and a virtual machine on the IPC running the ROS2-based R3M system architecture, described in Section 2: Architecture Overview. The physical architecture of the RECON cell, together with the use of the R3M software, affords reconfigurability in the control architecture and lends itself to integrating new software nodes.

Whilst RECON meets some of the established core characteristics of an RMS, particularly the modularity, integrability and scalability through its physical architecture, it lacks the intelligence to autonomously change its functionality or diagnose its state. RECON has therefore been adopted as the basis of the R3M project to develop the enabling digital architecture to support this physical hardware and realize a RMS.

1.4. Related work in distributed systems

Recent research in Automated Production Systems has introduced various architectural frameworks aimed at enhancing Overall Equipment Effectiveness (OEE). (Ng [6])One approach emphasizes a technology-agnostic perspective, featuring standardized interfaces and a common data model for the seamless integration of diverse hardware and software components. Another focuses on Service-Oriented Architecture (SOA) [7]frameworks, extending collaboration within Plant-Wide systems. Additionally, researchers have explored innovations such as modular control systems and dynamic optimization using digital twins and AI to address system reconfiguration. Despite these advancements, notable gaps in the current literature include the need for comprehensive standardization in reconfigurable software architecture, direct integration of robot movements in SOA frameworks, safety considerations during dynamic changes, and automated program generation in response to evolving system configurations.

A technology-agnostic architecture presented which aimed at enhancing Overall Equipment Effectiveness (OEE) by managing diverse big data in Automated Production Systems. [8] Their framework features standardized interfaces, a Data Management and Integration Broker, and a common data model, successfully integrating various hardware and software components using different communication protocols. Toquica et al. [9] propose a web-based solution for teleoperating industrial robots with open-source languages and standard communication protocols, employing cloud services, an edge local server, and physical or virtual robot systems, enabling remote control of robots and the development of new actions as services. Additionally, Romiti et al. [10] introduce a software architecture for a rapidly reconfigurable EtherCAT-based robot, organized into module, middleware, and application levels. Their system efficiently controls robots by utilizing URDF and SRDF data for robot configuration. These works collectively highlight the significance of software architecture in enabling reconfigurability and flexibility within manufacturing systems, employing standardized interfaces, data models, and innovative layering structures to manage diverse data and control reconfigurable robots effectively. In contrast, current literature remains focused and very specific, lacking a comprehensive, unified standardization that would enable the usage of a reconfigurable software architecture across different manufacturing domains.

The AESOP project by Karnouskos et al. [11] and Colombo et al. [12] introduced a Service-Oriented Architecture (SOA) system-of-systems framework that extends collaboration horizontally and vertically across diverse levels within a Plant-Wide system, simplifying the integration of monitoring and control systems in manufacturing. Additionally, Leitão et al.'s and Gosewehr eta al.'s PERFoRM system [13,14] employs SOA to connect hardware devices, software applications, and a dedicated Industrial Manufacturing Middleware, transitioning from point-to-point connectivity to a loosely coupled approach. This enables various operations, including data processing, aggregation, presentation, and protection, enhancing vertical integration via a Cloud-Based Integration Layer. Both systems showcase the advantages of SOA in facilitating extensive connectivity and integration within manufacturing environments, but lack the direct integration of robot or end-effector movements as part of a SOA-based service catalogue.

Wojtynek et al. [15] implemented a workcell setup assistance system based on Plug-and-Produce (PnP) principles encompassing both software and hardware aspects. Their software integrates an Identification Module (ID-Module) with a Management Shell for communication organization, employing AML for self-description and standardized connectors for wired networking and power transmission. On the hardware front, universal connectors allow components to connect seamlessly within the modular production cell. Simonic et al. [16] introduced a flexible and modular software design enabling the expansion of cell functionalities through Docker containers, facilitating the integration of new software modules while ensuring compatibility with existing components. These works collectively highlight the significance of PnP principles, utilizing standardized communication structures and modular software designs, allowing for easy integration and expansion of manufacturing systems.

Scheifele et al. [17] introduced a modular control system with a Workflow Manager (WFM) that automatically reconfigures control upon module changes, reducing manual tasks. Using service-oriented and real-time communication, the WFM interacts with higher-level systems, allocating tasks based on module capabilities. Similarly, Mo et al. [18] proposed a method merging digital twins and AI to dynamically reconfigure manufacturing systems, optimizing operations for changing market demands and enhancing system decision-making and performance. While both studies highlight innovative approaches to automatic system reconfiguration (one focusing on real-time control adjustments and the other on dynamic optimization through AI and digital twin integration for improved manufacturing performance), a notable gap in current literature lies in the limited consideration of safety protocols and measures when implementing these dynamic changes within the system.

Zoitl et al. [19] developed a production plan-driven system for automated machine program generation, emphasizing modular software for seamless interchangeability. In contrast, John et al.'s ROBIN [20] introduced an innovative Automatic Code Generation Tool, creating tailored XML excerpts for SoftPLC projects in ROS-controlled systems. This advancement significantly reduces development overhead, resolves design complexities, and enhances system productivity, serving as a pivotal bridge between ROS and PLCs in the realm of automation development. A significant gap in the current literature exists concerning the capability to automatically generate programs when the system undergoes dynamic changes in both hardware and software, as well as the absence of methods to generate programs directly from CAD files, especially when product requirements change in a process.

Several studies in the field of Automated Production Systems have explored different approaches to enhance system efficiency and flexibility. The AESOP project and PERFoRM system introduced Service-Oriented Architecture (SOA) frameworks, emphasizing extensive connectivity and integration within manufacturing environments. However, both lack direct integration of robot or end-effector movements within a SOA-based service catalog. Wojtynek et al. implemented a workcell setup assistance system based on Plug-and-Produce (PnP) principles, showcasing the significance of standardized communication structures and modular software designs for easy integration and expansion of manufacturing systems. Scheifele et al. introduced a modular control system with a Workflow Manager (WFM) for automatic reconfiguration, while Mo et al. proposed a method merging digital twins and AI for dynamic optimization in manufacturing systems. Notably, a gap in current literature involves the limited consideration of safety protocols during dynamic system changes. Zoitl et al. developed a production plan-driven system, and John et al.'s ROBIN introduced an Automatic Code Generation Tool, addressing the need for automated program generation but leaving a gap in capabilities during dynamic changes in both hardware and software configurations, as well as a lack of methods for generating programs directly from CAD files.

The R3M reconfigurable system represents a significant advancement over traditional manufacturing systems by introducing an adaptive framework designed to automatically respond to product variations and demand fluctuations. Unlike the DMS, which are inflexible and costly to customize, R3M offers high adaptability, enabling efficient production adjustments without substantial expense. In comparison to Flexible Manufacturing Systems FMS, which often experience lower throughput and higher costs due to dedicated equipment, R3M incorporates advanced sensor fusion and deep learning to enhance real-time performance and operational efficiency. Moreover, R3M surpasses the CMS by not being constrained to stable demand levels and extended product lifecycles. The system's seamless alignment of product, process, and equipment domains, facilitated by skill-based concepts and APG, ensures robust performance, superior flexibility, and enhanced integration. These features collectively position R3M as a comprehensive and dynamic solution for addressing the complexities of modern manufacturing challenges.

The paper is organized as follows: Section 2 discusses the rapid and automated configuration of robot manufacturing cells, emphasizing the need for flexible, adaptive systems and introducing the R3M architecture. Section 3 delves into the core components of R3M, detailing its integrated architecture, seamless data flow, and critical modules like Automated Risk Assessment and ROS2-based dynamic skill modification. Section 4 explains the use of AML for effective system integration and information consolidation, focusing on mapping product, equipment, and process domains. Section 5 showcases the architecture's real-world application in robotic lamination stacking, highlighting its robust perception abilities and adaptability. Section 6 outlines the key features of R3M, including high-resolution sensing, real-time performance, adaptive algorithms, and environmental robustness, ensuring efficient operations in diverse scenarios. Section 7 examines the system's automated programming and decision-making capabilities, supported by deep learning and AML, to meet dynamic manufacturing demands while maintaining safety through robust risk assessment protocols. Finally, Section 8 discusses future research directions, aiming to optimize recipe generation, automate AML model creation, enhance simulation stability, and improve perception system efficiency, thereby further streamlining manufacturing operations and enhancing the system's responsiveness to evolving needs.

2. Architecture overview

The architecture described in this study leverages the strengths of both SoA and the common communication platform, allowing the Requirements, Program Generation, and Sensing packages to be integrated into a single dataflow process. This integration allows for the seamless exchange of information and resources. The R3M Data Flow is described in the *Fig. 3* below.

The system architecture illustrated in *Fig. 4* of this system is designed with a modular approach, where different components are organized into a set of main modules. These main modules are C3S (Central Cell Control Services), CMS (Cell Management Services), RMS (Resource Management services), RSS (Resource Scheduler Service), PAS (Process Analyzer Services), SPS (Soft PLC Services) and DBMS (Database Management Services).

2.1. Architecture modules

The CMS is the primary module in charge of managing all the cells that are supported by the framework. It is also responsible for communicating between the different components of the architecture, ensuring that the system operates smoothly.

The C3S is responsible for controlling the individual cells and receiving the programs from the Program Generation package. This service also communicates with the resources, such as robots, by using supportive protocols such as TCP/IP, Ethernet IP, Profinet, Modbus, or WiFi. Additionally, the C3S runs the trained module for the sensing package, which is critical for real-time data collection and analysis.

The RMS is responsible for maintaining and managing the resources

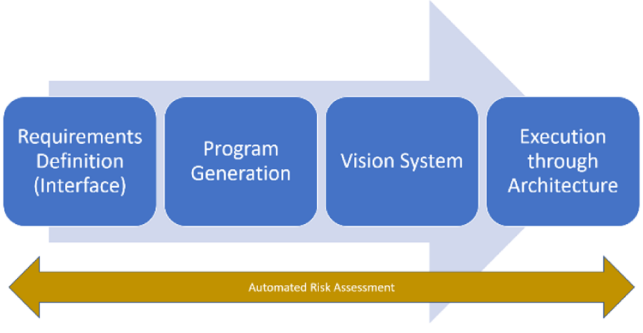


Fig. 3. R3M Data Flow

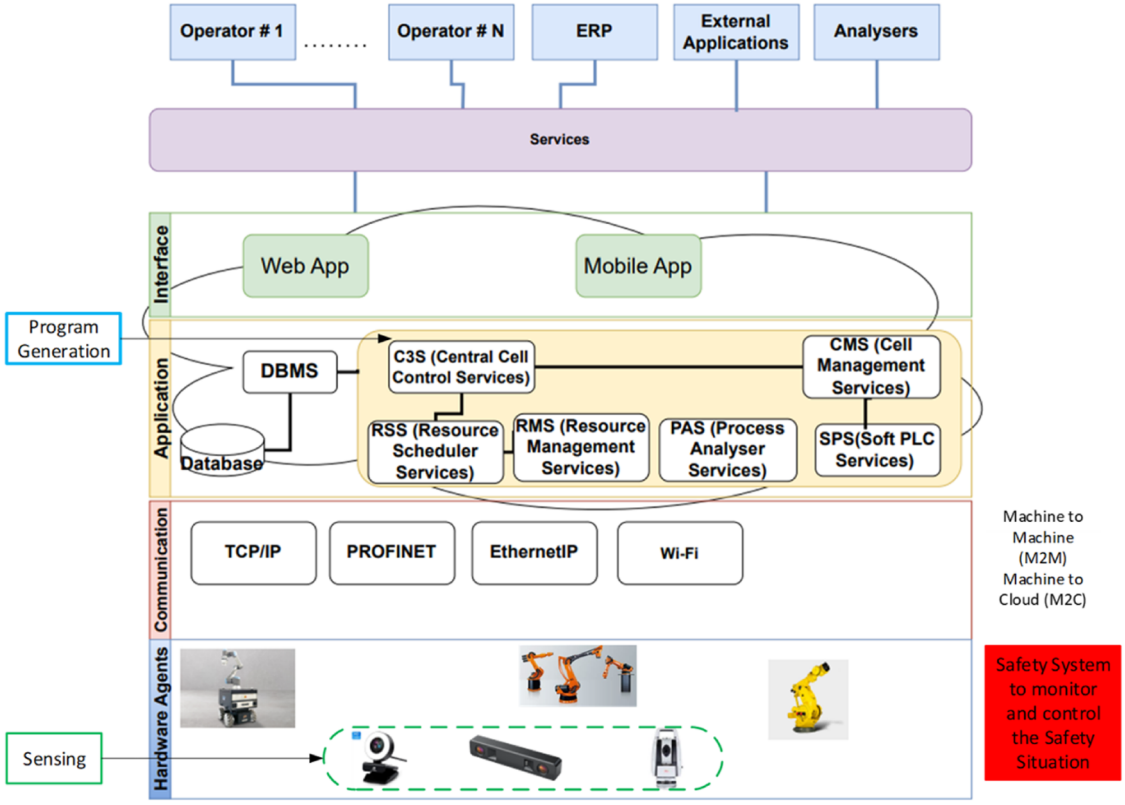


Fig. 4. R3M Software Architecture

used in the production process, such as robots and tools. This module ensures that the resources are properly maintained and that any issues are addressed in a timely manner.

The RSS is in charge of scheduling the resources based on their availability, and it has the ability to resolve and avoid conflicts such as deadlocks. This module enables the architecture to run multiple concurrent robotic cells with the help of the RMS.

The PAS analyses the processes to ensure that they are running efficiently and effectively. This module provides valuable insights into the production process and helps identify areas for improvement.

The SPS provides Soft PLC Services, which allow the architecture to control and monitor the production process through programmable logic controllers. This module is particularly useful for legacy components that cannot be controlled through modern frameworks like ROS2.

Finally, the DBMS is responsible for managing the database and ensuring that all data is stored and organized properly. This module is critical for the proper functioning of the system and the ability to access and analyse data in real-time.

The architecture of R3M system is designed with a modular approach and it is composed of several modules, including the CMS, C3S, RMS, RSS, PAS, SPS, and DBMS. The integration of these modules allows the system to operate smoothly, provide real-time data collection and analysis, and ensure efficient and effective production processes.

2.2. Automated risk assessment

Ensuring the safety of a newly deployed automation system is a key challenge in deploying an RMS, as each configuration may bring new hazards requiring a risk assessment to be conducted, adding necessary overhead to the reconfiguration process. The R3M architecture incorporates an automated risk assessment process that aligns with ISO-12100 standards [21], which provide guidelines for evaluating and managing hazards in production systems. This compliance ensures that the system adheres to internationally recognized safety protocols, enhancing its reliability and safety during operations. Table 1 summarizes the key clauses and requirements of BS EN ISO 12100:2010 related to risk assessment methods, hazard identification, risk level evaluation, risk reduction measures, and safety implementation.

The system employs Unified Modeling Language (UML) to define hazards, tasks, and processes within the manufacturing system. This modeling technique allows for the automation of risk assessments by creating a clear and consistent framework for identifying and evaluating risks. The UML profiles used in R3M define safety-related terms and processes, enabling a systematic approach to risk management. The UML-based nature of the risk assessment process makes it integrable with Object-Oriented Languages (OOL), enhancing its flexibility and extensibility. This integration facilitates seamless communication and data exchange across different system components, leveraging the strengths of OOL for better system modularity and maintainability.

Central to R3M's risk management is real-time monitoring and analysis of the production process. Integrated with the system's architecture, key components such as C3S, CMS, and RMS continuously communicate with the automated risk assessment module. This ensures a cohesive approach to safety, where the risk assessment system dynamically adapts to new configurations and production changes. The real-time capability of the system allows it to promptly identify and mitigate risks as they arise, maintaining both safety and operational efficiency. By incorporating automated risk assessment with advanced technologies like real-time data collection and UML modeling, R3M provides a robust, adaptable, and safe manufacturing environment capable of handling the complexities of modern production demands. This UML-based approach ensures that safety protocols are consistently and effectively managed, reducing human error and improving overall system reliability.

Table 1
Key Clauses and Requirements of BS EN ISO 12100:2010 for Risk Assessment and Risk Reduction.

Section	Clause	Description
Risk Assessment Methods	5.1	General: Comprises risk analysis and risk evaluation.
Risk Analysis	5.3	Determination of Limits of Machinery: Includes space limits, time limits, and other operational and environmental limits.
	5.4	Hazard Identification: Systematically identify foreseeable hazards, situations, and events throughout the machine's lifecycle, including transport, assembly, use, dismantling, and scrapping.
	5.5	Risk Estimation: Evaluate severity of harm and probability of occurrence, considering exposure, hazardous event occurrence, and possibility of avoiding or limiting harm.
Risk Evaluation	5.6	Risk Evaluation: Determine necessity for risk reduction and evaluate the adequacy of measures taken, ensuring no new hazards are introduced by protective measures.
Risk Reduction Measures	6.1	General: Aim to eliminate hazards or reduce risks by addressing severity of harm and probability of occurrence.
	6.2	Inherently Safe Design Measures: Achieve risk reduction through appropriate design features of the machine.
	6.3	Safeguarding and Complementary Protective Measures: Implement safeguards and protective devices, including physical guards and provisions for their maintenance.
	6.4	Information for Use: Provide detailed operating procedures, safe working practices, residual risks, and personal protective equipment requirements. Ensure this information is consistent with users' abilities and includes necessary training.
Documentation and Compliance	7	Documentation: Ensure thorough documentation of the risk assessment and risk reduction processes, including machinery descriptions, regulatory requirements, experiences of use, ergonomic principles, risk reduction measures, residual risks, and risk assessment results.

2.3. Service oriented architecture

To facilitate modularity and integrability, robot movements and actions are implemented as services which can be easily and rapidly added, removed or modified. For example, if a parallel gripper is added into the system (hardware component), the grasping skill would be added into the software side as a callable service. This approach is achieved due to the implementation of ROS2, which provides a service server/client tool that makes it simple to call different services while obtaining direct feedback about the performance and the execution of the specific skill.

The ROS2-based architecture is described in *Fig. 5*. Every unit is integrated into the system through a standard ROS2 wrapper, which converts all the data, functions, and services of each component into standardized information that is used by the C3S, RSS and RMS. This enables the seamless integration of any hardware or software component into the architecture, and offers a common practice for the management, control, and execution of the system.

ROS2 utilizes MoveIt!2 as the main framework for the control of the robot. It is considered as the robotic manipulation platform for ROS2, and incorporates the latest advances in motion planning, manipulation, 3D perception, kinematics, control, and navigation. A robot cell can be controlled using ROS2+MoveIt!2 by providing certain specifications of the cell itself, which are effectively handled by the platform to perform the robot movements and executions. Those specifications include:

- URDF model of the robot: The Universal Robot Description Format model is a collection of files that provide a robot's physical description.
- Control parameters: Numerical values that tune the robot controllers used by MoveIt!2.
- ROS2 Driver: When in simulation, Gazebo offers a physics engine that, combined with open-source ROS2 controllers, is able to effectively simulate any robot provided its URDF model. This solution works for simulation, but is not valid for real robots since hardware manufacturers use custom robot controllers for their robots. Therefore, a ROS2 driver is needed in this case, which translates the proprietary robot data to ROS2 and sends ROS2 control signals to the robot controller.

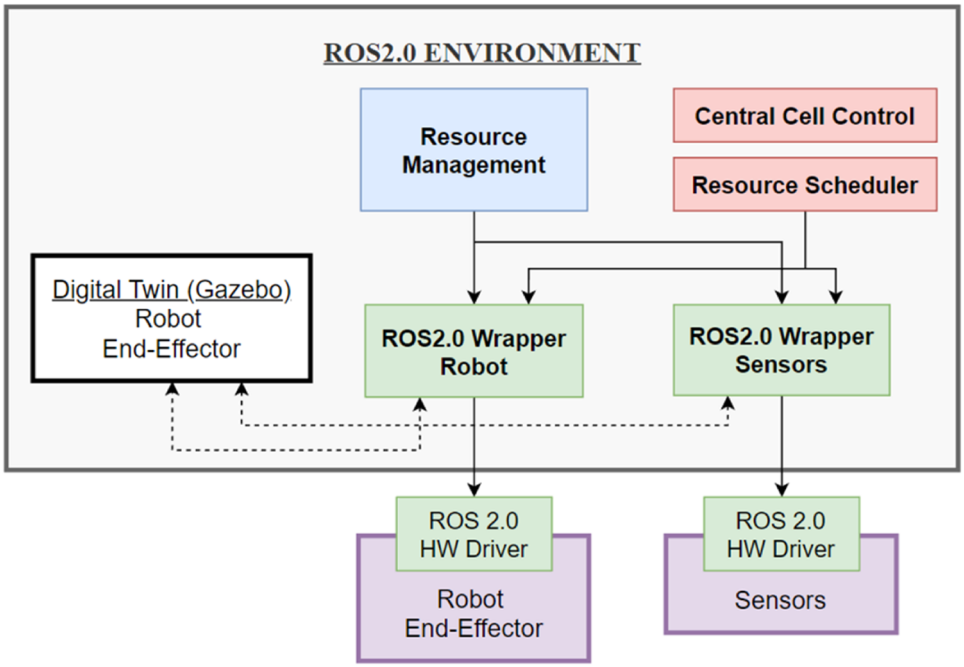


Fig. 5. Simplified Architecture of the R3M-ROS2 Environment

A ROS2-based Software Architecture limits the system to the availability of ROS2 Drivers for the specific hardware components that would potentially form it, but brings a key feature that makes reconfiguration at software level much simpler: The Robot Control framework is hardware/software agnostic.

3. Requirements definition in AML

R3M's vision is to deliver a system that can automatically reprogram itself to the changes in the product variants and their demand (volume). In this project, the hardware or the physical setup of the system is not expected to change significantly. The requirements definition focuses more on capturing all the essential information related to the assembly product and the equipment's capabilities. This information will be used to establish the rules and conditions that are required to reprogram the system. Towards this end, R3M uses AutomationML (AML) to capture all the relevant information for requirements definition. AML is a powerful framework for representing industrial automation information and ensures seamless integration and data exchange across various engineering domains and their tools [22].

Traditionally, the design or configuration of the system is achieved by processing the key relationships and interdependencies between three fundamental domains. These domains include the product to be manufactured, the processes required to produce them, and the equipment used to execute the processes [23]. *Fig. 6* provides an overview of how the requirements definition is built using the information from the three PPR domains. Using common semantic terminologies can significantly enhance the communication between domain experts and system integrators. In line to that statement, the R3M project aims to leverage existing semantic models developed as part of the openMOS project [24].

These models were used for the design, ramp-up, and operation of assembly systems. Usage of several skill-based concepts helps to define the relationships between the Product, Process, and Resource (PPR) domains. These skill-based concepts revolve around three key definitions: "Skill," "Skill Requirements," and "Skill Recipe" as shown in *Fig. 7*. The term "Skill" refers to the functional capability of an equipment module. "Skill Requirements" define specific assembly requirements of the product. And lastly, the "Skill Recipe" represents the parameterized instance of a "Skill" that can fulfil one or more "Skill Requirements". In summary, enabling automatic programming requires matching two key elements captured in the requirements definition. One is the product's 'skill requirements' and other is the 'equipment skills.'

Fig. 8 provides an overview of the AML libraries and their use to establish reusable components of modelling concepts and definitions. In summary, the Role Class (RC) library is used to establish the type

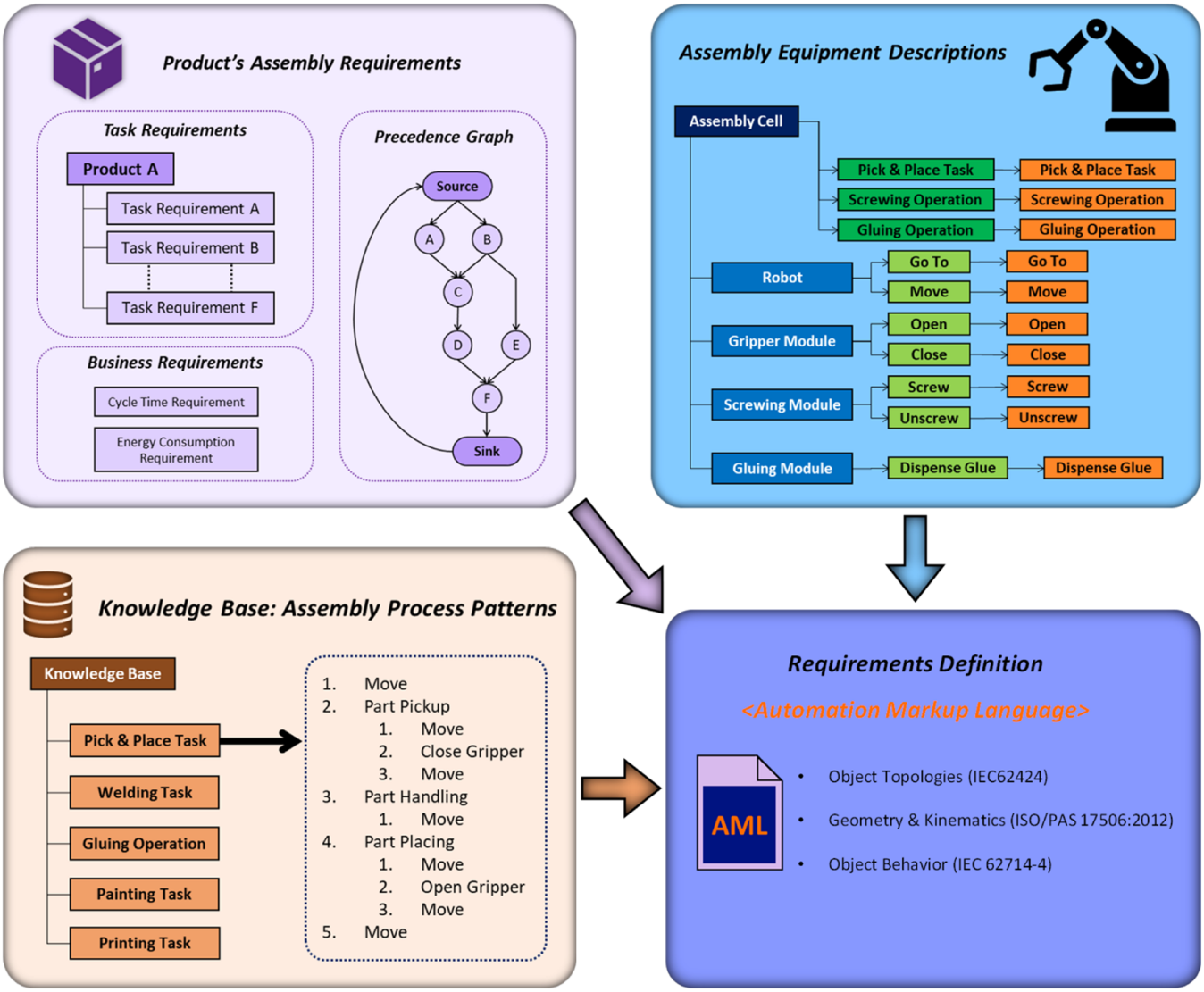


Fig. 6. Overview of Requirements Definition

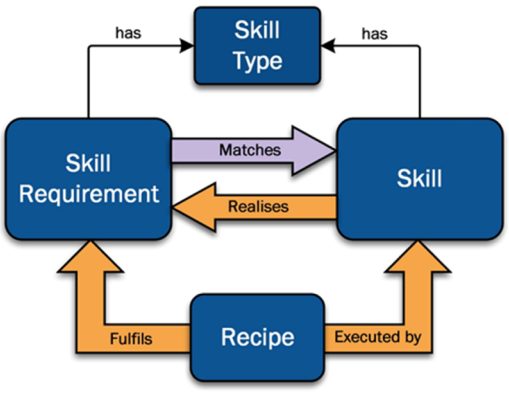


Fig. 7. Overview of the key definitions included in Skill-based Concept

definitions of the various modelling concepts. The Interface Class (IC) library helps to establish the various interface ports that is used to create links between various PPR objects. The Attribute Class (AC) library helps to define the various attributes of the modelling concepts. In essence it helps to define the characteristics and properties of the various model elements. Using the above mentioned three libraries, project or user specific reusable modelling elements can be established as generic classes in the System Unit Class (SUC) library. Finally, the specialized (project specific) instantiations of objects were created in the Instance Hierarchy (IH) and the relationship between them are established by linking appropriate ports.

3.1. Product description

The requirements definition for product domain includes two important elements: "parts" and "skill requirements" of the assembly product. The product definition may include one or more "sub-assemblies" or "parts" that needs to be put together. The other important element of the product description is the "skill requirements". The "skill requirements" definition includes few connectors such as: requirement connector, PPR connector, and precedence connector. All these connectors are defined as "ports" and can establish links with other "ports" that are of the same type. The requirement connector is used to establish the relationships between the "parts" that are required to fulfil specific "skill requirements". The PPR connector is used to express the links between the "skill recipes" that can fulfil specific "skill requirements". Lastly, the precedence connector is used to convey the sequence in which the "skill requirements" can be fulfilled. These links are particularly useful to generate the necessary conditions to achieve APG. The product domain includes some additional considerations that are primarily focused on enhancing the profitability of the business. For instance, including certain criteria such as: production rate, quality standards, material cost, and energy consumption can help to boost the operational efficiency of the business. The primary idea of gathering more information is to ease the process of configuring the desired system. It is also important to note that some of these considerations are strictly to be satisfied and some used to evaluate and improve the performance of the system.

3.2. Equipment description

The description of the assembly equipment includes details related to its physical configuration setup and the "skills" they offer. The "skills" can either be "atomic" or "composite". An "atomic skill" represents a standalone skill, while a "composite skill" comprises one or more "skill requirements" that can be fulfilled by executing the appropriate "atomic skills." This demonstrates the modular nature of the equipment modules while easing the effort to define complex systems. It is important to understand that the specifications for both "atomic skills" and "composite skills" are provided by different actors. The details regarding "atomic skills" are provided by equipment suppliers, and details regarding "composite skills" are provided by experts. This is because experts gain additional knowledge from their previous experience of using equipment modules in different combinations. Both "atomic" and "composite skill" definitions include "control ports", "parameter ports",

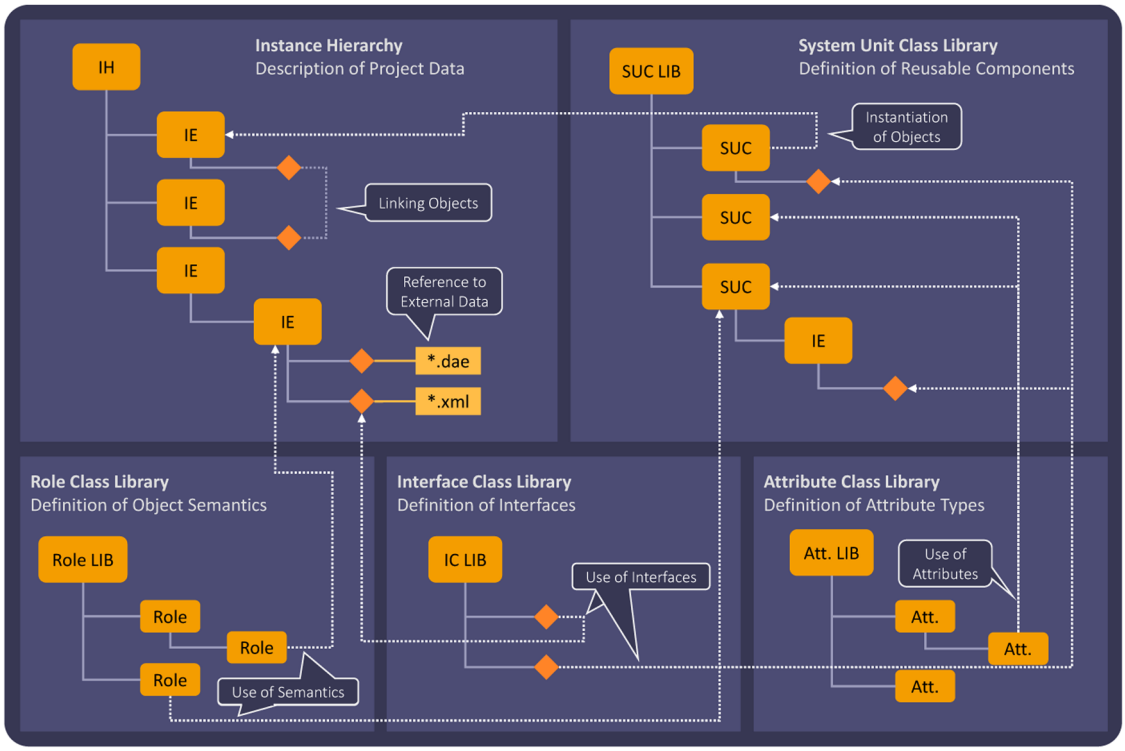


Fig. 8. Overview of the AML Class Libraries and its uses

and “information ports”. The “control ports” are employed to manage the “skill” execution with the help of “Input”, “output” and “error” ports. The “parameter ports” are exercised to pass the required parameters before and after the skill execution using “input” and “output” ports. The “information ports” are used to pass certain performance related information before and after the skill execution using “input” and “output” ports. This information can play a key role in identifying the optimal choice of “skill recipes” to support APG.

The definition of the “composite skill” includes two or more “skill requirements” that needs to be fulfilled in a specific sequence. The sequential requirements are expressed using the precedence port connections in a similar fashion as of the product requirements. In AML, the “skills” are listed in the same hierarchical level as of their representative equipment or combinations (sub-systems). The object-oriented style of representation in AML is handy to represent the library of equipment as generic classes that are grouped together. Particularly, a robot can be a generic class and can be instantiated multiple times to reflect its actual physical setup in various cells or workstations of the production facility. Similarly, the ‘skills’ are also presented as classes in AML, as its implementations can have various targets to fulfil.

3.3. Process description

It is crucial to recognize that it is a common practice to define the assembly task of a product as a high-level requirement. However, determining suitable processes to fulfil such requirements often requires a combination of human cognition and domain expertise. This is where the process domain plays a vital role. The process domain serves as a knowledge base that captures years of operational knowledge in the form of "process patterns." These process patterns act as a bridge between the higher-level product requirements and the lower-level equipment "skills." By utilizing these process patterns, it becomes possible to determine the specific compositions of lower-level equipment skills needed to fulfil the higher-level product’s skill requirements. Both the "Skill" and "Skill Requirement" definitions share a common attribute known as the "Skill Type" as shown in *Fig. 7*. This attribute plays a crucial role in establishing the mechanism for automated matching between skill requirements and skills. It enables the system to automatically identify and match the appropriate skills to fulfil the specific requirements of the product.

R3M defines assembly processes as the parameterized version of the equipment “skills” that can suitably fulfil the product’s requirements. The AML’s instance hierarchy includes all the parameterized version of equipment “skills” as “skill recipes”. The relationship between the “skill” and its “recipe” is realized as a class and an instance. The AML’s object header provides a reference base path which can be used to trace the representative ‘skill’ and its various ‘recipe’ instantiations. The “PPR connector port” is used to establish links between the “skill recipes” and the “skill requirements” that they can fulfil. Also, the “PPR connector port” is used to establish the links between the “skill recipes” and the corresponding “equipment” that executes them. These port connections together with the precedence links provide a good idea about the appropriate choice of recipes and the sequence in which they should be used. This information is used to generate appropriate functions to check the choice of recipes selected to fulfil the product’s requirements. In summary, one of the main objectives of requirements definition is to facilitate R3M’s vision to reduce human effort for reprogramming the system.

4. Automated program generation

The R3M project advances the foundational semantic models established by the openMOS project [24], with an enhanced focus on robotic systems. Central to R3M’s vision is the strategic reduction of engineer dependency in programming robotic cells, while still acknowledging the critical role of human expertise. To realize this goal, R3M introduces an innovative Automatic Program Generation (APG) process. This process adeptly combines minimal human intervention with advanced semantic technologies to efficiently generate executable programs for robotic operations. The effectiveness of this approach is encapsulated in a well-defined, three-stage methodology, detailed in the subsequent sections, and visually represented in *Fig. 9*.

4.1. Stage 1: recipe generation

In the initial phase of the Automatic Program Generation (APG), the R3M project undertakes the complex task of aligning the product’s high-level ’skill requirements’ with the system’s more granular capabilities, referred to as ’Skills’. This stage involves a nuanced analysis of the product specifications, which are often defined at a macro level, and their translation into specific, atomic-level skills that the equipment can execute. Recognizing the inherent challenge in this direct mapping, due to the differing granularity levels of product requirements and equipment capabilities, R3M incorporates the expertise of domain specialists. These specialists play a crucial role in developing and refining templates or composite skills that effectively bridge the gap between high-level product needs and the detailed capabilities of the equipment. While R3M’s approach is geared towards minimizing manual intervention, this human input is essential for ensuring that the generated mappings are both accurate and practically feasible. The involvement of experts aids in creating a more robust and context-sensitive mapping process, which is critical for the successful translation of broad product requirements into actionable, equipment-specific skills.

Central to this stage is the use of semantic parameterization to generate a wide array of potential skill configurations, referred to as ’skill recipes’. By leveraging the semantic attributes of skills, R3M can create multiple, varied configurations that represent different ways to meet the product’s requirements. This approach avoids the need for an exhaustive enumeration of all possible configurations, instead focusing on generating a broad and diverse set of potential solutions. These skill recipes are crafted to be both feasible and varied, providing a rich solution space for subsequent optimization.

The goal at this stage is not just to map skills to requirements, but to do so in a way that opens up a landscape of possibilities. By generating a substantial but manageable array of skill recipes, R3M lays the groundwork for the next phase of the APG process.

4.2. Stage 2: reinforcement learning

R3M’s RL Training Setup integrates sophisticated software and hardware components to facilitate robust and effective training for the RL agent. The RL agent operates within MATLAB’s Simulink platform. Gazebo emulates various actions related to the robotic manufacturing cell, ensuring a dynamic training environment. ROS2 enables seamless interaction between hardware and software. A publisher-subscriber mechanism transmits the RL agent’s actions as a message with unique ID. Parameter settings undergo rigorous risk and safety assessments, ensuring secure and efficient execution. A custom wrapper function triggers ROS2 actions based on the chosen recipe ID, enhancing the training process’s efficiency and accuracy. The R3M wrapper function serves as an intermediary, providing essential information to the MATLAB side for the RL agent’s input and receiving live feedback and outcomes as output responses as shown in *Fig. 10*.

For an effective RL training process, defining three primary inputs to the RL agent is imperative. Firstly, the collective signals representing the environment or system state provide comprehensive data for analysis and decision-making. The system state in R3M involves a meticulous combination of equipment, product, and process states, observed, and recorded during each skill execution. Leveraging various signals, such as joint angles and sensor feedback, allows R3M to comprehensively represent the system state, enabling the RL agent to autonomously detect and differentiate various states.

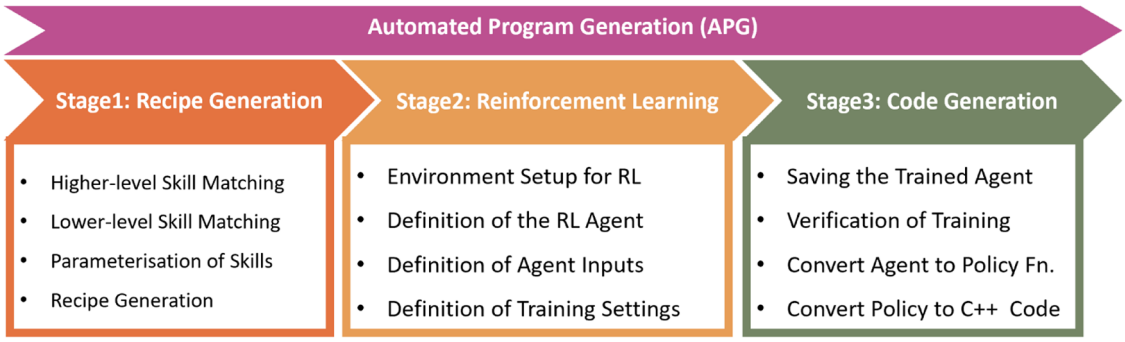


Fig. 9. Overview of the three-stage approach of achieving APG

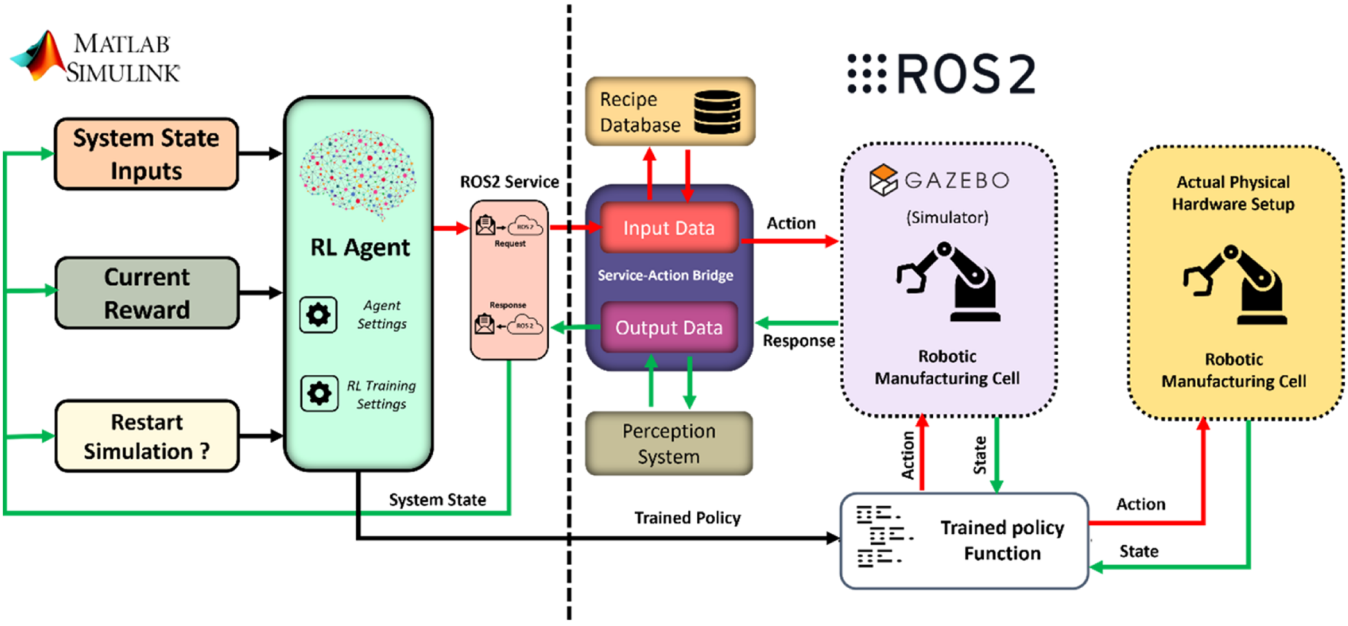


Fig. 10. Overview of RL Setup and ROS2-based Skill Execution

Secondly, the reward function evaluates the agent's actions and their impact on the environment, reinforcing desired behaviors and penalizing undesirable actions. A generic reward function by R3M facilitates a comprehensive evaluation of the agent's actions, ensuring adherence to the correct process sequence and optimizing task execution. Shaped rewards and penalties incentivize efficient task completion within the shortest timeframe. Additionally, considering the impact of the agent's actions and the duration of the skill execution further reinforces effective task completion strategies. An illustration of such a generic reward function is briefly explained in the text below.

$$R_{(s,a)} = \left[\alpha - \beta t_{(a)} + \gamma P_{(s,a)} - \delta U_{(s)}\right] \quad \text{(i)}$$

where:

- $R_{(s,a)}$: Reward received by the RL agent for taking an action 'a' in state 's'.
- α : Positive constant representing the base reward for completing each action successfully.
- β : Positive constant representing the penalty coefficient for execution time.
- $t_{(a)}$: Execution time of action 'a'.
- γ : Positive constant representing the reward weight for following the correct sequence.
- $P_{(s,a)}$: Function that determines the reward based on the action's adherence to the pre-defined assembly sequence in state 's'. This function can be:
  - 1: If the action follows the correct sequence in state 's'.
  - 0: If the action deviates from the correct sequence in state 's'.
- δ : Positive constant representing the penalty weight for unrecoverable states.
- $U_{(s)}$: Function that determines a penalty for reaching an unrecoverable state 's'. This function can be:
  - 1: If the state represents an unrecoverable situation.
  - 0: Otherwise.

This reward function balances three key factors:

- Action efficiency: The base reward (α) is received for completing any action. However, a penalty proportional to the execution time ($\beta t_{(a)}$) is applied to encourage faster completion.
- Sequence adherence: The agent receives an additional reward ($\gamma P_{(s,a)}$) for following the pre-defined assembly sequence. This ensures the actions are taken in the correct order.
- Unrecoverable state avoidance: A significant penalty ($\delta U_{(s)}$) is applied if the agent reaches an unrecoverable state. This discourages the agent from exploring actions leading to dead ends.

It is important to note that this is an illustration and the specific

functions for $P_{(s,a)}$ and $U_{(s)}$ might need adjustments based on specific assembly tasks and how unrecoverable states are defined in that system.

Lastly, the signal used to restart a training episode maintains the process's integrity and effectiveness. The episode reset function ensures the stability and integrity of the RL training process. R3M identifies two key conditions for restarting the training episode: the achievement of the assembly task's goal state and the identification of a system state from which the system cannot recover without human intervention. Monitoring the output and feedback from the execution of "passive skills" ensures appropriate intervals for resetting the training episodes, refining the RL agent's decision-making capabilities continually.

Moreover, R3M extracts essential parameters for the RL training settings from the process domain, facilitating informed decision-making regarding the termination of the RL training at an optimal time. Considering parameters such as the average number of actions required for task completion and the average reward necessary for successful task execution, R3M ensures the training process culminates at the most opportune moment, based on predefined criteria and constraints. This approach guarantees that the RL agent's training remains focused and efficient, mastering essential skills and executing complex manufacturing tasks proficiently.

4.3. Stage 3: code generation

The third and final stage focuses on translating the RL agent's trained policy into executable code tailored for specific product types and equipment configurations. This code serves as the operational blueprint for the robots, transforming learned behaviors into actionable instructions, thereby enabling autonomous task execution within the manufacturing cell. The generated code is specific to the trained setup and product types, and introducing new products or significantly modifying the cell will necessitate repeating the APG process, underscoring the importance of continuously evolving manufacturing demands.

In this stage, R3M leverages MATLAB's code generation feature to convert the trained RL agent's policy into an executable program. MATLAB can translate the agent's decision-making logic into efficient code, potentially in C/C++, that directly controls the robots. This approach offers several advantages:

- Efficiency: MATLAB's generated code can be highly optimized for real-time execution on robotic controllers.
- Stand-alone execution: The generated code can run independently without requiring MATLAB, reducing software dependencies.
- Integration with robotics libraries: The code can be seamlessly integrated with existing robotics libraries and frameworks, simplifying robot control.

By utilizing MATLAB's code generation capabilities, R3M ensures the smooth transition of learned behaviors from simulation to real-world robotic tasks, thereby enhancing the system's adaptability and operational efficiency in diverse manufacturing scenarios.

5. Perception systems and computational models for sensor fusion

The perception system can be an important foundation for many functions, such as task planning and scheduling, error handling and recovery, and safety mechanisms. In the context of reconfigurability, the robot functions need to be efficiently reconfigured and optimized in response to varying environmental conditions and task requirements, which relies on the real-time dynamic adaptability of the perception system. Compared with conventional robot cell, the reconfigurable robots place additional emphasis on the following requirements for the perception system:

- High-Resolution Sensing: To accurately recognize different components or tools required for various tasks, the perception system is expected to have high-resolution sensors to ensure the sufficient fine details can be captured.
- Real-time Performance: The ability to timely process sensor observation can be essential for not only the robot manipulating, but also the quick switching between tasks, ensuring minimum downtime during reconfiguration.
- Adaptive Algorithms: The perception algorithm should be capable of learning and adjusting to different tasks, enabling the robot to recognize new components required by different tasks, which may vary in color, size, texture, and geometry.
- Robustness to Environmental Changes: The perception system should be robust to changes in environmental factors along with the changing working content, such as lighting conditions, reflections, and occlusions.

In terms of sensing modality of perception systems, vision-based perception serves as a critical technology in reconfigurable robot. This is because, firstly, vision can be highly adaptable and flexible, and can provide sufficient details for many different manufacturing processes as well as their surroundings. Unlike conventional robot cell that relies on structured settings, vision-based perception has the potential to be effective in less structured settings, making it ideal for the reconfigurable robot dealing with various configurations and environments [25]. Secondly, thanks to recent developments in deep learning, the capabilities of vision data processing algorithms have been significantly improved in terms of real-time performance, accuracy, and adaptability, providing a crucial foundation for reconfigurable robot [26]. These advanced algorithms can be seamlessly integrated into vision-based perception systems, significantly enhancing their capabilities.

5.1. Perception system overview

The overall perception system can be illustrated by *Fig. 11*. The inputs of this system are 2 cameras fixed to the robot cell monitoring different axis of the working area and 1 camera mounted on the end effector of robot, and the output of this system are 6 Degree of Freedom (6DoF) and trajectory of the object of interest. The recognition from 2 fixed cameras provides an initial estimation based on the confidence level of different cameras to guide the robot to reach the object/aera of interest and the camera on the end effector will receive increased detail as it approaches the object, which can improve the precision of the initial estimation.

In addition, this system consumes the product design (CAD file) as one of its inputs, serving as the prior knowledge on the new object to be manipulated. The adaptive models can then identify and track the real object in the robot cell and estimate its 6DoF pose based on observations and the CAD file which provides the geometry and texture (optional) information of the object. For the conventional deep learning methods, models normally need to be retrained or fine-tuned by either domain adaption [27] or domain randomization [28] once the robot is reconfigured, which may take several hours or even days [29], making it difficult to be evaluated for different working content. In term of the adaptive models used in the R3M architecture, as the models were trained to identify unseen object purely based on the CADs, it can be easily adapted to new objects in which their CADs were available, enhancing the reconfigurability of robot significantly. As shown in *Fig. 11*, the adaptive model in R3M settings can be divided into 3 parts: object detection, object tracking, and pose estimation (6DoF). The function of detection model is to identify the aera of interest (the rough bounding box of the object) which will be used as the inputs of pose estimation and tracking model. The details of the adaptive models will be described in the following section, (Fig. 12).

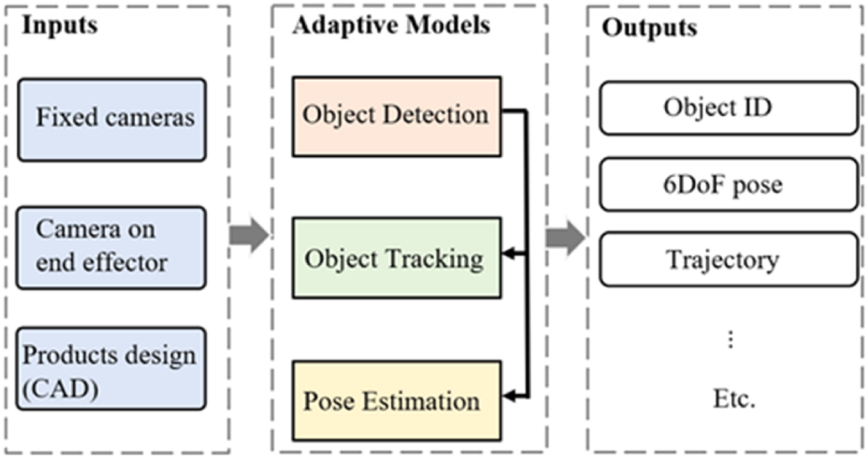


Fig. 11. Perception System Overview

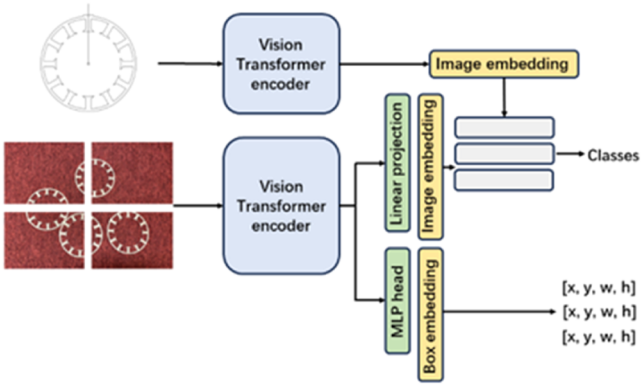


Fig. 12. OWL-ViT architecture

5.2. Adaptive vision-based perception

5.2.1. Novel object detection

Novel object detection can be the backbone of the downstream functions of perception system, namely 6DoF pose estimation and tracking. The detector can provide a rough bounding box of an unseen object given its CAD, enabling the object tracker and pose estimator to focus on the region near the object of interest, thus reducing the computational complexity and burden for them.

In R3M settings, as the fast reconfigurability can be the key feature, the time-consuming retraining process should be avoided when dealing with new workpiece. In literature, zero-shot [30] and one-shot detection [31] can be the prevailing solutions. While the former requires no appearance information of the object of interest, and the latter requires one or a few pieces of appearance information of the object. In manufacturing setups, it can be impractical for operators to accurately describe different workpieces, hence the one-shot detection was used here, as the CADs of the objects can be available in most cases. Therefore, Vision Transformer for Open-World Localization (OWL-ViT) [32] was used as the novel object detector in its one-shot detection mode. In this model, a standard Vision Transformer (ViT) was used as an encoder to embed the scene of robot cell being observed, and then the prototypical picture of novel object rendered from its CAD can be encoded by a separate ViT as query before being fed to a MLP module to estimate the regions of object. The full details can be found in [32].

5.2.2. Novel object 6DoF estimation

The unseen object poses estimation model used in the R3M architecture was developed based on MegaPose proposed in [29]. As shown in *Fig. 13*, MegaPose consists of a coarse estimator and a refiner. Given a rough bounding box of an object by the object detection model mentioned previously, the coarse estimator will first render 104 images from different viewpoints (evenly distributed) based on the CAD of object, and then perform a classification to identify the closest to the observed object pose among the rendered object pose images as a coarse pose estimation. Then, the coarse estimation can be passed to the refiner to fine tune the estimated pose.

As shown in *Fig. 13*, the refiner takes the rendering of coarse pose estimation and 3 perturbed pose renderings ensuring that the center points of rendered images are the anchor point O of the object 3D model as its prompts, providing the refiner the reference of the original point of rotation. Given the observation image, the refiner outputs the update of the pose (translation and rotation) compared with the coarse estimation. The regressed pose update consists of: (ii) 1 vector of size 3 describes the 3D translation in camera coordinate system, (iii) 2 vectors of size 3 describe the rotation originated from the anchor point of object:

$$e_1' = N(e_1) \tag{ii}$$

$$e_2' = N\left(e_2 - \left(e_1' \cdot e_2\right) e_1'\right) \tag{iii}$$

$$e_3' = e_1' \times e_2' \tag{iv}$$

where N (*) represents the normalization and the SO(3) rotation matrix can be recovered by $R(e'_1, e'_2, e'_3)$. According to [33], it has been demonstrated that this representation of rotation outperforms quaternions when used in conjunction with a neural network for regression due to its continuity.

The refiner can be operated recurrently by replacing the coarse estimation by the output pose of refiner, fine tuning the estimated poses. In this work, the iteration number of 5 was employed. The computational models of coarse estimator and refiner were ResNet34 [34]. As shown in *Fig. 14*, while the feature representation of ResNet34 was employed by a classifier in coarse estimator, the feature representation of a separate ResNet34 was used by a single layer neural network to perform regression task in refiner.

5.2.3. Novel object 6Dof TRACKING

6DoF tracking refers to continuous real-time tracking of object poses in video stream. In this work, the 6DoF tracking was achieved by reorganizing the MegaPose. The 6DoF estimate obtained in the previous section can be used as an initial pose for the first frame, and then the results can be updated using the refiner with one iteration on each of the next frames in the video. This is because the magnitude of the change in the object pose in any two adjacent frames lies within the capability of the refiner.

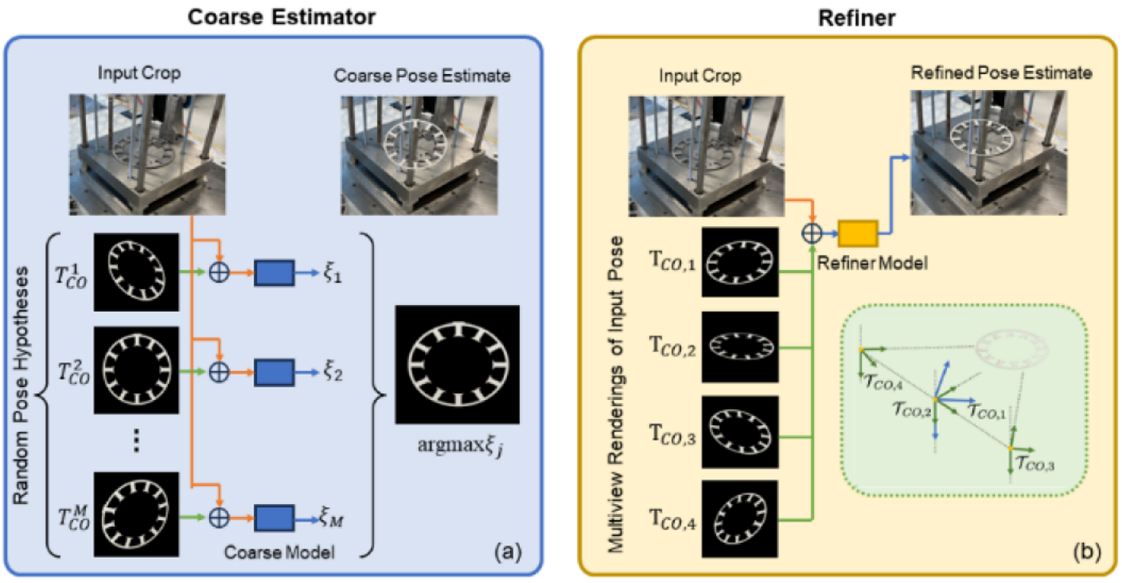


Fig. 13. MegaPose Architecture

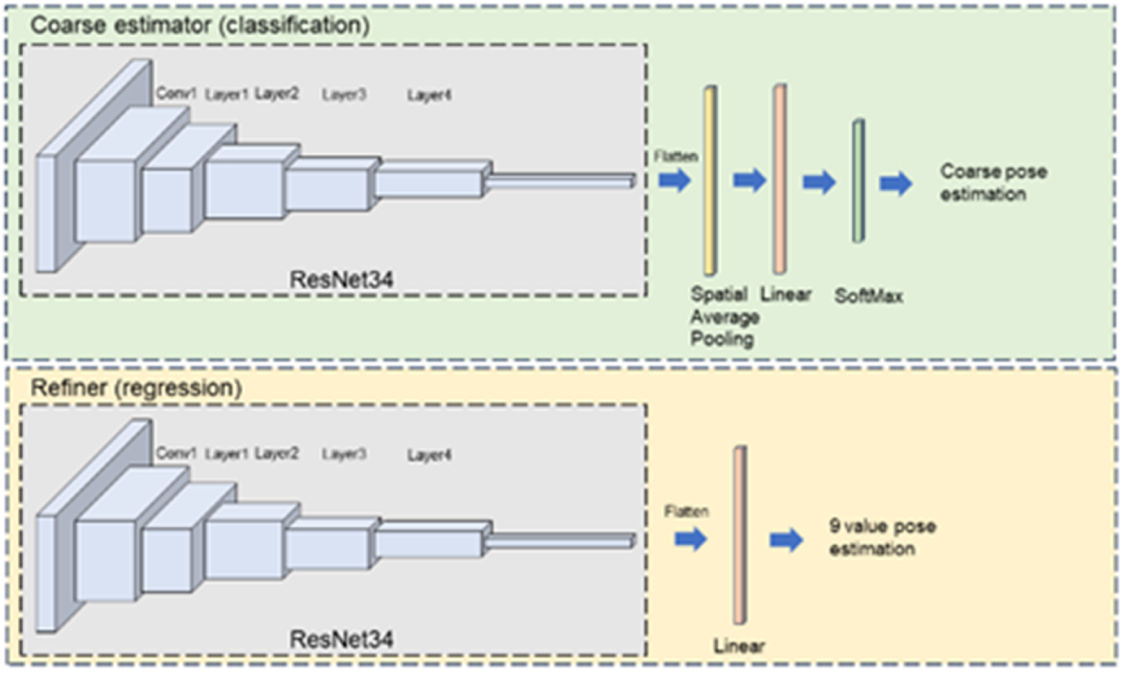


Fig. 14. Computational Models of MegaPose

6. R3M use-case: reconfigurable and responsive robot cell demonstration

6.1. Use-case description

To test the R3M architecture on an actual manufacturing process, an initial industrial use case was considered. This first use case consisted of robotic stacking of laminated cores for electrical machines. Here, we describe this process before going on to discuss the requirements and challenges it poses for the R3M system.

Laminated rotor and stator cores are an integral part of many electrical machines from consumer appliances to automotive, aerospace, and power generation. Such cores are formed of stacks of thin sheets of electrical steel, which shorten circulating paths of eddy currents induced in the steel by the changing magnetic field and therefore reduce the associated resistive losses. A laminated core within thinner lamina (layers) therefore increases the efficiency of the resulting machine but at an increased complexity and manufacturing cost.

A typical lamination stack consists of several hundred lamina between 0.1 mm to 0.5 mm thick, which are then held together by welding, mechanical fastening, or adhesive bonding. Each of these methods of joining the stack carries potential drawbacks: welding short-circuits the otherwise electrically isolated lamina, reducing the effectiveness of lamination; mechanical fastening can create local stresses which induces losses in the material, and allows delamination in the unconstrained areas; whilst adhesive can be problematic to ensure a uniform bond throughout the stack, particularly for thin lamina.

A key challenge with consolidating lamination stacks is in firstly

aligning the lamina. This is typically achieved with hard tooling that introduces an overhead cost, which can make short production runs or prototyping prohibitively expensive. The aim of this use case is to use a serial 6-axis robotic manipulator to align and stack the thin sheets of steel into a fixture and they will then go on to be pressed and bonded to form high quality laminated stator cores.

6.2. Use-case requirements and challenges

The robotic lamination stacking use case poses some interesting challenges to the R3M system. On the surface, it is essentially a simple, single robot pick and place application. However, there are some specific details that make things challenging.

The sheets are thin (approx. 0.15 mm) and therefore flexible. The design of the end effector should ensure sufficient contact points and suction of the sheet to enable a firm grip and to keep it flat as the sheet is moved into the fixture. A fixed camera system is employed to detect the notch in the lamination sheet. This information is crucial for orienting the end effector to accurately grasp and align the lamination sheet according to the notch, as depicted in Fig. 15. The coordinates of the objects, obtained in the camera's coordinate system, are transformed into the robot's coordinate system to perform this task accurately.

The real robot programs and the digital twin should be cognizant of the potential bending within the sheet and the toolpaths should be programmed to ensure the sheet is laid flat onto the stack of sheets in the fixture. The process initiates with the acquisition of position and orientation information of the lamination sheet through the detection system. The gripper is then controlled to secure the lamination sheet, ensuring its flatness throughout the movement. Finally, the lamination sheet is positioned in the designated stacking location, leveraging the alignment mechanism to guarantee precise placement of each sheet.

Another complication for any camera systems was that all the parts (sheet, fixture, and floor of the RECON cell) are all of a similar grey/metallic color. Hence there is little contrast between the sheet and its background. To alleviate this issue, a thin black sheet was placed over the bottom of the fixture where the sheets were placed, which provided sufficient contrast for the camera to detect the sheet.

This is an initial demonstration of robotic lamination stacking and so the process was intentionally kept uncomplicated. The sheets were symmetrical and uniform across the whole stack and a custom fixture to place the sheets in and align the notch to the desired orientation was already provided (see *Fig. 15*).

A specific, custom end effector for the smaller ABB IRB-1200 robot was designed for this particular sheet size. It used pneumatic suction cups to pick up the sheet and a camera was mounted to the palm to determine the orientation of the sheet. The end effector had to navigate around the various posts of the mounting fixture as shown in *Fig. 16*, whilst aligning the sheet to the back plate of the fixture.

The future objectives for this system include demonstrating robotic stacking for sheets of varying sizes and geometries using a flexible end effector without the need for bespoke tooling and fixtures. This can be achieved by implementing an algorithm that coordinates the end-effector movement with a camera system to verify the accurate placement of the lamination sheet. For instance, a camera mounted on the end effector can facilitate precise alignment and positioning, ensuring the system's adaptability and efficiency in handling diverse sheet configurations.

6.3. Use-case requirements definition

The focus of this section is to ensure that all the essential elements outlined in the requirements definition are appropriately incorporated in AML that describes the use case. *Fig. 17* shows a specific product used in this use case comprising three sheets, a base plate, and three "skill requirements". In AML, every element's parent information is displayed as "class", and its hierarchy or type is displayed as role. Here, the base plate and the sheets roles are recognized as "parts", and the "skill requirements" role defines "pick and place" as its "skill type". All these "skill requirements" and "parts" includes all the port connectors outlined in the requirements definition *section 3.1*. Similarly, all their appropriate internal were also established. However, the primary focus is on the information that supports APG. Under these grounds, the critical elements are the "skill requirements", its precedence links, and its "skill type" definitions. This information typically supports the first stage of the APG to match the product's "skill requirements" with the equipment "skills". The precedence links are used in the second stage of the APG to check the agent's action sequence and reward appropriately. Additionally, the cycle time business metric is used to reward the agent for its choice of recipe selection for every single episode of the RL training. This is checked by comparing the value of the accumulated durations of skill executions with the input value of the cycle time.

Moving on to the equipment domain, the key information that supports APG is the description of "skills". *Fig. 18* presents the two subsystems and the corresponding "skills" defined in this use case. The first sub-system is the combination of the robot and gripper modules. The second one is the vision system that includes three individual cameras. It is important to understand that in AML, the skills are only defined as classes and therefore, they can only be found under appropriate SUC libraries. This implies that the instance hierarchy of the AML, will only include the equipment instantiations and all their corresponding "skills" and "process patterns" can only be found under their parent SUC libraries. Another important aspect to note here is the "pick and place"



Fig. 15. R3M LamStack Use-Case. [left] Single sheet for the robotic laminated stack application. The sheet is symmetric except for a small flat notch on the outer edge used for alignment. The sheet is flexible, so when held freely, the sheet can be seen to bend in the image. [right] Part of the mounting fixture for the sheet stacking application. The flat notch on the sheet must be aligned to the vertical back plate of the fixture.

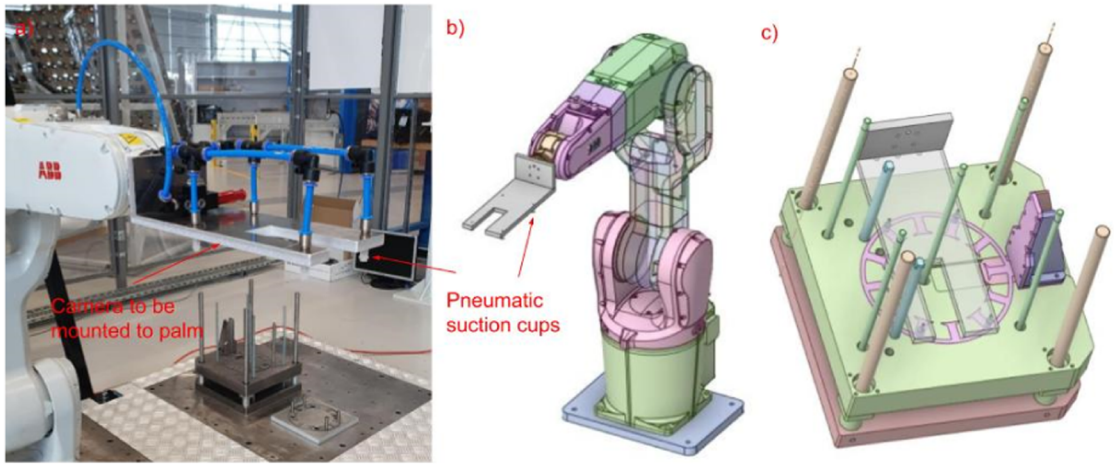


Fig. 16. (a) Image of the designed end effector on an ABB IRB 1200 robot with pneumatic suction cups for the lamination stacking application, (b) CAD model of the end effector and (c) CAD model showing the end effector passing over the fixture to place the sheet.

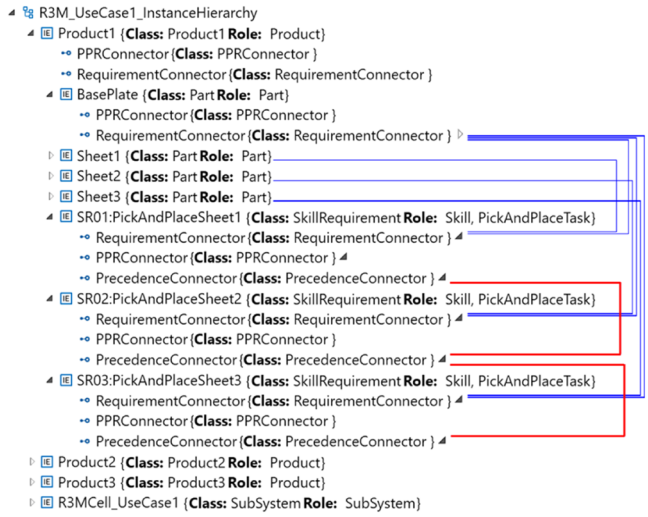


Fig. 17. Use-Case's Product Definition in AML

process pattern. This "pick and place" pattern is an expert human input recognized under the combination of robot, gripper, and camera modules. As mentioned in the earlier *Equipment Description* section 3.2, the process pattern is only a skeleton that allows matching between the higher-level product's requirements and the lower-level equipment "skills". The "pick and place" pattern includes a few "skill requirements" that have precedence links established between them. These "skill requirements" has matching lower-level equipment "skills" listed under each individual equipment modules. However, the links between them does not exist or it is not expected to be established as a part of requirements definition. The process of establishing these links is automated in the first stage of the APG with the help of the common attribute "skill type". This supports the R3M's vision to minimize human effort for reprogramming the system.

6.4. Program generation for use-case

In the first stage of the APG, the primary focus is to match the use case specific product's "skill requirements" with the "skills" of the equipment contained in the cell. The "pick and place" process pattern is used as a bridge to match the higher-level Product requirements and the lower-level equipment "skills". The core objective of this stage is to generate recipes that can fulfil all the product's requirements. Here, it is important to note that the process patterns are expert's inputs composed of "skill requirements", and no "skill recipes" will be generated for them. The overall objective of the APG is to covert the selected process patterns to executable programs that fulfils the product's requirements. Here, in this use case, all the "skill requirements" of the product and "pick and place" pattern are matched with the equipment "skills" using their inherent "skill type" definitions. Once this matching is done the next step is to generate "recipes" for all the matched "skills". An instance of generating a recipe for a selected skill is explained in the upcoming text. In the *Fig. 19*, it is shown that the gripper module has got two skills: open and close. These skills are elementary in nature without any parameters, and the possibility is limited that only one recipe can be generated for each of them. On the other hand, if the robot's move skill is considered, it has several parameters including move type, speed, etc. Each of these parameters can have two or more parametric settings. Each unique combination of these multi-parameter settings of a "skill" is a "recipe". This as a result leads to a large number of recipe generation as an outcome of the first stage of the APG.

In the second stage of the APG, the focus is to use RL training to select the optimal "recipe" choices and sequence them to fulfil the product assembly. The number of "recipes" generated becomes the action space of the RL agent. During the RL training the agent selects a "recipe ID" as its action and the appropriate skill is executed as a ROS2 action. The action response is fed as the system state input to the RL agent. The action response includes all the details related to the equipment, product, and process states. The reward for the agent's action in calculated based on some critical information contained in both action response and requirements definition. The information from the requirements definition is used to check if the agent's actions are following the correct sequence to fulfil the product assembly. The information from the action response is used to reward the agent's choice of action based on their execution time. Similarly, the information from the action response is also used to restart a training episode. After all these definitions the RL training was triggered and allowed to progress and reach its target average episode reward. The training automatically stops once this condition is reached, and then the trained agent is then saved. The trained agent is then tested to see if the recipe IDs of its actions are in the same order of the process pattern's "skill requirements". *Fig. 20* presents the progress of the RL training where the moving average of the episode reward converges to the expected value.

At the third stage of the APG, the trained agent is converted into a standalone executable C++ code. The generated code is exercised by providing state inputs and it takes the appropriate control actions until the product assembly is completed.

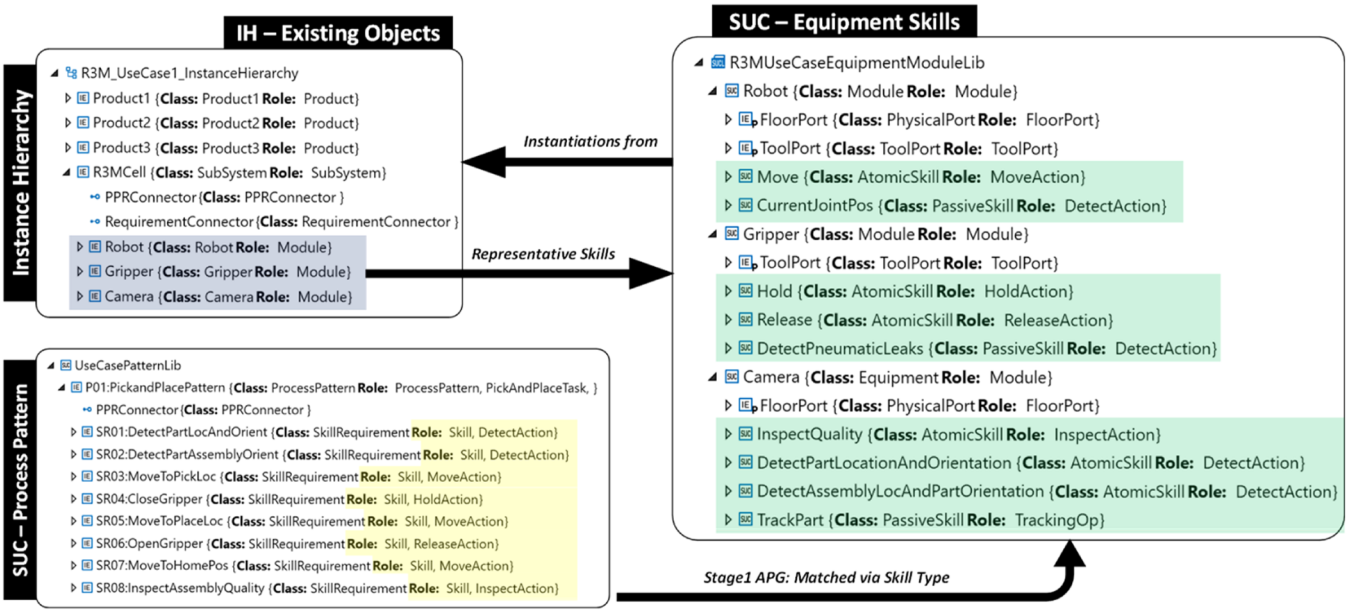


Fig. 18. Overview of Equipment, its Skills, and Process Pattern

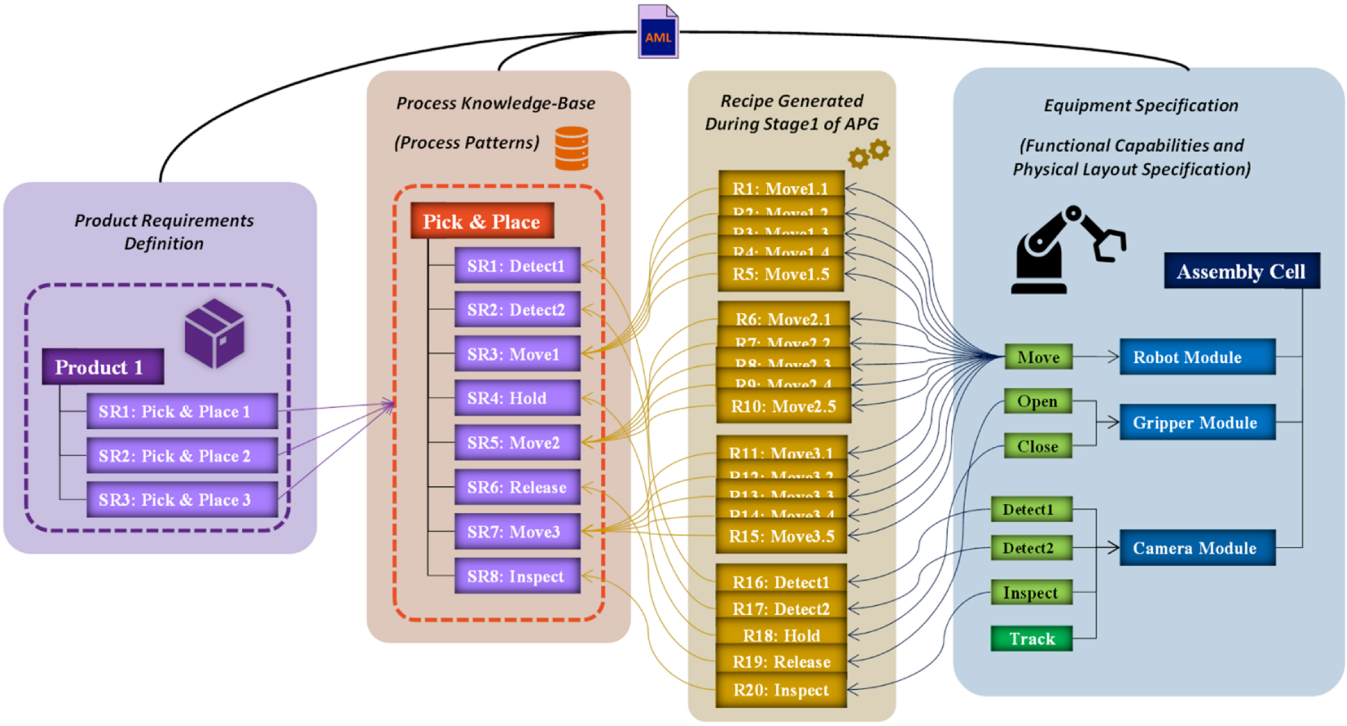


Fig. 19. Overview of the Use-Case specific Product Requirements and Equipment Skills

6.5. Results from the perception layer

The results depicted in the *Fig. 21* stem from the implementation of the perception system designed for R3M. As shown in *Fig. 21*, the left column indicates the original observation of cameras, the second column shows the results of zero-short detection, where the bounding box of the object of interests was given by referencing its CAD file. Then, the bounding box and the CAD file were sent to MegaPose to ascertain the 6DoF of workpieces. The resulting 6DoFs are presented in the last two columns of *Fig. 21*. The capability of referencing workpieces can be crucial as it facilitates the rapid modification of the robot cell's tasks in accordance with varying work requirements.

Generally, the perception system demonstrates commendable performance. It was extensively evaluated in the Gazebo simulation environment due to the ease of obtaining precise 6DoF poses. In the experiment, all objects were considered unknown to the models; the images and objects used in the evaluation did not appear in their training processes. Five CAD models of different objects were used, four from the publicly available dataset MVTec ITODD [35], and one was the lamination sheet from electrical motor production. For each item, 100 camera observations were generated and labeled with position (x, y, z) and orientation (quaternion) in the world coordinate system. The

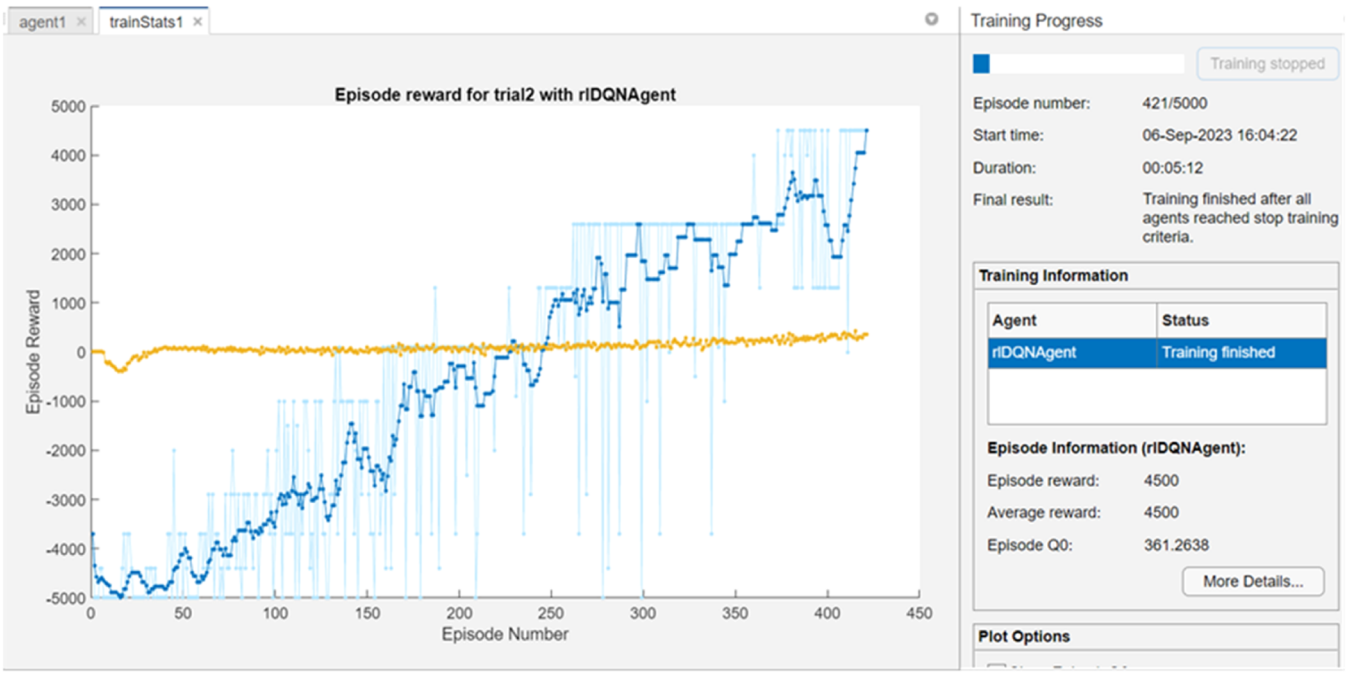


Fig. 20. Overview of the Progress of the RL Training Episodes

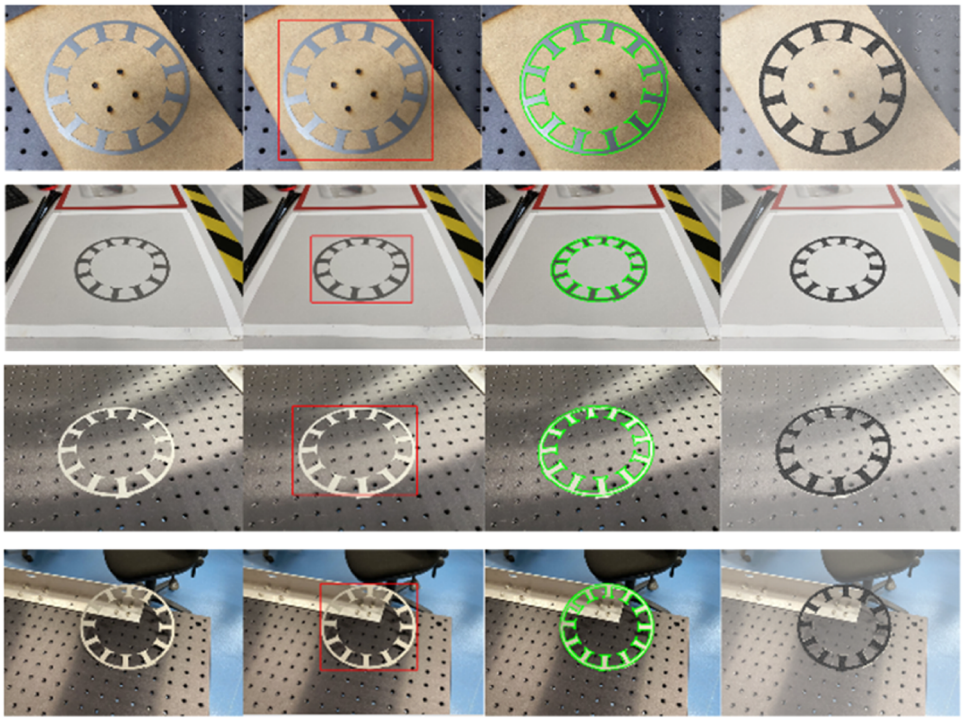


Fig. 21. (a) Camera observations, (b) Results of novel object detection, (c) Contour overlay rendered from estimated 6DoF pose, (d) Mesh overlay rendered from 6DoF pose.

evaluation covered the entire perception pipeline. Initially, the One-Shot detection pipeline was applied to each scene, resulting in a bounding box. Subsequently, based on the bounding boxes, the 6DoF pose was estimated using MegaPose.

The performance of the 6DoF pose estimation was assessed separately for position and orientation. For position evaluation, the Euclidean distance was calculated between the estimation and the ground truth. For the evaluation of rotation angle estimation, the difference between the estimated values and the ground truth was calculated using radians to provide an intuitive understanding. The statistics are presented in Tables 2 and 3.

As shown in Table 2, the position evaluation was divided into 3D (x, y, z) and 2D (x, y), revealing that the errors in 2D were significantly lower compared to the errors in 3D, suggesting imprecision along the z-axis (perpendicular to the frame). This discrepancy is attributed to the inherent uncertainty in depth estimation when using a monocular

Table 2
Positional error.

CAD name	3DoF		2DoF	
	Mean error (m)	Median error (m)	Mean error (m)	Median error (m)
adapter_plate_square	0.0153	0.0152	0.0039	0.0039
engine_part_cooler_round	0.0080	0.0060	0.0050	0.0048
injection_pump	0.0117	0.0093	0.0057	0.0051
Star	0.0211	0.0211	0.0100	0.0071
LamSheet	0.0155	0.0063	0.0076	0.0047

camera system. The mean and median orientation errors are presented in Table 3. It was observed that the yaw error for the lamination sheet (LamSheet) was significantly greater than the roll and pitch errors. Similarly, the roll error for the engine cooler round was notably larger. The challenge arises from the highly symmetric nature of these objects, which inherently lack distinctive features necessary for orientation determination, as illustrated in Fig. 22. This subtlety in the object's design leads to the system's inability to accurately detect its orientation, despite correctly determining the other parameters related to the 6DoF pose. This specific limitation highlights a scenario where the MegaPose system's efficacy is compromised due to minimal feature variation critical for orientation discernment.

7. Discussion

The R3M system, integrating Service-Oriented Architecture (SoA) principles, facilitates seamless information exchange and resource management. This fusion optimizes manufacturing processes by consolidating the Requirements, Program Generation, and Sensing packages into a unified dataflow system. Key modules within the architecture include those responsible for central cell control, resource management, scheduling, process analysis, soft PLC services, and database management, each playing a vital role in ensuring smooth operation and efficient utilization of resources. Automated risk assessment is a pivotal feature, aligning with international standards and reducing potential hazards in the production environment. Leveraging Unified Modeling Language (UML), the system systematically evaluates risks and implements measures for mitigation, ensuring both safety and productivity. The adoption of SoA enables the concept of Robot as a Service (RaaS), offering manufacturers flexibility and access to cutting-edge technologies. This approach aligns with Industry 4.0 principles, leveraging IoT and AI for enhanced efficiency and decision-making. Furthermore, the architecture's reliance on a common communication platform simplifies integration and enhances collaboration among various components, supporting real-time data collection and analysis, empowering manufacturers with valuable insights for continuous improvement. The integrated architecture provides a flexible, efficient, and safe framework for reconfigurable manufacturing, fostering seamless operations and enabling continuous optimization in industrial settings.

The first step in R3M approach is the use of AML for requirements formalization. This uses a multi discipline standardized data format for representing manufacturing systems, which is extended to incorporate the specificities of robotics. AML facilitates the integration of various components and systems within a robotic manufacturing environment, allowing for more efficient data exchange and process coordination. By employing AML, robotics can leverage a unified language for describing equipment, processes, and product information, which significantly enhances the adaptability and flexibility of manufacturing operations. Additionally, the use AML provides some expert-based structure and semantics, which helps with the reduction of the solution space. The R3M system's Automated Program Generation (APG) approach represents a paradigm shift in robotic cell programming, by leveraging semantic descriptions in combination with advanced Reinforcement Learning (RL) to generate and optimize the sequencing of robotic actions. This method aligns high-level product requirements with the detailed capabilities of equipment, requiring some initial human input for identifying emergent system capabilities. However, as the system's libraries expand, the need for human input is expected to diminish. The output from the APG process provides the system with the ability to dynamically adjust operations based on learned behavior, enhancing its real-time responsiveness in production scenarios. Furthermore, while the exploration and training of the RL agent mainly occurs in a virtual environment, real-world training cycles are also utilized to further refine the agent's capabilities. Once fully trained, the APG's output ensures the system's adaptability and efficiency across various production scenarios.

The perception system leveraging MegaPose 6DoF estimation showcases significant strengths, primarily its robustness in swiftly capturing the positional and orientational parameters essential for the reconfigurability of robotic cells. This adeptness allows for the rapid switching of tasks and adaptation to new workpieces, thereby enhancing the flexibility and efficiency of the production process. Nonetheless, the system's efficacy is somewhat diminished when discerning the minor features of symmetric objects, such as a minor notch that is pivotal for determining the correct orientation. In addition, some objects with special geometries can be difficult to distinguish in a particular viewpoint, which may make the perception system ineffective. Hence, an intelligent multi-camera fusion, which can harness the collective strengths of multiple viewpoints to form a more accurate and comprehensive understanding of an object's orientation is needed. It is expected to refine the precision of the perception system by integrating and

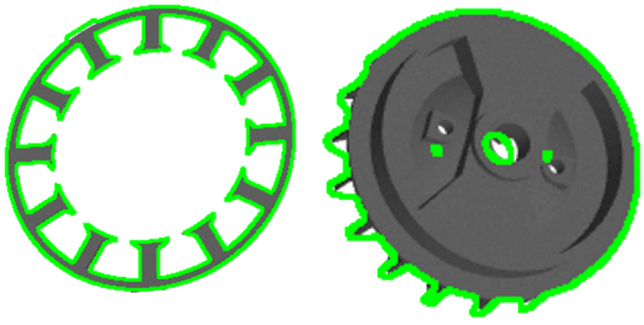


Fig. 22. Left: Yaw estimation failure for LamSheet. Right: Roll estimation failure for engine cooler round.

Table 3
Orientation error in roll, pitch and yaw.

CAD name	Mean (radian)			Median (radian)		
	Roll	Pitch	Yaw	Roll	Pitch	Yaw
LamSheet	0.2455	0.2349	1.6401	0.0425	0.0511	1.5773
adapter_plate_square	0.0631	0.0319	0.0443	0.0109	0.0168	0.0081
injection_pump	0.0508	0.0458	0.0242	0.0238	0.0266	0.0105
engine_cooler_round	0.4126	0.1055	0.1361	0.0373	0.0338	0.0234
Star	0.0570	0.0568	0.0138	0.0247	0.0212	0.0096

intelligently analyzing the data from various cameras, improving adaptability of robotic cells in industrial settings. This underscores an opportunity for improvement and constitutes an avenue for the future work of R3M.

8. Conclusion

R3M is pioneering an adaptive system designed to automatically respond to the dynamic landscape of product variations and demand fluctuations. This cutting-edge system aims to meticulously capture vital data pertaining to assembly products and the capabilities of equipment, ensuring a harmonious data flow and integration across different engineering spheres. The foundation of this initiative is the meticulous alignment of product, process, and equipment domains, leveraging the insights gained from the openMOS project's semantic models. The utilization of skill-based concepts such as "Skill," "Skill Requirements," and "Skill Recipe" plays a pivotal role in establishing a coherent relationship between these domains, thereby facilitating automated programming and enhancing efficiency.

The system intricately maps out product descriptions, including parts and requisite skills, and forges connections through various connectors like requirement, PPR, and precedence. These connectors are instrumental in setting the conditions necessary for Automated Programming Generation (APG), ensuring a seamless translation of product needs into executable tasks. On the equipment front, the description delves into the physical setup and the spectrum of skills required, distinguishing between atomic and composite skills with precision, thanks to expert input. This level of detail is critical for the successful implementation of APG. The process description aspect involves the delineation of assembly tasks as overarching requirements and the strategic use of process patterns to effectively bridge the gap between product needs and equipment capabilities, with semantic parameterization playing a key role in generating a variety of skill configurations, thereby laying a solid foundation for APG.

APG itself is a structured three-phase process, starting with Recipe Generation where product requirements are intricately mapped to the corresponding equipment skills, fine-tuned by expert insights to ensure utmost accuracy. This phase leverages semantic parameterization to create a multitude of skill recipes, opening up a range of solutions for optimization. The Reinforcement Learning (RL) stage then takes over, employing advanced software and hardware to train an RL agent capable of autonomous task execution. A well-crafted reward function in this stage ensures the reinforcement of desirable behaviors for efficient task completion, with training parameters meticulously extracted from the process domain to guarantee focused training. The final stage, Code Generation, translates the RL agent's trained policy into executable code, tailor-made for specific product and equipment setups, with ongoing updates and retraining essential to keep pace with manufacturing evolution.

The system's perception capabilities, bolstered by advanced computational models for sensor fusion, are critical in enabling robots to adeptly adjust to varied environmental conditions and task requirements. Key attributes of this system include high-resolution sensing for precise component recognition, real-time performance for minimal reconfiguration delays, adaptive algorithms for learning new task specifics, and robustness against environmental variability. These features ensure that the reconfigurable robots can maintain high efficiency and adaptability in the face of diverse manufacturing scenarios.

Integrating R3M's automated adaptability with the perception system's robustness crafts a formidable framework for contemporary manufacturing setups. This integration facilitates a seamless alignment of product, process, and equipment domains, streamlining operations and boosting efficiency. The adoption of AutomationML and deep learning technologies within R3M enables automated programming and agile decision-making, which are paramount for catering to the dynamic demands of manufacturing. The architecture's emphasis on risk assessment further ensures a safe operational environment while optimizing production workflows, signifying R3M as a beacon for the future of manufacturing where agility, efficiency, and safety converge to foster innovation and excellence.

Looking ahead, the paper outlines several areas ripe for future exploration to enhance the system's effectiveness and applicability. Optimizing the recipe generation process stands out as a promising avenue for further streamlining manufacturing operations by refining algorithms and enhancing the mapping between product needs and equipment skills, enabling more effective and tailored production strategies. Incorporating robot profile questions early in the requirements phase can ensure the system is cognizant of specific robot capabilities, leading to more accurate and effective programming. Automating the creation of AML models can expedite the integration of new manufacturing processes, hastening deployment and reducing manual interventions. Addressing instability issues in simulation models and optimizing the perception system for speed and efficiency are critical for improving the responsiveness and productivity of reconfigurable robots. Enhancing the perception system by incorporating multiple viewpoints for 6D pose estimation and extending it from single-object to multi-object scenarios aims to address current limitations related to depth estimation inaccuracies and orientation challenges with centrally symmetric objects. Real-world applications often involve multiple objects, which can lead to issues such as occlusion. By integrating multiple viewpoints, these uncertainties can be mitigated, improving the accuracy and robustness of the 6D pose estimation system in more complex environments. These advancements underscore the continuous evolution needed to meet the changing demands of modern manufacturing.

This research presents a comprehensive blueprint for an integrated robotic manufacturing architecture, highlighting key areas for future development to bolster the system's efficiency and relevance. The proposed enhancements, from refining the recipe generation process to automating AML model creation and improving simulation reliability, reflect the study's methodical and innovative approach to streamlining manufacturing operations. The focus on enhancing perception system performance further emphasizes the commitment to advancing the agility and responsiveness of reconfigurable robots, aiming to boost overall productivity. The inclusion of real-world application scenarios underscores the practicality and effectiveness of the proposed system, showcasing the embedded originality and thoroughness of the research methodology. By incorporating these advancements into the system's development trajectory, the architecture is poised to continue evolving, meeting the dynamic needs of modern manufacturing and reaffirming its value as a tool for optimizing industrial processes.

CRediT authorship contribution statement

Seemal Asif: Writing – review & editing, Writing – original draft, Supervision, Resources, Project administration, Methodology, Investigation, Funding acquisition, Conceptualization. **Mikel Bueno:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Data curation. **Pedro Ferreira:** Writing – review & editing, Resources, Investigation, Conceptualization. **Paul Anandan:** Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Conceptualization. **Ze Zhang:** Writing – original draft, Visualization, Validation, Software, Methodology, Formal analysis, Conceptualization. **Yue Yao:** Writing – original draft, Visualization, Validation, Software, Methodology, Conceptualization. **Gautham Ragunathan:** Writing – original draft, Methodology, Data curation, Conceptualization. **Lloyd Tinkler:** Writing – review & editing, Writing – original draft, Resources, Investigation, Conceptualization. **Masoud Sotoodeh-Bahraini:** Software. **Niels Lohse:** Supervision, Resources, Methodology, Funding acquisition, Conceptualization. **Phil Webb:** Resources, Project administration, Investigation. **Windo Hutabarat:** Supervision, Resources. **Ashutosh Tiwari:** Resources, Investigation.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Seemal Asif reports financial support was provided by Cranfield University. Seemal Asif, Mikel Bueno, Pedro Ferreira, Paul Anandan, Ze Zhang, Yue Yao, Gautham Ragunathan, Lloyd Tinkler, Masoud Sotoodeh-Bahraini, Niels Lohse, Phil Webb, Windo Hutabarat, Ashutosh Tiwari reports financial support was provided by Engineering and Physical Sciences Research Council. Seemal Asif reports a relationship with Cranfield University that includes: employment. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

This research is supported by EPSRC grant (EP/V051180/1) for the Reconfigurable Robotics for Responsive Manufacture - R3M.

References

[1] G. Kumar, K. Kumar Goyal, N.K. Batra, Evolution, principles and recent trends in reconfigurable manufacturing system, Journal of Physics: Conference Series 1240 (1) (2019) 012161, https://doi.org/10.1088/1742-6596/1240/1/012161.

[2] M. Bortolini, F.G. Galizia, C. Mora, Reconfigurable manufacturing systems: Literature review and research trend, J. Manuf. Syst. 49 (2018) 93–106, https://doi.org/10.1016/j.jmsy.2018.09.005.

[3] Y. Koren, X. Gu, W. Guo, Reconfigurable manufacturing systems: Principles, design, and future trends, Frontiers of Mechanical Engineering 13 (2) (2018) 121–136, https://doi.org/10.1007/s11465-018-0483-0.

[4] Y. Koren, U. Heisel, F. Jovane, T. Moriwaki, G. Pritschow, G. Ulsoy, H. Van Brussel, Reconfigurable Manufacturing Systems, CIRP Annals 48 (2) (1999) 527–540, https://doi.org/10.1016/S0007-8506(07)63232-6.

[5] J. Morgan, M. Halton, Y. Qiao, J.G. Breslin, Industry 4.0 smart reconfigurable manufacturing machines, J. Manuf. Syst. 59 (2021) 481–506, https://doi.org/10.1016/j.jmsy.2021.03.001.

[6] L.D.C. Ng Corrales, M.P. Lambán, M.E. Hernandez Korner, J Royo, Overall Equipment Effectiveness: Systematic Literature Review and Overview of Different Approaches, Applied Sciences (18) (2020) 10, https://doi.org/10.3390/app10186469.

[7] Haluška, T., Pauliček, R., & Važan, P. (2012). *SOA as A Possible Way to Heal Manufacturing Industry*. 1(2).

[8] Trunzer, E., Lötzerich, S., & Vogel-Heuser, B. (2018). *Concept and Implementation of a Software Architecture for Unifying Data Transfer in Automated Production Systems* (pp. 1–17). https://doi.org/10.1007/978-3-662-57805-6_1.

[9] J.S. Toquica, D. Benavides, J.M.S.T. Motta, Web compliant open architecture for teleoperation of industrial robots, in: *IEEE International Conference on Automation Science and Engineering, 2019-August*, 1408–1414, 2019, https://doi.org/10.1109/COASE.2019.8842847.

[10] E. Romiti, J. Malzahn, N. Kashiri, F. Iacobelli, M. Ruzzon, A. Laurenzi, E. M. Hoffman, L. Muratore, A. Margan, L. Baccelliere, S. Cordasco, N. Tsagarakis, Toward a Plug-and-Work Reconfigurable Cobot, IEEE/ASME Transactions on Mechatronics 27 (5) (2022) 3219–3231, https://doi.org/10.1109/TMECH.2021.3106043.

[11] S. Karnouskos, A.W. Colomboy, F. Jammesz, J. Delsing, T. Bangemann, Towards an architecture for service-oriented process monitoring and control, in: IECON Proceedings (Industrial Electronics Conference), 2010, pp. 1385–1391, https://doi.org/10.1109/IECON.2010.5675482.

[12] A.W. Colombo, S. Karnouskos, T. Bangemann, Towards the Next Generation of Industrial Cyber-Physical Systems. Industrial Cloud-Based Cyber-Physical Systems, Springer International Publishing, 2014, pp. 1–22, https://doi.org/10.1007/978-3-319-05624-1_1.

[13] F. Gosewehr, J. Wermann, W. Borsych, A.W. Colombo, Specification and design of an industrial manufacturing middleware, in: Proceedings - 2017 IEEE 15th International Conference on Industrial Informatics, INDIN 2017, 2017, pp. 1160–1166, https://doi.org/10.1109/INDIN.2017.8104937.

[14] P. Leitão, J. Barbosa, A. Pereira, J. Barata, A.W. Colombo, Specification of the PERFoRM architecture for the seamless production system reconfiguration, in: IECON Proceedings (Industrial Electronics Conference), 2016, pp. 5729–5734, https://doi.org/10.1109/IECON.2016.7793007.

[15] Wojtynek, M., Jochen, ·, Steil, J., & Wrede, S. (2019). *Plug, Plan and Produce as Enabler for Easy Workcell Setup and Collaborative Robot Programming in Smart Factories*. 33, 151–161. https://doi.org/10.1007/s13218-019-00595-0.

[16] M. Simonic, R. Pahic, T. Gaspar, S. Abdolshah, S. Haddadin, M.G. Catalano, F. Worgotter, A. Ude, Modular ROS-based software architecture for reconfigurable, Industry 4.0 compatible robotic workcells, in: 2021 20th International Conference on Advanced Robotics (ICAR), 2021, pp. 44–51, https://doi.org/10.1109/ICAR53236.2021.9659378.

[17] S. Scheifele, J. Friedrich, A. Lechler, A. Verl, Flexible, Self-configuring Control System for a Modular Production System, Procedia Technology 15 (2014) 398–405, https://doi.org/10.1016/J.PROTCY.2014.09.094.

[18] F. Mo, H.U. Rehman, F.M. Monetti, J.C. Chaplin, D. Sanderson, A. Popov, A. Maffei, S. Ratchev, A framework for manufacturing system reconfiguration and optimisation utilising digital twins and modular artificial intelligence, Robot. Comput. Integr. Manuf. 82 (2023), https://doi.org/10.1016/J.RCIM.2022.102524.

[19] Zoitl, A., Kainz, G., & Keddis, N. (2013). Production Plan-Driven Flexible Assembly Automation Architecture. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 8062 LNAI*, 49–58. https://doi.org/10.1007/978-3-642-40090-2_5.

[20] Arrais, R., Ribeiro, P., Domingos, H., & Veiga, G. (2020). ROBIN: An open-source middleware for plug'n'produce of Cyber-Physical Systems: Https://Doi.Org/10.1177/1729881420910316, 17(3). https://doi.org/10.1177/1729881420910316.

[21] International Organization of Standardization. (2010). *ISO 12100:2010(en), Safety of machinery — General principles for design — Risk assessment and risk reduction*. https://www.iso.org/obp/ui/#iso:std:iso:12100:ed-1:v1:en.

[22] AutomationML e.V c/o IAF. (n.d.). *AutomationML in a Nutshell*. https://www.automationml.org/download-archive/.

[23] H.K. Rampersad, Integrated and assembly oriented product design, Integrated Manufacturing Systems 7 (6) (1996) 5–15, https://doi.org/10.1108/09576069610151130.

[24] P.A. P.D. L.N. Ferreira. (2018). *openMOS Common Semantic Model*. Https://Www.Openmos.Eu/Downloads/Deliverables/.

[25] Y. Zhao, L. Gong, Y. Huang, C. Liu, A review of key techniques of vision-based control for harvesting robot, in: Computers and Electronics in Agriculture, 127, 2016, pp. 311–323, https://doi.org/10.1016/j.compag.2016.06.022. Elsevier B.V.

[26] J. Chai, H. Zeng, A. Li, E.W.T. Ngai, Deep learning in computer vision: A critical review of emerging techniques and application scenarios, Mach. Learn. Appl. 6 (2021) 100134, https://doi.org/10.24433/CO.0411648.v1.

[27] Farahani, A., Voghoei, S., Rasheed, K., & Arabnia, H. R. (2020). *A Brief Review of Domain Adaptation*. http://arxiv.org/abs/2010.03978.

[28] C. Shorten, T.M. Khoshgoftaar, A survey on Image Data Augmentation for Deep Learning, J. Big. Data 6 (1) (2019), https://doi.org/10.1186/s40537-019-0197-0.

[29] Labbé, Y., Manuelli, L., Mousavian, A., Tyree, S., Birchfield, S., Tremblay, J., Carpentier, J., Aubry, M., Fox, D., & Sivic, J. (2022). *MegaPose: 6D Pose Estimation of Novel Objects via Render & Compare*.

[30] Y. Liu, Y. Dang, X. Gao, J. Han, L. Shao, Zero-Shot Learning With Attentive Region Embedding and Enhanced Semantics, IEEE Transaction on Neural Networks and Learning Systems (2022) 1–12.

[31] Shugurov, I., Li, F., Busam, B., & Ilic, S. (2022). *OSOP: A Multi-Stage One Shot Object Pose Estimation Framework*.

[32] Minderer, M., Gritsenko, A., Stone, A., Neumann, M., Weissenborn, D., Dosovitskiy, A., Mahendran, A., Arnab, A., Dehghani, M., Shen, Z., Wang, X., Zhai, X., Kipf, T., & Houlsby, N. (2022). *Simple Open-Vocabulary Object Detection with Vision Transformers*.

[33] Zhou, Y., Barnes, C., Lu, J., Research, A., Yang, J., & Li, H. (n.d.). *On the Continuity of Rotation Representations in Neural Networks*.

[34] He, K., Zhang, X., Ren, S., & Sun, J. (n.d.). *Deep Residual Learning for Image Recognition*. http://image-net.org/challenges/LSVRC/2015/.

[35] B. Drost, M. Ulrich, P. Bergmann, P. Härtinger, C. Steger, Introducing MVTec ITODD — A Dataset for 3D Object Recognition in Industry, in: 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), 2017, pp. 2200–2208, https://doi.org/10.1109/ICCVW.2017.257.



https://dspace.org/

Rapid and automated configuration of robot manufacturing cells

Asif, Seemal; Bueno, Mikel; Ferreira, Pedro; Anandan, Paul; Zhang, Ze ...

2025-04-01
Elsevier
https://doi.org/10.1016/j.rcim.2024.102862
Asif S, Bueno M, Ferreira P, et al., (2025) Rapid and automated configuration of robot manufacturing cells. Robotics and Computer-Integrated Manufacturing, Volume 92, April 2025, Article number 102862

Downloaded from a DSpace repository. https://dspace.org/