A Software Architecture for ROS2–CNC Interoperability: Automated Collision-Free Robot Motion Planning and Deterministic Execution*

Samed Ajdinović[1,*], Matthias Marquart[1,*], Benjamin Kaiser[1,2], Siddieq Mansour[1], Andreas Wortmann[1], Oliver Riedel[1] and Alexander Verl[1]

[1]Institute for Control Engineering and Manufacturing Units (ISW), University of Stuttgart, Seidenstr. 36, 70174 Stuttgart, Germany
[2]ISG Industrielle Steuerungstechnik GmbH STEP, Gropiusplatz 10, 70563 Stuttgart, Germany

Abstract

CNC-controlled robots integrate well with PLC/CNC-based automation and enable deterministic motion execution, but typical CNC stacks offer limited support for automated collision-free robot motion planning. As a result, path generation and adaptation often remain manual and brittle, particularly under frequent reconfiguration. ROS2 provides mature motion-planning and perception capabilities (e.g., MoveIt 2), yet is difficult to adopt as an industrial execution backbone due to real-time, safety, and lifecycle constraints. This paper presents a reusable software architecture for ROS2–CNC interoperability that combines automated collision-free robot motion planning with deterministic execution. Sequencing and safety authority remain in TwinCAT 3 PLC/CNC, while planning and optional camera-based scene updates run in ROS2. An ADS-based interface with blockwise trajectory streaming and a PLC-coordinated handshake preserves scene validity during planning and transfers planned trajectories into CNC-executable motion. We validate the approach in software-in-the-loop using a digital twin and demonstrate integration into a CNC-controlled robot cell with 3D scene acquisition.

Keywords

ROS2 motion planning, CNC-based real-time control, Robotics software architecture, Systems integration,

1. Introduction

CNC technology is a backbone of production engineering, enabling precise and highly automated machining. With Industry 4.0, manufacturing systems face increasing product variability and pressure to raise automation levels despite skill shortages [1]. Consequently, robots are deployed more broadly in machine tending, handling, and hybrid cells where robots and CNC machines must coordinate tightly in shared coordinate systems and timing [2].

CNC-based robot execution is attractive in such settings because it reduces heterogeneity in programming concepts and facilitates synchronized motion with other CNC axes [3]. Yet CNC toolchains typically do not provide automated collision-free motion planning for robots. Collision avoidance therefore remains an offline and largely manual activity that must be revisited whenever parts, fixtures, or cell layouts change, which becomes a bottleneck in complex or frequently reconfigured cells.

ROS ecosystems provide complementary capabilities: MoveIt 2 and related components offer collision-aware motion planning and integrate naturally with 3D perception and scene modeling [4]. However, ROS is only partially suitable as the deterministic execution backbone in industrial automation, where PLC-centric sequencing, certified safety supervision, and predictable timing are key [5]. This motivates an architecture that combines ROS planning with industrial-grade execution.

This paper addresses the gap by presenting a reusable software architecture that couples ROS2-based collision-free planning with Beckhoff TwinCAT 3 CNC execution. Planning and camera-based scene capture run on Linux/ROS, while execution, sequencing, and safety authority remain in TwinCAT. A PLC state machine requests plans, coordinates a controlled planning phase, and receives planned joint

Vienna'26: International Workshop on Robotics Software Engineering (RoSE), June 01, 2026, Vienna, Austria
*Corresponding author.
†These authors contributed equally.
samed.ajdinovic@isw.uni-stuttgart.de (S. Ajdinović); matthias.marquart@isw.uni-stuttgart.de (M. Marquart)

trajectories that are translated into CNC-executable motion blocks and streamed via Beckhoff ADS. As a work in progress, we report initial software-in-the-loop validation using a digital twin, summarize integration effort in a CNC-controlled robot cell including a 3D-perception pipeline, and briefly discuss construction robotics as a future evaluation context for more variable environments. [6]

2. Related Work and Positioning

Our work connects three lines of research: (i) CNC-controlled robotics in manufacturing cells, (ii) ROS-based motion planning with perception-driven scene modeling, and (iii) integration patterns that combine best-effort planning with deterministic PLC/CNC execution.

CNC-controlled robotics: CNC-controlled robots reduce heterogeneity in complex workcells and facilitate synchronized motion with other CNC axes [3]. While CNC toolchains provide deterministic interpolation and robust industrial integration, they typically do not offer automated collision-free motion planning; collision avoidance therefore often remains an offline, manual engineering step that becomes brittle under high variance and frequent reconfiguration [7].

ROS motion planning: ROS ecosystems provide widely used components for collision-aware planning and scene management. MoveIt/MoveIt 2 combine planners (often via OMPL), inverse kinematics, and collision checking against a maintained planning scene [4, 8]. They also ease integration of 3D perception updates, but are not designed to replace safety-authoritative, deterministic execution in PLC/CNC infrastructures.

Bridging ROS and industrial control: Prior work demonstrates ROS–TwinCAT integration in different roles: for collaboration with industrial and mobile robots [9], digital-twin-centric frameworks with sensing [10], and ROS-based trajectory generation in industrial assembly contexts [11].

In summary, while ROS–industrial controller coupling is well motivated, retrofitting *CNC-controlled robot execution* with automated collision-free planning remains underexplored. The CNC focus requires translating planned trajectories into CNC-executable representations, transferring them under controller-side constraints (e.g., blockwise streaming), and executing them under PLC sequencing and safety supervision. Our contribution addresses this gap through an ADS-based interface, a PLC-coordinated handshake that preserves scene validity, and a ROS planning stack that fuses CAD geometry with camera-derived obstacles.

3. System Design

This section presents the end-to-end system design that augments a CNC-controlled robot cell with automated collision-free planning while preserving deterministic execution, central sequencing, and safety authority in the industrial control stack. The design follows a deliberate separation into a planning domain (ROS) and an execution domain (TwinCAT PLC/CNC) connected by a narrow integration boundary.

The system consists of a CNC-controlled robot cell, a TwinCAT 3 PLC/CNC control stack, and a ROS2 planning and perception stack (Figure 1). The physical cell comprises an industrial robot and a 3D perception system (Figures 3–4). The camera subsystem (Figure 2) supports (i) task-relevant pose detection (e.g., pick/place) and (ii) obstacle awareness by providing a 3D representation of the workspace. This enables collision checking against obstacles that are not part of the nominal CAD model, such as temporary fixtures or objects introduced during operation.

The industrial control backbone is implemented in Beckhoff TwinCAT 3 and combines a PLC runtime with a CNC/NC execution layer. The PLC implements process sequencing, peripheral control, and safety-relevant supervision, while the CNC layer provides deterministic motion execution (interpolation and enforcement of axis limits) and interfaces the robot axes. ROS2 runs on a dedicated Linux node and hosts MoveIt 2-based planning, scene modeling, and the TwinCAT–ROS interface node. ROS-internal communication uses DDS, whereas cross-domain communication uses Beckhoff ADS over the machine network [6].

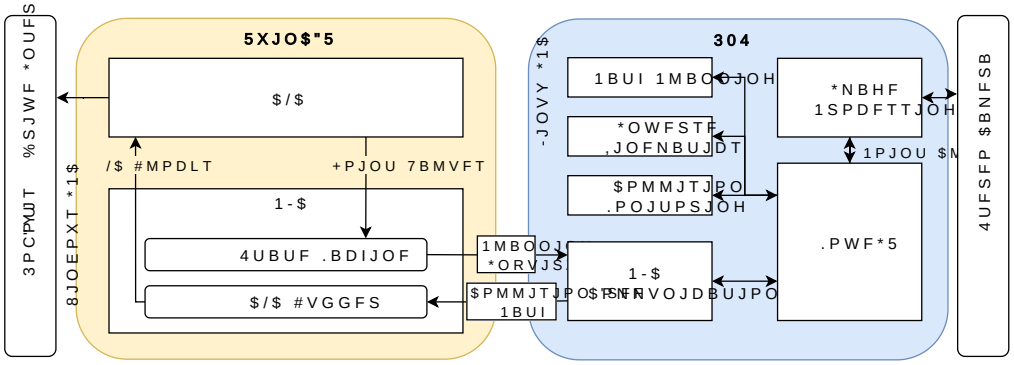

Figure 1: Software architecture of the ROS2–TwinCAT CNC interoperability stack.

The cell follows a request–plan–execute loop coordinated by the PLC state machine. During operation, the PLC maintains the current execution state by reading the robot joint configuration from the CNC/NC layer. When a new motion is required, the PLC issues a planning request to ROS containing (i) the current joint state as planning start, (ii) a goal description (typically a Cartesian pose or constraints), and (iii) planning context such as the active tool, reference frames, and optional process limits.

The ROS interface node converts the request into MoveIt 2 inputs and triggers collision-aware planning against a composite scene. Static collision geometry is derived from CAD models, while dynamic obstacles are inserted from the latest 3D perception update. The resulting joint-space trajectory is returned to TwinCAT, translated into a CNC-executable representation, and transferred blockwise via ADS streaming to satisfy controller-side constraints. TwinCAT reassembles the blocks and schedules deterministic execution in the CNC runtime.

To ensure that the planned trajectory remains valid at execution time, the PLC coordinates a controlled planning phase in which relevant motion is paused and the planning scene snapshot is stabilized. This baseline keeps the validity argument simple and aligns with industrial workflows in which short pauses during replanning are acceptable. Safety authority remains unchanged: interlocks, emergency handling, and mode management are enforced in the TwinCAT domain. ROS provides candidate trajectories, while the PLC/CNC domain decides when motion is enabled and can preempt execution at any time.

The design is intentionally generic: any application that requires advanced planning and perception while retaining deterministic PLC/CNC execution can adopt the same split and interface. Later sections instantiate the design in simulation and outline hardware integration; we also discuss transferability to more variable environments as future work.

4. Software Architecture of the ROS–TwinCAT CNC Interface

This section details the software architecture that couples ROS-based motion planning with TwinCAT 3 PLC/CNC execution. The architecture keeps perception and planning in ROS, while deterministic execution, sequencing, and safety authority remain in TwinCAT. The interface is designed to be explicit with respect to state, timing semantics, and data ownership to support robust commissioning and retrofit.

On the TwinCAT side, a PLC state machine orchestrates the planning–execution cycle. It maintains the current execution state by reading the robot joint configuration from the NC/CNC layer. When a collision-free motion is required, the PLC enters a planning state, pauses motion, and stabilizes the relevant process context. It then sends a planning request containing the current joint state, a Cartesian goal description, and tool/frame context. After a plan is received, TwinCAT assembles the streamed motion blocks into an executable program representation and starts deterministic execution in the CNC layer. During execution, the PLC supervises mode and safety conditions and can preempt motion if required.

On the ROS side, an interface node bridges ADS communication and ROS-native message passing. It receives planning requests, converts them into MoveIt 2 inputs, triggers planning, and returns status and trajectory results. MoveIt 2 combines inverse kinematics, a planning algorithm, and collision checking against a maintained planning scene [8]. The planning scene is composed of a static CAD-based model and dynamic obstacle updates derived from perception.

A dedicated perception node processes depth data from a stereo camera to update the MoveIt 2 planning scene (Figure 2). The pipeline generates an occupancy representation (OctoMap) from depth images and fuses it with the static collision model, enabling collision checking against dynamic obstacles. To reduce spurious self-collisions, the robot geometry is filtered from the depth data using a pose-dependent mask. In addition, selected known objects are localized via fiducials (e.g., ArUco markers) and inserted as predefined collision primitives. Scene updates are applied only in PLC-coordinated stabilized phases (e.g., before planning) to keep the planning scene consistent with subsequent execution.

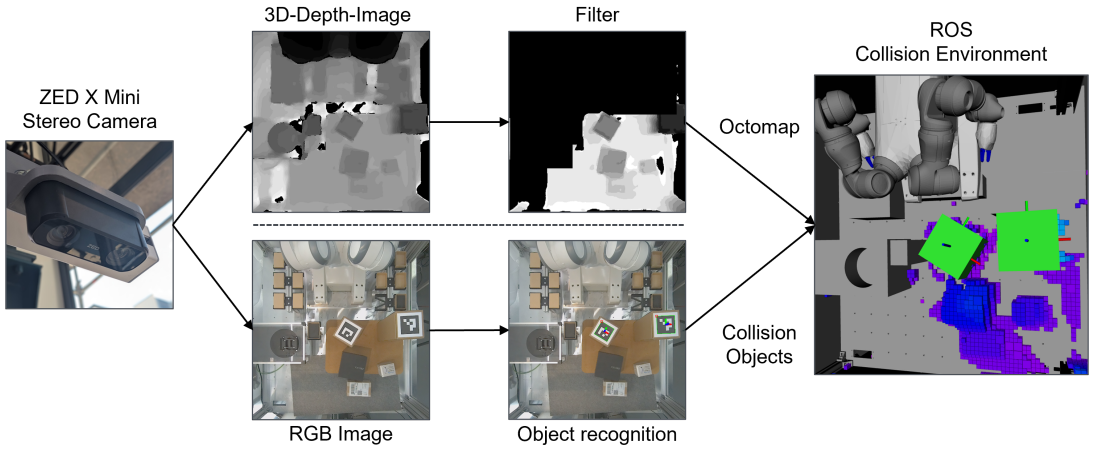

Figure 2: Vision pipeline for dynamic obstacle integration into the MoveIt 2 planning scene.

Cross-domain communication uses Beckhoff ADS between the Windows-based TwinCAT runtime and the Linux-based ROS node [6]. ADS is used for request/response coordination (planning requests, status, diagnostics) and for trajectory transfer. Planned trajectories are transferred blockwise due to controller-side payload limits; TwinCAT acknowledges blocks and reassembles them into a complete motion description prior to execution.

MoveIt 2 produces a joint-space trajectory that is mapped to a CNC-executable representation. In the current implementation, joint waypoints are translated into CNC motion statements suitable for TwinCAT CNC execution. The CNC runtime performs interpolation and enforces configured velocity and acceleration limits, preserving deterministic motion behavior even though planning runs on a non-real-time compute node. To support traceability during commissioning, each planned motion program is associated with a unique identifier and a success/failure status returned from the planning stack.

Validity is ensured through a PLC-coordinated handshake: planning is executed while motion is paused and the scene snapshot is stabilized, and execution is enabled only after the complete trajectory has been transferred and assembled. This baseline supports a clear safety argument for the current system. Future work will extend the baseline toward continuous collision monitoring and event-driven replanning, while preserving the authority boundary in which stopping and safety enforcement remain in the PLC/CNC domain and ROS provides updated trajectory proposals.

5. Validation and Integration Effort

This section summarizes feasibility evidence and integration effort for the proposed ROS–TwinCAT CNC interface. We validate the end-to-end concept in software-in-the-loop (SiL) using a digital twin

and outline integration into an existing CNC-controlled robot cell.

Software-in-the-loop couples the controller implementation to a real-time digital twin and enables safe testing of collision-free planning and deterministic execution under controlled variability. We use ISG-Virtuos for virtual commissioning and evaluate a dynamic pick-and-place scenario with varying start/goal poses and collision contexts, representative of high-variance automation. Figures 3–4 illustrate two collision scenarios with matching planning and real-cell views.

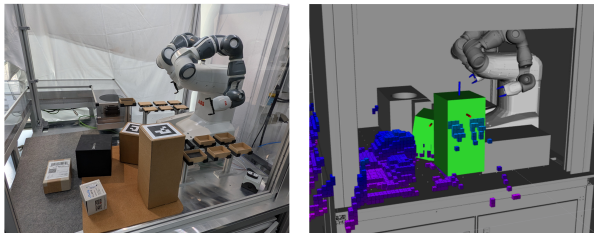

Figure 3: Collision scenario 1: real cell (left) and planning scene (right) with detected collision objects.

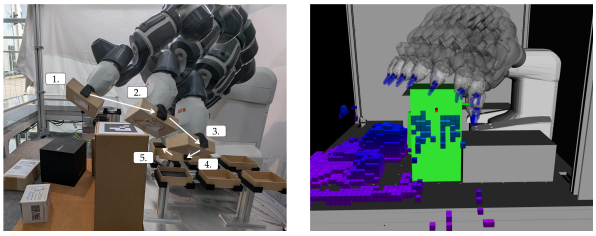

Figure 4: Collision scenario 2: real cell (left) and planning scene (right) with the planned trajectory.

Within this environment, we use MoveIt and evaluate representative OMPL planners (KPIECE, RRT-Connect, CHOMP). RRT-Connect is used as the default for point-to-point motions; Cartesian segments are supported where required. Beyond planner selection, the setup validates the full data path from ADS planning requests to deterministic CNC execution.

We demonstrate hardware integration on an ABB IRB 14000 (YuMi) cell controlled by TwinCAT, co-located with a CNC-controlled XPlanar transport system. A separate Linux IPC hosts ROS2 and the planning stack and exchanges robot state, planning requests, and planned trajectories with TwinCAT via ADS. Dynamic obstacles are captured by a 3D camera; its output is fused into the MoveIt planning scene. Integration primarily requires consistent robot/cell models across domains, ADS protocol mapping with trajectory streaming, PLC state-machine extensions for planning/handshake coordination, and ROS planning/scene configuration. The approach does not require modifying the robot controller or changing the validated safety concept, which supports retrofit into existing cells.

The current results indicate that collision-free planning can be coupled to deterministic CNC execution through a lightweight interface and deployed with moderate additional hardware. Next steps include continuous collision monitoring, event-driven replanning, and systematic evaluation under representative workload and perception-update conditions.

6. Conclusion and Outlook

We presented a software architecture for ROS2–CNC interoperability that combines automated collision-free robot motion planning with deterministic execution on TwinCAT 3 PLC/CNC. The approach preserves PLC-centric sequencing and safety authority while integrating ROS 2 planning and optional 3D scene updates via an ADS-based interface, blockwise trajectory streaming, and a PLC-coordinated handshake. Validation in software-in-the-loop and on a CNC-controlled robot cell demonstrates feasibility and supports retrofit into existing automation environments.

As next evaluation contexts, we target more variable environments such as construction robotics and on-site production, where less structured workspaces and frequent reconfiguration increase the need for perception-driven planning [12]. Future work focuses on two extensions: (i) evolving plan–pause–execute toward continuous collision monitoring and event-driven replanning while preserving safety authority in the PLC/CNC domain, and (ii) connecting motion, execution, and process data to manufacturing data ecosystems [13], including Digital Product Passport (DPP) workflows, to improve traceability across the product lifecycle [14].

Acknowledgments

This research was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC 2120/2 – 390831618.

Declaration on Generative AI

During the preparation of this work, the authors used Grammarly in order to: Grammar and spelling check.

References

[1] UN Trade, Development, Digital Economy Report 2019: Value Creation and Capture: Implications for Developing Countries, United Nations Publications, New York, 2019. URL: https://unctad.org/system/files/official-document/der2019_en.pdf.

[2] M. Soori, F. K. G. Jough, R. Dastres, B. Arezoo, Robotical automation in cnc machine tools: A review, Acta Mechanica et Automatica 18 (2024) 434–450. doi:10.2478/ama-2024-0048.

[3] D. Pfeifer, S. Fauser, C. Scheifele, J. Fehr, Reusage of extended digital twins from virtual commissioning for dynamics-based trajectory planning in cnc controlled robotics, Springer, Cham, 2024, pp. 151–159. doi:10.1007/978-3-031-74482-2_18.

[4] A. Koubaa (Ed.), Robot Operating System (ROS): The Complete Reference (Volume 1), volume 625 of *Studies in Computational Intelligence*, Springer International Publishing, Cham, 2016.

[5] Open Robotics, Ros: Robot operating system, 2024. URL: https://ros.org/.

[6] Beckhoff Automation GmbH & Co. KG, Beckhoff information system: Ads, 2025. URL: https://infosys.beckhoff.com/index.php?content=../content/1031/tcinfosys3/11291871243.html&id=.

[7] L. Larsen, Auf dem Weg zu einer flexiblen Produktion: automatische und kollisionsfreie Bahnplanung für kooperierende Industrieroboter, Universität Augsburg, Augsburg, 2019. doi:654940.

[8] D. T. Coleman, I. A. Sucan, S. Chitta, N. Correll, Reducing the barrier to entry of complex robotic software: a moveit! case study (2014). doi:10.6092/JOSER.

[9] S. Mansour, S. Ajdinović, A. Lechler, A. Verl, Autonomous mobile robot localization for material transport in construction prefabrication, in: 2025 25th International Conference on Control, Automation and Systems (ICCAS), 2025, pp. 495–500. doi:10.23919/ICCAS66577.2025.11301087.

[10] A. Sterk-Hansen, B. H. Saghaug, D. Hagen, M. F. Aftab, A ros 2 and twincat based digital twin framework for mechatronics systems, in: 2023 11th International Conference on Control, Mechatronics and Automation (ICCMA), IEEE, 2023, pp. 485–490. doi:10.1109/ICCMA59762.2023.10374978.

[11] N. Terei, R. Wiemann, A. Raatz, Ros-based control of an industrial micro-assembly robot, Procedia CIRP 130 (2024) 909–914. doi:10.1016/j.procir.2024.10.184.

[12] M. Marquart, S. Ajdinović, S. Mansour, F. Dorsch, A. Lechler, A. Verl, Enhancing automation in robotic coreless filament winding through camera-based teach-in, in: 2025 7th International Conference on Control and Robotics (ICCR), 2025, pp. 308–315. doi:10.1109/ICCR67607.2025.11372094.

[13] Michael Neubauer, Lukas Steinle, Colin Reiff, Samed Ajdinović, Lars Klingel, Armin Lechler, Alexander Verl, Architecture for manufacturing-x: Bringing asset administration shell, eclipse dataspace connector and opc ua together, Manufacturing Letters 37 (2023) 1–6. doi:10.1016/j.mfglet.2023.05.002.

[14] S. Ajdinović, M. Strljic, A. Lechler, O. Riedel, Interoperable digital product passports: An event-based approach to aggregate production data to improve sustainability and transparency in the manufacturing industry, in: 2024 IEEE/SICE International Symposium on System Integration (SII), IEEE, 2024, pp. 729–734. doi:10.1109/SII58957.2024.10417487.