Lasers in Manufacturing Conference 2025

Integration of galvanometer scanners in ROS for path planning in robot-based laser material processing

Jan-Niklas Schneider[a*], Leon Gorißen[a], Thomas Kaster[a], Christian Hinke[a]

[a]*Chair for Laser Technology, RWTH Aachen University, Steinbachstr. 15, 52074 Aachen, Germany*

Abstract

The integration of galvanometer scanners and robotic manipulators in laser material processing poses challenges due to their independent hardware and software architectures. Synchronizing these systems requires precise coordination of robotic motion and galvanometer scanner control, particularly for complex path planning tasks. This paper introduces a framework that leverages the Robot Operating System (ROS) and its open-source, modular, and widely-adopted capabilities in robotics to unify the control of robot and galvanometer scanner. A service-oriented, publish-subscribe wrapper and a ros2_control hardware interface are implemented to manage scanner mirror positions and robotic joint movements, ensuring accurate laser beam positioning and process consistency. Future research may explore additional factors such as dynamic power control, deep learning, and vision-based system integration. This scalable and flexible approach demonstrates the potential of ROS to drive innovations in laser material processing.

Keywords: ROS (Robot Operating System); Galvanometer Scanners; Path Planning; Laser Material Processing; Process Control;

1. Introduction

The combination of industrial robotic manipulators and galvanometer scanners is a well-established approach in laser material processing. Hybrid systems leverage the flexibility and workspace of robots with the high-speed deflection capabilities of galvanometer scanners, enabling applications such as laser marking, cleaning, and welding. Despite their proven industrial relevance, such systems are typically implemented using proprietary hardware–software solutions, often delivered as closed systems by equipment manufacturers or system integrators. These approaches limit interoperability, hinder system customization, and complicate integration into broader automation environments.

In parallel, the Robot Operating System ROS (Quigley, 2009) — particularly its second generation ROS 2 (Macenski et al., 2022) — has become a widely adopted open-source standard in robotics research and increasingly in industrial applications. Its modular architecture, standardized communication patterns, and active community support the integration of heterogeneous components in complex robotic systems (Robinson, 2022). Moreover, ROS 2 addresses key industrial requirements such as real-time communication, deterministic execution, and support for safety-critical deployment, thereby bridging the gap between academic prototyping and production-grade solutions (Puck et al., 2021).

ROS 2's significance is reflected in the growing support of major industrial stakeholders. The recent formation of the Open Source Robotics Alliance (OSRA) unites companies such as Intrinsic (Alphabet), Bosch, NVIDIA, and Qualcomm with the open-source community to ensure the long-term sustainability of key robotics tools like ROS and the Gazebo — a widely used simulator for testing robot behavior in virtual environments (Bosch, 2024; Open Source Robotics Foundation, 2024). This industry-backed initiative highlights the relevance of ROS 2 as a standard platform for both research and industrial-scale automation.

* Corresponding author. Tel.: +49 241 8040425.
E-mail address: jan-niklas.schneider@llt.rwth-aachen.de.

Although ROS 2 has established itself in fields like mobile robotics, automation, and manufacturing, its application in laser material processing — and particularly in systems integrating galvanometer scanners — remains limited. Early projects, such as OpenLMD, demonstrate how ROS can be used to control laser-based processes like laser metal deposition by leveraging its modular middleware for integrating sensors, actuators, and control strategies (García-Díaz et al., 2018). These examples suggest that ROS 2 holds untapped potential for laser-based applications, especially in systems requiring flexible integration and adaptive control.

Moreover, ROS 2 is particularly well-suited for the development of non-standard kinematic systems, where modularity and configurability are critical. This flexibility supports research and industrial efforts to adapt novel robotic architectures for specialized processes — including those in laser material processing — as shown by recent works in the field of sensor-driven path planning and adaptive kinematics (Kaster et al., 2025).

This work presents a ROS 2-based integration framework for galvanometer scanners combined with industrial robots. It focuses on hardware integration by wrapping the proprietary scanner software development kit (SDK) into standardized ROS 2 components, enabling scanner control within modular robotic environments. While motion planning integration remains a future research goal, this foundation facilitates the use of ROS 2 tools like ros2_control and MoveIt 2 in subsequent developments.

The remainder of this paper is structured as follows: Section 2 discusses related work on robot–scanner integration and ROS-based industrial applications. Section 3 outlines the system architecture and the ROS 2-based software design. Section 4 describes the implementation of scanner control components and their integration with the robot. Section 5 provides a discussion of system performance, current limitations, and future research opportunities. Section 6 concludes the paper.

2. Background and Related Work

Robot-mounted galvanometer scanners are well established in industrial applications such as marking, welding, and cutting. Commercial systems — including Blackbird/SCANLAB intelliWELD for welding (Blackbird Robotersysteme GmbH, 2017) and TRUMPF PFO for programmable focusing (TRUMPF SE + Co. KG, 2018) — enable synchronized robot–scanner operation with integrated monitoring. However, these solutions are delivered as proprietary hardware–software stacks, restricting interoperability and hindering open integration.

In research, multi-device control architectures like master–slave or dual-controller setups have been explored for synchronizing robot and scanner motion, typically focusing on motion coordination for "on-the-fly" or step-scan strategies. Yet, such approaches remain closed or application-specific, with no standard framework for general robot–scanner integration. (Lange et al., 2025; Kuppuswamy et al., 2007)

The Robot Operating System (ROS) has emerged as a modular platform for integrating heterogeneous robotic components, extended to industrial applications by initiatives like ROS-Industrial (ROS-Industrial Consortium), which fosters standardized control and interoperability for manufacturing systems. ROS-based systems like ROSWELD (Thomessen and AS, 2020), which supports multi-pass robotic welding with modular path planning, MyWelder (Ferraguti et al., 2023), a collaborative MIG/MAG welding assistant focusing on human–robot interaction, and OpenLMD (García-Díaz et al., 2018), which integrates laser metal deposition with real-time sensor feedback and adaptive control, exemplify how ROS can support complex material processing workflows. These projects leverage ROS's capabilities for modular integration of planning, control, and monitoring, often with extensions for process-specific tasks.

However, despite their success in specific applications, none of these frameworks target the integration of high-dynamic galvanometer scanners or explore their real-time coordination with robotic manipulators — especially within the standardized interfaces of ROS 2. This underscores a critical gap in applying ROS to hybrid laser processing systems, where fast scanner deflection must be tightly coupled with robot motion and sensor-based feedback.

To date, no open-source ROS 2 framework supports the integrated control of galvanometer scanners in combination with robotic manipulators. This gap highlights the opportunity for a modular, open approach that bridges proprietary scanner control with ROS 2's standardized interfaces — enabling flexible hardware integration and paving the way for adaptive, sensor-driven laser processing.

3. System Overview and Architecture

The integration of a galvanometer scanner with a robotic manipulator for laser material processing requires a coordinated hardware and software system capable of precise control. Given the fundamentally different control architectures of industrial robots and galvanometer scanners, a unified framework is essential to ensure accurate laser beam positioning, consistent motion execution, and flexible task planning. This chapter presents the system architecture developed to address these challenges, focusing on a modular and extensible implementation using the ROS 2. The following sections describe the hardware setup, the layered ROS 2-based integration approach, and the mechanisms used to control the galvanometer scanner in ROS 2 environment.

3.1. Hardware Setup

The experimental hardware setup consists of three main components: a collaborative robot, a galvanometer scanner, and a pulsed fiber laser (see Fig. 1). Together, these components form a compact and flexible platform for robotic laser material processing for e.g. laser marking or laser cleaning processes (Gorißen et al.).

The robotic manipulator used is a Universal Robots UR5e, a 6-degree-of-freedom (DOF) collaborative robotic arm with a payload capacity of 5 kg and a reach of 850 mm. The robot offers ±0.03 mm repeatability, making it suitable for applications requiring precise positioning and smooth trajectory execution.

At the robot's end-effector, a Scanlab ScanCube7 galvanometer scanner is mounted. This device enables high-speed laser beam deflection via two rotating mirrors (X and Y axes) and is controlled by a Scanlab RTC6 control board. The scanner features a 300 mm focal length, resulting in a beam waist diameter of approximately 105 µm. This setup allows for fast process movements independent of the robot's mechanical motion.

The laser source is an SPI redEnergy G4 pulsed fiber laser with an average power of 100 W, a wavelength of 1060 nm, and a pulse duration range from 4 to 2000 ns. It operates with a repetition rate of 1 to 4000 kHz, supporting material processing tasks such as marking or cleaning.

All components are connected through a centralized control PC running Ubuntu 24.04 with a ROS 2 middleware stack. The PC communicates with the UR5e robot over Ethernet using the official ROS-compatible Universal Robots driver. The galvanometer scanner is also connected via Ethernet through a Scanlab RTC6 control board, which is accessed using the proprietary RTC6 SDK. A custom-developed ROS 2 wrapper and a custom-developed ROS 2 hardware interface communicate with this SDK to expose scanner control functionality via standard ROS communication mechanisms such as services and topics.

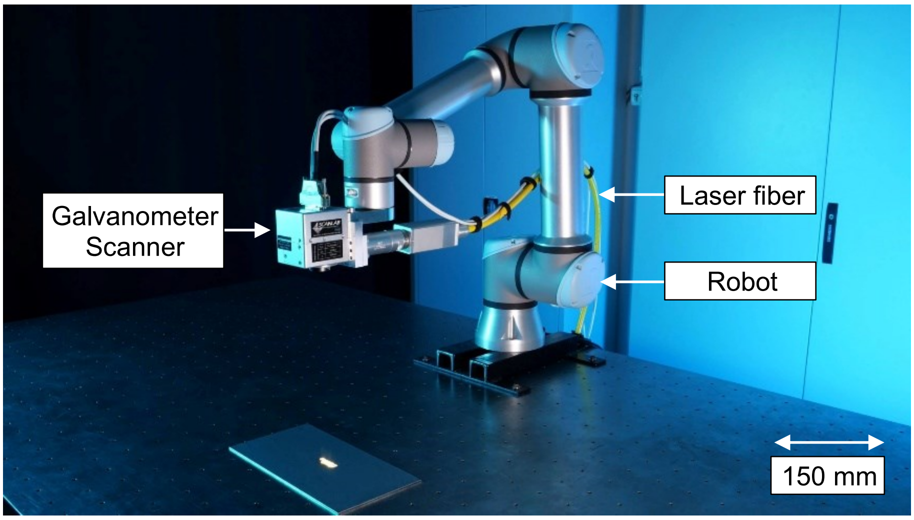

Fig. 1.Experimental setup for the integration of a galvanometer scanner into a robotic laser processing system. The setup includes a UR5e collaborative robot, a Scanlab ScanCube7 galvanometer scanner controlled via a Scanlab RTC6 control board, and a SPI redEnergy G4 pulsed fiber laser. The scanner is mounted on the robot's end-effector, enabling combined positioning through both robotic movement and mirror deflection. The hardware is controlled via ROS 2-based modular architecture.

3.2. ROS 2-Based Software Architecture

The ROS 2-based software architecture follows a layered and modular design, reflecting the separation between motion planning, hardware abstraction, and hardware communication. The architecture supports different levels of scanner integration depending on the control strategy and system requirements.

At the core of the software stack are two control strategies for operating the galvanometer scanner via the RTC6 interface:

- Wrapper-based control (see section 4.1):
  A stateless ROS 2 node wraps the RTC6 SDK, exposing scanner functions via services and topics. This approach enables flexible control and testing but does not support trajectory planning or closed-loop control.
- Hardware-interface-based control (see section 4.2):
  A custom hardware interface within ros2_control treats the scanner mirrors as joints, enabling compatibility with standard ROS controllers and providing a basis for future synchronized operation.

All hardware components are represented in standardized URDF/Xacro models, translating the physical system into a virtual description. The combined model references modular sub-descriptions for robot and scanner and serves as a common basis for control, planning, and visualization within the ROS 2 ecosystem, see section 4.3.

Motion planning is supported by two dedicated MoveIt 2 configuration packages — scancube_moveit_config for standalone scanner planning and ur5e_scancube_moveit_config for combined robot–scanner applications. These configuration packages provide the necessary setup for MoveIt 2, including robot descriptions, planning group definitions, kinematic parameters, and default planning pipelines. They enable joint visualization, collision checking, and system validation within the MoveIt 2 framework but do not implement custom motion planning algorithms.

In contrast, the scancube_motion_planner package complements this setup by providing custom motion planning utilities tailored to scanner-specific path generation. While the configuration packages define the planning environment, the scancube_motion_planner implements functions that compute application-specific scan paths, which are then passed to the RTC6 control card for precise low-level execution.

System visualization and state monitoring are supported by visualization tools such as RViz, which receive model descriptions, state feedback, and planning data from the various software components.

Fig. 2 illustrates the modular architecture of the ROS 2 software stack, including control layers, planning modules, description packages, and their interconnections within the ROS 2 ecosystem. The robot-related packages are provided by the manufacturer and are integrated unchanged into the system. For completeness, the combined motion planner is included in the diagram, although its implementation remains a subject of future research. The visualization tool is omitted from the figure for clarity, as it receives input from all relevant nodes, as previously described.

Robot and scanner differ in their dynamic behavior — the robot being capable for precise positioning in large workspace yet comparatively small motion speed and the scanner executing fast beam deflections. While both are integrated into the ROS 2 control framework, they currently operate independently. The architecture is prepared for future synchronization, either through coordinated ROS 2 control or hardware triggers, while the decomposition of combined motion tasks remains a topic of ongoing research.

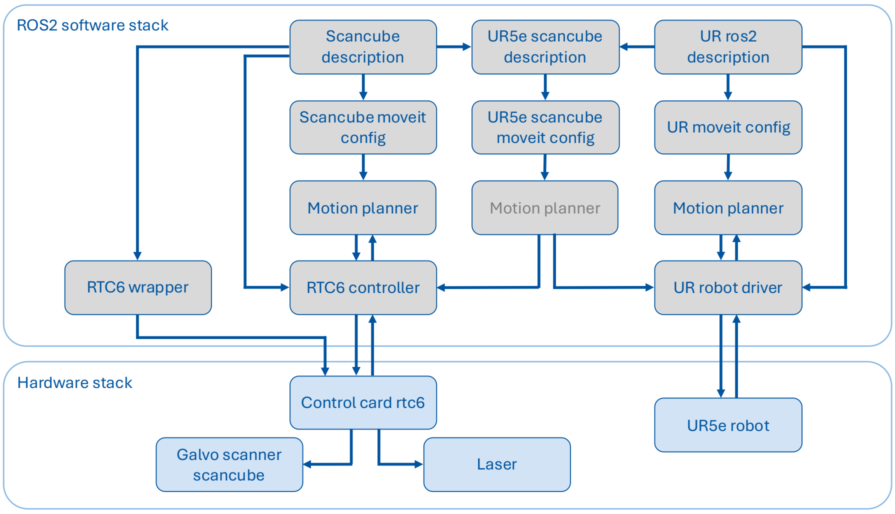

Fig. 2. System architecture of the ROS 2-based integration framework. The architecture follows a layered design with a central control layer, a modular ROS 2 software stack—including description packages, motion planning components, hardware interfaces, and the RTC6 wrapper—and the hardware layer. The motion planner for combined motion is not yet included and subject of future research.

4. Implementation

4.1. RTC6 Wrapper Node

The rtc6_wrapper package implements a stateless ROS 2 node that directly interfaces with the proprietary RTC6 SDK provided by the scanner manufacturer. Its primary purpose is to expose scanner control functions via standardized ROS 2 communication mechanisms, enabling seamless integration into ROS-based robotic systems.

The wrapper node provides:

- Service-based control commands such as pilot laser activation (PilotLaserOn), deactivation (PilotLaserOff), and list execution (ExecuteList).
- ROS topics for direct scanner commands, including mirror positioning (GotoXY), absolute marking (MarkAbs), and jump commands (JumpAbs).
- Trajectory command handling, enabling the node to receive JointTrajectory messages (e.g., from ROS controllers or planners) and translate them into scanner commands based on joint names.

The wrapper follows a stateless design — each command or service call is directly mapped to a corresponding SDK function without maintaining internal states. This ensures flexible integration in various system configurations and simplifies debugging. Initialization requires the scanner's IP address and a correction file path to configure the RTC6 board. During runtime, the node listens to incoming ROS 2 messages and executes the corresponding SDK functions, optionally publishing feedback. While the rtc6_wrapper enables flexible scanner control, it does not implement any closed-loop control or trajectory planning. These tasks remain with higher-level components such as ROS 2_control or dedicated motion planners.

4.2. RTC6 Controller

While the rtc6_wrapper provides a lightweight, stateless interface for direct scanner commands via ROS 2 services and topics, the rtc6_controller integrates the scanner into the ros2_control framework as a hardware component with standardized joint interfaces. This integration allows higher-level tools — such as motion planners, controllers, or monitoring nodes — to interact with the scanner using the same interfaces applied to conventional robotic joints. The scanner's X and Y mirror axes are represented as virtual joints within ROS 2, exposing state interfaces for position and velocity reporting, as well as command interfaces for target positions.

At runtime, these interfaces pass the commanded positions directly to the RTC6 hardware. All low-level motion execution, trajectory timing, control loops, and precise mirror regulation remain fully handled by the RTC6 control card. The hardware interface additionally manages scanner initialization, including Ethernet discovery, correction file loading, and parameter configuration.

Complementing the hardware interface, a dedicated trajectory controller implements a standard ROS 2 action server for trajectory execution. It accepts trajectory goals, converts them into scanner command sequences (such as jump or mark instructions), and triggers their execution on the RTC6. The controller monitors hardware execution and reports goal outcomes accordingly.

This architecture ensures that ROS 2 provides unified access for planning, control, and monitoring, while real-time motion control, timing, and beam regulation remain strictly within the RTC6 hardware. Compared to the stateless wrapper, the hardware interface allows tighter ROS 2 integration, treating the scanner as a fully integrated component for multi-device planning and visualization.

4.3. Combined Robot–Scanner Integration

The scancube_description package defines the ScanCube7 scanner as a chain of virtual links and joints. Its core element is a configurable Xacro macro that allows the scanner to be attached to any robot or reference frame. The model includes fixed joints for the scanner housing, visual and collision meshes for simulation and planning, a prismatic joint for each mirror axis, and a simplified representation of the optical axis.

Although the physical scanner uses rotational mirrors, the axes are modeled as prismatic joints in the URDF. This design choice matches the input format expected by the RTC6 SDK, which uses Cartesian X/Y deflections. Modeling prismatic joints avoids unnecessary conversions within the ROS stack, as the RTC6 hardware internally handles the transformation from Cartesian positions to mirror angles.

The description also integrates with ros2_control by registering the scanner as a hardware system and assigning command and state interfaces to the virtual joints. The combined model is extended by the ur5e_scancube_description package, which attaches the scanner to the UR5e's tool frame without duplicating definitions.

This modular setup ensures consistency across planning, control, and visualization, supports reuse in different scenarios, and simplifies system maintenance. Fig. 3 shows the combined UR5e and ScanCube7 model rendered in RViz, illustrating the joint hierarchy and the modular integration of both components within the ROS 2 environment. The visualization serves both for model validation and system monitoring during runtime.

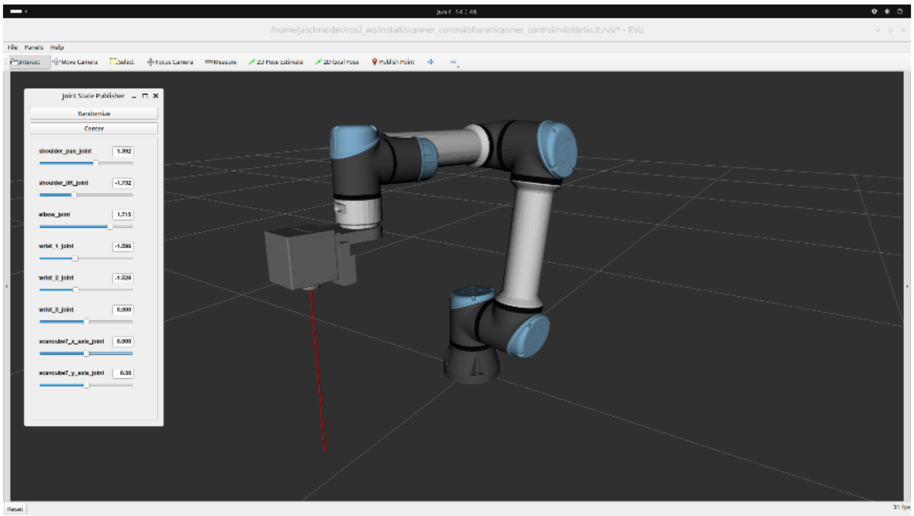

Fig. 3. Visualization of the integrated robot–scanner system in RViz. The UR5e robot and ScanCube7 galvanometer scanner are modeled in a unified ROS 2 description, enabling realistic simulation and visualization of combined motion. The simulated scanner beam path and mirror positions support development and debugging of path planning algorithms prior to deployment on physical hardware.

5. Discussion

This work implements two complementary integration strategies for the RTC6-controlled scanner: a stateless wrapper exposing SDK functions via ROS 2 services and topics, and a ros2_control hardware interface that represents the scanner's mirror axes as joints. While the wrapper facilitates rapid prototyping and debugging, the hardware interface connects the scanner to standard controllers and planning workflows, delegating precise timing to the RTC6. This dual approach combines fast initial integration with consistent long-term architecture.

However, robot and scanner motions are currently executed independently. The robot handles macro positioning, while the scanner follows predefined intra-field paths — without deterministic synchronization or active hardware triggers linking their execution. Real-time timing and laser control remain confined to the proprietary RTC6 firmware.

The system's modular URDF/Xacro setup allows easy substitution of different robot arms or scanner devices with minimal adjustments. This modularity also leaves room for future synchronization strategies, whether through time-aware executors or external trigger signals between robot controller and scanner hardware.

For trajectory planning, robot and scanner motions are presently handled separately. Still, the MoveIt 2 configuration already co-visualizes both joint sets, providing a basis for future composite planners. Planned developments include joint optimization of robot poses and scanner paths to minimize repositioning time and field distortion, as well as adaptive control loops that co-plan laser power and scan velocity using sensor feedback.

Compared to existing systems, proprietary remote laser solutions offer tight coupling and integrated monitoring but are constrained by closed synchronization chains. Academic approaches using cascaded controllers achieve higher update rates with specialized setups but lack a reusable open middleware abstraction. In contrast, this work prioritizes openness and reusability over maximum throughput.

Ultimately, a unified abstraction of robot and scanner addresses a key integration gap while aligning with ROS 2's industrial sustainability goals. Future research will evaluate deployment efficiency, hardware portability, and the potential of adaptive control strategies inspired by existing ROS-based process frameworks.

6. Conclusion and Outlook

This work presents a ROS 2 integration framework that unifies a galvanometer scanner and a collaborative robot through two abstraction layers: an SDK wrapper for fast prototyping and a ros2_control hardware interface that exposes the mirror axes as virtual joints in a modular URDF/Xacro description. The approach enables consistent visualization and planning configuration while the proprietary RTC6 controller retains low-level timing and laser triggering.

The framework closes a gap between proprietary industrial remote laser processing solutions and specialized academic synchronization stacks by offering a reusable, open foundation. Alignment with ROS 2's modular ecosystem and active industry support positions hybrid laser processing cells for sustainable evolution.

Forthcoming work will quantify latency, timing jitter, and positional repeatability to establish baseline performance of the unified robot–scanner system. These results will guide development and comparison of synchronized and fully coupled path planners that co-optimize robot macro motion and mirror deflection. Subsequent extensions will embed laser-specific control (e.g. power modulation, adaptive scan velocity) and sensor hooks to enable process-aware and eventually closed-loop execution.

Acknowledgements

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC-2023 Internet of Production – 390621612. The authors acknowledge the financial support by the Federal Ministry of Education and Research of Germany in the framework of Research Campus Digital Photonic Production (project number: 13N15423).

References

Blackbird Robotersysteme GmbH, 2017. Datasheet IntelliWELD. Accessed July 20, 2025. https://www.blackbird-robotics.de/fileadmin/user_upload/pdf/19_intelliWELD_scan_systems.pdf.

Bosch Joins the Open Source Robotics Alliance (OSRA), 01.07.2024. News release. July 1, 2024. Accessed July 19, 2025. https://www.bosch.com/research/news/open-source-robotics-alliance.

Ferraguti, F., Villani, V., Storchi, C., 2023. MyWelder: A Collaborative System for Intuitive Robot-Assisted Welding. *Mechatronics* 89:102920. doi:10.1016/j.mechatronics.2022.102920.

García-Díaz, A., Panadeiro, V., Lodeiro, B., Rodríguez-Araújo, J., Stavridis, J., Papacharalampopoulos, A., Stavropoulos, P., 2018. OpenLMD, an Open Source Middleware and Toolkit for Laser-Based Additive Manufacturing of Large Metal Parts. *Robotics and Computer-Integrated Manufacturing* 53:153–61. doi:10.1016/j.rcim.2018.04.006.

Gorißen, L., Schneider, J.-N., Kaster, T., Hinke, C., Häfner, C. Towards the Application of Low-Cost Collaborative Cobots in Laser Materials Processing, in" *Proceedings of LAMP2025 – the 9th International Congress on Laser Advanced Materials Processing*".

Kaster, T., Rissom, J.-H., Breit, E., Schneider, J.-N., Gorissen, L., Hinke, C., 2025. Laser Welding Using Mobile Robots: An Approach for the Efficient Processing of Large Components, Proceedings of SPIE.

Kuppuswamy, N., Schmidt, R., Seeger, F., Zhang, S., 2007. Finite Element Modeling of Impact Strength of Laser Welds for Automotive Applications, Safety and Security Engineering II, p. 375.

Lange, F., Andrade Both, L. de, Zecherle, M., Schlüter, H., Stollenwerk, J., Holly, C., 2025. Controlling Ultrafast, Adaptable and Synchronized Cascaded Scan Systems for Laser Materials Processing. *Optics & Laser Technology* 192:113108. doi:10.1016/j.optlastec.2025.113108.

Macenski, S., Foote, T., Gerkey, B., Lalancette, C., Woodall, W., 2022. Robot Operating System 2: Design, Architecture, and Uses in the Wild. *Science robotics* 7 (66). doi:10.1126/scirobotics.abm6074.

Open Robotics Launches the Open Source Robotics Alliance: A New Membership Initiative from the OSRF Will Oversee and Support the ROS Suite Projects, 18.03.2024. News release. March 18, 2024. Accessed July 19, 2025. https://osralliance.org/open-robotics-launches-the-open-source-robotics-alliance-2/.

Puck, L., Keller, P., Schnell, T., Plasberg, C., Tanev, A., Heppner, G., Roennau, A., Dillmann, R., 2021. Performance Evaluation of Real-Time ROS2 Robotic Control in a Time-Synchronized Distributed Network, 2021 IEEE 17th International Conference on Automation Science and Engineering (CASE), p. 1670.

Quigley, M., 2009. ROS: An Open-Source Robot Operating System, IEEE International Conference on Robotics and Automation.

Robinson, M., 2022. From Academia to Industry and Beyond: ROS Is Advancing Robotics for Tomorrow's Challenges. Accessed July 19, 2025. https://ifr.org/post/from-academia-to-industry-and-beyond.

ROS-Industrial Consortium. ROS-Industrial: Open Source Middleware for Industrial Robot Integration. Accessed July 22, 2025. https://rosindustrial.org.

Thomessen, T., AS, P., 2020. ROSWELD: A ROS-Based Framework for Planning, Monitoring and Control of Multi-Pass Robot Welding. https://www.rosin-project.eu/ftp/rosweld.

TRUMPF SE + Co. KG, 2018. Intelligent Process Machining “On-the-Fly”. Accessed July 20, 2025. https://www.trumpf.com/filestorage/TRUMPF_Master/Products/Lasers/02_Brochures/TRUMPF-I-PFO-processing-optics-Flyer-EN.PDF.