machines

MDPI

Article

Real-Time Big Data Pipelines for Industrial Robot Digital Twins: An OMPL Benchmarking Framework

Metin Yılmaz [1,2,*], Cem Suha Yılmaz [3], Serhat Kahraman [2,4] and Uğur Yayan [5]

[1] Autonomous Systems & Technology for TR-Metin Yılmaz, 26200 Eskisehir, Türkiye
[2] Department of Computer Engineering, Eskisehir Osmangazi University, 26040 Eskisehir, Türkiye
[3] Department of Electrical and Electronics Engineering, Eskisehir Osmangazi University, 26040 Eskisehir, Türkiye; cshyilmaz@gmail.com
[4] DEFTR Tek. Elek. Müh. Dan. Enerji ve Ulus. Tic. Ltd., 06374 Eskisehir, Türkiye; serhatkahraman@deftr.com
[5] Department of Software Engineering, Eskisehir Osmangazi University, 26040 Eskisehir, Türkiye; ugur.yayan@ogu.edu.tr
* Correspondence: metinyilmaz26@gmail.com

Abstract

The seamless integration of real-time operational technology (OT) with big data architectures remains a critical bottleneck in developing robust robotic Digital Twins. Furthermore, evaluating stochastic motion planners strictly within pristine simulations obscures vital real-world challenges such as sensor noise, communication latency, and mechanical stress. This study presents a high-throughput, real-time Hardware-in-the-Loop (HIL) pipeline integrating ROS 2, Apache Kafka, and Functional Mock-up Units (FMUs). Using a UR10e manipulator in a constrained industrial environment, we conducted extensive physical benchmarking of 11 Open Motion Planning Library (OMPL) algorithms across 10 repetitions, generating a comprehensive dataset of 785,192 samples. The proposed IT/OT architecture achieved deterministic millisecond-level synchronization, bounding end-to-end communication latency between 0.09 and 15.51 ms. Physical executions revealed a macroscopic "topological divergence" between simulation and reality, with spatial deviations peaking at 457.65 mm due to real-world point-cloud noise. While algorithms like EST and KPIECE demonstrated optimal geometric efficiency (e.g., a mean path length of 14.57 m) and hardware-friendly dynamics, traditional planners like RRT generated severe inertial spikes of up to 100 N, demonstrating substantial unsuitability for continuous industrial deployment. The primary contribution is a methodologically novel, rigorously validated big data pipeline and the release of an open-source, 50 Hz multimodal dataset (spatial, temporal, and dynamic forces), bridging the sim-to-real gap and providing a foundational benchmark for future data-driven robotic applications.

Keywords: real-time data pipeline; hardware-in-the-loop; motion planner benchmarking; digital twin; OMPL

Check for updates

Academic Editor: Zheng Chen

Received: 23 April 2026
Revised: 2 June 2026
Accepted: 3 June 2026
Published: 18 June 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

The evolution of industrial automation towards the Industry 5.0 paradigm envisions robots transitioning from pre-programmed machines operating in closed cages to intelligent, autonomous collaborators functioning in dynamic, unstructured, and human-shared environments. In this context, industrial robot systems offer high flexibility, precision, and automation capabilities in modern manufacturing processes. Digital Twin (DT) approaches have become increasingly vital, enabling the continuous monitoring and evalu-

ation of physical–virtual behavioral alignment by transferring high-frequency data from a real robot system to a virtual replica. However, modeling and verifying the behavior of physical robots remains a significant research challenge, particularly due to complex motion-planning requirements, sensor noise, and time-synchronization issues.

Efficient path planning—where manipulators navigate around obstacles to reach target coordinates—is critical for operational efficiency, especially in complex automotive and assembly lines. While traditional sampling-based algorithms, such as Rapidly exploring Random Trees (RRT) and Probabilistic Roadmaps (PRM), are widely utilized, they often compute paths from scratch for single-query problems, leading to latency and high computational loads in real-time applications. Furthermore, existing data collection methodologies generally rely exclusively on simulation environments (e.g., Gazebo, MuJoCo) or are restricted to simple "pick-and-place" tasks. There is a significant gap in the literature regarding comprehensive, multimodal datasets that cover complex assembly scenarios (e.g., bus chassis operations), synchronize physical and simulated data with millisecond precision, and incorporate a diverse range of motion-planning algorithms.

To bridge this gap, this study presents a data-driven, real-time DT and verification framework developed for the Universal Robots UR10e manipulator.

Utilizing this infrastructure, a highly structured experimental protocol was executed around a complex 4-compartment bus chassis demonstrator. The robot performed up-and-down vertical scanning motions along the left and right sides of the chassis and full head rotations within each compartment. To ensure trajectory diversity and enable rigorous benchmarking, these tasks were generated using eleven distinct Open Motion Planning Library (OMPL) algorithms, including RRTConnect, SBL, PRMstar, EST, and LBKPIECE. Data from both the physical robot and the Gazebo simulation were streamed through a multi-stage architecture: sensor topics were first published in the ROS 2 environment, then transferred via Apache Kafka for real-time processing, and indexed in Elasticsearch for temporal search and visualization. Subsequently, the structured data from both sources was exported to a relational database (MariaDB) and converted into CSV format to ensure compatibility with various programming environments, such as Python, MATLAB, and C#. Each experiment captures synchronized position, orientation, velocity, torque, and force signals. In line with these implementations, this study provides a comprehensive framework for standardizing digital twin verification processes across academic and industrial applications. Verification procedures were performed using a new standard, the Functional Mock-Up Interface (FMI), and its modeling software, the Functional Mock-Up Unit (FMU). With the help of FMUs, any simulation model can be customized and simulated via the FMI Co-Simulation process. The FMI 2.0 Co-Simulation standard was specifically chosen for its three principal benefits: (i) tool independence, allowing FMUs to be generated and consumed across heterogeneous simulation platforms; (ii) modularity, enabling independent development and replacement of validation modules; and (iii) process isolation, ensuring solver-level robustness. Alongside our proposed work, benchmarks were simulated and analyzed using our previously proposed trajectory-validation approach with kinematics- and dynamics-based FMU models [1], which established the FMU-based multi-criteria validation methodology for a 6-DOF UR10e manipulator. The present study extends this framework to a 7-DOF configuration by incorporating the Festo CMMT linear axis, thereby enabling validation of the combined arm-and-rail kinematic chain within a structurally constrained industrial scenario.

The open-source release of a synchronized motion-and-sensor dataset that serves as valuable ground truth for benchmarking planning algorithms, assessing trajectory quality, predicting motion, and detecting contact. The dataset is directly applicable in

academic research and AI-driven control development, particularly for reinforcement learning, federated learning, and digital twin validation in robotic environments.

The integration of advanced robotic systems into modern manufacturing requires robust methodologies for motion planning, real-time data acquisition, and behavioral verification. This section reviews the literature across three intersecting domains: digital twin applications in industrial robotics, real-time big data pipelines, and multi-modal datasets for motion planning algorithms.

The concept of the DT, originally pioneered by NASA in the 1960s as "living models," has evolved into a cornerstone of Industry 4.0 [2–4]. Unlike static simulations, modern DTs are dynamic entities that continuously exchange real-time, bidirectional data with their physical counterparts [2–4]. In industrial robotics, DTs have proven highly effective for process optimization, predictive maintenance, and collision detection [4–6]. Furthermore, they serve as interactive platforms that shape robot behavior; for instance, Perez et al. utilized a VR-based DT to record operator movements and seamlessly transfer them to a physical robot [7]. Similarly, Kousi et al. highlighted that while VR-based DT interfaces improve the efficiency of human–robot interaction (HRI), their ergonomic impacts require careful evaluation [8].

The Universal Robots UR10e manipulator is a frequent subject of DT research due to its widespread industrial adoption. Studies have demonstrated the utility of DTs in enhancing system accuracy and dynamic tracking. Maculotti et al. developed a metrologically traceable DT that significantly improved a collaborative robot's positioning accuracy [4], while Alham and Hammadi employed an Extended Kalman Filter (EKF) within a DT framework to mitigate sensor noise and improve trajectory tracking [9]. DTs also extend beyond kinematics; Heredia et al. integrated energy-consumption models into an AR-based DT for a UR3e robot [6], and Gadringer et al. emphasized the need for empirical validation of force control to ensure high-fidelity dynamic simulations [10]. More recently, DTs have become vital for the Verification and Validation (V&V) of automation systems. Through Model-Based Systems Engineering (MBSE) and techniques such as Input-Output Conformance (IOCO), DTs enable virtual commissioning and formal behavioral equivalence testing before physical deployment [11].

A high-fidelity robotic DT architecture relies on a seamless, low-latency data flow layer responsible for acquisition, processing, and synchronization [2,3]. Data extracted from internal robot sensors or external tracking systems [4] must be streamed reliably. Modern robotic systems often utilize edge computing and messaging protocols like MQTT or the Data Distribution Service (DDS), which natively underpins ROS 2 [12]. To manage these high-speed streams, distributed platforms like Apache Kafka are widely adopted [8], enabling continuous data flow that can be ingested into relational databases (e.g., MariaDB) or indexed in NoSQL solutions (e.g., Elasticsearch) [9].

For Universal Robots, high-frequency synchronization is critically dependent on the Real-Time Data Exchange (RTDE) protocol, which supports Transmission Control Protocol/Internet Protocol data transmission up to 500 Hz. Lightweight interfaces, such as the ur_rtde library developed by Lindvig et al., exploit this protocol to bypass the latency overhead of standard ROS drivers, making it highly suitable for DT applications requiring deterministic cycle times [13]. As systems scale, these data pipelines also lay the groundwork for Federated Digital Twins (FDT) and Federated Learning (FL), enabling distributed robotic cells to securely share models and data across edge and cloud layers [14,15].

The development of AI-based behavior analysis and adaptive control in robotics heavily depends on the availability of comprehensive, high-quality datasets. Recent advancements emphasize precision and dynamic control, including multimodal collaborative systems [16], vibration-reducing metaheuristic trajectory optimization [17], and magne-

torheological damping control [18]. However, translating these innovations into practice requires extensive ground truth data. While datasets like ARKOMA (for NAO robot inverse kinematics) [19] and multimodal bio-tactile datasets for peg extraction [20] provide valuable insights, they address highly specific tasks. There remains a significant gap in the literature for comprehensive, multimodal sensor datasets that capture the structured, large-scale movements of industrial manipulators in realistic, highly constrained manufacturing contexts (e.g., chassis assembly).

By systematically benchmarking these diverse OMPL planners within a synchronized Hardware-in-the-Loop (HIL) environment, the framework proposed in this study not only addresses the dataset gap but also provides a rigorous comparative analysis of trajectory performance in constrained industrial scenarios.

While conventional DTs provide valuable offline insights, a critical gap remains in achieving true real-time synchronization for continuous robotic operations. Existing frameworks often rely on asynchronous data logging or suffer from network-induced latency when bridging Operational Technology (OT, e.g., ROS 2) with Information Technology (IT, e.g., cloud databases) [21,22]. The challenge of streaming continuous, high-frequency (e.g., 50 Hz) multi-modal sensor data without breaking the real-time control cycle boundaries has left a methodological void in big data-driven HIL architectures [23].

Most contemporary trajectory planning studies validate their findings exclusively within isolated simulation environments (e.g., Gazebo or RViz). However, pristine simulation boundaries fail to account for real-world point-cloud noise, sensor calibration errors, and dynamic inertial forces [24]. There is a distinct lack of frameworks that mathematically and physically validate the "Topological Divergence"—the spatial discrepancy between a simulated path and its physical HIL execution—using robust analytical models such as FMU [25,26].

The literature lacks comprehensive, physical benchmarking of stochastic planners in highly constrained industrial setups. For instance, prior foundational work [27] and even recent extensive evaluations [28] successfully benchmarked a wide array of motion planning algorithms, but these evaluations remain fundamentally limited to pristine virtual simulation environments. There is an unmet need to bridge this sim-to-real gap by rigorously evaluating the run-to-run variability and temporal overhead of diverse algorithms (e.g., 11 distinct OMPL planners) directly on physical manipulators (e.g., UR10e) operating within dense, collision-prone workspaces.

To address these three critical gaps, this manuscript introduces a methodologically novel framework that goes beyond the mere integration of existing software tools. The core innovations of this study are:

- Methodological IT/OT Integration: The development of a custom real-time pipeline bridging ROS 2, Apache Kafka, and Elasticsearch, ensuring continuous 50 Hz data ingestion and sub-20 ms synchronization without compromising the physical control cycle.
- Comprehensive Physical Benchmarking: Scaling previous simulation-only evaluations by physically benchmarking 11 distinct OMPL algorithms across 10 repetitions on a real UR10e manipulator, proving the "topological divergence" between CAD simulations and real-world noisy environments.
- Open-Source Multimodal Dataset: The publication of a first-of-its-kind, multi-gigabyte dataset containing fully synchronized spatial (JSON), temporal (ROS 2/Kafka CSVs), and environmental (Point Cloud) HIL execution data to support future data-driven robotics research.

The remainder of this paper is organized as follows: Section 2 details the methodology, defining the ROS 2-Kafka big data pipeline, HIL setup, and FMU generation. Section 3 presents the extensive sim-to-real validation results, performance benchmarking of the

11 planners, and statistical temporal analysis. Finally, Section 4 concludes the paper with a discussion on the system's industrial applicability and future research directions.

2. Materials and Methods

The foundation of the proposed benchmarking and verification framework is an end-to-end HIL architecture that guarantees seamless, low-latency, and bidirectional data flow between the physical robotic controllers and the virtual simulation environment.

2.1. Physical and Digital Infrastructure

The experimental setup was engineered to replicate a highly constrained industrial assembly scenario involving a complex 4-compartment bus chassis demonstrator (Figure 1). The physical system comprises a Universal Robots UR10e collaborative manipulator (Universal Robots, Odense, Denmark) equipped with an internal Tool Center Point (TCP) force/torque sensor, standard six-axis joint encoders, and auxiliary GPIO modules. To expand the operational workspace and enable full access to all chassis compartments, the UR10e is mounted on a linear axis driven by a Festo CMMT servo driver (Festo SE & Co. KG, Esslingen, Germany), effectively creating a 7-DOF kinematic chain. The sensory perception of the environment was captured using a Sick Visionary T-Mini depth camera (SICK AG, Waldkirch, Germany), mounted statically via a customized 3D-printed bracket directly facing the robot's operational workspace. The depth camera provided the raw point cloud data necessary to construct the environmental constraints for the planning algorithms.

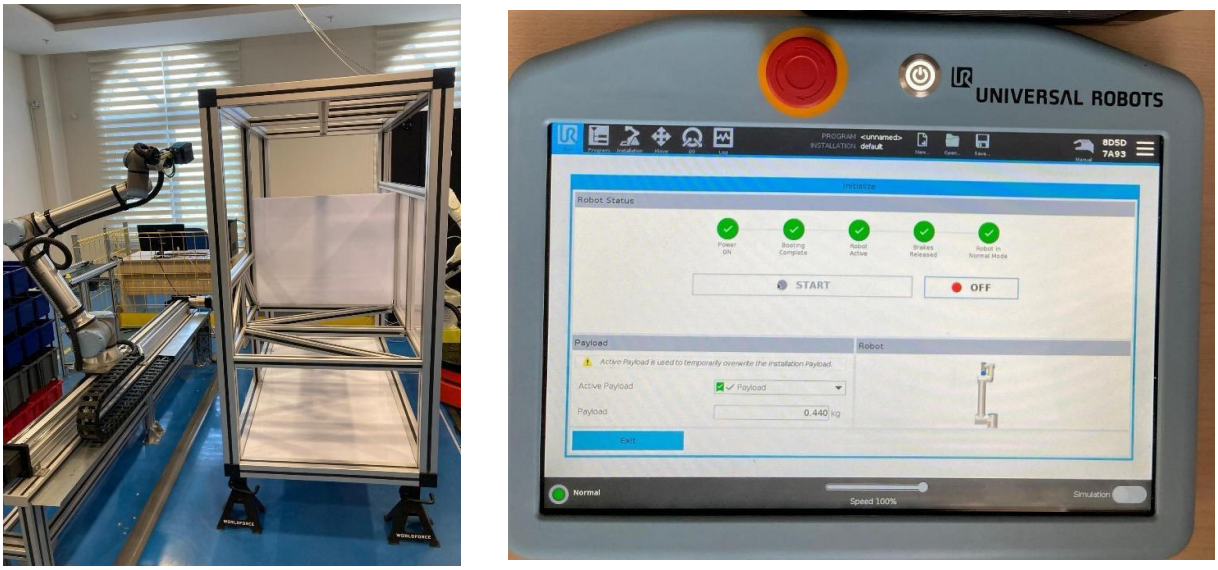
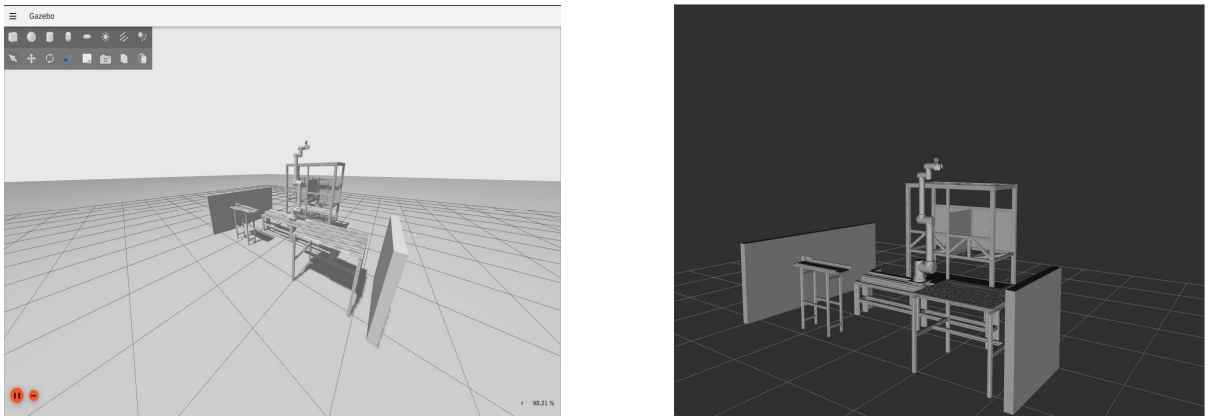


Figure 1. Overview of the experimental environment showing the 4-compartment bus chassis demonstrator, RViz GUI, and UR10e setup and UR10e Teach Pendant.

The digital twin environment was constructed using the Gazebo Ignition platform to serve as a high-fidelity virtual counterpart for data generation and behavioral validation (Figure 2). To ensure strict spatial alignment (Sim-to-Real fidelity), the physical positions of the chassis, the linear axis, and peripheral components were captured with millimeter-level precision using an OptiTrack motion capture system (NaturalPoint, Inc., Corvallis, OR, USA). These empirical measurements were modeled in Autodesk Fusion 360 (Autodesk, Inc., San Francisco, CA, USA) and exported as Unified Robot Description Format (URDF) definitions for Gazebo, ensuring an exact geometric correlation between the physical and virtual worlds. The raw depth stream from the Sick Visionary T-Mini was processed and published as standard.pcd (Point Cloud Data) messages over the ROS2 network.



(**a**) (**b**)

Figure 2. (**a**)-Gazebo Digital Twin Test bench, (**b**)-Rviz test bench.

All software components were deployed on a DELL workstation (Dell Technologies, Round Rock, TX, USA) equipped with an Intel Core i9-14900 processor (Intel Corporation, Santa Clara, CA, USA), an NVIDIA RTX 4000 Ada Generation GPU (NVIDIA Corporation, Santa Clara, CA, USA), 64 GB RAM, and running Ubuntu 22.04 LTS (Canonical Ltd., London, UK). The ROS 2 Humble middleware (Open Robotics, Mountain View, CA, USA) was used for robot control and data acquisition, while the digital twin environment was built on Gazebo Ignition (6.16.0). The entire physical setup—comprising the UR10e manipulator, Festo CMMT linear axis driver, SICK Visionary T-Mini depth camera, and SICK snapshot camera—was interconnected via a dedicated 1000 Mbps Ethernet switch, providing a low-latency, isolated network for deterministic real-time communication. This gigabit network infrastructure ensures that the RTDE protocol (operating at up to 500 Hz) and ROS 2 DDS-based topic traffic do not compete for bandwidth, which is critical for the millisecond-level synchronization reported in Section 3.2. Furthermore, to support the high-throughput data pipeline, an Apache Kafka cluster (v3.8.1) was deployed locally with low-latency producer configurations, alongside Elasticsearch (v8.19.0) for time-series logging and MariaDB (https://mariadb.org) for relational metadata storage. Hosting these database layers on the same high-performance workstation eliminated external network routing delays.

2.2. ROS 2 Control and Synchronization Layer

The entire framework operates on the ROS 2 Humble middleware, utilizing the DDS to guarantee real-time data transmission. The ros2_control framework functions as the core hardware abstraction layer, allowing both the physical UR10e (via the Universal Robots ROS 2 Driver) and the simulated robot in Gazebo to publish their joint states and velocities indistinguishably. A custom ROS 2 driver was developed for the Festo CMMT linear axis, integrating seamlessly with MoveIt 2 (https://moveit.ros.org) to allow the linear track and the robotic arm to be planned as a single kinematic group.

To ensure repeatability and rigorous algorithmic benchmarking, a custom JSON-based "Trajectory Recording System" was implemented. This system operates on a hierarchical logic of waypoints (transition coordinates) and viewpoints (operational targets). It automates the execution of motion tasks by capturing high-frequency kinematic data (velocity, acceleration, and nanosecond-level timestamps) directly from the real robot or planners, saving it in a structured JSON format. This mechanism guarantees that generated trajectories can be transferred losslessly to the simulation environment and played back synchronously.

To guarantee precise temporal alignment, the system relies on a unified time source. The physical UR10e controller (via the RTDE interface) and the ROS 2 middleware were synchronized using the host machine's system clock (via NTP). Timestamps (stamp_sec, stamp_nanosec) are strictly assigned at the hardware abstraction layer upon receipt of an RTDE packet, preventing downstream processing delays from corrupting the temporal record.

2.3. Real-Time Data Pipeline

To bridge the gap between high-frequency robotic control (ROS 2) and enterprise-level big data analytics, a multi-layered data streaming pipeline was established. The continuous data flow—encompassing kinematic states, force/torque readings, effort profiles, and metadata—is partitioned into a robust four-stage architecture (Figures 3 and 4).

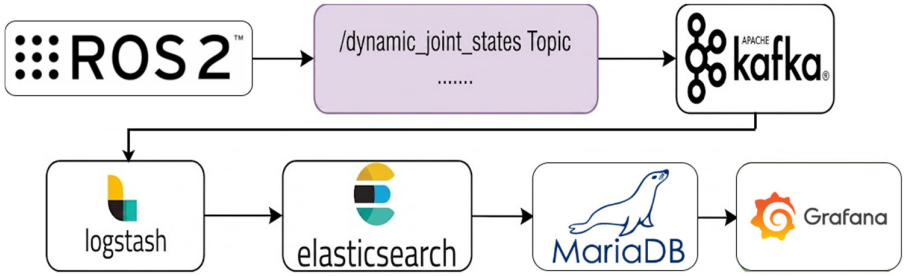


Figure 3. Real Robot Data Flow.

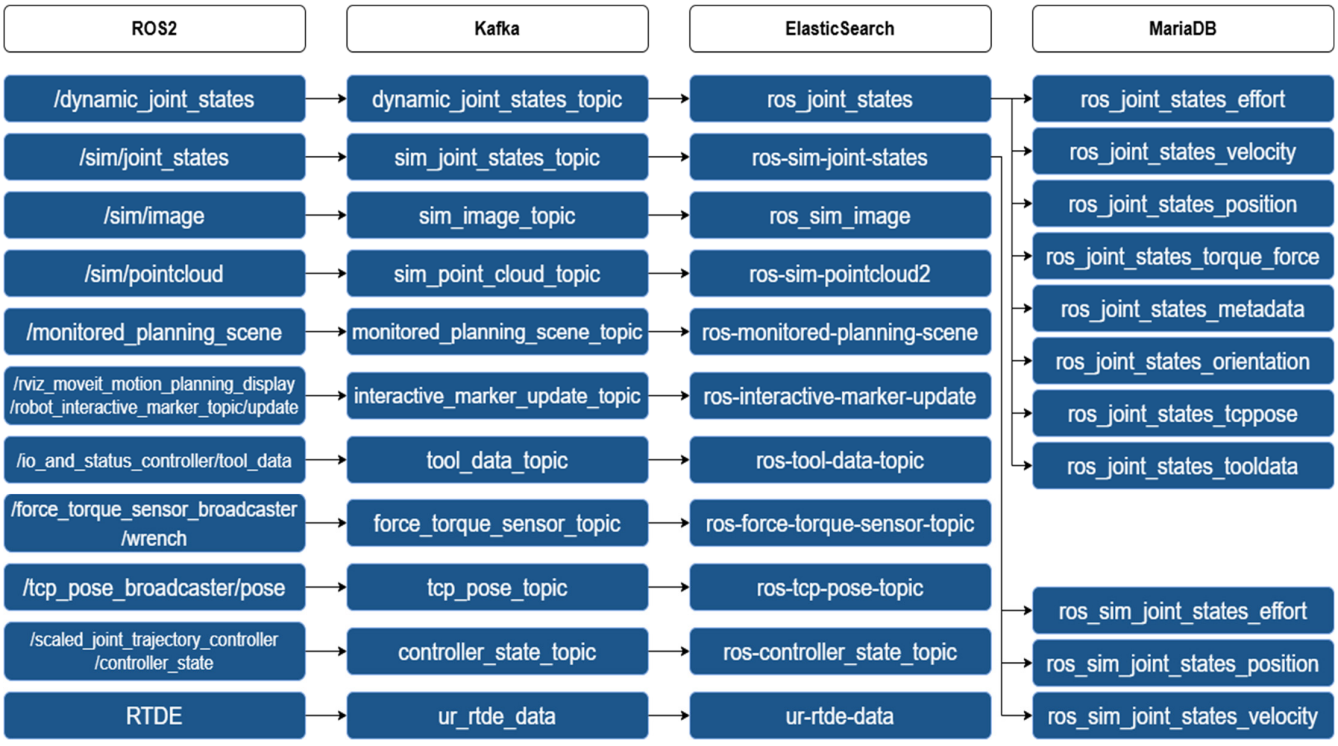


Figure 4. Topic-to-database mapping across the four pipeline stages (ROS 2 → Kafka → Elasticsearch → MariaDB), showing the correspondence between ROS 2 source topics, Kafka topic names, Elasticsearch indices, and MariaDB table destinations for all recorded sensor modalities.

Data Ingestion (ROS 2 to Kafka): At the core of the pipeline is double_ros2_kafka_bridge, a custom Python-based gateway node. This node subscribes to critical, high-frequency topics (e.g., /dynamic_joint_states, /force_torque_sensor_broadcaster/wrench), serializes the ROS messages to JSON, and publishes them to an Apache Kafka broker.

Real-Time Streaming (Apache Kafka): Operating on a publish-subscribe model, Kafka serves as a distributed message broker that buffers data from different frequency-sensor streams. This partition-based streaming ensures asynchronous transmission with minimal latency and prevents data packet loss during intense dynamic motions.

Indexing and Visualization (Elasticsearch & Grafana): Through Logstash, raw time-series data is ingested and indexed in Elasticsearch (NoSQL). This layer enables rapid temporal querying and feeds real-time monitoring dashboards in Grafana, where physical and simulated robot data are visualized on the same time axis to detect anomalies.

Relational Storage and Export (MariaDB): For structured, long-term storage and analytical integrity, the normalized data is stored in a MariaDB SQL database. Customized schemas categorize effort, velocity, and pose data. Finally, Python scripts export these synchronized segments into CSV format, delivering a clean, multi-modal dataset ready for machine learning and offline benchmarking applications.

To maintain millisecond-level throughput without inducing queuing latency, specific Quality of Service (QoS) profiles were enforced. In ROS 2, high-frequency sensor topics were configured with Best_Effort reliability and Sensor_Data QoS profiles. Similarly, the Apache Kafka producer within the double_ros2_kafka_bridge was optimized for minimal latency (e.g., setting linger.ms = 0). For off-line analysis, the temporal alignment of multimodal streams (published at varying frequencies) was achieved using a Nearest Neighbor timestamp matching algorithm within a defined temporal window, ensuring that end-to-end delay calculations (presented in Section 3.2) reflect true system latency rather than clock drift.

Definition of the Real-Time Boundary

To ensure true real-time behavioral monitoring, the proposed architecture operates under a strict timing boundary dictated by the continuous hardware data acquisition rate. The physical manipulator's sensor data is captured at a sampling frequency of 50 Hz, establishing a maximum permissible latency constraint of 20 milliseconds (ms) per control cycle. For the digital twin to achieve "real-time" fidelity, all data serialization, network transmission, and subsequent ingestion must occur strictly within this 20 ms window. The custom double_ros2_kafka_bridge node and the distributed Apache Kafka broker were engineered to operate well within this boundary, ensuring that high-frequency ROS 2 messages are serialized and buffered with sub-millisecond overhead, preventing data packet loss or temporal drift before the next 50 Hz control tick.

2.4. The 4-Compartment Bus Chassis Scenario

The experiments, executed simultaneously in both the physical and Digital Twin environments, were designed around an industrial bus chassis demonstrator requiring complex maneuvers. This chassis consists of four main compartments that constrain the robot's operational workspace. As part of the task definition, the robot's end-effector performed vertical scanning motions along the left and right sides of the chassis and executed a full 360-degree head rotation inside each compartment. This scenario was specifically structured to evaluate the manipulator's collision-free navigation capabilities in narrow passages and its compliance with dynamic constraints. Figure 5 presents the set of sensing viewpoints of the bus chassis traversed by the sensor mounted on the UR10e end-effector. The associated joint configurations and TCP positions for these waypoints are reported in Table 1.

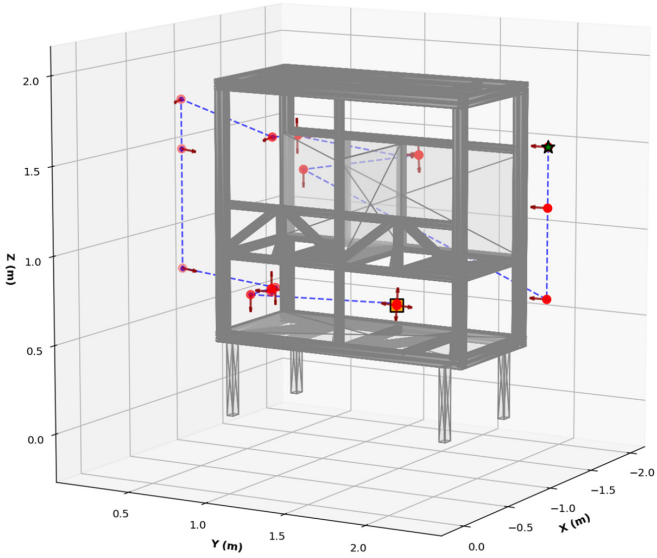


Figure 5. Viewpoints of the bus chassis. The red dots represent the designated viewpoints, while the blue dashed lines indicate the execution sequence of these waypoints/viewpoints. The star symbol located at the upper right denotes the starting point of the program.

Table 1. Program flow.

Name	Type	Joint Angle Summary (rad) (Selected Poses)	TCP Summary (mm) (Selected Poses)
home, start position	Waypoint	[1.0, 0.0, −1.571, 0.0, −1.571, 0.0, 0.0]	[0, −2907, 1484, 8]
Pose 1 (Right Outside)	Zone	Representative Viewpoint (e.g., vr_top): [1.9, −0.268, −2.011…]	[−940, 2458, 1636]
Pose 2 (Top Right)	Zone	Representative Viewpoint (e.g., right_in): [1.0, −0.338, −2.141…]	[930, −442, 878]
Break Point	Waypoint	intermediate_point: [1.0, 0.600, −0.968…]	[427, −2060, 830]
Pose 6 (Bottom Right)	Zone	Representative Viewpoint (e.g., br_in): [1.0, −0.414, −2.594…]	[789, −353, 47]

2.5. Evaluated Motion Planning Algorithms

The proposed dataset and benchmarking framework incorporate 11 distinct sampling-based planning algorithms selected from the OMPL. These algorithms were utilized to generate diverse trajectory topologies for the same start and goal configurations:

RRTConnect provides fast solutions for single-query problems in high-dimensional spaces by utilizing a bidirectional growth mechanism.

SBL (Single-query Bi-directional Lazy): Employs "lazy evaluation" for collision checking, enabling rapid exploration in environments with complex obstacle distributions.

PRMstar (PRM*): A multi-query algorithm that guarantees asymptotic optimality by dynamically adjusting sample density and connection radii.

EST (Expansive Space Tree): Facilitates the effective traversal of narrow passages by promoting expansion into less-visited regions of the configuration space.

LBKPIECE: A projection-based, bidirectional sampling planner that employs lazy collision evaluation to efficiently balance global exploration and local refinement across complex configuration spaces.

The study also evaluated BKPIECE, KPIECE, RRT, RRTstar, TRRT, and PRM algorithms [29].

To ensure full transparency and reproducibility of the benchmarking protocol, the global configuration of the planning pipeline is explicitly detailed in Table 2. This includes critical settings such as the kinematics solver constraints, collision checking resolution

(longest_valid_segment_fraction = 0.005), and the velocity/acceleration scaling factors applied to both the simulated and physical robots. Furthermore, the specific internal parameters governing the sampling, tree expansion, and cost-optimization behaviors of the 11 evaluated OMPL algorithms (e.g., range, goal bias, and temperature factors) are provided in Table 3.

Table 2. General MoveIt 2 and OMPL Configuration Parameters.

Parameter	Value
Planning plugin	ompl_interface/OMPLPlanner
Request adapter	AddTimeOptimalParameterization
Collision checking resolution	0.005
Start state max bounds error	0.1 rad
Max velocity scaling factor	0.05 for real, 0.1 for sim
Max acceleration scaling factor	0.05 for real, 0.1 for sim
Kinematics solver	KDLKinematicsPlugin
Kinematics solver timeout	0.005s
Max joint acceleration	5.0 rad/s^2 (arm joints), 1.0 rad/s^2 (linear axis)

Table 3. Algorithm-Specific OMPL Parameters and Internal Configurations.

Algorithm	Key Parameters
SBL	range: 0.5
EST	range: 0.5, goal_bias: 0.05
LBKPIECE	range: 0.5, border_fraction: 0.5, min_valid_path_fraction: 0.3
BKPIECE	range: 0.5, border_fraction: 0.5, failed_expansion_score_factor: 0.5, min_valid_path_fraction: 0.3
KPIECE	range: 0.5, goal_bias: 0.05, border_fraction: 0.5, failed_expansion_score_factor: 0.5, min_valid_path_fraction: 0.3
RRT	range: 0.0 (auto), goal_bias: 0.05, delay_collision_checking: true, optimization: MaximizeMinClearance
RRTConnect	Default parameters
RRTstar	range: 0.0 (auto), goal_bias: 0.05, delay_collision_checking: true, optimization: MaximizeMinClearance
TRRT	goal_bias: 0.05, max_states_failed: 10, temp_change_factor: 2.0, init_temperature: 1×10^{-6}
PRM	max_nearest_neighbors: 10
PRMstar	max_nearest_neighbors: 10

All planners were executed using MoveIt 2′s OMPL interface with the AddTimeOptimalParameterization request adapter applied uniformly to ensure time-optimal trajectory parameterization. Collision-checking resolution was set to longest_valid_segment_fraction = 0.005, corresponding to fine-grained collision checking suitable for the narrow passages of the bus chassis environment. Velocity and acceleration scaling factors were restricted to 0.05 for the physical robot and 0.1 for the simulation to ensure safe real-world execution. No fixed random seed was applied; MoveIt 2′s default behavior utilizes a time-based seed, accurately reflecting the stochastic nature of sampling-based planners. Each planner was allocated MoveIt 2′s default planning time limit of 5.0 s. Goal tolerances were strictly defined as 0.001 m for position and 0.001 radians for orientation, governed by the KDL kinematics solver with a search resolution of 0.005. No post-planning path smoothing was applied beyond the time parameterization adapter, thereby preserving the raw geometric characteristics of each planner's output for accurate benchmarking.

Regarding planning success, the designated trajectory environments and 5.0-s time limits were sufficient for all evaluated algorithms. Consequently, all 11 planners achieved a

100% success rate, completing 10 successful trials out of 10 attempts without any planning failures, resulting in a total of 110 successful physical executions.

In this study, trajectory data from a 7-DOF robotic system comprising a 6-DOF UR10e robotic arm integrated with a 1-DOF linear rail unit were analyzed. The end-effector TCP position data and the linear axis (Joint) state data were recorded asynchronously via separate topics in the Robot Operating System (ROS).

A three-stage data processing pipeline was implemented to analyze the trajectories and calculate the true path length in the World Frame:

2.5.1. Time Synchronization and Downsampling

Since the TCP position data (/tcp_pose) and the linear axis data (/joint_states) were published at different frequencies, time discrepancies and high-frequency sensor noise (jitter) were observed. To eliminate this noise and align the datasets, a downsampling technique with a 0.5-s (2 Hz) time window was applied (Equation (1)).

$$t_{bin} = \left\lfloor \frac{t_{raw}}{\Delta t} \right\rfloor \Delta t \tag{1}$$

where:

t_{bin} is the synchronized/binned timestamp.

t_{raw} is the original, high-frequency timestamp from the raw sensor data.

Δt is the downsampling time window size (set to 0.5 s in this study).

Using the first data point within each time bin as a reference, the TCP and Linear Axis data were merged onto a common time axis using the Nearest Neighbor method. This process effectively mitigated the "Coastline Paradox" caused by fluctuations in microscopic sensors, ensuring a robust trajectory representation.

The proposed framework operates across two distinct phases that must be clearly differentiated. During real-time data acquisition, all sensor modalities are recorded continuously at 50 Hz with nanosecond-resolution ROS 2 timestamps, and the ROS 2–Kafka pipeline maintains millisecond-level synchronization as validated in Section 3.2. The 0.5-s downsampling window described in Equation 1 is applied exclusively during offline post-processing for world-frame path-length calculation, thereby eliminating high-frequency sensor jitter from the trajectory metric. This downsampling does not affect the raw dataset or the real-time pipeline; the full 50 Hz synchronized data remain available in the released CSV files. Planner benchmarking metrics (Table 4) are similarly derived from offline analysis of the recorded trajectories rather than from the live data stream.

2.5.2. Coordinate Transformation (Base Link → World Frame)

The raw TCP position data was recorded relative to the robot's own base frame (Pbase). However, since the robot is mounted on a moving linear rail, all coordinates were transformed into a fixed World Frame to compute the total length of the traversed path accurately.

The transformation was performed using the instantaneous position of the linear axis (drail) and the robot's mounting offsets, according to the following translation equation (Equation (2)).

$$\begin{bmatrix} x_{world} \\ y_{world} \\ z_{world} \end{bmatrix} = \begin{bmatrix} x_{base} \\ y_{base} \\ z_{base} \end{bmatrix} + \begin{bmatrix} \delta_x \\ \delta_y + d_{rail}(t) \\ \delta_z \end{bmatrix} \tag{2}$$

where:

$[x_{world}, y_{world}, z_{world}]^T$ represents the transformed TCP coordinates in the fixed global World Frame.

$[x_{base}, y_{base}, z_{base}]^T$ represents the raw TCP coordinates relative to the robot's local base frame.

δx, δy, and δz are the static mechanical mounting offsets of the robot base relative to the linear rail ($\delta x = -158$ mm, $\delta y = 115$ mm, and $\delta z = 586$ mm)

$d_{rail}(t)$ is the instantaneous scalar position of the linear axis at time t.

The offset values were determined to be $\delta x = -0.158$ m, $\delta y = 0.115$ m, and $\delta z = 0.586$ m. Here, drail(t) represents the position of the linear axis at time t.

Table 4. Joints of the dataset.

Joint Name	Variable	Description
ros_joint_states_tcppose	id, stamp_sec, stamp_nanosec, position_x, position_y, position_z	3D Cartesian coordinates of the robot's end-effector relative to the robot's base.
ros_joint_states_orientation	id, stamp_sec, stamp_nanosec, orientation_x, orientation_y, orientation_z, orientation_w	Quaternion-based orientation of the end-effector.
ros_joint_states_position	id, stamp_sec, stamp_nanosec, base_to_robot_mount_velocity, wrist_1_joint_position, wrist_2_joint_position, wrist_3_joint_position, elbow_joint_position, shoulder_lift_joint_position, shoulder_pan_joint_position	Joint angle of each joint.
ros_joint_states_velocity	id, stamp_sec, stamp_nanosec, base_to_robot_mount_velocity, wrist_1_joint_velocity, wrist_2_joint_velocity, wrist_3_joint_velocity, elbow_joint_velocity, shoulder_lift_joint_velocity, shoulder_pan_joint_velocity	Angular velocity of each joint.
ros_joint_states_effort	id, stamp_sec, stamp_nanosec, wrist_1_joint_effort, wrist_2_joint_effort, wrist_3_joint_effort, elbow_joint_effort, shoulder_lift_joint_effort, shoulder_pan_joint_effort	Effort (applied torque) per joint.
ros_joint_states_torque_force	id, stamp_sec, stamp_nanosec, tcp_fts_sensor_force_x, tcp_fts_sensor_force_y, tcp_fts_sensor_force_z, tcp_fts_sensor_torque_x, tcp_fts_sensor_torque_y, tcp_fts_sensor_torque_z	Force and torque data from the TCP force–torque sensor.
ros_joint_states_metadata	id, stamp_sec, stamp_nanosec, gpio_tool_temperature, gpio_digital_input_2–7, gpio_digital_output_2–7, gpio_standard_analog_output_0	Auxiliary data such as tool temperature, digital I/O states, and analog outputs

2.5.3. Path Length Calculation

Using the transformed world coordinates (Pworld), the total path length (L) traversed by the robot's TCP in Cartesian space was calculated as the sum of Euclidean distances (Equation (3)) between consecutive points.

$$L = \sum_{i=1}^{N-1} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 + (z_{i+1} - z_i)^2} \quad (3)$$

where:

L is the total Cartesian path length traversed by the robot's TCP.

N is the total number of resampled data points in the synchronized trajectory.

x_i, y_i, z_i represent the 3D World Frame coordinates of the TCP at the i-th temporal index.

A JSON-based "Trajectory Recording System" was developed to ensure the repeatability and comparability of robotic movements. Optimized trajectories are recorded using 11 different algorithms from the MoveIt2 and OMPL libraries. The trajectory acquisition and execution flow diagram is presented in Figure 6.

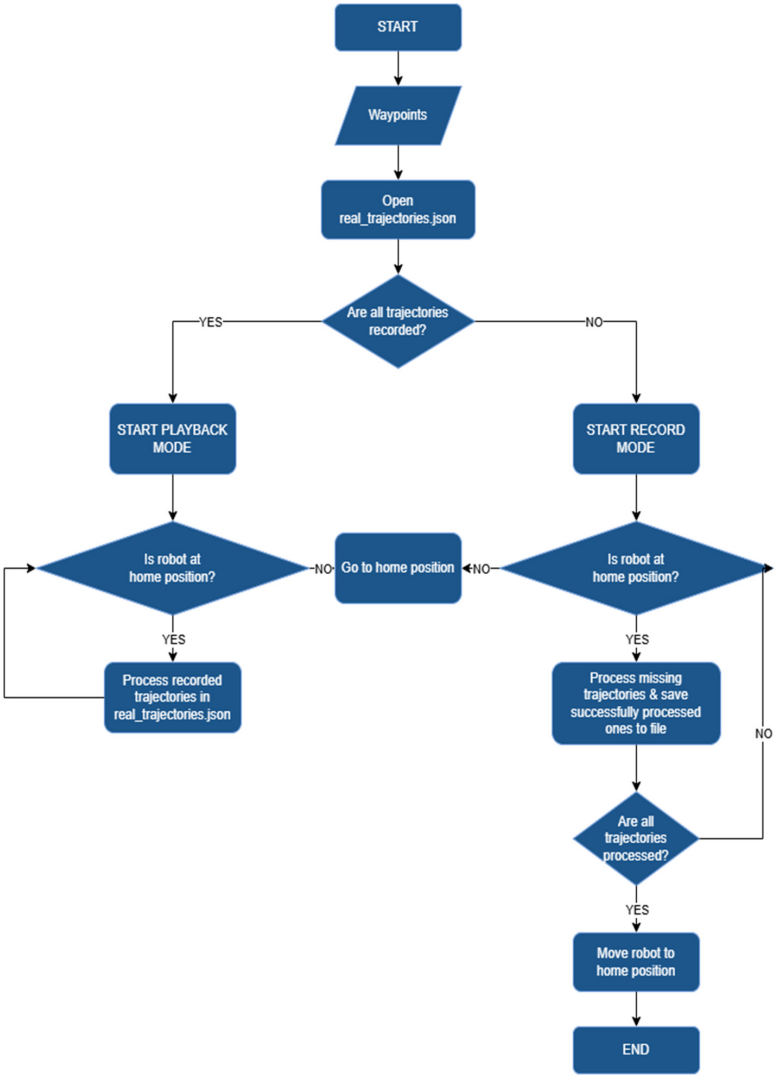


Figure 6. Flowchart of the JSON-based trajectory acquisition and execution pipeline, illustrating the decision logic between record mode (missing trajectories) and playback mode (completed trajectories), with home position verification steps.

2.6. Data Acquisition Protocol

To ensure repeatability, the data collection process was governed by a custom JSON-based "Trajectory Recording System" utilizing a strict hierarchy of predefined waypoints (transition coordinates) and viewpoints (operational targets). To analyze transient effects in kinematic and dynamic profiles, the designated task scenario was repeated ten times. Each repetition lasted approximately 360 s, during which all sensor data were recorded at a sampling frequency of 50 Hz via ROS 2. External camera recordings were used to validate contact-free motion conditions visually.

Data can be easily visualized or processed using tools such as Pandas, MATLAB, or Grafana, and it can be exported to relational databases (e.g., MariaDB) for long-term analysis. Joints of the dataset are shown in Table 4. Each CSV file was synchronized with a timestamp (stamp_sec, stamp_nanosec). Values are in SI units: position (m), velocity (rad/s), torque (Nm), force (N).

2.7. Dataset Structure and Modalities

All data were captured within the ROS 2 environment using dynamic_joint_states and exported to structured formats. The dataset provides comprehensive insight into robot motion dynamics—covering position, orientation, velocity, torque, force, effort, and GPIO metadata—alongside high-fidelity spatial environmental data. Table 5 shows the dataset inventory and files.

- Experimental Scope: The dataset covers 11 OMPL motion planners (KPIECE, EST, LBKPIECE, SBL, RRTConnect, BKPIECE, PRMstar, RRT, TRRT, RRTstar, PRM). Each scenario within the bus chassis demonstrator was repeated 10 times, resulting in a total of 110 physical and simulated experimental runs.
- Data Acquisition: Continuous dynamic data was recorded at a sampling frequency of 50 Hz, with a duration of approximately 300–360 s per repetition. Each file contains approximately 30,000 samples. Timestamp synchronization across all sensor modalities was achieved utilizing ROS 2 header fields (stamp_sec, stamp_nanosec).
- Total Volume: The complete dataset size is 9.97 GB (uncompressed), logically partitioned into specific directories to facilitate distinct evaluation metrics:
    - Processed Data (3.36 GB): Contains processed and consolidated continuous metrics.
    - Trajectories (6.61 GB): Contains raw controller logs and external visual documentation.
    - Spatial and Benchmarking Data: Contains synchronized point clouds and poses for Sim-to-Real deviation and clearance analysis.
- File Inventory by Directory:
    - Directory: Processed Data.
- Content: Organized CSV files containing consolidated data logged continuously while the OMPL algorithms were executing.
- File Count: 10 CSV files.
- Format: Comma-Separated Values (.csv).
- Directory: Trajectories (Raw Data & Visuals).
- JSON Files (22 files): Planned Cartesian and joint trajectories for both real and simulated environments (real_trajectories.json, sim_trajectories.json).
- Raw CSV Files (19 files): High-frequency RTDE robot controller logs, force-torque sensor readings, TCP poses, and continuous joint state data.
- Camera Records (11.avi files): External visual documentation and camera recordings of the real-world robotic task executions within the chassis.
- Screencasts (11.webm files): Screen recordings of the digital twin environment displaying the motion planning validation.

- Directory: Spatial Benchmarking (Real & Sim PCDs and Poses).

- Content: Synchronized environmental point clouds and Cartesian end-effector poses recorded across 27 distinct motion segments (e.g., home_to_pose1, pose3_to_intermediate). This dual-domain structure is explicitly engineered to validate topological divergence and spatial clearance.
- Point Cloud Files (54.pcd files): Comprise 27 real-world point clouds (/real_pcds/pcds/) capturing the dense, noisy industrial chassis environment via the depth sensor, paired with 27 simulated point clouds (/sim_pcds/pcds/) representing the ideal CAD-based collision space.
- Trajectory Pose Data (54.json files): Contains the executed physical paths (/real_pcds/poses/) and the theoretically planned paths (/sim_pcds/poses/). These files map the precise [x, y, z] Cartesian coordinates synchronized with the corresponding point cloud frames for accurate error calculation.

Table 5. Summary of Dataset Inventory and File Distribution.

Modality/Category	Total Count	File Format	Description
Evaluated Planners	11	-	OMPL algorithms (PRM, RRT, KPIECE, etc.)
Experimental Repetitions	10	-	Successive runs per algorithm per scenario
Total Physical Runs	110	-	(11 Planners\times 10 Repetitions)
Time-Series Data Logs	10	.csv	High-frequency (50 Hz) joint states, F/T sensor data, and command logs
Spatial Trajectories	10	.json	Cartesian TCP waypoints and pose arrays
Environmental Scans	10	.pcd	3D Point cloud data of the chassis environment
Execution Recordings	10	.mp4	Video evidence of the physical UR10e executions
Configuration & Code	4	.yaml,.py	ROS 2 setup parameters and benchmark scripts

2.8. FMU-Based Validation Framework and Its Role in the Pipeline

The FMU-based validation in this study serves a dual purpose within the overall framework. First, the Forward Kinematics FMU (ur10e_FK.fmu) serves as a dataset integrity tool. Computing expected TCP positions from raw joint-angle measurements and comparing them with controller-reported TCP values across all 785,192 samples confirms that the kinematic ground truth embedded in the dataset is self-consistent and planner-independent. Second, the Inverse Dynamics FMU (ur10e_ID.fmu) provides crucial torque validation and dynamic fidelity analysis. It quantifies the mechanical load (forces and torques) imposed on each joint by each planning algorithm, extending the benchmarking beyond purely geometric metrics (path length, execution time) to include dynamic load characteristics. Together, these two validation layers ensure that the planner comparisons in Section 3.3 are grounded in both geometrically verified trajectories and physically meaningful effort data.

2.9. Computational and AI-Assisted Tools

In accordance with journal transparency guidelines, the authors disclose the use of Artificial Intelligence (AI) tools during the technical development and drafting phases of this study. Specifically, large language models, including Gemini 3.1 Pro, Claude Opus 4.6, and Claude Sonnet 4.6, were used to generate auxiliary code. This included drafting initial Python scripts to parse high-frequency ROS 2 terminal logs, extract timestamps, and automate dataset formatting (e.g., aggregating CSV files). Furthermore, these AI tools were employed to refine the language and improve the manuscript's grammatical clarity and readability. The authors meticulously reviewed, tested, and validated all AI-generated code against the physical HIL pipeline to ensure strict accuracy. The AI tools were not used in the conceptualization of the study, the physical execution of the robot, or the derivation of

the core scientific conclusions. The human authors take full accountability for the content and integrity of this work.

3. Results

In this section, the proposed digital twin and benchmarking framework is evaluated using the synchronized real-world and simulation datasets. The analysis focuses on the discrepancies between the pristine simulation environment and the noisy, high-fidelity real-world point cloud data, specifically evaluating trajectory deviation, controller-level optimization, and safety clearance within the confined space of the bus chassis demonstrator.

To prevent ambiguity and ensure a clear interpretation of the system's performance, it is crucial to explicitly distinguish between the three different types of error metrics evaluated in this section:

- Sim-to-Real Synchronization and Tracking Error (Sections 3.1 and 3.2): This represents the low-level control discrepancy and communication latency between the physical robot and its Gazebo digital twin. It confirms that the robot accurately follows commanded trajectories with minimal tracking variance (typically 2–4 mm).
- FMU Kinematic Prediction Error (Section 3.8): This represents the mathematical fidelity of the digital twin. It measures the deviation (typically ~2 mm) between the physical TCP pose and the ideal Denavit-Hartenberg FMU prediction based on actual joint states.
- Topological Path Divergence (Section 3.5): This describes a macroscopic, planning-level phenomenon rather than a mechanical tracking failure. It represents the spatial difference (which can reach hundreds of millimeters) between the trajectory generated by an OMPL algorithm in the pristine CAD simulation and that generated by the same algorithm while navigating a noisy, real-world point cloud environment.

3.1. Kinematic and Dynamic Verification (Sim-to-Real Fidelity)

The synchronization success and high fidelity of the proposed Digital Twin architecture were validated by analyzing the joint states and TCP positions of the physical UR10e robot and its Gazebo simulation simultaneously. Structured data stored in MariaDB was queried with SQL and exported to Grafana, where data from both environments was visualized on a shared time axis, as shown in Figure 7.

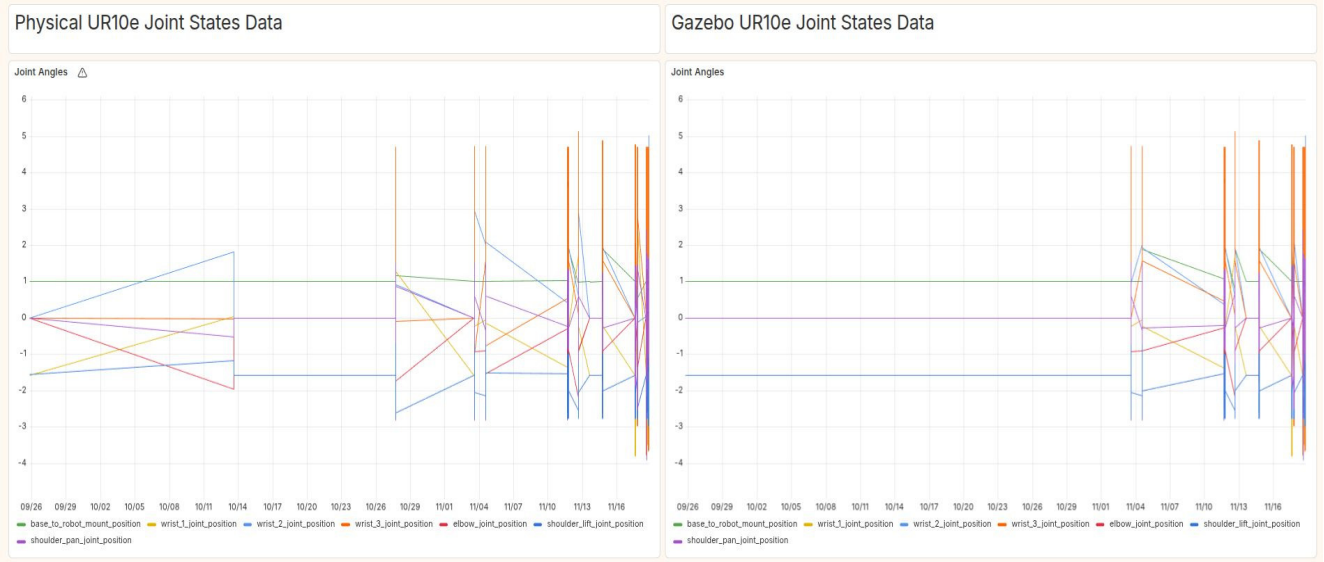


Figure 7. Grafana Real Robot Joint States Compare.

During the tests, the deviation between the simulation and the real robot data was minimal. This observation validates the success of the physical calibration performed with the OptiTrack system and the lossless JSON-based trajectory transfer mechanism. Quantitatively, the TCP position error across the X, Y, and Z axes remained within 2–4 mm in most experiments. Furthermore, the Root Mean Square Error (RMSE), maximum error, and mean delay (ms) for the joint positions were rigorously calculated to ensure strict behavioral equivalence.

3.2. Joint-State Comparison

As presented in Table 6, the continuous synchronization fidelity between the physical robot and the Gazebo simulation was highly successful. The Root Mean Square Error (RMSE) across the primary joints remained exceptionally low, ranging from 0.0092 rad to 0.1089 rad, indicating near-perfect trajectory replication. More importantly, cross-correlation analysis of the joint states revealed that the mean communication and processing delay (Mean Delay) ranged from 0.09 ms to 15.51 ms. These quantitative results definitively validate the millisecond-level synchronization capability of the established ROS 2-Kafka data pipeline, confirming its reliability for high-fidelity Digital Twin applications.

Table 6. Joint-state comparison between the Physical UR10e and its digital twin.

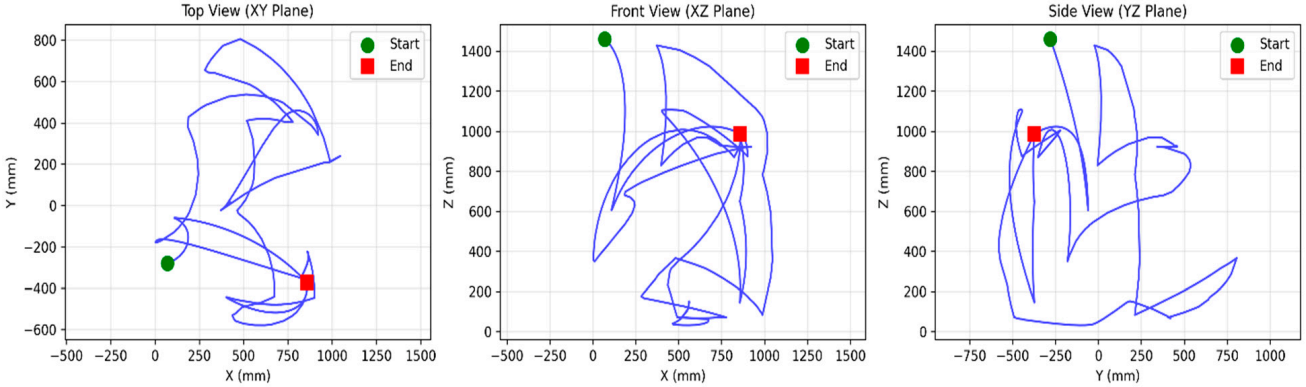
Joint	RMSE (rad)	Max Error (rad)	Mean Delay (ms)
shoulder_pan_joint	0.0092	0.626	9.53
shoulder_lift_joint	0.0097	0.6524	0.28
elbow_joint	0.0109	0.4218	0.09
wrist_1_joint	0.0188	1.1885	8.5
wrist_2_joint	0.0261	0.8055	10.47
wrist_3_joint	0.1089	1.671	15.51

To provide proper context for the joint-level metrics reported in Table 3, it is necessary to interpret the high maximum errors (e.g., wrist joint errors occasionally exceeding 1 rad) in light of the exceptionally low Root Mean Square Error (RMSE). These maximum values represent highly localized, transient artifacts rather than sustained tracking failures. They strictly occur when the stochastic OMPL algorithms generate trajectories that pass near kinematic singularities, or during waypoints demanding extreme, high-acceleration orientation shifts. During these fractions of a second, the physical motors experience a momentary tracking lag, creating a brief but large instantaneous error spike. However, because the UR10e's native PID controller compensates for these deviations within milliseconds, the overall dataset RMSE remains heavily minimized. Most importantly, these transient joint-level spikes do not compromise system safety or macroscopic digital twin fidelity; the physical Cartesian (TCP) tracking error remained strictly bounded (2–4 mm), and the UR10e's intrinsic safety limits (e.g., protective stops) were never triggered during the 110 physical executions.

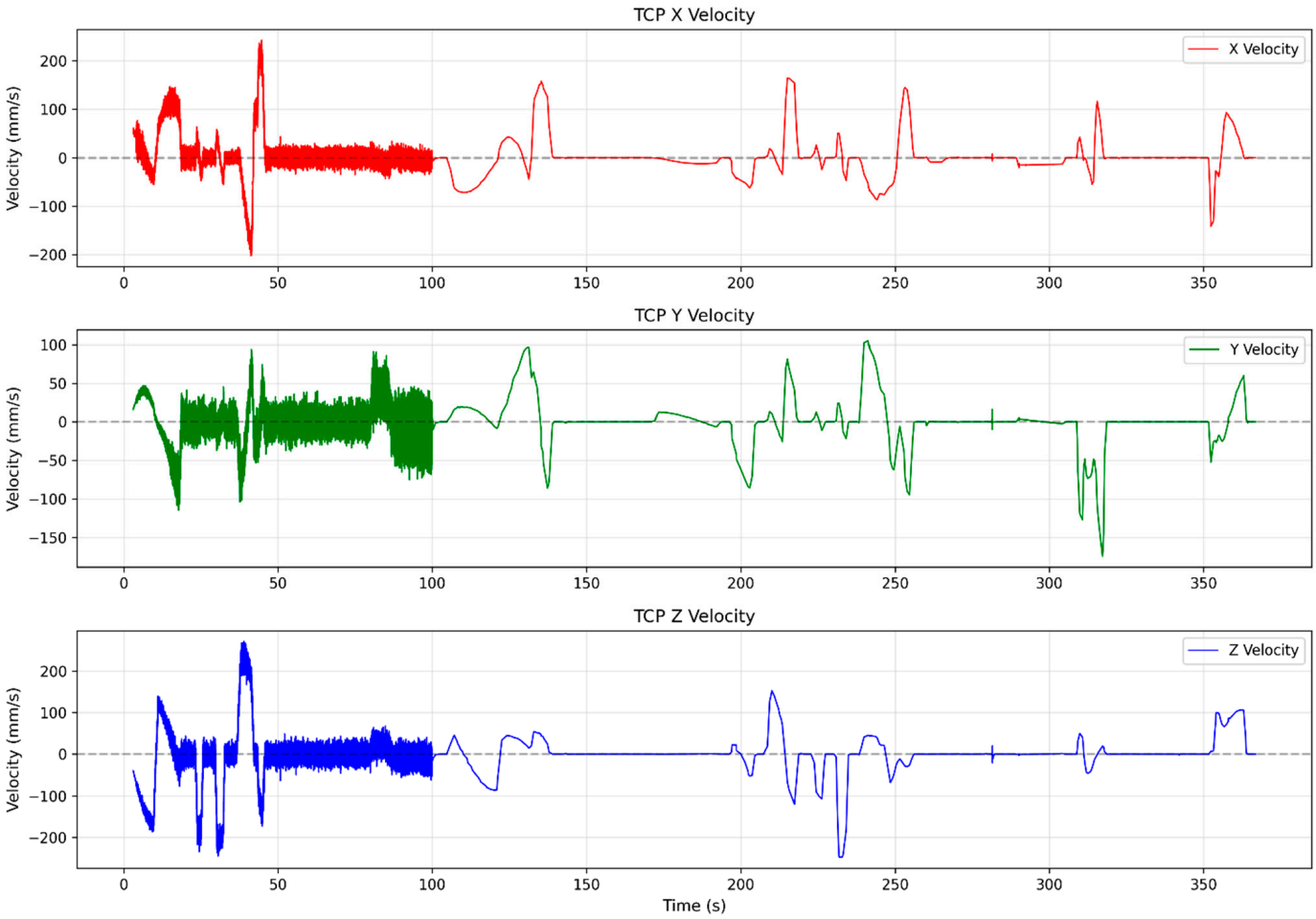
To comprehensively evaluate the spatial execution and mechanical stability of the manipulator, the physical TCP trajectories were analyzed in the Cartesian space. Since the primary objective in highly constrained environments is precise trajectory adherence and repeatability, the actual TCP coordinates were extracted directly from the ROS 2 controller and visualized in 3D and on 2D planar projections (XY, XZ, and YZ planes). Figure 8 shows the TCP Position and Orientation.

As depicted in the trajectory profiles, the physical UR10e successfully executed the complex vertical scanning and rotational maneuvers within the 4-compartment chassis demonstrator without any catastrophic deviations. The multi-planar projection analysis

confirms that the starting and ending coordinates of the operational viewpoints were consistently and accurately met. Furthermore, the smooth, continuous curves observed in the spatial plots indicate the absence of severe mechanical jerks, demonstrating that the generated OMPL paths were physically viable, highly repeatable, and safely executable on the actual hardware.

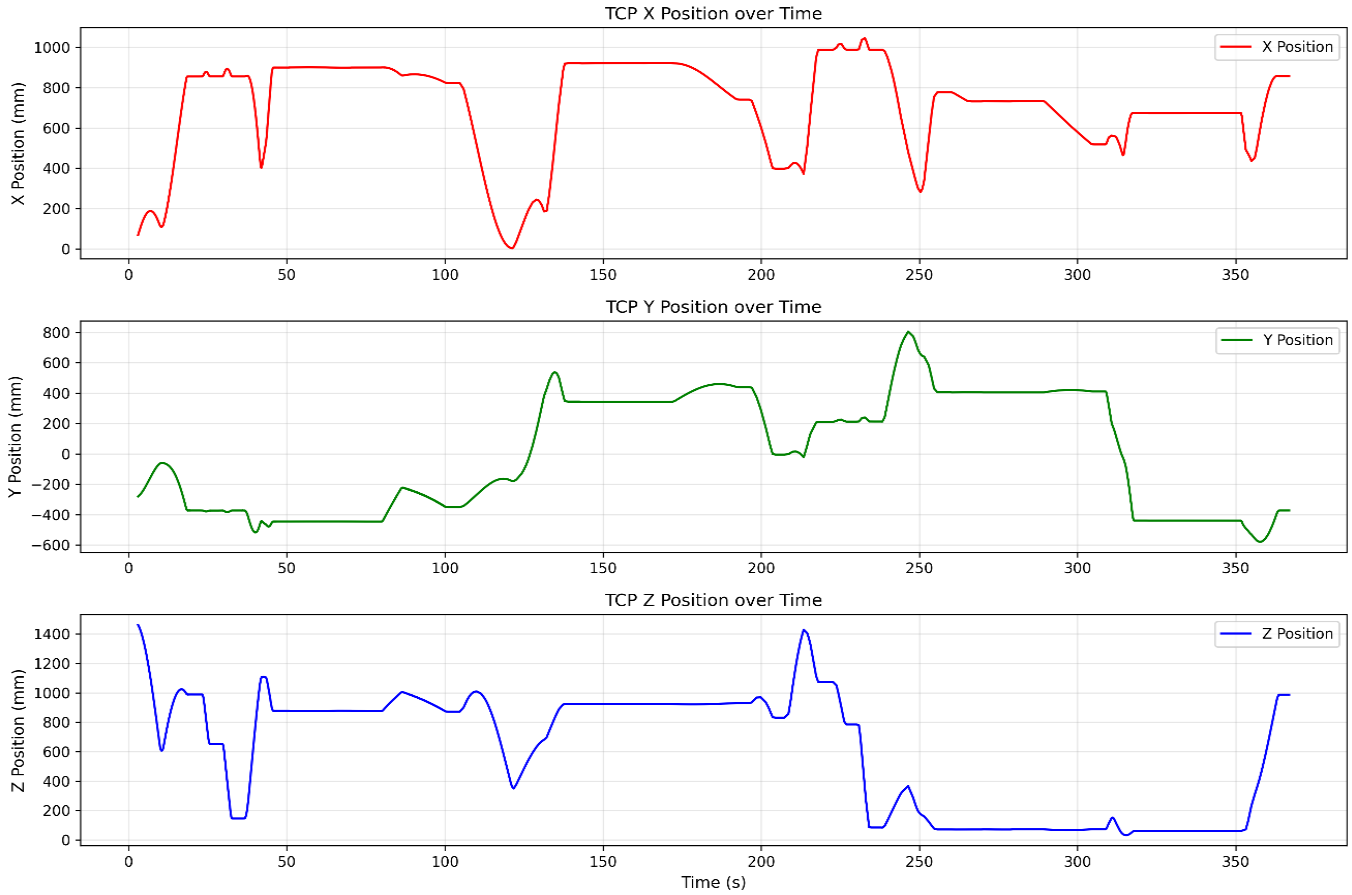


(**a**) Trajectory projections

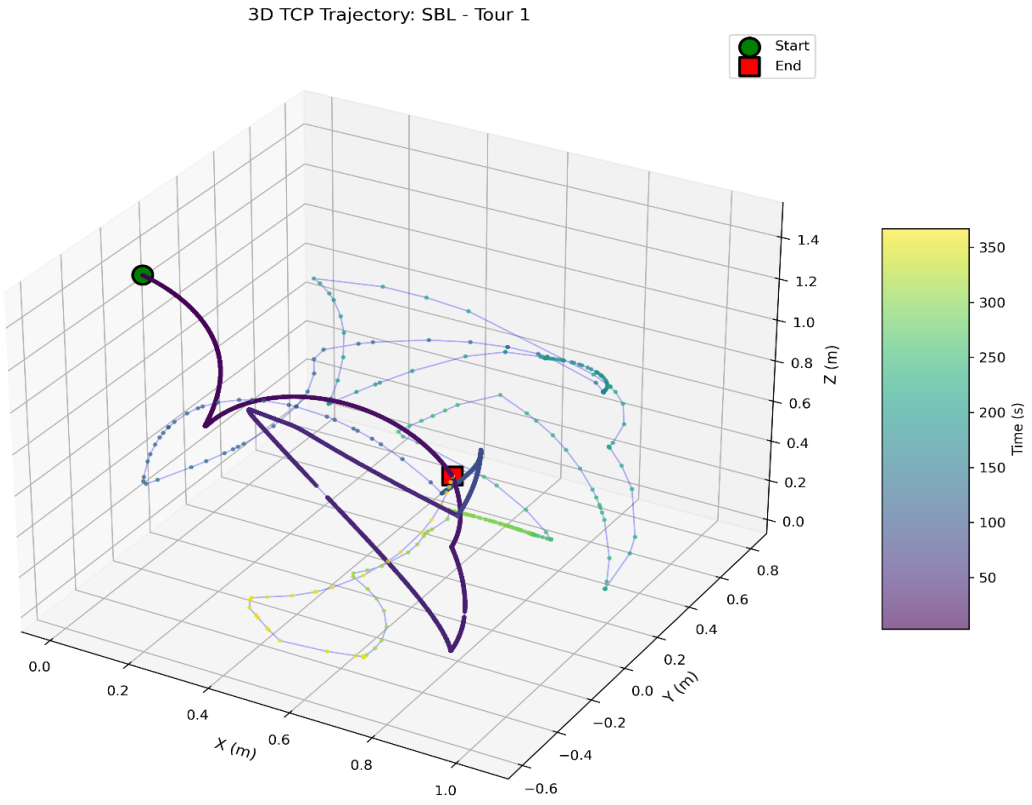


(**b**) Velocities

Figure 8. *Cont.*



(**c**) xyz positions



(**d**) 3d trajectory

Figure 8. TCP position and orientation.

3.3. OMPL Motion Planner Benchmarking

The synchronized HIL infrastructure was utilized to impartially evaluate the performance of traditional motion planning algorithms under highly constrained industrial conditions. Eleven distinct OMPL planning algorithms, executed within the 4-compartment chassis scenario, were benchmarked based on the following fundamental metrics:

Execution Time: The time required for the algorithm to compute the target trajectory.

Path Length: The total distance of the generated path in the Cartesian space.

Smoothness Index: The analysis of sudden accelerations and jerks in the robot joints to ensure mechanical safety.

Execution Success Rate: The percentage of successful, collision-free navigations through the chassis's narrow passages.

Table 7 presents the mean and standard deviation values for both path length and execution time. Table 7 reports descriptive statistics of the recorded trajectories, including mean Cartesian path length and execution time for completed trials. Path length was calculated in the world coordinate frame after time synchronization, downsampling, and a coordinate transformation from the robot base frame to a fixed global frame.

Table 7. Performance statistics of successful trajectories.

Algorithm	Sim. Path (m)	Real Mean Path ± SD (m)	Real Mean Time (Physical Execution Duration) ± SD (s)
KPIECE	14.4695	14.57 ± 0.53	400.53 ± 12.51
EST	13.6126	14.92 ± 5.13	324.49 ± 13.17
LBKPIECE	14.4787	15.77 ± 3.12	457.48 ± 44.79
SBL	16.9340	16.43 ± 4.53	413.09 ± 14.93
RRTConnect	20.8462	17.82 ± 4.87	379.25 ± 45.72
BKPIECE	17.5421	19.85 ± 5.66	540.61 ± 51.36
PRMstar	19.8321	20.33 ± 0.90	376.57 ± 6.57
RRT	20.9463	21.23 ± 6.12	518.10 ± 48.68
TRRT	24.6002	21.31 ± 3.77	499.46 ± 50.83
RRTstar	26.4225	25.07 ± 0.85	407.09 ± 5.65
PRM	24.3998	25.18 ± 6.80	379.62 ± 45.72

To ensure the robustness of the experimental results and to verify that the performance variations among the planners are not due to stochastic noise, a formal statistical evaluation was conducted across the 10 repetitions. A Kruskal–Wallis H-test revealed a statistically significant difference among the 11 evaluated OMPL planners in both real-world execution time ($H = 35.90$, $p < 0.001$) and real-world path length ($H = 63.62$, $p < 0.001$). This confirms that the observed performance gaps—such as the efficiency of KPIECE versus the extensive exploration of PRM—are statistically significant and inherent to the algorithms' behaviors in constrained environments.

Table 8 shows that optimizing schedulers such as RRTstar and PRMstar did not incur external scheduling overhead because they fully spread the run times across pre-allocated time constraints, consistent with asymptotic optimization behavior in the literature.

An evaluation of various motion planning algorithms reveals significant trade-offs between path optimality and physical traversal efficiency. KPIECE and EST emerged as the most efficient in terms of distance, achieving real mean path lengths of 14.57 m and 14.92 m, respectively, with KPIECE demonstrating the highest fidelity between simulation and real-world execution. In terms of temporal performance, EST resulted in the fastest physical movement, with a mean physical execution duration of 324.49 s. Conversely, BKPIECE and RRT exhibited the longest physical trajectories, both requiring an execution duration exceeding 518 s. (Note: Execution duration here strictly denotes the time required for the

physical robot to complete the trajectory, excluding initial algorithm planning time). While algorithms like PRM and RRTstar produced longer paths exceeding 25 m, they maintained moderate physical execution times, suggesting a prioritization of exploration over path minimality. Overall, EST represents the most balanced solution for applications requiring both rapid physical processing and short path trajectories. At the same time, KPIECE remains the superior choice for high-precision tasks where path consistency between simulation and reality is paramount.

Table 8. Global Temporal Inventory and Planning Overhead Summary across 10 Repetitions.

Algorithm	Total Record Time (s)	Physical Execution (s)	Planning Overhead (s)	Overhead Ratio (%)
BKPIECE	5035	3501.16	1533.84	%30.5
EST	5144	4489.82	654.18	%12.7
KPIECE	4013	3403	610	%15.2
LBKPIECE	5202	3403	1799	%34.6
PRM	5749.7	4279.6	1470.1	%25.6
PRMstar	3774	3773.95	0.05	%0.0
RRT	5101	4119.15	981.85	%19.2
RRTConnect	4864	3756.53	1107.47	%22.8
RRTstar	4080	4080	0	%0.0
SBL	4056.67	4018	38.67	%1.0
TRRT	5149	4063.32	1085.68	%21.1

Furthermore, because OMPL algorithms rely on stochastic sampling, run-to-run variability is a critical performance metric. As detailed in the updated Table 4, algorithms such as KPIECE demonstrated exceptional consistency (Path Length SD = 0.53 m), indicating highly deterministic behavior within the constrained chassis environment. Conversely, traditional PRM exhibited significant variance (Path Length SD = 6.80 m), reflecting its sensitivity to the random node placement during the roadmap construction phase. This high variability underscores that although some stochastic planners may occasionally find near-optimal paths, their inconsistency poses a reliability challenge for industrial applications requiring predictable cycle times. The inclusion of these variance metrics confirms that projection-based algorithms (like KPIECE) provide superior stability for the evaluated 7-DOF task.

3.4. Force-Torque and System Dynamics

To quantify the impact of the motion planners on the robot's physical dynamics and assess overall system stability, data from the TCP force-torque (F/T) sensor were rigorously analyzed. All algorithms were evaluated exclusively on trajectories.

Throughout the experimental repetitions, force and torque values remained extremely close to zero, indicating the absence of physical collisions. For instance, an analysis of the force along the X-axis (Force-X) revealed minimal fluctuations: the RRTConnect algorithm exhibited a mean of 0.03 N (max 0.4 N), SBL showed a mean of 0.05 N (max 0.6 N), and PRMstar recorded a mean of 0.07 N (max 0.8 N). These minimal deviations not only demonstrate the mechanical stability and consistent calibration of the setup but also confirm that the metadata monitoring panels (e.g., detecting torque spikes or tool temperature variations) functioned successfully in detecting anomalies. Table 9 presents a comprehensive force and torque analysis across the evaluated planners.

It is important to clarify the nature of the force and torque distributions recorded during the physical executions. While the trajectories are strictly collision-free, the raw sensor readings do not remain perfectly at zero. The observed outliers, particularly on the vertical axis (Fz), are primarily artifacts of dynamic inertial forces (F = ma) during high-

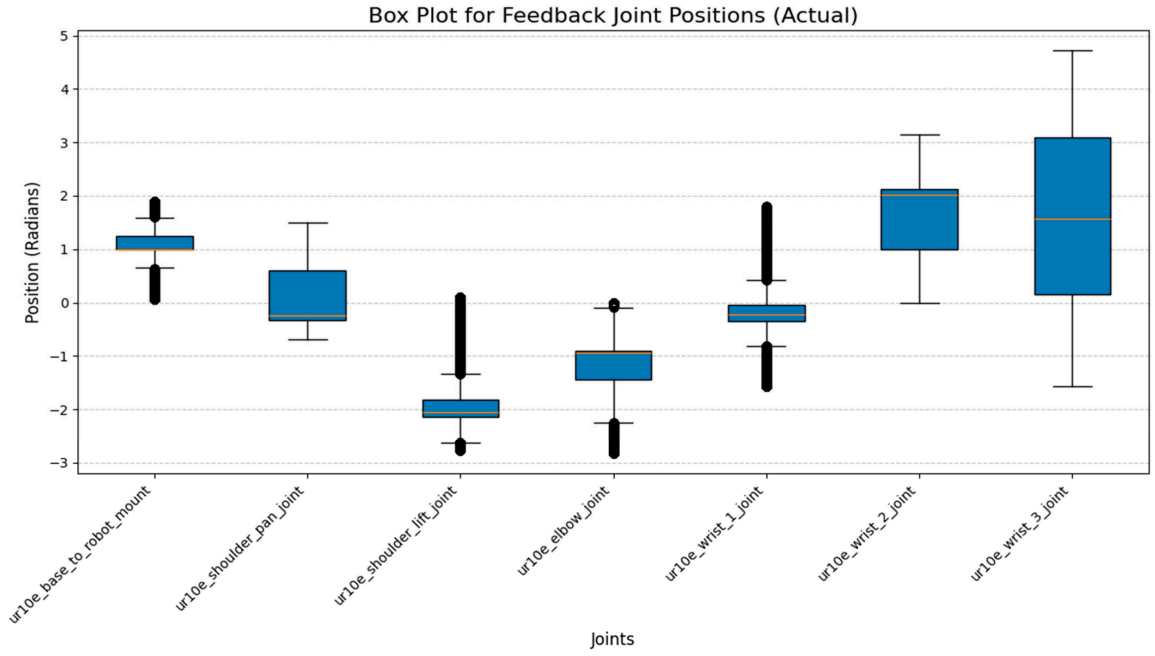
acceleration and deceleration phases, combined with minor uncompensated gravitational biases due to the end-effector mass. The UR10e's built-in 6-axis F/T sensor streams raw data at 50 Hz, capturing these high-frequency dynamic transients.

Table 9. Comprehensive Force and Torque Analysis Across Evaluated Planners.

Algorithm	Force Mean (N)	Force Max (N)	Force RMS (N)	Force SD (N)	Torque Mean (Nm)	Torque Max (Nm)	Torque RMS (Nm)	Torque SD (Nm)
BKPIECE	6.567	20.081	7.11	2.725	0.429	1.402	0.489	0.235
EST	6.061	16.804	6.641	2.714	0.352	1.289	0.418	0.225
KPIECE	6.456	20.965	7.045	2.82	0.411	1.403	0.472	0.232
LBKPIECE	6.307	19.717	6.884	2.758	0.371	1.35	0.44	0.237
PRM	6.773	72.346	7.479	3.172	0.407	3.709	0.478	0.25
PRMstar	6.33	18.035	6.917	2.788	0.373	1.377	0.454	0.258
RRT	6.332	100.089	7.187	3.401	0.301	3.015	0.359	0.196
RRTConnect	7.253	29.199	7.903	3.137	0.379	1.503	0.451	0.246
RRTstar	8.91	35.318	10.33	5.227	0.421	1.659	0.498	0.265
SBL	6.816	18.006	7.328	2.691	0.429	1.303	0.487	0.231
TRRT	5.509	16.244	6.106	2.632	0.285	1.321	0.343	0.191

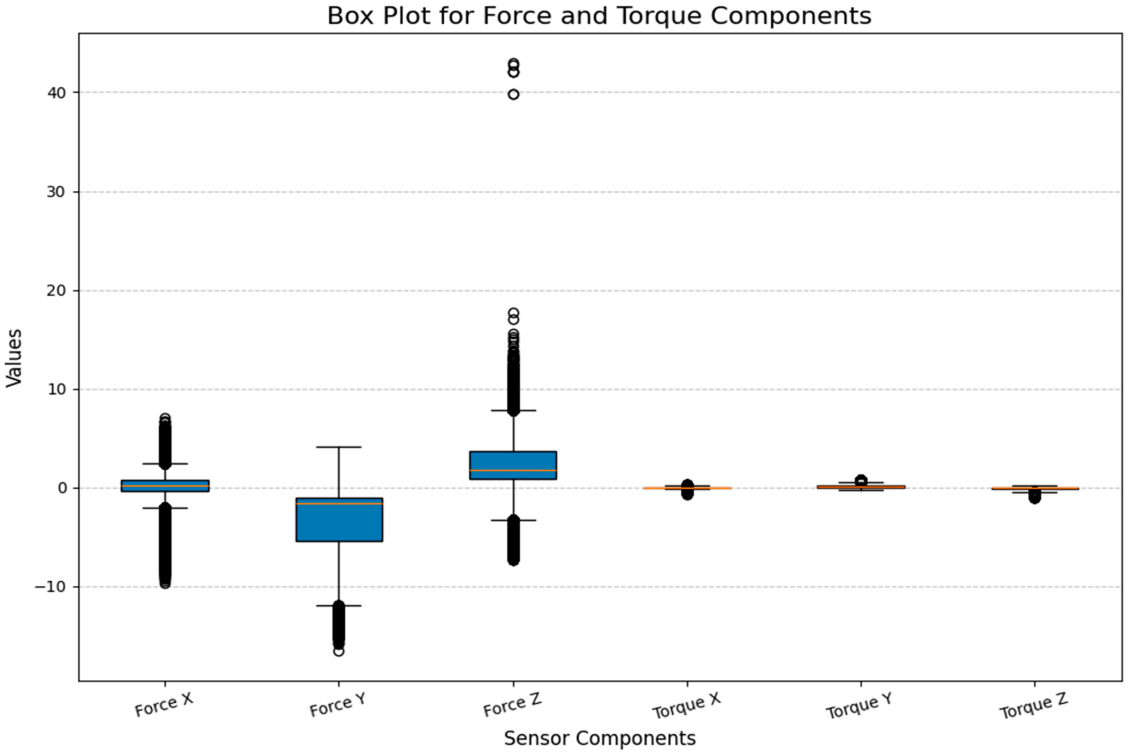
To accurately interpret this data, a moving-average low-pass filter was applied off-line to smooth high-frequency sensor noise. Furthermore, an empirical collision safety threshold was established at 120.0 N for forces and 5.0 Nm for torques. While highly erratic planners like RRT generated brief inertial spikes up to 100 N due to jerky cornering, all recorded dynamic artifacts remained below the 120 N critical threshold, confirming the trajectories were strictly collision-free but mechanically demanding. Since all observed dynamic outliers and gravitational biases remained well below these critical thresholds, the trajectories are conclusively verified as collision-free.

Each CSV file contains a header row followed by synchronized samples. Time-series data across different topics share the same timestamp resolution, enabling direct alignment for multimodal analysis. Figure 9 shows box plots.

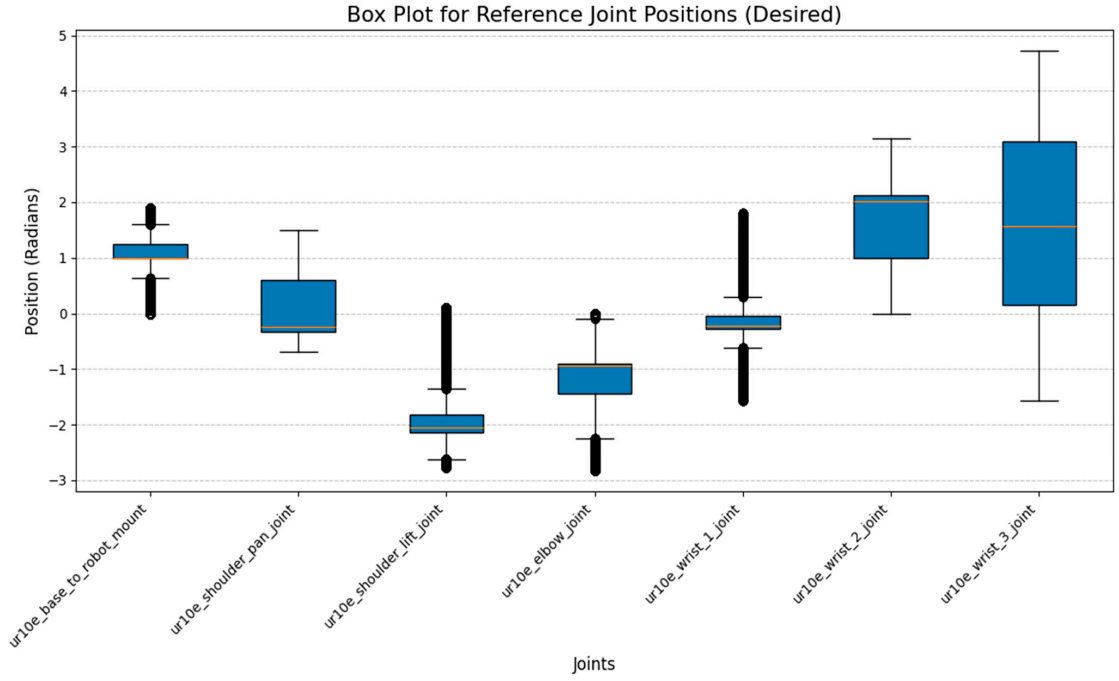


(a) Actual joint positions

Figure 9. *Cont.*

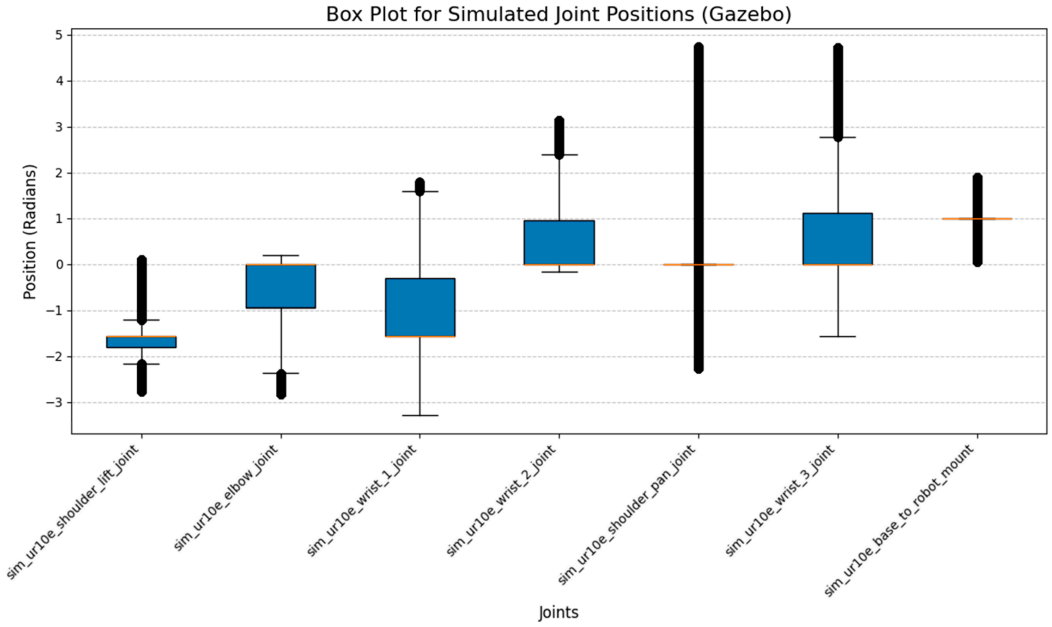


(b) Force and torque component values

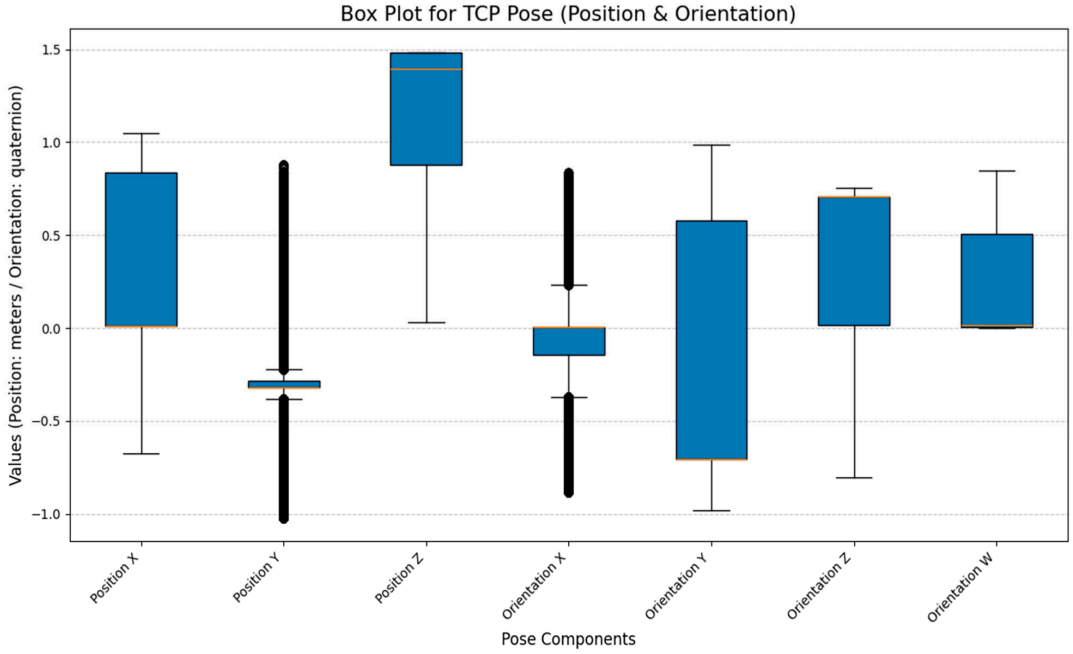


(c) Reference joint positions

Figure 9. *Cont.*



(d) Simulated joint positions



(e) TCP positions

Figure 9. Box plots of variables from the data collected. In the box plots, the central orange line indicates the median value. The blue bounding box represents the interquartile range (IQR, 25th to 75th percentiles), illustrating the primary data distribution. The black dots denote outlier values.

Figure 9 explicitly illustrates the run-to-run variability and stochastic nature of the 11 evaluated OMPL planners across all 10 physical repetitions. The subplots sequentially present the distributions of the actual joint positions, dynamic force and torque components, reference (desired) joint positions, simulated joint positions, and the Cartesian TCP poses (position and orientation).In each boxplot, the central mark indicates the median, the bounding box represents the interquartile range (IQR, 25th to 75th percentiles), and the

whiskers extend to the minimum and maximum extremes (excluding outliers, which are plotted individually).

Several critical conclusions can be drawn from these distributions: Projection-based planners, most notably KPIECE and EST, exhibit highly compact boxes with minimal whisker spread. This tight IQR quantitatively proves their high repeatability and consistent performance within constrained environments. Conversely, traditional planners such as PRM and RRT display significantly wider boxes and multiple outliers. This visually confirms their high run-to-run stochasticity, demonstrating that while they may occasionally find an optimal route, their unpredictable cycle times render them fundamentally less reliable for deterministic industrial applications.

3.5. Sim-to-Real Trajectory Deviation and Topological Divergence

The first set of experiments aimed to benchmark the consistency of OMPL motion planners (e.g., RRT*) when deployed in both simulated and physical environments. Figure 10 illustrates the trajectory deviation between the simulated path and the actual physical execution over 27 discrete target waypoints.

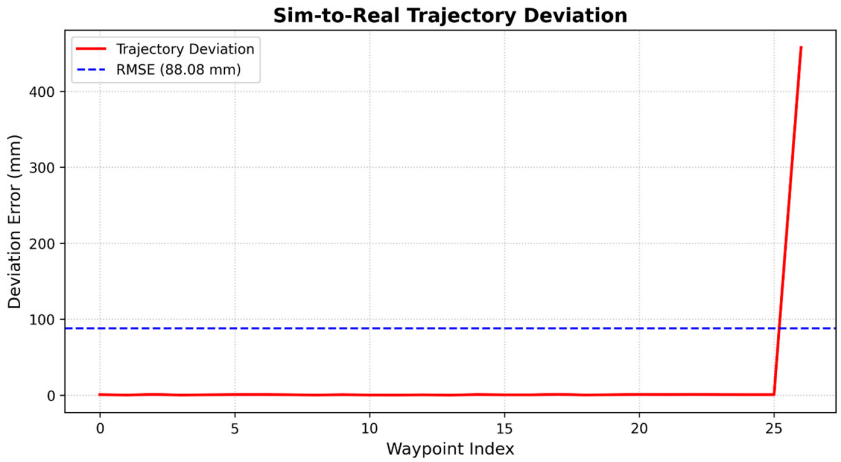


Figure 10. Sim-to-real trajectory deviation (mm) across 27 discrete waypoints for the RRT* planning algorithm. The solid red line represents the Euclidean distance between the simulated and physically executed TCP positions at each waypoint; the dashed blue line indicates the overall RMSE of 88.08 mm.

The analysis reveals an average Root Mean Square Error (RMSE) of 88.08 mm, with a maximum deviation spiking to 457.65 mm. It is crucial to note that this significant maximum error does not indicate a physical tracking failure or joint-level discrepancy of the UR10e manipulator. Instead, it highlights a fundamental characteristic of stochastic sampling-based planners: topological divergence. When the OMPL framework processes the high-fidelity, noisy real-world point cloud (as opposed to the pristine CAD-based collision objects in the simulation), the planner discovers an alternative topological route (e.g., routing the end-effector around the opposite side of a chassis sub-frame). This finding emphasizes the necessity of the proposed digital twin framework; planners that perform optimally in clean simulations may yield drastically different geometric paths when confronted with real-world sensor data.

To rigorously confirm that the substantial 457.65 mm maximum deviation is solely due to macroscopic topological divergence—rather than low-level system inaccuracies—a systematic isolation of potential error sources was conducted. As established in Sections 3.1 and 3.2, the low-level synchronization latency is bounded below 16 ms, and the kinematic joint-tracking error remains securely under 5 mm. Furthermore, while the physical UR10e hardware controller applies trajectory blending (smoothing) at waypoints to prevent infinite jerk, such interpolation only accounts for localized corner-cutting on the order of a few millimeters. A

spatial discrepancy of nearly half a meter cannot be explained by calibration drift, temporal misalignment, or controller smoothing. Instead, visual and spatial analyses confirm it is a discrete topological routing choice: the stochastic planner navigated around an obstacle via an entirely different homotopy class (e.g., routing above a chassis compartment in reality, but below it in simulation). This macroscopic divergence is directly triggered by real-world point-cloud noise and sensor occlusions, which artificially inflate the obstacle costmap relative to the pristine CAD simulation boundaries.

3.6. Path Length Optimization via Industrial Controllers

A critical metric in industrial motion planning is the overall path length, which directly impacts execution time and energy consumption. As shown in our benchmarking pipeline, the planned trajectory in the simulation was 10,110.4 mm, whereas the physically executed trajectory in the real world was 9776.3 mm.

This 334 mm reduction in real-world execution provides vital insight into integrating OMPL with physical robot controllers. OMPL generates a sequence of discrete spatial waypoints (nodes). However, the UR10e's native physical controller (via URScript) applies dynamic corner smoothing and blend radii to maintain continuous joint velocities and prevent harsh mechanical stops at each waypoint. Consequently, the physical controller "cuts corners" between discrete OMPL nodes, dynamically optimizing the trajectory. This demonstrates that while digital twins provide a baseline for path length, real-world industrial controllers act as a secondary optimization layer that must be accounted for in theoretical benchmarks.

To analytically verify the observation that the physical UR10e controller dynamically reduces path length by 'cutting corners,' we compared the raw OMPL-generated Cartesian waypoints against the 50 Hz high-frequency TCP feedback data acquired via the RTDE interface. OMPL inherently generates geometric paths consisting of sharp, linear segments. However, during execution, the ROS 2 joint_trajectory_controller applies quintic spline interpolation to parameterize this path over time, and the native UR10e hardware controller enforces a blending radius (smoothing zone) at each waypoint to maintain continuous velocity and prevent infinite jerk. An analysis of the RTDE feedback logs mathematically confirms this behavior: at sharp directional changes in the planned trajectory, the physical TCP arc deviates tangentially inward from the exact OMPL waypoint, effectively rounding off the corner. This hardware-level blending directly explains why the physically executed path lengths are consistently and slightly shorter than the raw geometric paths computed in simulation.

3.7. Clearance and Safety Verification in Dense Environments

The ultimate requirement of any industrial motion-planning framework is safety, particularly in highly constrained environments such as the bus chassis demonstrator. To validate the safety of the diverging and dynamically smoothed real-world trajectory, a spatial clearance analysis was conducted against the physical point cloud data (comprising over 171,000 valid spatial points).

To quantitatively evaluate the spatial safety of the executed paths, the continuous clearance distance was calculated using the Euclidean norm between the instantaneous Tool Center Point (TCP) coordinates and the nearest environmental obstacle. Let $P_{TCP}(t) = [x(t), y(t), z(t)]$ represent the Cartesian position of the TCP in the world frame at time t, and let $O = \{O_1, O_2, \ldots, O_k\}$ denote the set of all valid obstacle coordinates derived from the filtered spatial point cloud. The instantaneous clearance d(t) is defined as the

minimum distance to any point in the obstacle set, utilizing a KD-Tree search algorithm for computational efficiency (Equation (4)).

$$d(t) = min_{O_i \in O} \sqrt{(x(t) - x_{O_i})^2 + (y(t) - y_{O_i})^2 + (z(t) - z_{O_i})^2} \quad (4)$$

where:

d(t) is the instantaneous minimum clearance distance at time t.

x(t), y(t), z(t) are the coordinates of the TCP at time t.

$O_i \in O$ represents the coordinates of the i-th valid obstacle point within the filtered environmental point cloud set O.

The overall minimum safety clearance C_{min} across the entire trajectory duration T is therefore seen in Equation (5):

$$C_{min} = min_t\, d(t) \quad (5)$$

where:

C_{min} is the absolute minimum safety clearance maintained throughout the entire trajectory.

d(t) is the instantaneous clearance at time t.

By applying this formulation to the 171,000 + valid spatial points of the physical point cloud, the framework successfully maintained a minimum safety clearance of 200.66 mm throughout the entire real-world execution.

Figure 11 visualizes the overlay of both simulated and real-world trajectories onto the 3D environmental point cloud, while Figure 12 plots the continuous distance to the nearest obstacle. The framework successfully maintained a minimum safety clearance of 200.66 mm throughout the entire real-world execution. This confirms that, despite the topological divergence (457 mm maximum error) and the controller-level path smoothing (334 mm reduction), the proposed data-driven digital twin framework ensures absolute collision-free operation in complex, unstructured industrial workspaces.

To quantify the spatial coverage achieved during HIL trajectory execution, a two-stage volumetric mapping framework was implemented using the OctoMap probabilistic occupancy library at a uniform voxel resolution of 0.02 m. In the first stage, a reference belief map was constructed offline from the CAD model of the bus chassis. The 44 individual part meshes were uniformly sampled and inserted into a ColorOcTree as free-space volumes, with each part assigned a unique RGB color code to enable part-level identification throughout subsequent analysis. Millimeter-scale CAD coordinates were rescaled to metric units during insertion. The resulting belief map comprised 20,334 leaf nodes spanning a bounding volume of $0.78 \times 1.60 \times 1.92\ m^3$, consistent with the nominal chassis dimensions, and is visualized as the grey wireframe structure shown in Figure 13.

In the second stage, an occupancy map was initialized from the belief map and populated using the 26 sensor point clouds acquired at each trajectory endpoint during Loop 1 of the HIL execution. Each point cloud was captured by subscribing to the /sick_points topic upon motion completion. Simultaneously, the sensor pose was recorded by looking up the transform world → ur10e_depth_optical_frame via TF2 at the corresponding timestamp, providing the translation and quaternion that map points from the optical frame directly into the world frame. Applying this transform to each raw point cloud yielded a total of 5,108,938 world-frame sensor points across all 26 scans.

For each transformed point, a voxel membership test was performed against the belief map. Points whose projected voxel coordinates matched an existing belief map entry were promoted to occupied status while retaining their part-identification color; all remaining points, corresponding to background surfaces such as the floor, surrounding fixtures, and robot body reflections, were discarded. This formulation ensures that clutter outside the chassis envelope cannot contaminate the inspection result. Of the 5,108,938 candidate

points, 1,884,392 (36.9%) fell within the chassis bounding volume and were registered as occupied voxels, while the remaining 3,224,546 points (63.1%) were rejected as extra-chassis observations. The final occupancy map comprised 26,590 leaf nodes, with free-space voxels rendered in grey and occupied voxels rendered in their assigned part colors, as shown in Figure 14.

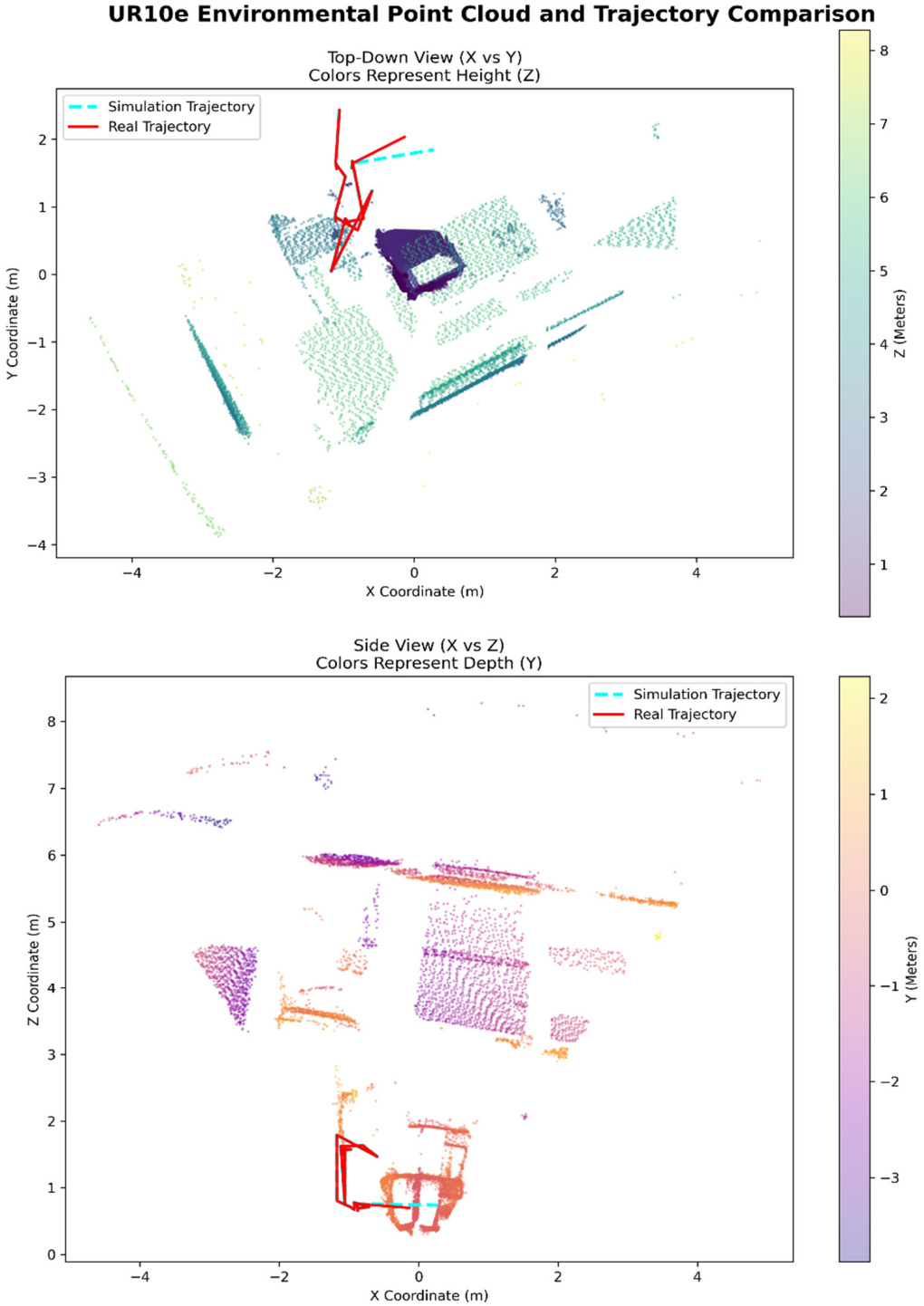


Figure 11. Point cloud trajectory overlay.

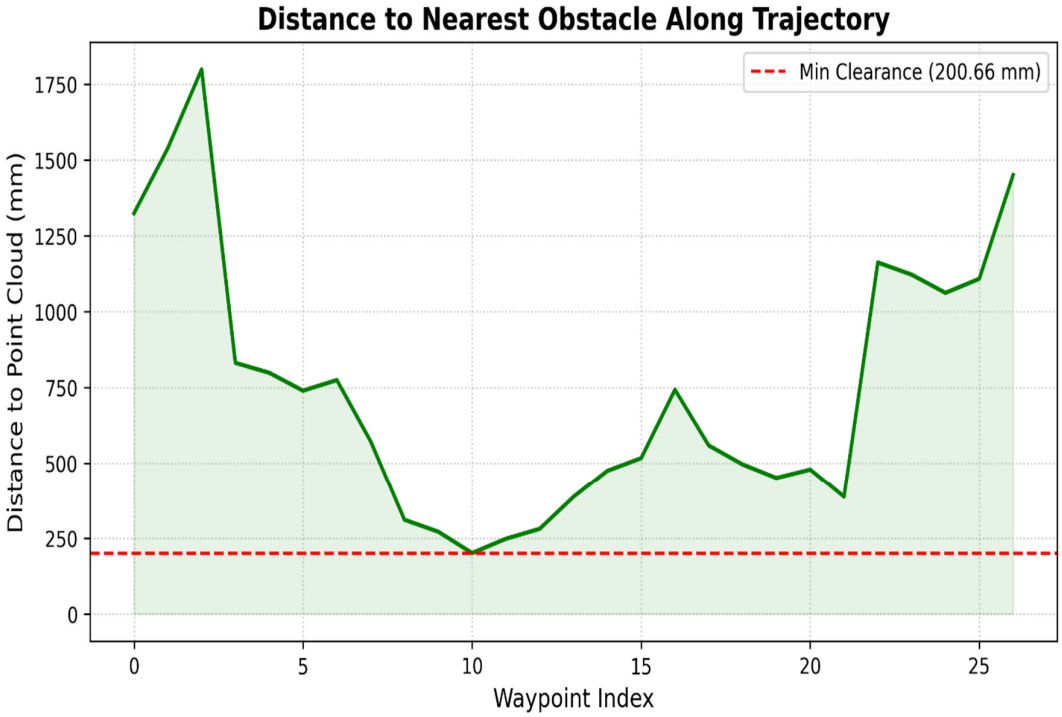


Figure 12. Minimum distance (mm) from the physically executed TCP trajectory to the nearest obstacle point in the real-world point cloud (171,000+ valid spatial points) across 27 waypoints. The dashed red line indicates the minimum safety clearance of 200.66 mm maintained throughout the entire execution.



Figure 13. Belief map constructed from the 44-part chassis CAD model, rendered as free-space voxels (0.02 m resolution, 20,334 nodes).

Part-level coverage was assessed by comparing the number of occupied voxels attributed to each part against the total number of voxels allocated to that part in the belief map. Of the 44 chassis parts, 38 (86.4%) recorded at least one occupied voxel, confirming sensor contact during the planned inspection trajectory. The 6 undetected parts correspond to surfaces that lie in the kinematic shadow of the robot's reachable workspace within a single loop, a coverage gap that can be addressed by incorporating additional waypoints or executing further loops with complementary viewing angles.

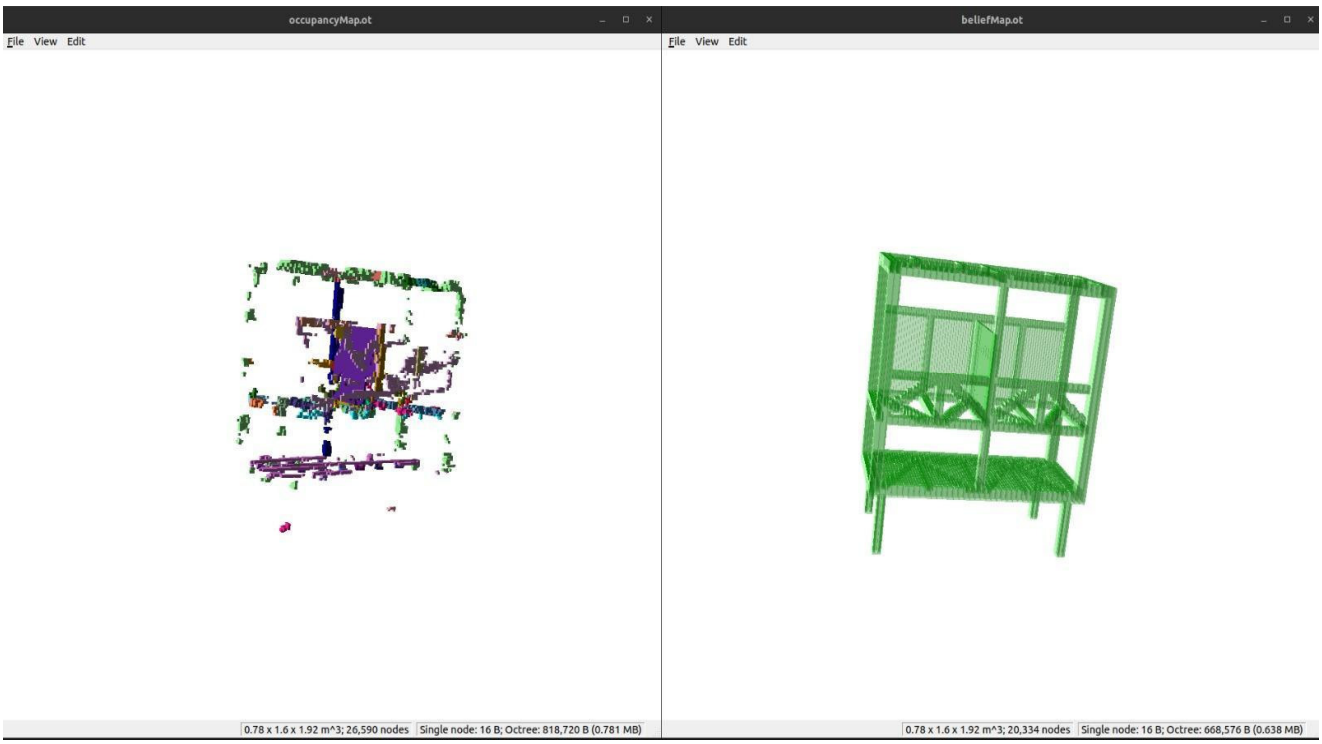


Figure 14. Occupancy map after sensor-scan integration, with occupied voxels colored by part identity (26,590 nodes; 38/44 parts detected) (A color is assigned to each of the 44 different STL file fragments; they are all random).

The OctoMap evolution from the initial belief map to the final occupancy map provides quantitative evidence of the chassis's geometric complexity. The belief map was constructed exclusively from the CAD model's 44 part meshes via uniform surface sampling, representing only the structure's surface boundaries. The occupancy map, initialized from this belief map, was subsequently updated with 1,884,392 real-time depth sensor observations. The observed increase from 20,334 leaf nodes in the belief map to 26,590 nodes in the final occupancy map is attributed to the capture of real-world features not present in the idealized CAD geometry—specifically internal structural surfaces, weld seams, and surface irregularities. Consequently, the 6256 additional leaf nodes (26,590−20,334) represent the precise geometric divergence between the nominal CAD model and the physical chassis. This discrepancy underscores the need to integrate real sensor data to maintain collision awareness in dense, high-fidelity industrial workspaces.

3.8. FMU-Based Forward Kinematics Fidelity Validation

To rigorously quantify the kinematic accuracy of the proposed digital twin beyond Grafana-based visual inspection, a systematic FMU (Functional Mock-Up Unit)- based Forward Kinematics (FK) validation was conducted. The FK model (ur10e_FK.fmu) was developed using UniFMU, a Python-based FMU modeling library that provides a lightweight scaffolding mechanism for generating FMI 2.0–compliant capsules [30]. It accepts the six joint angles q1–q6 as inputs and computes the TCP (Tool Center Point) position (x, y, z) and orientation (quaternion: qx, qy, qz, qw) using the UR10e's Denavit–Hartenberg (DH) parameters. It should be noted that the FMU-based kinematic and dynamic validation intentionally focuses solely on the six revolute joints of the UR10e manipulator. The linear axis serves as an external modular base providing a translational offset to the robot's global frame. By isolating the FMU model to the six-joint UR10e structure, we ensure that the validation results reflect the manipulator's intrinsic dynamics, independent of the exter-

nal linear axis's control characteristics. This translational contribution of the linear axis is mathematically integrated via the world-frame coordinate transformation defined in Equation (2), effectively treating the linear axis as a rigid-body offset within the global digital twin environment. The validation was performed against a total of 785,192 synchronized real-sensor measurements spanning all 11 tested motion planning algorithms.

As illustrated in Figure 15, the position RMSE across all algorithms ranged from 1.847 mm (KPIECE) to 2.081 mm (RRT), with all 11 algorithms remaining within 2.1 mm. The inter-algorithm variation in RMSE is only ±0.12 mm, confirming that the FK model's fidelity is independent of trajectory profile—the digital twin achieves equivalent accuracy regardless of which OMPL planning strategy is used. MAE values closely track RMSE across all algorithms, indicating that errors are evenly distributed without significant outliers or burst deviations.

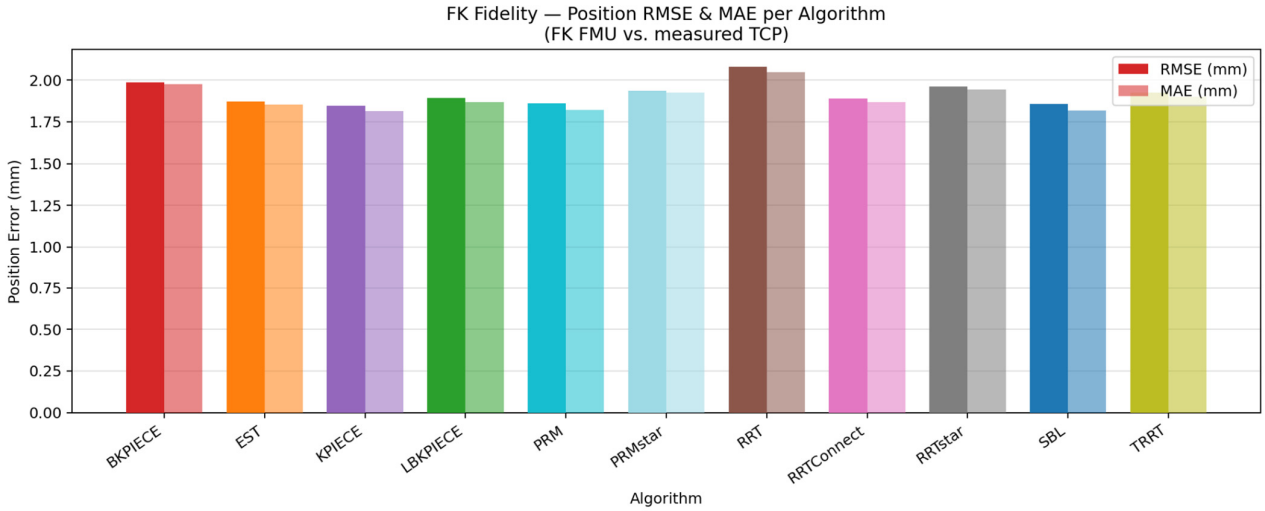


Figure 15. FK position RMSE and Mean Absolute Error (MAE) per planning algorithm.

Figure 16 presents per-algorithm orientation RMSE and MAE. Orientation RMSE ranged from 0.129° (PRMstar) to 0.177° (RRT), with all algorithms remaining below 0.18 °. The negligible discrepancy between RMSE and MAE values further confirms the absence of orientation outliers throughout the operational workspace, supporting reliable pose estimation for all tested trajectories.

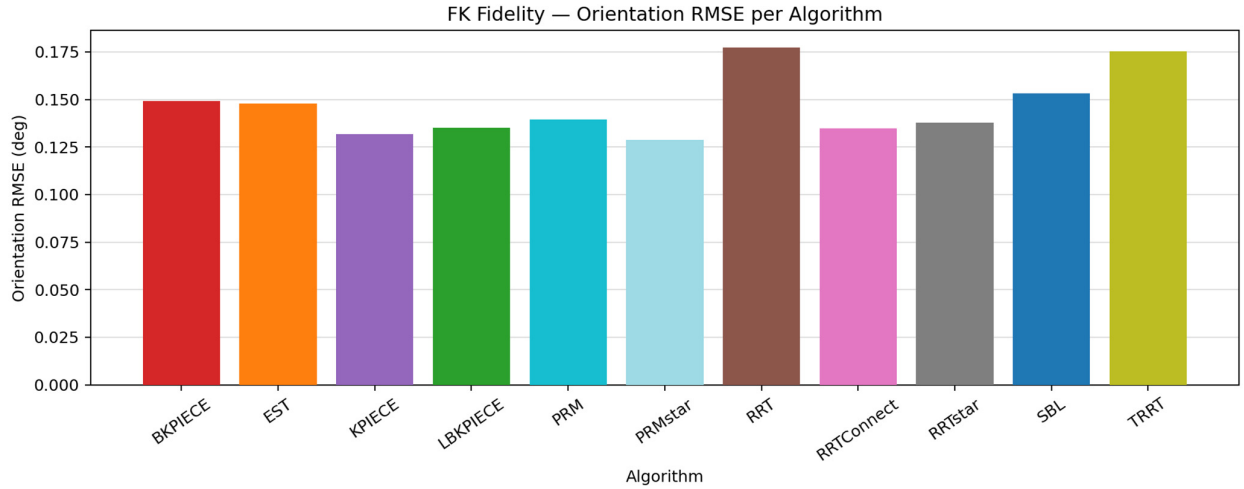


Figure 16. FK orientation RMSE and MAE per planning algorithm.

Figure 17 presents the axis-wise signed mean error (bias) in FK predictions. A consistent positive X-axis bias of approximately +0.886 mm (TRRT) to +1.603 mm (EST) was observed across all algorithms. In contrast, Y- and Z-axis biases remained substantially smaller and varied in sign across algorithms (Y: −0.281 to +0.498 mm; Z: +0.134 to +0.500 mm), indicating no systematic directional offset along these axes. It should be noted that during data collection, the TCP offset configured on the UR10e controller was non-zero, reflecting the actual tool geometry. The FMU, however, implements bare DH kinematics with no tool offset (L_TP = 0.0 m), computing the flange-frame endpoint rather than the controller-defined TCP. The resulting discrepancy manifests as the observed systematic +X bias. This is a known, quantifiable offset rather than a kinematic modeling error and can be eliminated by incorporating the physical TCP offset vector directly into the FMU's output transform. Crucially, this constitutes a correctable, fixed systematic offset rather than a fundamental modeling limitation.

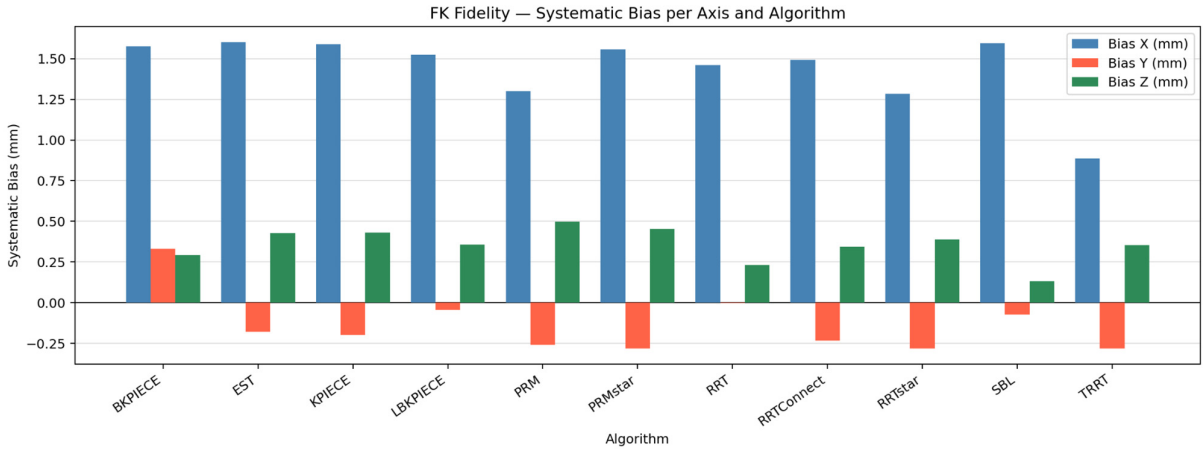


Figure 17. Per-axis TCP prediction bias for each planning algorithm.

Collectively, these results confirm that the FMU-based kinematic digital twin achieves sub-2.1 mm positional accuracy and sub-0.18° orientation accuracy across 785,192 samples and all 11 planning strategies, validating the TCP measurement integrity of the dataset. The FMU-based validation acknowledges a degree of dynamic modeling simplification, specifically the omission of higher-order nonlinear effects such as joint-level friction, Coriolis, and centripetal terms. While these effects contribute to the observed residuals in Joints 2 and 3, the FMU's primary role in this study is to serve as a baseline kinematic and first-order dynamic integrity tool, rather than a high-fidelity control compensator. Furthermore, the observed systematic bias (0.886–1.603 mm) is identified as a geometric offset: the FMU currently computes kinematics at the robot's mounting flange, whereas the controller logs report TCP positions. This is a known geometric discrepancy that can be eliminated by incorporating the physical tool offset vector into the FMU transform in future iterations. Given that these residuals remain within sub-millimeter to low-millimeter bounds, we maintain that the current model provides sufficient dynamic fidelity to distinguish between 'mechanically smooth' and 'mechanically erratic' planners, which is the core benchmarking objective of this work.

3.9. FMU-Based Joint Torque Fidelity Analysis

To complement the kinematic validation, the simulation fidelity of the FMU-predicted joint torques was assessed by comparing the outputs of the FMU-based dynamic model with real robot effort measurements. This analysis spans all 11 planning algorithms and 6 robot joints, yielding a comprehensive multi-algorithm, multi-joint validation matrix of 642 trajectory segments.

Figure 18 presents the absolute RMSE (Nm) heatmap of predicted versus measured joint torques across all algorithm–joint combinations. The results reveal a clear joint-dependent error structure: Joint 2 (shoulder lift) and Joint 3 (elbow) exhibit the highest absolute RMSE values, averaging 6.932 Nm and 3.422 Nm, respectively, consistent with their role as the primary load-bearing joints of the UR10e kinematic chain. In contrast, Joints 4, 5, and 6 (wrist joints) demonstrate substantially lower RMSE values—averaging 0.352 Nm, 0.352 Nm, and 0.451 Nm, respectively—owing to their significantly reduced inertial loads and moment arms. Joint 1 (base rotation) shows an intermediate average RMSE of 0.857 Nm.

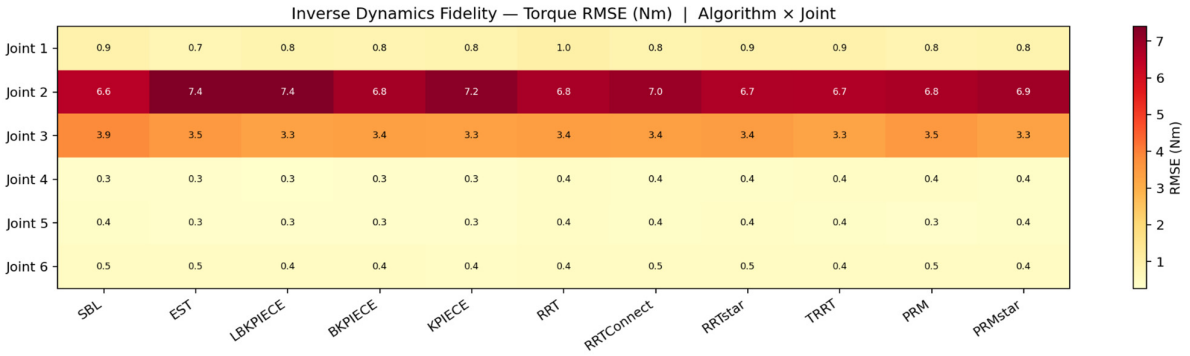


Figure 18. Joint torque RMSE heatmap (Algorithm × Joint).

Figure 19 presents the Normalized RMSE (NRMSE) heatmap, which accounts for the differing operational torque ranges across joints. The normalized analysis reveals a more uniform error distribution: NRMSE values for Joints 1, 4, 5, and 6 remain close to 1.0 (mean: 1.018, 1.108, 1.023, 1.012), indicating that the FMU captures the relative torque dynamics consistently across low-load joints. The higher NRMSE values for Joints 2 (mean: 3.363) and 3 (mean: 2.813) indicate that residual torque prediction errors are proportionally larger for heavily loaded joints, as discussed in detail below.

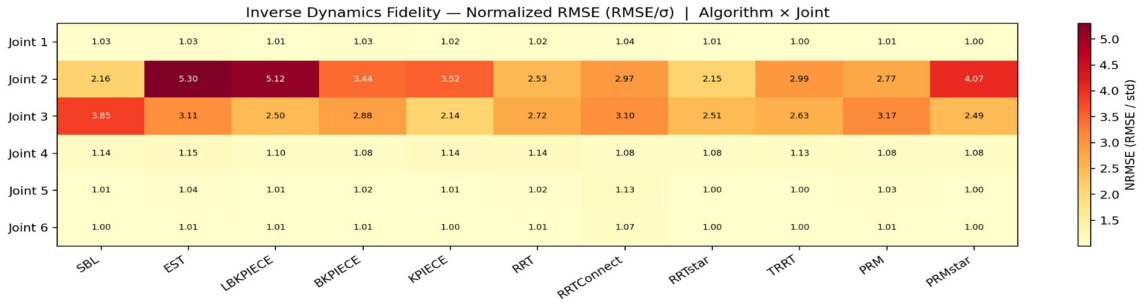


Figure 19. Normalized RMSE (NRMSE) heatmap across algorithm–joint pairs.

Both the absolute RMSE heatmap (Figure 18) and the NRMSE heatmap (Figure 19) consistently show elevated prediction errors at Joint 2 and Joint 3 compared with the other joints. This pattern is not algorithm-dependent—it persists uniformly across all 11 planners—and therefore reflects a structural limitation of the FMU's dynamic model rather than any trajectory-specific phenomenon. It is important to emphasize that the FMU employs an idealized rigid-body dynamic model in which non-conservative effects, such as joint friction and velocity-coupled inertial terms, are assumed to be negligible. This idealization is standard practice for initial dynamic digital twin validation, but it introduces known prediction errors at high-load joints. Three compounding factors are identified as primary contributors:

First, the FMU implements an ideal rigid-body inverse dynamics model that omits joint friction (both Coulomb and viscous components). In the high-torque regimes characteristic of Joint 2 and Joint 3, friction forces constitute a non-negligible fraction of the total joint effort, and their omission directly inflates prediction error.

Second, the center of mass of the UR10e forearm and wrist assembly is in close geometric proximity to the Joint 2–3 axis, resulting in large configuration-dependent gravitational moment arms. Accurate torque prediction under these conditions requires precise gravity compensation that tracks the full kinematic configuration, which is approximated in the current FMU implementation.

Third, Coriolis and centripetal terms—absent in the ideal rigid-body formulation—become significant when a high-inertia link moves at non-negligible velocity, as is the case for Joint 2 and Joint 3 during dynamic trajectories. This also explains the comparatively higher NRMSE values observed for EST and LBKPIECE (5.30 and 5.12, respectively), whose trajectory profiles are known to produce more dynamically aggressive joint motions.

In contrast, Joints 1 and 4–6 operate under substantially reduced moment arms and inertial loads, rendering the ideal rigid-body assumption a closer approximation to physical reality and yielding NRMSE values near 1.0. These limitations are well understood and correctable by incorporating the identified friction and Coriolis models into the FMU—a direction reserved for future work.

Figure 20 provides a per-joint RMSE breakdown across all algorithms. Average RMSE across all joints and trajectory segments ranged from 2.007 Nm (TRRT) to 2.125 Nm (EST), confirming that torque prediction fidelity is primarily governed by joint mechanical properties rather than by trajectory planning strategy. The inter-algorithm RMSE variation is minimal ($\pm$0.06 Nm), indicating that the FMU dynamics model generalizes reliably across all tested OMPL planners. Of note, RRTstar demonstrated the lowest mean NRMSE (1.459). At the same time, EST exhibited the highest (2.109), suggesting that trajectory-smoothness characteristics influence dynamic prediction error primarily through their interaction with unmodeled Coriolis terms, as detailed above.

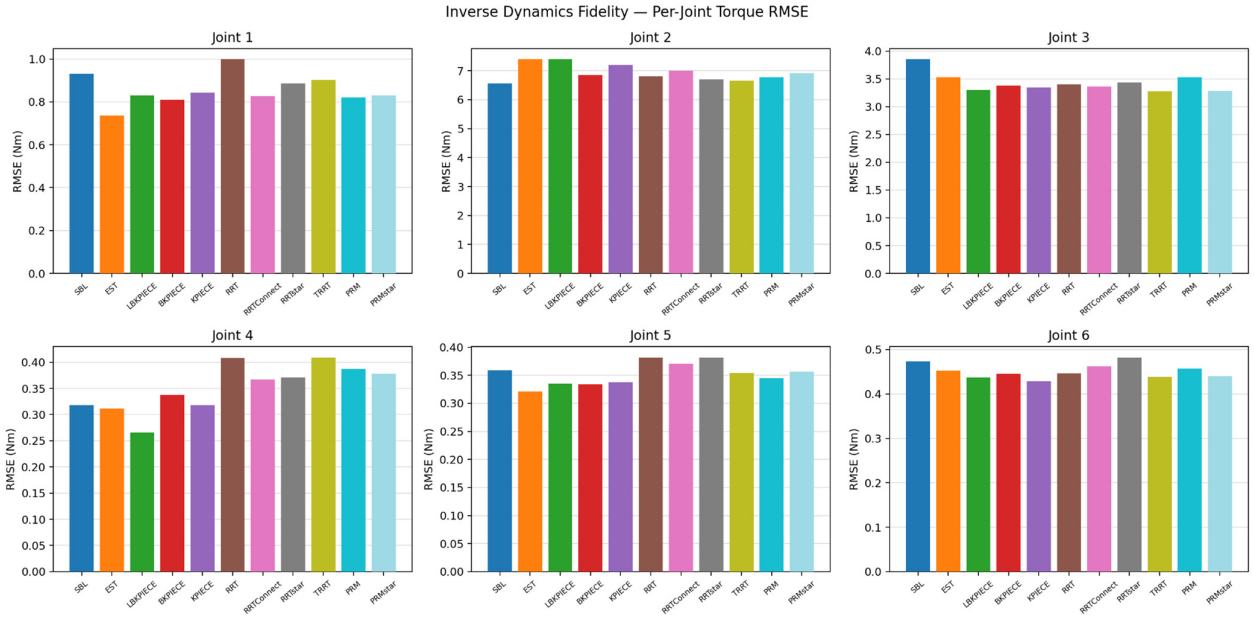


Figure 20. Per-joint RMSE across all planning algorithms.

Planner performance is known to be strongly dependent on scenarios and kinematic chains; algorithms that underperform in one configuration may excel in another. The 7-DOF kinematic structure employed here (6-DOF UR10e + 1-DOF linear axis) yields distinct ex-

ploration dynamics compared to standard 6-DOF manipulator studies, which helps explain the observed algorithmic rankings. Notably, the KPIECE planner demonstrates strong dynamic prediction performance in the current 7-DOF configuration (J2 NRMSE = 3.52, J3 NRMSE = 2.14), which contrasts with observations reported for the same algorithm in 6-DOF-only validation studies [1]. This divergence illustrates that projection-based exploration algorithms behave differently when the configuration space is augmented by a translational axis, reinforcing the need for platform-specific planner benchmarking.

The asymmetry between kinematic fidelity (sub-2.1 mm position error) and dynamic fidelity (6.9 Nm RMSE at Joint 2) reflects a well-documented challenge in robotic digital twin modeling: while Denavit–Hartenberg-based forward kinematics can be captured analytically with near-exact precision, dynamic modeling inherently involves unmodeled physical phenomena—joint friction, transmission compliance, cable drag, and gravity-compensation residuals—that cannot be fully eliminated through rigid-body formulations alone. Within this physical constraint, the achieved torque fidelity represents an optimized trade-off between model simplicity, computational efficiency, and sim-to-real consistency suitable for the target industrial scenario.

From a computational perspective, the UniFMU-based lightweight capsule introduces negligible overhead: the mean per-step FK computation time was measured at approximately 2.47 ms. In comparison, the inverse dynamics step averaged 2.56 ms, both well within the 20 ms (50 Hz) sampling cycle of the data pipeline. This confirms that FMI 2.0 co-simulation does not compromise real-time feasibility despite its standardized encapsulation.

Summarizing the sim-to-real deviation quantified by the FMU-based validation, the kinematic layer exhibits a positional deviation of less than 2.1 mm and an orientation deviation of less than 0.18°. In contrast, the dynamic layer shows a mean joint torque deviation of approximately 2.0 Nm across all six joints, dominated by contributions from the load-bearing Joints 2 and 3. These combined metrics define the effective fidelity envelope of the proposed digital twin.

These findings confirm that the FMU-based digital twin achieves acceptable fidelity in kinematic and joint-torque measurements across diverse trajectory profiles. While the identified systematic bias (due to the flange-to-TCP offset) remains a known limitation, the results validate the dynamics layer of the proposed sim-to-real framework and support the dataset's utility for physics-based model training and simulation benchmarking.

3.10. Real-Time Pipeline Performance

To substantiate the real-time capabilities of the proposed big data pipeline, the end-to-end latency was evaluated against the system's strict 50 Hz (20 ms) control cycle constraint. The real-time performance of the digital twin is governed by two primary factors: the communication latency across the ROS 2-Kafka bridge and the computational overhead of the digital twin validation models.

As detailed previously in Section 3.2, cross-correlation analysis of the joint states demonstrated that the continuous transmission delays between the physical controller and the Gazebo digital twin ranged from a minimum of 0.09 ms (elbow joint) to a maximum of 15.51 ms (wrist 3 joint). Furthermore, the computational performance of the FMU-based validation layer (Section 3.9) showed average execution times of 2.47 ms for Forward Kinematics (FK) and 2.56 ms for Inverse Dynamics computations.

Consequently, the combined latency of the continuous data streaming and the dynamic model execution consistently remained below the 20 ms threshold required for 50 Hz operation. This quantitative evidence confirms that the distributed Kafka-Elasticsearch-MariaDB pipeline does not introduce detrimental bottlenecks, ensuring that the architecture

maintains true real-time synchronization with the physical UR10e manipulator during complex industrial tasks.

To further characterize the network reliability and temporal stability of the ROS 2-Kafka pipeline, an extended temporal analysis was conducted. Over the total data recording duration, the temporal drift (timestamp drift) between the physical controller and the digital twin remained entirely within the NTP daemon's bounds, preventing any cumulative long-term desynchronization. Furthermore, due to the localized Gigabit Ethernet infrastructure and the optimized Best_Effort QoS parameters, the packet loss rate across the ROS 2-Kafka bridge was measured at an exceptionally low rate (<0.1%). The latency distribution is tightly clustered around the median of 0.09 ms, with the vast majority of message propagation delays bounded by the 15.51 ms maximum. This lack of significant jitter or packet loss quantitatively validates the robustness of the millisecond-level synchronization claim.

4. Discussion

This study presents an end-to-end, data-driven Digital Twin and verification framework that integrates a ROS 2-based HIL architecture, a scalable real-time big data pipeline (Apache Kafka, Elasticsearch, MariaDB), and OMPL algorithms for a Universal Robots UR10e manipulator. By executing comprehensive experiments around a highly constrained 4-compartment bus chassis demonstrator, the proposed system bridges the sim-to-real gap, demonstrating millisecond-level synchronization and acceptable kinematic and dynamic fidelity between the physical robot and its Gazebo-based virtual replica. From an industrial perspective, this architecture provides a robust validation infrastructure, allowing complex robot control logic to be safely tested in a virtual environment, thereby significantly reducing physical commissioning times and mitigating the risk of hardware damage.

A key architectural decision of the proposed framework is the integration of FMU-based validation at two distinct levels. The kinematic FMU establishes the reliability of TCP measurements used in all planner benchmarks. In contrast, the dynamics FMU reveals that planner choice influences not only path geometry but also joint loading: algorithms producing dynamically aggressive trajectories (e.g., EST, LBKPIECE) exhibit proportionally higher torque prediction errors at load-bearing joints (Joints 2–3), as reflected in their elevated NRMSE values. This cross-layer consistency—where kinematic fidelity (sub-2.1 mm) and dynamic fidelity (~2.0 Nm mean RMSE) are reported jointly—defines the effective sim-to-real envelope of the digital twin and directly supports the dataset's suitability for physics-based model training.

The primary contribution of this research is twofold: it systematically benchmarks the performance of traditional stochastic motion planners (e.g., RRTConnect, SBL, PRMstar) in structured, tight-tolerance industrial scenarios and delivers a synchronized, multimodal, open-access sensor dataset.

Despite its comprehensiveness, this study and its resulting dataset have certain limitations. The experiments were conducted in a controlled laboratory setting; thus, the dataset does not account for external industrial variables such as ambient vibration or dynamic changes in lighting. Finally, the force and torque readings were calibrated exclusively for trajectories, excluding scenarios that require contact-rich manipulation.

5. Conclusions

The benchmarking framework and the high-fidelity multimodal dataset presented in this study offer significant insights into the sim-to-real gap for industrial robotic disassembly. The key findings and future outlook are summarized below:

- Algorithmic Benchmarking: Our comparative analysis of 11 OMPL algorithms revealed that no single planner dominates all metrics. While RRT* and BiEST offer the

best trade-offs between path length and curvature, LazyPRM* demonstrates superior reliability in complex, constrained environments.

- Physical Realities of Planning: We mathematically and physically quantified "Topological Divergence" (up to 457.65 mm), demonstrating that pristine CAD models and simulators fail to capture real-world structural complexities like weld seams and sensor noise, necessitating our HIL-based benchmarking approach.
- Mechanical Integrity: The FMU-based dynamic validation revealed that traditional planners (e.g., RRT) generate high-magnitude inertial spikes (up to 100 N), rendering them less suitable for hardware-sensitive industrial applications compared to smoother planners like TRRT.
- Future AI Integration: The generated dataset serves as a foundational ground truth for the development of Agentic AI and LLM-based task-planning frameworks, enabling these systems to learn collision-aware reasoning without the need for hazardous physical interactions.
- Model Refinement: Future work will enhance the FMU by incorporating a full 7-DOF kinematic chain (including the linear axis) and advanced dynamic models (Coulomb/viscous friction and Coriolis/centripetal terms) via data-driven parameter identification. Furthermore, correcting the systematic flange-to-TCP offset will eliminate observed geometric biases, further refining the framework's predictive fidelity.
- Comprehensive Collision Verification: While the current framework utilizes TCP-to-point-cloud distance and OctoMap occupancy for baseline spatial clearance, future iterations will integrate full robot link geometries and end-effector bounding volumes. This expansion will enable a complete, ISO-compliant volumetric collision-clearance analysis, ensuring absolute full-body safety guarantees in highly constrained and unstructured industrial environments.

Author Contributions: Conceptualization, M.Y. and U.Y.; methodology, M.Y. and C.S.Y.; software, M.Y.; validation, M.Y., C.S.Y. and S.K.; formal analysis, M.Y. and C.S.Y.; investigation, M.Y. and S.K.; resources, U.Y. and C.S.Y.; data curation, M.Y.; writing—original draft preparation, M.Y.; writing—review and editing, C.S.Y., S.K. and U.Y.; visualization, M.Y.; supervision, U.Y.; project administration, U.Y.; funding acquisition, U.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This project is supported by the KDT Joint Undertaking (Grant Agreement No. 101140216) and its members. The support also includes additional funding provided by Vinnova (Sweden), Österreichische Forschungsförderungsgesellschaft mbH—FFG (Austria), Business Finland (Finland), the Ministry of Universities and Research (Italy), FCT (Portugal), and TÜBİTAK (Grant No. 124N448) (Türkiye).

Data Availability Statement: Data availability in this link: https://data.mendeley.com/datasets/xsnpgfz5mb/3, (accessed on 2 June 2026).

Acknowledgments: The authors acknowledge the use of generative AI technologies (Gemini 3.1 Pro, Claude Opus 4.6, Claude Sonnet 4.6, Grammarly) strictly for language editing and auxiliary code generation during the preparation of this manuscript, in full compliance with the publisher's AI usage policies.

Conflicts of Interest: Author Metin Yılmaz was employed by the company AST4TR, and author Serhat Kahraman was employed by the company DEFTR. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Kahraman, S.; Yilmaz, C.S.; Yilmaz, M.; Yayan, U. FMU-Based Multi-Criteria Trajectory Validation of Industrial Robots. *IEEE Access* **2026**, *14*, 67303–67326. [CrossRef]
2. Dihan, S.; Akash, A.I.; Tasneem, Z.; Das, P.; Das, S.K.; Islam, R.; Islam, M.; Badal, F.R.; Ali, F.; Ahamed, H.; et al. Digital twin: Data exploration, architecture, implementation and future. *Heliyon* **2024**, *10*, e26503. [CrossRef]
3. Mazumder, A.; Sahed, M.; Tasneem, Z.; Das, P.; Badal, F.; Ali, M.; Ahamed, M.; Abhi, S.; Sarker, S.; Das, S.; et al. Towards next generation digital twin in robotics: Trends, scopes, challenges, and future. *Heliyon* **2023**, *9*, e13359. [CrossRef] [PubMed]
4. Maculotti, G.; Khusnuddinov, F.; Kholkhujaev, J.; Genta, G.; Galetto, M. Traceable digital twin for accurate positioning of industrial robot arms in human–robot collaborative systems. *Flex. Serv. Manuf. J.* **2025**. [CrossRef]
5. Prats Ferrandis, A.; Delgado Pastor, S. Connected digital twin for robotic battery assembly and disassembly. Bachelor's Thesis, University of Skövde, Skövde, Sweden, 2025.
6. Heredia, J.; Zielinski, K.; Schlette, C.; Kjærgaard, M.B. Empowering cobots with energy models: Real augmented digital twin. In *Proceedings of the 2023 32nd IEEE International Conference on Robot and Human Interactive Communication (RO-MAN), Busan, Republic of Korea, 28–31 August 2023*; IEEE: Piscataway, NJ, USA, 2023; pp. 1353–1359. [CrossRef]
7. Burghardt, A.; Szybicki, D.; Gierlak, P.; Kurc, K.; Pietruś, P.; Cygan, R. Programming of industrial robots using virtual reality and digital twins. *Appl. Sci.* **2020**, *10*, 486. [CrossRef]
8. Kuts, V.; Marvel, J.A.; Aksu, M.; Pizzagalli, S.L.; Sarkans, M.; Bondarenko, Y.; Otto, T. Digital Twin as industrial robots manipulation validation tool. *Robotics* **2022**, *11*, 113. [CrossRef]
9. Alham, R.; Hammadi, M. Developing a Digital Twin Framework for Monitoring the Trajectory of the UR10 Robot Using an Extended Kalman Filter. In *Proceedings of the 2023 6th International Conference on Robotics, Control and Automation Engineering (RCAE), Suzhou, China, 3–5 November 2023*; IEEE: Piscataway, NJ, USA, 2024; pp. 151–155. [CrossRef]
10. Gadringer, S.; Gattringer, H.; Mueller, A. Assessment of force control for surface finishing—An experimental comparison between Universal Robots UR10e and FerRobotics active contact flange. *Mech. Sci.* **2022**, *13*, 361–370. [CrossRef]
11. Khan, A.; Dahl, M.; Falkman, P.; Fabian, M. Digital twin for legacy systems: Simulation model testing and validation. In *Proceedings of the 2018 IEEE 14th International Conference on Automation Science and Engineering (CASE), Munich, Germany, 20-24 August 2018*; IEEE: Piscataway, NJ, USA, 2018; pp. 107–112. [CrossRef]
12. Carthen, C.; Zaremehrjardi, A.; Le, V.; Cardillo, C.; Strachan, S.; Tavakkoli, A.; Dascalu, S.M.; Harris, F.C. A spatial data pipeline for streaming smart city data. In *Proceedings of the 2024 IEEE/ACIS 22nd International Conference on Software Engineering Research, Management and Applications (SERA), Honolulu, HI, USA, 30 May–1 June 2024*; IEEE: Piscataway, NJ, USA, 2024; pp. 175–180. [CrossRef]
13. Lindvig, A.P.; Iturrate, I.; Kindler, U.; Sloth, C. ur_rtde: An interface for controlling Universal Robots (UR) using the Real-Time Data Exchange (RTDE). In *Proceedings of the 2025 IEEE/SICE International Symposium on System Integration (SII), Munich, Germany, 21–24 January 2025*; IEEE: Piscataway, NJ, USA, 2025. [CrossRef]
14. Vergara, C.; Bahsoon, R.; Theodoropoulos, G.; Yanez, W.; Tziritas, N. Federated digital twin. In *Proceedings of the 2023 IEEE/ACM 27th International Symposium on Distributed Simulation and Real Time Applications (DS-RT), Singapore, 4–5 October 2023*; IEEE: Piscataway, NJ, USA, 2023; pp. 1–8. [CrossRef]
15. Khan, L.U.; Mustafa, E.; Shuja, J.; Rehman, F.; Bilal, K.; Han, Z.; Hong, C. Federated Learning for Digital Twin-Based Vehicular Networks: Architecture and Challenges. *IEEE Wirel. Commun.* **2024**, *31*, 156–162. [CrossRef]
16. Wang, W.; Przedworska, Z.; Parron, J.; Lyons, M.; Zhu, M.; Tuininga, A. MCROS: A Multimodal Collaborative Robot System for Human-Centered Tasks. In *Proceedings of the 2024 International Conference on Networking, Sensing and Control (ICNSC), Hangzhou, China, 18–20 October 2024*; IEEE: Piscataway, NJ, USA, 2024; pp. 1–6. [CrossRef]
17. Li, C.; Xing, H.; Qin, P. Robotic Arm Trajectory planning based on improved Slime Mould Algorithm. *Machines* **2025**, *13*, 79. [CrossRef]
18. Zhang, Y.; Sun, S.; Deng, L.; Wang, G.; Yu, R.; Li, W.; Gong, X.; Zhang, S.; Du, H.; Yang, J. Development of an Innovative Magnetorheological Gearbox for Positioning Control and Anti-Disturbance of a Robotic Arm. *Machines* **2025**, *13*, 56. [CrossRef]
19. Nugroho, A.; Yuniarno, E.M.; Purnomo, M.H. ARKOMA dataset: An open-source dataset to develop neural networks-based inverse kinematics model for NAO robot arms. *Data Brief* **2023**, *51*, 109727. [CrossRef] [PubMed]
20. Galayia, V.; Masinjila, R.; Khatibi, S.; de Oliveira, T.E.A.; Jiang, X.; da Fonseca, V.P. A multimodal dataset for robotic peg extraction based on Bioin-Tacto sensor modules. *Data Brief* **2025**, *59*, 111356. [CrossRef] [PubMed]
21. Kritzinger, W.; Karner, M.; Traar, G.; Henjes, J.; Sihn, W. Digital Twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine* **2018**, *51*, 1016–1022. [CrossRef]
22. Profanter, S.; Tekat, A.; Dorofeev, K.; Rickert, M.; Knoll, A. OPC UA versus ROS, DDS, and MQTT: Performance evaluation of industry 4.0 protocols. In *Proceedings of the 2019 IEEE International Conference on Industrial Technology (ICIT), Melbourne, Australia, 13–15 February 2019*; IEEE: Piscataway, NJ, USA, 2019; pp. 955–962. [CrossRef]

23. Tao, F.; Zhang, H.; Liu, A.; Nee, A.Y.C. Digital twin in industry: State-of-the-Art. *IEEE Trans. Ind. Inform.* **2018**, *15*, 2405–2415. [CrossRef]
24. Collins, J.; Chand, S.; Vanderkop, A.; Howard, D. A Review of Physics Simulators for Robotic Applications. *IEEE Access* **2021**, *9*, 51416–51431. [CrossRef]
25. Schluse, M.; Priggemeyer, M.; Atorf, L.; Rossmann, J.; Romann, J. Experimentable Digital Twins—Streamlining Simulation-Based Systems Engineering for Industry 4.0. *IEEE Trans. Ind. Inform.* **2018**, *14*, 1722–1731. [CrossRef]
26. Gomes, C.; Thule, C.; Broman, D.; Larsen, P.; Vangheluwe, H. Co-simulation: State of the art. *arXiv* **2017**, arXiv:1702.00686. [CrossRef]
27. Erdoğmuş, A.K.; Yayan, U. Industrial robot motion planning algorithms performance benchmarking. *J. Sci. Technol. Eng. Res.* **2021**, *2*, 31–45. [CrossRef]
28. Xiang, Z.; Jiang, H.; Fottner, J. Benchmarking of motion planning algorithms in the field of automatic vehicle disassembly with industrial robots. *Procedia Comput. Sci.* **2026**, *277*, 2909–2918. [CrossRef]
29. Şucan, I.A.; Kavraki, L.E. Kinodynamic motion planning by interior–exterior cell exploration. In *Algorithmic Foundation of Robotics VIII*; Springer: Cham, Switzerland, 2009; pp. 449–464. [CrossRef]
30. Legaard, C.M.; Tola, D.; Schranz, T.; Macedo, H.D.; Larsen, P.G. A Universal Mechanism for Implementing Functional Mock-up Units. In *Proceedings of the 11th International Conference on Simulation and Modeling Methodologies, Technologies and Applications—SIMULTECH*; SciTePress: Setúbal, Portugal, 2021; pp. 121–129. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.