Solution Brief

Industrial and Robotics
Intel® Core™ Ultra Processors

Open, Real-Time AI Robotics Control at the Edge

Powered by Intel® Open Edge Platform, OpenVINO™, and Ubuntu

Introduction

Manufacturing is moving toward open, software-defined automation platforms that unify Artificial Intelligence (AI), control, and orchestration at the edge. This solution brief covers the Robot Vision and Control (RVC) demo, which is based on Robot Operating System 2 (ROS 2) and unifies AI-powered perception with precise robotic control at the edge. Intel and Canonical support open, modular robotics solutions that integrate real-time-capable control, AI inference, and orchestration on Ubuntu—helping accelerate deployment, reduce the risk of vendor lock-in, and improve operational efficiency. Intel® Open Edge Platform and Robotics AI Suite Technologies [1], with Ubuntu as the operating system (OS—a Long-Term Support (LTS) release [2]) and Intel® Time Coordinated Computing (Intel® TCC) real-time tuning guidance for Intel platforms [3,4], provide a stable Linux foundation for the edge robotics stack. Where bounded-latency control behavior is required, a real-time kernel option (PREEMPT_RT) can be enabled as part of the system configuration. For manufacturing and automation engineers, system integrators, and edge AI developers, the ability to deploy flexible, intelligent robotics at the edge is becoming a key competitive advantage. RVC supports the integration of vision, motion, and automation, consolidating workloads with different timing requirements. The demo includes both real-time (RT) and non-real-time (non-RT) functions such as RT control, Human–Machine Interface (HMI), AI inference, computer vision, and automation control.

System Architecture and Technologies

Figure 1 presents the system architecture diagram of the RVC demo. It runs on an industrial PC based on an Intel® Core™ Ultra processor with Ubuntu and an optional real-time kernel configuration (PREEMPT_RT). It uses OpenVINO™ [5] to optimize AI vision workloads across available Intel® compute engines, including Central Processing Units (CPUs), Graphics Processing Units (GPUs), and Neural Processing Units (NPUs), where available.

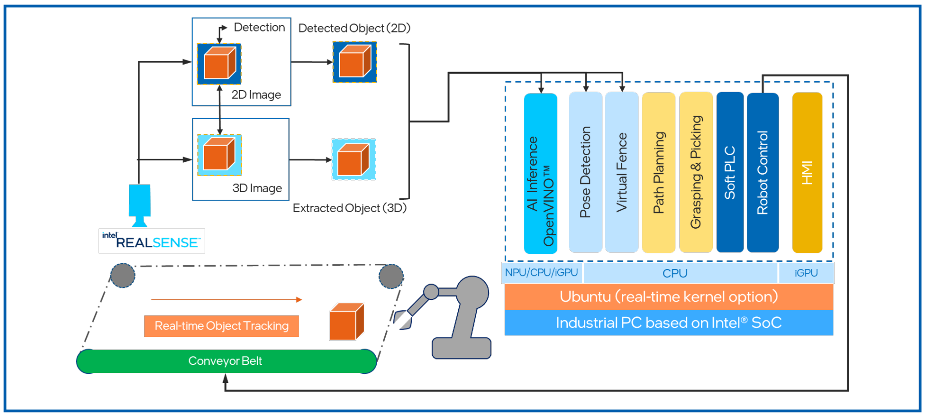

Figure 1. System Architecture Diagram

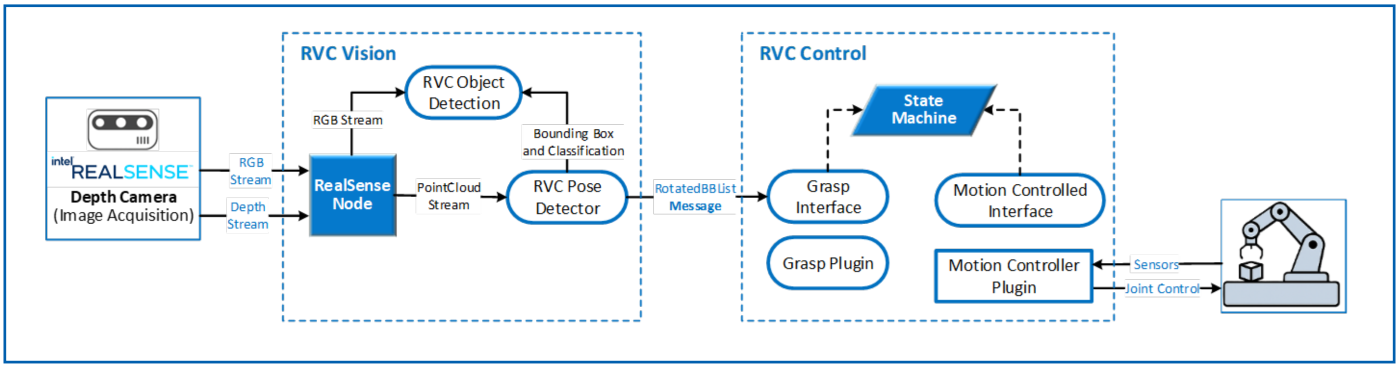

Figure 2. RVC System Workflow

Key architectural highlights include:

- Plugin-based vision pipeline: Easily swap 2D/3D AI models.
- Hardware abstraction: Replace robots or Intel® RealSense™ cameras with minimal reconfiguration.
- Container-ready: Runs in Docker* and virtual machine environments.

Intel AI Acceleration: RVC can take full advantage of the various AI capabilities offered by Intel® Open Edge platforms [1]:

- Intel® CPUs: Leverage built-in instruction sets for efficient AI inference on all Intel Core Ultra processors.
- Intel® GPUs: Integrated Intel® Arc™ and Intel® X^e graphics deliver parallel AI processing with OpenVINO acceleration.
- Intel® NPUs: Enable power-efficient, dedicated AI inference — ideal for selected always-on edge workloads (platform dependent).
- OpenVINO Runtime: Helps distribute inference workloads across CPU, GPU, and NPU for performance and deployment flexibility (hardware and configuration dependent).

System Workflow

Figure 1 provides the system-level architecture, while Figure 2 zooms into the ROS 2 component workflow and key interfaces used in the demo. As illustrated in Figure 2, RVC integrates two subsystems: RVC Vision and RVC Control, working together to deliver real-time perception and intelligent robotic actuation at the edge.

Sensor input (demo): Intel RealSense depth camera streams RGB/depth data to the ROS 2 RealSense node, feeding the RVC Vision pipeline. The Vision subsystem detects and localizes target objects, tracks them in 2D using a deep neural network, and estimates full six degrees of freedom (6DoF) pose in 3D through depth-based point-cloud matching. The Control subsystem uses the object pose to compute motion commands and updates the robot in a predictable, bounded-latency manner (configuration dependent) to support smooth and precise movement.

ROS 2 vision flow (demo configuration): RVC Vision publishes perception results using the custom ROS 2 messages RotatedBBList — an array of RotatedBB entries such as bounding box parameters, object class ID, and confidence.

ROS 2 control flow (demo configuration): The control node loads the Grasp Interface and Motion Controller plugins and runs a state machine; the Grasp Plugin subscribes to RotatedBBList to compute a target pose, and the Motion Controller executes the commanded task steps (details may vary with configuration).

In parallel with AI vision and robot control, RVC can also run additional workloads such as:

- Software-based programmable logic control (SoftPLC) for managing conveyor belts, laser breaks, and indicator lights.
- Integrated safety logic to halt robot operation upon human detection in the work area based on computer vision (demo concept; implementation depends on system safety requirements).

Canonical's Robotics Reference Architecture

Figure 3 summarizes Canonical's robotics reference architecture [6] for production-oriented deployment and operations. It shows how robotics applications can be packaged, distributed, deployed, and monitored consistently across devices.

- Demo runtime baseline: Ubuntu Long-Term Support (LTS) with an optional real-time kernel (PREEMPT_RT), as configured for the RVC demo.
- Production option (reference architecture): Ubuntu Core (the immutable flavor of Ubuntu for embedded devices [7]) with snaps for application packaging, over-the-air (OTA) updates, and Expanded Security Maintenance (ESM) for ROS (supported distributions/package sets), along with an observability stack for fleet monitoring and operations.

As shown in Figure 3, the Canonical robotics reference architecture stack supports the robotics application lifecycle through four phases:

- **Develop & package:** Build robotics applications on Ubuntu Desktop with native ROS 2 ecosystem support. Package them as snaps for reliable application confinement.

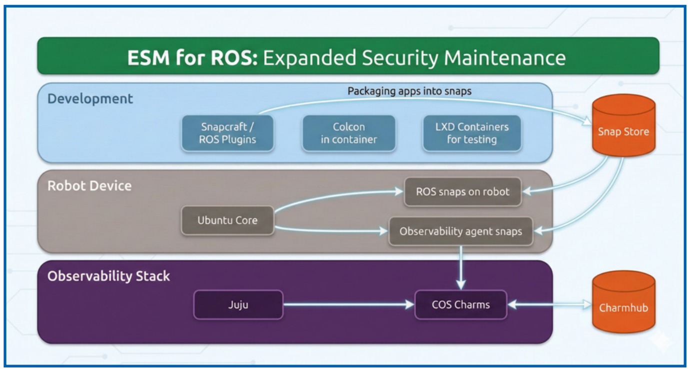

Figure 3. Canonical's Robotics Reference Stack Overview

- **Distribute:** Publish snaps via the Snap Store for deployment to a fleet of devices. Use snaps' native OTA mechanism for controlled updates.
- **Deploy & run:** Run applications and observability agents on Ubuntu Core-based robot devices.
- **Monitor:** Use the Canonical Observability Stack (COS), managed by Juju, to collect telemetry and enable monitoring and troubleshooting.

Conclusion

The RVC demo illustrates how mixed-criticality workloads—such as real-time control, non-real-time services, and AI workloads—can be consolidated on a single Intel silicon-based edge platform running Ubuntu. Together with Intel Open Edge technologies and Canonical's Ubuntu software foundation, it is designed to empower manufacturing and automation engineers, system integrators, and edge AI developers to accelerate deployment, adapt faster, and reduce time-to-market (TTM). Future updates may enable Intel® TCC on supported platforms to further improve real-time responsiveness and timing stability.

Reference

[1] Intel® Open Edge Platform, Robotics AI Suite (2025.2): https://docs.openedgeplatform.intel.com/2025.2/ai-suite-robotics.html

[2] Ubuntu for Intel platforms: https://ubuntu.com/download/iot/intel-iot

[3] Intel® Time Coordinated Compute (Intel® TCC) User Guide: https://www.intel.com/content/www/us/en/content-details/851159/public-intel-time-coordinated-compute-tcc-user-guide.html?DocID=851159

[4] Optimizing real-time performance on Intel CPUs: https://documentation.ubuntu.com/real-time/latest/tutorial/intel-tcc/

[5] OpenVINO™ toolkit, 2025 https://docs.openvino.ai/2025/get-started.html

[6] Canonical's Robotics reference architecture: https://canonical-robotics.readthedocs-hosted.com/en/latest/references/ref_architecture/reference_architecture/

[7] Ubuntu Core-the embedded Linux OS for devices: https://ubuntu.com/core

Authors

Intel - www.intel.com

Candy Wei
Dario Russo
Tyler Lewis
Michael Schmidt

Canonical - canonical.com

Serkan Uygungelen
Florian Nebout