Yugmil Patel

Robotics Engineer · Teleoperation · Arm Integration · Kinematics and Control · Safety-Critical Software
Ottawa, ON | +1 (343) 543-8232 | patelyugmil@gmail.com | linkedin: yugmilpatel | yugmilautomates.com

Summary

Robotics engineer who takes arms from vendor parts to robots an operator can control, and owns everything that runs on them. Primary author of the software for Sielo Robotics' assistive robot arm, from bare motors on a CAN bus to units in the field: the C++ hardware interface for seven actuators from two vendors, a 250–400 Hz control loop with a closed-form IK solver, a layered safety system with heartbeat and watchdog failsafes, and the teleoperation and recording tools used to collect demonstrations, with VR controllers, a 6-DoF SpaceMouse and joysticks arbitrated into one command stream. Previously founding engineer at Open Droids, integrating several vendors' hardware into a mobile manipulator for CES 2025 and building a VR teleoperation pipeline operated from Ottawa with the robot in San Francisco. Python and C++ on Linux.

Focus: teleoperation and demonstration data for embodied AI, and building a deep understanding of foundation models and world models for robotics.

Selected Work

Teleoperation over long-distance networks *Open Droids and Sielo Robotics*

At Open Droids, built a VR teleoperation pipeline that streams a robot's onboard depth to a Meta Quest 3 as a point cloud, operating the robot at 100–200 ms latency on the local network and 300–400 ms from Ottawa with the robot in San Francisco. At Sielo, built the Meta Quest 3S teleoperation client (Unity/OpenXR over UDP) with deadman-gated 1:1 pose tracking, so the arm follows the operator's hand only while the deadman is held.

Operator input devices arbitrated into one command stream *Sielo Robotics*

Integrated four input devices for the same arm: Meta Quest 3S controllers, a 6-DoF SpaceMouse, the wheelchair's own joystick and a browser joystick in the operator web app, arbitrated into one command stream behind a joystick arming interlock. Added teach-by-demonstration trajectory recording and replay, and a zero-torque hand-guiding mode for positioning the arm by hand.

Integrating arms from vendor protocols into one control interface *Sielo Robotics and Open Droids*

Wrote the C++ `ros2_control` hardware interface and CAN driver for seven actuators using two vendors' protocols on one bus (Robstride extended-ID and Damiao MIT mode), with vendor-neutral motor types, and led the swap to Damiao actuators on the base joints, flashing and tuning the drives. At Open Droids, wrote interfaces for several vendors' RS232 and RS485 protocols on a Jetson Orin NX and the ROS 2 layer that ran off-the-shelf components as one mobile manipulator, delivered in four months for CES 2025.

Kinematics and control for a 6-axis arm *Sielo Robotics*

Wrote and tuned the 250–400 Hz control loop: per-joint PD gains, velocity and friction feedforward, filtered feedback and gravity compensation. Ported and extended a closed-form IK plugin for MoveIt (C++, Eigen) with a gtest suite and an A/B benchmark: 1.26 µs mean solve vs. 5.1 ms for TRAC-IK on the Jetson, with 100% success for both. Kept the URDF/Xacro and SRDF descriptions, joint limits and joint directions matched to the hardware through three CAD re-exports, verified joint by joint.

Safety when the link, the data or the hardware misbehaves *Sielo Robotics*

Designed the layered safety system: a hardware E-stop relay, ESP32-C6 safety firmware with a 20 Hz host heartbeat and 150 ms failsafe, a host-loop watchdog, a velocity guard at the CAN boundary, and a torque-residual collision detector with a staged halt sequence. Traced an uncommanded arm motion through MCAP recordings to stale lift-motor feedback plus a mechanical jolt, with joystick input confirmed at zero, then shipped the arming interlock and snapshots on operator stop. Added start-up plausibility checks against wrapped-encoder jumps and motor-fault watchdogs; preparing the arm's ISO 14971 risk assessment.

Recording and robot-learning data *Sielo Robotics and personal*

Wrote an off-process C++ MCAP recorder that subscribes to raw serialized messages without deserializing them and keeps a 5-minute rolling buffer, cutting recording CPU use from 0.76 to 0.10 cores. Sole author of SieloStudio, which records teleoperated episodes to MCAP with provenance, converts them to LeRobot datasets, trains ACT, Diffusion Policy, SmolVLA and π_0, and pins every model to a data-contract version so a deployed policy receives the same inputs it was trained on. Tested π_0, $\pi_{0.5}$ and Diffusion Policy on the arm through the Rosetta LeRobot-to-ROS 2 bridge, and ACT, Diffusion Policy and $\pi_{0.5}$ on a LeRobot SO-ARM101 at home.

Assisted manipulation prototypes *Sielo Robotics and Open Droids*

Shipped click-to-go in the operator app: a clicked pixel is back-projected through depth into the robot's base frame and turned into a pose facing the target. Built a prototype OAK-D stereo pipeline running YOLOv8 on the camera's onboard processor, converting detections into 3D targets through a calibrated camera-to-base extrinsic for a C++ grasp planner. At CES 2025, the Open Droids robot detected an object, estimated its 3D position from depth, planned a grasp with MoveIt and handed it to a person.

Experience

Sielo Robotics **May 2025 – Present**
Robotics Engineer *Ottawa, ON*
Arlo, a wheelchair-mounted 6-axis assistive robot arm on NVIDIA Jetson (about 90% of commits). Beyond the work above:

- Moved the control and teleoperation processes to SCHED_FIFO with CPU pinning on a PREEMPT_RT kernel, with a fail-safe fallback, eliminating a power-on jerk caused by the scheduler preempting the CAN write loop.
- Profiled a Jetson at about 78% load process by process and recovered headroom through RelWithDebInfo builds, gated debug publishers, rate tuning and camera encoding only while a viewer is connected.
- Brought up the custom safety carrier PCB and its RS485/Modbus IMU link, isolating four stacked faults (unwired driver enable, swapped A/B lines, inverted TX/RX silkscreen, parser bug) with a multimeter and a serial console.
- Built the Gazebo digital twin (`ros_gz_sim`, `ros2_control`, MoveIt 2, RViz) and validated kinematics, controller behavior, trajectory execution and robot-state synchronization across simulation and hardware.
- Built the operator web app (React/TypeScript on FastAPI) with live multi-camera video, and a ROS 2 diagnostics health tree for every subsystem and camera.
- Built the release and fleet tooling (GitLab CI on a self-hosted Jetson runner, health-checked deploys with automatic rollback, over-the-air updates, Grafana Cloud alerts) and wrote the system architecture guide (about 1,900 lines) and protocol contracts.

Open Droids **May 2024 – July 2025**
Founding Robotics Engineer *Remote – San Francisco, CA*
Founding engineer at a domestic-robotics startup. Led system design and integration of its first robot, *R1D1*. Beyond the work above:

- Validated control, sensor integration and manipulation in Gazebo before hardware, and worked asynchronously with mechanical, electrical and systems teams in the US and China on hardware integration and robot model changes.

Projects

Four-Wheel Swerve-Drive AMR with Modular Attachments **2024**
ROS 2, Ignition Gazebo, URDF/SDF, Python/C++

- Derived the swerve kinematics from scratch and wrote a custom ROS 2 controller that runs with any standard gamepad, tested in Ignition Gazebo with sensor topics bridged into ROS 2. The post reached 30,000+ views on LinkedIn and was reposted by Open Robotics.

Autonomous Power-Line Inspection – National Research Council Canada **2026**
NVIDIA Isaac Sim, ROS 2, Python/C++

- Built an Isaac Sim environment connected to ROS 2 for robot state, perception, planning, control and data collection, with reusable test scenarios across robot poses, viewpoints and operating conditions.

Kisaan Rover – Autonomous Agricultural UGV **2022 – 2023**
ArduPilot, NVIDIA Jetson Nano, RTK GPS

- Led development of an eight-wheel autonomous UGV for seeding and weeding, with INR 250,000 in government funding; integrated ArduPilot, a Jetson Nano and RTK GPS, and solved suspension and drivetrain control problems through kinematic analysis and testing.

Technical Skills

Robot Interfaces:	`ros2_control`, CAN (SocketCAN; Robstride and Damiao MIT-mode protocols), RS232, RS485/Modbus, ESP32 firmware, real-time Linux (PREEMPT_RT, SCHED_FIFO)
Kinematics & Control:	Closed-form and numerical IK (Eigen, KDL, TRAC-IK), PD and feedforward control, gravity compensation, joint limits, MoveIt 2, TF2, URDF/Xacro/SRDF
Teleoperation & XR:	Meta Quest 3/3S (Unity/C#, OpenXR), 6-DoF SpaceMouse, joystick arbitration, deadman gating, UDP and WebSocket clients, point-cloud streaming, record/replay
Safety & Diagnostics:	Heartbeat and watchdog failsafes, E-stop and velocity guards, torque-residual collision detection, ROS 2 diagnostics, Grafana Cloud, gtest, multimeter and serial-console debugging, ISO 14971
Perception & Learning:	OAK-D stereo depth (DepthAI), on-camera YOLOv8, OpenCV, camera-to-base extrinsics; LeRobot, PyTorch, Hugging Face Hub; ACT, Diffusion Policy, SmolVLA, π_0, $\pi_{0.5}$
Data & Software:	MCAP, Foxglove; Python, C++17, TypeScript/React, FastAPI, Linux, Git, GitLab CI, Docker, CMake; NVIDIA Jetson (Orin, Orin NX, Nano), Isaac Sim, Gazebo

Education

University of Ottawa, Ottawa, ON **2024 – 2026**
Master of Engineering in Electrical and Computer Engineering

Gujarat Technological University, Gujarat, India **2019 – 2023**
Bachelor of Engineering in Mechatronics Engineering