WHITEPAPER

ROBOTICS RESEARCH IS SIMPLIFIED AND ENHANCED WITH A NATIVE, ROS-BASED SYSTEM

TORMACH®

ROS: INTRODUCTION

ROS (Robot Operating System) is a powerful open-source platform for robotics research, but until recently it lacked industrial-quality hardware that is tightly integrated with the ROS software stack. Robot equipment manufacturers use proprietary, closed-source software and control systems for their manipulators, leaving researchers with a steep hill to climb in order to use ROS on industrial robots.

Addressing this need and advancing the capabilities of the ROS development community, Tormach has created a ROS-based industrial robotic manipulator and control system that avoids "black box" issues that plague modern robotics applications. Additionally, Tormach's control system, PathPilot®, uses Python as the robot programming language, creating an intuitive programming interface for robot motion and unleashing the potential of the Python package ecosystem.

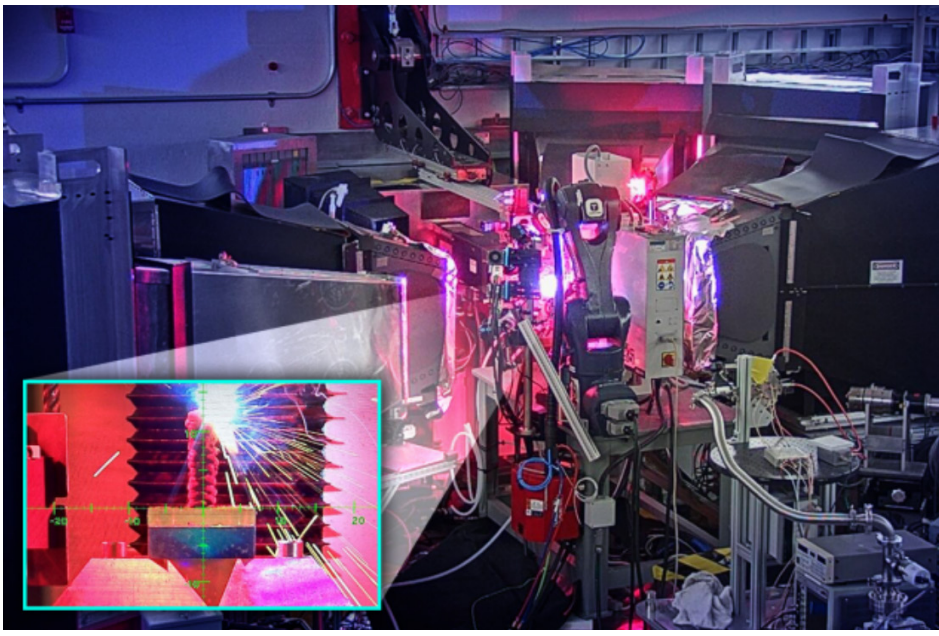
This open-source, ROS-based robotics platform — which includes the control system, industrial robot hardware, and full access to all system parameters — creates a fast, accessible solution that brings industrial robotics to more researchers, developers, and students.

OAK RIDGE NATIONAL LABORATORIES USING THE TORMACH ZA6 ROBOT

Tormach's control, called PathPilot®, provides a simple-to-use operator interface. Source code is public on GitHub for review, use, or modification, delivering a level of flexibility not typically offered by industrial controllers. Instead of trying to force a controller to work for a specific application, users can alter PathPilot® ROS as they see fit.

Oak Ridge National Laboratories (ORNL) are using the Tormach ZA6 to 3D print metal on a rotating platform, which is then measured microscopically with a beam of neutrons. The research is highly relevant to using additive manufacturing in the aerospace and automotive industries.

See how **Oak Ridge National Laboratories uses Tormach robots to print metal in their 3D metal additive work cell**.



The OpeN-AM experimental platform, installed at the VULCAN instrument, features a Tormach ZA6 robotic arm that prints layers of molten metal to create complex shapes. Studying the 3D-printed welds microscopically with beams of neutrons allows researchers to better understand factors such as stress caused by heating and cooling. The experiments will help to optimize the fabrication technique for more mainstream use. (Credit: ORNL/Jill Hemman)

THE CHALLENGE

THE PROBLEM WITH ROS AND PROPRIETARY ROBOT CONTROL SYSTEMS

Robot control manufacturers are hesitant to allow ROS developers to access all the system parameters in their closed-source controls for the following reasons:

- They have invested significant resources into developing and testing their proprietary control systems and don't want to expose the inner workings of their system to external researchers for fear of losing intellectual property
- There are risks associated with exposing the system parameters to external researchers. Untested usage may introduce bugs or otherissues that could compromise the safety or reliability of the system
- Legal or contractual obligations often prevent the manufacturers from sharing proprietary information with external parties
- They may be concerned about potential liability issues if their closed-source control systems are modified by external parties
- There is little financial incentive for most manufacturers, and in many cases there is a strong disincentive: the need to invest in additional documentation, training, and support infrastructure to enable researchers to work with their control systems effectively

For these reasons, integrations between ROS and commercially available robot hardware are limited. While drivers exist to connect ROS to other industrial robots, their low (10 - 100Hz) bandwidth implementations simply drip-feed waypoints to a proprietary, closed-source controller.

As a result, the user may not have access to whether or not the robot adheres to timing, velocity, and path accuracy intents. Data like motor torque, current, following error is usually unavailable, and the slow control loop severely limits what researchers can accomplish.

MOTOR AND DRIVE FEEDBACK TO ROS

The ROS/HAL hardware and software stack offers feedback that can provide valuable control opportunities.

The ZA6 provides the following:

- Feedback from each joint, standard configuration:
  - Position, velocity feedback
  - Torque feedback in SI units
  - Following error
  - Diagnostics-like error code
  - With configuration, drives can also report additional diagnostics like motor/encoder temperatures and error code history
- 10 digital inputs + 12 digital outputs (one digital input usable as probe input)
- HAL can report RT latency
- Feedback from ROS and MoveIt, especially Cartesian pose

Most of these feedback elements focus on only the lower-level control layers. Higher-level control layers can provide other opportunities, depending on research needs.

THE SOLUTION

HAL:
THE OPEN-SOURCE HARDWARE ABSTRACTION LAYER

The connection between ROS and a robot's hardware relies on a hardware abstraction layer (HAL). HAL evolved out of the open-source Enhanced Machine Controller (EMC) project that had its origin 25 years ago at the National Institute of Standards and Testing (NIST).

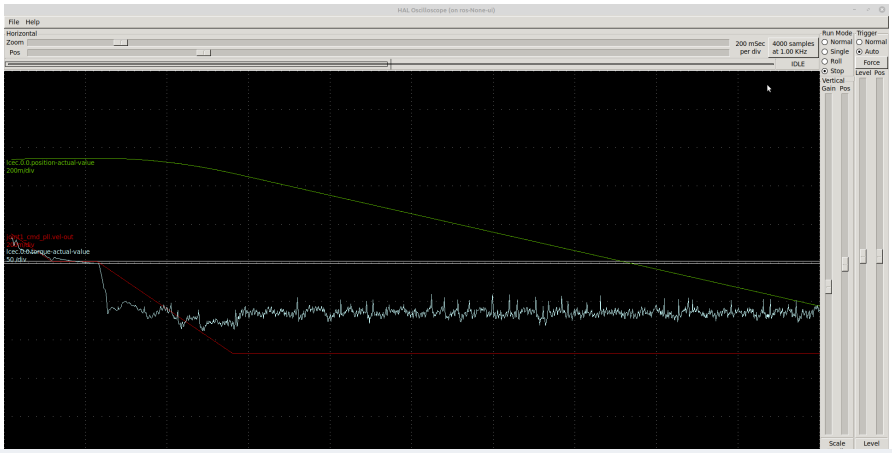
Active development of HAL continues today via the LinuxCNC and Machinekit projects because HAL is flexible, 100% open-source, and is used in thousands of machines around the world.

HAL consists of modular components (loadable binary modules) that communicate with each other by updating, reading, and writing named pins that connect via named signals. In some ways, HAL is like ROS, but there are important differences:

- Using PREEMPT-RT Linux extensions, HAL components written in C execute in a 1kHz realtime thread with minimum jitter
- HAL has many pre-written components design for low-level hardware control (PWM generators, stepper driver step generators, BLDC and three-phase motor controls, and more. **A full list can be found here.**

The Tormach robot bridges the gap between ROS with the **open-source hal_ros_control component**. The combination of HAL and ROS allows a wealth of robot data to be exposed to the user. All process data is accessible via shell commands, data logger utilities, and a graphical scope. All information on the EtherCAT bus, including torque, current, following error, position, velocity, and more are available at 1 kHz and exposed via HAL to ROS.

Since HAL is modular and flexible, users can alter their robot's HAL configuration using pre-built HAL components or by writing new components in C or Python, allowing easy integration with almost any external device or process.



HALScope, and included software oscilloscope, shows a trace of joint 1 position feedback and torque during a move. Any system parameter, including torque, position, following error, and many more is available at 1kHz to the user.

PRECONFIGURED FOR ROS

Previously, using a commercially available robot with ROS requires finding and downloading the appropriate driver for the control, a URDF file to describe kinematics; creating a moveit configuration, choosing one or more planners, IK solvers, perhaps finding and bringing solid models into Rviz. Configuring a new robot for use with ROS is challenging even for experienced ROS developers.

An optimized default ROS configuration for the manipulator, like that provided by Tormach as part of the control, helps alleviate many of these issues. The URDF model (unified robot description format) is defined, motion pipelines are configured, and trajectory planners and kinematics solvers are selected and optimized, so that the robot is ready to work out of the box.

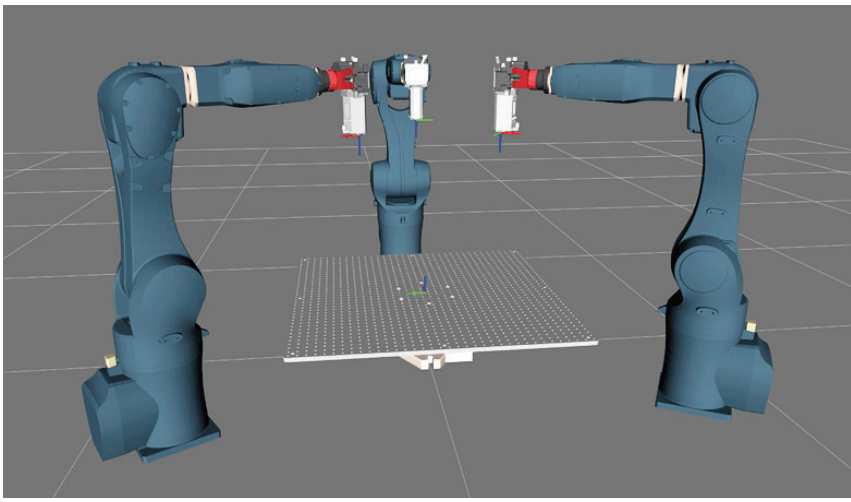
The robot hardware, user interface, and robot programming language are fully documented and supported by Tormach. Go here for documentation: **tormach.atlassian.net/wiki/spaces/ROBO**

The robot's default ROS configuration will be ideal for most applications, saving months of configuration time, and it's also open-ended to allow users to develop their own unique configurations at will.

GEORGIA TECH BUILDS 3D PRINTER WITH MULTIPLE ZA6 ROBOTS

A PhD student at Georgia Tech developed a multi-robot wire-arc/ polymer 3D printer using the ZA6.

See how Georgia Tech employs three Tormach robots in a collaborative multi-robot 3D printing cell.



PYTHON:
THE ROBOT'S PROGRAMMING LANGUAGE

The lack of an industry-standard robot programming language led Tormach to choose Python for its ZA6 robot. The Tormach Robot Programming Language (TRPL) uses the Python 3 interpreter and works similarly to other common robot programming languages, with commands for different move types, commands to read and set inputs and outputs, commands to set and change tool and user frames.

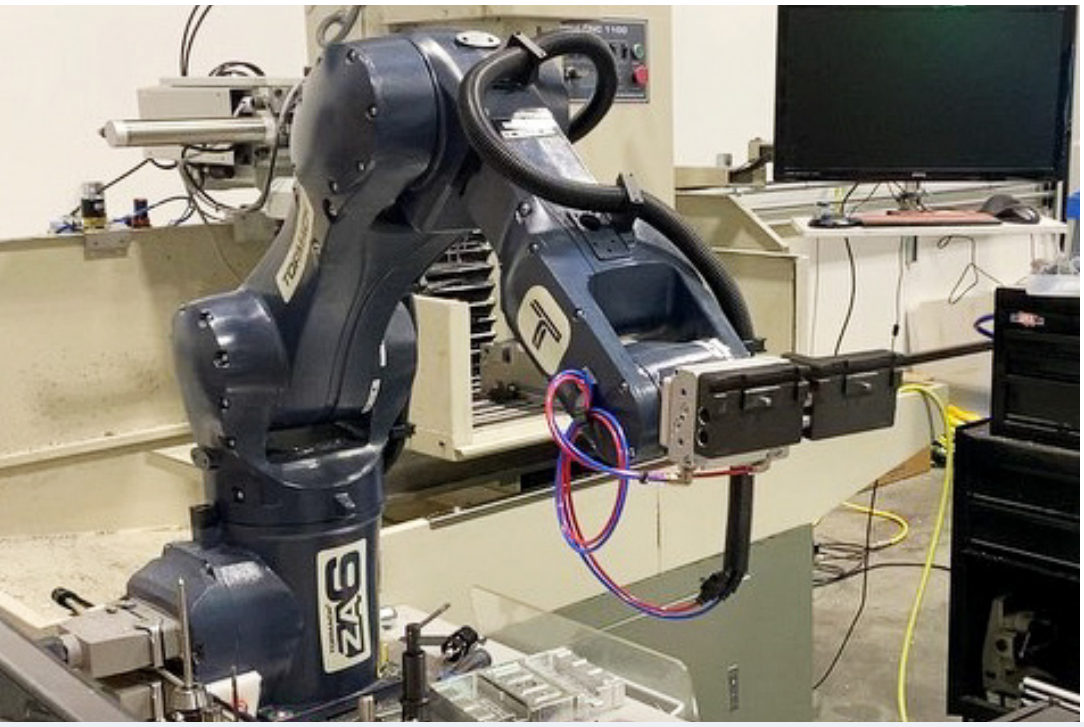
The language is documented here.

It is important to note that any Python 3 program is a valid robot program. The robot's ability to interpret any Python program means that almost any Python package can be imported to help more challenging robot tasks. Examples include:

- Using the csv and http requests libraries to upload data files recorded by the robot to a web server
- Using opencv to recognize ArUco markers for visual servoing and localization
- Using numpy and kdl to calculate forces in cartesian space from the joint torque feedback and robot Jacobian
- Using Twilio to send text messages from the robot
- Using ChatGPT and the Python OpenAI API to conversationally create robot programs, **example here.**

While the TRPL interpreter simplifies a lot of programming tasks like move commands and offsets, power users who are familiar with ROS are able to access the underlying ROS API directly.

Find more information here.



Python, a programming language that's been in use for decades, makes programming a Tormach robot accessible for many.

The sheer number of devices and softwares that are run on Python mean the ZA6 has a seemingly countless number of integrations that are possible.

SUITABLE FOR RESEARCH AND EDUCATION

The PathPilot® user interface makes it easy to write simple teach-mode programs to help students learn the concepts they need to be successful in industrial robotics. Unlike other robots designed for the classroom, the ZA6 teaches industrial robot concepts like user frames, tool frames, waypoint programming, and Cartesian-versus-joint angle waypoint types. Another reason to use the robot as a teaching tool is its easy-to-learn user interface.

The PathPilot® user interface makes it easy to write simple teach-mode programs to help students learn the concepts they need to be successful in industrial robotics. Unlike other robots designed for the classroom, the ZA6 teaches industrial robot concepts like user frames, tool frames, waypoint programming, and Cartesian-versus-joint angle waypoint types. Another reason to use the robot as a teaching tool is its easy-to-learn user interface.

Dr. John Wen at Rensselaer Polytechnic Institute is developing Robot Raconteur®, which is a royalty free project intended to provide a solution to distributed control and component interfaces. The system is designed precisely for the scenario of an engineer wanting to control a component from a high level language in distributed or non-distributed conditions.

See how researchers at RPI have integrated the Tormach robot into its Robot Raconteur program for force-torque control.

THE ZA6 SHINES AS AN EDUCATIONAL TOOL

Leif Sorgule, tech educator teacher in Peru, New York, released a **set of educator-focused projects**.

Leif Sorgule teaches engineering and technology in the Peru Central School District in Peru, New York.

TORMACH IS A DIFFERENT OEM

While the Tormach organization is large enough to provide a variety of CNC and automation equipment, the organization is and always has been built around a community mindset. Tormach support is U.S.-based, and our engineers and support staff participate in web forums that house an array of valuable content.

Tormach is also a participating member of the ROS-Industrial Consortium and has long been a supporter and contributor on open source projects.

Thanks to open source, ROS-based robotic resources, it has never been easier to teach and develop industrial robot applications.

To learn more about the Tormach PathPilot® controller, TRPL programming language, and ZA6 manipulator, **please visit our website.**

Tormach® is a member of the ROS Industrial Consortium, committed to empowerment in the open-source community.

Advance Concrete of Madison WI set up their ZA6 as a welding robot. Despite having no prior robotics or programming experience they were able to develop their process quickly and triple output relative to their previous, manual process.

Learn more here.