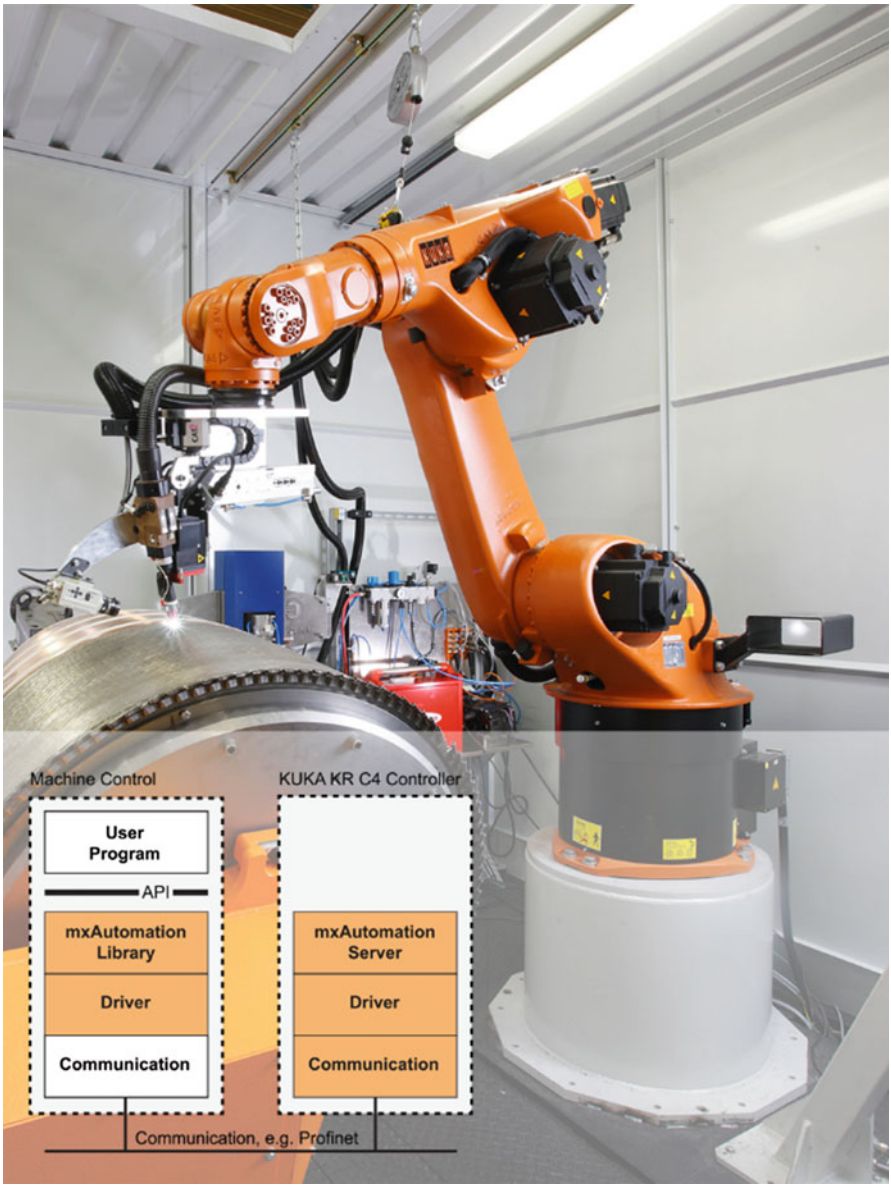
Machine Control
KUKA KR C4 Controller
User Program
API
mxAutomation Library
Driver
Communication
mxAutomation Server
Driver
Communication
Communication, e.g. Profinet

Direct Robot Control with mxAutomation: A New Approach to Simple Software Integration of Robots in Production Machinery, Automation Systems, and New Parametric Environments

Heinrich Munz, Johannes Braumann and Sigrid Brell-Cokcan

Abstract The industry-crises of the past have made it clear how existentially important it is to have flexible, "living" production facilities. Automation by means of industrial robotics has proven to be a key technology in this regard. However, truly dynamic processes can only be achieved when the robots and the environment to be automated—machines, handling equipment, etc.—are perfectly integrated, both operationally as well as from the operator's perspective. KUKA's mxAutomation interface now allows a granular remote operation of the robot in interaction with modern industrial real-time communication—and beyond that also entirely new, flexible workflows from design to production towards fabricating highly customizable products in the creative industry.

Keywords KUKA · Direct robot control · Remote operation · Mass-customization · Robotic arms

1 Introduction

Robots are not an end in themselves but fulfill a wide variety of tasks that contribute to an overall automation solution. In industry they are used e.g. for the preparation or post-processing of tools or work pieces of a production machine, or for handling tasks, such as loading and unloading.

H. Munz (✉)
KUKA Roboter GmbH, Augsburg, Germany
e-mail: Heinrich.Munz@kuka.com

J. Braumann
Association for Robots in Architecture, University for Arts and Design Linz, Linz, Austria
e-mail: johannes@robotsinarchitecture.org

S. Brell-Cokcan
Association for Robots in Architecture, RWTH Aachen University, Aachen, Germany
e-mail: sigrid@robotsinarchitecture.org

D. Reinhardt et al. (eds.), *Robotic Fabrication in Architecture, Art and Design 2016*, DOI 10.1007/978-3-319-26378-6_35

Fig. 1 KUKA KR16 use-cases in the creative industry: Tripod-mounted as part of an art-installation by Conrad Shawcross (*left*), milling wooden elements with an entry-level spindle (*right*)

While robots have been used for such purposes for decades, it has only been a few years that accessible visual-programming environments coupled with dynamic components for robot control have also opened up the use of robotic arms to the creative industry where they are now being utilized for a huge variety of tasks, from fabrication and assembly to interactive installations and performance art (Fig. 1).

Today, plugins such as KUKA|prc (Braumann and Brell-Cokcan 2014) allow architects, artists, and designers to work from design to fabrication within a single visual programming environment, where they can not only define an object's shape and form, but also visually assemble robotic processes and immediately simulate the robot's movements, automatically checking for singularities and unreachable positions. Rather than dealing with the intricacies of a robot control data file, movements and other commands are assembled in a visual system that allows fine-grained control over most properties such as speed and interpolation. The automatically generated KRL (KUKA Robot Language) code can then be immediately copied to the robot, enabling a rapid turnaround time as well as an efficient feedback loop. Today, also users of other industries use KUKA|prc to quickly prototype processes, e.g. in the aeronautical industry.

However, an issue with such a file-based workflow is the delay between writing the file and executing it. While KRL files representing hundreds of design variations can be generated within a fraction of a second, they have to be manually copied onto the robot's command execution memory before they can be executed.

Ideally, to fully utilize the potential of accessible visual programming environments, we would need a direct link between PC and robot, through which we can then dynamically "stream" commands to the robot. A similar challenge can be observed within the context of Industry 4.0, where one of the main ideas is to enable machines to communicate with each other in order to form smaller, functional units, rather than just hierarchical structures with a central control unit.

New industrial solutions and interfaces by KUKA thus enable communication between automation machines or external controllers of any kind and robots. We expect that building upon these interfaces will also lead to entirely new and flexible workflows for direct robot control in the creative industry.

2 Control Systems and Communication in Industry

In order to coordinate the operation of external controllers and robots, communication is necessary between their control systems. Originally such tasks were fulfilled by signals wired in parallel, until conventional field buses took over these functions.

With the emergence of modern Ethernet-based real-time field buses and Time Sensitive Ethernet Networks (IEEE 802.1 TSN), the technological basis for efficient communication between external controllers and robots was established. An important characteristic of such new communication means is their ability to run several protocols and thus services via the same Ethernet cable or to operate multiple buses at the same time. In addition to the standard field buses EtherCAT, Profinet and Ethernet/IP, the new KUKA KR C4 robot controller also optionally supports standard UDP communication over Ethernet.

Based on these improved communication mechanisms, the machine builder or system integrator can solve automation tasks as usual by creating separate machine programs, robot programs and the corresponding operator screens for the machine and the robot separately and putting them into operation by using self-defined bits and bytes on the communication means. For this the builder or integrator, and any other participants, such as service personnel, require detailed knowledge of robot programming.

In order to keep this learning curve lower, and to be able to program the movements of the robot and the machine at just one location, KUKA has created mxAutomation (KUKA Robotics 2014) so that machine builders or system integrators can integrate the KUKA robot controller into their remote machine controller. All programming and operator tasks for the robot can then be done exclusively from the machine controller and its operator panel. Special knowledge of robot programming is no longer necessary.

mxAutomation consists conceptionally of two main parts (Fig. 2). A server program created by KUKA runs on the robot controller, which waits for commands to arrive via one of the field buses or UDP-Ethernet mentioned above. The actual robot control program runs on the machine controller; this program is created in the programming language and with the programming methods of the machine controller. In order to make this possible, KUKA provides the mxAutomation client library for several controllers, which is integrated by the machine firmware programmer into the firmware. The programmed robot commands and their parameters are put into the defined data format and finally streamed via the communication means to the mxAutomation server running on the robot controller. The server

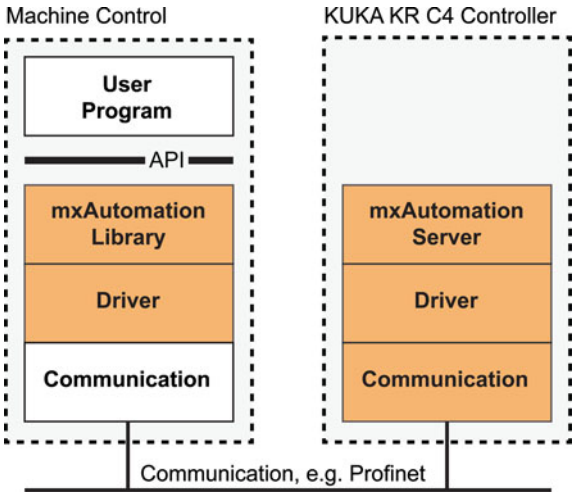

Fig. 2 mxAutomation control concept. Software components provided by KUKA are marked orange

interprets the data, executes the desired robot commands and sends return parameters, status messages, etc. back to the mxAutomation library on the machine controller. In order for the data transfer of the robot commands and their return values to take place as quickly as possible and deterministically, cyclical process data of the particular field bus are used exclusively for this. Several robots can be simultaneously remotely controlled by a single machine controller.

The mxAutomation library has been created in portable Structured Text and is available in adapted form for Siemens Simatic, and based on that for Sinumerik 840d, as well as for Rockwell PLCs and CodeSys or ProConOS systems, with more to be added in the near future. Through an adaptation kit other controller manufacturers can integrate the mxAutomation library into their controllers themselves after concluding a cooperation agreement with KUKA.

The Application Programming Interface (API) of the mxAutomation library is based on the programming paradigm of PLCOpen's Motion Control Function Blocks (MCFBs) Part 4 and has been specifically expanded for KUKA robots. Virtually all functions which can be directly programmed on the robot are also available via the mxAutomation library. This includes general functions, such as reading of the current robot position, speed and acceleration, and also functions such as reading and writing local robot I/Os and variables. An application example for the advantageous use of system variables is sensitive gripping of work pieces or packaged goods, where the use of axis-specific torque limitation is a proven means. Within mxAutomation, the system variable TORQMON_DEF like several other robot system variables can be assigned to an mxAutomation variable, which can then be written via the KRC_SetSysVar function.

Corresponding blocks are of course also available for the axis-specific or Cartesian motion of the robot with LIN, PTP or CIRC. Approximate positioning is also possible, as are exact time-distance functions or interrupts.

3 Building upon mxAutomation

In fall 2015, the client library in the "adaptation kit" (see Sect. 2) has been enhanced by additional programming languages such as C#, and C++. As such, mxAutomation is no longer depending on a particular brand of controller, but could be run on basically any computer, from an embedded controller, a compact SoC, a smart device to a regular laptop or PC, even from the cloud.

Building upon KUKA's mxAutomation C# client sample library as well as Robots in Architecture's KUKA|prc framework we created a custom client software that communicates with one or more connected KUKA robots running the mxAutomation server software. First, the native KUKA|prc commands are processed, and—rather than being written into a *.src file—mapped to their corresponding C# classes in the mxAutomation library, so that e.g. a KUKA|prc LINear Movement becomes a KRC_MoveLinear-Absolute object.

Once the server is up and running on the robot controller, data are continuously exchanged as UDP datagrams between the PC running KUKA|prc and the robot controller at the robot's sub cycle time of 4 ms. Command blocks such as the aforementioned KRC_MoveLinearAbsolute are executed in sequence and either put into the robot's command buffer, or immediately return status values from the robot, such as current robot positions, the read of variables or I/Os.

By design, the presence of a buffer makes mxAutomation usable even for non "hard" real-time applications running i.e. on Microsoft Windows, while at the same time making it highly resistant against communication timeouts as long as the buffer is sufficiently filled. This is especially important for regular PCs, where Windows is not real-time capable and processes such as garbage collection may cause timeouts of more than several 10 s of milliseconds.

Fig. 3 Dynamic stone surface structuring utilizing the mxAutomation interface to allow dynamic adjustments of angle and force during the structuring process

Using task-based programming via async/await we created a resilient, high-performance mxAutomation controller and implemented it into Grasshopper. Thus, the user can first simulate a robot program via KUKA|prc and then simply reconnect the commands to the mxAutomation component in order to stream them to the robot. The interface is already utilized within the scope of the AROSU research project (Fig. 3) (Brell-Cokcan and Braumann 2015).

4 New Interaction Strategies

A communication process between an automation machine and an industrial robot is usually quite straightforward, with the machine instructing the robotic arm to perform a certain task and waiting upon its completion. However, in the context of the creative industry one of the main appeals of direct robot control is to utilize the feedback of the robot to inform the process. As such, we have identified and implemented four different dataflow-modes for mxAutomation (Fig. 4):

- **Default Mode** uses a rather large buffer to store and process commands. When the input data of the mxAutomation component changes, the robot's buffer is wiped and the new commands are streamed for immediate execution.
- **Adaptive Mode** relies on a much smaller buffer with the goal of keeping data parametric as long as possible. New data does not cause the process to restart, but rather adapts the existing commands, which only become static once they are placed in the robot's buffer.
- **Iterative Mode** waits for the completion of a set of commands before accepting new inputs, so that the returned data can be used to inform the next fabrication step.

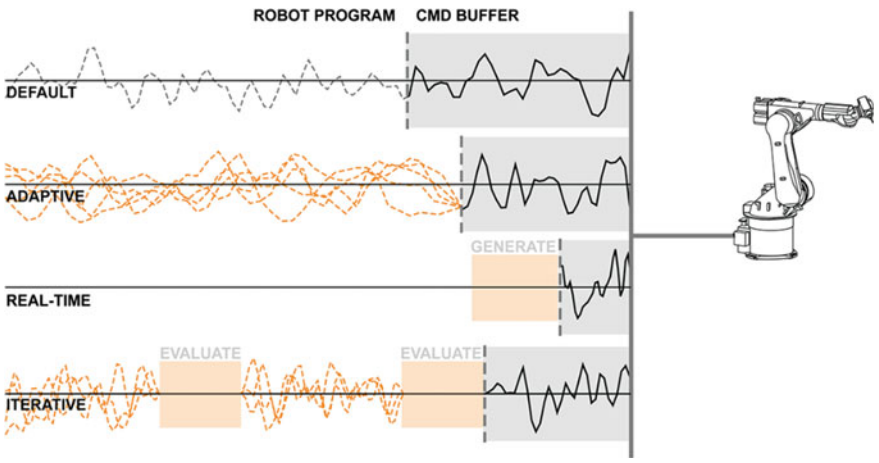

Fig. 4 mxAutomation data flow strategies (Braumann and Brell-Cokcan 2015)

- **Real-time Mode** does not achieve hard real-time, but rather uses a shallow buffer. Commands are streamed stepwise towards the given (and possibly moving) target position.

Together, these four modes enable a wide range of applications, from straightforward fabrication, to iterative assembly processes that are continuously evaluated and fully interactive installations.

5 Analysis and Outlook

mxAutomation's approach provides several advantages over other solutions, where the machine controller completely takes over control of the robot's drives or motors. Most importantly, mxAutomation works straight away with all KR C4-based industrial KUKA robot types—from KR AGILUS to the KR 1000 Titan. There is no risk that the machine controller, due to lack of knowledge of the mechanical limits of robot motors, gear units and mechanical components, will exceed these load limits, resulting in mechanical damage to the robot. Additionally, the machine controller retains all advantages and features of the KUKA robot controller, such as energy-efficient motion algorithms adapted for the specific robot arms, loads and moments of inertia, sophisticated exception handling routines or the entire safety functions.

mxAutomation has proven itself to be a simple and efficient tool and an end-user-friendly robot integration system within a wide variety of different environments. Its open approach and the availability of the mxAutomation library in many programming languages enable completely new applications, as demonstrated by its quick integration into KUKA|prc, where it now provides a direct and immediate interface between design and fabrication.

mxAutomation enables a new way of system thinking as it is necessary for Industry 4.0 or the Internet of Things: not only the robot arm (a component) should be in the focus of the user but the whole and integrated system (the solution).

References

Braumann, J and Brell-Cokcan, S 2014, 'Visual Robot Programming—Linking Design, Simulation, and Fabrication' Proceedings of the 5th Annual Symposium on Simulation for Architecture and Urban Design (SimAUD).

Braumann, J and Brell-Cokcan, S 2015, 'Adaptive Robot Control', Proceedings of the 33rd eCAADe Conference, Vienna, Austria.

Brell-Cokcan, S and Braumann, J 2015, 'Towards Adaptive Robot Control Strategies', Proceedings of the 35th ACADIA Conference, Cincinnati, USA.

KUKA Robotics, 2014, CODESYS Library for KUKA.PLC mxAutomation 2.0.