6

ROBOTICS / RESPONSIVE ENVIRONMENTS 1

TOWARD ADAPTIVE ROBOT CONTROL STRATEGIES

Sigrid Brell-Cokcan
Robots in Architecture
RWTH Aachen University

Johannes Braumann
Robots in Architecture
University for Arts and Design Linz

ABSTRACT

The use of robots in the creative industry is becoming increasingly widespread. While cutting edge research attempts to apply new, architectural concepts to industrial processes using high-end software and hardware, affordable, used robots are opening up exciting new possibilities for small businesses and creative individuals. In this research we explore KUKA|prc as a modular, adaptable tool that can be applied towards both purposes. Building upon the visual programming environment Grasshopper, our software allows the user to program, simulate, and control robotic arms. As such, all the program logic is open to the user, making it possible to optimize programs for older robots by e.g. reducing the density of programs, or directly importing G-code from regular CAM (Computer Aided Manufacturing) software. At the same time, we are developing a KUKA|prc-compatible interface that is capable of directly controlling a robot from basically any PC, based on a software library that was initially developed to link robots with CNC controllers. While the strategies for older-generation robots allow users to fully realize the potential of their machines, we expect that adaptive robot control will open up entirely new approaches for interactive, parametric production processes.

INTRODUCTION

Within just a few years, industrial robots have become a significant field of research within the creative industry. Due to their inherent multi-functionality they are being used for a wide range of applications, from conceptualized ideas of human-robot interaction, to interactive media (refer, for example, to the work of Bot&Dolly; Byrne et al., 2014) and full-scale fabrication (Menges, A., J. Knippers, V. Schwieger 2014). While the idea of utilizing robotic labor has been appealing to creative minds for decades, the past decade has seen several key developments that have actually brought robots first into research, and now even into architectural practices such as Snøhetta *(Figure 1, left)*.

A very significant enabler of this process has been the price, which steadily decreased over the past decade (IFR 2006) towards a point that made robots reasonable investments for research institutes and larger practices. In parallel the prices for used robots have fallen even more drastically, finally enabling even individual artists and designers to purchase robots and realize large scale projects in their garages (Johns and Foley 2012) or ateliers (Neugebauer and Kölldorfer 2012) *(Figure 1, right)*.

Figure 1
Robot setup at Snøhetta during a KUKA|prc workshop (left); Red Bull Arch at the Formula 1 track in Austria (right). Eighty-three positive molds for the aluminum arch fabricated with a used robot via KUKA|prc and Grasshopper.

Another key factor with at least the same significance as affordability has been developments in the area of software interfaces. The most common way of programming a robot is and has been "teaching", where positions are saved and replayed directly at the robot. As early as in the mid-1990s, researchers utilized "generic" software such as MatLab (Corke et al. 1995) to program and simulate robot-movements offline. Nearly twenty years later, visual programming environments—most notably Grasshopper by David Rutten—are allowing the creative industry to define complex parametric relationships in an accessible, relatable way. While originally intended mostly for geometrical operations, the approach of defining parametric, robot movement strategies via visual programming proved to be a powerful combination (Brell-Cokcan and Braumann 2010; Schwartz 2012). Instead of being limited to specific commercial software, the creative industry is now able to define its own robotic processes in a scope that goes beyond regular CAD-CAM.

In this ongoing research we explore how parametric robot control can be used at two very different levels, from fabrication with affordable, used robots to cutting-edge machine networks. For both approaches we build upon our software tools KUKA|prc—parametric robot control—to program, simulate, and control robotic processes (Braumann and Brell-Cokcan 2014).

NEW DEVELOPMENTS IN THE ROBOTIC INDUSTRY

It is interesting to observe that the most notable developments in the "core" business of industrial robots (i.e. heavy-payload robotic arms) can be found in the area of software, rather than hardware. As such, the key parameters of a KUKA KR150-2 from the year 2000 and a current KUKA Quantec KR150R2700 differ only slightly, with the new robot being about 10 /s faster but having the same (ISO 9283) repeatability of ±0.06 mm.

While the improved speed, along with advances in energy efficiency, are highly significant for e.g. the automotive industry, they are not as relevant in the creative industry, where the lower price per produced unit often does not pay for a slightly improved cycle time. Just as well, robotic arms are still not accurate and fast enough for several engineering applications, while they surpass most criteria that are posed by the creative industry: construction site tolerances are measured in millimeters or centimeters rather than micrometers, and the human factor has got a much bigger impact on cycle times than in fully automated factories.

Therefore one of the main efforts within the creative industry is to move beyond current limitations in regards to programming, rather than in regards to mechanics. This leads to the situation that previously used, affordable robots would be highly capable machines for the creative industry, but require research in the area of software to compensate for some shortcomings, as most recent software add-ons from the manufacturers are only available for the newest-generation of robot controllers.

OUTSOURCING COMPUTATION

Offline robot programming in Grasshopper has allowed the creative industry to develop new workflows for production-immanent design and to create entirely new fabrication strategies based on accessible programming tools such as KUKA|prc. However, another significant advantage towards a sustainable "recycling" of used robotic technology is that offline programming allows us to use the comparably cheap computational power of regular PCs and laptops rather than the much more expensive and limited robot controller. Thereby, the entire computation-intensive geometric logic is placed within the visual programming environment, and only a lightweight, static robot control data file containing a simple series of movement commands is sent to the robot. In our initial research we have identified three exemplary applications that greatly benefit from this approach:

Reducing Point Density. One of the core limitations of older robots is their limited memory and processing power, which limits both the maximum allowed file size, as well as the process speed in the case of excessive point density. We have, therefore, implemented functionality to split files into multiple parts and, more importantly, a way to reduce the density of robot positions. Compared to polylines, reducing a list of robot positions requires more elaborate calculations as the one has to consider not only the Cartesian position of the tool center point, but also the full spatial orientation of the tool and any peripheral changes that may take place at the same time. Thus our custom scoring algorithm analyzes each position and then decides if it can be reduced or not.

CNC Import. Instead of directly converting 5-axis G-code on the robot controller (e.g. KUKA.CNC), we have implemented a similar process in Grasshopper which reads G-code and turns it into native robot movement commands, which can then be simulated (and reduced) within Grasshopper. This approach works for regular CAM software as well as for more specialized code such as Slic3r 3D-printing toolpaths. While such a process is also possible with commercial software such as KUKA CAMRob/SimPro, the integration into a visual programming environment allows us to combine static data with parametric data, while also saving costs, seeing that such postprocessors can cost more than a used robot.

New Robot Movements. Spline movements are one of the more recent movement types that are supported by KUKA robots and can smoothly blend a sequence of points into a continuous curve, leading to more efficient toolpaths. While some aspects—such as the optimized speed interpolation—are exclusive to the spline movements, we can use visual programming to first interpolate a curve through the set of given points and then fit arcs and lines in order to get a series of circular and linear movements. Thus, we get an optimized toolpath that allows us to represent even complex NURBS curves with a greatly reduced memory footprint on older machines *(Figure 2)*.

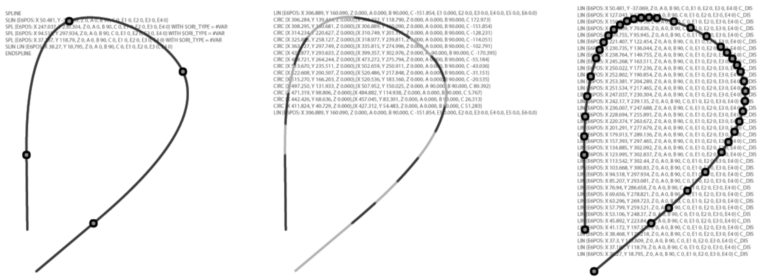

Figure 2
Spline movement (left), fitted arcs and lines (middle), optimized linear movements (right). Identical tolerance settings were used for arc and polyline fitting.

MACHINE NETWORKS

The easy availability of accessible robot programming software and used machines has enabled exciting projects that would otherwise not have been realized. As such, industry users are now becoming increasingly aware of these new programming paradigms while creative users want to utilize "industrial" automation strategies. These requirements have led to a new, ongoing investigation into the "streaming" of control data using industry-tested interfaces for highly adaptive robotic processes. By doing so, the entire logic is placed within an external control unit, making it possible to get away from the

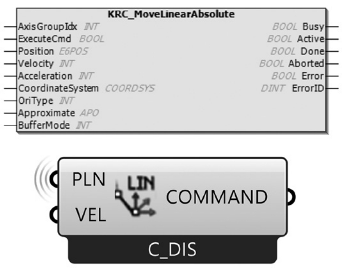

Figure 3
Similarities between Codesys and Grasshopper visual programming: PLC function block for a linear movement in accordance with IEC 61131, internally mapped to a Grasshopper component (above); (KUKA, 2014), KUKA|prc component within Grasshopper (below).

necessity of turning dynamic visual algorithms into static code, rather streaming it in real-time and concurrently adjusting core parameters.

In industry, there is a range of fieldbus systems that can be used to control robots such as EtherCAT and Profibus, which are very low-level and not easily deployed. On a higher level there are programmable logic controller (PLC) systems such as Sinumerik and Rockwell, which are commonly found in CNC machines. KUKA developed their mxAutomation software-add-on so that regular KUKA robots can be directly controlled by these PLCs without requiring additional hardware or even deeper robot knowledge from the operator.

The KRC4 controller, therefore, acts as an interpreter that accepts buffers and executes commands from the PLCs. Currently KUKA is working on expanding the scope of interfaces to also include generic UDP packets. While this may seem like a small change, it actually opens up huge possibilities as now basically every computer with an Ethernet port can act as a PLC to control any kind of KUKA robot. As a proof of concept, building upon a pre-release version of mxAutomation *(Figure 3)* and our existing robot control and simulation framework, we created a range of custom components for Grasshopper that act as a "soft-PLC" and interface with the interpreter running on the robot. As systems such as Codesys also build upon visual programming, the translation of the logic between systems is facilitated *(Figure 3)*.

The interpreter accepts each command, places it inside a buffer and then executes it in sequence, exactly as if they were written in regular KRL, i.e. with all safety measures active and regular robot parameters in place. At every cycle the robot also returns a range of customizable values to the controller such as the current position, sensor values, axis speeds, etc. This allows us to rely on a single interface to send and receive data to and from the robot. The only real limitation is that a smooth movement always requires at least two positions to be placed in the buffer, otherwise the robot would just go to a position, stop, and then approach the next position. Therefore, "hard" real-time with just a few milliseconds of reaction time (refer e.g. to Schreiber et al. 2010) is not possible with that approach.

In this current stage we have defined and integrated four different interaction strategies that define how data is streamed from the visual programming environment to the robot. In default mode, the server will attempt to store as many positions as possible in the interpreter's buffer to avoid buffer underruns. If the data in the visual programming environment changes, the server will wipe the buffer and continue immediately with the new commands. Iterative mode works similarly, but does not cancel the running job and will only accept new commands once the current set has been processed. This can be used to, for example, process a workpiece, scan the result, and then decide on the next iterative step. Adaptive mode keeps the entire process parametric, with the process-parameters only becoming static once they are passed into a relatively shallow buffer.

When executing a regular KRL file, only the override speed can be adjusted, while the adaptive interface allows changes to parameters such as speed, position, orientation, and more. Finally, real-time mode works similarly to adaptive mode as it keeps a shallow buffer, but positions are created iteratively with the robot moving in a pre-set step-size, e.g. 1 degree and 1mm, towards its target.

At the current stage of this research, the developed adaptive robot control system is being applied for natural stone processing as part of the EU-funded research project AROSU (see AROSU n.d.). It allows us to directly and fluently apply visual programming strategies developed within Grasshopper in an industrial process, without having to manually transfer files. In future research we expect that this new interface will enable us to dynamically adjust processes to changing parameters. Similar approaches are applied in industrial applications where e.g. a force-torque sensor adjusts the pressure of the robot's tool (Garcia 2004). However, rather than just offsetting a position in XYZ, the software will enable us to use the full capabilities of Grasshopper to even radically adjust toolpaths and process parameters *(Figure 4)*.

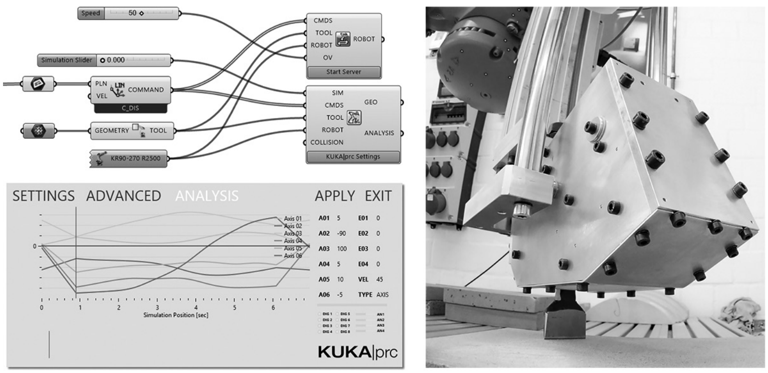

Figure 4
Using visual programming to define and simulate a chiseling process and to stream command data directly to a robot via the same programming environment.

We expect that many future application will result out of this basic framework, especially towards integrating robots into larger machine networks and the internet of things of Industry 4.0.

OUTLOOK

At the current stage, both the "recycling" of robotic technology as well as cutting edge R&D are equally important for the creative industry. Used robots are already highly capable and well-priced as to be widely used in a creative context. On the other side, high-end research in cooperation with the robotics industry will enable entirely new application and hopefully lead to a productive cross-pollination between different disciplines.

In a few years, we expect that new technologies such as the KUKA Sunrise controller, which allows object-based robot programming using Java, will offer even larger degrees of customization, optimization, and ultimately integration into programming systems.

ACKNOWLEDGEMENTS

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n.606453.

REFERENCES

AROSU (Artistic Robotic Surface Processing for Stone). (n.d.). Accessed July 1, 2015. *http://www.arosu.eu*.

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

Brell-Cokcan, S. and J. Braumann. (2010). A New Parametric Design Tool for Robot Milling. In ACADIA 2010: LIFE in:formation, Proceedings of the 30th Annual Conference of the Association for Computer Aided Design in Architecture.

Byrne, K., J. Proto, B. Kruysman and M. Bittermann. (2014). The Power of Engineering, The Invention of Artists. In Rob|Arch: Robotic Fabrication in Architecture, Art, and Design 2014, eds. W. McGee and M. Ponce de Leon. Switzerland: Springer International Publishing.

Corke, P.I. (1995). A Computer Tool for Simulation and Analysis: The Robotics Toolbox for MATLAB. In Proceedings of the 1995 National Conference of the Australian Robot Association, July 1995, Melbourne, Australia, pp 319-330.

Garcia, J.G., A. Robertsson, J.G. Ortega, and R. Johansson. (2004). Sensor Fusion of Force and Acceleration for Robot Force Control. In Proceedings of the 2004 International Conference on Intelligent Robots and Systems (IROS).

IFR (International Federation of Robotics). (2006). World Robotics 2006, Industrial Robots: Statistics, Market Analysis, Forecasts, Case Studies and Profitability in Robot Investment. Geneva: United Nations Publications.

John, R.L. and N. Foley. (2012). Irregular Substrate Tiling. In Rob|Arch: Robotic Fabrication in Architecture, Art, and Design 2012, eds. S. Brell-Cokcan and J. Braumann. Vienna: Springer Verlag.

KUKA Robotics. (2014). CODESYS Library for KUKA.PLC mxAutomation 2.0.

Menges, A., J. Knippers, V. Schwieger. (2014). Landesgartenschau Exhibition Hall: Robotically Fabricated Lightweight Timber Shell. Schwäbisch Gmünd, Germany. Accessed July 1, 2015. *http://icd.uni-stuttgart.de/?p=10631*.

Neugebauer, C. and M. Kölldorfer. (2012). Fabricating the Steel Bull of Spielberg. In Rob|Arch: Robotic Fabrication in Architecture, Art, and Design 2012, eds. S. Brell-Cokcan and J. Braumann. Vienna: Springer Verlag.

Schreiber, G., A. Stemmer, and R. Bischoff. (2010). The Fast Research Interface for the KUKA Lightweight Robot. In Proceedings of the IEEE ICRA 2010 Workshop on ICRA 2010 Workshop on Innovative Robot Control Architectures for Demanding (Research) Applications:How to Modify and Enhance Commercial Controllers. Anchorage, May 2010, p. 15-21.

Schwartz, T. (2012). HAL. In Rob|Arch: Robotic Fabrication in Architecture, Art, and Design 2012, eds. S. Brell-Cokcan and J. Braumann. Vienna: Springer Verlag.

SIGRID BRELL-COKCAN + JOHANNES BRAUMANN

Sigrid Brell-Cokcan and Johannes Braumann founded the Association for Robots in Architecture in 2010 with the goal of making industrial robots accessible to the creative industry. Towards that goal, the Association is developing innovative software tools such as KUKA|prc (parametric robot control) and initialized the Rob|Arch conference series on robotic fabrication in architecture, art, and design that—following Vienna in 2012 and Ann Arbor in 2014—will be held again 2016 in Sydney.

Robots in Architecture is a KUKA System Partner and has been validated as a research institution by national and international research agencies such as the European Union's FP7 program. In 2015, Sigrid set up the Chair for Individualized Production at RWTH Aachen University. Johannes is visiting professor at the University for Arts and Design Linz and heads the development of KUKA|prc.

Their work has been widely published in peer reviewed scientific journals, international proceedings, and books, as well as being featured in formats such as Wired, Gizmodo, and RBR.