Real-Time Adaptive 3D Printing with Robotic Arms

Exploring variability in digital fabrication

Ondřej Cigáník[1], Daniel Sviták[2], Adam Varga[3], Hugo Fekar[4], Samuel Karaš[5]
[1]CTU Faculty of Architecture, [2,3,4]UMPRUM Academy of Arts and Design Prague,
[5]Charles University Faculty of Mathematics and Physics
[1]ondra.ciganik@gmail.com, [2,3,4]{daniel.svitak|adam.varga|hugo.fekar}@umprum.cz,
[5]karass@o365.cuni.cz

This paper describes a system for large-scale 3D printing with 6-axis robotic arms, enabling real-time interaction with toolpaths for adaptive and responsive fabrication. It addresses limitations in current robot control workflows by combining a communication layer with a toolpath adjustment layer that incorporates sensor data or other inputs to influence the printing process. An exploration of existing solutions as well as a Python-based communication script were tested to provide continuous code feeding from Rhino to the robotic Large Factor Additive Manufacturing (LFAM) process. This approach brings the printing process closer to interactivity enabling both creative exploration and functional adjustments. By moving beyond pre-generated toolpaths, this approach enables fabrication processes to adapt to changing conditions during the process of fabrication.

Keywords: *Robotic Fabrication, Interactivity, 3D Printing, Large Format Additive Manufacturing, LFAM*

INTERACTIVITY IN FABRICATION

The motivation for exploration of this concept lies in the idea of real-time data adjustments for robotic 3D printing as a next step that could help with adaptation and user-end-simplification of the larger scale additive fabrication.

Escaping the box-style printer and entering the realm of robotic fabrication, the new challenges seem endless. Among many, adapting the printing process for use on uneven surfaces or existing objects, implementation of broader extrusion control through linking of the immediate robot speed to the extruder speed like the linear advance function of desktop 3D printers. From the list of these challenges, we tried to approach the variability and interactivity of the printing process, as in large scale the process often shifts, and the results can vary.

Modes of interaction

The idea of constant updating of the fabrication process was explored multiple times by teams of ICD-ITKE, but often with a different fabrication strategy, such as glass-fibre placement, weaving, and carbon fibre deposition (ITKE pavilions 2015, 2016). Among other adaptive robotic strategies, updating of toolpaths can be noted in a few experimental additive fabrication projects based on ceramics or concrete (Breseghello, Naboni, 2021).

The printed digital geometry could adjust to the real-time geometry based on the previous extrusion sensing and recalibration, as the larger scale models suffer from cooldown deformation, high temperature sagging, overhang

deformations and other issues (Daedalus pavilion, 2016; The Endless Wall, ETH Zurich, 2011).

Addressing some of these challenges in the context of plastic LFAM could bring higher adaptability of the whole process and uncoupling of the code being uploaded to the robot in a single batch or multiple batches, instead utilizing a data exchange between a PC and the robot control. Access to this type of control has been possible using a custom Python interface and alternatively the Machina application and plugin.

Solid state data vs "liquid" state data

This paper explores the connection between the process of generating live data and its integration within an extrusion-based robotic system.

The concept of live adjustments in 3D printing has been around for some time (e.g., live Z adjust on desktop 3D printers like Prusa). However, in Large Factor Additive Manufacturing (LFAM), the printed product often changes during the process due to factors such as temperature shifts, thermal shrinkage, and robotic scale tolerance drift. Each of these factors are being addressed with a multitude of countermeasures, however they are impossible to eliminate completely. As the adverse effects might be considered purely problematic, tracking and adapting to these shifts in real-time could offer new possibilities for improving the process. Conventional pipeline of exporting static robotic programs for fabrication is increasingly less viable and will be converted to fully interactive in the near future. According to Hagedorn (2019): "I think that… in-process monitoring, which means using different sensors to obtain as much information as possible out of your process, but also on using that information to do in-process control… is the Holy Grail for AM because with in-process control you're able to almost immediately react on deficiencies within your process."

In short, the gap between generating and running a close 3D printing loop and achieving a fully interactive printing process was successfully narrowed. Establishing a live link between the robotic and data generation aspects of fabrication enables multiple adjustments during printing (Singline et. al., 2022). These adjustments fall into three main categories: the end effector—specifically, the extruder (e.g., extrusion speed and extrusion flow compensation); the robot's movement (e.g., speed, approach angle); and the geometry of the printed object (e.g., compensation of sagging and thermal shrinkage).

PC to robot data transfer

While working with a robotic LFAM system, we encountered several bottlenecks in streaming data to the robot. Transferring the generated toolpath via USB or Ethernet and loading the program into the robot manually is the baseline mode that lacks convenience, and loading gets significantly slow if the target amount exceeds approx. 8500 points. This method also doesn't easily allow for adaptive behaviour and creating an additional informational loop between the source code, sensors and outcomes. Alternative solutions, that allow for such loop rely on direct printing data streaming from PC to robot.

METHODS AND TOOLS

This section provides an overview of the used hardware and software tools, as well as tested methods for data transfer and plugins for Rhino/Grasshopper.

Robotic system and end effector – extruder and material

Current extruder setup used for the research is a custom-made large-scale polymer pellet extruder mounted on UR10 e-series six-axis robotic arm. The extruder is operating semi-independently from the robot, meaning the start/stop extrusion signal is operated by the robot script. Other functions such as heat control and extrusion speed are controlled manually via the extruder

controller. The extruder has a separate motor for the extrusion and feeding system with independent speed controls that needs to be adjusted and synchronised according to the used material type and shape characteristics.

The material used for printing at the workshop was a virgin pellet PLA (Cattage - キャッテージ) and PrusaResearch recycled PLA (shredded filament), either separately or mixed in various ratios.

Grasshopper plugins

Main program used for turning the geometry into a toolpath for the robot was Grasshopper with various plugins. The source data for the toolpath generation is a curve turned into a polyline and its knots are used as targets for the robot. As Robots and Machina use slightly different logic, there is a need for a dual pipeline to prepare data for the robot where the Robots plugin is used predominantly for preview of planned movements, and Machina for execution, as Machina does not support preview functions.

Robots plugin

Plugin that allows for control of multiple robot brands. The program can detect basic self-collisions and preview the generated toolpath. The toolpath is defined by targets, where each target offers additional coding possibilities such as controlling robot output signals (on/off/pulse) or changing robot's speed.

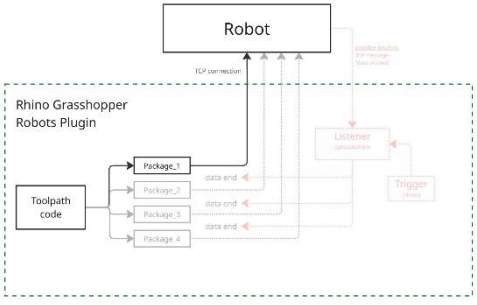


Figure 1
Robots plugin workflow diagram

The plugin can also upload a short program (up to approx. 2500 targets) to the robot and offers a possibility to pause it or play it. Using this utility, all the lines of the code are injected at once, therefore no further adjustments to the code are possible, if being sent in one batch. This limitation can be somewhat overcome by setting up a timer or a command into the code for sending data of the next batch. This method will produce a semi-controllable system that has no clean transition between data batches and therefore is not suitable for 3D printing. Robots plugin has no built-in feedback from the robot, regarding its position or last reached target, although obtaining this information is possible through other means.

Machina plugin and application

RobotExMachina/Machina Bridge is a program that allows PC to robot communication for ABB and UR robots. It uses its own proprietary robot coding language and requires having an additional plugin that communicates with the Machina Bridge. The Machina plugin sends a given batch of data to Machina Bridge which forwards them further to the robot (Figure 1).

As the reading of the remaining code is not continuous and requires an update trigger, which often results in lost or duplicate data on skipped triggers or targets. There seems to be also a timing issue. These laggy position readings also sometimes cause repeated push of toolpath data in the original real time control setup resulting in repeating some of the previous movement. There are several solutions that could be explored in continuation of this workflow such as generating short Boolean pulses rather than a Boolean gate to operate the data injection timing, which should solve random repetition of parts of the toolpath.

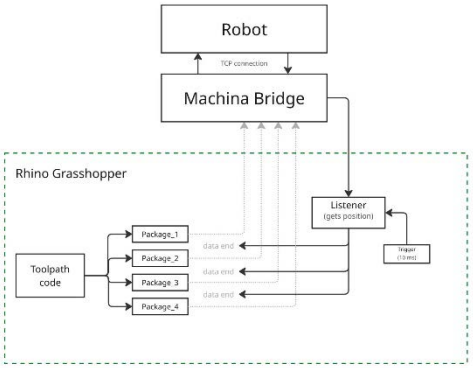


Figure 2
Process diagram of Machina Bridge operation in the adaptive process

Custom data-push function script

Last of the tested options was a custom python script, written utilizing the ur-rtde 1.6.0 library and which combines functionality of Robots and Machina plugins. The custom script uses several program layers to upload printing data, parsed into smaller batches, with the update driven by the robot's position. As the main data-update control is externalised from Grasshopper, this communication script provides continuous code feeding to robot, using Rhino Grasshopper purely for toolpath generation. It also allows basic program controls such as start, pause, and go, while receiving feedback on the robot's position. The script works in addition to Robots add-on, using its nodes and logic for toolpath programming (Figure 2).

The script is still a work-in-progress, but it shows promising results in ease of use and leaving much of the functionality of other parts of the system intact. Further revisions need to be aimed and continuity of the data stream as the robot pauses after finishing each set of moves. This setup has potential to overcome limitations in tools like Machina and Robots by allowing the real-time control externalised from the Grasshopper.

Toolpath adjustment layer

The real-time toolpath adjustment layer is implemented using the Firefly plugin that processes sensor inputs to modify toolpath geometries and printing parameters in real time, enabling both creative exploration and functional adjustments like live extrusion control.

The real-time adjustment is heavily dependent on previously mentioned pc-to-robot communication systems, therefore at the time of writing, this function was tested primarily on the first use case of modifications to the shapes or patterns, however the print quality improvement aspiration can't be omitted.

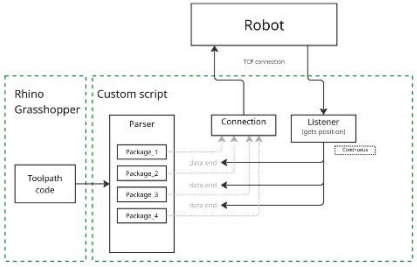


Figure 3
Custom script operation diagram

WORKSHOPS AS TESTING GROUNDS

The system has been initially tested through experiments of increasing complexity, starting with simple linear toolpaths influenced by sensor signals, progressing to cylindrical vases with variable inputs, and culminating in complex grid geometries that demonstrate its generative design capabilities.

To further test the workflow and expose it to varied use cases the workflow was explored and introduced with a few workshops with about 5-10 participants.

These included a few small, studio-based sessions with bachelor and master students and one main iteration held with PhD students at a conference. Each session hosted approximately 5–10 participants, who were introduced to different methods of integrating sensors and real-time interaction into robotic fabrication and workflows. The goal output of the main run was a

large-scale installation composed of 3D-printed panels.

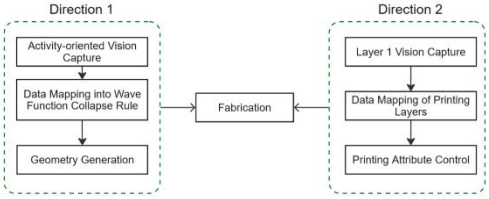


Figure 4
Process diagram of two fabrication strands

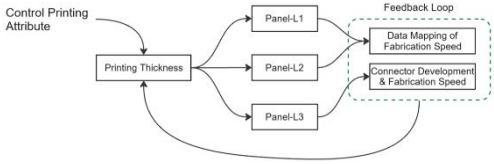


Figure 5
Layer thickness adjustment

Discussions following each workshop focused mainly on the system's potential applications in architectural practice, guided by participant feedback.

Offering two workflows of live data generation vs live-data adjustments (Figure 3), all the participants were given a choice to explore the direction they preferred, sometimes being advised which workflow they should choose, depending on their intentions.

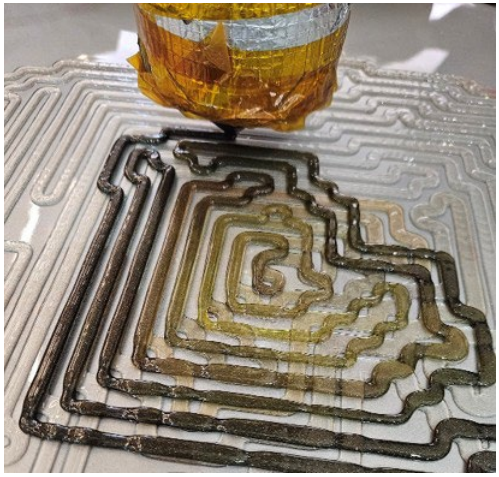


Figure 6
Robotic 3DP of a generated toolpath using WFC

The preferred topic was live data generation, most likely due to simpler implementation. Live data obtained via image-based feeds was used for generating informed fields for grid-like geometries.

Closed data-based workflow

The live data as a source for geometry generation workflow often ended up as a closed loop system and therefore Robots plugin was used to send a generated toolpath script into the robot and no further geometry changes were applied during the printing process.

Data for geometry generation were often collected via video feed or Kinect – depth, colour or human skeleton data), as it offers simple workflow and multiple data readings using Firefly plugin and used to adjust or modify the printing parameters or patterns (Figure 5).

Some of the participants tried to push this solution into semi-interactive approach by consecutively using previous layers as a data source to inform and further adjust following layer based on several factors as e.g. line colour, line thickness or print density. This way, the first layer informed the latter about the calibration, Z height, adjusting the speed of movement by the width of the previously extruded line. This was done by mapping the image over the geometry and linking the pixel values to the printer movement speed (Figure 8).

Realtime based workflow

Other, usually advanced, participants chose to explore the more experimental possibility of real-time robot control. With various geometries to test, the printing process often run into issues with continuous data feeding.

Machina based script was often getting in the loop and repeating data sets (and therefore running over existing geometry), while the custom script had a problem to smoothly transition from one data set to following one.

To simplify and better test the process and strip it from further unknown factors, the sensor-based data were substituted by Grasshopper value sliders and 2D curve deformations, allowing simple interactions with the process e.g. changing layer height or adding curve disturbance on the go.

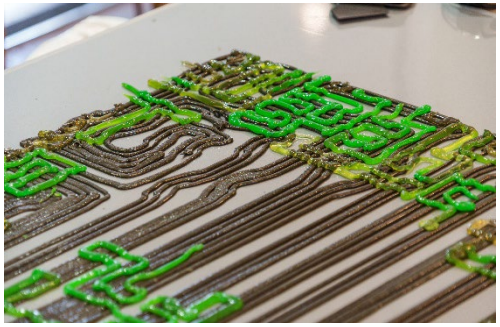


Figure 7
Printed grid panel of informed geometry

RESULTS OF THE WORKSHOPS

The workshops confirmed that there is a wide interest in possibility of real-time data manipulation during the printing process. The input from workshop attendees differed based on their experience with parametric approach, but most of the ideas pointed out, that the real time manipulation could have a use for self-correcting printing, while the semi-interactive workshop workflow is more of an analytic tool that could generate toolpath based on preexisting geometry (Figure 8, Figure 9).

Therefore, an interesting aspect of offering students to try both approaches hands-on is that it helped them to compare both approaches and successfully implemented at least one. The stretch goal of combining both approaches should be explored in further sessions.

Conclusion and outlook

Even though the described approach is very wide, research by design oriented and taking more of an artistic approach, there is a potential for further applications. Real-time robot control while printing and geometry adjustments loops during printing have potential for further applications of a technical character as the principles found in the workshop could be applied to the extrusion adjustment (keeping extrusion width constant by adjusting extrusion speed / robot movement speed) or to some degree self-correcting toolpaths that could compensate for sagging or thermal drift, important e.g. when building space frames.

Future development will possibly branch into refinement of the feedback loop and improvement of live toolpath streaming logic and continuity. Applications could include error correction or modifications to the object while being printed.

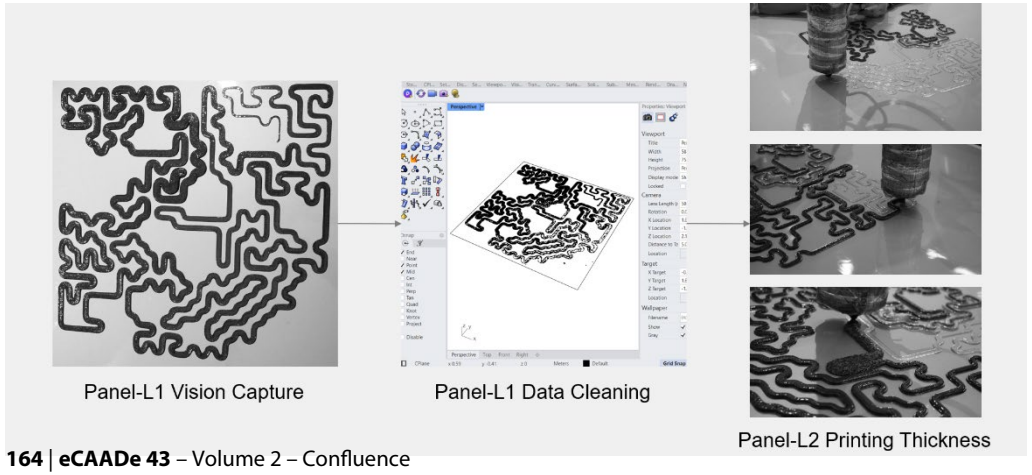
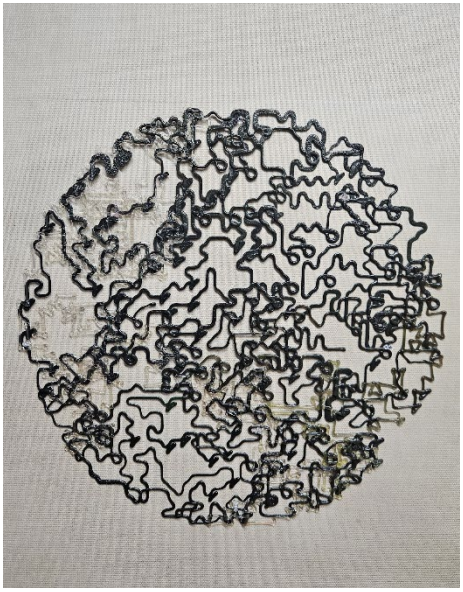


Figure 8
Robotic printing based on thickness of printed path

Figure 9
Printed panel of informed grid geometry



REFERENCES

Breseghello, L. and Naboni, R. (2022) 'Adaptive Toolpath: Enhanced design and process control for robotic 3DCP,' in *Communications in computer and information science*, pp. 301–316. https://doi.org/10.1007/978-981-19-1280-1_19.

Braumann, J., Gollob, E. and Singline, K. (2022) 'Visual Programming for Interactive Robotic Fabrication Processes - Process flow definition in robotic fabrication,' *eCAADe Proceedings*, 2, pp. 427–434. https://doi.org/10.52842/conf.ecaade.2022.2.427.[Accessed March 28, 2025].

Braumann, J., Gollob, E. and Singline, K. (2022b) 'Visual Programming for Interactive Robotic Fabrication Processes - Process flow definition in robotic fabrication,' *eCAADe Proceedings*, 2, pp. 427–434. https://doi.org/10.52842/conf.ecaade.2022.2.427. [Accessed March 31, 2025].

Çapunaman, Ö.B. and Gürsoy, B. (2022) 'See - Sense - Respond: An adaptive robotic fabrication framework for post-processing of glass fiber-reinforced plastic architectural panels,' *eCAADe Proceedings*, 2, pp. 419–426. https://doi.org/10.52842/conf.ecaade.2022.2.419. [Accessed March 28, 2025].

Doerstelmann, M., Knippers, J., Koslowski, V., Menges, A., Prado, M., Schieber, G. and Vasey, L. (2015) 'ICD/ITKE Research Pavilion 2014–15: Fibre placement on a pneumatic body based on a Water Spider Web,' *Architectural Design*, 85(5), pp. 60–65. https://doi.org/10.1002/ad.1955. [Accessed March 20, 2025].

Ergin, E., Stumm, S. and Brell-Cokcan, S. (2024) 'Robotic IOT-Enabled 1D line scanner integration for 3D point cloud data processing,' *Proceedings of the ... ISARC* [Preprint]. https://doi.org/10.22260/isarc2024/0151.

Hagedorn, Y. (2019) *Expert Interview: Aconity3D Managing Director Yves Hagedorn on Helping Companies Innovate with Additive Manufacturing*, https://amfg.ai/2019/08/14/expert-interview-aconity3d-managing-director-yves-hagedorn-on-helping-companies-innovate-with-additive-manufacturing/ (accessed at 31.3.2025)

Im, H.C., AlOthman, S. and Del Castillo, J.L.G. (2018) 'Responsive Spatial Print. Clay 3D printing of spatial lattices using real-time model recalibration,' *ACADIA Quarterly* [Preprint]. https://doi.org/10.52842/conf.acadia.2018.286.

Kazhymurat, T., Shehab, E. and Ali, Md.H. (2022) 'IOT-Based Real-Time 3D Printing Monitoring System,' *2022 International Conference on Smart Information Systems and Technologies (SIST)*, pp. 1–5. https://doi.org/10.1109/sist54437.2022.9945778.

Li, S., Lan, T., Nguyen, H.-X. and Tran, P. (2024) 'Frontiers in construction 3D printing: self-monitoring, multi-robot, drone-assisted processes,' *Progress in Additive Manufacturing* [Preprint]. https://doi.org/10.1007/s40964-024-00794-8.

Luu, Q.K., La, H.M. and Ho, V.A. (2021) 'A 3-Dimensional printing system using an industrial robotic arm,' *2022 IEEE/SICE International Symposium on System Integration (SII)*, 10, pp. 443–448. https://doi.org/10.1109/ieeeconf49454.2021.9382645.

Naboni, R., Breseghello, L. and Sanin, S. (2022) 'Environment-Aware 3D Concrete Printing through Robot-Vision.' *eCAADe Proceedings* pp. 409-418. Available at: https://doi.org/10.52842/conf.ecaade.2022.2.409. [Accessed March 28, 2025].

Slepicka, M., Helou, J. and Borrmann, A. (2023b) 'RTDE robot control integration for Fabrication Information Modeling,' *Proceedings of the ... ISARC* [Preprint]. https://doi.org/10.22260/isarc2023/0016.