Demonstrating Dynamic Toolchains for Machine Control

Hannah Twigg-Smith and Nadya Peek
University of Washington
Seattle, WA, USA
{htwigg,nadya}@uw.edu

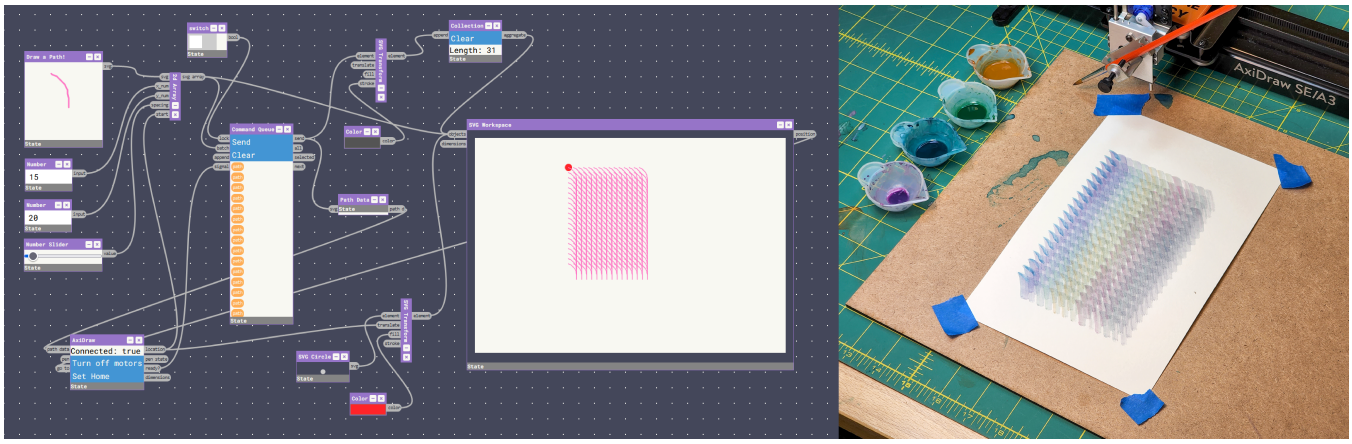

Figure 1: Dynamic toolchains can be used to interactively control machines for niche applications. Here, a dynamic toolchain for watercolor painting is used to step through a parametric toolpath created from user input.

ABSTRACT

Humans are increasingly able to work side-by-side with desktop-scale digital fabrication machines. However, much of the software for controlling these machines does not support live, interactive exploration of their capabilities. We present ***Dynamic Toolchains***, an extensible development framework for building parametric machine control interfaces from reusable modules. Toolchains are built and run in a live environment, removing the repetitive import and export bottleneck between software programs. This enables humans to easily explore how they can use machine precision to manipulate physical materials and achieve unique aesthetic outcomes. In this demonstration, we build a toolchain for computer-controlled watercolor painting and show how it facilitates rapid iteration on brush stroke patterns.

CCS CONCEPTS

• **Human-centered computing** → **Interactive systems and tools**; • **Software and its engineering** → **Integrated and visual development environments**.

KEYWORDS

toolpath design, plotting, watercolor, digital fabrication

UIST '22 Adjunct, October 29-November 2, 2022, Bend, OR, USA

ACM ISBN 978-1-4503-9321-8/22/10.
https://doi.org/10.1145/3526114.3558662

ACM Reference Format:
Hannah Twigg-Smith and Nadya Peek. 2022. Demonstrating Dynamic Toolchains for Machine Control. In *The Adjunct Publication of the 35th Annual ACM Symposium on User Interface Software and Technology (UIST '22 Adjunct), October 29-November 2, 2022, Bend, OR, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3526114.3558662

1 INTRODUCTION

The CAD/CAM/CNC software pipeline traditionally used for digital fabrication workflows separates design and machine execution into distinct programs [9]. This separation enforces a "static design practice" [8], where a bottleneck imposed by static interchange files prevents machine users from easily iterating on machine behavior. Every program includes decision points that influence how a machine will execute a fabrication task, which ultimately determines the aesthetic qualities of the eventual outcome [11]. The lack of feedback between CAD, CAM, and CNC software forces machine users to predict how design decisions made early in their workflows will translate to machine behavior, which can make aesthetic outcomes hard to anticipate and difficult to explore [2, 4, 5].

For example, Figures 2 and 3 show how the path a machine will follow while painting with watercolor does not indicate the nuanced way paints will mix, layer, and pool. Under a CAD/CAM/CNC workflow, a machine user must engage in a lengthy trial-and-error process of tweaking designs, importing and exporting files, and repeatedly setting up jobs to explore how different toolpaths will translate to different aesthetic outcomes. Alternatively, developing custom software for this workflow would require extensive implementation time because there are few tools to support fabrication software development outside of the CAD/CAM/CNC paradigm.

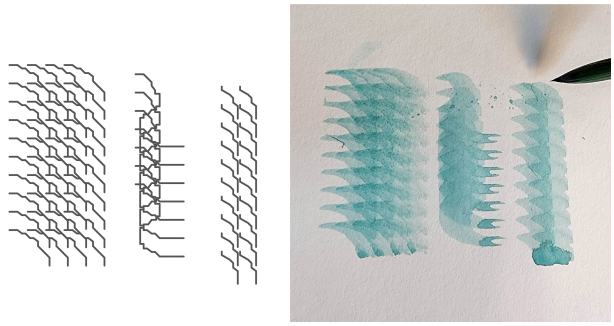

Figure 2: The toolpath for a series of brushstrokes looks very different from the eventual painting. Watercolor brushstrokes include pools of paint, overlapping layers of color, and bristle marks patterns. All of these are expressive dimensions that should be available to a machine user.

Seeking to re-couple human design intent with machine behavior, researchers have proposed new fabrication paradigms such as *interactive*, *lucid*, or *bidirectional* fabrication, where fabrication systems mediate human input and machine behavior [3, 7, 10]. These contributions are often accompanied by promising proof-of-concept systems tailored to individual application domains.

We want to support further development of novel fabrication systems that enable unique collaborations between humans and machines. To this end, we contribute ***Dynamic Toolchains***, a extensible development framework for building alternative fabrication interfaces from live, interconnected, and reusable software tools. Our framework builds on prior work that has explored interactive machine control from programming environments [1, 6]. Dynamic toolchains additionally provides support for custom graphical interface components, event-driven feedback between tool modules, and representations of real-world state. In the following section, we provide a high-level overview of our framework's implementation. Finally, we introduce our demonstration by describing a hypothetical scenario for building and using a toolchain for watercolor painting.

2 IMPLEMENTATION

When designing the dynamic toolchains framework, we drew inspiration from parametric design tools (e.g. Grasshopper, Sverchok), component frameworks (e.g. React, Lit), and live programming environments. Our system has two main parts: a *live environment* for constructing and executing dynamic toolchains, and a *component framework* for developing the software tool modules which comprise toolchains. Tool modules resemble miniature, full-stack web applications with clearly defined APIs. Tools consist of three components:

- **JSON configuration file.** A short configuration file specifies the tool's basic information, including the tool name and description, input and output data ports, and internal state variables.
- **Python backend.** A tool's Python backend can include methods that handle human input, define custom behavior, manage internal state, and communicate with connected tools.

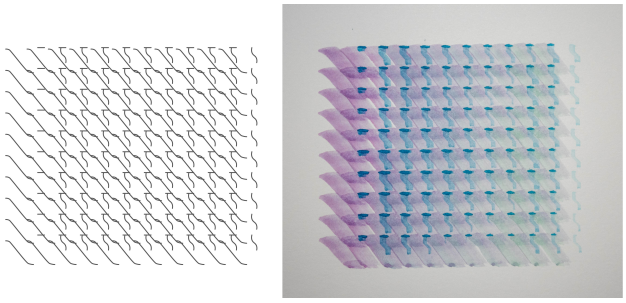

Figure 3: Left: a toolpath built interactively via a dynamic toolchain. Right: The resulting painting.

- **Web Component.** Optionally, tools can include a Web Component to serve as a graphical interface in the browser frontend. Web Components are a modern standard for encapsulating HTML, CSS, and JavaScript into a reusable custom element.

Our framework provides Python and JavaScript parent classes for tool modules. These have very few dependencies outside of standard language features and include lifecycle methods that tool modules can override for custom behavior Ultimately, we place very few restrictions on module implementation and functionality. We also provide a browser interface for constructing and manipulating toolchains. Our environment dynamically creates and maintains WebSocket connections between the front- and back-ends of each tool module. Additionally, our system includes an event planner that handles communication between tools. Because tools maintain internal state, we can also "freeze" active toolchains, save them to a JSON file, reload them at a later time, and continue from the frozen state. These features enable tool developers to focus on their tool's implementation without worrying about how information will be passed through the toolchain.

3 DEMONSTRATION SCENARIO: PAINTING WITH A PLOTTER

Robin is a generative artist who often uses an AxiDraw plotter to create plotted artworks. Plotters are simple two-axis CNC machines that move a tool, often a pen or knife, around a two-dimensional space. In addition to traditional pen plotting, Robin has begun to use a plotter for watercolor painting. They are particularly interested in exploring how machine precision can be combined with fluid, dynamic watercolor paints. Robin builds a dynamic toolchain to explore brush stroke patterns for watercolor plotting, the interface for which can be seen in Figure 1. Their toolchain consists of a number of tools, which can be broken into four categories: AxiDraw communication, visualization, parametric toolpath generation, and event planning.

AxiDraw Communication. The AxiDraw communication tool wraps the functionality of the AxiDraw Python library The tool module connects to the machine over USB and sends it movement commands, which can either be absolute move commands or a series of relative move commands. The tool module outputs the current position of the machine head, the size of the drawing envelope, and a boolean "ready" state (whether the machine is ready to accept new

commands). Using the AxiDraw communication tool as a starting point, Robin is able to build the rest of the toolchain.

Visualization. The SVG workspace tool accepts any number of SVG elements and renders them to a canvas. It outputs the location of user clicks on the workspace. The SVG transform tool can modify attributes of SVG elements, such as the translate or color attributes. By connecting the position output of the AxiDraw communication tool to an SVG transform of a SVG circle element, Robin can visualize the current position of the AxiDraw. If they then connect the position output of the SVG visualization tool to the AxiDraw's absolute move input, they can now move the machine by clicking on the SVG workspace.

Parametric toolpath generation. The path input tool accepts human drawing input and outputs an SVG path containing their drawing. The 2D array tool creates a two-dimensional array of SVG path elements and outputs them in a list. Using generic number and slider inputs along with these tools enables parametric control of the path pattern. Because the output paths are SVG elements, they can be rendered to the SVG canvas in order to preview them. Additionally, Robin uses the AxiDraw's current location to translate the path elements. Now, the machine will draw them starting from its current position.

Event planning. Finally, the command queue tool module enables Robin to manage the path commands sent to the AxiDraw. Using the output from the toolpath generation modules, Robin can create a queue of commands to be sent to the AxiDraw. They can either step through these commands one at a time by clicking on the buttons in the graphical interface, or set the toolchain to run autonomously by connecting the AxiDraw's "ready" port to the command queue's "signal" port. At any point, they can pause execution of the command set. Robin uses this feature to progressively drop different colors into the paint brush, enabling them to create colorful paintings such as the one seen in Figures 1 and 3.

3.1 Demonstration Setup

Our demonstration setup includes a computer for interacting with our software and an AxiDraw plotter configured with watercolor paints and paintbrushes. Visitors to the demo can create toolpaths by drawing with a stylus or uploading their own SVG input files. They can preview the resulting toolpaths, send them to the machine for testing, and create machine-painted artworks to take home.

4 CONCLUSION

We demonstrate *dynamic toolchains*, an extensible framework for building live digital fabrication machine control interfaces from modular interconnected components. Toolchains are authored in a browser-based environment by connecting modules with various functionalities, such as designing paths, creating machine commands, or sending commands to a machine. In this demonstration, users can modify the base path of a plotting toolpath to explore the different aesthetic outcomes of watercolor brushstrokes.

ACKNOWLEDGMENTS

This research was funded in part by the NSF IIS Human-Centered Computing program (#2007045).

REFERENCES

[1] Frikk Fossdal, Rogardt Heldal, and Nadya Peek. 2021. Interactive Digital Fabrication Machine Control Directly Within a CAD Environment. In *Symposium on Computational Fabrication (SCF '21)*. Association for Computing Machinery, New York, NY, USA, 1–15. https://doi.org/10.1145/3485114.3485120

[2] Nathaniel Hudson, Celena Alcock, and Parmit K. Chilana. 2016. Understanding Newcomers to 3D Printing: Motivations, Workflows, and Barriers of Casual Makers. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. Association for Computing Machinery, San Jose, California, USA, 384–396. https://doi.org/10.1145/2858036.2858266

[3] Jeeeun Kim. 2017. Shall We Fabricate? Collaborative, Bidirectional, Incremental Fabrication. In *Adjunct Publication of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. Association for Computing Machinery, New York, NY, USA, 83–86. https://doi.org/10.1145/3131785.3131844

[4] Jeeeun Kim, Haruki Takahashi, Homei Miyashita, Michelle Annett, and Tom Yeh. 2017. Machines as Co-Designers: A Fiction on the Future of Human-Fabrication Machine Interaction. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '17)*. Association for Computing Machinery, New York, NY, USA, 790–805. https://doi.org/10.1145/3027063.3052763

[5] Behnaz Norouzi, Marianne Kinnula, and Netta Iivari. 2021. Making Sense of 3D Modelling and 3D Printing Activities of Young People: A Nexus Analytic Inquiry. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. Association for Computing Machinery, New York, NY, USA, 1–16. https://doi.org/10.1145/3411764.3445139

[6] Blair Subbaraman and Nadya Peek. 2022. P5.Fab: Direct Control of Digital Fabrication Machines from a Creative Coding Environment. In *Designing Interactive Systems Conference (DIS '22)*. Association for Computing Machinery, New York, NY, USA, 1148–1161. https://doi.org/10.1145/3532106.3533496

[7] Rundong Tian, Vedant Saran, Mareike Kritzler, Florian Michahelles, and Eric Paulos. 2019. Turn-by-Wire: Computationally Mediated Physical Fabrication. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology (UIST '19)*. Association for Computing Machinery, New Orleans, LA, USA, 713–725. https://doi.org/10.1145/3332165.3347918

[8] Cesar Torres and Eric Paulos. 2015. MetaMorphe: Designing Expressive 3D Models for Digital Fabrication. In *Proceedings of the 2015 ACM SIGCHI Conference on Creativity and Cognition (C&C '15)*. Association for Computing Machinery, New York, NY, USA, 73–82. https://doi.org/10.1145/2757226.2757235

[9] Hannah Twigg-Smith, Jasper Tran O'Leary, and Nadya Peek. 2021. Tools, Tricks, and Hacks: Exploring Novel Digital Fabrication Workflows on #PlotterTwitter. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. Association for Computing Machinery, New York, NY, USA, 1–15. https://doi.org/10.1145/3411764.3445653

[10] Karl D.D. Willis, Cheng Xu, Kuan-Ju Wu, Golan Levin, and Mark D. Gross. 2010. Interactive Fabrication: New Interfaces for Digital Fabrication. In *Proceedings of the Fifth International Conference on Tangible, Embedded, and Embodied Interaction (TEI '11)*. Association for Computing Machinery, New York, NY, USA, 69–72. https://doi.org/10.1145/1935701.1935716

[11] Malgorzata A Zboinska and Delia Dumitrescu. 2021. On the Aesthetic Significance of Imprecision in Computational Design: Exploring Expressive Features of Imprecision in Four Digital Fabrication Approaches. *International Journal of Architectural Computing* 19, 3 (Sept. 2021), 250–272. https://doi.org/10.1177/1478077120976493