Radical interfaces

By Abha Patil

A thesis exhibition presented to OCAD University in partial fulfillment of the requirements for the degree of Master of Design in Digital Futures

OCADU Waterfront Campus, 130 Queens Quay East, March 27 – April 1st.

Toronto, Ontario, Canada, 2025

Creative Commons Copyright Notice

Abstract

'Radical interfaces' wanders beyond the conventional boundaries of interaction design by rethinking the utilitarian frameworks that dominate the design industry. It aims to discover the playful and open-ended spirit of making by combining unconventional web applications, DIY (Do It Yourself) electronics, and atypical graphic design methods that prioritize experimentation and individuality. It approaches interfaces as *playgrounds* rather than *tools*.

Through five core prototypes - a font that you can physically pump air into, an Excel styled spreadsheet as a drawing tool, a 3D application that uses words as coordinates, typing with a rotary phone, and an animation pipeline for a dot-matrix display, I try to articulate 'radical', by addressing issues of openness, materiality, criticality, and reflection.

This research combines mutually informed research through design and observational analysis by developing prototypes, documenting their evolution and reflecting on their intended and unintended uses, allowing for analysis focused on their reflective and critical potentials.

By building open source and malleable tools that can be built upon, re-purposed or even reinvented, 'Radical interfaces' aims to refresh our relationship with technology from being passive consumers to active makers.

Keywords: *graphic design, interfaces, physical computation, human computer interactions, hardware hacking, alternative controllers, open source, DIY citizenship*

Acknowledgements

Kate Hartman - You have truly been the anchor of this ship. Your work, your teachings, your enthusiasm and your support has been key in shaping this thesis.

Nick Puckett - Thank you for the endless tech support and most importantly, for steering me back on course every time I drifted away.

Aranya and Juan - This journey would have been extremely boring without having you guys around! Can't think of better people to have elaborate discussions on how *empanadas* and *samosas* are basically the same thing in the middle of a chaotic semester.

Mandar - For accompanying me to every thrift store in the city as we hunted for the *radical*. Here's hoping that we never run out of strange finds, and a space to store them.

Aai and Baba - I miss you every single day. Thank you for the endless support, even when you had no clue what I am up to. Your love and encouragement mean the world to me.

Table of Contents

List of Figures

List of Tables

Introduction

I have always been a *maker,* and most of my creative outlets as a child would manifest in the most absurd forms possible. I would make odd-looking figurines with little trinkets I'd find on the road. They were awesome! They existed simply as a result of curiosity, play, and the joy of making something from nothing.

When I was 12, someone said, ***'Hey, you are such an artist, you should become a Graphic Designer!'*** I held onto that advice, and eventually, it led me to pursue Graphic Design. Design, as I learnt, was built around usability and efficiency. There were industry standards and workflows that dictated how things were made. The interfaces that I primarily worked with were designed to draw straight lines, snap to grids and measure every pixel. Over time, these workflows replaced the odd and raw bits of my expression with extreme precision and a sense of order.

While 'structure' serves its purpose, I still find myself drawn to the raw, exploratory and sometimes whimsical nature of *making* which often take a backseat in traditional workflows. This led me to question – *What if design interfaces allowed for more open-endedness and spontaneity?*

This research explores '*making'* as an interactive and iterative practice. By experimenting with unconventional and sometimes impractical interactions, it approaches interfaces as *playgrounds* rather than *tools*.

To begin with, I started making simple tools for graphic design work, rethinking interactions through non-traditional interfaces and alternative controllers. Instead of choosing traditional workflows, I chose to unlearn and rethink the processes from the ground up. Which led to the creation of five open-source, malleable interfaces and a *Radical interfaces playbook* - a set of guiding principles that advocate for open-ended, playful, and experimental approaches to interface design.

Each of these prototypes explores a different way of engaging with design tools. *I used Excel to make animations* reimagines a spreadsheet-style application as a medium for illustration and animation. *125* explores 3D modelling by replacing traditional spatial manipulation with a language-based system, where objects are formed by stringing together combinations of words. *A font that inflates* brings physicality into typography, using an air pump to alter letterforms, making text feel tangible. *Animating flipdots* transforms animations into a tactile experience by leveraging an electromechanical display, creating an interplay between motion and pressure. And lastly, *typing with a rotary phone* repurposes nostalgic technology for slow, intentional communication.

Together, these prototypes form a family of experimental tools - each radically different in form and function yet united by a shared philosophy of openness, play, and reimagination. The Radical interfaces family photo (see Fig. 1) captures this collective spirit of the five interfaces as a broader exploration into alternative modes of design interaction.

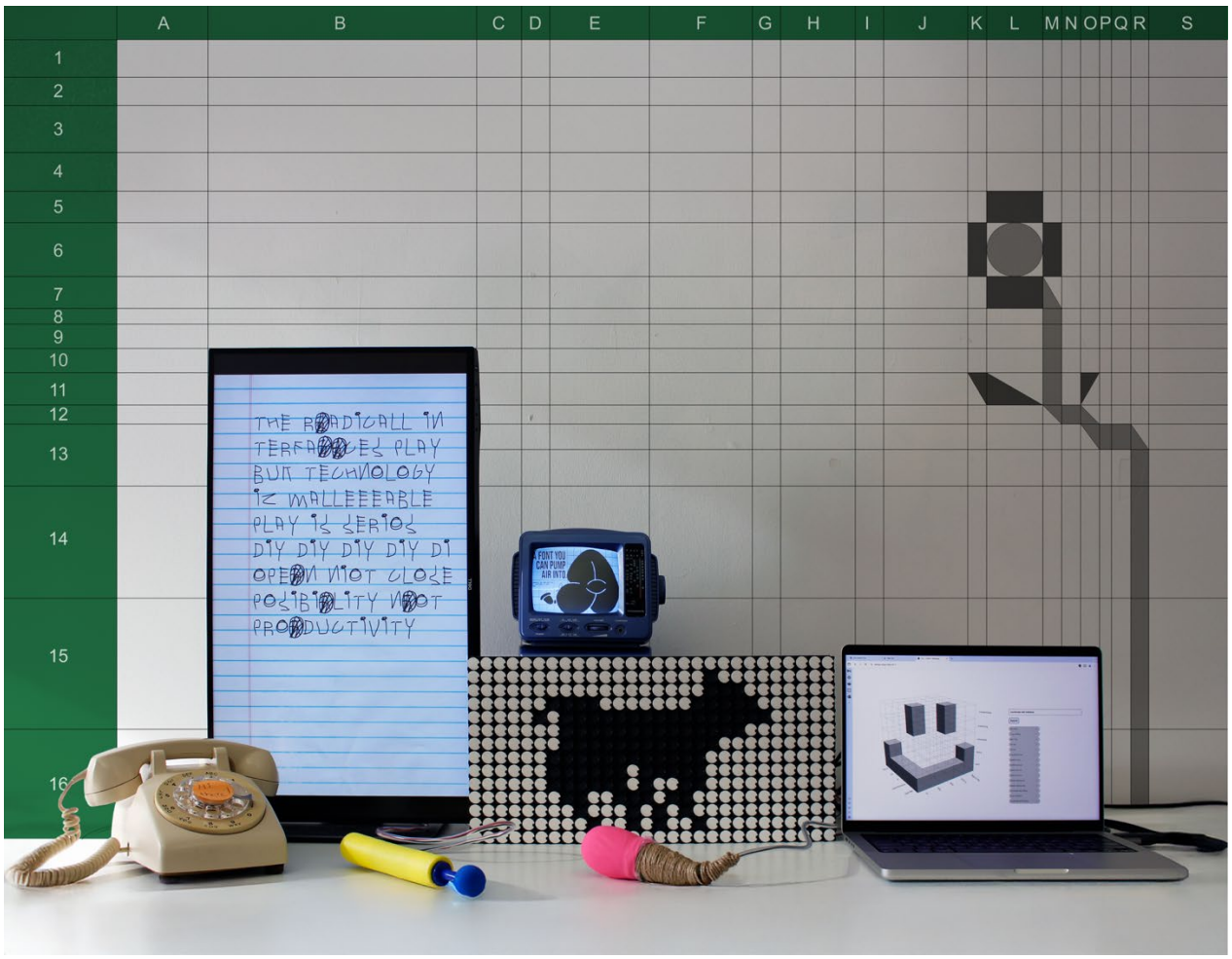


Figure 1: Radical interfaces family photo

The goal with Radical interfaces was never to produce finished products, but rather to develop interfaces that invite modification, experimentation and reinvention. More than tools, they are starting points meant to inspire, excite and welcome.

By rethinking interaction, these prototypes offer new perspectives on what design tools *can* be.

Research Questions

1. How can I design interfaces that encourage curiosity, play, and exploration rather than just usability?

2. What does it mean to *make* in graphic design when computation and tangible interactions are involved?

Literature Review

This section of the document will dissect different ideas within the research, situating them within broader historical, technological and theoretical contexts. And as these ideas converge, they will help us shape our understanding of Radical interfaces.

A recurring word in this research is *malleable*, which by definition means something that is flexible rather than being rigid or predefined. I use this word in relation to technology, emphasizing that technology itself is inherently malleable, yet its trajectory depends on how we choose to engage with it. If left solely to corporate interests or efficiency-driven models, it risks becoming stagnant, restrictive, or even regressive. A prime example of this is the evolution of the web, which once *was* an open, experimental and a deeply personal space as described in JR Carpenter's essay - A Handmade Web (2015).

Another important theory is *Critical Making,* which frames *making* as a form of questioning rather than just production. Here, Alternative Controllers become a form of Critical Making through hacked hardware and unconventional interactions that question our understanding of interfaces.

Lastly, all these ideas make their way into graphic design – my primary mode of expression and a field that I am the most familiar with. The final section examines what it means to *make* in graphic design when computation and physical interactions are involved, encouraging a more hands-on and experimental approach.

a. Evolution of the Web

The World Wide Web (or simply known as the Web) was invented by Tim Berners-Lee, a computer scientist working at CERN in 1989. He was motivated by the problem of storing, updating and retrieving data in a large and constantly changing organization.[1]

Initially, the web functioned as an academic tool, used primarily by research institutions and universities to share information. However, as access expanded in the mid-1990s, it evolved into an experimental space. No longer restricted to academia, the web became a playground for everyday users and hobbyists, who played an important role in its transformation.

1. The European Organization for Nuclear Research (CERN), *A Short History of the Web*, accessed February 14, 2025, https://home.cern/science/computing/birth-web/short-history-web

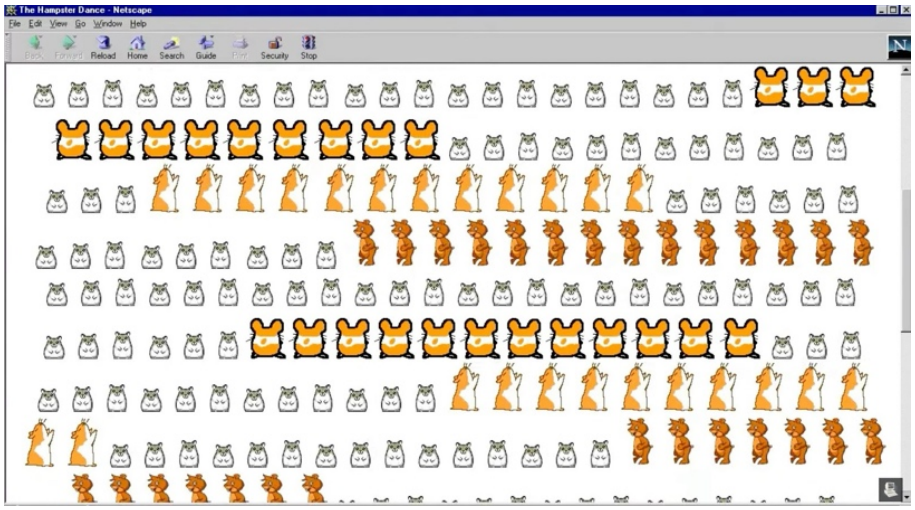


Figure 2: Hampster Dance Website

In just a decade, web pages went from simple text-based layouts to loud, colourful sites with flashy animations. Figure 2 shows a picture of The Hampster Dance Website that went viral through forwarded emails. It featured a row of dancing hamsters, topped off with a never-ending "*dodadidadodadodo"* audio in the background.[2] If you recognize this, you likely witnessed the web transition from its academic origins to an experimental playground and eventually became the corporate-driven space we know today.

During its experimental phase, the web was a malleable, open space, where early users shaped its culture, aesthetics and interactions. During its formative years, webpages were *hand-coded* (manually written using programming languages, without the help of website builders or content management systems) by individuals. This resulted in pages that were always a work in progress and DIY, which challenged the notions of reading, writing, design, ownership, privacy, security, or identity.[3] These aesthetics were largely shaped by slow internet speeds, which required websites to be designed with efficiency in mind. Heavy images and complex graphics were avoided in favour of tiled backgrounds, pixelated GIFS, and other lightweight elements that minimized data load. This gave rise to a distinct visual language, one defined by low-resolution graphics and quirky animations.

Olia Lialina, in her essay, A Vernacular Web (2005), talks about these iconic, and now almost nostalgic elements like the Under Construction Sign, Large home buttons, Outer Space Glittery Backgrounds, Blinking New Buttons, Guestbooks that visitors could sign in and the MIDI files that later became a subjects

2. Webflow, *A Look Back at '90s Website Design (the Good, the Bad, and the Ugly)*, accessed February 17, 2025, https://webflow.com/blog/90s-website-design.

3. J. R. Carpenter, "A Handmade Web," *Lucky Soap*, March 26, 2015.

of mockery.[4] As she observes, these early amateur users shared a genuine excitement about the web as a medium - a sentiment that has largely faded now.

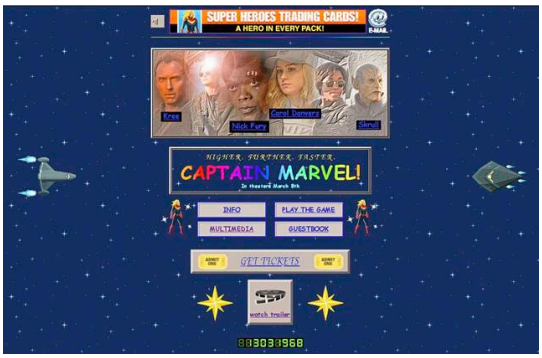


Figure 3: Captain Marvel Website

Figure 3: Space Jam Website

A defining feature of this era was Macromedia Flash, introduced in 1996[5], which brought dynamic graphics and interactivity far beyond the limitations of basic HTML. Flash allowed creators to experiment with movement, sound, and playful user interactions, leading to the rise of web-based games, experimental navigation, and highly stylized personal websites.[6]

While today's web is more functional, Lialina raises the question of what has been lost is the sense of individuality, the aesthetic quirks, and personal engagement. The nostalgia surrounding these early digital experiences speaks to a lost culture of customization, unpredictability, and personal expression that Radical interfaces aim to bring back. Some prototypes revive obsolete technologies and forgotten interaction styles in unexpected ways to speak to this very culture. They emphasize slow, intentional and hands-on engagement with technology.

Instead of using tools as they are designed, what happens when we break them, modify them, or create our own? By shifting from being mere users to *makers,* there is an opportunity here to shape technology rather than being shaped by it.[7]

4. Olia Lialina, "A Vernacular Web," https://art.teleportacia.org/observation/vernacular/.

5. Adobe Flash," *Wikipedia, The Free Encyclopedia*, last modified [insert date], https://en.wikipedia.org/wiki/Adobe_Flash.

6. Cybercultural. "1996: Flash, CSS & Web Design." *Cybercultural*, https://cybercultural.com/p/1996-flash-css-web-design/

7. Ruben Pater, Caps Lock: How Capitalism Took Hold of Graphic Design, and How to Escape from It (Amsterdam: Valiz, 2021).

b. Critical Making:

Critical Making originates from Critical Design - a term introduced by Dunne & Raby in the 1990s. Critical Design challenges the idea that design should only solve problems, it instead uses design as a tool for critique, speculation, and questioning societal and technological norms.[8] While Radical interfaces share this mindset by questioning the relationship between technology and society, it does so primarily through Critical Making.

Matt Ratto introduced the concept of Critical Making in 2011 that combines hands on making with critical reflection. Unlike traditional design, where objects are created for others to experience, critical making focuses on the process itself as a site of exploration and understanding.[9] Through tinkering, experimentation, and iteration, makers develop new relationships with technology - ones that are shaped by curiosity and intent.

The emphasis on *making* extends beyond just the outcome; it highlights the interactions and discoveries that happen along the way.[10] This was observed while developing '*I used Excel to make animations,'* where unexpected outcomes led to shifts in direction that were not initially anticipated. It showed that the process shaped the outcome just as much as the intent behind it.

This emphasis on active participation connects to broader models of collaborative creation. Yochai Benkler, in *Freedom in the Commons* (2003), describes "peer production" as an alternative model where individuals contribute to shared projects in decentralized ways.[11] This model is already visible in open-source communities, maker movements, and experimental design practices, where individuals modify, remix, and repurpose tools rather than using them as intended. Unlike top-down systems, this approach thrives on collaboration and self-motivation, empowering individuals to contribute in ways that align with their skills and interests.

Similarly, Critical Making is fundamentally collective and open-ended. It is not just about individual experimentation but also about sharing knowledge and experiences that were gathered through *making*. This approach relies on open design technologies and processes that encourage the distribution and sharing of technical work and its results. Blueprints and instructions become the means of extending these experiences to others.[12]

---

8. Anthony Dunne and Fiona Raby, *Speculative Everything: Design, Fiction, and Social Dreaming* (Cambridge: MIT Press, 2013)

9. Matt Ratto, "Critical Making," in *Open Design Now: Why Design Cannot Remain Exclusive*, ed. Bas van Abel, Lucas Evers, Roel Klaassen, and Peter Troxler (Amsterdam: BIS Publishers, 2011), 202–213.

10. Manning, Erin, and Brian Massumi. *Thought in the act: Passages in the ecology of experience*. Minneapolis: University of Minnesota Press, 2014.

11. Yochai Benkler, "Freedom in the Commons: Towards a Political Economy of Information," *Duke Law Journal* 52, no. 6 (2003): 1245–1276.

12. Matt Ratto, "Critical Making," in *Open Design Now: Why Design Cannot Remain Exclusive*, ed. Bas van Abel, Lucas Evers, Roel Klaassen, and Peter Troxler (Amsterdam: BIS Publishers, 2011), 202–213.

c. Alternative Controllers

When we think about how we interact with screen-based technology, traditional interfaces like mouse, keyboard and touchscreen dominate. While these interfaces have become second nature, they also limit the way we engage with digital systems. If I give you 20 seconds to imagine a digital interaction beyond a click, hover and a scroll, it will get surprisingly difficult.

In gaming, alternative controllers refer to any input device beyond standard gamepads, keyboards, or mice. This term gained popularity through the Game Developers Conference (GDC), where designers and artists experimented with unconventional input methods.[13] They ranged from DIY custom controllers to modified traditional devices, repurposed in unexpected ways.

My first introduction to an alternative controller was in the *Advanced Wearables* class taught by Kate Hartman. The assignment was to create a one-of-a-kind, never-seen-before Alternative Controller. I came up with *Booger Bonanza* - a mask with a large, felted nose where the player navigated left and right by digging into its nostrils. It consisted of an Adafruit Flora microcontroller, which was connected to switches in the nostrils activated by the wearer's fingers.

Figure 4: Booger Bonanza

13. "[alt.ctrl] Indie Games with Weird Controllers from GDC '22," YouTube video, 17:02, posted by MechBird, March 16, 2022, https://www.youtube.com/watch?v=IDVHkcUB9jI&t=54s

Various unconventional input methods have been illustrated by Mika Satomi and Hannah Perner-Wilson in their collective Kobakant. One of which is Joyslippers - a pair of slippers used for drawing.[14] Each slipper has two pressure sensors in the sole that detect weight shifts between the toe and heel. These sensors are connected to an Arduino[15] via spiral telephone cords, which transmit pressure data to a Processing[16] sketch. The sketch then reads and processes these values, translating them into drawing directions, allowing users to create images using their feet.

Figure 4: Joyslippers

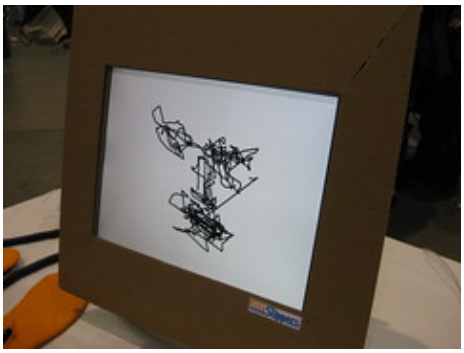


Figure 5: A sketch made by Joyslippers

This demonstrates that interfaces could be *anything*. They could be embodied, playful and tactile. They could engage our full range of senses, inviting movement and intuition. Radical interfaces explore these alternative ways of engaging with technology, encouraging interactions that are surprising and absurd. One of the most exciting ways this is explored is through hardware hacking - a practice of repurposing old, obsolete devices. Hacking challenges the idea that tools and systems must be used as intended. Instead, it sees technology as malleable - something to be taken apart, reconfigured, and adapted.

An example of this comes from Reed Ghazala, who pioneered the practice of 'circuit bending'. This technique takes found objects such as battery-powered children's toys and inexpensive synthesizers and modifies them into DIY musical instruments and homemade audio generators.[17] The most notable example of Ghazala's work is the Incantor series of devices - Speak & Spell, Speak & Read and Speak & Math devices that reconfigure the synthesized human voice circuitry within a toy to make alien-like sounds.

Figure 5: Reed Ghazala's Incantor

14. Kobakant, *JoySlippers*, accessed [Mar 7, 2025], https://www.kobakant.at/DIY/?p=567

15. Arduino, *Arduino Software (IDE)*, Version 2.3.0 (Arduino, 2024), https://www.arduino.cc/.

16. Casey Reas and Ben Fry, *Processing*, Version 4.0 (Processing Foundation, 2024), https://processing.org/.

17. Garnet Hertz and Jussi Parikka, "Zombie Media: Circuit Bending Media Archaeology into an Art Method," *Leonardo* 45, no. 5 (October 2012): 424–30, https://doi.org/10.1162/leon_a_00438.

Beyond sound, hardware hacking has extended into visual and interactive art. The rise of low-cost microcontroller boards like Arduino and the availability of DIY videos, blogs, and tutorials have made hardware hacking more accessible, making it increasingly popular as an artistic technique.

Another example of re-purposing is Makey Makey, by Jay Silver - a STEM kit designed to encourage users to experiment and learn through *making*. It allows users to connect everyday objects - bananas, buckets of water, aluminum foil to computers, turning them into DIY input devices. Jay emphasizes that Makey Makey involves the most important part of interaction design - *assigning meaning*.[18] It requires re-purposing something from the physical world and the online world, and hooking them together. They result in thousands of permutations and combinations of objects and tools, each one taking on a new meaning and function.

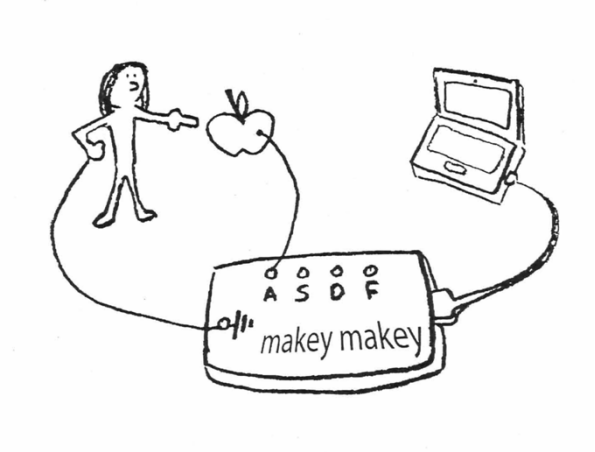


Figure 6: Makey Makey by Jay Silver

By challenging what qualifies as a controller, alternative controllers help reframe technology as participatory and personal. This is not just about repurposing old gadgets but about shifting our perception of *what an interface can be.* If an apple can be a space bar and a toy can be a synthesizer, what else can be reimagined?

18. Jay Silver, "Makey Makey," in *Critical Making: Projects*, ed. Garnet Hertz (2012), 5–6.

d. Graphic design and computation

Graphic design is a form of visual communication that helps us engage with information, whether through branding, editorial design, digital interfaces, or printed media in an accessible manner. Graphic designers use typography, images, colour, and composition to construct meaning that informs most of graphic design. Traditionally, industry-standard software like Photoshop [19]and Illustrator [20] have been the primary toolset for graphic designers. However, within the field, there are specialized areas that require more specific software. For example, type design is a discipline of its own, using software like Glyphs.[21] 3D design involves tools such as Maya[22], Blender [23], Cinema 4D [24], while motion graphics rely on programs like After Effects [25] and Cavalry.[26] Sometimes, these specializations overlap; for instance, type design can be combined with 3D modelling and motion graphics to create 3-dimensional animated typography, or motion graphics sometimes gets clubbed with user interfaces to provide visual cues for navigation. Graphic design includes a wide range of niches, allowing designers to specialize in various sub-disciplines, each being vast in its own right.

I never specialized in these sub-disciplines but have explored them briefly from time to time, only to realize how little I knew about each. While I was curious to learn more, understanding how these different systems worked felt intimidating, given the complexity of the software and how distinct they were from one another.

When I learnt the basics of programming, I naturally tied it back to my roots - graphic design. I started building my own small systems to simplify the sub-disciplines I had never fully explored. *A font that inflates* was an attempt at making a variable typeface, while 125 is an easy-to-use 3D tool that maps pixels in a 3D coordinate system. These rudimentary systems were shaped by my own understanding of how I wanted things to work, rather than how they traditionally did. Some experiments were long and time-consuming, while others turned out to be outright hilarious. Some generated interesting ideas of what they could be used for instead. The possibilities of design that programming offered, expanded my perspective in ways traditional software hadn't.

While graphic design has long been defined by tools made *for them*, rather than *by them,* learning programming offers a way to change that. Following are some tools/ platforms/ systems/ interfaces made by designers that Radical Interfaces draws inspiration from.

---

19. Adobe Systems, *Adobe Photoshop*, Version 2024 (San Jose, CA: Adobe, 2024).

20. Adobe Systems, *Adobe Illustrator*, Version 2024 (San Jose, CA: Adobe, 2024).

21. Georg Seifert and Rainer Erich Scheichelbauer, *Glyphs*, Version 3 (Mekkablue & Georg Seifert, 2024), https://glyphsapp.com/.

22. Autodesk, *Maya*, Version 2024 (San Rafael, CA: Autodesk, 2024), https://www.autodesk.com/products/maya/.

23. Blender Foundation, *Blender*, Version 4.0 (Amsterdam: Blender Foundation, 2024), https://www.blender.org/.

24. Maxon Computer, *Cinema 4D*, Version 2024 (Friedrichsdorf, Germany: Maxon Computer, 2024), https://www.maxon.net/en/cinema-4d.

25. Adobe Systems, *Adobe After Effects*, Version 2024 (San Jose, CA: Adobe, 2024), https://www.adobe.com/products/aftereffects.html.

26. Scene Group, *Cavalry*, Version 2024 (London: Scene Group, 2024), https://cavalry.scenegroup.co/.

1) *GD with GD (Graphic Design with Gabriel Drozdov) by Gabriel Drozdov*

Figure 7: GD with GD

GD with GD is a platform dedicated to making creative and technical design education more accessible. It offers a collection of fun, free resources designed to help students and teachers explore graphic design, typography, web design, and creative coding.[27] The platform starts with basics like setting up a GitHub repository and gradually moves into HTML and CSS. And as users build confidence, they are introduced to JavaScript. It features a variety of projects, tools, and course materials that bridge traditional design principles with computational methods. What makes GD with GD different is how it presents information. Instead of a technical, code-heavy approach, it focuses on visual learning, making it easier for creatives to understand and apply new skills.

2) *Face painter by Nahuel Gerth*

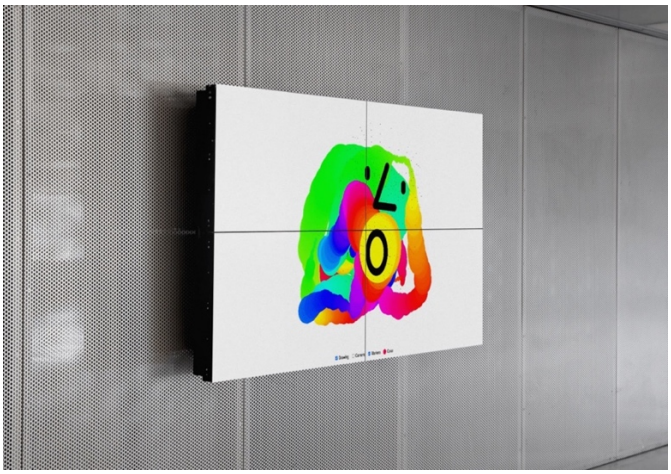


Figure 8: Face painter by Nahuel Gerth

27. Gabriel Drozdov, *GD with GD*, accessed February 17, 2025, https://gdwithgd.com/.

Face Painter is a tool that lets people use their facial expressions to draw on a screen. It was designed to help elderly users interact with technology in a fun and accessible way.[28] The interactions are simple; opening one's mouth adjusts the size of the brush, while head movements control the drawing on the screen. Designed as a tool for play and accessibility, it encourages self-expression in a way that is intuitive and effortless, even for those who may struggle with traditional digital interfaces.

3) *An Investigation of Humor in Design by Ritika Kedia*

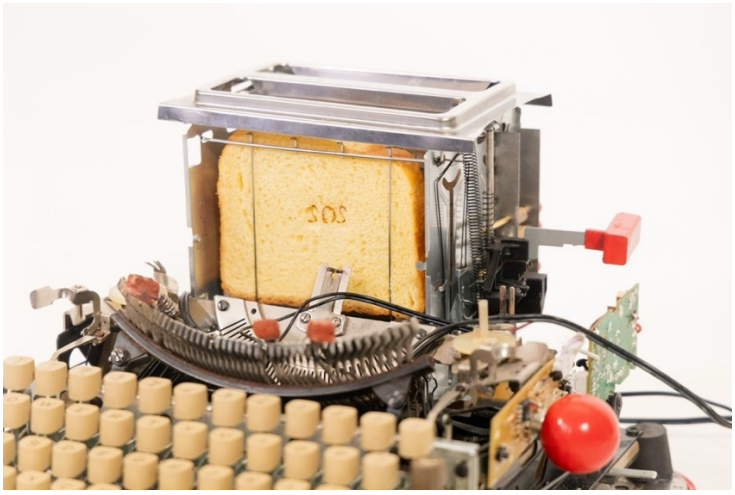


Figure 9: Humour in design by Ritika Kedia

An Investigation Humor in Design by Ritika Kedia is a project that examines how humor can be integrated into everyday objects to create more playful and engaging experiences. One of the key outcomes of this project is 'The Toaster-Typewriter'; a machine that combines the functionality of a toaster and a typewriter, challenging our expectations of how technology should behave.[29]

28. Nahuel Gerth, *Face Painter*, https://nahuelgerth.de/lab/face-painter.

29. Ritika Kedia, *Toaster Typewriter*, https://ritikakedia.com/Toaster-typewriter.

These projects show how computation and unconventional thinking can expand the possibilities of design, whether through education, accessibility, or humor. They demonstrate what it means to *make,* beyond conventional design tools, adopting computation as a medium in itself.

By integrating programming in their workflow, designers can expand the possibilities of graphic design into a space where design is not just created but also coded, assembled, and performed. It also helps designers build a more personal, experimental, and open-ended relationship with technology.

These ideas are the foundation of Radical interfaces, which expand the role of *making* in design. The result of it is a playbook; a set of principles that guides this way of thinking.

The Radical interfaces playbook

As I developed the prototypes in this project, I found myself returning to a set of recurring ideas and concepts that shaped my approach to *making*, hacking, and rethinking interactions. These ideas weren't strict rules but rather guiding principles that emerged through the process. I realized that beyond the interfaces themselves, this project was shaping an approach; one that valued play, openness, and the act of *making*. This was penned down in the form of a Radical interfaces playbook as follows:

The Radical interfaces playbook

1. *Technology Is Malleable*

   Technology is not rigid. It is meant to be bent, stretched, and poked at. A washing machine can be a musical instrument[30], a GPS app can be a canvas [31], and a toaster can be a typewriter.[32] The most compelling interfaces are the ones we invent for ourselves.

2. *Play is Serious Work*

   Fun needs to be taken seriously. Building interfaces that are playful, absurd, and surprising demands planning, patience and dedication. Behind every Radical interface is a relentless cycle of trial, iteration, and care.

3. *DIY Over Default*

   The most effective tools aren't always found; they're made. Work with what you have, adapt what exists, and create what's missing.

4. *Open, Not Closed*

   Design thrives when knowledge is shared. Radical interfaces are open-source, remixable, and accessible. They exist to be modified, adapted, and improved upon by anyone and everyone.

5. *Possibility, Not Productivity*

   Not everything has to be optimized, made efficient, or profitable. Some of the most meaningful designs come from embracing imperfection, experimentation, and the weird, wonderful spaces in between.

30. Schneider, Kurt Hugo. "HARRY POTTER Theme but Played on My WASHER & DRYER." YouTube, June 10, 2020. https://www.youtube.com/watch?v=HmltHR0uB9E.

31. Page, S. "Man Ran 700 Miles to Make 'Insanely Impressive' Art on GPS Fitness App." *The Washington Post*, 2024. https://www.washingtonpost.com/lifestyle/2024/12/02/strava-art-run-toronto-mccabe/.

32. Ritika Kedia, *Toaster Typewriter*, https://ritikakedia.com/Toaster-typewriter.

Methodology

In 1993, Christopher Frayling proposed a new method of research that focuses on generating knowledge through the process of designing, known as Research through Design (RtD).[33] Here, design itself becomes a process of inquiry. Rather than aiming for definitive solutions, RtD allows for exploration, iteration, and discoveries, which has been key in shaping this research. Following similar principles, Radical interfaces generated new knowledge through the act of *making*, treating design not as a means to an end but as an ongoing conversation between the *maker*, the medium, and the user. This emphasis on *making,* whether material, digital or other, highlighted different connections that formed during the process. These connections existed between the maker and the material, between tools and techniques, and between ideas and the act of creation. [34]

This approach made *Radical interfaces* a cyclical process of questioning, making, and iterating. It involved material exploration, learning new skills, taking things apart, understanding their mechanics, and figuring out how they could be reassembled in new ways. Each step led to new discoveries, shaping the direction of the next. The outcome of this process was five prototypes, each different in form and function but built on the same values, following the same processes.

The iterative process shown in the table below illustrates the stages in which all the prototypes were developed. Successful iterations are marked by (✓) and unsuccessful attempts are marked by (X), indicating challenges such as technical difficulties, skill limitations, or other practical constraints, discussed in detail within the individual prototype sections.

33. Isley, C. Grey, and Traci Rider. "Research-through-Design: Exploring a Design-Based Research Paradigm through Its Ontology, Epistemology, and Methodology." *Proceedings of DRS* 1 (June 28, 2018). https://doi.org/10.21606/drs.2018.263.

34. Erin Manning and Brian Massumi, *Thought in the Act: Passages in the Ecology of Experience* (Minneapolis: University of Minnesota Press, 2014)

Sr. No.	Name of Prototype	Description	Sketch 1	Sketch 2	Sketch 3	Sketch 4	Sketch 5
1.	I used Excel to make animations	An excel styled spreadsheet that makes illustrations and animations	✓	X	✓	-	-
2.	125	A 3D application that uses strings of words as coordinates	✓	-	-	-	-
3.	A font that inflates	A variable font that you can pump air into	✓	✓	X	✓	✓
4.	Animating flipdots	An animation pipeline for a dot matrix display	✓	✓	-	-	-
5.	Typing with a rotary phone	An obsolete technology repurposed to serve a new function	✓	-	-	-	-

Table 1: Stages of development in all the prototypes

The process of ideation is never singular. Most of the ideas come to me while doing mundane chores, observing mundane things. Inspiration is found in small, overlooked details. Sometimes, it's an observation, a question, or simply the absurdity of a situation. More often than not, these ideas don't come fully formed; they start as curiosities that grow and evolve eventually.

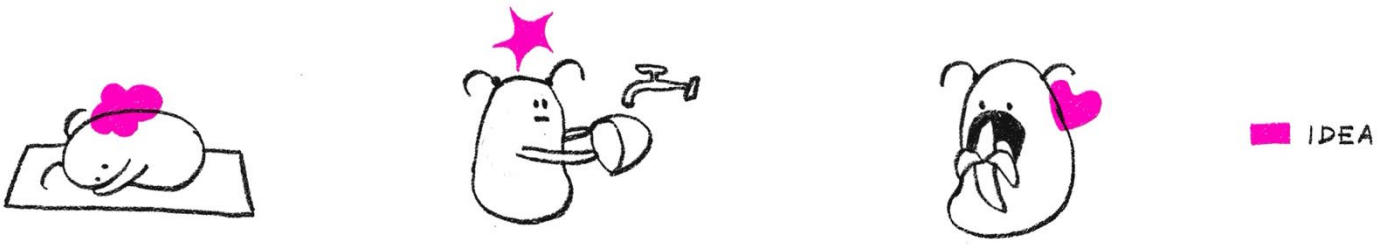


Figure 10: Ideation by author

Once an idea has formed, the next step is to pursue it through *making*, arguably the most exciting part of the process. This unfolds in three key stages. The first is planning, where I outline the steps. Next is identifying the materials needed, and an assessment of skills required to bring the idea to life and lastly, the actual process of *making*, which serves as the core of this research. This phase often leads to unexpected discoveries, insights about materials, new ideas, or even a shift in the original concept. With each learning, I usually loop back to the planning stage, refining the approach before moving forward again.

The process is cyclical, moving through repeated cycles of *making*, observing, and refining. Each iteration deepens the understanding of the subject, allowing the research to evolve in response to the act of *making* itself.

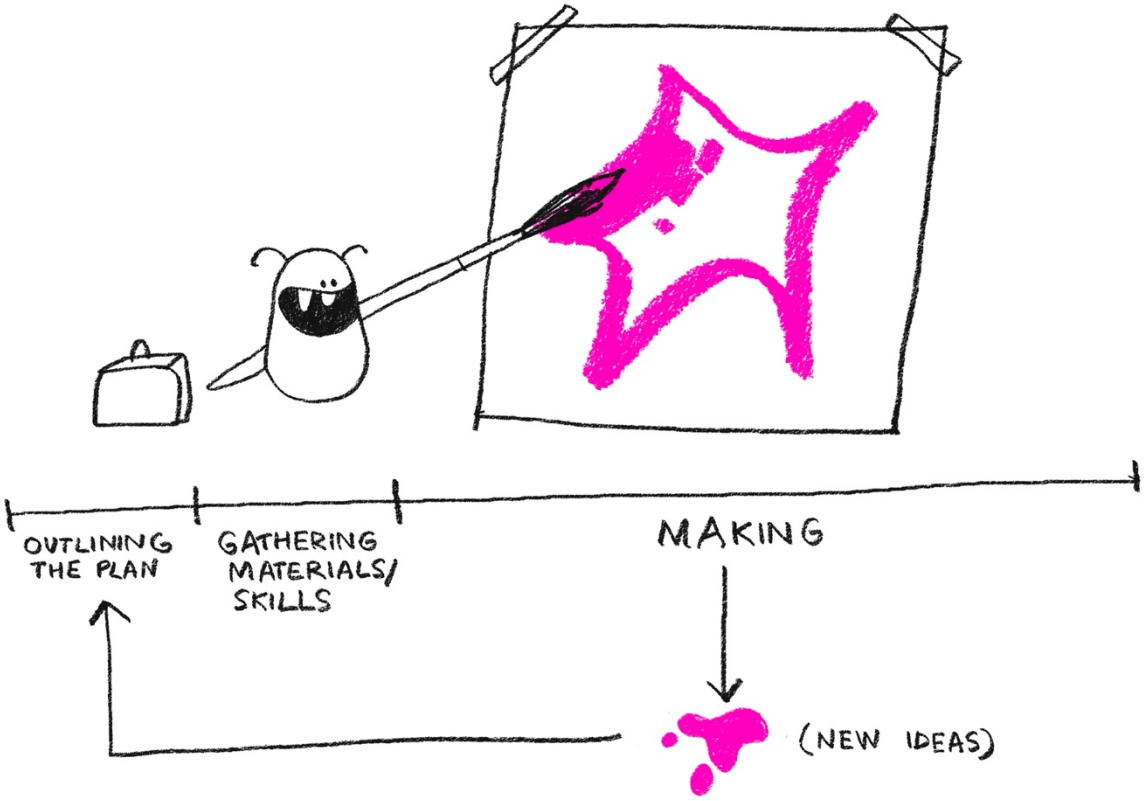


Figure 11: Making by author

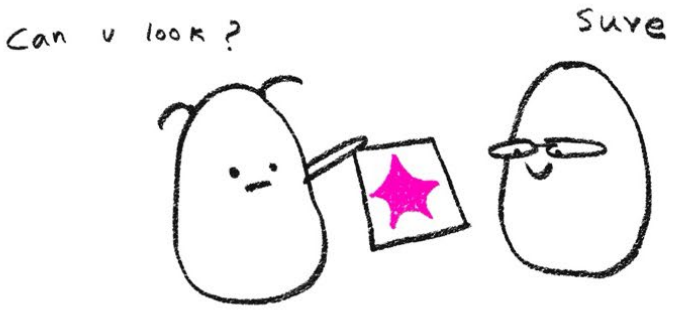


Figure 12: Sharing by author

The final and perhaps the most crucial step is *sharing*. Many of these prototypes were created in isolation, with my own perspective being the only critique. However, testing them with different users was essential. This feedback helped refine the work and push it beyond the controlled space which it was built in.

User testing for Radical interfaces was approached in 2 ways: Digital testing and Physical testing. To ensure ethical considerations were met, a Research Ethics Board (REB) review was conducted, and approval was granted on January 13th, 2025 (REB no. 2025-07). This allowed for structured and ethical engagement with participants, ensuring informed consent and responsible data handling.

For purely digital interfaces like 'I used Excel to make animations' and '125', I primarily relied on my own intuition to refine their usability before sharing them with a small group of 5 people. These users were given web links to access the tools remotely and were encouraged to explore them freely. Their feedback was collected through direct messages, annotated screenshots, and shared visuals of their experiments. While some feedback focused on usability, others demonstrated creative applications that I hadn't anticipated.

For physical interfaces, such as 'A Font That Inflates', 'Animating flipdots' and 'Typing with a rotary phone', an in-person user testing with 15 participants was conducted. No instructions were provided, allowing users to figure out the interactions on their own. Each participant brought their unique intuition, which influenced how they interacted with each interface. Observing their engagement firsthand gave me insights into usability, accessibility, and unexpected behaviours that might occur.

After interacting with the prototypes, participants shared verbal feedback, reflecting on their experience. Some described their initial confusion and how they figured out the interactions, while others commented on the intuitiveness (or lack of it) in certain designs. They also suggested potential improvements, pointed out any difficulties they faced, and in some cases, shared ideas for how the interfaces could be used in new ways. This exchange of ideas not only refined the prototypes but also brought together thinkers, makers, and users in the same space, enabling a collaborative dialogue that could take various shapes and forms.

Prototypes

This section outlines the motivation, design, development, testing and evaluation of the 5 prototypes. It details out the incremental stages of development, tools used, encountered roadblocks, etc. providing a peek into the iterative process of *making*.

a. I used Excel to make animations

After being appointed as the treasurer of the co-op I live in, I had to open Microsoft Excel [35] way more than I would have liked to. As I mindlessly entered budgets for the month, a silly thought came to my mind. This thought manifested into what I call the 'I used Excel to make animations.'

Sketch 1

(Note: Here, 'sketch' refers to an iterative stage of prototyping):

The first version of this prototype was a simple program developed using P5.js[36], inspired by the interface of Microsoft Excel. The goal was to replicate some of Excel's basic functionality while reimagining it as a creative tool. This prototype allowed users to resize rows and columns freely, mimicking the grid layout of a spreadsheet. Users could also toggle between black and white fills by clicking on individual cells. This feature opened up the platform for simple typographies, patterns, and illustrations.

By stripping Excel down to its essentials and reinterpreting its functions, this prototype laid the groundwork for exploring the *radical* with mundane tools.

35 Microsoft Corporation, *Microsoft Excel*, Version 2024 (Redmond, WA: Microsoft, 2024), https://www.microsoft.com/excel.

36 Lauren McCarthy, Casey Reas, and Ben Fry, *p5.js*, Version 1.6.0 (Processing Foundation, 2024), https://p5js.org/.

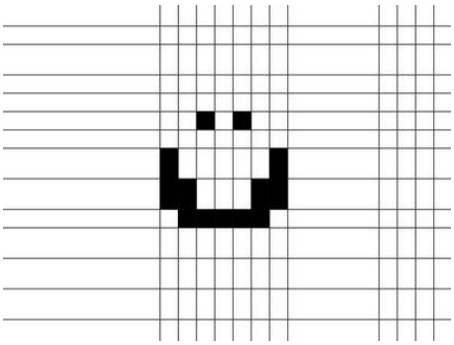


Figure 13: Smiley face on Sketch 1

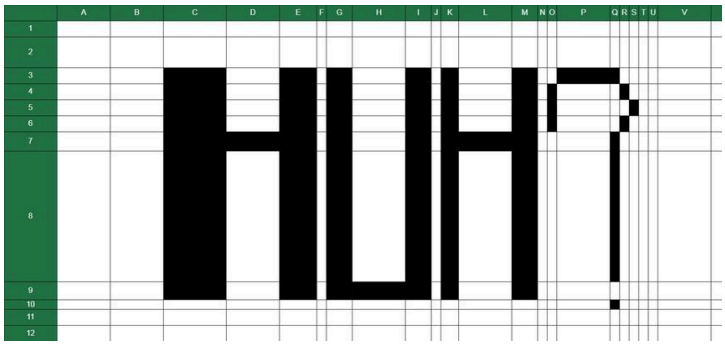


Figure 14: Typography on Sketch 1

Sketch 2:

The concept of this sketch was to wrap the Excel grid around three-dimensional objects. What excited me most about this sketch was the dimensionality it could introduce to a traditionally flat, two-dimensional design. By extending the functionality of Sketch 1 - featuring the flexible grid and the ability to toggle the fill of individual cells, I envisioned integrating these elements with three-dimensional objects. The idea was to start with simple shapes like a cube and a sphere, and eventually progress to more complex structures, like the warped space-time fabric.

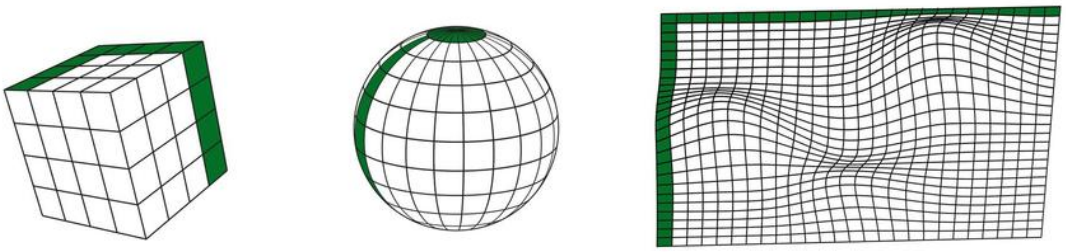


Figure 15: Sketch 2 Mock-up

Scalability of the application was another aspect, as it could theoretically be applied to any three-dimensional object while maintaining Excel's inherent adaptability. However, Sketch 2 remains unrealized to date due to its technical complexity.

Sketch 3:

After the failure of Sketch 2, I pursued a more reasonable idea, where I introduced a few additional features to Sketch 1 to enhance its functionality. New buttons were added to the console, including a round button and a triangle button alongside the existing fill button. These new tools allowed me to generate slightly more dynamic images in comparison to the previous sketch 1. To further improve usability, I introduced a few colour options at the bottom of the interface, enabling more customization. Additionally, a download button was incorporated at the top, making it easy to save the generated designs.

While the design has some limitations, it can still be customized. Users can modify the source code to add their own shapes, images, and colours, adapting it for different projects.

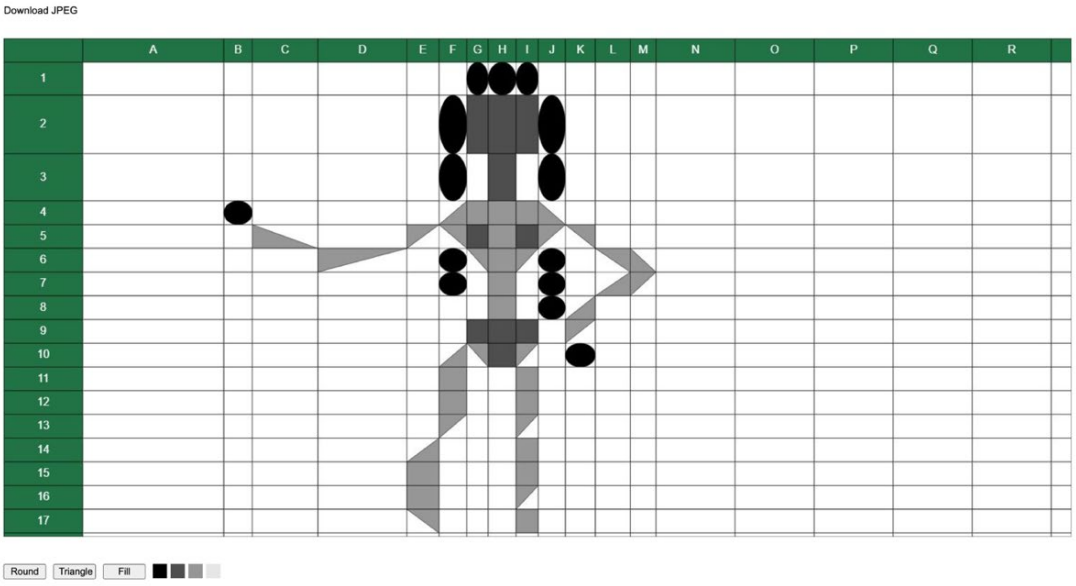


Figure 16: Illustration by Mandar Mhaskar on Sketch 3

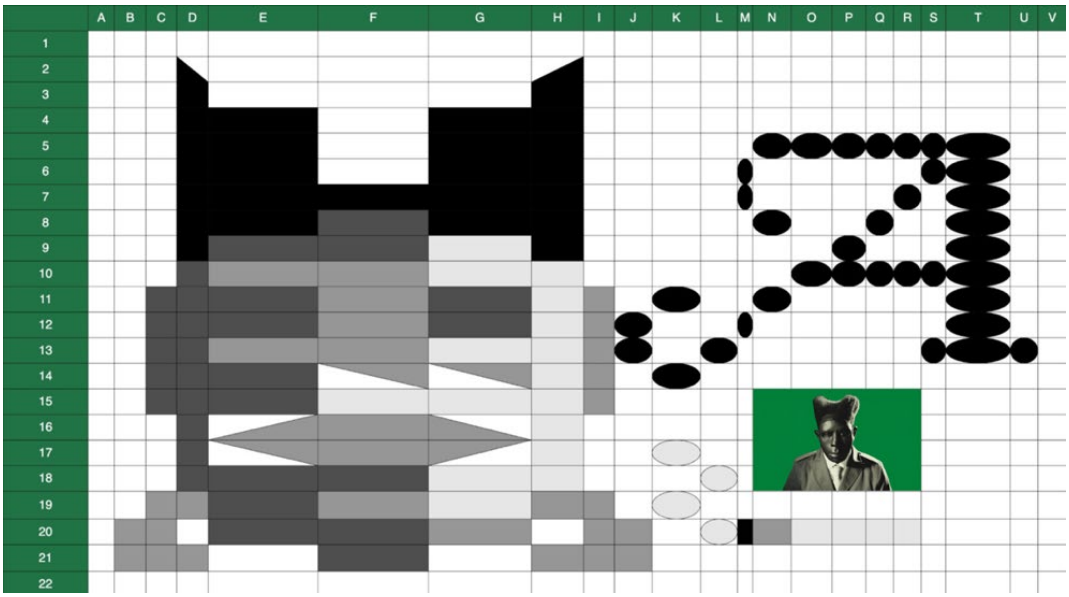


Figure 17: Anushka Shinde's sketch of Tyler, the creator from Chromakopia

People experimented with the spreadsheet interface with links posted online. Some treated individual cells like building blocks or pixels, filling them with different shapes that combined to represent a form. Others tried to mimic reference photos, to capture the likeness and subtle details of the subject, like the light and shadow effect. This approach allowed them to suggest volume and dimensionality to the illustration, even within the limitations of a basic spreadsheet interface.

While I thought this tool would mostly be used for creating illustrations, it ended up serving a surprising purpose during a co-design session where participants used the shapes and colour options to plan out a classroom space based on their preferences. By creating a simple map using the limited options, they marked where they wanted private work areas, collaborative zones, teaching spaces, etc. These visuals helped participants show how they envisioned their ideal classroom in a collaborative and illustrative manner.

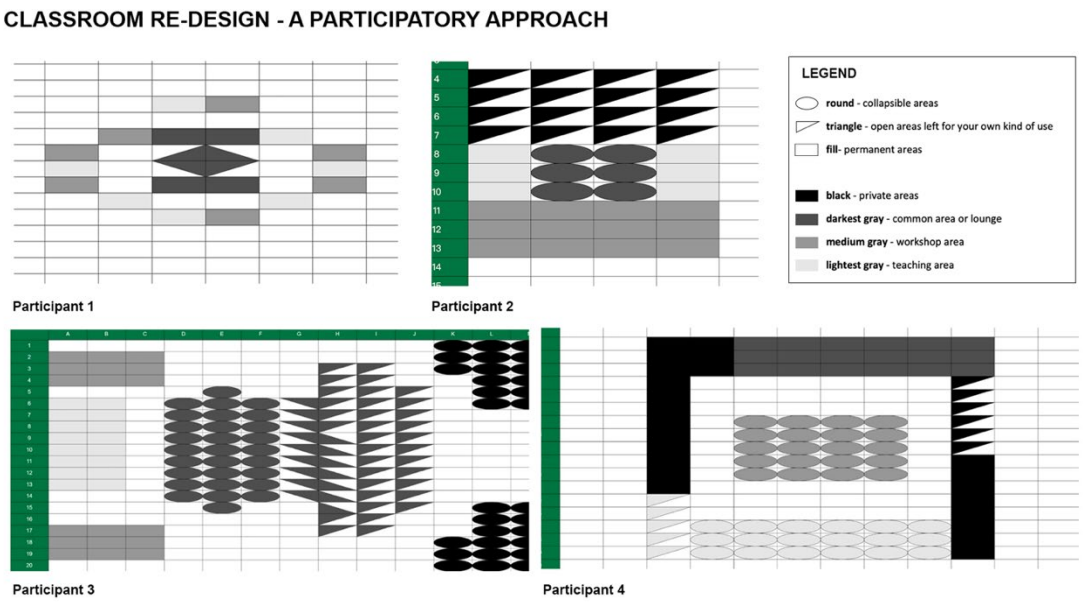


Figure 18: Outcome of the co-design session

Happy Accidents: The download button on the top led to some unexpected outcomes. While working on a particular illustration, I ended up downloading a few extra frames with slight changes in each. When I compared those images, I realized they created some motion when viewed together. This accidental discovery eventually led to my first intentional animation on the interface. What started as a mistake turned into a feature.

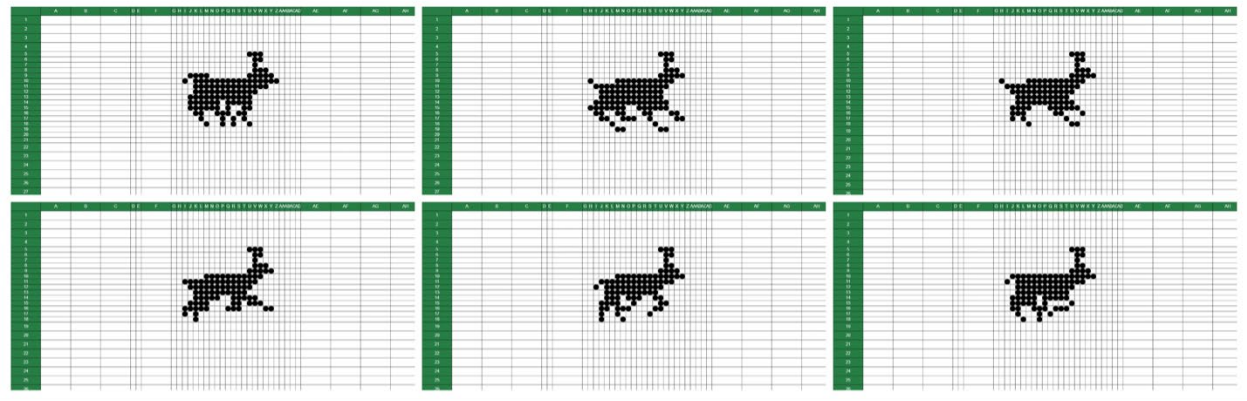


Figure 19*: First animation on Excel Graphics*

A link to try the app ↗

GitHub link for code ↗

A sketch that *could have been*:

'I made animations on Excel' started as a simple illustration application had somehow taken the shape of a rudimentary animating software. This transition happened naturally. The flexibility of the grid was a standout feature, which allowed me to easily stretch and squeeze shapes, very useful while animating.

I shared the app with a few animators and had informal discussions with them about how it could be improved. They provided some great suggestions for features that are essential in an animation software:

a. *Adjacent Frame Visibility:* They explained that in animation, it's important to see adjacent frames at a lower opacity to better visualize motion while working on the current frame. To address this, one friend suggested adding a "sheet" system, similar to the tab functionality at the bottom of Microsoft Excel. This would let users layer sheets on top of each other and view the previous sheet at a reduced opacity.

Figure 20: Sheet system on Microsoft Excel

b. *Play Button:* Another useful suggestion was to include a play button to cycle through all the frames and preview the animation.

c. *Download as Zip:* Lastly, they recommended a feature to download all frames as a zip file for easy export and further use.

b. 125

125 was born as a sequel to my frustration of not being able to transform the Excel grid on a three-dimensional object. I decided to make a 3D app.

Despite trying several 3D modeling tools like Blender, Maya, and Cinema 4D, I still find their interfaces incredibly intimidating. Most of my time on these softwares is spent orbiting, panning, zooming in and out rather than creating. For this, I decided to develop my own 3D software using HTML, CSS and JS that removes all the features I find overwhelming or distracting (along with quite a few others, admittedly).

"125" maps 3D pixels in a three-dimensional coordinate system. This program takes a slightly longer and more cumbersome route to achieve something that could likely be done more efficiently with better features and functionality. That being said, in earlier sections, we talked about the slow and rewarding process of the handmade, and in some ways, this project intentionally takes a longer route to find meaning along the path.

A link to try the app ↗

Sketch 1:

(Note: This application was built on a single sketch. Further iterations were conceptualized as future explorations but never actually made.)

I first modelled a generous cube consisting of 5 x 5 x 5 cubes inside it (which explains the name of the app – 125) on SketchUp[37]. Once the structure was ready, I exposed the wireframes and traced them precisely in Illustrator. This tracing provided me with a clear guide for creating the three-dimensional interface.

37. Trimble Inc., *SketchUp*, Version 2024 (Sunnyvale, CA: Trimble Inc., 2024), https://www.sketchup.com/.

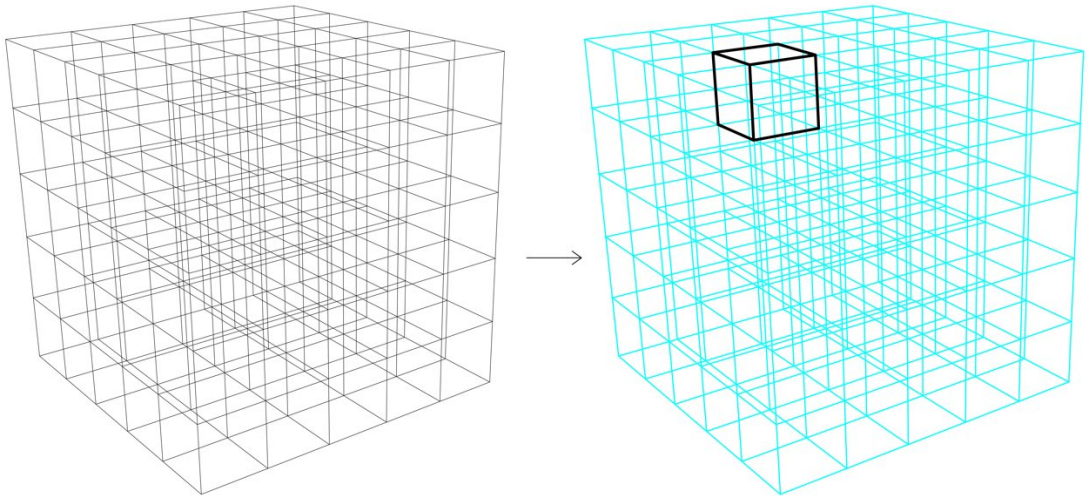


Figure 21: 3D interface guide

Since this app worked a bit differently from traditional 3D building software, I thought it would be interesting to let the user figure out how it worked - presenting it as a puzzle. Instead of providing clear instructions, the user would try to understand the workings of the app through trials and discovery.

As mentioned previously, the app mapped 3D pixels within a 3D coordinate system. To summon these pixels, users were required to input the X, Y, and Z coordinates. Each of these coordinates was assigned specific words, which served as subtle hints on how this application must be used when read together. This approach challenged the traditional expectations of software usability while making the act of discovery and reward a part of the process.

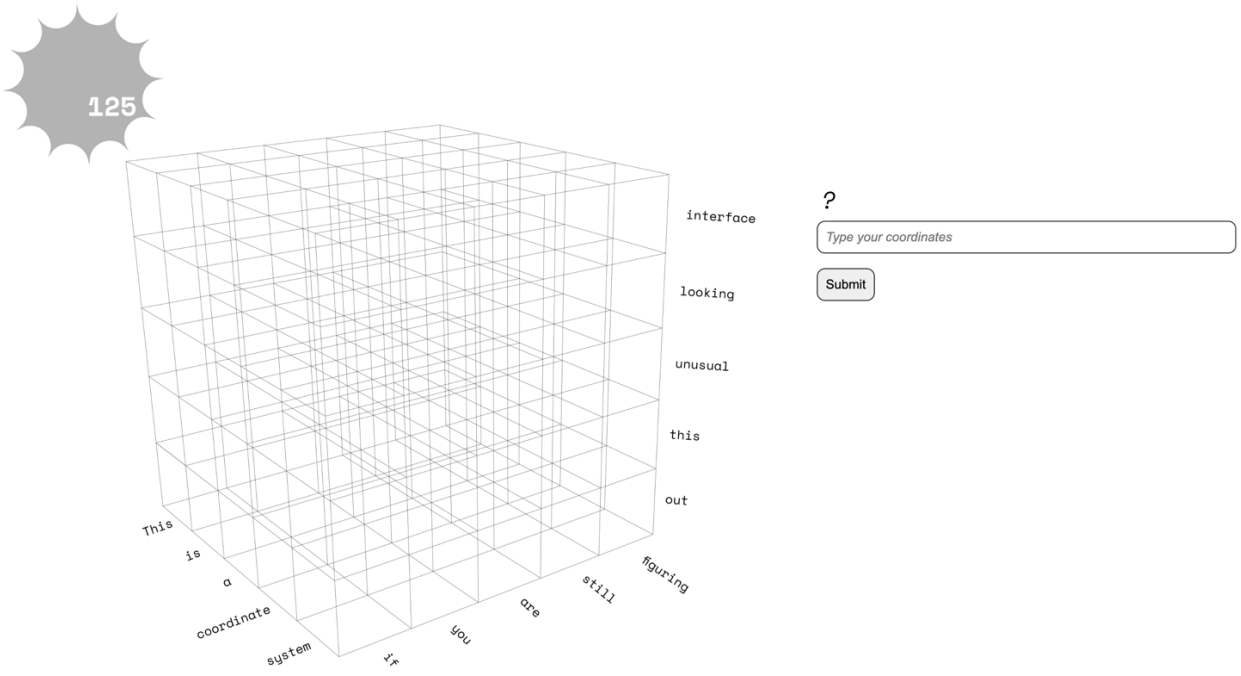


Figure 22: Final interface

How was this made?

Each of the 125 cubes that could ever be summoned on this app was painstakingly drawn in its exact location using Illustrator. These cubes were then exported as 125 individual PNG files, each named according to its corresponding coordinates (e.g. system if out). The coordinate names were assigned as the file names during the export process, which had to be done one cube at a time. In the app, inputting these coordinate-based names would reveal the corresponding hidden images on the screen.

While this approach was mainly taken due to a lack of technical skills, it highlighted the value of handmade work. The slowness, intentionality and the lack of automation added materiality to this otherwise digital process. The cubes felt more tangible over time, that were individually crafted and assembled.

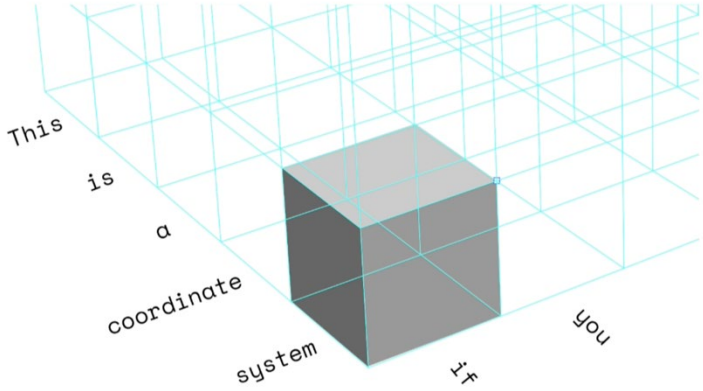


Figure 22: Process: drawing a cube on illustrator

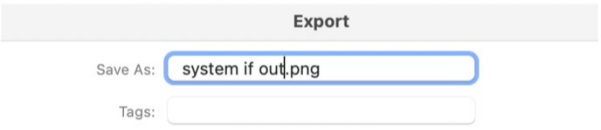


Figure 23: Process: exporting a cube on illustrator

X 125 times

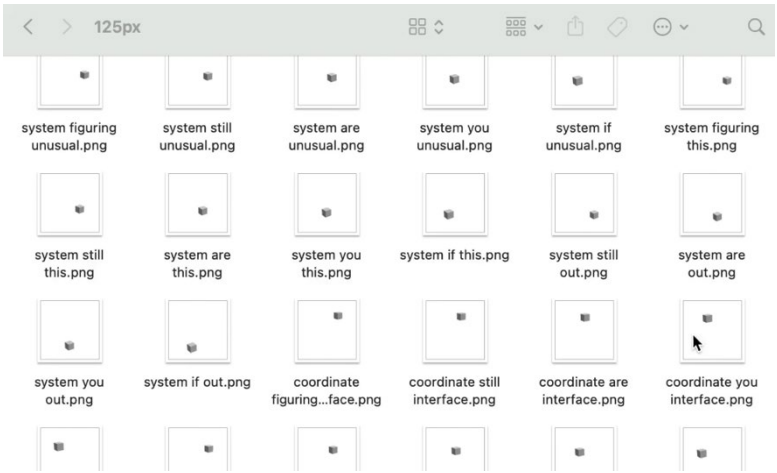


Figure 23: Exported PNG's

GitHub Link ↗

What did I make out of it?

I used this app to create some three-dimensional typographical explorations. Admittedly, my opinion might be biased toward the app I built myself, but it evoked similar emotions of materiality and tangibility during the modelling process.

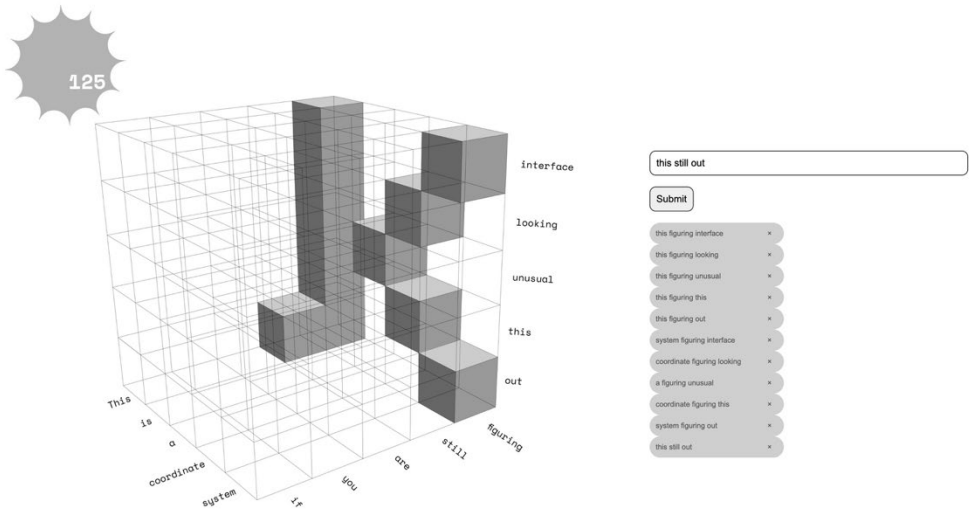


Figure 24: Letter K recipe – this figuring interface, this figuring looking, this figuring unusual, this figuring this, this figuring out, this still out, system figuring interface, coordinate figuring looking, a figuring unusual, coordinate figuring this, system figuring out

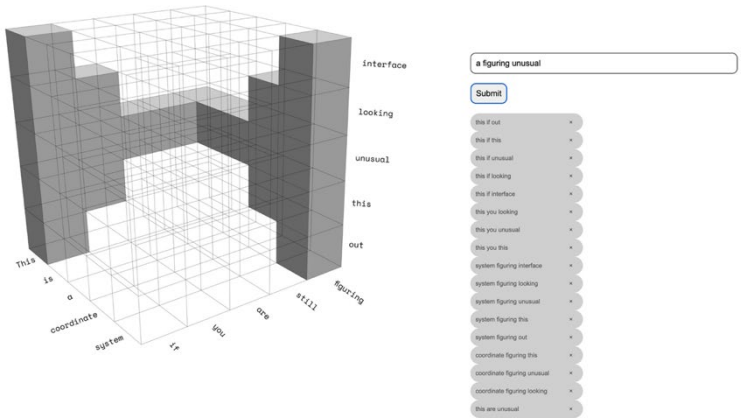


Figure 25: Letter H recipe: this if interface, this if looking, this if unusual, this if this, this if out, this you looking, this you unusual, this you this, this are unusual, this still unusual, this figuring unusual, is figuring unusual, a figuring unusual, coordinate figuring unusual, coordinate figuring looking, coordinate figuring this, system figuring interface, system figuring looking, system figuring unusual, system figuring this, system figuring out

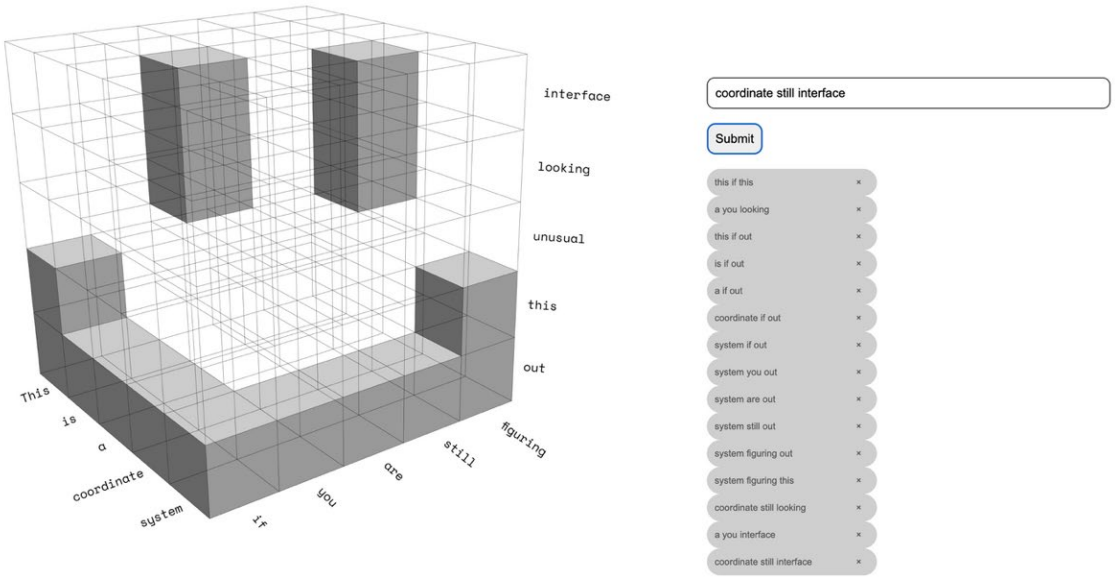


Figure 26: Smiley face recipe: this if this, a you looking, this if out, a if out, coordinate if out, system if out, system if out, system you out, system are out, system still out, system figuring out, system figuring this, coordinate still looking, a you interface, coordinate still interface

A sketch that could have been:

As I drew and exported the 125 images, one image at a time, there came a point where I questioned this decision and noted down all the points on what ideally should have been the sketch.

a. An Automated System: Instead of manually drawing and exporting, the system would generate pixels automatically. These pixels wouldn't have to be limited to cubes, they could be spheres, triangles, or *insert_a_shape_of_your_preference.*

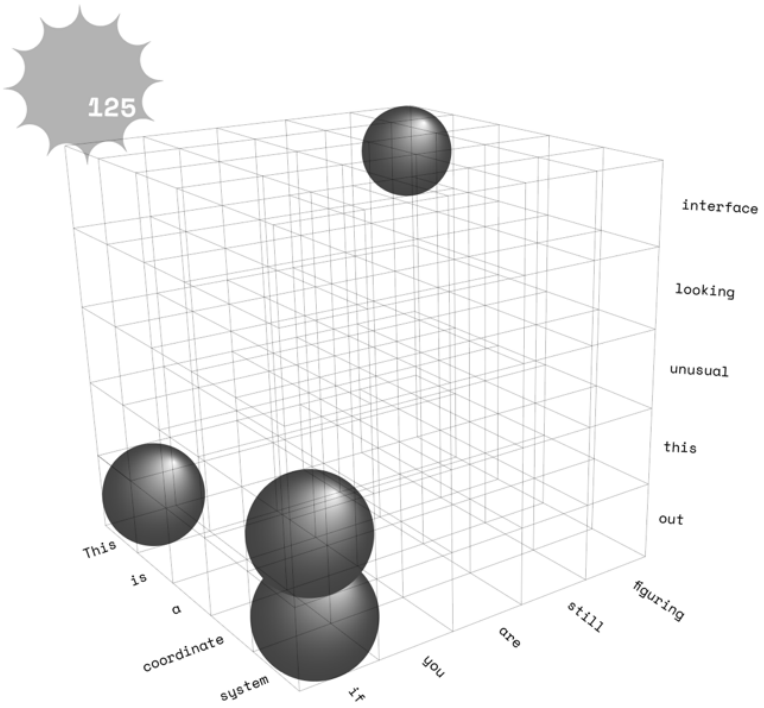


Figure 27: Spherical Pixels mock-up

b. Scalable Pixels: The size of the pixels would be adjustable, allowing for more flexibility in creating designs.

c. Colour Options: The system would include basic colour options to make the outputs more diverse.

d. *The Grid Could Be Used in So Many Different Ways!!*: One fascinating application of this grid could be inspired by *Tipkyanchi Raangoli*, a traditional art form from the Maharashtrian community in Western India (a community I come from). This design is created during the festival of Diwali using rice powder and colours to form symmetrical patterns. The *Tipke* (meaning dots) act as a guide for placing the powder, which is then used to create these beautiful shapes and filled with vibrant colours. This web app could take the concept of the *Tipke* and translate it from a 2D format into a 3D grid, opening up possibilities for multiple permutations and combinations.

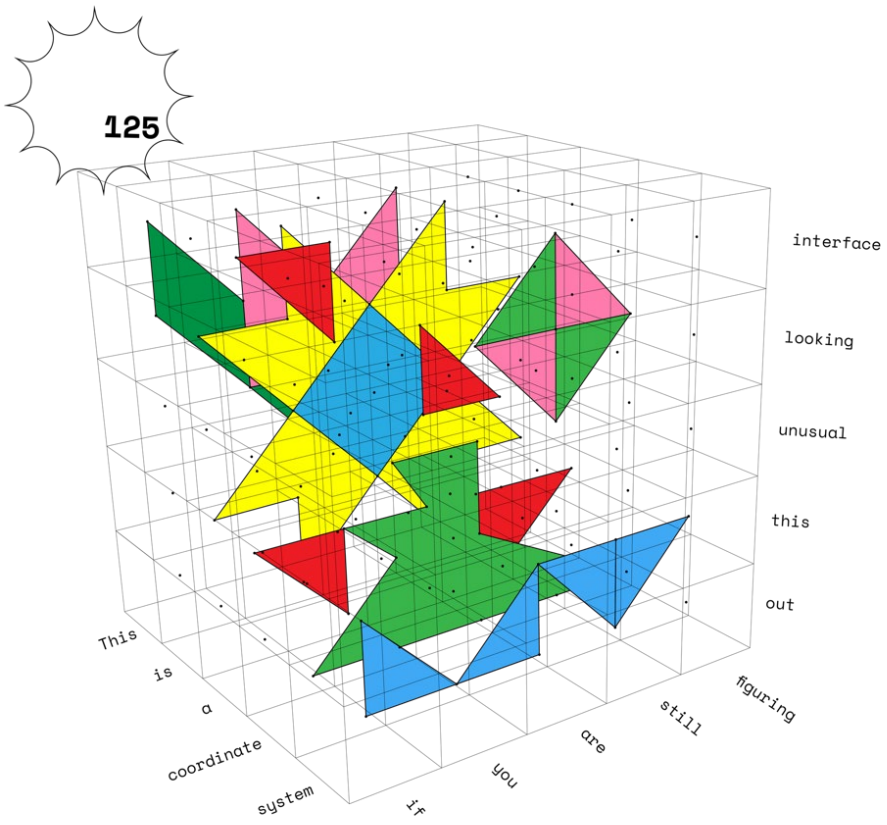


Figure 29: 125 re-imagined as a Tipkyanchi Rangoli

Figure 30: Kanishtha, Abha and Aloran making a Tipkyanchi Rangoli during Diwali 2022

Figure 28: The finished rangoli

c. A font that inflates

After two radical web apps, it was time to introduce some tactility and physicality into the interfaces. Being a graphic designer, I am incredibly fond of fonts and typeface design. Recently, I attempted to create a variable font, and I failed.

The process for making a variable font includes paying for a software (Glyphs) only to realize that it works exclusively on iOS. While there are other tools available, none of them come close to offering the capabilities and features that Glyphs provides. Even if you manage to overcome the initial barriers of access and compatibility, the actual process for building the font is incredibly cumbersome.

Designing a variable font* requires planning each axis of variation, such as weight, width, or slant, and ensuring that all interpolations work seamlessly. This involves creating multiple master designs and fine-tuning the transitions between each of them. Every curve, anchor point, and spacing detail needs to align perfectly, and even minor inconsistencies can disrupt the entire process. The sheer precision and attention to detail required can make the process feel overwhelming, especially for someone new to this, which led me to rethink my entire approach to typeface design.

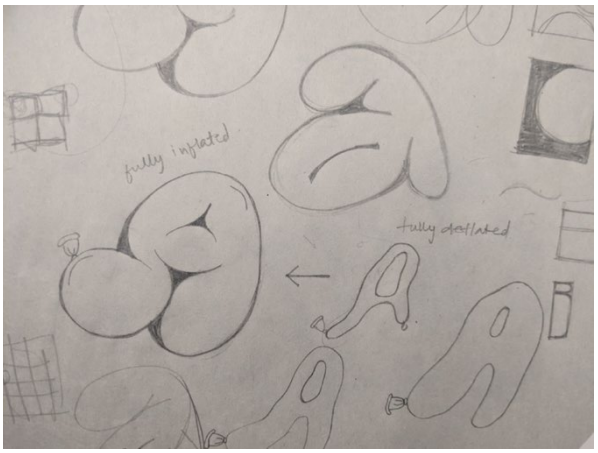


Figure 29: A sketchbook doodle of a font that inflates

*Variable fonts typically are fonts whose Weights (Regular, Medium, Bold, etc.), Widths (Condensed, Extended) and vertical axis' (Italics) can vary.

But what if I want a font that inflates and deflates like a balloon?

Sketch 1:

The first sketch was created using Processing and the Minim library[38]. The sketch worked by increasing the font size of a letter based on the volume of the audio input. As the audio grew louder, the letter would scale up proportionally. Watch video ↗

Figure 30: Sketch 1- Increases font size with sound

Sketch 2:

A GIF of the alphabet A was created to respond to increasing and decreasing audio inputs. The audio threshold was lowered so that the user would need to blow air near the computer microphone, mimicking the action of blowing a balloon. This GIF was designed using traditional drawing and animation software - Procreate[39] and Photoshop. Watch Video ↗

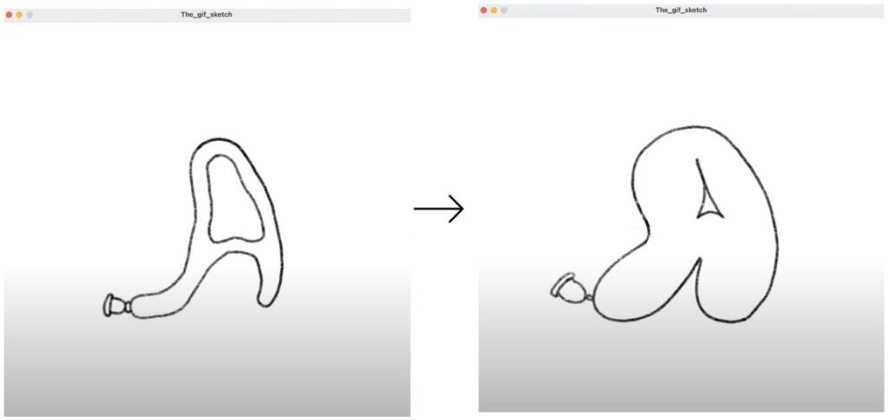


Figure 31: Sketch 2

38. Damien Di Fede, *Minim: An Audio Library for Processing*, Version 2.2 (Processing Foundation, 2024), https://github.com/ddf/Minim.

39. Savage Interactive, *Procreate*, Version 5.3 (Hobart, Australia: Savage Interactive, 2024), https://procreate.com/.

Sketch 3:

In this sketch, a pink balloon was introduced. The idea was to map the pink pixels on the screen to define how much the letter should be inflated or deflated. However, I encountered a roadblock with this approach. The pink colour of the balloon varied under different lighting conditions, making the mapping inconsistent. Additionally, as the balloon inflated, the pink colour faded and became whiter, further complicating the process.

Figure 32: Pink balloon losing its pinkness

Sketch 4:

This sketch optimized the code by incorporating machine learning with p5.js.[40] Using Teachable Machine[41], I trained the model with several images of deflated and inflated balloons. This process created two classifiers: "Inflated" and "Deflated." Based on these labels, the p5.js code determined whether to play the GIF in the forward direction (for an inflated balloon) or the reverse direction (for a deflated balloon), effectively linking the balloon's state to the animation in real-time. Watch Video ↗ p5.js ↗

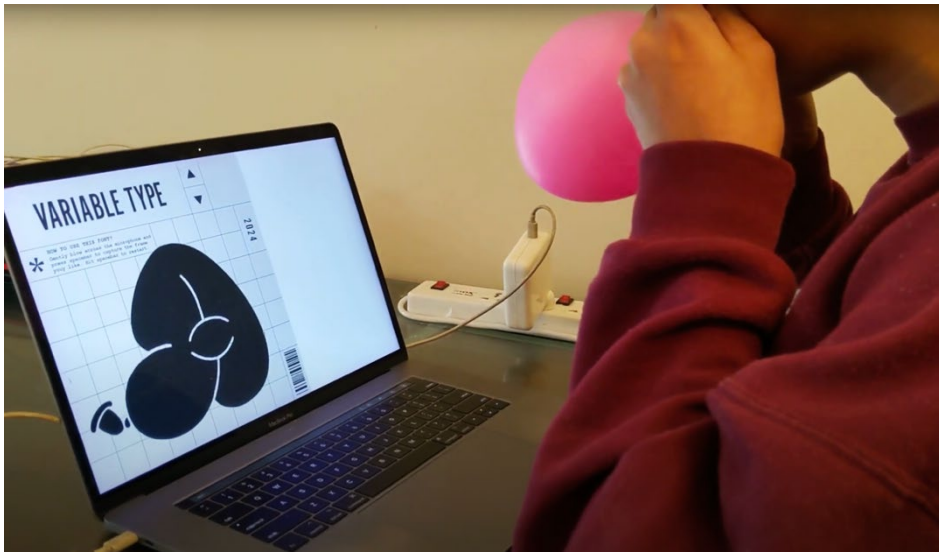


Figure 33: Sketch 4

40. The Coding Train, "Teachable Machine 1: Image Classification." YouTube video, November 7, 2019. https://www.youtube.com/watch?v=kwcillcWOg0.

41. Google, *Teachable Machine*, Version 2024 (Google, 2024), https://teachablemachine.withgoogle.com/.

Sketch 5:

The whole point of a variable font is its ability to transition smoothly between two or more points, offering a range of states rather than fixed extremes. The issue with Sketch 4 was that it only recognized the extremes - fully inflated or completely deflated. There was no way for the GIF to pause or stop at a frame between these two states.

To truly mimic the behaviour of a variable font, we needed a way to access the intermediate frames. This would require a physical gadget or input mechanism that could continuously detect and map the gradual states of the input, providing finer control over the variable font. To address this issue, I created a DIY potentiometer[42] using conductive fabric, a floating pin (commonly known as a wiper) and a microcontroller. This setup functioned as a slider, allowing me to map the position of the floating pin based on its proximity to either ground or voltage. The closer the pin was to the voltage, the higher the value, and vice versa. This solution helped capture a continuous range of values, which were then corresponded to all the frames of the alphabet.

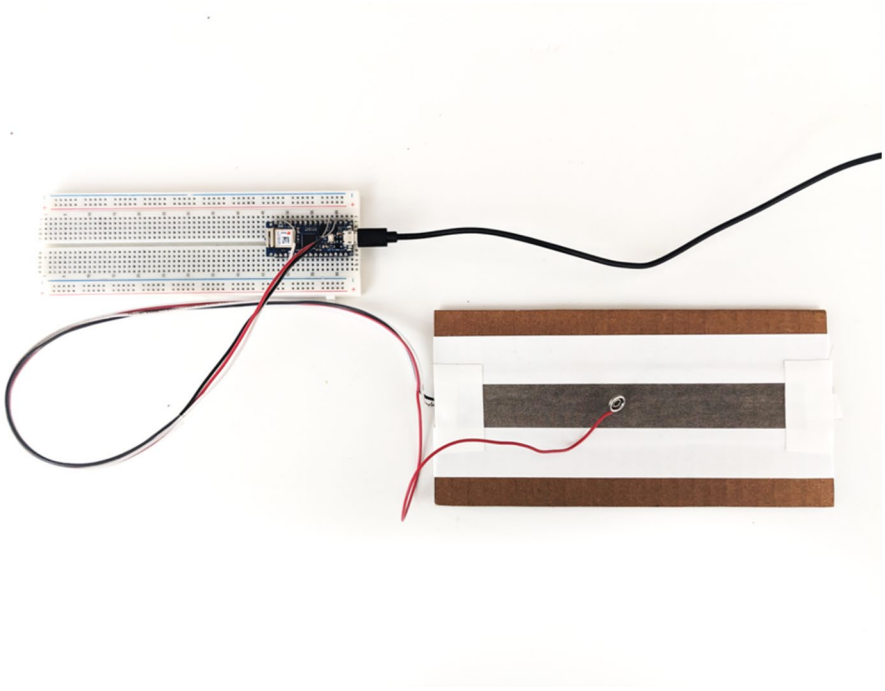


Figure 34: DIY potentiometer

42. Bare Conductive. "Make A Potentiometer With Electric Paint." Weblog post. February 13, 2021. https://www.bareconductive.com/blogs/resources/making-a-potentiometer-with-electric-paint?srsltid=AfmBOooqYZzEqMbTjqF7mpbmvZ5y9xKL8Y9AzkOykPY2hYzAZ1HFSYU3.

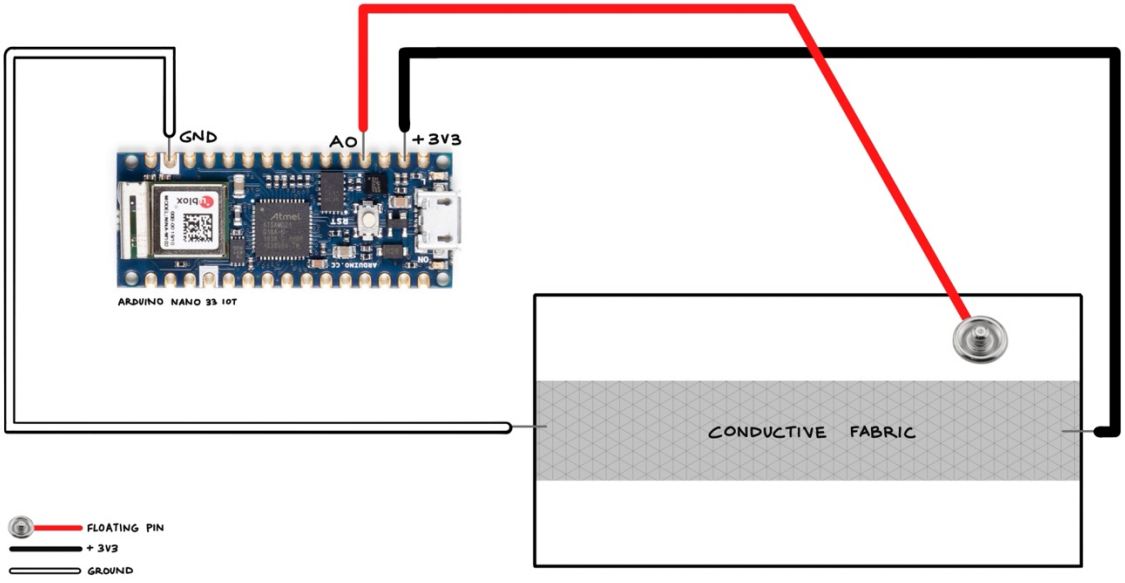


Figure 35: DIY Potentiometer circuit diagram

Since this technique used a slider mechanism, where the different positions of the floating pin mapped a range of values, I realized it was possible to repurpose existing toys to recreate a similar but better-looking potentiometer. In this context, an air pump made the most sense. The action of pumping air would directly control the font on the screen, linking the physical action to the screen-based output.

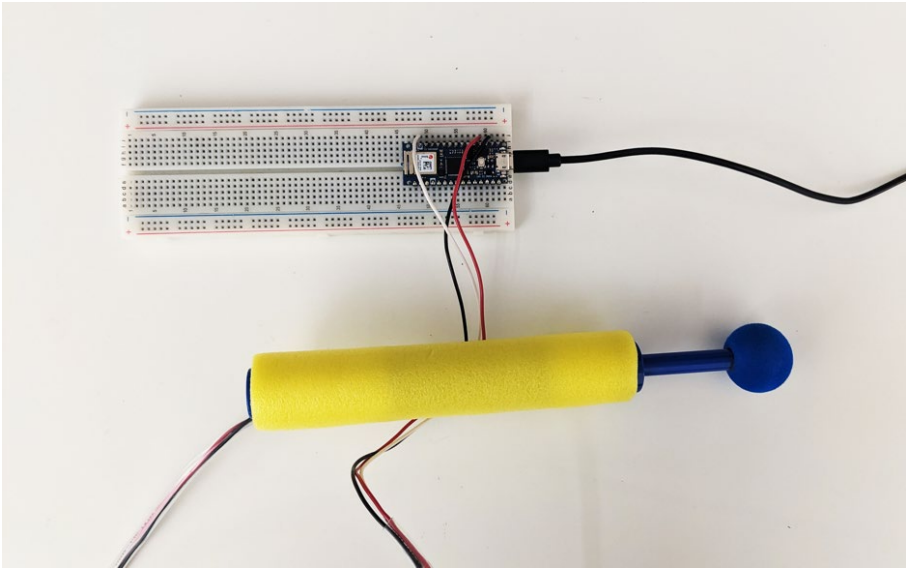


Figure 36: Air pump as potentiometer

The tube of the air pump was lined with conductive fabric on two opposite sides, on the inside. The piston, coated with aluminum foil, acted as a floating pin that connected these two sides as it moved. This setup allowed the piston to function as a slider, completing the circuit at different points along its path. As the piston moved up and down, it generated a range of values depending on its relative position to the ground and voltage. The closer it moved to one end, the higher or lower the values became. This system effectively turned the air pump into a DIY potentiometer, where the physical act of pumping directly controlled the variability of the alphabet.

Figure 37: Air pump as potentiometer

Figure 36: Piston coated with aluminium foil

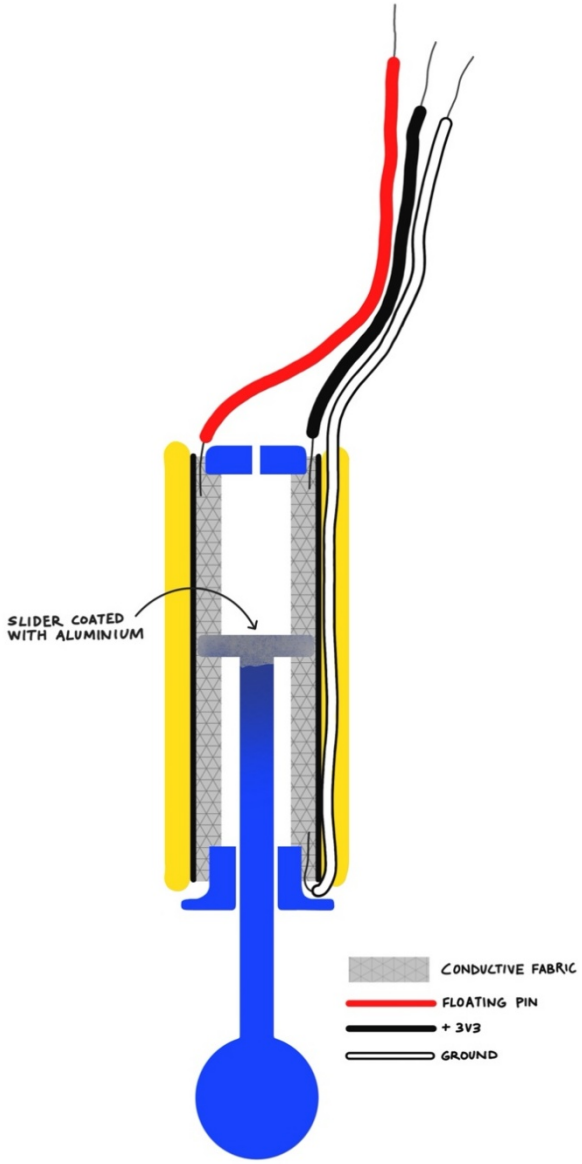


Figure 37: Cross section of the air pump

This apparatus was created using Arduino IDE and Processing. The Arduino handled the physical input from the air pump, reading the values generated by the piston's movement and then sent it to Processing. Processing then visualized this data to as the font inflation or deflation on the screen.

Watch Video ↗

What next?

The core functionality of the system is now in place. The next step involves the more labor-intensive process of designing the complete set of letters (B–Z) and the number system (0–9), with each character requiring 21 in-between frames to allow smooth transitions.

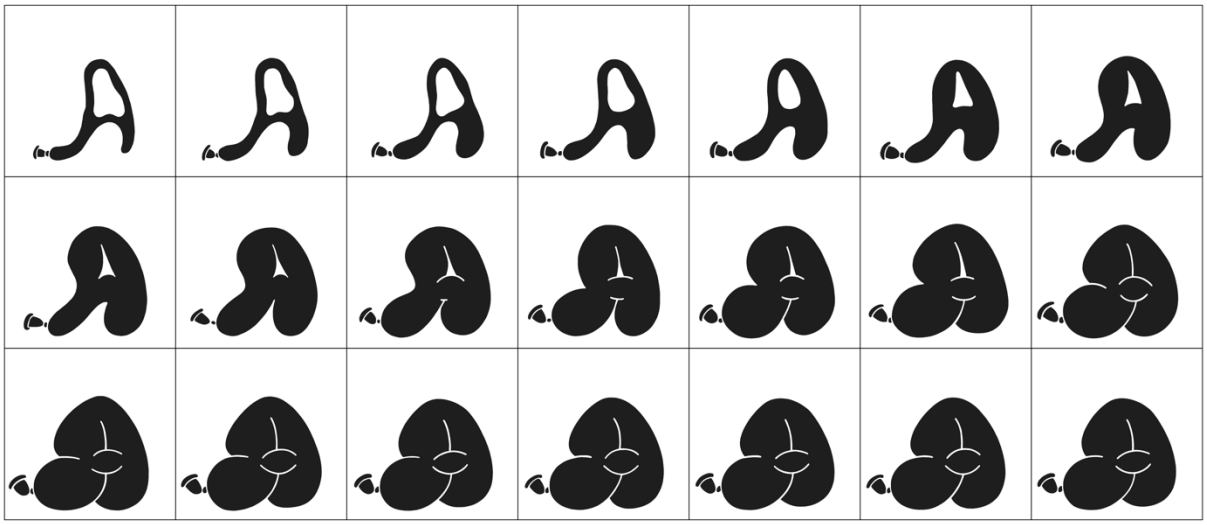


Figure 38: The letter A

Further exploration is needed to determine how all these letters would come together to form a string of words and sentences, as well as how they can be controlled, either individually or collectively through the air pump interface.

User Testing:

Goal: Through personal evaluation, I was confident in the sensitivity, feedback, and ergonomics of this interface. So, the goal of this user testing was to compare two different pieces of code. In the first version, the sensor was mapped between the lowest and highest values of the air pump, directly corresponding to the least and most inflated states of the alphabet, with smooth transitions between them. The interaction was straightforward; the digital output always reflected the position of the wiper in the air pump.

The second version introduced a more organic interaction. Instead of a direct mapping of values, this version mimicked the real-world experience of inflating a balloon with a small hole. Participants had to continuously pump to keep the letter inflated, as it would slowly deflate if there was no change in the potentiometer values.

Setup: The setup included a small CRT (Cathode-ray tube) screen display with a curious little air pump placed in front of it. The screen displayed the words: 'A font you can pump air into,' which clearly indicated what the user was expected to do.

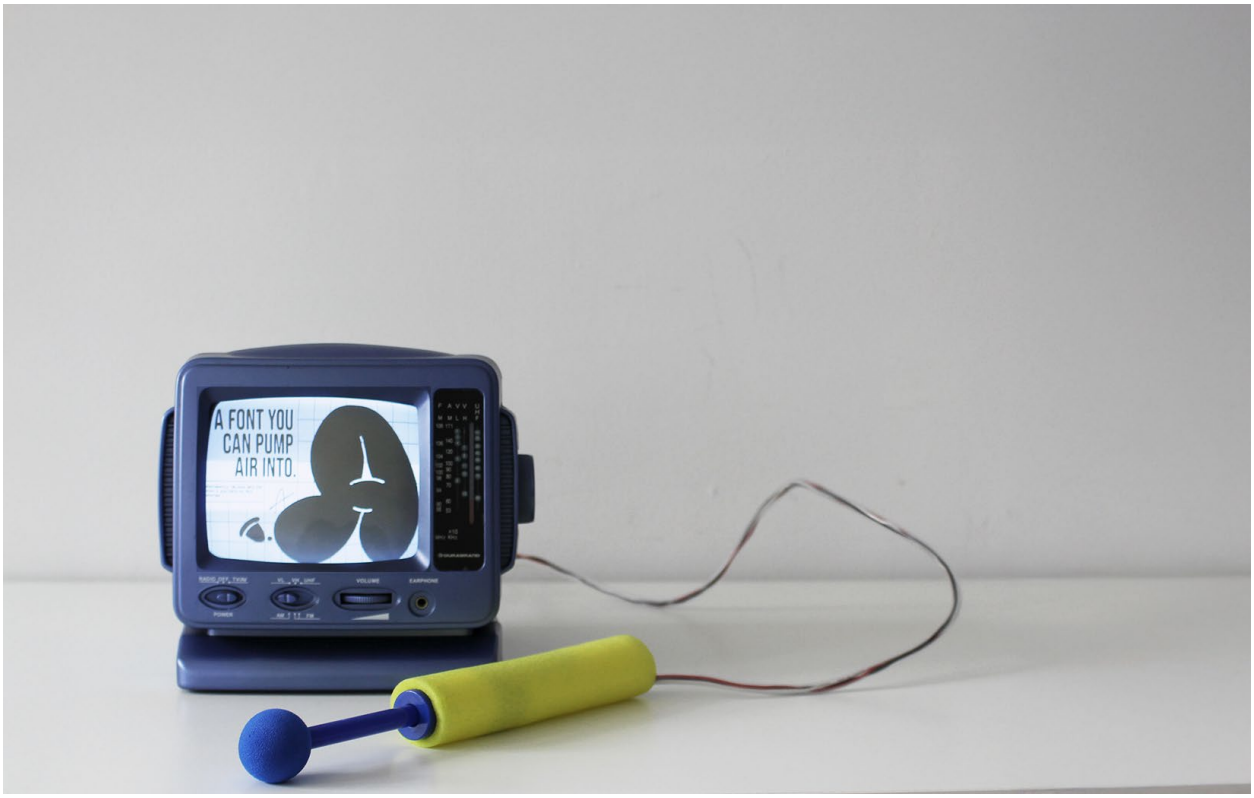


Figure 39: User testing setup

Interactions: Participants promptly picked up the air pump and pumped ‘air’ into it. As the alphabet responded to their action of pumping, they asked questions like - ‘*Is it sensing the air pressure?*’

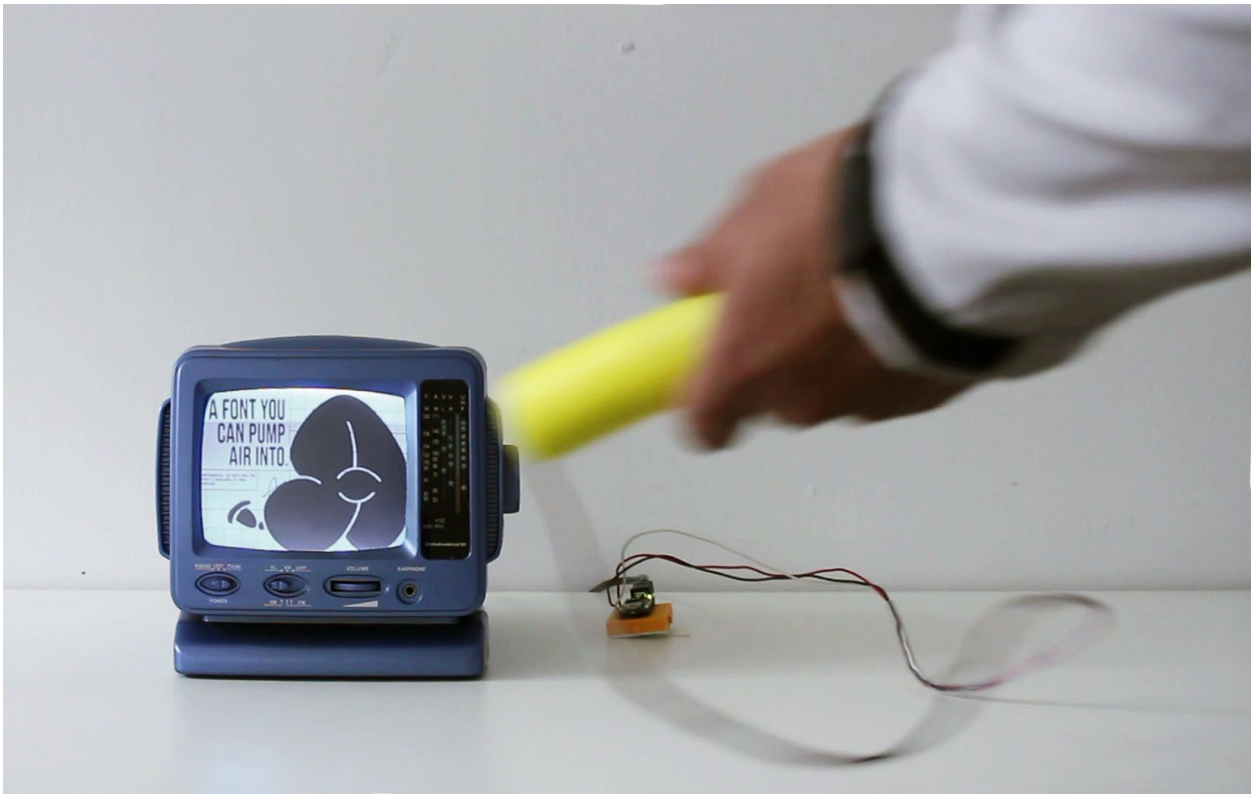


Figure 40: Participant pumping 'air' into the font

While testing the second code, I asked the participants to maintain the alphabet in an in-between state; neither fully inflated, nor completely deflated, for at least 5 seconds. It resulted in an interesting play of trying to pump just enough to maintain that equilibrium.

Insights: Between the two versions of the code, I observed that participants engaged with the second one for a longer period. Participant feedback indicated that in the first version, once they pumped air, their task was essentially complete, making the interaction feel more passive. However, in the second version, they had to continuously pump to maintain the inflated state, as the letter would slowly deflate if not actively pumped. This ongoing engagement made them feel more involved, turning the experience into a challenge rather than a one-time action.

‘*Can the alphabet burst if the pressure is too much?*’

The setup created an illusion of direct cause and effect, making the interaction feel natural and responsive. This highlighted how intuitive interactions, paired with familiar physical gestures, can create a strong sense of immersion and perceived physicality in phygital experiences.

d. Animating flipdots

A flipdot[43] display is an electromagnetic dot matrix display technology. It consists of small circular discs arranged in a grid, where each disc has two sides, typically black and white. The direction of the current determines which side of the disc is visible, allowing it to flip between black and white states. This mechanism enables the display of text, images, and animations. Traditionally, flipdots have been widely used in contexts such as scoreboards at sports events, railway station information boards, and storefront signage.

The first time I saw the PCBs on the back of the flipdots I was truly intimidated. I thought I would never be able to understand the mechanism, let alone work with it. Fortunately, the groundwork for setting up the system to control the flipdots via Processing had already been done by Owen McAteer [44]. My role was to build upon his existing work and take it forward from there.

Sketch 1:

The first sketch done on the 28 X 14-pixel flipdots was a typographic poster that read 'AMBO' (*Ambo* is a Nepali phrase that loosely translates to 'Ohno!'. In this situation, it truly reflected my distress in working with an alien system.) The poster was created by drawing only rectangles in very specific locations. It had a slight interactive element, as it allowed the poster's colours to invert with a mouse click. This sketch truly set a tone for how I would work with the flipdots in the future. Watch Video ↗

Figure 41: AMBO!

43. Flipdots, https://flipdots.com/en/home/.

44. Owen McAteer, *FlipDots*, 2023, GitHub, https://github.com/owenmcateer/FlipDots.

Sketch 2: (36 days of Type)

36 days of Type is a yearly call that invites designers, illustrators and visual artists to design a letter or a number of the Latin alphabet each day for 36 consecutive days.[45] It offers a space for artists to explore the graphic possibilities of different letterforms. I decided to use the 28 x 14 flipdot display with its pixel grid limitations to initiate this project.

While I continued using my previous method of drawing with rectangles, I quickly realized that it was not design-friendly. I spent more time counting dots on the grid than actually designing. To streamline this process and allow for freer experimentation, I developed an HTML, CSS and JavaScript web sketch that mimicked the 28x14 flipdot display.

This tool allowed me to freely draw on the grid by toggling between white and black dots with a simple click. I also added an invert colour button to quickly reverse colours without manually clicking on each dot. A key feature of this sketch was the console's ability to generate the code for those specific rectangles, which I could then copy and paste directly into the Processing sketch. This made the design process far more efficient and intuitive.
Web Sketch ↗

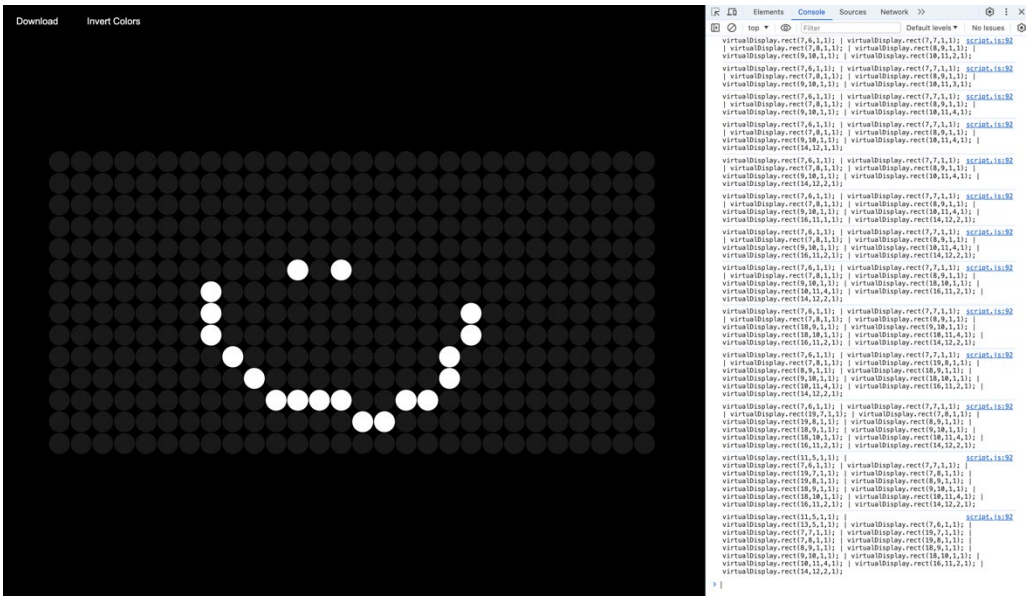


Figure 42: Web app and its console

45. Nina Sans and Rafa Goicoechea, *36 Days of Type*, https://www.36daysoftype.com/.

I used this process to make the 36 Days of Type. Each letter I designed included 4 variations, which played sequentially as animations. From my earlier prototype, I realized that creating 21 variations for a single letter was a terrible idea. Doing this would mean designing 26 x 21 = 546 frames in total, which I am yet to do for the inflatable font. Following are the letters I was able to design:

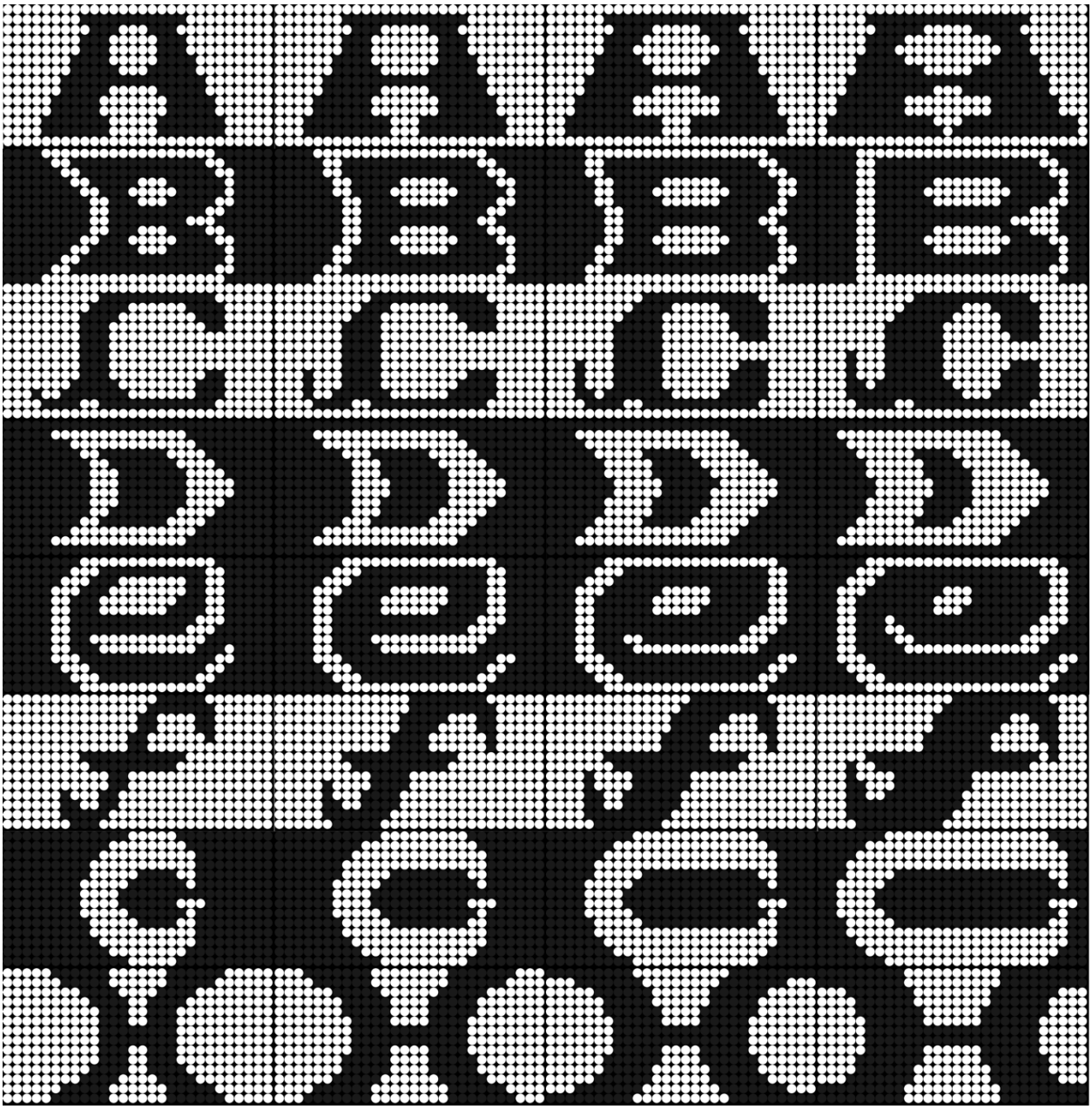


Figure 43: Alphabets A-H with 4 variations each

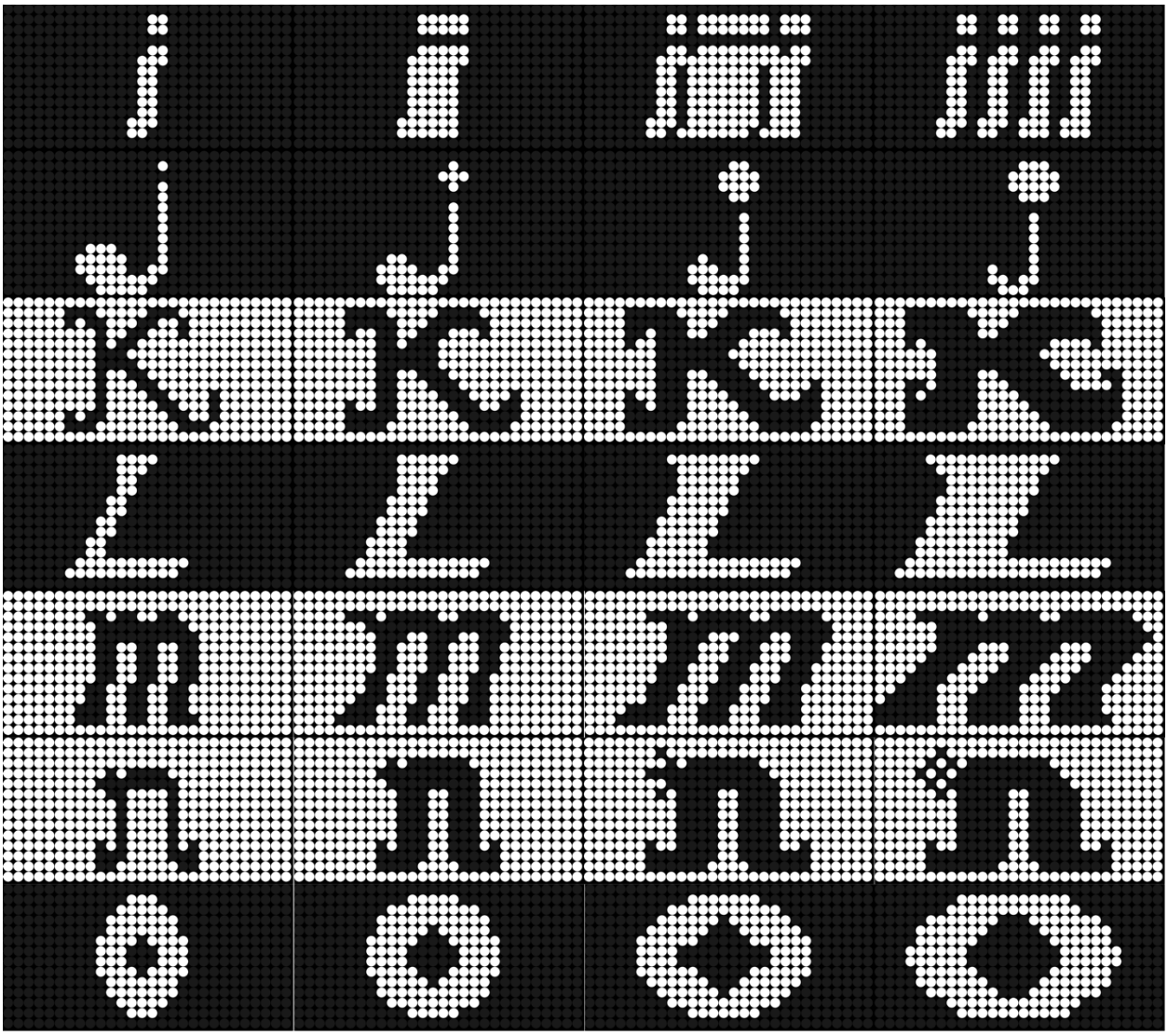


Figure 44: Alphabets I-O with 4 variations each

(Note: Alphabets P-Z are still to be done)

Animations:

The running dog animation from my very first prototype (*I used Excel to make animations*) was repurposed for the flip-dot display since it shared a similar pixel-based graphic format. The individual frames from a total of 9 frames were mapped on the web sketch I created, that provided the corresponding rectangle coordinates which were pasted into the Processing sketch one by one.

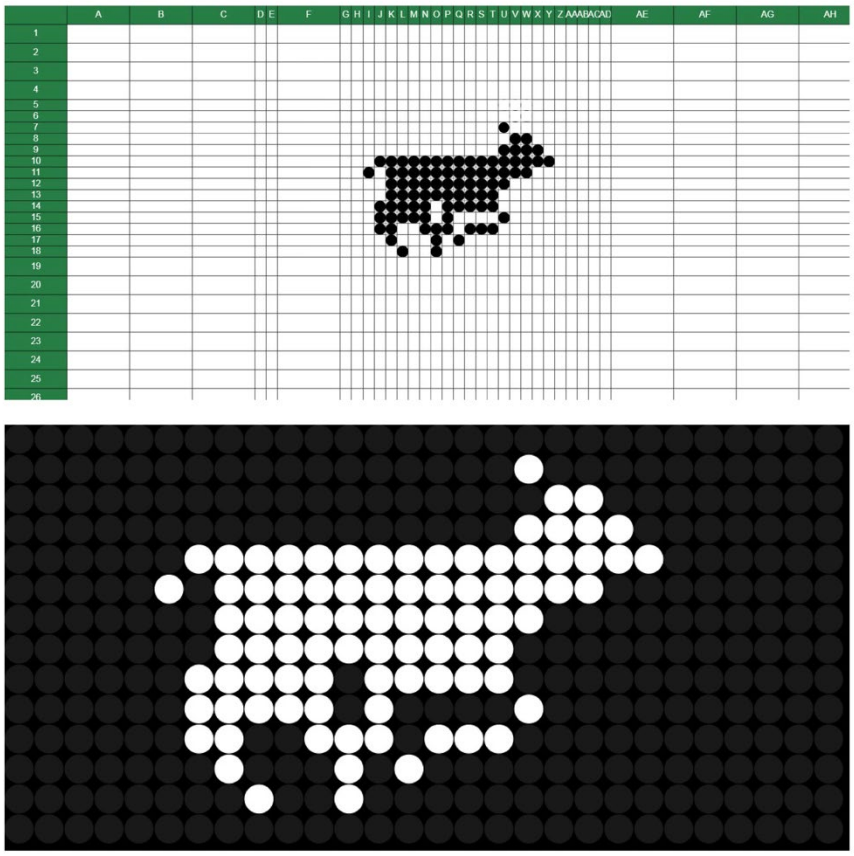


Figure 45: Old dog, new dog

The sketch looked beautiful as the dog ran indefinitely on the canvas. It was a stimulating sensory experience to watch the flipdots turn and make sounds. However, the user was still a passive observer. To invite the user to be an active part of this experience and make it more tangible, I added a DIY pressure sensor to the sketch. It was programmed to control the framerate of the animation. The harder one pressed, the faster the dog ran.

The pressure sensor was built using layers of aluminum foil, velostat - a material that increases conductivity when pressure is applied, and a final layer of conductive fabric. This entire setup was wrapped in a soft, felted material to provide a comfortable grip. Felt also offers slight cushioning, which enhances the feedback when pressing the sensor, making the interaction feel more intuitive

and responsive. Additionally, it also absorbs minor pressure variations, allowing for a more gradual and controlled input rather than abrupt, binary responses.

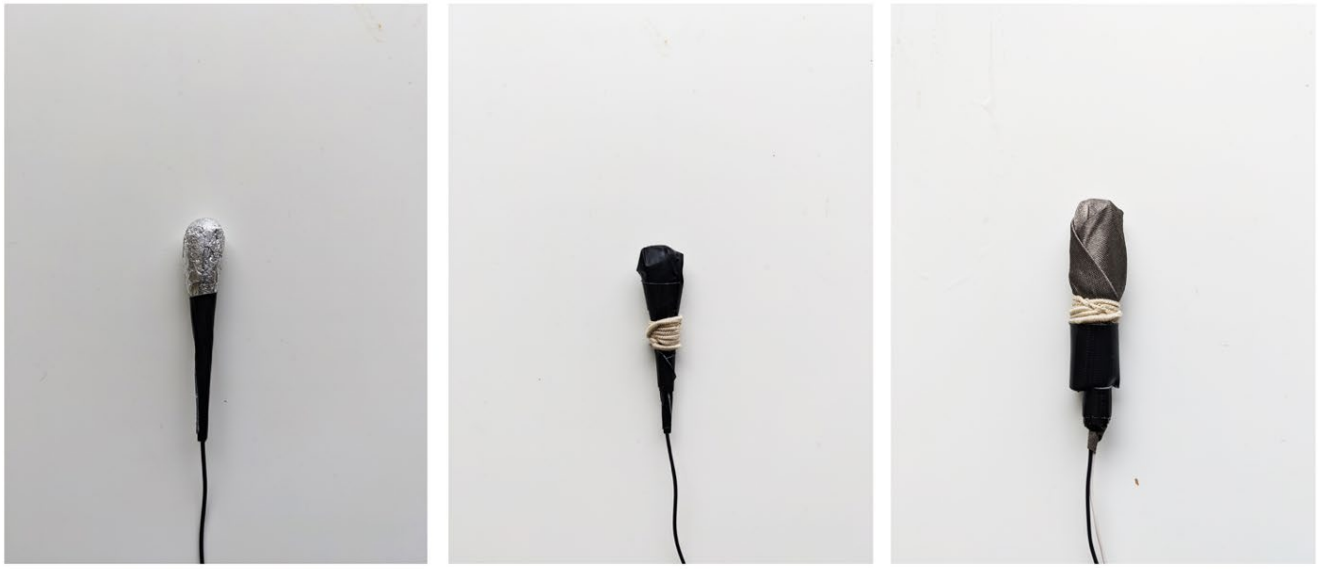


Figure 46: Layering of the Pressure Sensor - (L to R) Aluminium Foil, Velostat and Conductive fabric

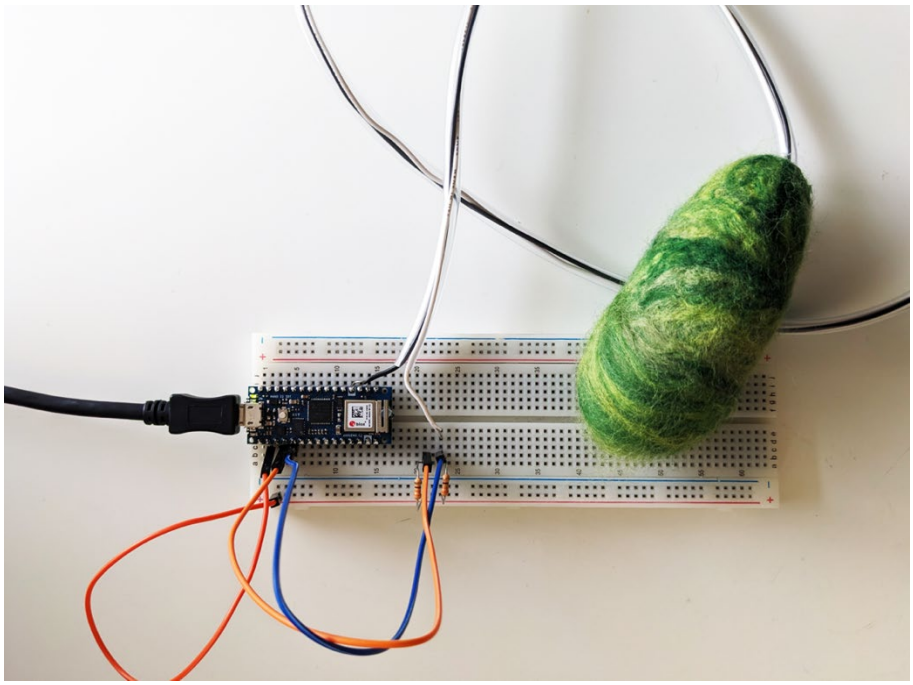


Figure 47: DIY pressure sensor coated with felt

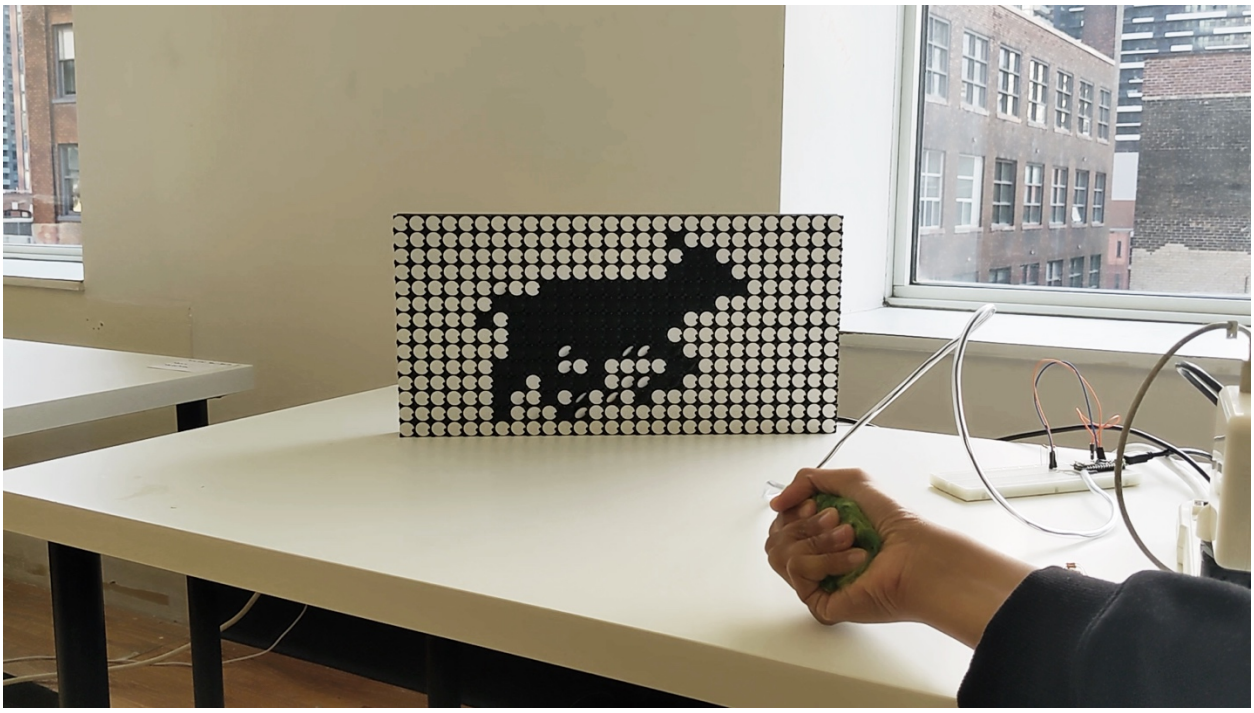


Figure 48: The Setup

Watch Video ↗

The animation sparked a lot of interest. Many people had questions, and several wanted to create animations for the flipdot display. However, I knew that animators and designers typically prefer working in familiar software like Photoshop, Procreate, or After Effects and no one would actually end up using the Web link I made. To make this easier, I created a template that allowed them to simply paste this on their designs as a foreground layer.

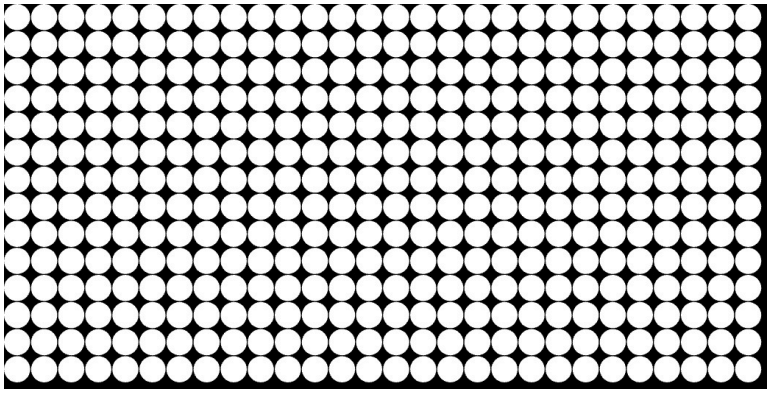


Figure 49: The template

The first submission I received was a 7-second animation by Vaibhav Nanchahal, which consisted of 71 frames. At the time, I was still inputting data manually, which made handling multiple frames impractical. This led me to develop a pipeline that streamlined the process a bit more, allowing me to input and process animations efficiently without manually entering all 71 frames.

GitHub Link ↗

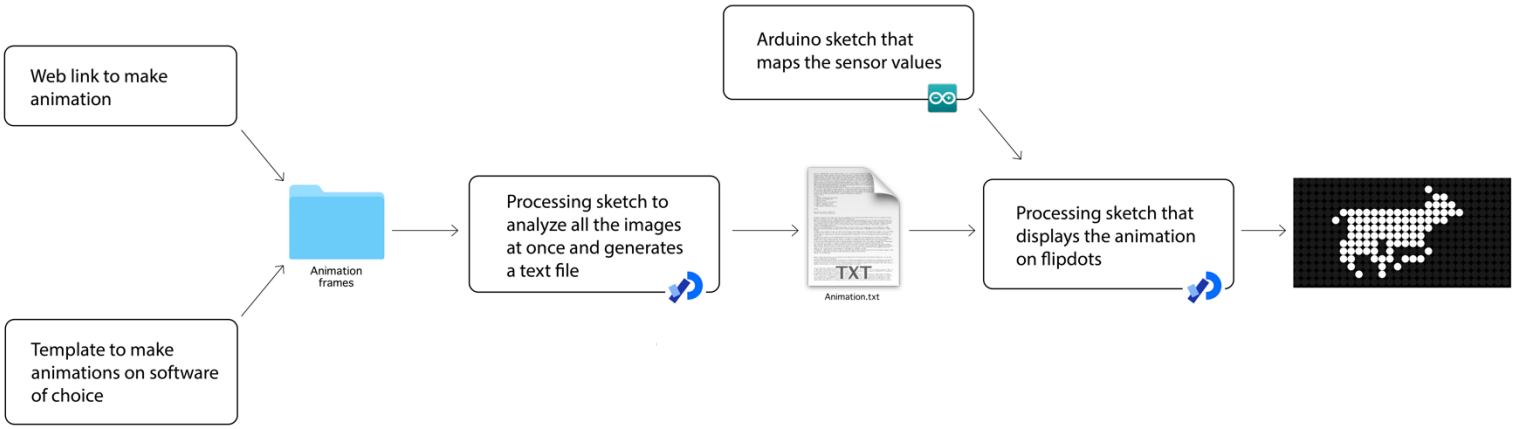


Figure 50: Flipdots animation pipeline

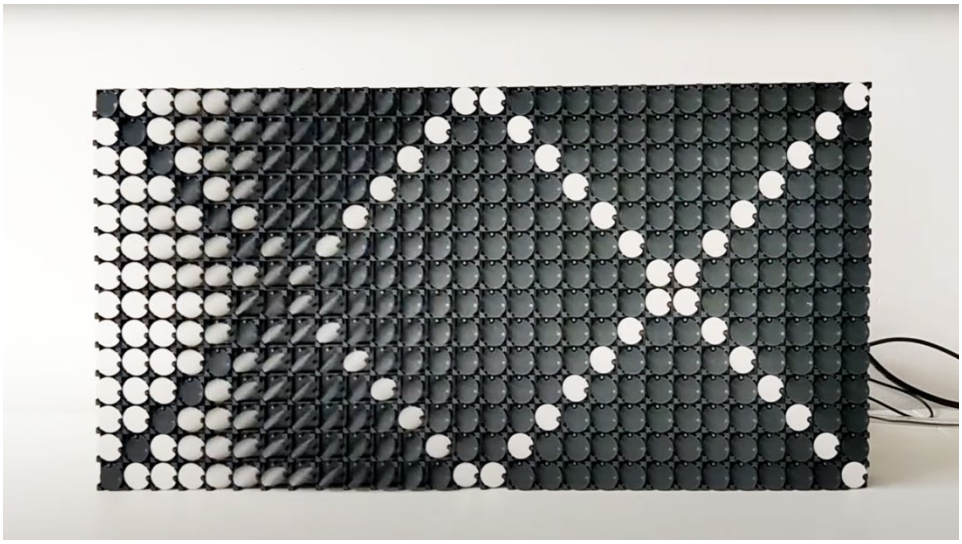


Figure 51: A frame from Vaibhav Nanchahal's 71 frame animation

I documented all the information I gathered about flipdot displays during this process on a dedicated website: https://flipdots.cargo.site/. The website includes resources such as buying links, relevant source codes, details about my projects, and animation templates. Additionally, it features a submission section where users can upload their frame-by-frame animations. Once submitted, I would play their animations on the flip-dot display and share a video of the output with them. This website served as a hub for learning and sharing ideas related to flipdots. It was a step to push this research project from the isolated space it was made into the real world, inviting others to engage with it.

To celebrate collaborative and individual contributions, the website also hosts a gallery section showcasing submitted animations, which is continually updated.

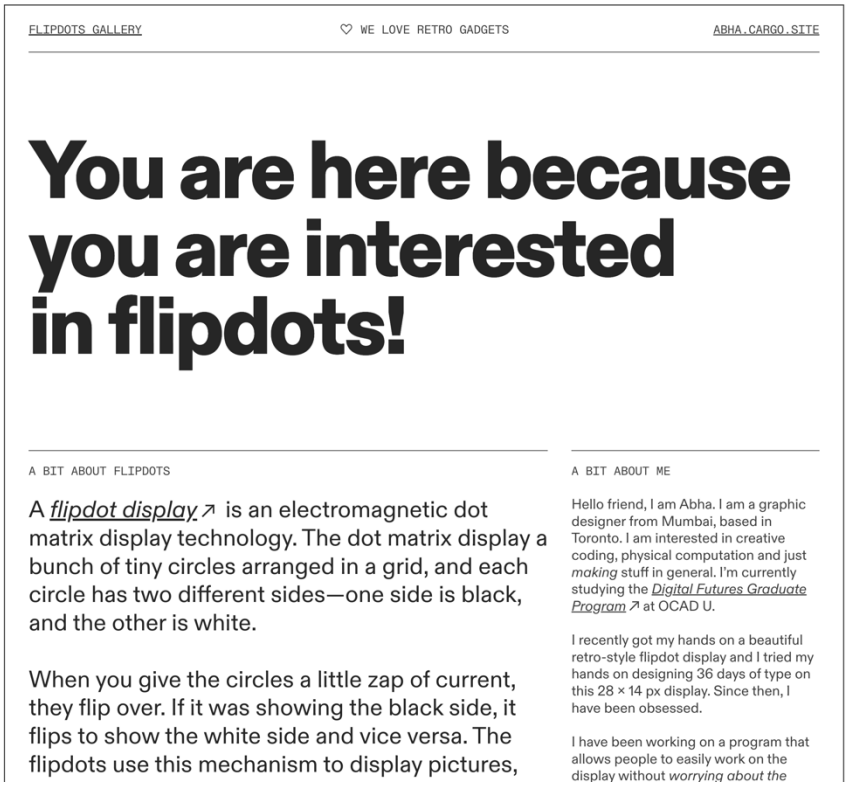


Figure 52: Flipdots Website

Compilation of videos submitted through the website portal ↗

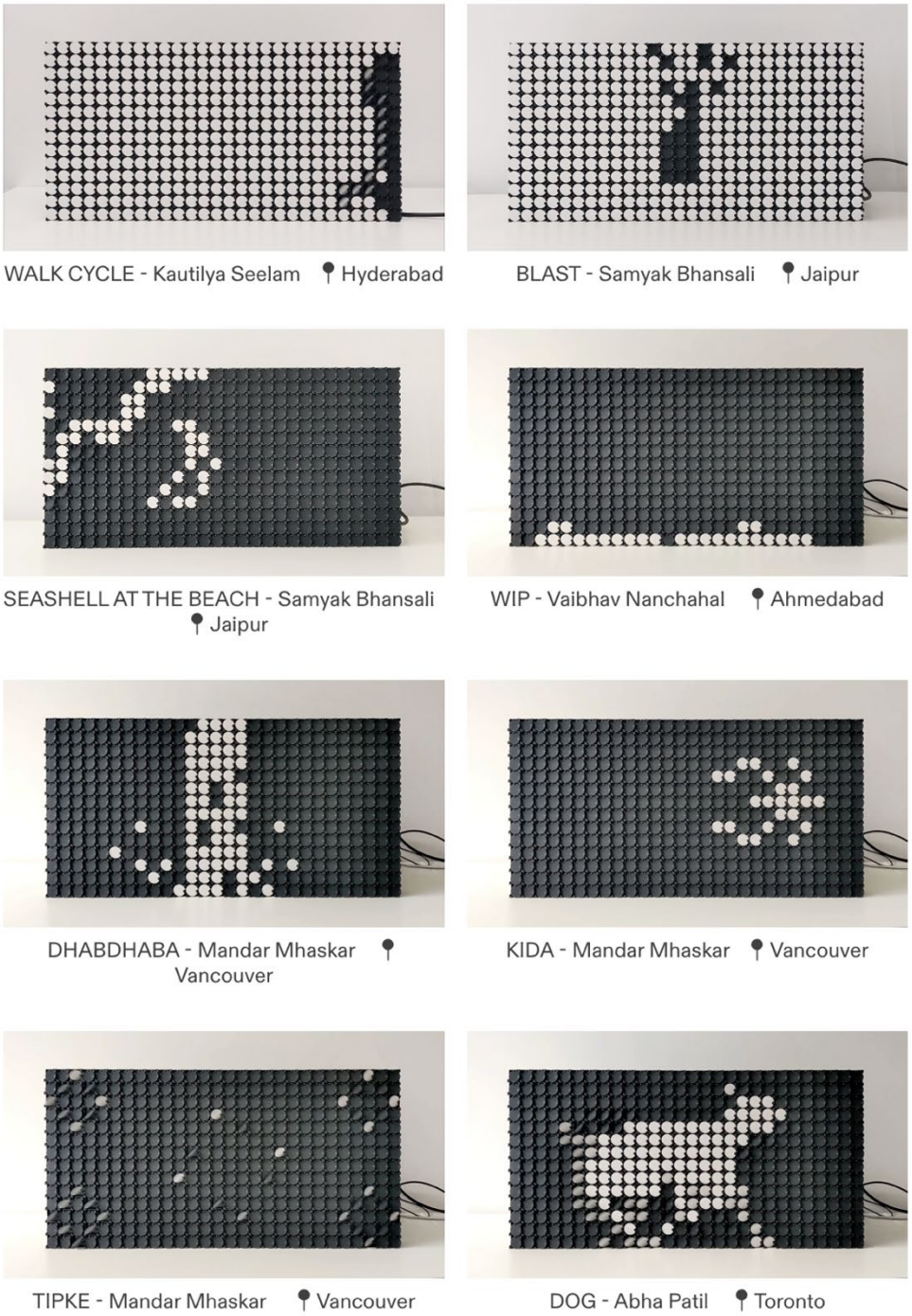


Figure 53: Website Gallery

User testing:

Goal: The user testing aimed to evaluate how participants responded to an unfamiliar setup. It also assessed the ergonomics, usability, and control of the pressure sensor while gathering general feedback and suggestions for improvement.

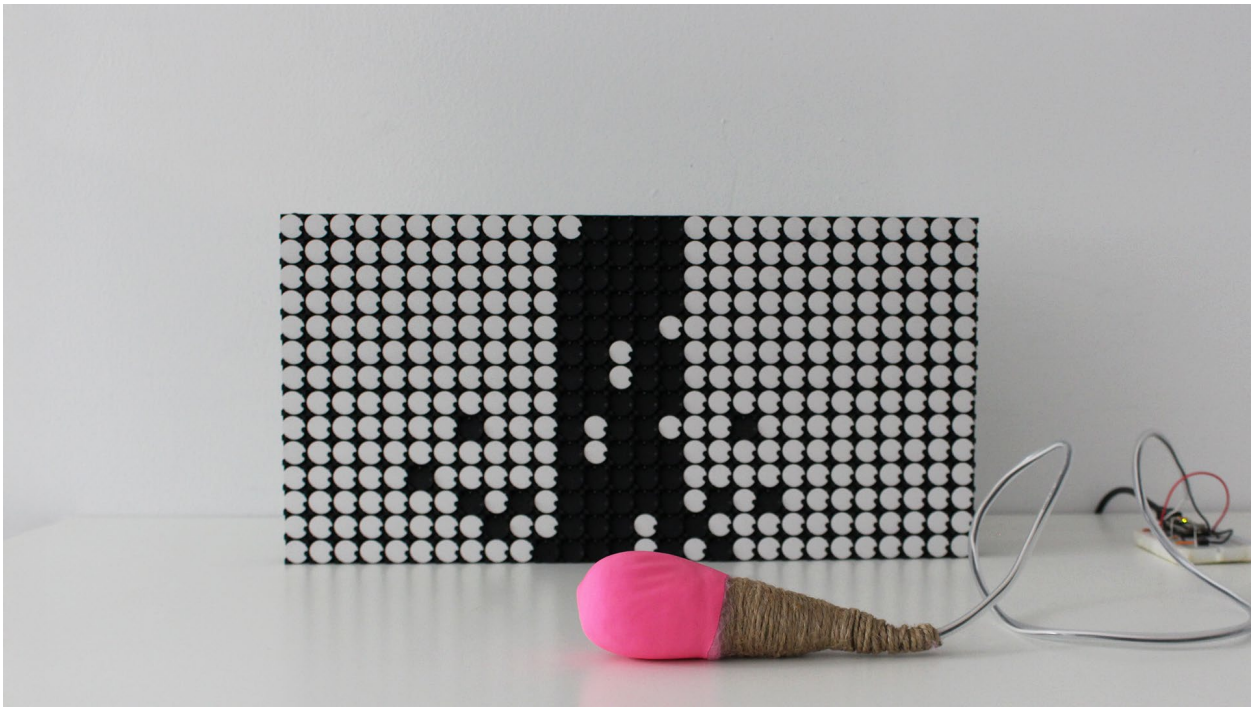
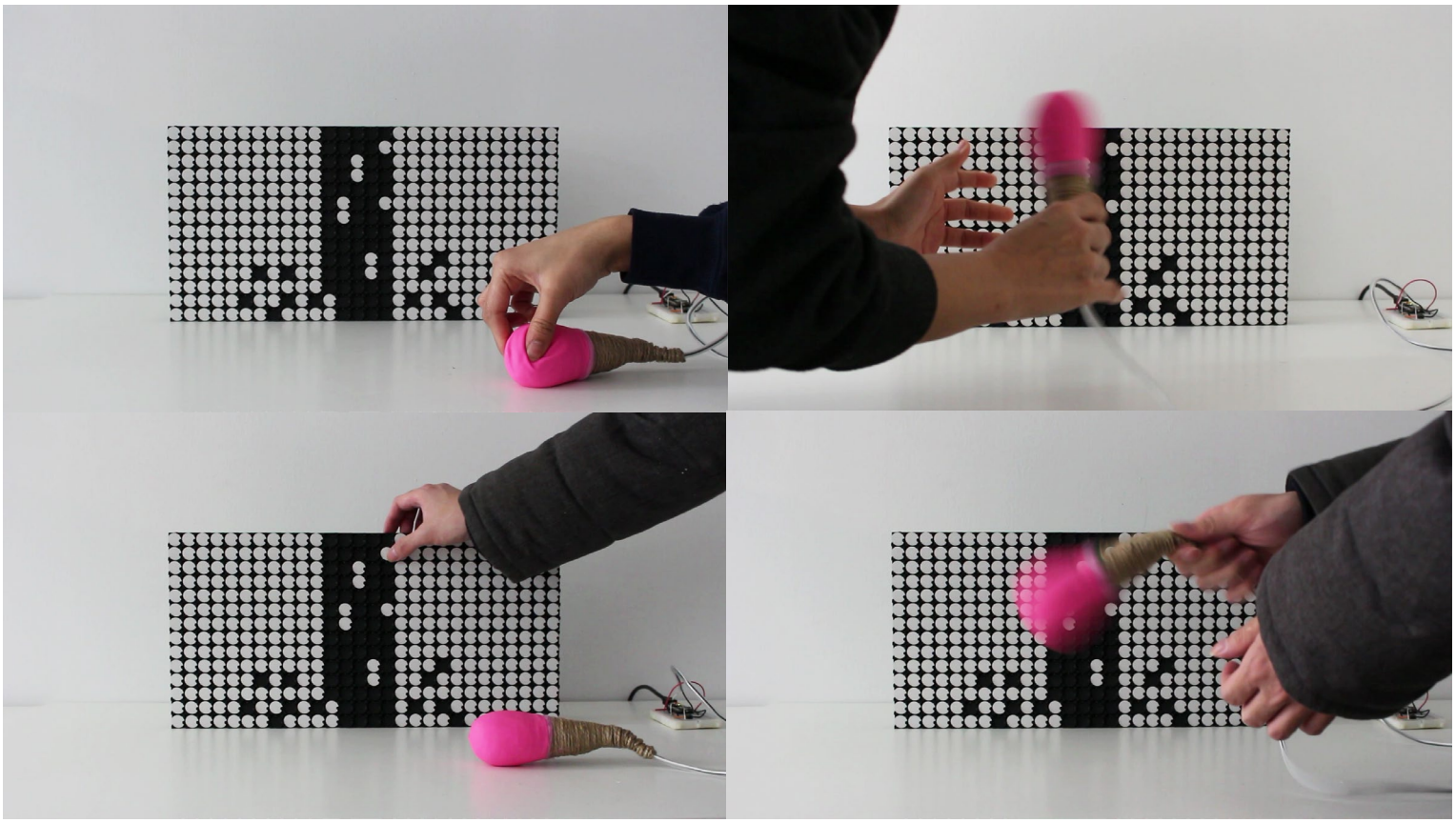


Figure 54: User testing setup

Setup: Participants were greeted with a waterfall and what looked like an ice-cream cone in front of it. They were not given any instructions; instead, they were encouraged to interact with the setup in whatever way that felt intuitive to them. The ice-cream cone-shaped pressure sensor was intentionally designed in hot pink to stand against the binary flipdots background, while its soft and squishy texture served as an invitation to be touched. *(Note: Originally made with felt, the pressure sensor was later covered with latex cut from a balloon, as continuous use caused the felt to appear shabby and deformed. This new covering helped protect it from further damage, while giving it a much cleaner look.)*

Interaction: At first, participants hesitated, observing the interface before interacting with it. The unfamiliar form and lack of instructions led them to test it in different ways, including tapping, shaking, and speaking into it. These responses indicate how people experiment with new objects, using gestures and habits they already know.



(Clockwise from top-left) *Figure 55: Prodding, speaking into, shaking and inspecting the interface*

As participants identified the interaction, they began pressing the dollop in different ways to observe its effect. Pressing it triggered the animation, and a button on the breadboard allowed them to cycle through eight different animations submitted via the website portal.

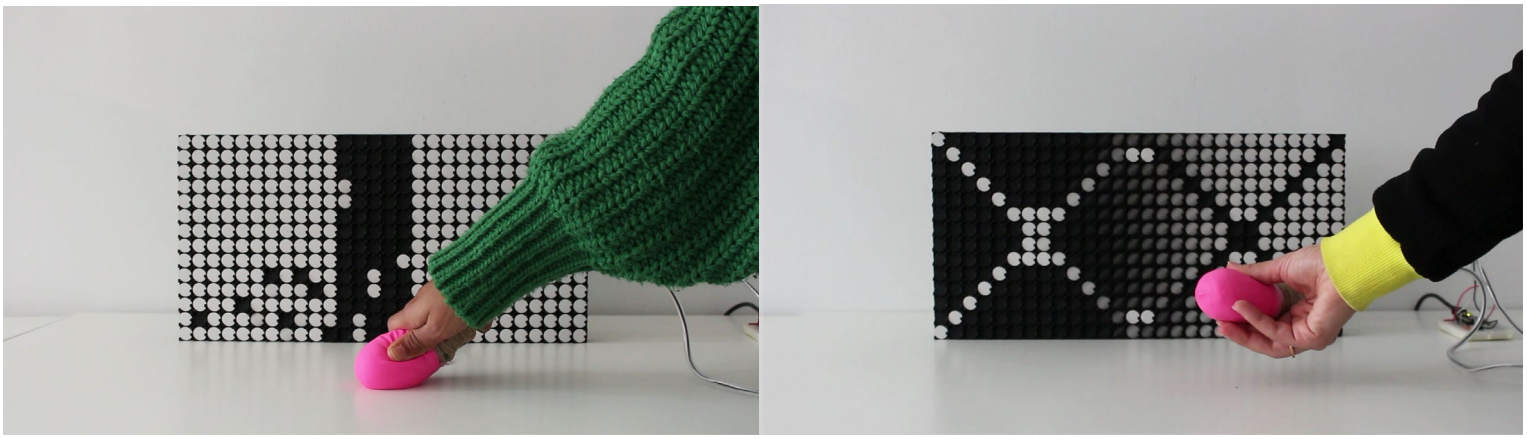
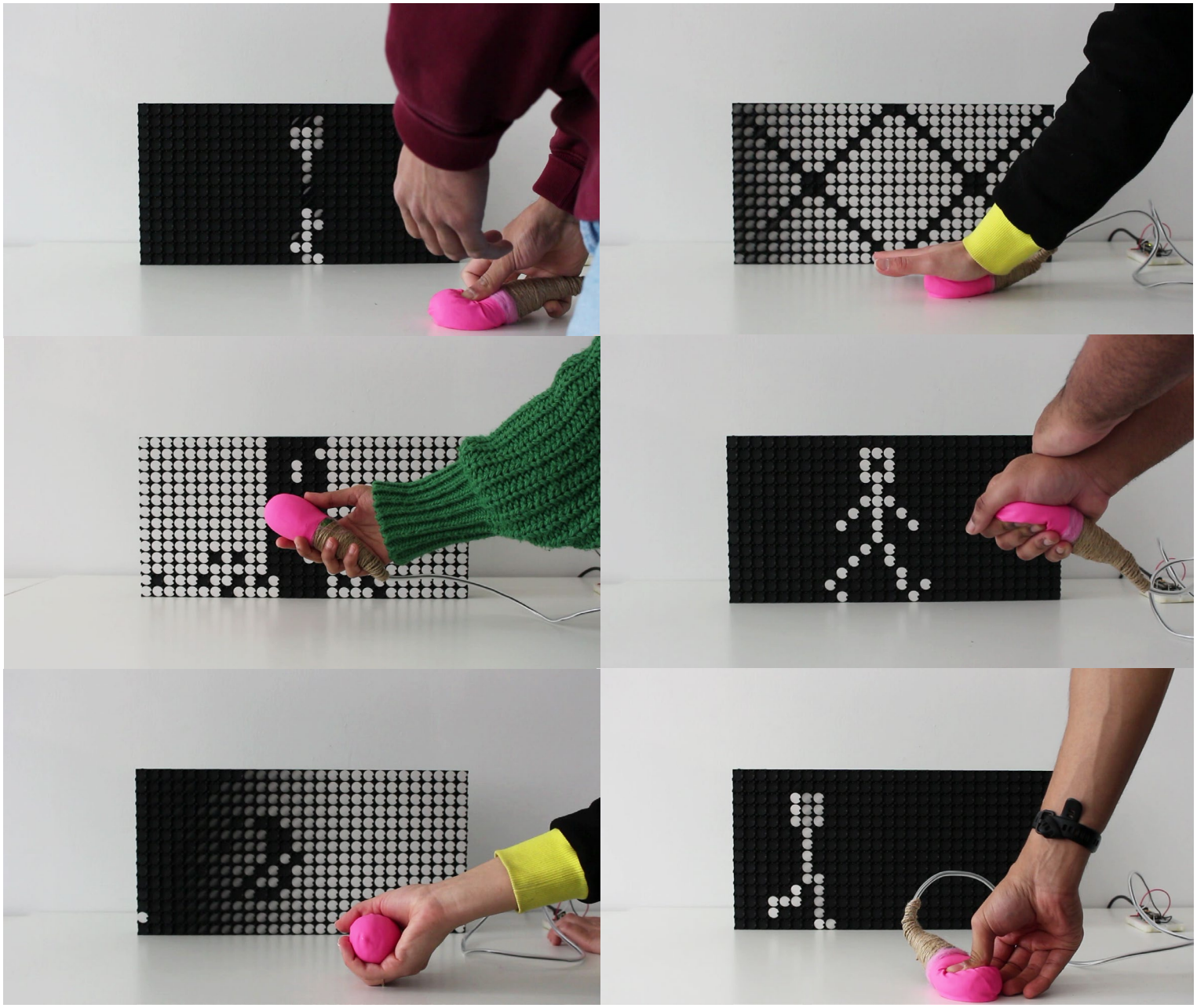


Figure 56: Pressing against the table and squeezing

Participants observed that the speed of the animation varied based on the pressure applied to the dollop. By applying different levels of pressure, they explored the system's response and tested its range of interaction. Some participants tried to control the animation speed by applying gradual pressure changes, testing how smoothly they could manipulate the output. Some pressed as hard as possible to see if there was a maximum speed or a breaking point.



L-R from Top to Bottom

Figure 57: Row 1: Pressing against the table, flattening it on the table; Row 2: Inspecting the sensor, testing the sensor limits by pressing too much; Row 3: Squeezing the sensor while cycling through different animations, Pressing against the table.

Insights: Insights from this testing highlighted areas for refinement, especially in an exhibition setting. The first being the distance of the user from the flipdots. Given the 28px by 14px setup, the animations are best viewed from a distance. However, in this setup, the user's distance was limited by the length and fragility of the pressure sensor wire. As a result, I often noticed participants squinting at the screen. A more rigid setup and longer cable connecting the pressure sensor would provide greater flexibility, allowing users to position themselves at an optimal viewing distance. Similarly, the control for switching animations (currently, a button on the breadboard) could be made visually noticeable. Placing it closer to the pressure sensor would also reduce unnecessary back-and-forth movement.

The pressure sensor itself was a site of exploration. There were spots in the sensor that did nothing when pressed. Some users quickly learned to adjust by pressing in more responsive areas, while others had to be guided into it. Ideally, the pressure sensitivity throughout the sensor should be uniform. While achieving complete consistency is challenging when working with organic materials and DIY techniques, it is an area that can be optimized further.

e. Typing with a rotary phone

Figure 58: An old Rotary phone

A rotary phone carries a strong sense of nostalgia; almost everyone has seen or used one at some point. For the final interface, I wanted to tap into that familiarity of how the device worked, but in ways that it hadn't. This one sat on my desk for weeks before I even knew what I wanted to do with it. It lingered in my peripheral vision, waiting (almost demanding) to be taken apart.

Then the day came when I had an idea of turning this relic of analog communication into a fully functional keyboard. I took it apart. It was a fully mechanical device that used a system of electric pulses. When a number is dialled, the user rotates the dial to a specific digit, pulling it to a stop before releasing it. As the dial returns to its resting position, it generates a series of clicks or pulses, each corresponding to the number dialled. Unlike other push button phones, the rotary phones require us to wait for the dial to fully reset before entering the next digit, making the entire process inherently slower.

I am drawn to interactions that slow the user down. I believe, this deliberate pace creates a space for more conscious interactions.

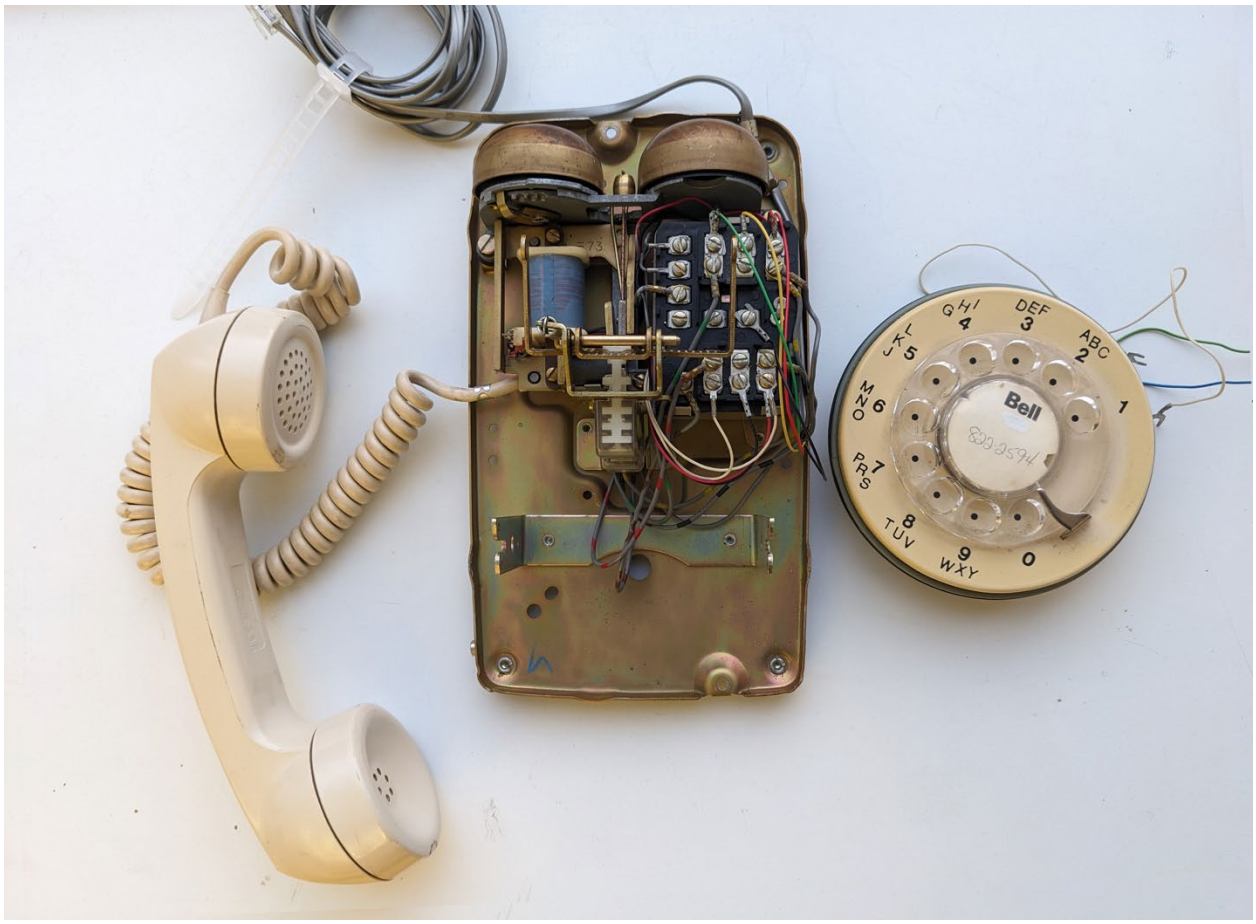


Figure 59: The rotary phone taken apart

Sketch 1:

(Note: This application was built on a single sketch, with iterations focusing primarily on aesthetics, not functionality.)

After identifying the switches that controlled the electric pulses, wires were connected from these switches to an Arduino board.[46] The Arduino then sent signals to a Processing sketch, determining which number was dialled based on the number of pulses received.

46. "Interface a Rotary Phone Dial to an Arduino," *Instructables*, accessed January 11, 2025, https://www.instructables.com/Interface-a-rotary-phone-dial-to-an-Arduino/.

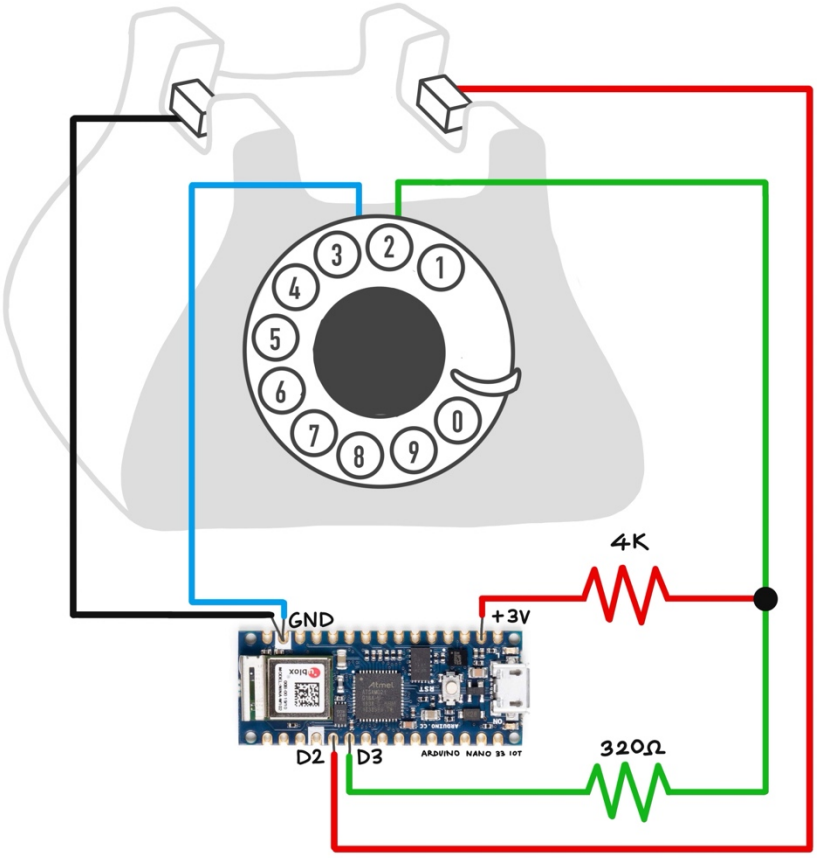


Figure 60: Circuit diagram

Each group of letters was mapped to the corresponding number; ABC was assigned to 2, DEF to 3, and so on. To accommodate essential functions, 1 was designated as the backspace and 0 as the spacebar. This mirrored the experience of texting on an old button phone, where users had to press a key multiple times to cycle through letters.

Figure 61: The rotary phone dial

To cycle through letters, the system needed to distinguish between the user typing an ‘AA’ and just a ‘B.’ To do this, a brief waiting period was introduced. If the user dialled the same letter again within a given timeframe, it would cycle to the next variation. This made timing a crucial part of the interface. Since different numbers take varying amounts of time to dial based on their placement, a timeout map was introduced to adjust the system’s waiting period accordingly.

```
// Timeout mapping for each digit (milliseconds)
int[] timeoutMap = {
  10, // 0: Space
  10,  // 1: Backspace
  1000,  // 2
  1200,  // 3
  1600,  // 4
  2000,  // 5
  2400, // 6
  2800, // 7
  3200, // 8
  3600  // 9
};
```

Figure 62: Timeout map

Additionally, a switch was placed where the receiver rests. The system was programmed so that the keyboard would only function when the receiver was lifted, reinforcing the familiar interaction of having to "pick up" the phone before use.

GitHub Link ↗

Aesthetic choices: The deliberate slowness of the interface played into the idea of conscious communication. With communication being so effortless today, words don’t always carry the same weight they once did. I wanted this interface to encourage more intentional communication.

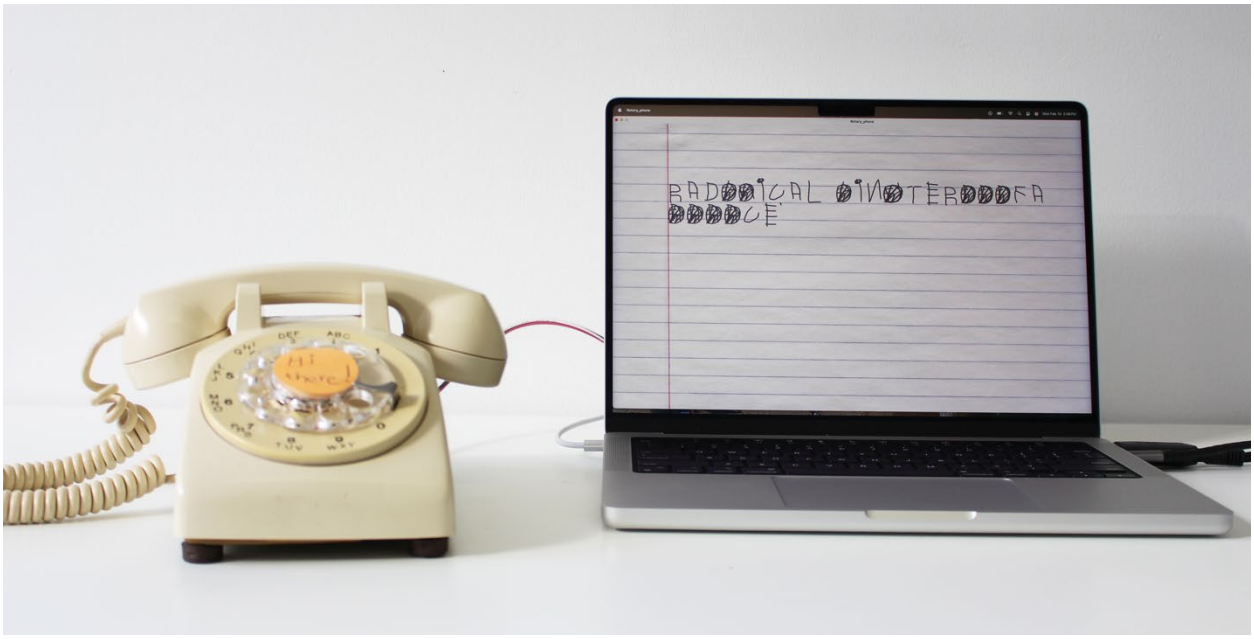


Figure 63: Spelling Radical interface on a Radical interface

A paper-pencil-like aesthetic was created, paired with an experimental font called PIA[47], designed by Andrew Bellamy using scans of his three-year-old's handwriting. The backspace button was transformed into a scribble, mimicking the experience of writing on paper without having the option to erase.

Technical choices: This interface deliberately avoids using the T9 predictive text algorithm[48], which was originally designed to speed up text entry on mobile keypads by predicting words based on a few key presses. This was done to keep any automation features away from this interface, making it a fully manual and intentional experience.

47. Otherwhere Collective, *PIA*, designed by Andrew Bellamy (2024), https://otherwherecollective.com/pia/.

48. Wikipedia contributors, *"T9 (Predictive Text),"* Wikipedia, The Free Encyclopedia, https://en.wikipedia.org/wiki/T9_(predictive_text).

User Testing:

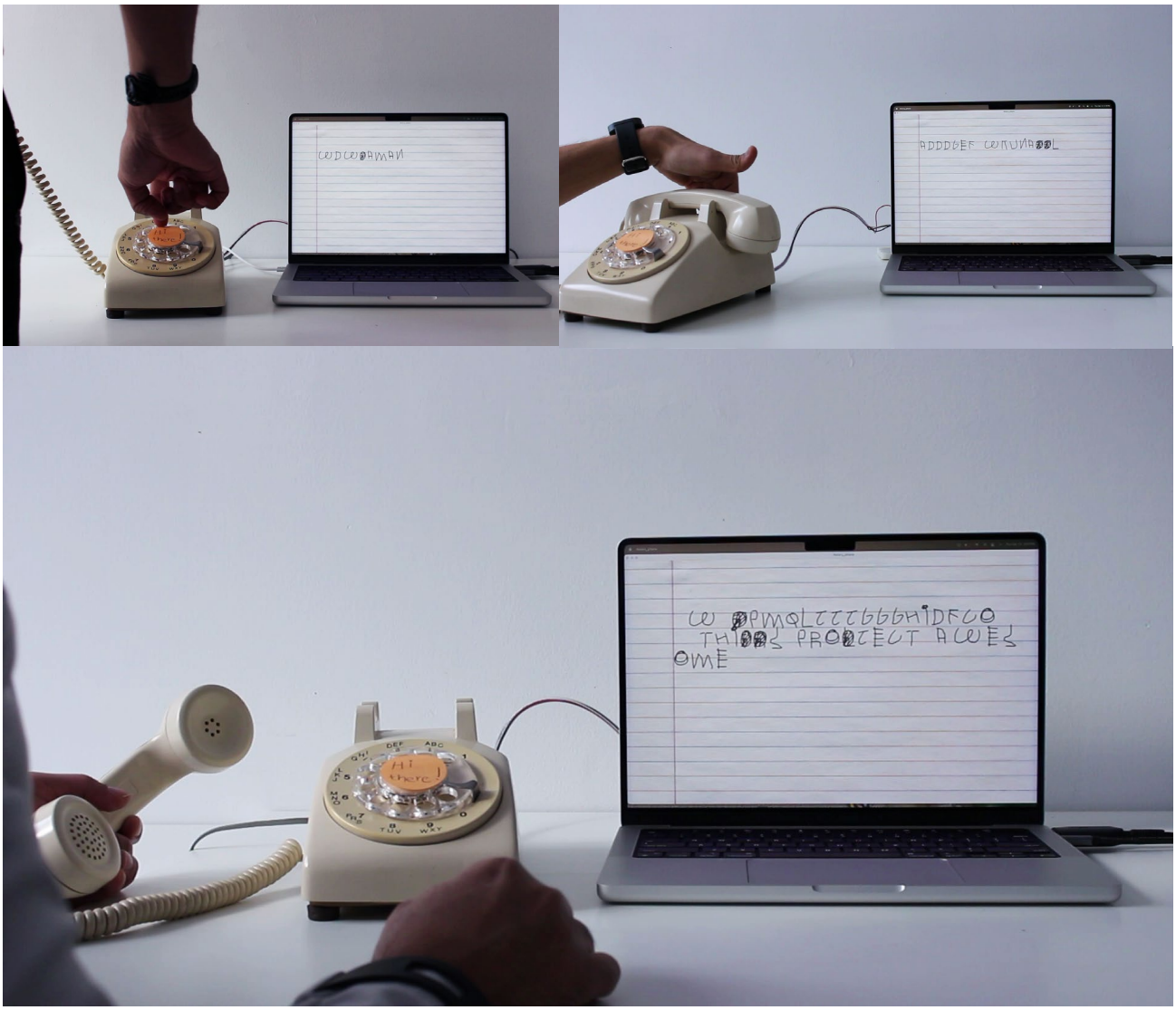
Goal: The goal of this user testing was to evaluate the timeout map, which had so far only been tested by me. This phase aimed to see how well it worked for other users and identify any necessary adjustments. Another goal was to see if the deliberate slowness influenced how participants used this interface.

Setup: The setup consisted of a laptop screen that displayed a blank page of a notebook and a rotary phone next to it.

Figure 64: User testing setup

Interaction: Interaction was intuitive for the older participants. They immediately started dialling, though most forgot to pick up the receiver at first. When they realized it, they laughed at how silly the interaction was, as picking up the phone served no real function beyond reinforcing an old habit. The younger participants recognized the device but were unsure how to use it correctly. I eventually had to guide them through the interaction.

The timeout map only worked effectively after a few attempts, once participants became comfortable with the interface and were determined to try again.



(Clockwise from top-left)

Figure 65: Participant spelling their name; Participant keeping the receiver; Participant trying to write "This project awesome"

Insights: Most users were able to intuitively figure out how the rotary phone interface worked, though a few challenges emerged. The slight delay in displaying letters on the screen caused confusion, as some users were unsure if the interface was functioning. Participants suggested that providing subtle feedback, such as a beep in the receiver or a visual cue on the screen, could help clarify this confusion.

Additionally, users, being accustomed to computer-based text input, expected a more familiar experience. The absence of a visible cursor left them uncertain about where the next letter would appear, and the

lack of an option to move to the next line felt restrictive. A helpful suggestion was to enable a double spacebar press [00] to shift to a new line.

Most participants either wrote their names or attempted to converse with me through the interface. This was particularly interesting, as the phone is inherently a tool for conversation, and having someone to converse with seemed to be an essential part of the experience. Their interactions highlighted the social nature of the medium, reinforcing that a phone, even in its unconventional form, still encouraged a dialogue. Interestingly, some participants anticipated that their typed messages would automatically erase once they placed the phone back on the receiver, interpreting this as the natural end of communication.

Observing these responses highlights the importance of intuitive design and how physicality continues to inform our digital interactions. Moving forward, these insights will guide further refinements, exploring ways to better align user expectations with interface behaviour. The goal remains to create an experience that feels both experimental and intuitive, encouraging users to engage, explore, and rethink familiar tools in new ways.

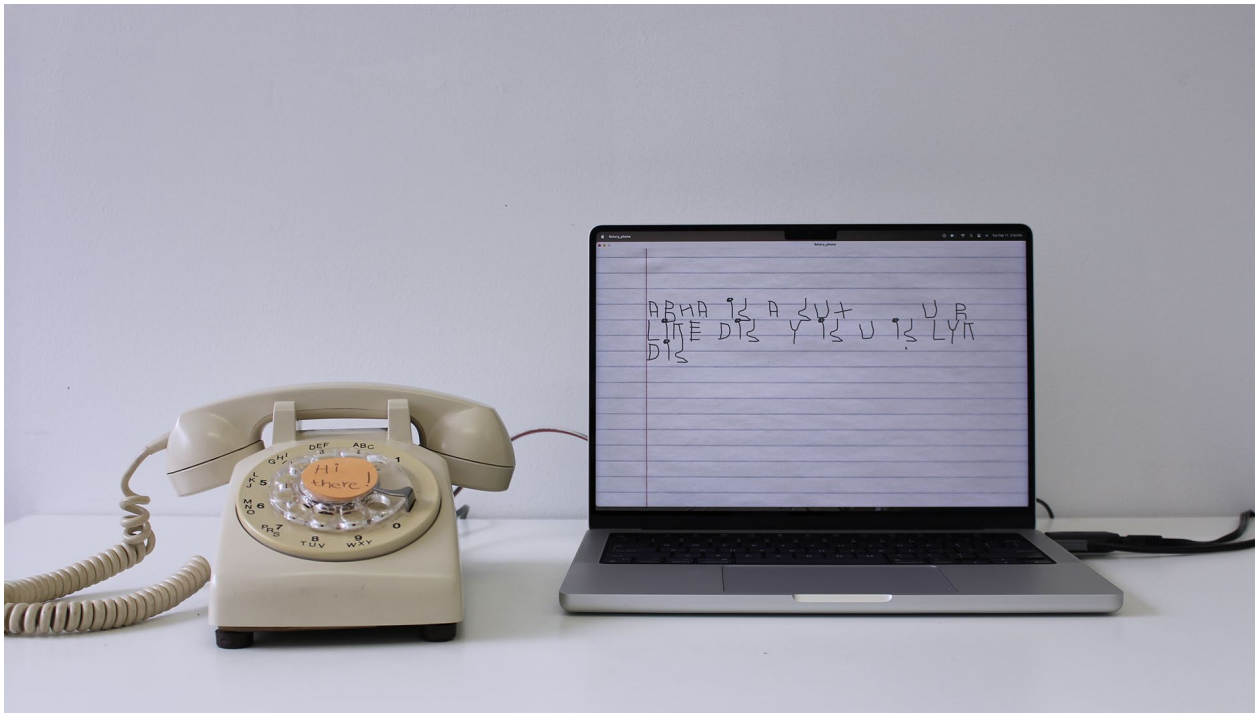


Figure 66: Participant trying to communicate with me

Figure 67: A conversation about lunch on the rotary phone

Reflections and Future Work

Figure 68: Me and my hats

Throughout the creation of Radical interfaces, I found myself constantly switching roles, moving between a *maker*, a user, an observer, a designer and more importantly, a learner. Each of these roles shaped my understanding of what it means to create and interact with technology.

As a *maker*, I learnt new tools, materials, technologies and crafts. This was only possible because of the countless tutorials, instruction manuals, and open-source resources shared by others over the years. I pieced together knowledge from these sources to build what I did. In many ways, Radical interfaces is a patchwork of ideas, assembled from these contributions. The biggest takeaway while wearing the *maker* hat was to stay curious, share what you make, and honour those who came before you and ones who will come after.

The user hat was worn by me on different occasions; one while using the tools I made and the other while using existing tools to make my tools. Consciously or unconsciously, the tools I used informed the way I made tools. A classic example of this is 125, a 3D app I built that wasn't inherently 3D. Instead of traditional volumetric modelling, it functioned as a flat graphical interface, influenced heavily by my extensive usage of Adobe Illustrator. Without realizing it, I had translated a 2D design mindset into a 3D environment, treating space as a layered composition rather than a true spatial construct. This interesting play between the user and the maker raised an important question: *while I was actively challenging traditional frameworks in the industry, was I ever truly able to escape them?*

Wearing the observer's hat was perhaps the most humbling experience of all. When you ideate, build, and use an interface yourself, it feels like a breakthrough. You know every detail, every function, and every hidden quirk of how it is supposed to work. You expect the same amount of understanding and

enthusiasm from others. But more often than not, that's not the case. You will notice people struggle with the interface you thought was pretty intuitive and build their own understanding around it. For example, when the rotary phone interface was tested, I noticed a participant keep the receiver every time they wrote a letter as a way to 'lock it in' somehow. That was never the intended interaction, but to them, it made perfect sense. It taught me that interfaces may not exist in their intended form but are always shaped by people's understanding of them. Opening an interface up to interpretation and reinterpretation gives it room to grow and evolve.

Throughout the process, the designer's hat was a constant. There was always a delicate push and pull between the designer and the *maker* roles. The designer wanted clarity; to define the goal, refine the function, optimize the details and make it look better, while the *maker* thrived on experimentation. Despite the differences, the two roles informed each other. The designer provided a structure to the chaos of *making,* and *making* was able to loosen up the rigid bits in the design. As I worked with different textiles, like conductive fabric, velostat, and felt, I realized that the materials informed a sizable chunk of the process. They had their own constraints and behaviours that couldn't be tamed. Over time, I was able to develop an understanding with the materials; not by forcing them, but by responding to them. This is where the designer in me found the balance between intention, intuition and improvisation.

So, what makes any of these interfaces *critical*?

All prototypes were conceived with different goals in mind, and they surely changed their courses over time, with lessons from one informing another. The overarching goal, however, was to rethink the traditional interfaces by turning them into playgrounds rather than tools. *I made animations in Excel* disrupts the rigid associations of the tool, turning it into a medium for visual expression. While its design limits what can be created, it remains tweakable. Users can modify the source code to introduce custom shapes, images, and colours, adapting it in new ways for different projects. Similar to branding projects in graphic design work, where designers define the colours, icons, typography, layout, etc., this tool would allow you to set parameters to fit different creative needs within the Excel framework.

125 emerged as a response to Excel when my lack of 3D skills limited what I could do. Having worked mostly in 2D software, I had little understanding of how things function with an added dimension. With that limited knowledge, I built *125*; not fully 3D, but a system that worked within my grasp of dimensionality.

Animating flipdots, a font that inflates, and *typing with a rotary phone* introduce tangibility as a key part of the interaction. The goal was to shift people's perspective from being passive consumers of technology to active participants, engaging with the mechanics behind it.

The flipdots project was documented in a website format, which became a living portal to not only access information but also to make and share work. People were free to engage with the flipdots website (*simply browsing, making something, learning about how it worked, all of them or some of them*) in whatever way

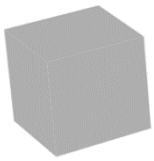
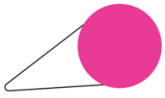
that felt comfortable to them. By lowering the barrier to entry, the experience stayed open and inviting - no pressure, just possibilities.

A font that inflates went through the longest journey of iterations, experiments, and failures. It explored countless technological approaches starting with sound detection, moving to color detection and machine learning, before finally settling on physical computation. While the motivation was to create a variable font, the focus shifted from designing the font itself to developing a pipeline for it. While I was able to develop the pipeline, the actual font remains unrealized due to the constraints of time. This process helped me define what it meant to *make* in graphic design when computation is involved. I see it as a two-part process, first, defining your systems through computation (making pipelines or frameworks), and then using them in your work. While I have completed the first part, I fully intend to finish the rest as time allows.

Typing with a rotary phone was the last interface created. It involved repurposing an old piece of communication technology that has largely fallen out of everyday use, bringing back a sense of nostalgia for a time when communication was slower and more intentional. The final prototype played into this very idea, choreographing the act of picking up the receiver, thinking (*what is it that I want to say/type*), dialing, waiting and receiving the message across on the screen.

At its core, this project was never about arriving at solutions but opening up possibilities. It was about finding ways to break away from prescriptive interfaces by making room for curiosity and play. This gave rise to the Radical interfaces playbook - not as a set of rules but as guiding principles that emerged through the process. And just like the interfaces themselves, the playbook remains open-ended; meant to be challenged, expanded, and rewritten.

Each prototype developed during the course of this project reflects different aspects of the Radical interfaces playbook, showing how the principles shaped both the making and the experience of these interfaces. While multiple principles resonate across different prototypes, the table below highlights the one principle that each prototype most strongly embodies.

Prototype	Playbook principles	Reflection
I made animations on Excel 	Technology is malleable.	A rigid tool built for mathematics and numbers was reimagined as a space for visual expression. This showed that while technology exists with intended uses, it doesn't have to dictate how it is used. We can adapt, repurpose, and create new meanings through creativity and play.
125 	Possibility, not productivity.	The world is wired to make things faster, automate tasks, and prioritize productivity. But building this prototype meant slowing down and deliberately going through a long manual process of crafting every pixel. This process, however long and time-consuming; opened up a new way of thinking about what a 3D application could be. The tool didn't just take time to make, it also asks for time to use. The usage takes an intentional route of trial, error, and discovery, encouraging users to engage with the process of building itself.
A font that inflates 	DIY over default.	This prototype came from a personal frustration: I wanted to create a variable font but couldn't find a system that matched my way of working. So, I built my own tool. It showed me that there is no singular way of doing things; and when something doesn't fit, we can always pave an alternate path to reach the same destination.
Animating flipdots 	Open, not closed.	The flipdots prototype was built on frameworks generously shared online by other people. As I expanded on those foundations and lowered the barrier to entry, it quickly evolved into a shared space, through websites and collaborators. The more people contributed, the more the project thrived, fueled by newer ideas and contributions.

Typing with a rotary phone 	Play is serious work.	Play often looks simple on the surface, but creating it requires patience and deliberate effort. Turning a rotary phone into a keyboard meant taking it apart, learning how pulses worked, and mapping those pulses to letters with exact timing. What looked simple and fun on the surface was the result of careful thought, persistent tinkering, and many cycles of trial and error.

Table 2: Mapping Prototypes to the Playbook principles

These prototypes are invitations to rethink how we interact with technology. Each carrying a spirit of Radical interfaces to stay curious and to make room for new possibilities. And this will continue to be an ongoing experiment, evolving through new ideas, new collaborators, and new provocations. Because, in the end, the *most radical interface* is the one that is yet to be made.

Bibliography:

1. Charny, D. (2012). Power of making. In G. Hertz (Ed.), *Critical Making: Manifestos* (pp. 1-6).

2. Carpenter, J. R. (2015, March 26). A Handmade Web. Lucky Soap.

3. Lialina, O. (n.d.). A vernacular web. Retrieved from https://art.teleportacia.org/observation/vernacular/

4. Dunne, A., & Raby, F. (2013). Speculative everything: Design, fiction, and social dreaming. The MIT Press.

5. Bouroullec R., Bouroullec E. algues (algae) screen. 2004 | MOMA. (n.d.). Retrieved from https://www.moma.org/collection/works/98628

6. Benkler, Y. (2003). *Freedom in the commons: Towards a political economy of information*. Duke Law Journal, 52(6), 1245–1276.

7. Ratto, M. (2011). Critical making. In P. van Abel, L. Evers, R. Klaassen, & B. Troxler (Eds.), *Open design now: Why design cannot remain exclusive* (pp. 202-213). BIS Publishers.

8. White, N. (2012). A Summary of My Work Modes and Objectives. In G. Hertz (Ed.), *Critical Making: Manifestos* (pp. 9-12).

9. Ratto, M., and Boler M. DIY citizenship: Critical Making and Social Media. Cambridge, MA: The MIT Press, 2014.

10. Ruben, P. Caps lock: How capitalism took hold of graphic design, and how to escape from it. Amsterdam: Valiz, 2021.

11. Hertz, G, and Parikka J. "Zombie Media: Circuit Bending Media Archaeology into an Art Method." Leonardo 45, no. 5 (October 2012): 424–30. https://doi.org/10.1162/leon_a_00438.

12. Silver, J (2012). Makey Makey. In G. Hertz (Ed.), *Critical Making: Projects* (pp.5-6).

13. Hertz, G. "Methodologies of Reuse in the Media Arts: Exploring Black Boxes, Tactics and Archaeologies." Plenaries: After Media — Embodiment and Context, Digital Arts and Culture 2009, Arts Computation Engineering, UC Irvine, 2009.

14. Madsen, R. (n.d.). Programming Design Systems. https://programmingdesignsystems.com/introduction/

15. Groten, Anja. *First... Then, Repeat. Workshop scripts in practice*. self-published by Hackers & Designers, www.hackersanddesigners.nl, 2011.

16. Schwult, L. "My Website Is a Shifting House next to a River of Knowledge. What Could Yours Be?" Web log. *The Creative Independent* (blog), May 21, 2018. https://thecreativeindependent.com/essays/laurel-schwulst-my-website-is-a-shifting-house-next-to-a-river-of-knowledge-what-could-yours-be/.

17. Page, S. "Man Ran 700 Miles to Make 'Insanely Impressive' Art on GPS Fitness App." *The Washington Post*, 2024. https://www.washingtonpost.com/lifestyle/2024/12/02/strava-art-run-toronto-mccabe/.

18. Processing Foundation. Accessed December 13, 2024. https://processingfoundation.org/.

19. Schneider, K. “HARRY POTTER Theme but Played on My WASHER & DRYER.” YouTube, June 10, 2020. https://www.youtube.com/watch?v=HmltHR0uB9E.

20. Zimmerman, E. "Play as Research: The Iterative Design Process." July 8, 2003.

21. Diagram Design. “How to Conduct Research Through Design.” *Medium*, July 25, 2022. https://medium.com/diagraam/how-to-conduct-research-through-design-e3a5d19fb79f.

22. Manning, Erin, and Brian Massumi. *Thought in the Act: Passages in the Ecology of Experience*. Minneapolis: University of Minnesota Press, 2014.

23. The Coding Train, “Teachable Machine 1: Image Classification.” YouTube video, November 7, 2019. https://www.youtube.com/watch?v=kwcillcWOg0.

24. Bare Conductive. "Make A Potentiometer With Electric Paint." Weblog post. February 13, 2021. https://www.bareconductive.com/blogs/resources/making-a-potentiometer-with-electric-paint?srsltid=AfmBOooqYZzEqMbTjqF7mpbmvZ5y9xKL8Y9AzkOykPY2hYzAZ1HFSYU3.

25. McAteer, O. *FlipDots*. 2023. GitHub. https://github.com/owenmcateer/FlipDots.

26. Sans, N., and Goicoechea, R. *36 Days of Type*. https://www.36daysoftype.com/.

27. The European Organization for Nuclear Research (CERN). *A Short History of the Web*. Accessed February 14, 2025. https://home.cern/science/computing/birth-web/short-history-web

28. Webflow. *A Look Back at ‘90s Website Design (the Good, the Bad, and the Ugly)*. https://webflow.com/blog/90s-website-design.

29. Cybercultural. "1996: Flash, CSS & Web Design." *Cybercultural*. https://cybercultural.com/p/1996-flash-css-web-design/

30. "[alt.ctrl] Indie Games with Weird Controllers from GDC '22." YouTube video, 17:02. Posted by MechBird, March 16, 2022. https://www.youtube.com/watch?v=IDVHkcUB9jI&t=54s

31. Drozdov, G. *GD with GD.* https://gdwithgd.com/.

32. Gerth, N. *Face Painter.* https://nahuelgerth.de/lab/face-painter.

33. Kedia, R. *Toaster Typewriter.* https://ritikakedia.com/Toaster-typewriter.

34. "Interface a Rotary Phone Dial to an Arduino." *Instructables*. Accessed January 11, 2025. https://www.instructables.com/Interface-a-rotary-phone-dial-to-an-Arduino/.

35. Kobakant, *JoySlippers*, accessed [Mar 7, 2025], https://www.kobakant.at/DIY/?p=567

Appendix

Radical interfaces were tested in different exhibition settings at various stages of their development. Each setting provided an insight into their functionality, usability, and interaction. Following is the list of exhibitions where Radical interfaces was exhibited:

a. **notQuietThere(yet);**

The notQuietThere(yet); as the name suggests was a thesis demo exhibition of our works in progress held on 22nd Oct 2024. I roughly had 4 working prototypes, each at different stages of development. (Note: Most prototypes were worked on simultaneously rather than finishing one and getting on to another, because my productivity thrives on chaos).

While I had made these prototypes individually without thinking about how they would come together, this exhibition provided a great opportunity to figure that out. Since most of my interfaces involved typefaces in some form, I decided to create a Ransom Letter compilation. Participants were invited to spell their names using the presented interfaces. This concept also complemented the DIY aesthetic that my thesis often talks about. By inviting participants to construct their names, the project became less about presenting artifacts and more about enabling interaction, improvisation, and personal meaning-making.

The process also surfaced challenges. Some interfaces, like 125 and flipdots, required more guidance than I anticipated, revealing potential friction in their usability. Additionally, participants were reluctant to blow balloons for the font to inflate, which was in Sketch 4 of the development stage. Watch Video ↗

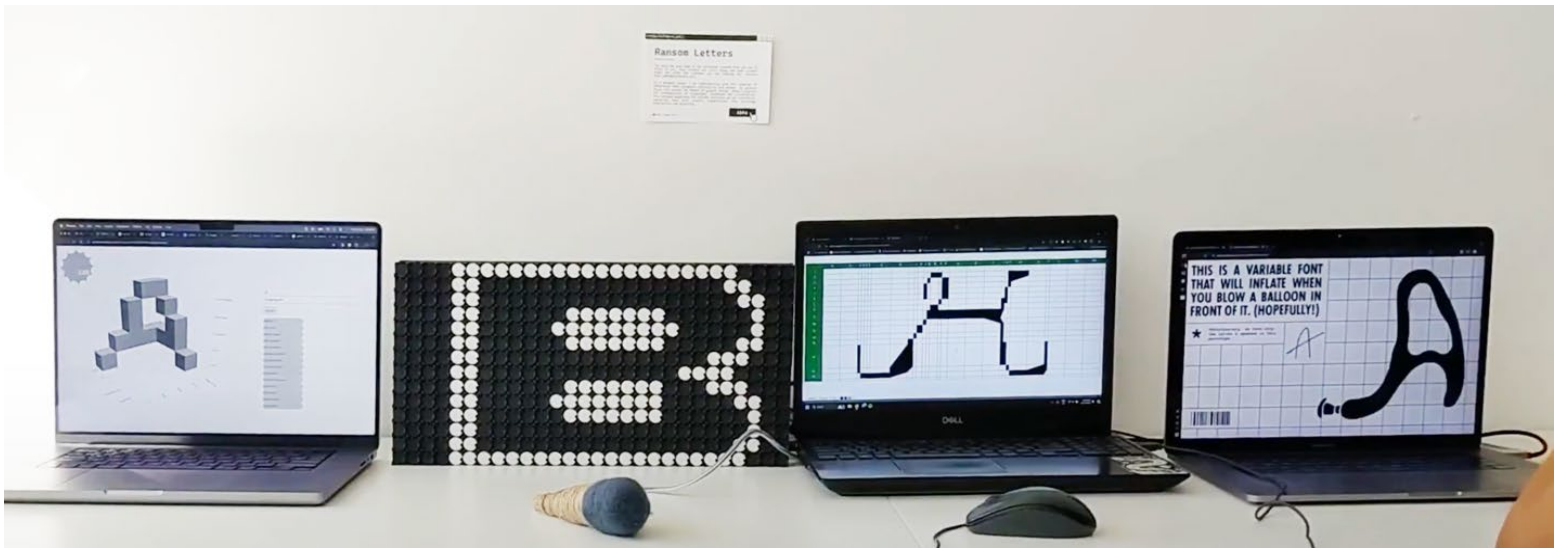


Figure 69: Author name (ABHA) spelt on different interfaces

b. Digital Futures Open Show

For the Digital Futures Open Show, I collaborated with my colleague Juan Sulca to experiment with a new sketch on the flipdots. This time, we used a panel three times larger than before - 42 x 28 pixels. We ran Conway's Game of Life on this panel, a cellular automaton created by British mathematician, John Horton Conway.[49] In this game, cells survive, die, or multiply based on their neighbouring conditions.

Visitors interacted with the display using a clunky ball mouse and a small reference screen to draw their designs. They could either replicate patterns provided on reference cards or create their own. Once finished, participants clicked the play button, triggering the evolution of the cells based on their placement. Some participants attempted to follow the reference cards and were surprised to see the animations evolve indefinitely. However, most used the panel as a drawing canvas, enjoying the process of seeing their designs appear on the flipdots in real time.

Watch Video ↗

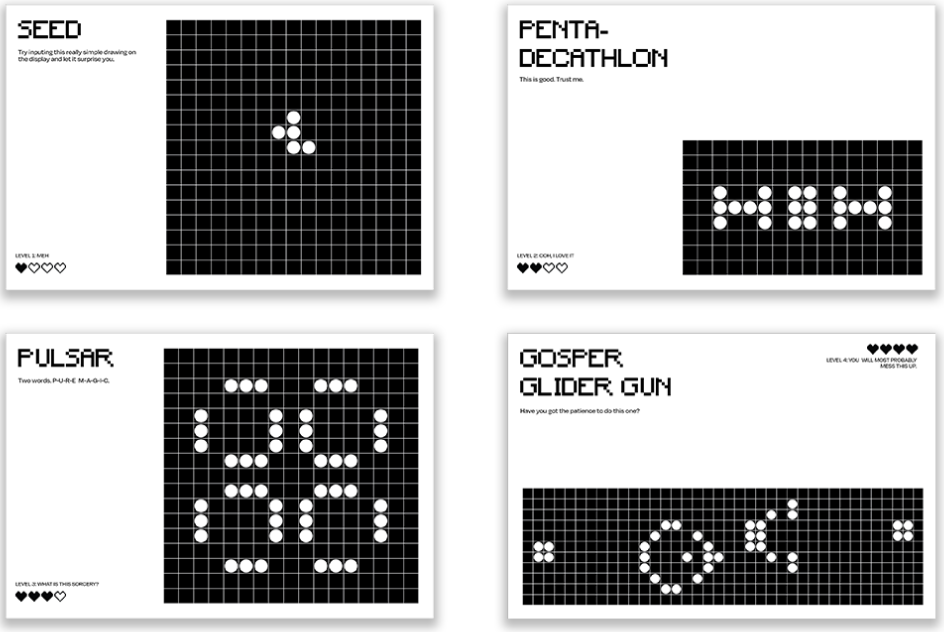


Figure 70: Reference cards

49. Wikipedia contributors, "*Conway's Game of Life,*" *Wikipedia, The Free Encyclopedia*, last modified 11 February 2025, https://en.wikipedia.org/wiki/Conway%27s_Game_of_Life.

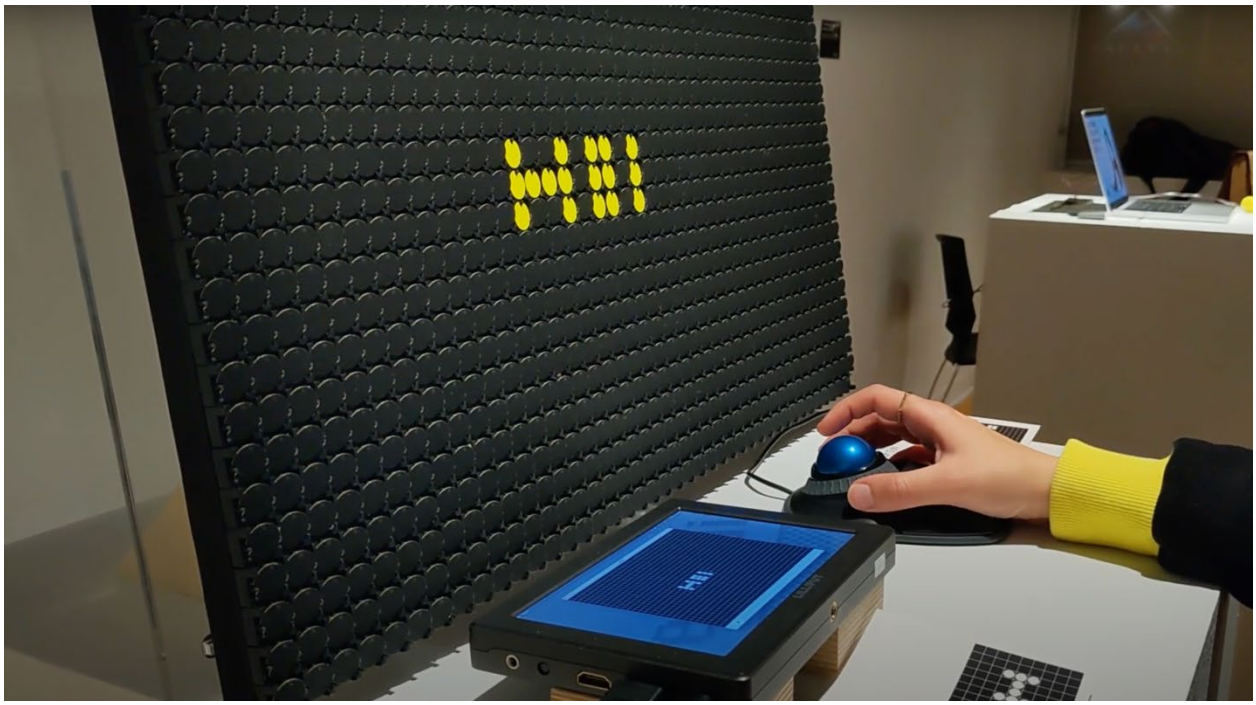


Figure 71: Tiny screen, clunky ball mouse and the flipdots

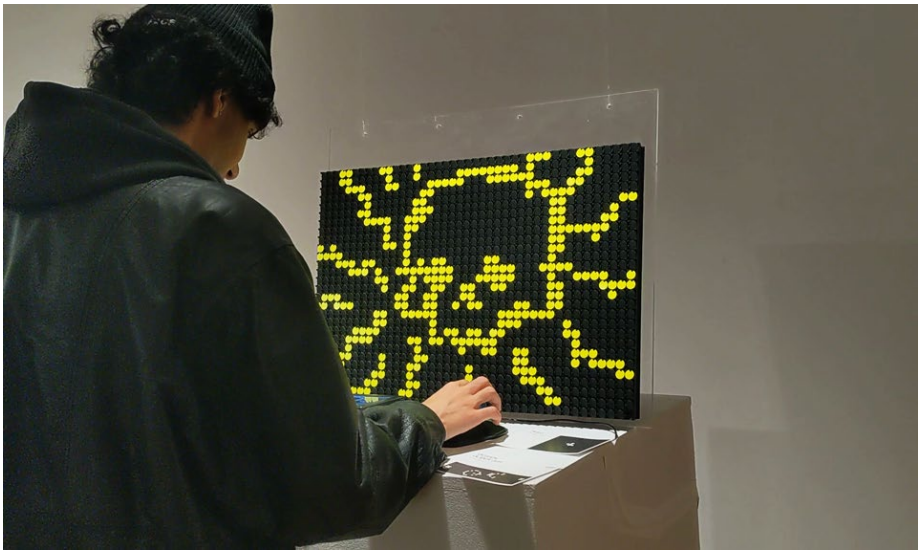


Figure 72: Visitors using the screen as a drawing canvas

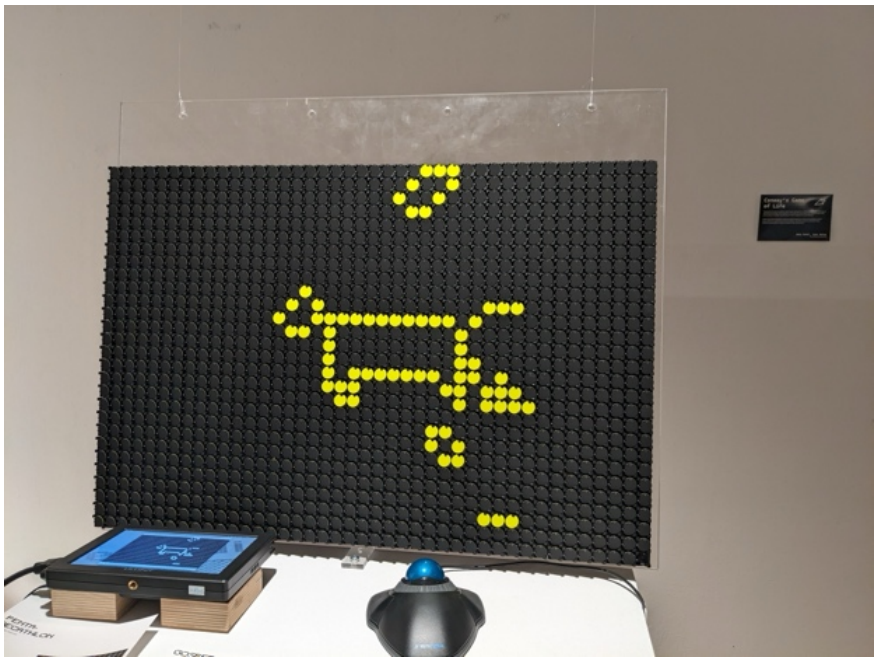


Figure 73: A dog doing its 'business' in a park by Anonymous artist

This installation was developed in response to a repeated YouTube comment on a flipdot's video I uploaded, asking to play the Conway's Game of Life. At first, I had no idea what that meant until a colleague explained it to me and offered to help execute the idea. Even though participants may have interacted with it in a way that wasn't intended, the unintended use case seemed more fun. The biggest takeaway from this process was the importance of sharing work and inviting as many people as possible into the process.

c. DFX 2025

Radical interfaces was presented at the Digital Futures Graduate Exhibition at OCAD University Waterfront Campus from March 28 to April 2, 2025.

Link to the video walkthrough ↗
Link to the user testing compilation displayed on the monitor ↗

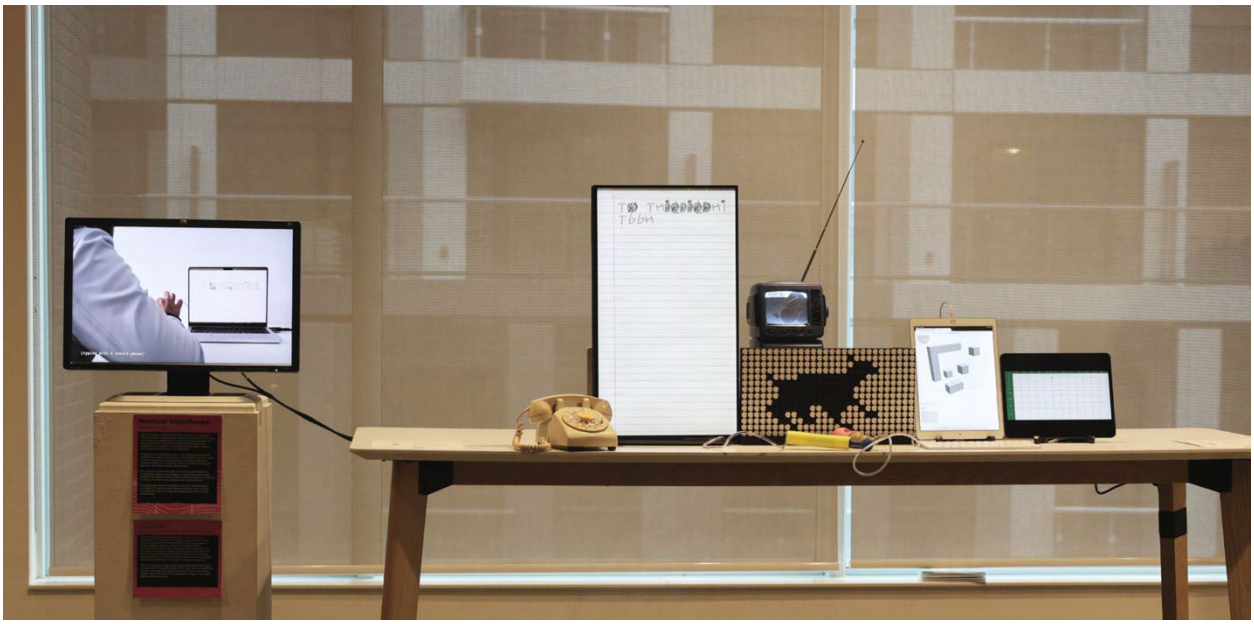


Figure 74: The setup

Figure 75: A visitor typing on the rotary phone

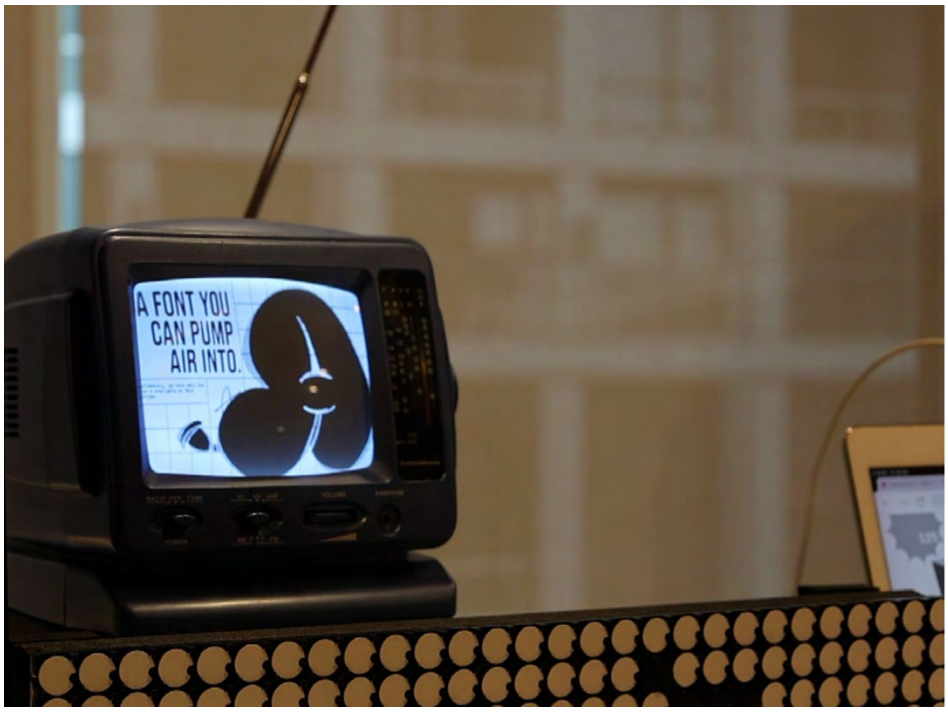


Figure 76: A font that inflates displayed on a tiny CRT

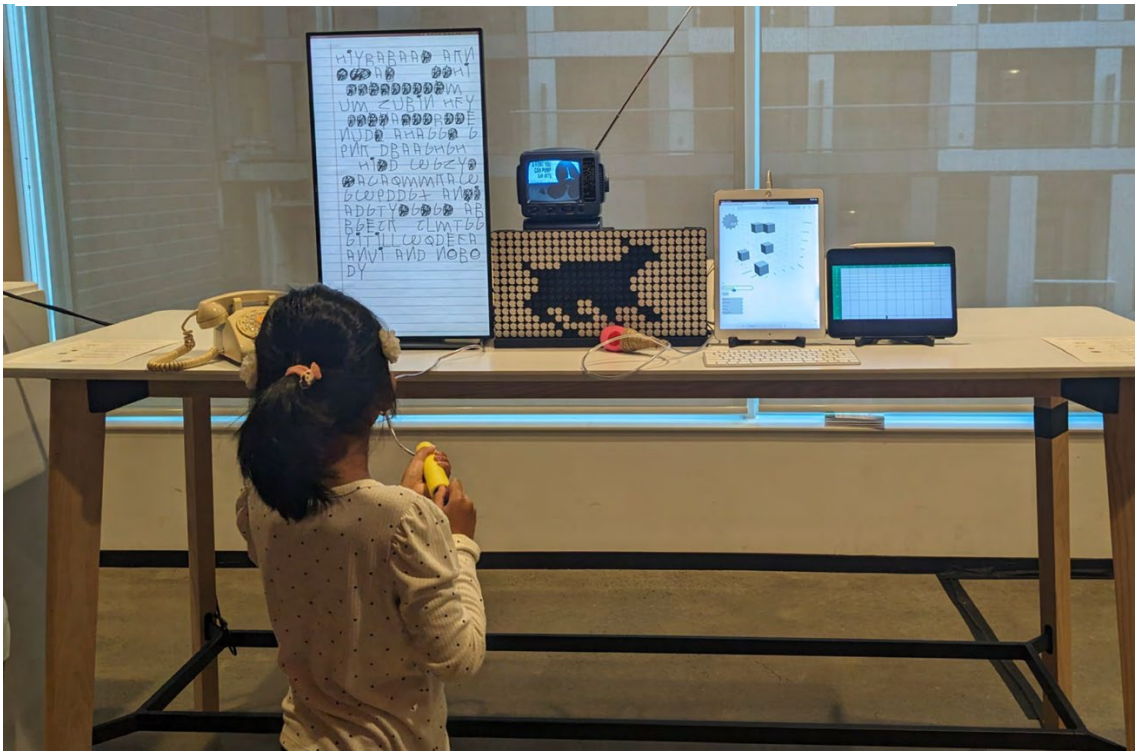


Figure 77: A young visitor pumping air into the fonts

RADICAL INTERFACES is a collection of five playful experiments that combine unconventional web apps, DIY electronics, and atypical graphic design methods to investigate how interfaces can be more curious, open-ended, and fun.

In order of appearance from left to right:

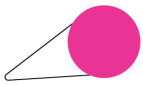
1. TYPING WITH A ROTARY PHONE

Pick up the receiver and the dial to type something.
1 = Backspace, 0 = Spacebar

Wait for the letters, they take their time.

2. A FONT THAT INFLATES

Use the airpump to pump air into the alphabet you see on the screen. Keep pumping to keep it inflated.

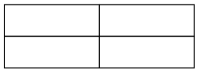


3. ANIMATING FLIPDOTS

Press the squishy pink sensor to make the dog run. The harder you press, the faster it will run.

4. 125

Build 3D by typing the x, y, and z coordinates using the keyboard. The order of the x, y, and z coordinates should not be changed.



5. I MADE ANIMATIONS ON EXCEL

Watch the animations made on a tool inspired by Microsoft Excel. (Note: This prototype is currently not interactive.)

Scan the QR code for more information and links.

Figure 78: User guide kept next to the prototypes