UNIVERSIDADE D
COIMBRA

Joana Maria da Cunha Oliveira

COMPUTATIONAL DESIGN OF LETTERS
FROM MODULES TO BODY THROUGH TEXTURE

**Dissertation in the context of the Master's in Design and Multimedia,
advised by Tiago Filipe dos Santos Martins and João Manuel Frade Belo Bicker,
presented to the Department of Informatics Engineering
of the Faculty of Sciences and Technology
of the University of Coimbra.**

September of 2022

À FAMÍLIA
Aos meus pais, José e Paula, e ao meu irmão, Daniel, pelo apoio e paciência,
À Ju que sempre percebeu o meu percurso,
À minha avó que mesmo sem perceber bem o que é *design,*
me apoiou e acalmou os meus receios e incertezas.

A COIMBRA
A todos que tive oportunidade de conhecer
e que me marcaram o caminho como designer e estudante,
Ao Alex, Petra, Surrador, Churro, João e ao Luís,
que se tornaram na minha segunda família,
que me acompanharam todos os dias destes cinco anos,
À Sofia e ao André por todas as noites e conversas,
pela paciência, por sempre me ouvirem e ajudarem,
A todos não mencionados pela amizade e memórias.

AOS DE BRAGA
A todos que estavam lá quando voltava a casa,
À Eva, Mariana, Carina, João, Jota, Bia, Faria e Álvaro
que sempre estiveram disponíveis para a minha indisponibilidade.

AOS MEUS ORIENTADORES
Obrigada pela disponibilidade, partilha de conhecimento
e pricipalmente pelo voto de confiança.

ABSTRACT

Typography, specifically the Roman Alphabet, is deeply rooted in western society. Apart from being a way to visualise language, its visual form is so embedded in our subconscious that it appears in worldly things. As a designer, we must understand how type can impact the reader and, foremost, how it can convey a message.

The technological advances in the realm of design made type a subject for exploration. Anatomic rules were broken, and typography delved into a world where straightforward communication was no longer prioritised. Letterforms evolved as a visual element of their own, and type development became more accessible, leading to its proliferation.

In this context, the presented dissertation aims to use the new technological advances to create a computational system capable of creating letterforms. Aside from exploiting their anatomic features, we also aim to explore their relation to other mediums, such as images. Therefore, the practical project of this dissertation focuses on developing a web-based system able to generate letterforms filled and textured with image crops. This system can produce various visually interesting outputs by implementing different ways to texturise and customise the letter.

First, we started with some experimentations, a quicker way to prove and demonstrate some initial ideas. This step led to creating a letter *skeleton* (monolinear and modular letters) which was later transformed into a computationally readable syntax. Eventually, we developed a method that reads the syntax and draws the intended letter skeleton into a web-based platform. Additionally, this method also applies the different image crops and textures to the skeleton.

Finally, we present a web-based platform that compounds an intuitive user interface with the aforementioned method, enabling any user to access and explore the system and its customisation parameters. This system also allows the user to apply a particular filling technique and customise a specific module of the skeleton created. Moreover, we present some real-world applications of the letterforms, showcasing the different mediums they can contaminate and how they can create relevance and another layer of information in a design context.

KEYWORDS
Computational Design, Multimodality, Texture, Type Design

RESUMO

A tipografia, e especificamente o alfabeto romano, está profundamente enraizada na sociedade ocidental. Além de ser uma forma de visualizar a linguagem, o seu formato visual está tão consolidado no nosso subconsciente que aparece nas coisas mais mundanas.

Os avanços tecnológicos no mundo do design fizeram da tipografia um alvo de exploração. As regras anatómicas foram quebradas e a tipografia imergiu num mundo onde a comunicação direta não era mais tida como prioritária. As letras evoluíram como um elemento visual próprio. Em acréscimo, o desenvolvimento de tipos tornou-se mais acessível, levando à sua proliferação.

Neste contexto, esta dissertação visa utilizar os novos avanços tecnológicos para criar um sistema computacional capaz de criar letras. Além de explorar as suas características anatómicas, também pretendemos explorar a sua relação com outros meios, como as imagens. Assim, o projeto prático desta dissertação foca-se no desenvolvimento de um sistema *web-based* capaz de gerar letras preenchidas com secções/recortes de imagens. Foram implementadas diferentes formas de texturizar e personalizar as letras, o que permite que o sistema produza diversos resultados visualmente interessantes.

Primeiro, começamos com algumas experimentações, uma forma mais rápida de provar e demonstrar algumas ideias iniciais. Esta etapa levou à criação de um esqueleto de letras (letras monolineares e modulares) que mais tarde foram transformadas em uma sintaxe legível computacionalmente. Eventualmente, desenvolvemos um método que lê esta sintaxe e desenha o esqueleto da letra pretendido numa webpage. Em acréscimo, este método também aplica os diferentes recortes de imagem e texturas ao esqueleto.

Por fim, apresentamos uma plataforma baseada na web que junta uma interface intuitiva com o método previamente mencionado, permitindo que qualquer utilizador aceda e explore o sistema e os seus parâmetros de customização e apresentamos algumas aplicações das letras, mostrando os diferentes meios que estas podem contaminar e como podem criar relevância e outra camada de informação num contexto de design.

PALAVRAS-CHAVE

Design Computacional, Multimodalidade, Textura, Desenho de Tipos

ACRONYMS

API: Application Programming Interface
CMYK: Cyan, Magenta, Yellow, and Key
CD: Compact Disk
CSS: Cascading Style Sheets
GTL: Generatore Tipografico di Libertà
HTML: Hypertext Markup Language
LED: Light Emitting Diode
OTF: Open Type Font
PDF: Portable Document Format
PNG: Portable Network Graphics
RGB: Red, Green and Blue
SFMA: San Francisco Museum of Modern Art
SVG: Scalable Vector Graphics
TXT: Text File
TTF: True Type Font
UI: User Interface
WOFF: The Web Open Font Format

LIST OF FIGURES

TABLE OF CONTENTS

1. INTRODUCTION

Humans have several cognitive capacities other animals lack, such as full-fledged language ability. It started with pictorial representations and later evolved to the alphabets now known by humankind. For an extended period, typography was only perceived "as a humble craft in the service of the written word", created to serve the communication needs of humans. A field restricted by tradition and elitism which slowly evolved beyond the printed page's constraints onto other mediums (van Leeuwen, 2006).

Aside from expressing the meaning of the writing word, typography started challenging it. Through stylistic resources and by exploring the impact of its features on the reader, typography began concerning more about the relationship among letters rather than the identity of individual characters (Lupton, 2010). Due to the needs of the digital era, communication has become more and more visual. Typography may now communicate even more meaning than it could before, transcending traditional restrictions and playing with the virtually limitless potential of the digital world. Similar to typography, and according to Barthes & Heath, pictures are polysemous; they can convey a message effectively and be full of communication value since they can tell a story, catch a moment in time, or even evoke an emotion (Barthes & Heath, 1977).

Designers are increasingly interested in such illustrative uses of typography, and in blurring the boundaries between letter forms and images, something which, in the 'old typography' was often frowned upon.

(van Leeuwen, 2006)

Technology advances and the appearance and spread of personal computers in the 1980s opened doors for unlimited possibilities. As expected, designers began adapting to these cultural and technological changes, and there was an increasing interest in the creative potential of code and programming. Type design was no exception. Initially, experimentation with type and its separation from the written word was frowned upon, especially by traditional type designers. However, exploring letterforms and all their features became much faster, easier and more accessible. Today, the alphabet is increasingly viewed as a topic for creativity and experimentation rather than merely a collection of tools used to express language (Willen & Strals, 2009).

Considering this window of opportunity, this dissertation intends to explore these unlimited possibilities. We started by developing a computational system able to create letterforms that explored the several features of typography and its ability to interact with different mediums, such as images, expanding the relationship between typography and colour. Moreover, we sought to create a system where users can choose a letter *skeleton* and customise its appearance by changing several parameters and options. We want to demonstrate that, even with the same skeleton base, the letterforms generated by the system might be radically different by just tweaking the available customisation settings and parameters. The core purpose of this project is to delve into typography as a conceptual and somewhat abstract way of conveying meaning. We also intend to comprehend and explore how it can ultimately bring relevance and visual interest to a design context and beyond the written word.

1.1. MOTIVATION

Typography is part of our daily lives as humans and crucial to the professional life of a designer. Designers are responsible for exposing typography to the world, hence the need for a keen look at type. Now, more than ever, typography is everywhere; it has surpassed the analogue and invaded the digital world. Type is no longer used exclusively to represent concepts and languages but to impact, express and sometimes shake the reader (or user) with the end message. With the technological revolution, a world of uncertainty and novelty was exposed, where letterforms go beyond the written word and become subjects for computational experiments. However, this proliferation of type raises a question of quantity versus quality.

A quote that can be applied to all worries related to type experimentation is "the rules that govern typographic design are not the dogmatic enslavement of creativity that some would have us believe" (Samara, 2011). This quote summarises how our approach to typography should be, especially when aiming to explore its possibilities and limitations.

A letter structure is composed of diverse aspects that implicate its appearance and style. Besides all its anatomic structural components, other variables, such as colour, can influence its appearance. Colour has always been part of type design, from the traditional black ink to the full range of colours allowed by colour systems such as RGB, CMYK, and Pantone. However, the relation of type with texture and other mediums, such as images, is yet poorly explored. Furthermore, human interaction with typography used to be fairly limited, but the digital era has opened up a world of possibilities. Taking advantage of these lacks and new prospects, we set out to explore typography to its full potential as a carrier of meaning while using different mediums, such as images and explore how these approaches can become visually and conceptually interesting when applied to a design context.

1.2. CONTEXT

The emergence of computers and technological advances created many tools for different areas, including design. Computational design is a relatively recent term that challenges designers, type designers inclusive, to rethink their processes. Historically, typography suffered a series of transformations and is now facing another challenge with the technological advances of the last century. The widespread use of programming languages and personal computers allowed other methods to be explored. In addition, it allowed for different and more alternatives for type design, far from the humanist need to conquer the perfect letterform.

Typography is turning a new page in its history, rebelling against its duty towards the written text and straightforward communication. We want to use technological and computational advances to explore and create letterforms that can convey, express and communicate not necessarily through their legibility but from an aesthetic standpoint, not letting its conventions limit the experiments.

1.2. SCOPE AND OBJECTIVES

As mentioned before, the primary goal of this dissertation is to explore typography as a conceptual and somewhat abstract means of transmitting meaning and information beyond the written word, as well as to understand how it might eventually provide relevance and aesthetic appeal to a design environment.

To achieve this goal, we developed a web-based system, available at https://phototyper.dei.uc.pt/, that creates letterforms from a base *skeleton* which is later filled and textured with crops of a source image. This filling method accepts a multitude of parameters that may be modified and together have a discernible influence on the appearance of the glyphs. Accordingly, we aim to show that the system can provide diverse and highly adjustable letter outputs even with the same skeletal foundation. Therefore, the users can choose the input image used to texturize the glyphs and manipulate their end aspect.

Finally, we planed to create a system that may be used by anybody, regardless of experience with typography or computational design methodologies. In addition, the outputs of these systems must be exportable, making them appealing to designers and other users who may utilise them in various scenarios.

1.3. DOCUMENT STRUCTURE

This dissertation is structured in four chapters: Introduction, State of the Art, Project Specifications, Practical Project, and Conclusions. The first chapter (Introduction) contextualises and introduces the project's theme, as well as what drives its development and what are the main objectives.

In the second chapter (State of the Art), we summarise and expose the theoretical foundation for the project. We begin with an overview of the history of the Roman alphabet before moving on to the multimodality of typography, which explores how it can convey meaning beyond the written word. Then we demonstrate the different structural components of letterforms, and finally, we look into different projects that explore type design, manually or computationally.

The third chapter (Project Specifications) begins with a conceptualisation of the project and also its objectives are specified. Then, we expose the development plan for this dissertation where each task is briefly explained and allocated into a time period in this subsection. In addition, we describe which methodology will be used to develop this practical project.

The fourth chapter (Practical Project) refers to the practical project itself, where we describe and detail all the work developed as part of the practical component of this dissertation. Finally, the last chapter (Conclusion) summarises all the work accomplished and what was concluded upon its development, in the context of this project, and also, we expose some ideas and goals for possible future work.

2. STATE OF THE ART

This chapter aims to create a theoretical foundation by compiling all the relevant information gathered while researching the theme of this dissertation. In order to sustain the practical project, we found it important to first understand the history of typography and how it has adapted throughout the years until today. Then we explore the multimodality of typography and image as a semiotic mode. Furthermore, we need to comprehend what constitutes each letterform and what is required to create a viable type. Finally, we list and discuss some related work that directly or indirectly explore type design, both manually or computationally.

2.1. HISTORY OF TYPOGRAPHY: AN OVERVIEW OF THE ROMAN ALPHABET

Typography makes at least two kinds of sense, if it makes any sense at all. It makes visual sense and historical sense.

(Bringhurst, 2004)

Bringhurst affirms that if typography makes any sense, it makes visual and historical sense. He adds that the first is more visible and approachable to the masses, while the other is hidden from most (Bringhurst, 2004). This first section will overview the Latin Alphabet to understand the evolution of its letterforms and their origin.

In Europe, the first evidences of those letterforms are the Greek Capitals carved in stone. During the second century BCE, the Roman Empire invaded Greece. Besides appropriating Greek art and religion, Romans also used the Greek letterforms as models for the formal lettering of the Roman Empire. Even after its collapse, the empire's legacy is still visible in western society. Throughout the centuries, these letters served as models for calligraphers and type designers, and the Latin/Roman Alphabet became the visual form of language in the Western World (Bringhurst, 2004; Meggs & Purvis, 2016).

European letterforms continued evolving and changing through time, leading to the familiar dichotomy, *majuscules* and *minuscules*, upper and lower case, as we call them now (Bringhurst, 2004). In the early fifteenth century, Johannes Gutenberg created and introduced movable type in

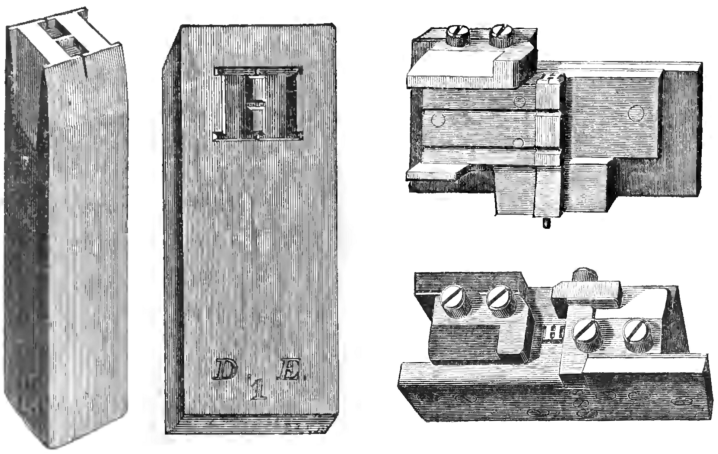


FIGURE 2.1.
Gutenberg's system for casting type. A punch (left) created a matrix (centre), which was placed into a handheld mold for casting type (right).

Europe (Figure 2.1.). This invention, employed earlier in China, revolutionised writing in the West. The difference in the amount of characters in the Latin Alphabet vs the ten thousand in the Chinese writing system makes the former more ideal for mechanisation (Lupton, 2010).

The fifteenth century also introduced italic letters, which became an economical option for calligraphers because it saved time, whereas, in printing, italic letters saved space. Today, the italic style in most typefaces is no longer a slated variation of the roman letter; it also includes the curves, angles, and smaller proportions associated with cursive forms (Bringhurst, 2004; Lupton, 2010).

In the 1500's, renaissance artists believed that all proportions should reflect the ideal human body, which also applied to the anatomy of letters. In 1529, Geofroy Troy published a series of diagrams that merged letters' anatomy with the human body's anatomy. This scientific approach was even more accentuated with the creation of the neoclassical typeface *roman du roi* (Figure 2.2.), developed by a government committee of Louis XIV in the 1690s in France (Bringhurst, 2004; Lupton, 2010).

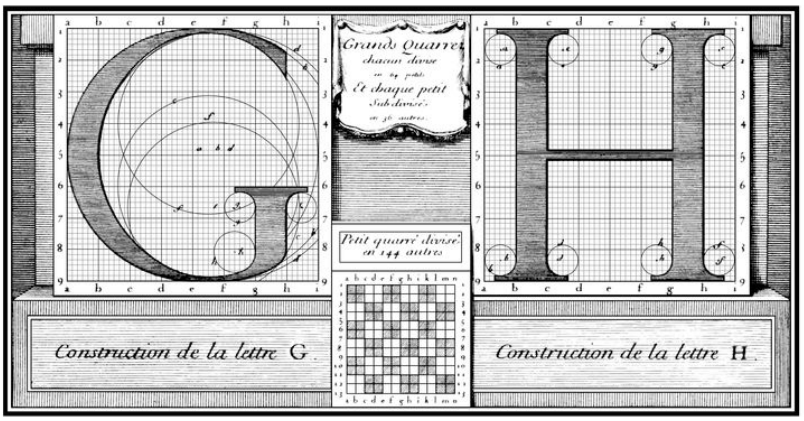


FIGURE 2.2.
Romain du Roi, construction of the letter G and H.

In the 1720s and 1750s, printers like William Caslon and John Baskerville abandoned the inflexible nib of humanism, favouring the flexible steel pen and pointed quill (writing instruments that rendered liquid). Baskerville's fonts were so crisp and contrasty that his contemporaries accused him of "blinding all the Readers in the Nation" because the strokes of his letters were "too thin and narrow". Giambattista Bodoni in Italy and Firmin Didot in France elevated Baskerville's severe vocabulary in the nineteenth century. Their fonts were the starting point for an explosive vision of typography untethered from calligraphy. This dehumanisation of the letter allowed for bizarre experiments (Lupton, 2010).

Later, in the nineteenth century, the rise of industrialisation and consumerism demanded new typography styles. Advertising was the new form of communication, which led type designers to create bigger and bolder typefaces. There was a need to create new typography styles and rethink how these were printed and later distributed to the masses. Many were the technologies used throughout the centuries that allowed and helped the evolution of typography. In the 1880's, Ottmar Mergenthaler invented the Linotype Machine (Bringhurst, 2004; Lupton, 2010).

KERNING

The adjustment of the space between two letters. (Lupton, 2010).

Although innovative, the creation of typefaces for the Linotype Machine was limited by three primary factors. For starters, kerning was impossible to achieve without using special compound matrices. Secondly, the em is only split into eighteen units. At last, the italic and roman matrices are frequently paired. As a result, in most cases, each italic letter had the same width as its roman equivalent. Many typefaces were still designed despite limitations, such as *Aldus* and *Optima* by Hermann Zapf (Bringhurst, 2004). Later, in 1887, Tolbert Lanston invented a machine that stamped individual letters in cold metal, in rivalry with Mergenthaler. However, this machine was quickly replaced by the Monotype Machine, made by Lanston's colleague John Bancroft in 1900. It resembled most computer-driven typesetting machines, formed with an output device and a terminal that carried a large mechanical keyboard composed of seven alphabets. However, the Monotype machine did not solve all the Linotype constraints. The em was still composed of eighteen units, but italic and roman were independent, kerning was possible and typeset lines could be adjusted by hand (Bringhurst, 2004).

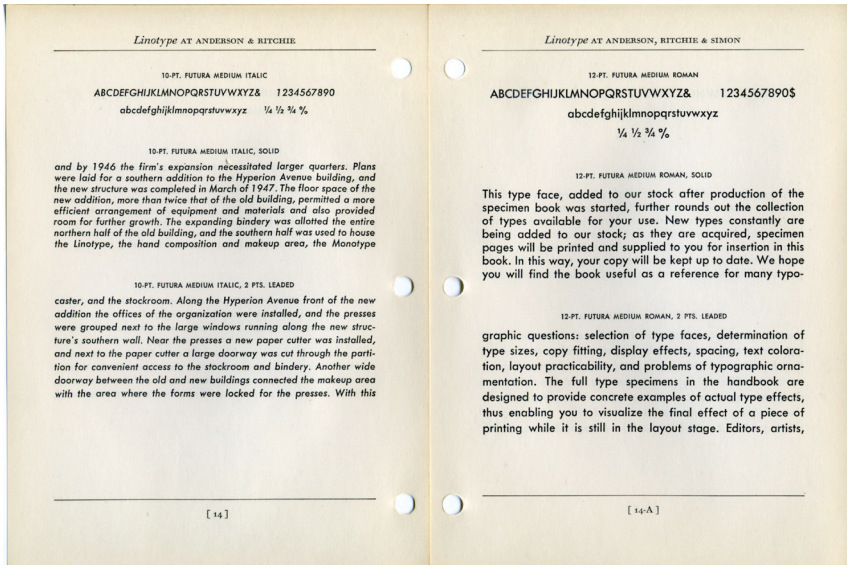
Other processes were used, such as two-dimensional printing and phototype machines. Some outstanding phototype faces were created like the Adrian Frutiger's *Apollo* (1962) and Bram de Does's *Trinite* (1982). However, as Bringhurst put it, "the era of phototype seems only a brief interregnum between hot metal and digital composition" (Bringhurst, 2004).

Many typefaces were designed throughout the centuries, and many were resurrected and reformed to fit the Linotype/Monotype machine. As ATF (American Type Founders) in the United States or Deberny & Peignot in France, several type foundries kept reviving faces from Garamond and Miklos Kis. They embraced new creations from designers like Hermann Zapf, Jan van Krimpen and Adrian Frutiger. However, the rediscovery of the history and principles of typographic form in the twentieth century was not linked to any technology (Bringhurst, 2004).

Some designers saw the alphabet's deformation as immoral and linked to a destructive industrial system. In 1906, Edward Johnston resurrected the search for a universal alphabet. Inspired by the Arts and Crafts movement of the nineteenth century, Johnston went to the Renaissance and Middle Ages for clean letterforms. Despite his admiration for history and uncorrupted letters, he and other reformers helped redefine the designer as a distinct intellectual from the commercial mainstream (Lupton, 2010).

While members of the De Stijl movement in the Netherlands reduced the alphabet to perpendicular elements, Herbert Bayer and Josef Albers at the Bauhaus constructed letters from primary geometric forms believed to be the universal visual language. The alphabet was regarded as a system of abstract relationships by these avant-garde designers, who ignored historical forms. They rejected the search for fundamental letters rooted in the human body. However, they discovered theoretical alternatives in place of "the solicitous novelty of mainstream advertising" and commercial pressures (Lupton, 2010).

Futura, designed by Paul Renner in 1927 (Figure 2.3.), encapsulated the avant-garde's concerns in a multifunctional, commercially available typeface. Renner tempered the geometry of *Futura* with subtle changes in stroke, curve, and proportion, despite disdaining the movement of calligraphy in favour of "calming" and abstract shapes (Lupton, 2010).



Linotype AT ANDERSON & RITCHIE

10-PT. FUTURA MEDIUM ITALIC

ABCDEFGHIJKLMNOPQRSTUVWXYZ& 1234567890

abcdefghijklmnopqrstuvwxyz ¼ ½ ¾ %

10-PT. FUTURA MEDIUM ITALIC, SOLID

and by 1946 the firm's expansion necessitated larger quarters. Plans were laid for a southern addition to the Hyperion Avenue building, and the new structure was completed in March of 1947. The floor space of the new addition, more than twice that of the old building, permitted a more efficient arrangement of equipment and materials and also provided room for further growth. The expanding bindery was allotted the entire northern half of the old building, and the southern half was used to house the Linotype, the hand composition and makeup area, the Monotype

10-PT. FUTURA MEDIUM ITALIC, 2 PTS. LEADED

caster, and the stockroom. Along the Hyperion Avenue front of the new addition the offices of the organization were installed, and the presses were grouped next to the large windows running along the new structure's southern wall. Near the presses a new paper cutter was installed, and next to the paper cutter a large doorway was cut through the partition for convenient access to the stockroom and bindery. Another wide doorway between the old and new buildings connected the makeup area with the area where the forms were locked for the presses. With this

[14]

Linotype AT ANDERSON, RITCHIE & SIMON

12-PT. FUTURA MEDIUM ROMAN

ABCDEFGHIJKLMNOPQRSTUVWXYZ& 1234567890$

abcdefghijklmnopqrstuvwxyz

¼ ½ ¾ %

12-PT. FUTURA MEDIUM ROMAN, SOLID

This type face, added to our stock after production of the specimen book was started, further rounds out the collection of types available for your use. New types constantly are being added to our stock; as they are acquired, specimen pages will be printed and supplied to you for insertion in this book. In this way, your copy will be kept up to date. We hope you will find the book useful as a reference for many typo-

12-PT. FUTURA MEDIUM ROMAN, 2 PTS. LEADED

graphic questions: selection of type faces, determination of type sizes, copy fitting, display effects, spacing, text coloration, layout practicability, and problems of typographic ornamentation. The full type specimens in the handbook are designed to provide concrete examples of actual type effects, thus enabling you to visualize the final effect of a piece of printing while it is still in the layout stage. Editors, artists,

[14-A]

FIGURE 2.3.

Specimens of Linotype Composition: Specimens of Hand Composition – Futura Type Specimen, Anderson & Ritchie, 1951.

The fast spread of technologies for machine composition and a corresponding fall in conventional type-founding activities facilitated the typographic advancements of the first half of the twentieth century (McNeil, 2017). Following World War II, many individuals craved a sense of order and neutrality, which culminated in the International Typographic Style (Swiss Style). This movement prioritised simple organisation and readability in visual communication. It's popularity sparked a revived interest in simplicity, prompting the development of numerous new sans serif types, most notably *Univers*, *Folio*, and *Neue Haas Grotesk*, which eventually became *Helvetica*, all of which were launched in 1957 (McNeil, 2017).

In the 1960s and 1970s, there was an explosion of new display types, not merely for print but also for cinema and television, moving primary activities away from conventional printing companies (McNeil, 2017). In addition, since the beginning of the 1970s, the digital era began liberating old technologies from commercial pressures and allowing them to reclaim their importance (Bringhurst, 2004). Opposition to the modernist tradition's cold formalism arose and expanded worldwide in the 70s.

I took "Swiss Typography" as my starting point, but then I blew it apart, never forcing any style upon my students. I never intended to create a 'style'. It just happened that the students picked up—and misinterpreted—a so-called '"Weingart style" and spread it around.

Wolfgang Weingart (Phillips, 2015)

FIGURE 2.4.

Plakatentwurf, Kunstgewerbemuseum Zürich – Schreibkunst, Wolfgang Weingart, exhibition poster, 1981.

FIGURE 2.5.

The Swiss Poster, Wolfgang Weingart, exhibition poster, 1984.

Wolfgang Weingart started questioning the premises and norms that were hardening the inventions of the Swiss masters. He explored a different path focusing on offset printing and film technology (Figures 2.4.–2.5.). Weingart began to embrace collage as a method for visual communication, moving away from strictly typographic design. Unprecedented and innovative methods emerged to combine detailed visual information, juxtapose textures, and unify typography (Meggs & Purvis, 2016).

The introduction of personal computers to the graphic design industry in the early 1980s altered how design was perceived. Adobe's *PostScript* language appeared as a device-independent system for turning digital data into visual output. Subsequently, the first inexpensive, user-friendly design software packages for Apple Macintosh computers were introduced. They ushered in a seismic upheaval in the design, printing, and publishing sectors known as the desktop-publishing revolution (McNeil, 2017).

Early pioneers who embraced and explored the creative possibilities of the new technology include designer April Greiman and *Emigre* magazine designer/editor Rudy VanderLans and Zuzana Licko. Greiman looked at the visual qualities of bitmapped typefaces, the stacking and overlaying of computer-screen information, the synthesis of video and print, and the tactile patterns and forms that the new technology enabled. Bitmapped type and computer-generated textures were photostatted to a large scale

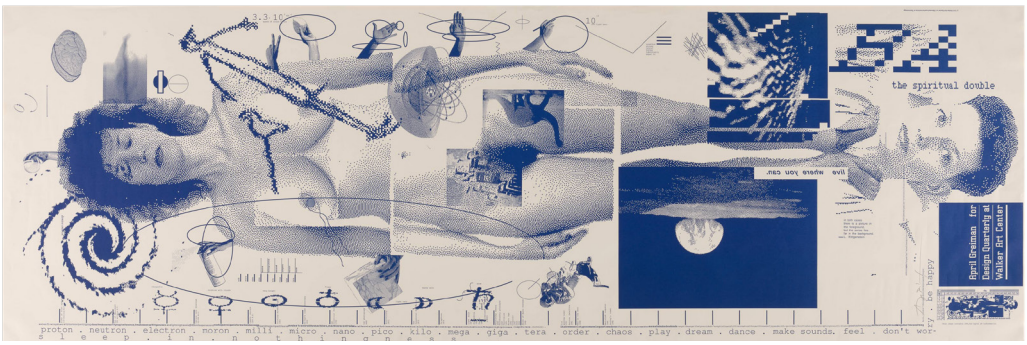


FIGURE 2.6.

Graphic for Design Quarterly – no. 133, April Greiman, 1987.

and pasted up through traditional typesetting in her first graphic design employing Macintosh output (Figure 2.6.) (Meggs & Purvis, 2016).

Zuzana Licko and Rudy VanderLands co-founded *Emigre* magazine in 1984 (McNeil, 2017). *Emigre*'s experimental approach, both in its editorial design and by displaying work that was frequently too experimental for other design publications, helped define and showcase the potential of the new technologies (Figures 2.7. – 2.8) (Meggs & Purvis, 2016).

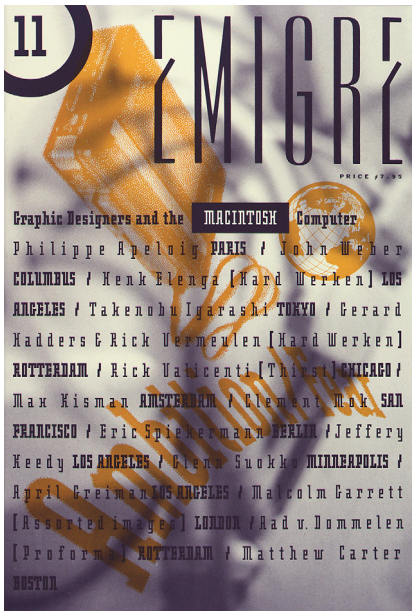


FIGURE 2.7.
Cover for *Emigre* – no. 11, Rudy VanderLans, 1989.

FIGURE 2.8.
Cover for *Emigre* – *no.* 10, Glenn A. Suokko and *Emigre* Graphics, 1989.

Another aspect that made *Emigre* arise as an original was its typefaces. Typefaces like *Oakland*, *Modula*, and *Matrix*, developed by Rudy's partner Zuzana Licko, propelled *Emigre* to the forefront of the emerging independent type foundry (Mr. Keedy, 1993).

Intolerance for different ideas is the biggest obstacle in these proscribed and dogmatic times.

(Mr. Keedy, 1993)

Simultaneously, a new generation of young designers questioned post-war modernist ideas and the established standards. By taking advantage of the new technology and despite their lack of experience, they could seize opportunities never seen before. These design thinking shifts helped fuel an era of intense creativity in type design in the 1990s. Many new types were inspired by considerable experimentation with the capabilities of digital technology. Other designs looked into Postmodern themes by blending vernacular and historical components to make hybrid letterforms that worked better as a conceptual expression than a functional product. Some examples of this method are the typefaces created by Barry Deck, P. Scott Makela and Jonathan Barnbrook (McNeil, 2017).

During this time, many art schools and university design education programs were crucial hubs for rethinking graphic design via theoretical debate and computer technology experimentation.

From 1971 until 1995, graphic designer Katherine McCoy co-chaired the design department at Michigan's Cranbrook Academy of Art with her husband, product designer Michael McCoy. It became a magnet for people interested in pushing design possibilities (Meggs & Purvis, 2016).

Edward Fella, a Detroit graphic designer who eventually attended Cranbrook's graduate program from 1985 to 1987, was a regular Cranbrook guest critic. He studied entropy and the dissolution of form caused by repetitive copying. Fella used various techniques, from found typography, scribbles, and brush writing to typesetting, rubdown letters, public-domain clip art, and stencils (Figure 2.9.) (Meggs & Purvis, 2016).

The extensive changes that occurred in the 1980s substantially influenced the economy. Anyone with interest, whether officially educated or not, could produce type once digital type-making tools became as inexpensive as conventional graphic design software. The previous argument, along with many other aspects of digital culture, had a significant role in developing and devaluing the typographic arts (McNeil, 2017).

David Carson, a former competitive surfer and schoolteacher, transitioned to editorial design in the 1980s. He rejected conventional ideas of typographic syntax and images in favour of exploring the expressive potential of each subject, page, or spread. Carson's article titles were often unevenly placed or organised in expressive rather than conventional ways. Carson's text type frequently questioned the essential standards for legibility. *Ray Gun* 14 was the first magazine Carson delivered to the printer as electronic data, despite being universally perceived as the pioneer of the computer revolution (Figure 2.10.). (Meggs & Purvis, 2016).

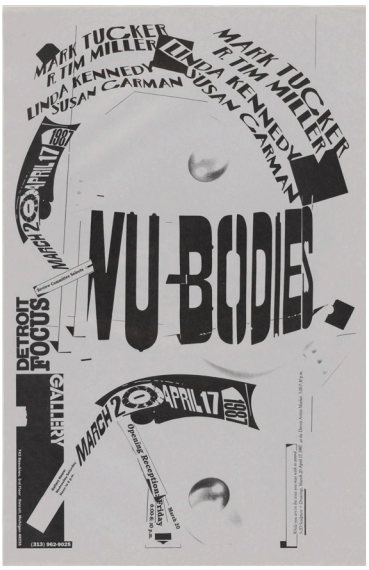


FIGURE 2.9.
Mailer for Detroit Focus Gallery, Edward Fella, 1987.

FIGURE 2.10.
Morrissey: The Loneliest Monk – Ray Gun, David Carson and Chris Cuffaro, 1981.

Early in the 1990s, graphic designers could produce outcomes almost equal to those produced using traditional working techniques because of fast advancements in computers software, and output devices. By this time, Carson had attracted much controversy. While many young designers found great inspiration in him, he also angered others who thought

he was crossing the line between order and chaos. He thinks one should not confuse communication with legibility since more expressive designs may draw readers in and keep them interested, when many highly legible traditional printed messages provide nothing of visual interest for the reader (Meggs & Purvis, 2016).

In 1991, Jon Wozencroft and Neville Brody published the first of eighteen issues of FUSE Magazine. Each issue was delivered in a cardboard box and a floppy disk containing the typefaces used in the posters that made up each issue (Hamamoto, 2012). FUSE was constantly looking for novel, unusual forms of communication, believing that type was not only meant to be read but also to be read about. Although the design community now considers it standard practice, the Brody and Wozencroft's project at the time ignited an unheard-of and unparalleled typographic revolution. Among the bold and ambitious designers, it encouraged radical invention, iconoclastic experimentation, and liberating self-expression (PrintMag, 2012).

At the start of this decade, design education and the design industry both had one foot in the then-emerging digital technology and one foot in more conventional means of design production. The Cranbrook Academy of Art was no exception. As was already noted, Katherine McCoy, the head of Cranbrook's graphic design program, stressed creation and pushed students to create their design philosophies and processes.

Martin Venezky is one of those students whose education at Cranbrook continues to influence his work until today. The *Sundance Film Festival*, Reebok, the San Francisco Museum of Modern Art, and *Speak* Magazine are among his most well-known creations. He frequently employs collage materials, digital photographs, and altered or warped type in his work, which expertly fuses handwork with technology. He is fascinated by patterns, rhythm, and the structural features of letterforms (Meggs & Purvis, 2016). Venezky was abble to capture this idea of "uncontrolled excess" in the *Speak* Premiere Issue, which was not a rarity in the mid-90s (Venezky, n.d.).

FIGURE 2.11.

FUSE-1 Magazine (left), Jon Wozencroft and Neville Brody, 1991.

FIGURE 2.12.

Spread from the *Sundance Film Festival* program (right), Martin Venezkya, 2001.

As a result of the new technologies, small digital type foundries like *Emigre*, *GarageFonts*, T-26*, Hoefler & Co., The Font Bureau,* and *FontFont* saw rapid growth. At the same time, larger companies like Monotype, Linotype, Adobe and others launched significant initiatives to digitise, update, and expand their type libraries. Numerous successful historical reconstructions were attempted, along with brand-new designs created especially for digital output (McNeil, 2017).

Emigre Fonts started getting many unique and eccentric fonts from outside designers. Many of these entries' underlying formal ingenuity and uniqueness were noticed by Licko and VanderLans, who then started licensing and disseminating the designs. Some examples of typefaces were P. Scott Makela's *Dead History* and Jonathan Barnbrook *Exocet* typeface. These typefaces frequently raised controversy despite being widely and quickly employed in important advertising campaigns and publishing designs (Meggs & Purvis, 2016). With the World Wide Web's growth, network connections became more critical. As a result, several fonts were created to be easily readable on low-resolution computer displays and later proven to be equally effective in print (McNeil, 2017).

Due to the minimal protection against the unauthorised use of copyrighted fonts, type for the Web was not warmly embraced by the profession when it was initially introduced in the late 1990s. However, since 2008, web browsers have been created to make licensing easier. In the twenty-first century, typography is more accessible, pervasive, and intimate than ever. The mechanised word is being used in novel ways thanks to the Web, and interactive media, opening up formerly unimaginable possibilities for the use of type in modern society. Typographic text is increasingly employed on displays with resolutions virtually on par with printing, not just for mass communication but also for interpersonal social interactions. Unlike printed publications, interactive technologies allow designers to create a flexible framework for visual communications while retaining some degree of creative freedom. As a result, communication as an ongoing process rather than a fixed broadcast faces previously unrecognised challenges (McNeil, 2017).

2.2. MULTIMODALITY OF TYPOGRAPHY

Prehistoric humans left countless petroglyphs, all across the planet, from Africa to North America to the islands of New Zealand. These pictographs or ideographs (symbols for thoughts or concepts) evolved in two ways: first, they marked the beginning of pictorial art; second, they formed the basics of writing. Through time, the pictorial character of the petroglyphs was slowly retained at the expense of a more syllabic representation, making them symbols of the spoken language sounds (Meggs & Purvis, 2016). Resembling Darwin's natural selection theory, writing systems and visual signs suffered a cultural evolution, where the better varieties survived while the less desirable ones perished (Changizi, 2010).

This section focuses on the multimodality of typography and how it can convey meaning beyond the textual and verbal context. Communication has become increasingly visual due to the demands of the digital era. In this new era and far from the limitations of printing technology, typography can convey even more meaning, breaking previously established boundaries and experimenting with the unlimited possibilities of the digital world.

Multimodality refers to the use of many sensory and communication modes to create meaning in a message, including sound, text, video and images. In a way, every communication is multimodal because its meaning is created via writing and through typography, drawings, page design, and other methods, especially in pre-digital periods (Dressman, 2019).

Kress and Theo van Leeuwen argued in *Reading Images – The grammar of visual design* book that images are a semiotic mode in and of itself, a form of language. Halliday's metafunctional theory states that spoken and written texts fulfil three major communication roles. The first is the ideational metafunction, which creates representations of what is happening in the world (and in our minds). The second one, interpersonal, describes how language creates social relationships and communicates feelings. At last, the textual metafunction enables us to use language to represent individual interactions into coherent texts. Images can convey meaning and meet these metafunctions through composition, framing, angle, colour and light (van Leeuwen, 2006).

...this in the name of a certain mythical idea of Life: the image is re-presentation, which is to say ultimately resurrection, and, as we know, the intelligible is reputed antipathetic to lived experience.

(Barthes & Heath, 1977)

In a related remark, Roland Barthes claimed that some believe that signification cannot correctly represent the infinite complexity of an image. In contrast, others believe an image is a rudimentary system compared to language. Barthes believes that an image's details can contain three messages: a linguistic message, a non-coded iconic message, and a coded iconic message (Barthes & Heath, 1977).

Today, it seems like every image has a linguistic message, whether in the form of a title, caption, news article, movie dialogue, or comic strip balloon. Writing and speech continue to be the two main compo-

nents of the informational framework of our society. Barthes claimed that the linguistic message had two purposes: anchoring and relaying. The first function is identified when text replies — more or less directly — to the inquiry: what is it? The text helps to identify purely and simply the elements of the scene, preventing the connotative meanings from expanding. The text directs the readers through the signs of the image, causing them to avoid some and receive others; it remote-controls them towards a meaning chosen in advance, which is very common in advertising images. The function of relay is less common in fixed images, becoming very important in a film. Here the dialogue functions advance the action by specifying the sequence of meanings not found in the image itself (Barthes & Heath, 1977).

The second message, a non-coded iconic message or denoted image, is hard to encounter, at least in advertising. A denoted image as such, cleared of all its connotations, would become radically objective or innocent. Until a certain level, we can find this in a photograph (in its literal state), which seems to constitute a message without a code by its analogical nature. Without intervening within the object, a photograph can transmit the (literal) information without forming it through discontinuous signs and transformation rules. On the other hand, a drawing, even when denoted, contains a coded message intervening in the scene being captured. In other words, the denotation of the drawing is less pure than that of the photograph, for there is no drawing without style. A photograph can, in some sense, elude history and represent a 'flat' anthropological fact, becoming a message without a code (Barthes & Heath, 1977).

Barthes also points out that the third message (the *symbolic* message, cultural or connoted) is discontinuous. The variation of cognitive signs found in one image is not anarchic; it varies according to individuals and depends on the different kinds of knowledge — practical, national, cultural, and aesthetic — invested in the image. An image, in its connotation, is constituted by an architecture of signs drawn from a variable depth of idiolects (Barthes & Heath, 1977).

In conclusion, all images are polysemous; they imply, underlying their signifiers, a "floating chain" of signs, and the reader is able to choose some and ignore others. They can be easily perceived as bearers of meaning and can be full of communication value, as they can tell a story, capture a moment or even an emotion (Barthes & Heath, 1977).

On the other hand, typography can be a little more complex, and for a long time, it was not perceived as a semiotic mode in its own right. However, new typography is emerging that no longer is a slave to the "humble craft" of serving the writing word. The designer Neuenschwander calls typography "a fully developed medium of expression, possessing a complex grammar by which communication is possible", and Swiss designer Hans-Rudolf Lutz has also claimed that design is also information — "Gestaltung ist auch Information" (van Leeuwen, 2006).

Designers are becoming more interested in blurring the lines between letterforms and images, which was once frowned upon by traditional typographers. Typography may also operate as a medium for enacting relationships and expressing feelings about what is being portrayed. Many typographical indications that are not letterforms, such as punctu-

ation marks, realise the textual meaning, and they are continually developing new applications (van Leeuwen, 2006).

Semiotic resources can be organised as a *medium* or as a *mode*. For example, the semiotic resource of colour has evolved from a medium to a mode. In the Mediaeval age, pigments had values in themselves, and each pigment was used as a unique identity and character. For instance, the Ultramarine pigment was imported through the sea and made from lapis lazuli. Therefore it was only used in majestic and valuable artefacts, like the mantle of the Virgin Mary. Then around 1600, were produced, in the Netherlands, oil paintings that could be mixed. This discovery turned colour into a combinatory system of five primary/abstract colours ('blue' in general, rather than a specific blue). This system allowed for other colours to be created. By then, colour was no longer a 'lexis' but now had a 'grammar' associated with it, just like language has a system of rules and principles for speaking and writing (van Leeuwen, 2006).

For a long time, typography has been only seen as a 'medium', where letterforms were treated as distinct individuals with different provenances rather than systematically. Although traditional typographers rejected typography as a *'system',* typographer and book designer Jan Tschichold attempted to change it. He analysed letterforms into many interchangeable components possible (e.g. the bowls of a, b, p, d, g and q were considered identical when typically they were not) (van Leeuwen, 2006).

Ruari McLean wrote in the Manual of Typography that "to a very limited extent, lettering may help to express a feeling or a mood that is in harmony with the meaning of the words'', but for the most part 'lettering and calligraphy are abstract arts [...] What moves us is something formal, and, in the last resort, inexplicable". However, this slowly changed. Nowadays, there is a provocation to break and rethink the boundaries between typography and other graphic and photographic arts while recognising its semiotic nature (Van Leeuwen, 2005).

Ordinary words convey only what we already know, it is from metaphor that we can best get hold of something new.

Aristotle, 400 BCE
(Van Leeuwen, 2005)

Metaphors are the critical principle for semiotic innovation, but nowadays, they are used not to create new ideas but to express them. When a semiotic resource is a *medium*, its meaning comes about through two principles: experiential metaphor and connotation. The experiential metaphor principle believes that there is a meaning potential that stems from our physical experience and our capacity to, metaphorically, expand it. For instance, metaphorically, a bolder letterform carries meaning, such as assertiveness, whereas the opposite can be perceived as insubstantial'or timid. The second principle, connotation, refers to the idea that signs can be imported from one context into another. Usually, letterforms are regarded as denotative — a descender with a bowl on the right denotes the sound 'p'. However, this cannot be said of the logo created for the art magazine Parkett by Bice Curiger. The logo, embroidered by the designer's mom, 'imports' the traditionally handcrafted product to the realm of logo design.

It does so by rejecting standard corporate logo designs as overly institutional and affirming the significance of handcrafted, traditional forms of expression. The semiotic potential is within typography and can be externalised through its different features such as weight, expansion, curvature and orientation (van Leeuwen, 2006).

In the age of computer-mediated communication, typography is crucial in forging the new relationships between images and graphics. Letterforms require a form of communication that is, on the one hand, far more oriented toward writing than previous screen media such as film and television, but on the other hand far more visually oriented than previous media such as books. Typography is becoming more and more multimodal and is being increasingly integrated and challenged with other means of semiotic modes such as images, movement, colour and texture (Van Leeuwen, 2005; van Leeuwen, 2006).

2.3. LETTER ANATOMY

The practical component of this dissertation aims to explore and experiment type to its full potential. For this to be possible, there is a need to understand all its components and features. Besides serving as a visual representation of language, typography also has its own terminology that help type designers to create and use letterforms correctly.

Before diving into its anatomy, it is essential to first understand some basic concepts. The terms *character* and *glyph* are sometimes used interchangeably. However, *character* is related to a specific symbol (letter, number or punctuation), whereas *glyph* is a distinct expression of a given *character*. For example, the *character* a in lowercase, can be presented by many *glyphs* — á, ã (Lupton, 2010).

The terms *font* and *typeface* are also commonly misused. Historically, *font* was used to refer to a single set of metal type in a specific size and style. However, the digital revolution allowed typefaces to embrace a range of sizes, reforming the term which now is used to refer to a single style (light, bold, italic, and so on) (Cheng, 2020).

The term *typeface* was once used to describe a family or group of related *fonts* (for example, *Garamond* Regular and Italic in 10, 12, and 14 pts). This term is still correct in this context. However, when referring to a *typeface*'s design how it looks or works — the term *typeface* is also accurate. Whereas, when referring to the actual digital software file (the *OpenType* or .otf file), we use *font* (Cheng, 2020).

Different measurements and guides help the creation and even the readability and legibility of type (Figure 2.12). It is crucial to have them in mind when assembling text and also to create a typeface. The *baseline* and *x-height*, help determine the consistency and personality of both lettering and type (Willen & Strals, 2009). All of the letters are placed on the *baseline*, making it the most stable axis along a line of text. Therefore it is the most critical reference for aligning text with images or other text (Lupton, 2010). The *x-height* is the vertical measurement of the main body of a lowercase letter, commonly indicated by the x. Increasing the *x-height* may make the letters appear larger and enhance legibility at small sizes; however, when excessive, it can have the opposite effect, lowering overall readability and making letters appear graceless. If the *x-height* is too little, the letters will appear top-heavy or stunted (Willen & Strals, 2009).

CAP HEIGHT

Refers to the distance between the baseline and the top of a capital letter, determining its point size (Lupton, 2010).

ASCENDER HEIGHT/ DESCENDER HEIGHT

Ascender height refers to the ascender of the letter. In turn, descender height is related to the descender (Lupton, 2010).

FIGURE 2.13.

Anatomic measurments of typography with *Garamond* Regular.

FIGURE 2.14.
Terminology of letter's anatomy.

The Figure 2.13. illustrates different terms used to describe structural characteristics of type. *Terminals* and *serifs* indicate the pen's entry and exit markings (Figure 2.14.). The origins of diverse serif forms may be traced back to distinct writing meth- ods, tools, pen angles, and pressure levels (Willen & Strals, 2009).

Serifs can be attached to a letter's main strokes using brackets, which are curved supports. They ease the transition between opposing strokes, enabling them to flow more elegantly (Cheng, 2020).

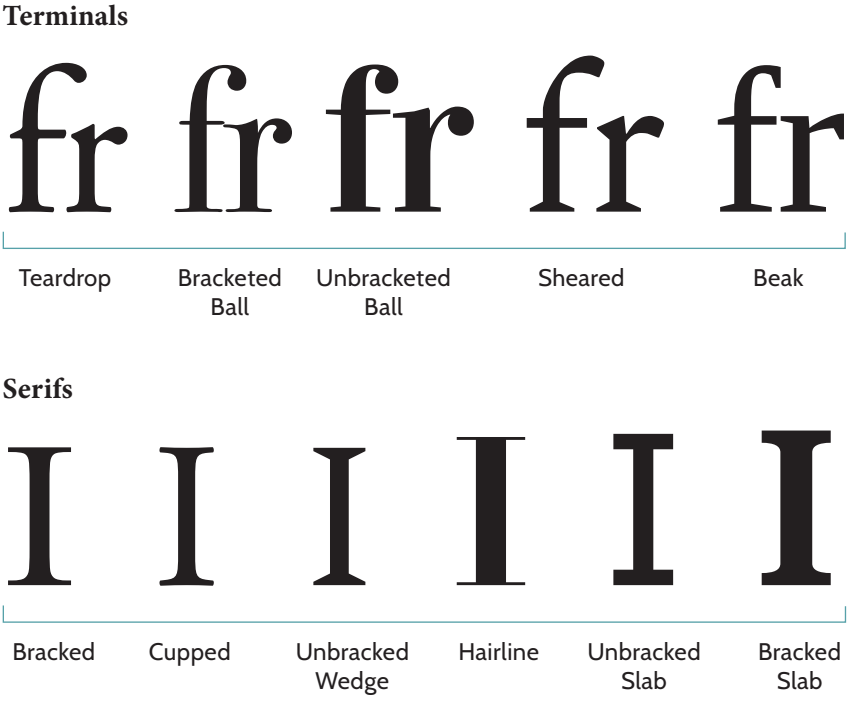


FIGURE 2.15.
Terminals and serifs terminology.

RELATIONS BETWEEN CHARACTERS

Letterforms are iconic and so deeply embedded in Western culture that they have stayed primarily unaltered for 2,000 years. The Roman alphabet's twenty-six letters are interconnected, forming a closed system of lines and spaces that serve as a code for the understanding of typography (Samara, 2011). However, there are different approaches to their construction and development.

The capital, or uppercase, are the direct descendants of the Roman imperial inscriptions. Despite being the oldest, they have proven to be the most simply drawn (Samara, 2011). The capitals' forms are more restricted (all the same height) and frequently built according to specified rules. Additionally, numerous letters have the same shape in both cases (O/o, C/c, S/s, V/v, W/w, X/x, Z/z), thus creating the capitals first gives a bit of a learning scaffold into the lower case (Cheng, 2020).

Cheng adds that in order to facilitate the creation of letterforms, they can be grouped in three groups, based on their shapes: circles — O Q C G S —, triangles — V A W X — and squares — E F L H I T. Some letters which are more complex need to combine different shapes: D, B, P, R are

circular-square and M, N, K, Y are diagonal-square. Throughout the book, she suggests other categorisations for letterforms, for example, double-story letters or open-side letters.

Alternately, Samara groups capital letters into different groups according to the variety of linear forms that constitute them. The letters E, F, H, I, J, L, T, for example, are composed only of horizontal and vertical strokes. The A, K, M, N, V, W, X, Y, Z are all letters that share another structural variation, the diagonal. The same holds for the set of letterforms created with curved strokes, often in conjunction with straight ones, C, D, O, P, S, Q, U. And finally a fourth category that combines curved, straight, and diagonal strokes, B, G, R (Samara, 2011).

Capital letters can be created using one of two proportional systems: classic (old style) or modern. Classic proportions are based on inscriptional models. The Romans employed a square, or its geometries (the golden rectangle and the root five rectangle), for the widths of capital letters for both artistic and functional reasons.

On the other hand, modern proportions place a premium on colour consistency. Every letter must have the same amount of white space (Cheng, 2020). Minuscules, or lowercase letters, were invented considerably later, be- tween the seventh and ninth centuries, and were incorporated into the written language in just a few hundred years. Their drawings are more intricate and include rounder strokes. Lowercase letters have more variation and aesthetic differences (Samara, 2011).

Due to their more complex structure, Cheng adds other groups to the previous categories for the upper case letters, namely branched letters — n, h, m, u, r —, letters based on a single vertical stem — i, l — and hooked forms — j, f, and t (Cheng, 2020).

Being able to recognize tiny differences between the forms helps designers understand what makes letters act a certain way.

(Samara, 2011)

Balancing the proportions and sizes of these various shapes is a known difficulty in type design. Square forms (like the H and E) are physically larger than round and triangular shapes (like the O and V). Therefore, round and pointed shapes must be stretched over the capline and below the baseline in order to balance the forms. Overshoot (at the capline) and undershoot (below the baseline) are terms used to describe this type of adjustment in both upper and lower case letters. The location of the visual centre creates a similar illusion. A letter's optical centre is a little higher than its mathematical centre. As a result, a crossbar in the precise centre of the H will look to be too low. The crossbar adjustments also influence other letters with similar appearances, specifically double-story letters (H, E, B, S, 6, 8). Despite being praised among type designers, some typefaces deliberately ignore these rules of optical compensation as part of their design concept (Cheng, 2020).

2.4. RELATED WORK

This section demonstrates different projects where the experimental side of typography is welcomed, may that be manually or computationally. These projects were chosen based on their potential to be relevant references for the objectives of the practical section of this dissertation. Moreover, they are historically and conceptually relevant to the overall project. This section is composed with two subsections, 2.4.1. Conceptual and modular typography and 2.4.2. Digital and generative typography.

2.4.1. CONCEPTUAL AND MODULAR TYPOGRAPHY

As briefly mentioned in section 2.1., a reform marked the twentieth century in the ways designers perceived and interacted with typography.

There was a shift that pushed letterforms beyond the traditional shapes of the Roman Alphabet into a more abstract world. The mechanised cruelty of World War I ultimately put a stop to the Art Nouveau movement, ushering in a slew of new styles of lettering and typography (Willen & Strals, 2009).

In 1909, Fillipo Marinetti opened the doors to the Futurist movement with his publication of the Manifesto of Futurism in the Paris newspaper *Le Figaro*. War, the machine era, speed, and contemporary living were all mentioned in the manifesto. It astonished the public when it declared, "We will destroy museums, libraries, and fight against moralism, feminism, and all utilitarian cowardice". The Futurism movement's followers created explosive and emotionally charged poetry, defying proper syntax and language, advocating for a typographic revolution against its classical heritage. In response, parole in libertá, or "words in freedom", formed a new and fresh, impersonal, and artistic typographic style (Figure 2.16.–2.17.). They explored linguistics and typography as a concrete and expressive visual form (Meggs & Purvis, 2016).

FIGURE 2.16.

Le Soir, couchée dans son lit, elle relisait la lettre de son artilleur au front (left), Filippo Marinetti, 1919.

FIGURE 2.17.

Les mots en liberté (right), Filippo Marinetti, 1919.

Shortly after Futurism, the Dada movement appeared as a reaction to the carnage of World War I having a strong negative and destructive character (Meggs & Purvis, 2016). Similarly to Futurism, Dadaism hoped to

undermine the meaning of words by pushing the bounds of legibility and readability (Willen & Strals, 2009). One of the most influential Dada artists was Raoul Hausmann. His work was characterised by the representation of visual speech sounds (Figure 2.18.) (Bargues, 2016).

In the De Stijl movement and at the Bauhaus, modernist designers experimented with strictly geometric interpretations of the alphabet that erased any humanist vestiges from their letterforms (Willen & Strals, 2009). Although different, the designer Théo van Doesburg considered Dada and De Stijl as opposing, but complementary movements: Dada might demolish the old order, while De Stijl could rebuild it "on the razed site of prewar culture". In the poster *Kleine Dada Soirée* (Figure 2.19.), Kurt Schwitters joined Van Doesburg to create a typographic composition illustrating his Dada side (Meggs & Purvis, 2016).

FIGURE 2.18.

Kp'erioum,
Raoul Hausmann,
1886-1971.

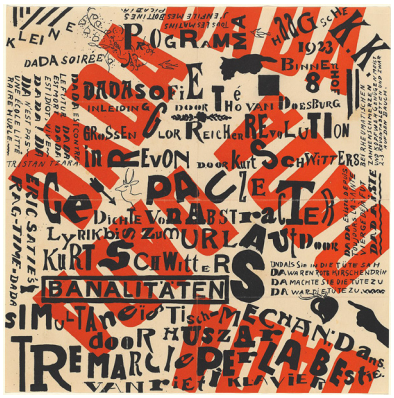


FIGURE 2.19.

Kleine Dada Soirée,
Theo van Doesburg and
Kurt Schwitters, 1923.

The twentieth century was also known for its experiences with modular typography. Almost all fonts are modular in one sense: their systems are frequently based on a comparable collection of shapes and markings. However, most fonts modify their qualities to suit each character's demands and structure whereas, modular letters follow a rigorous system with a defined set of modules. These parts started as geometric and basic shapes, but designers experimented with more elaborate forms and even tangible items (Willen & Strals, 2009).

The De Stijl movement believed that letters, as paintings, architecture and objects, should be reduced to their elementary components. Théo van Doesburg, one of the founders of the movement, designed in 1919 an alphabet with only perpendicular lines (Figure 2.20.) (Lupton, 2010).

FIGURE 2.20.

Square alphabet,
Theo van Doesburg,
1919.

From 1917 until his death in 1931, he edited and produced the periodical De Stijl. This journal promoted the movement's ideas and called for practical art to absorb pure art. De Stijl became a natural vehicle for conveying the movement's beliefs in graphic design. Vilmos Huszár designed the journal De Stijl logo with letters constructed from an open

grid of squares and rectangles (Figure 2.21.). Herbert Bayer created a universal type (Figure 2.22.) at the Bauhaus that reduced the alphabet to simple, and rationally constructed letterforms (Meggs & Purvis, 2016).

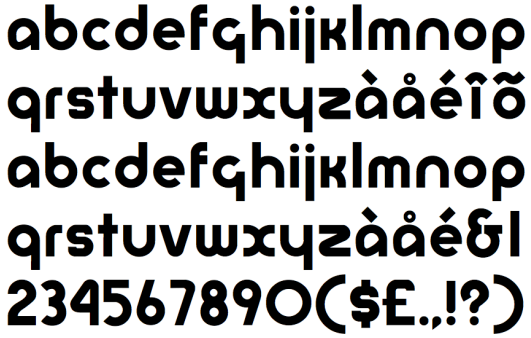
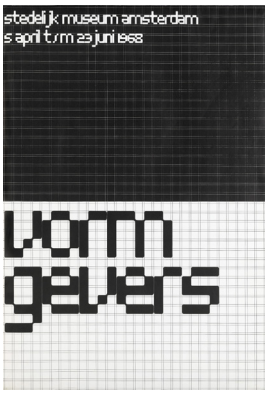


FIGURE 2.21.
Magazine *De Stijl,*
Vilmos Huszár, 1917.

FIGURE 2.22.
Universal alphabet,
Herbert Bayer, 1925.

In 1997, Wim Crouwel designed a "new alphabet" (Figure 2.23.). His typography is well organised and adheres to rigorous grid schemes. This characteristic is clearly shown in the posters he created for the Stedelijk Museum (Figure 2.24-2.25) (Grrr.nl, n.d.).

FIGURE 2.23.
New Alphabet,
Wim Crouwel, 1968.



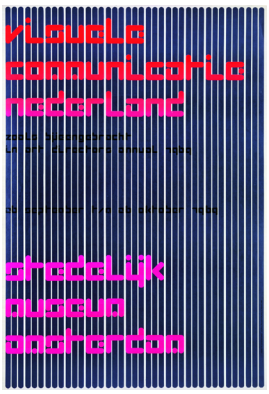


FIGURE 2.24.
Designers,
Wim Crouwel, 1968.

FIGURE 2.25.
Visual communications in the Netherlands,
Wim Crouwel, 1969.

2.4.2. DIGITAL AND GENERATIVE TYPOGRAPHY

The rising of new technologies and their increasing accessibility provided the needed resources for typographic exploration and experimentation. In 2011, the Museum of Modern Art in New York added a first to its Architecture and Design Collection: a collection of digital fonts. Only 23 designs were chosen for the *Standard Deviations* display, including *Template Gothic, Dead History* and *FF Beowolf* (Antonelli & Carmody, 2011).

Template Gothic, developed by Barry Deck in 1990, is a significant milestone in the history of digital typefaces not just because of its widespread use but also because of the designer's distinct voice and the source of inspiration he used — a sign displayed in his local laundry. By the end of the 1990s, *Template Gothic* had become widespread, expressing the

aesthetic of imperfection favoured by several designers throughout the grunge era (Figure 2.26.). Like other typographers, Deck expressed a wish to relinquish modernist letterform perfection (Deck, 2011; Lupton, 2010).

Also, in 1990, P. Scott Makela created *Dead History* (Figure 2.27.) typeface when digital techniques were beginning to gain traction as commonplace tools for visual and communication designers. The past often influences present designers, especially in typography; yet, Makela's design labelled history as "dead". He started his *Dead History* design by combining two pre-existing digital fonts — the Linotype's *Centennial* and Adobe's *VAG Rounded* — to create something new and surprising, using the sampling method frequently used in contemporary art and music composition (Lupton, 2010; Makela, 2011).

In 1991, a similar approach was used by Max Kisman for designing the *FF Fudoni* typeface (Figure 2.28.). This display font was created by chopping sections of two of the world's most popular typefaces: *Bodoni* and *Futura* (FontShop, n.d.b).

FIGURE 2.26.
Template Gothic,
Barry Deck, 1990.

FIGURE 2.27.
Dead History,
P. Scott Makela, 1990.

FIGURE 2.28.
FF Fudoni,
Max Kisman, 1991.

Wood and metal letterforms wore down unevenly in the pre-digital age, resulting in randomised variances, even slight ones; truly consistent letterforms were only achievable with digitised type. In 1990, Just van Rossum and Erik van Blokland created *FF Beowolf* (Figure 2.29.). This typeface restored the uncertainty in typography by including a randomisation mechanism in its code that causes its forms to change shape each time they are printed, ensuring that no letter looks the same again. They achieved this effect by messing with the programming commands in the PostScript code. They replaced *lineto* and *curveto* with their own command *freakto*, which, when printed, designed letters with randomly generated outlines (Van Blokland & Van Rossum, 2011).

FIGURE 2.29.
FF Beowolf,
Just van Rossum and
Erik van Blokland, 1991.

This shaken, distraught appearance caused an initial outrage among the typographers and graphic designers but was soon used in posters, CD covers, headlines and others. Through time, new printer drivers and

operating systems learned to ignore the PostScript code, and for a while, *FF Beowolf* was thought to be doomed, but OpenType (OT) brought new hope. However, unlike their predecessors, these OT fonts do not change form in the printer. Instead, they utilise a type of pre-programmed randomness where each glyph in each font has ten possible forms. The *FF Beowolf* OpenType fonts are available in four different weights. When typing in any macOS or Windows program that supports OpenType, the randomness of each letter appears on the screen (FontShop, n.d.a).

Tobias Frere-Jones's typeface *Reactor* (Figure 2.30.), was discussed in one of FUSE's eighteen issues (mentioned in section 2.1.). This typeface is an example of how the FUSE's effort anticipated some of our worries about the letterform beyond its traditional printable restrictions. As you type, each subsequent character gradually fuzzes up prior characters with visual noise (Lewis & Nadeau, 2010).

FIGURE 2.30.
Reactor, Tobias Frere-Jones, 1993.

The *History* type system began in the early 1990s, but due to technological limitations, the project was paused. In 2002, Bil'ak was asked to participate in an international competition to create a font for the Twin Cities. Instead of proposing a single new typeface for St. Paul and Minneapolis, he proposed a typeface system based on typography's progression, a conceptual typeface that utilised existing typefaces. *History*, the typeface created, was linked to the computer's calendar, and when selected, a predetermined database of typefaces would be used to offer a different font every day (Bil'ak, 2010).

In 2008, Bil'ak launched *History* (Figure 2.31.), which features twenty-one layers, twenty-one separate fonts that share widths and other metric information so that they may be recombined, all based on a *skeleton* of Roman inscriptional capitals (Bil'ak, 2010).

FIGURE 2.31.
History, Peter Bil'ak, 1990.

When the idea of cultural progress is supplanted by technological progress, the less obvious motifs of type design such as continuity or self-awareness are neglected.

(Bil'ak, 2010)

The superimposition of layers can construct an unlimited number of distinct styles. However, combining all layers can be a daunting text. To approach this problem, Bil'ak also supplied a *History* Remixer Application. This web-based program processed text input using an interface that allowed users to manipulate layers and open them as PDF files. This program was decommissioned in 2015 and replaced with online *History* layer previewing options (Bil'ak, 2008).

While careless use might produce bizarre outcomes, more deliberate experimentation can provide entertaining and unexpectedly fresh and functional typeface examples. *History* allows to duplicate typographic history or, on the other side, to expand it (Figure 2.32.) (Bil'ak, 2010).

FIGURE 2.32.
Everybody Dance Now, exhibition at the AIGA in NY, 2009.

In 2008, the Mainz University of Applied Sciences held a *Generative Typography* course oriented by Philipp Pape and Florian Jenett that explored conceptual design and media informatics. Using the *Processing* programming language, the students created interactive systems, resulting in experimental typefaces such as *Bastard*, *Blast*, *Broken Grid*, *Elien*, *Irratio*, *PDE Lubanoise*, *Pong* and *Zwirn* (Pape & Florian, n.d.).

Bastard was developed with *Processing* using the *Geomerative* library (Figure 2.33.) (Marxer, n.d.). In the same way, *Dead History* and *FF Fudoni* were produced by combining parts of multiple types; this typeface was

FIGURE 2.33.
Bastard font, Tobias Tschense, 2008.

developed similarly. Tobias Tschense, its creator, added another dimension to this concept, requiring flawless transitions between the fragments of each letter while maintaining their individuality (Tschense, n.d.).

Addressing the interactive component of the course, Denis Klein developed *Blast*, a typeface that reacts to sound. The analysis of the sound characteristics (bass, mids and treble) influences the appearance of each letterform, acting as a way for data visualisation (Figure 2.34.)(Klein, n.d.).

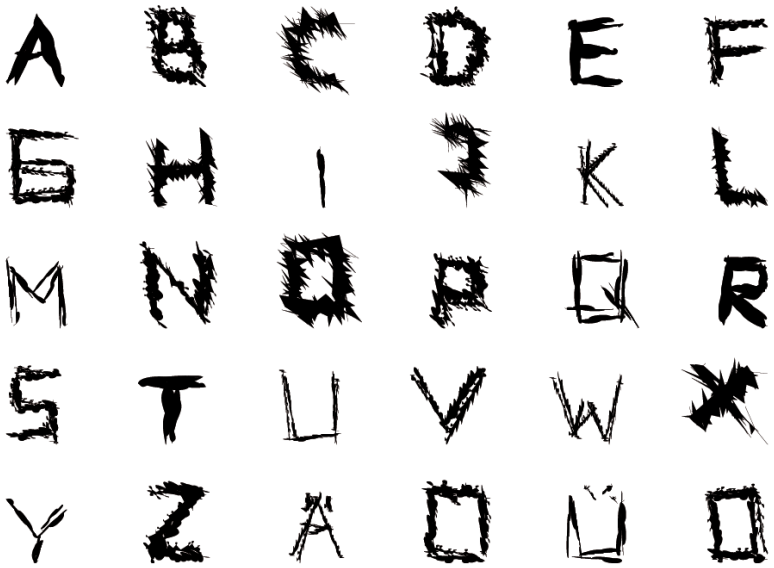


FIGURE 2.34.
Blast font, Denis Klein, 2008.

PDF Lubanoise (Figure 2.35.), created by Il-Ho Jung, uses a pre-existing font, Lubilin. Each letterform is filled with dots and has a shifted movement. The system also allows the manipulation of different parameters, like dot size, noise value, dot space and font length (drag characteristic) (Jung, n.d.).

FIGURE 2.35.
PDF Lubanoise font, Il-Ho Jung, 2008.

The *Irratio* and *Broken Grid* typefaces have similar approaches; both use existing typefaces and Bezier Curves to create them. *Irratio*, created by Ingo Reinheimer, creates Bezier Curves using three points of the original font (Figure 2.36.). A Bezier curve is created from P1 to P3 with handle points on P2. Then the anchor point is slid further, creating a Bezier curve from P2 to P4 with handle points on P3, and so on, running through all the anchor points (Reinheimer, n.d.).

Broken Grid by Nils Holland-Cunz, uses anchor points of the original typeface to create Bezier curves, creating a more abstract representation of each letterform (Figure 2.37.) (Holland-Cunz, n.d.).

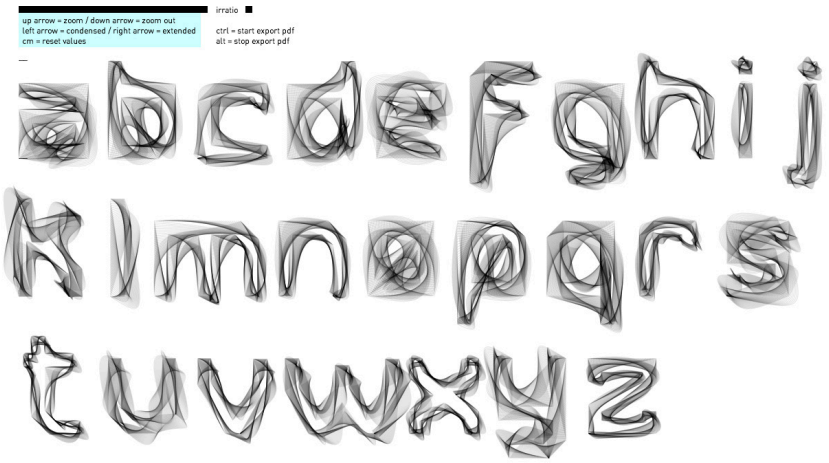


FIGURE 2.36.
Irratio font,
Ingo Reinheimer,
2008.

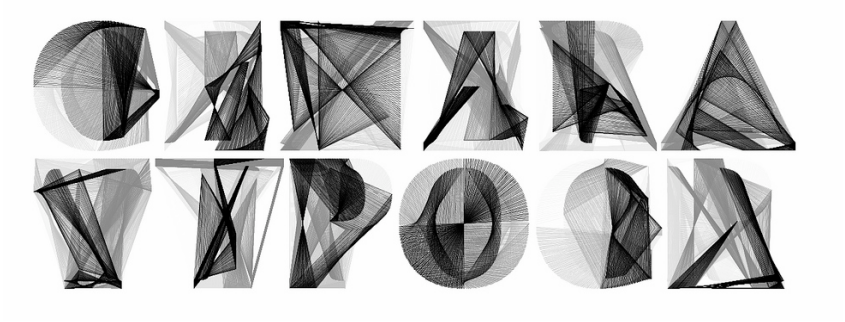


FIGURE 2.37.
Broken Grid font,
Nils Holland-Cunz,
2008.

Unlike the above, Lisa Reimann created *Zwirn*, which experiments with straight lines. Because it may be used with any font in the *True-Type* format, *Zwirn* is a sort of font add-on. It connects all the letters of a word by putting invisible points on the writing's outline and connecting them with lines. The look of the typeface is influenced by several parameters that the user can adjust independently (Figure 2.38.) (Reimann, n.d.).

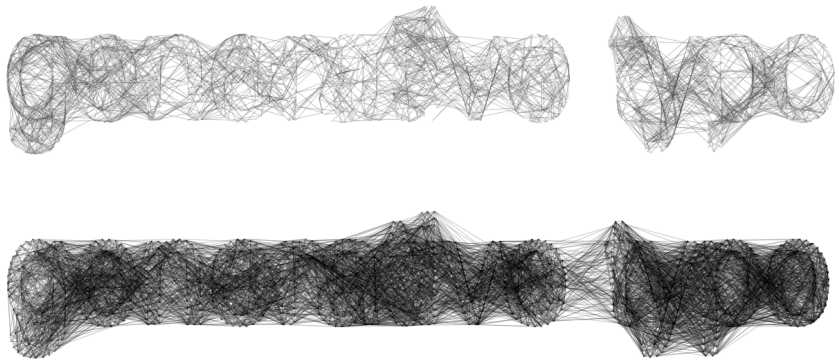


FIGURE 2.38.
Zwirm font,
Lisa Reimann,
2008.

Other typefaces created in the *Generative Typography* course were *Pong* and *Elien*. The first, created by Kersten Stahl, is based on the metaphor of a ball bouncing off the borders of a letterform while its path is drawn with a line. These lines may be presented in a variety of ways, including different thicknesses and colours, as well as curves. *Pong* edged and *Pong* rounded are the two fundamental font types. The font's name is a nod to the computer game of the same name, which follows a similar idea (Figure 2.39.) (Kersten, n.d.).

FIGURE 2.39.
Pong font,
Kersten Stahl,
2008.

Elien is a generative monospace typeface made using metaballs created by Tatevik Aghabayan. The metaballs are objects that may include up to five layers of circles that are nested and produce transitions when the space between them is small. A random principle determines the size of the metaballs and their circles. Connections are made between the circles that appear on the same level. The metaballs are formed along the *skeleton* line of each letter, resulting in novel forms between the individual metaballs and nearby letters. The system includes controls that experiment with display parameters like spacing (distance between letters and the number of links between neighbouring letters), density (number of metaballs in the *skeleton*), contrast (difference in metaball sizes) and levels (number of circles in each metaball) (Figure 2.40.) (Aghabayan, n.d.).

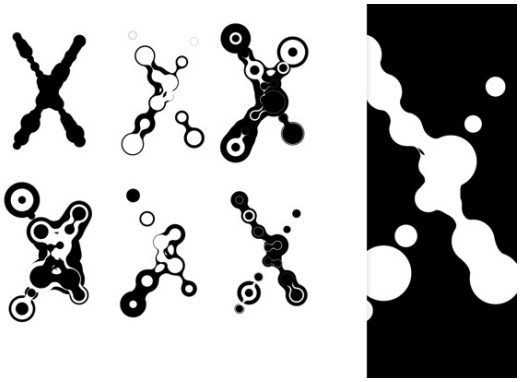


FIGURE 2.40.
Elien font,
Tatevik Aghabayan,
2008.

MuirMcNeil was founded in 2009 by Paul McNeil and Hamish Muir with the goal of investigating systematic and algorithmic approaches in type design, graphic design, and moving images (MuirMcNeil, n.d.a). Two Point is a monospaced geometric type system that tackles progressive permutations in the generation of typographic forms. It allows for the creation of a wide range of visual outputs and is inspired by early dot matrices and LED display lettering. This type system comes in four styles and seven weights, ranging from Extra Light to Black. Without changing any positions, the weight is increased. Instead of thinning or thickening the stroke width, the sizes of individual dots inside each letter alter progressively (Figure 2.41.) (MuirMcNeil, n.d.f).

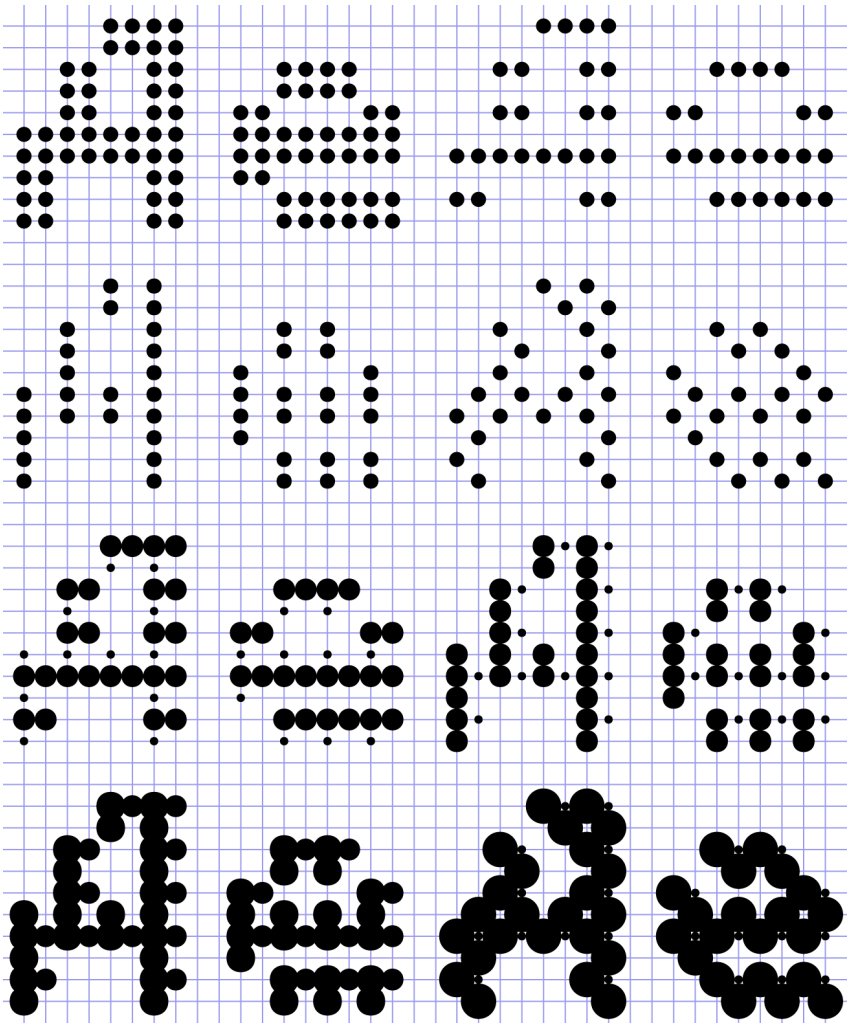


FIGURE 2.41.
TwoPoint,
MuirMcNeil.

TwoPlus and *TwoBit* (Figure 2.42.–2.43), have a shared grid with the *TwoPoint* typefaces. This grid determines positioning for both letter and spaces. As with other typefaces developed by MuirMcNeil, each letter operates as a variable component within differential visual systems. In *TwoPlus*, seven monospaced type collections comprise sixty-two typefaces, and the *TwoBit* type system offers five styles, each with nine weights. The *TwoPlus* font also features a set of rectangular background panels in corresponding grid patterns (MuirMcNeil, n.d.e).

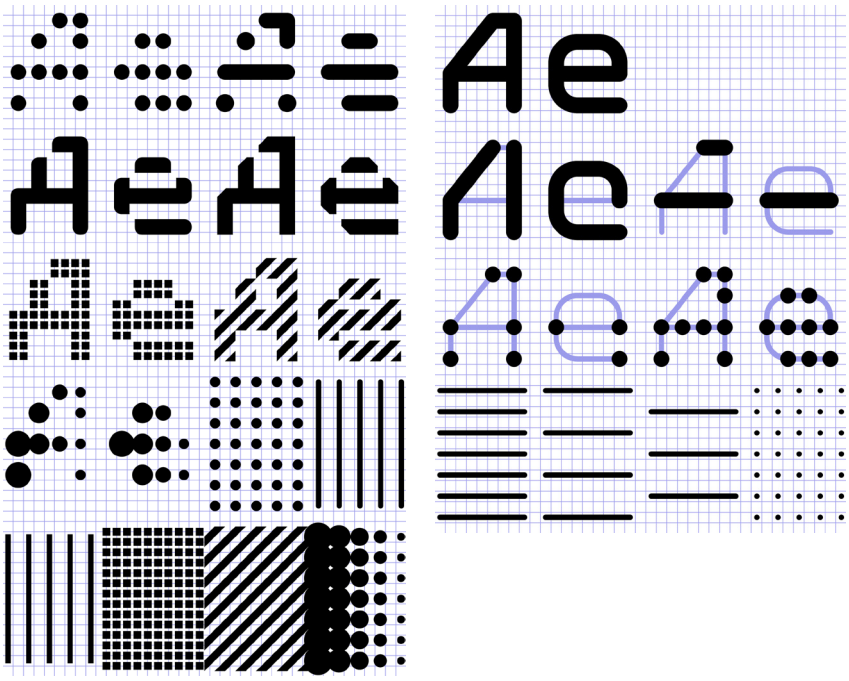


FIGURE 2.42.
TwoPlus,
MuirMcNeil (left).

FIGURE 2.43.
TwoBit,
MuirMcNeil (right).

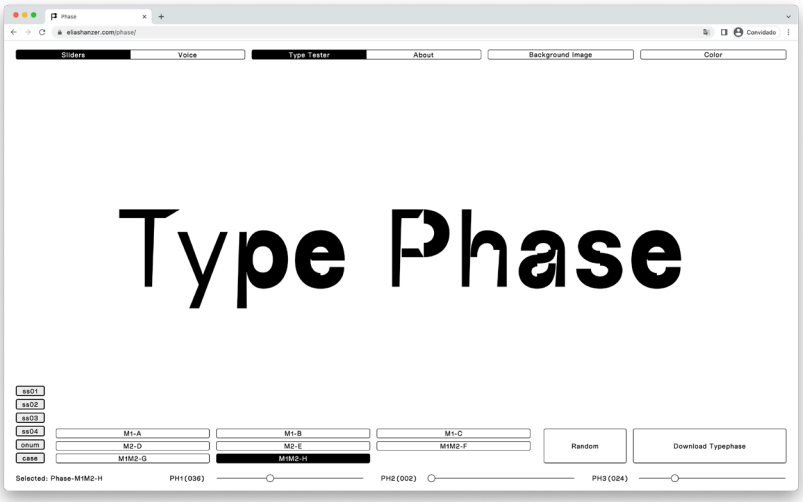
MuirMcNeil created these three different type system, designed to interact and overlap with one another and with each other, offering a vast range of visual possibilities. Users can apply specified styles to letterforms and background panels using page layout, bitmap, or vector design tools, either in precisely interlocked layers, using outlines, tints, colours, textures, patterns, and transparencies as needed. When working with print and fixed media, these characteristics open a world of possibilities (Figure 2.43)(MuirMcNeil, n.d.d).

FIGURE 2.44.
TypeCon – Fluid Identity,
MuirMcNeil, 2016.

Phase is a web interface that explores the concept of generative type. It is constructed methodically using modular components that serve as the foundation for an endless variety of forms. Thanks to variable font technology, each module can be parameterised in real-time or with sound

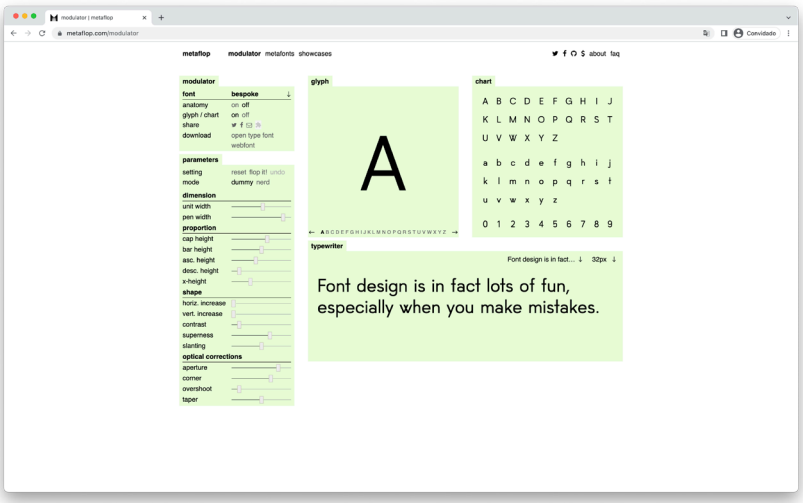
via voice input and exported the font as a TTF (Figure 2.44) (Hanzer & Zia, n.d.). The interactive aspect of this project brings closer to the public the infinite possibilities of type, however, the interface can be unclear and for some frustrating. This problem could be easily fixed by denominating each parameter according to the changes it creates on the letterforms.

FIGURE 2.45.
Phase,
Elias Hanzer
e Floria Zia.



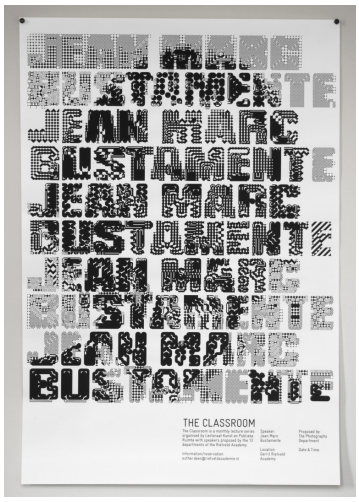
Similar to the project above, *Metaflop* is a web system created by Marco Müller e Alexis Reigel. It allows the user to choose a predefined font and parametrise each anatomy feature of the letterforms, like dimension, proportion, shape and optical corrections. They use the language Metafont to create these transformations, and the system enables exporting the created font as an OTF or WOFF file (Figure 2.45)(Reigel & Müller, n.d.).

FIGURE 2.46.
Metaflop,
Marco Müller
e Alexis Reigel.



In 2002, Jürg Lehni launched an *Adobe Illustrator* plug-in called Scriptographer. It allowed users to enhance Illustrator's capability by using JavaScript. From this plug-in, different projects were created, such as the Tile Tool by Jonathan Puckey, which draws previously created

tiles along a selected path (Puckey, 2006). In 2007, Jonathan Puckey compiled the best Tile Set designs that he had created and designed the exclusive Special Box Typography (Figure 2.47 - 2.49) (Puckey, 2007). Through time, updating the script to the constant updates of *Adobe Illustrator* became unsustainable, so a new framework emerged called Paper.js, which runs on top of HTML5 Canvas (Lehni, 2012).



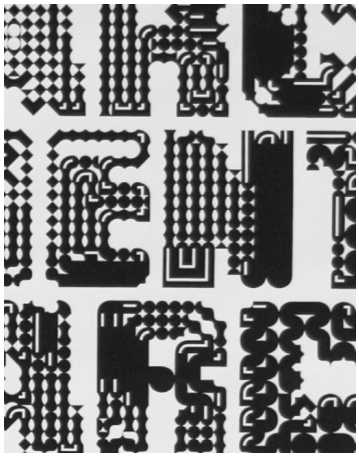


FIGURE 2.47.
First poster design using *Tile Tool,* Jonathan Puckey, 2006 (left).

FIGURE 2.48.
The Classroom, Jonathan Puckey, 2006 (centre).

FIGURE 2.49.
Detail from Figure 2.48., Jonathan Puckey, 2006 (right).

In 2015, Leon Butler designed the *Generative Sans* font (Figure 2.50). It all began with a Processing sketch that replaced the vector data with a randomised *for* of eclipses with a free sans serif typeface. These produced forms may be adjusted and regenerated to develop a form that seems like it would give the best geometry while also being very legible. The shapes included in the final sample later represented the general character of the font, existing in a plastic state that was both mailable and reactive, with the aid of the *Geomerative* library. Additionally, ligatures would be used to link characters in lengthy text strings (Butler, 2015).

FIGURE 2.50.
Generative Sans Font, Leon Buttler, 2015.

In 2019, the FF3300 design studio used the *Generatore Tipografico di Libertà* (GTL), a python library for the creation of generative fonts (XYZ2018 Lab, 2019/2021), to create an dynamic identity system for

Nòva (Cultural centre and youth community). The GTL library allowed them to not only create letterforms but also to create textures and quarters of explosions that relate to the energy and unpredictability of the client (Figure 2.51) (FF3300, 2019). The GTL library, created by Daniele Capo created a flexible and simple syntax (.0#), which creates endless variations for the same letter structure using only elementary geometric elements, a circle and square.

FIGURE 2.51.
Nòva,
FF3300.

Also, in 2019, Catarina Maças, David Palma and Artur Rebelo developed a generative system that created a typeface that adapts its shape to sentiments expressed in a text. An application was created which accepts text as input and by using an external JavaScript library called Synesketch it extract sthe emotions present in the text, based on Paul Ekman's research. For every analysed text, Synesketch returns the weight values for each emotions, the valence, and a general weight value. Later, it morphs the typographic glyphs used in each word to represent the corresponding emotion. Then a composition is made with the text and with the resulting glyphs. Finally, a typeface is generated, prevailing emotion described in the text (Figure 2.52) (Maçãs, Palma, & Rebelo, 2019).

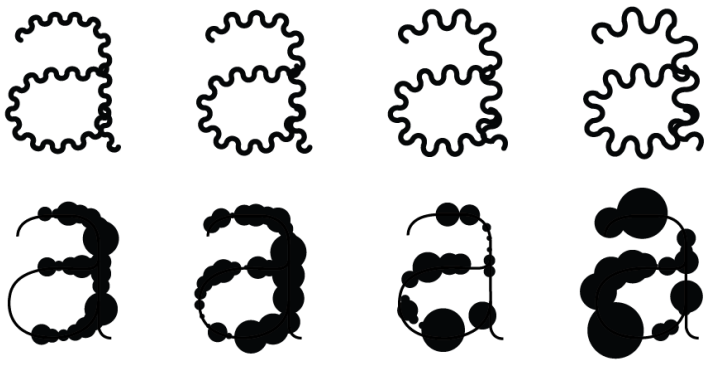


FIGURE 2.52.
Two approaches for the design of '*a*' of positive valences, TypEm, 2019.

The German design studio *Schultzschultz* was founded by Marc Schütz and Ole Schulte in 2007. The studio is renowned for its distinctive visual styles. In 2022, they released *Touch Type* (Figure 2.53), a free-to-use browser-based letter creation tool on their webpage. They started

experimenting at the studio with programming their tiny experimental tools for amusement and as a means of getting out of their creative ruts. This interface allows the creation of larger, thinner, broader and bolder letterforms by recognising the contact of the fingertips of the user. With a simple interface, Schultzschultz opened the door to immediate and accessible exploration of type.

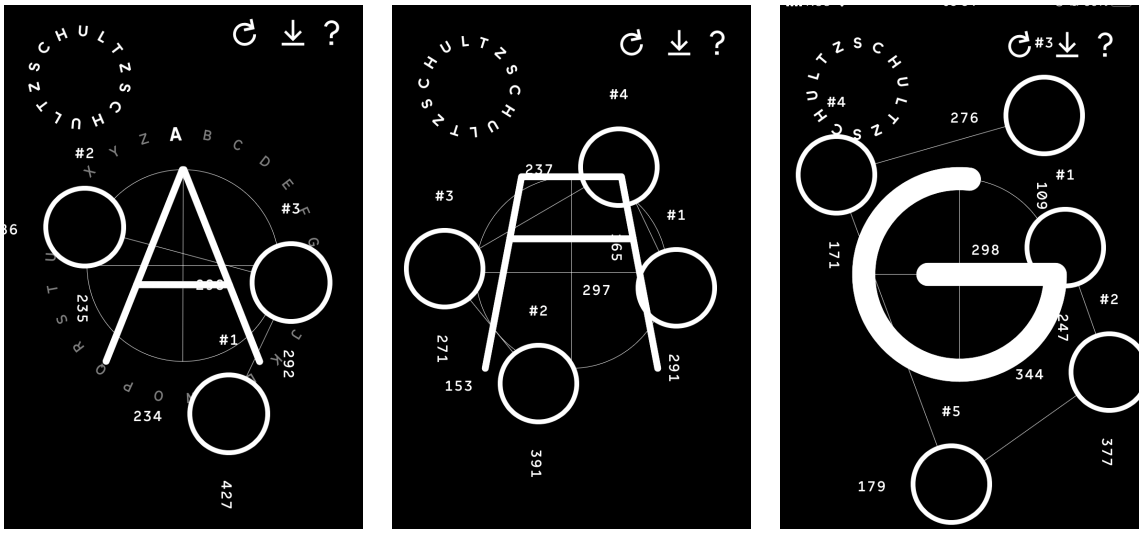


FIGURE 2.53. Touch Type, *Schultzschultz*, 2022.

All the curated work described and detailed above helped to idealise and conceptualise the next stage and chapter of this dissertation. Understanding the background and process of such projects, brought to the surface new ideas for instance, the extraction of anatomical parts, the use of layers and the creation of dynamic systems that allow the letterforms to adapt to different contexts.

3. PROJECT SPECIFICATIONS

A suggested strategy for the dissertation's practical component is presented in this chapter. We begin by discussing the concept and primary objectives of the system we aim to build before moving on to the development plan for this dissertation. At last, we focus on the preliminary work, which marks the initial experimentations and the starting point for the practical component of this dissertation. In addition, that section will look at the problems faced in this initial phase and the respective solutions.

3.1. CONCEPTUALISATION

The practical project of this dissertation revolves around developing a web-based system capable of creating letterforms. One of the main objectives is to explore the capability of typography to convey meaning beyond the written word and how its typographic body can adapt to different mediums besides colour. Therefore, this system designs each letter with a base structure (*skeleton*) that is a foundation for other visual components. The body of each letterform is filled with different textures or filling techniques created with the crops of a source image. The system receives three inputs, (i) the typographic skeleton as a TXT file, (ii) the input image and (iii) the texturizing/filling techniques (filling method and parameterisation). These different approaches create several letterforms able to convey visual interest in themselves. In addition, we aimed to prove the power of texturizing a letter's body by showing how different filling methods can create such disparate results even though using the same skeleton. Further, we set out to explore how these outputs can be of visual interest when used in different design products/materials.

As mentioned, different skeletons were developed and became subject to the system. These skeletons consider the traditional rules of type design, such as the visual weights of letters and what their construction and design entail. However, a level of flexibility and adaptation of the rules might be needed to explore different possibilities in type creation. These skeletons are created with *adjustable* modules that, by suffering geometric transformations (scale, translation and rotation), can adapt and transform to distinct parts of different glyphs.

The system accepts different inputs, which impact the visual appearance of each letterform. The user can customise a set of parameters related to the chosen texture, thus covering a new world of possibilities and results. Additionally, the user can define different styles and parameter values for specific modules of the skeleton chosen. The parameter changes directly affect the letterforms, allowing the user to visualise the resulting glyphs in real-time and thus provide visual feedback regarding the impact of each parameter on the glyph design. The text written with the system letterforms can be edited, allowing the user to experiment with the same texture and parameters with different letters. Furthermore, each typeface can be exported as an SVG or PNG, allowing for its application in different contexts as an editable file or an image.

Rather than developing the user interface and the texturizing/filling method simultaneously as a single platform, we instead decided to develop them separately. First, we started by developing and exploring

the filling method, the core of the project, as a stand-alone framework. Afterwards, we developed and conceptualised the user interface, which allows anyone to intuitively control the so-called system in a webpage. Therefore, the practical project produces two main components: (i) a texturizing/filling method that runs entirely programmatically and can read, draw, and fill a letter skeleton; and (ii) the system, a publicly accessible website that incorporates a user interface around the prior method. This method was created using a Javascript library which allowed for an SVG renderer approach and was later combined with the system, which was employed with the usual programming languages most suited for the web (JavaScript, HTML, and CSS). The next chapter explains these technologies and processes and credits them whenever relevant.

3.2. DEVELOPMENT PLAN

This section provides an overview of the plan for the development of this dissertation. This plan is divided into different tasks considered crucial to achieving the objectives defined for this dissertation. In subsection 3.2.1, each task is briefly described and mapped into a Gantt Chart, which aims to guide and keep accountable the progress of this dissertation. At last, a methodology is chosen and explained in subsection 3.2.2.

3.2.1. TASKS

The development plan requires the identification of the different tasks required to accomplish the different objectives of the dissertation. This step is necessary to overview each task and the workload/time required.

1. WRITING DISSERTATION
This task entails documenting all the theoretical research and the development process of the practical component of this dissertation. It encapsulates all the research needed to create the State of the Art, detailed in chapter 2, essential to fundament the practical project and the procedure for gradually recording the development of the actual project, which is covered in the following chapter.

2. EXPERIMENTATION
The practical project could be addressed in multiple ways; hence this task allotted a brief period of time during which we prototyped and evaluated a potential final strategy for achieving the project's goals. Thus, it involved the development of the first and most basic filling technique by experimenting with different programming languages.

3. LETTER SKELETON CREATION
We created three skeletons throughout this project's development. There were times where these had to be rearranged and redrawn making this task certainly the most iterative. Thus, it involved drawing the primary skeleton and later developing the secondary skeletons.

4. SYNTAX CREATION AND DEVELOPMENT

This task entails developing a syntax that converts previously drawn skeletons into a grammar of rules described in a TXT file, which can be later computationally read and drawn. Therefore, changes to the letter skeleton correspond to needed modifications to this syntax, making the current task dependent on the previous one.

5. DEVELOPMENT OF THE TEXTURIZING/FILLING METHOD

As previously alluded to, this task covers the reading, drawing and filling of the skeleton detailed on the syntax. Also, it involves the iterative process of developing the collection of features and elements that make up the filling method, which allows the texturisation of the letter skeleton. Further, the span of time allocated for this task also encapsulates the creation of the different textures and filling techniques influenced by a set of parameters that implicate the end aspect of the letterforms.

6. SYSTEM DEVELOPMENT

After completing the previous task, we began conceptualising, designing, and implementing the system, a web-based platform that allows anyone to use the previously developed texturising method on the letter skeletons. This task covers designing a user interface and building the system, among other lesser tasks.

7. APPLICATIONS

There is a need to assess the applicability of this project in other contexts besides the academic. This iteration allows for possible corrections and improvements of specific aspects, serving as final proof of concept.

8. DISSIMINATION

This last task is devoted to sharing and exposing the project to the external world. Due to time restrictions and changes in the initial development plan, we were not abble to write and submit a paper for an international conference. However, we still intend to do so in the near future.

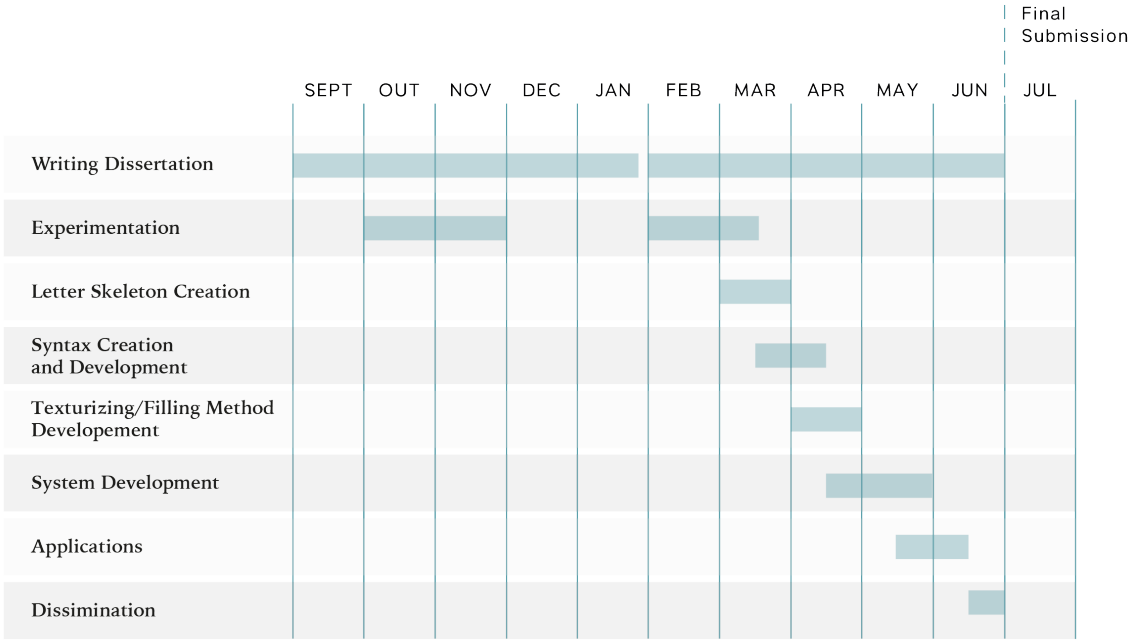


FIGURE 3.1.
Initial Gantt chart mapping each task with a prediction of its duration.

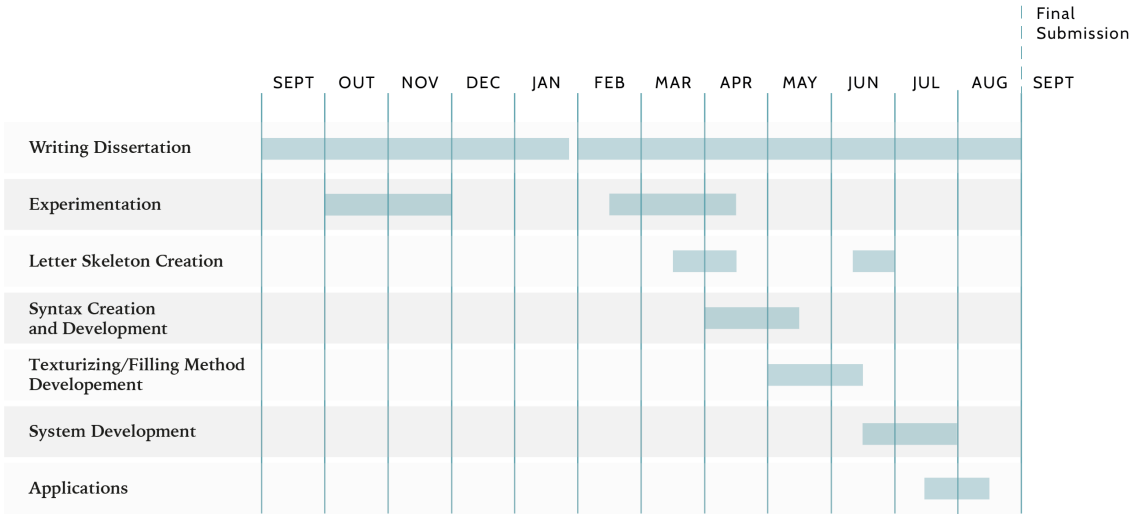


FIGURE 3.2.
Initial Gantt chart mapping each task with a prediction of its duration.

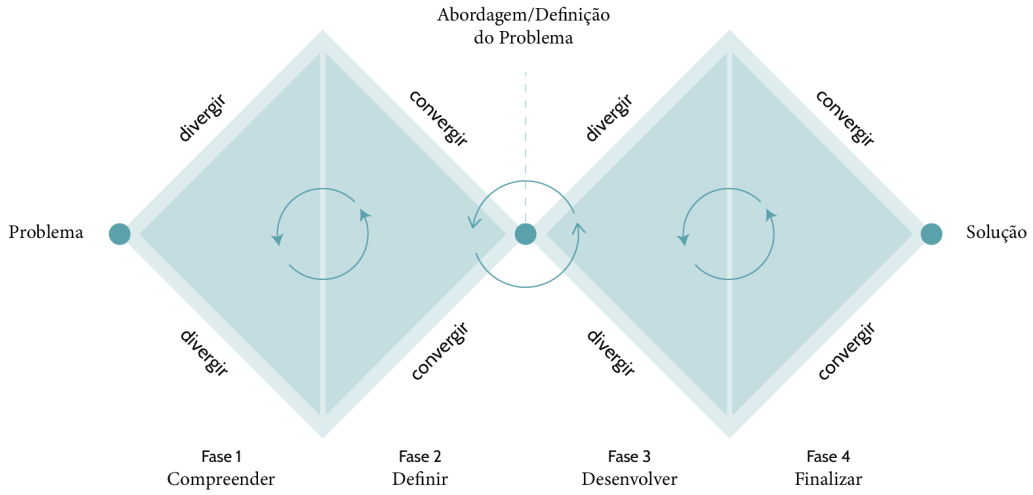
3.2.2. METHODOLOGY

Bearing in mind the experimental and iterative component of the practical project, we found the Double Diamond design process a more sustainable and suitable option for our methodology.

This model allows us to go back and forth between the different stages to fully comprehend the challenge and how it can be solved (Figure 3.3.). There is also a focus on *divergent* and *convergent thinking*, in which numerous ideas are generated first before being refined and narrowed down to a better solution. This paradigm happens twice: once to validate

FIGURE 3.3.
Double Diamond model.

the challenge ahead and again to produce the solution. Reflection and iteration are valued, creating an opportunity to improve the different outputs developed (Design Council UK, 2015).



4. PRACTICAL PROJECT

This chapter integrates all the work developed as part of the practical component of this dissertation. The chapter is divided into four main sections. It begins with an overview of the initial experiments made at the project's genesis and discusses its relevance to the project (section 4.1.). The second section details and explains the development process for the texturising method, a stand-alone framework responsible for filling and drawing the letters as intended (section 4.2.). This section is divided into two sub-sections, one where we demonstrate the first step in the process, manually drawing the letter skeletons, and another where we detail the implementation of the texturising method itself. The third section presents the end system, a public web platform that integrates a user interface with the aforementioned texturising method. This section encapsulates the UI (User Interface) developed and all the prototyping and implementation processes (section 4.3.). Finally, the last section demonstrates and analyses various applications and results from the developed system (section 4.4).

4.1. EARLY EXPERIMENTATIONS

Ideas and opportunities for developing the practical part of this project arose throughout the State of the Art research. In order to realise the potential of each idea, we needed to visualise these concepts, which helped to define this project course.

There was always a general idea of what we intended for the practical component of this dissertation; however, many aspects were still unknown. One of the initial ideas was to use images to texturise and fill the bodies of the letters. More specifically, using the repetition/drag of a crop of an image.

MANUAL DRAG

Eventually, this idea led to the creation of a simple Processing sketch that merely allowed the user to drag an image. This drag was drawn with mouse coordinates, and only two parameters could be adjusted. First, the size of the image used and second, the intervals between them, which depended on the acceleration of the mouse (Figure 4.1).

These early experiments gave us the first perspective on this preliminary idea. We could see how the manual aspect of this sketch (drawing the letters using the mouse) created room for a more calligraphic look in the project. The outcomes portrayed the *writing movement* made to create each character and allowed us to recognise this approach's dimensional and dynamic potential.

This first experiment helped prove this technique's plausibility, but it was still necessary to understand how it would be applied in a more geometric and non-manual letterform. The practical component of this dissertation does not aim to explore the calligraphic/manual side of typography thoroughly but to create original letters respecting the basic rules for type creation. Consequently, we needed to find a more geometric way of creating this letterform and experiment with the same technique.

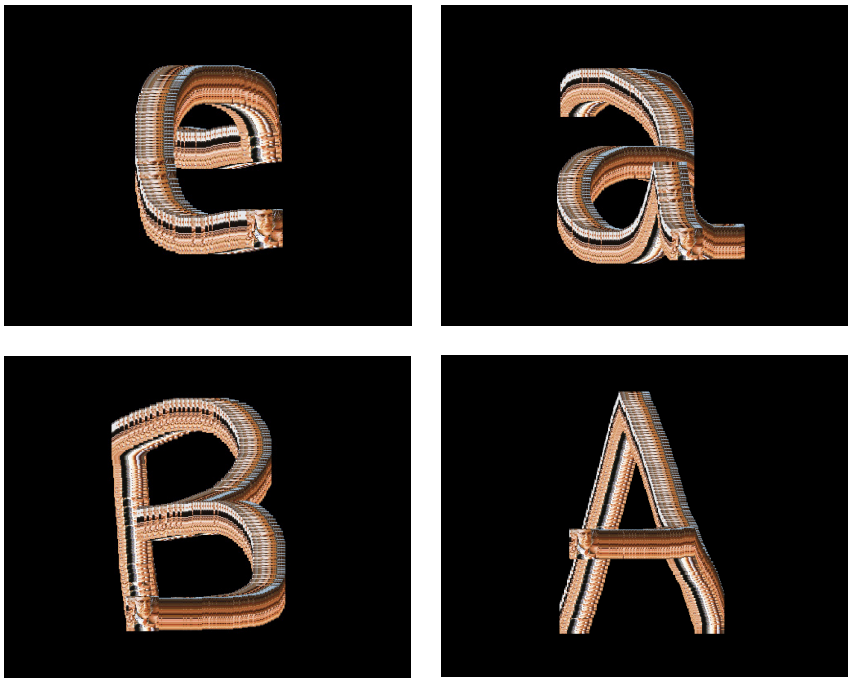


FIGURE 4.1.
First experiments with drag method.

DRAG ON A LINEAR SKELETON

In a more automated approach, we first needed to specify a base structure in which the letters could form. Also, using Processing, we started developing a system to create more geometric and clean letters.

First, it was necessary to create a grid that could serve as a base for the skeleton of each letter. The initial grid created had 8x8 cells, but it became too limiting after initial experiments. It did not allow for a flexible approach to type development, especially regarding the visual balance required in type design. Therefore, the final grid was 36x36 cells, allowing more flexibility and balance among letters (Figure 4.2.).

In order to simplify the drawing of each letter, we decided to exploit their modularity. Different modules were developed to optimise the creation of each letter and allowed to test the concept faster. This abstraction allowed us to create a system that joined the different parts of each letter and created a set of uppercase skeletons to which it was possible to apply the technique previously explored.

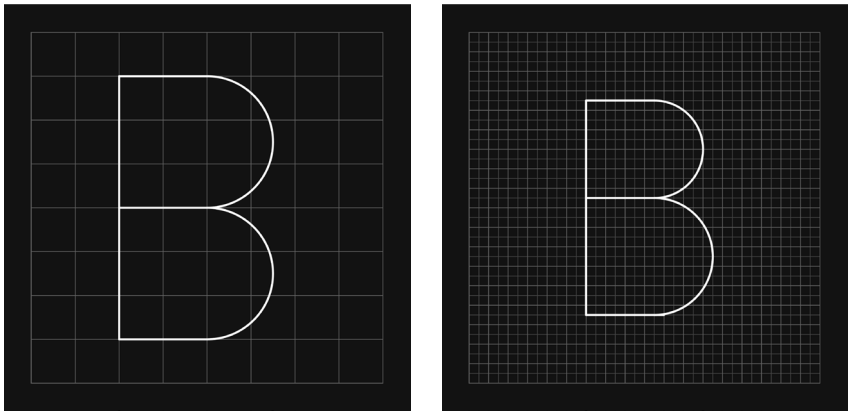


FIGURE 4.2.
B, initially created on the 8x8 grid (left). B on the final 36x36 grid (right).

The next step was to apply the drag of an image into the skeleton. Opposite to the first experiments, this system allowed the selection of different sections of an image where their position and dimensions could be adjusted (Figure 4.3.). In pursuance of a similar effect to the first experiment, the image section was repeated throughout each module segment and together created letterforms. We implemented two methods to fill the letter skeleton: quantity and spacing. The first method receives a specific number of sections to be repeated in a skeleton segment, while the second method fills each segment with the maximum possible sections equally spaced (Figure 4.4.).

The spacing method was chosen because it allowed each module to be more uniform if needed, especially in the curved modules (ellipses and arcs). However, this method also raised another problem. As shown in the diagram in Figure 4.4., the repetition of each section could end before the end of the respective module. If needed, the solution was to create a slight offset on the intervals between sections. Hence, the space among sections is slightly modified to position the first and last sections on the opposite ends of the module (path).

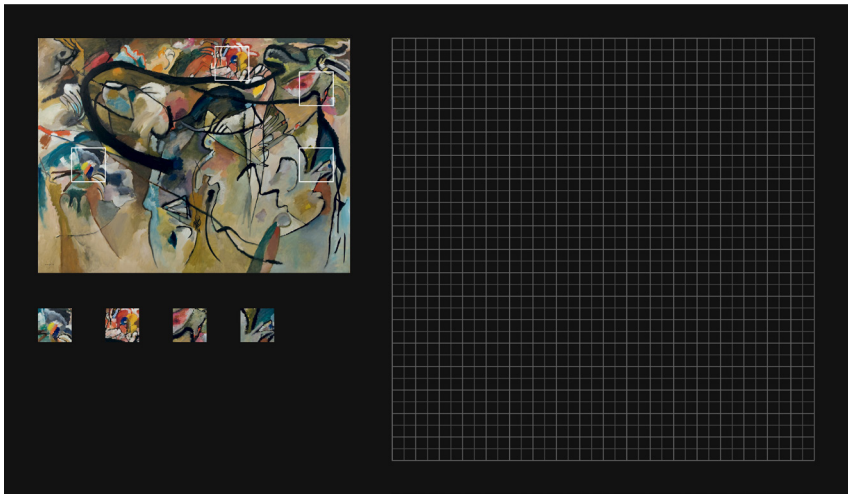


FIGURE 4.3.
Selection of different sections of an image.

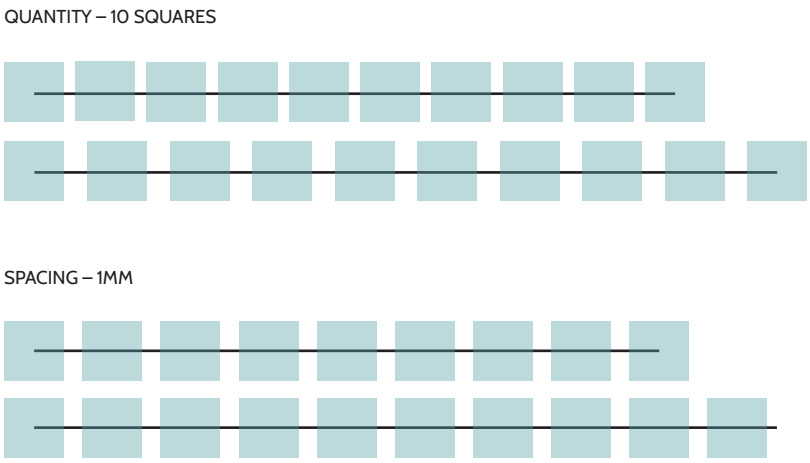


FIGURE 4.4.
Sketch of the difference between the two methods implemented to fill the letter skeleton: quantity and spacing.

With the skeleton and the dragging/filling method implemented, we started exploring some possibilities of this technique. The first experiment tampered with the intervals between sections in different letter modules, resulting in more dynamic but less uniform letters (Figure 4.5.).

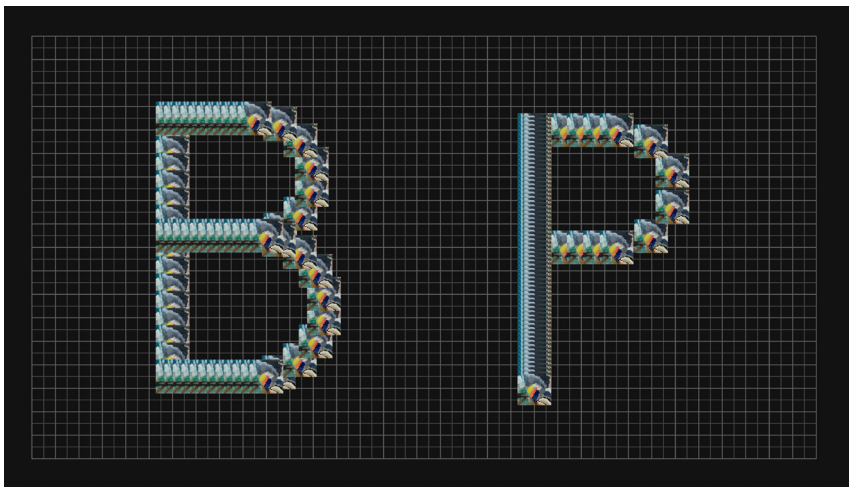


FIGURE 4.5.
Outputs with different intervals between sections, letters: B and P.

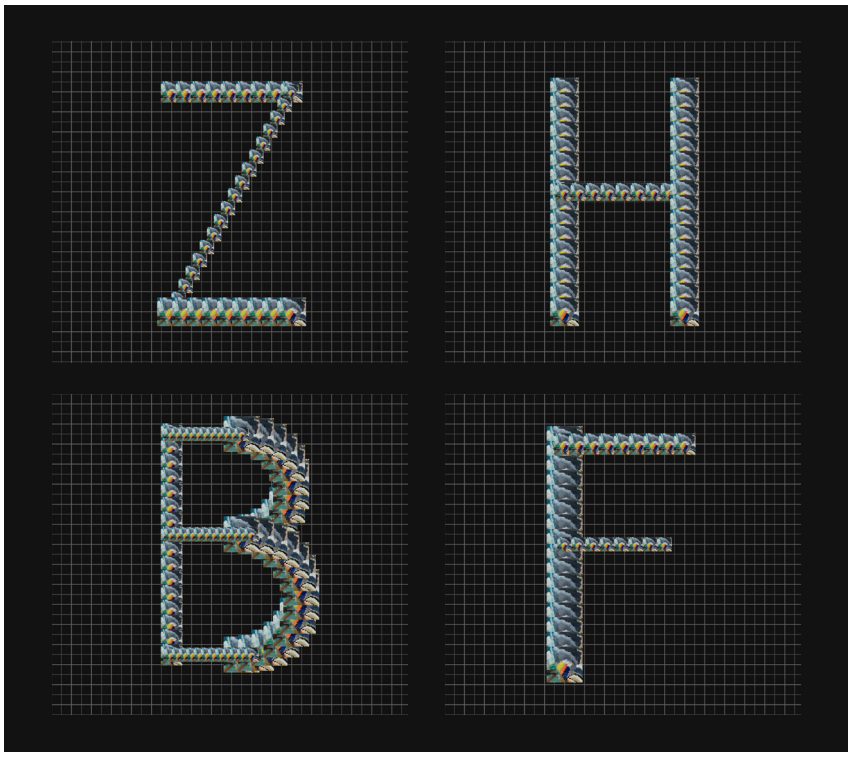


FIGURE 4.6.
Outputs with different sized sections, letters: Z, H, B and F.

Another possibility explored was increasing or decreasing the size of the sections. The results were a variety of outputs that created a dramatic difference between modules, whereas they can unify a letter when drawn in the same size (Figure 4.6.). In addition, another option explored focused on the simultaneous application of different sections of the source image through the skeleton. Like the last, this parameterisation re-enforces the modularity of the letterforms (Figure 4.7.).

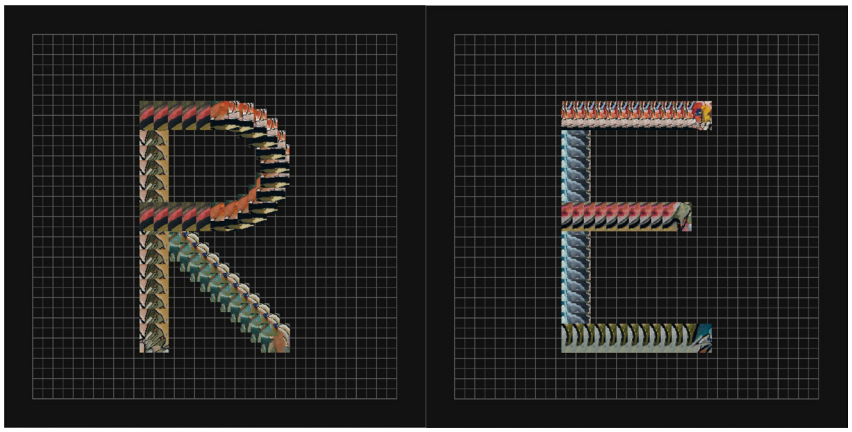


FIGURE 4.7.
Outputs with different sections, letters: R and E.

FIGURE 4.8.
Initial outputs with mixed techniques (next page).

We also had the idea to explore the possible outcomes of using different shapes to crop the source image by using elementary shapes such as a circle, triangle and rectangle to crop the source image. This experiment enhanced these techniques' *three-dimensional* potential. Other parameters and experiments were made, such as the rotation of the sections along the skeleton path.

There was one last experiment to be made in this sketch; we wanted to explore how the letter reacted if the source image was a live transmission. These experiments showed enormous potential in the animated/interactive aspect of the letterform. However, the sketch became much slower, even though we were using Processing which has a video library to capture and display live video. With all these different parameterisations and experiments, the potential of using images to fill the body of a letter became evident. Despite working alone, the possibilities explored also allow for joint experimentation (Figure 4.8.).

DISCUSSION

The conclusion of this subsection marks a decision on the development of this project. Because of these first experiments, we understood that creating modular letterforms in a standard grid was something to be pursued in the next phase of this practical project. First, modular letterforms allowed us to create different skeletons, and secondly, it uniformised the application method of the image's sections.

Furthermore, the experiments explained and demonstrated above underline the potential of texturising a letter's body and how typography can embrace different mediums, such as images.


BAK
NFP
KRB

4.2. TEXTURIZING METHOD

Since the start of this dissertation, we have wanted to create a system that could be easily accessed. We decided to develop a web-based system that could draw skeletons and apply different filling techniques to them. Besides, being a web page made the system outputs available, which helped with the project's future dissemination and exposure.

This section details the development process of manually creating the letter skeleton and how it was implemented and used as a base for filling techniques.

4.1.2. MANUAL DRAWING OF A MONOLIENAR TYPEFACE

When thinking of filling a letter, we inevitably have to begin by discussing where and how we will get its base typographic skeleton. Initially, we wondered about extracting skeletons of pre-existing typefaces; however, the early experiments described in the previous subsection proved that we needed access to all its points, dimensions and positions to apply the texture to a skeleton. Therefore, using a created modular skeleton as a base was more straightforward. Furthermore, we had control and, if needed, visual access to its points, which could come in handy to solve any particular implementation problem.

It is important to note that drawing a typeface requires ample typographic knowledge that can only be obtained by experience and practice. Moreover, letterforms are, ideally, drawn in a font design software. Although essential, we knew this was not the most crucial part of this dissertation but more a means to an end. Additionally, we sought to create a sans serif, monolinear and modular typeface because of its simplicity; we could avoid the learning curve and, consequently, the time required to use that type of software.

Note that modular typically describes a letter assembled from a limited palette of distinct elements, repeated, flipped but not scaled. However, the typeface created used geometric transformations, such as scale, to avoid a certain level of character abstraction, standard in modular typefaces. Additionally, they were drawn in *Adobe Illustrator* using Cheng's literature (Cheng, 2020) as a guide. We began by creating a simple 15x15 grid (Figure 4.9.)., which soon showed a lack of detail

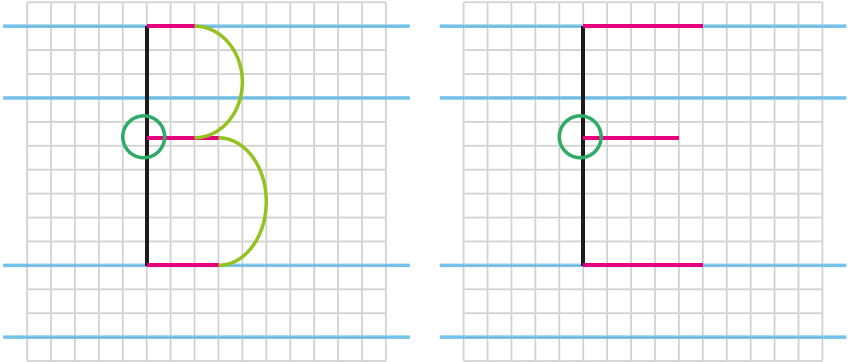


FIGURE 4.9.

B, E and F created with 15x15 cells grid (top).

and complexity needed for the visual centre of letters. Thus, we increased the grid cells to 60x30 and adjusted the anatomic measurements of the letter (ascender, x-height, baseline and descender). When adjusting each letter to the new metrics, we noticed that curved modules' centres and control points, such as arcs and ellipses, were not positioned on a grid coordinate (Figure 4.10). We understood that this could create feature problems in the implementation, so we once more increased the grid to its final aspect, a 64x64 grid (Figure 4.11.).

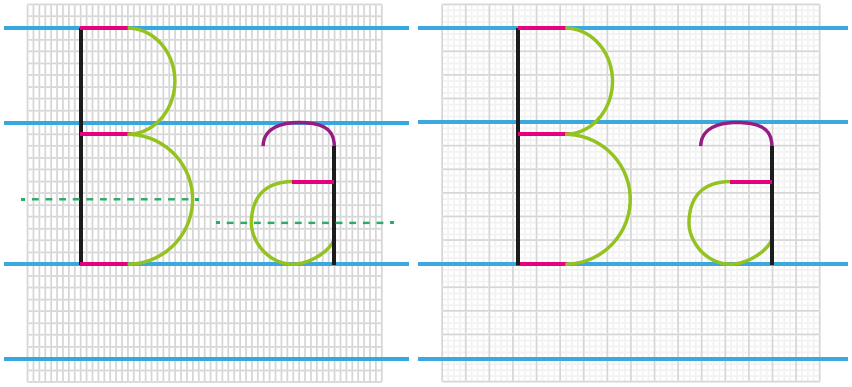


FIGURE 4.10.
Letter B and a in 60x30 grid (left).

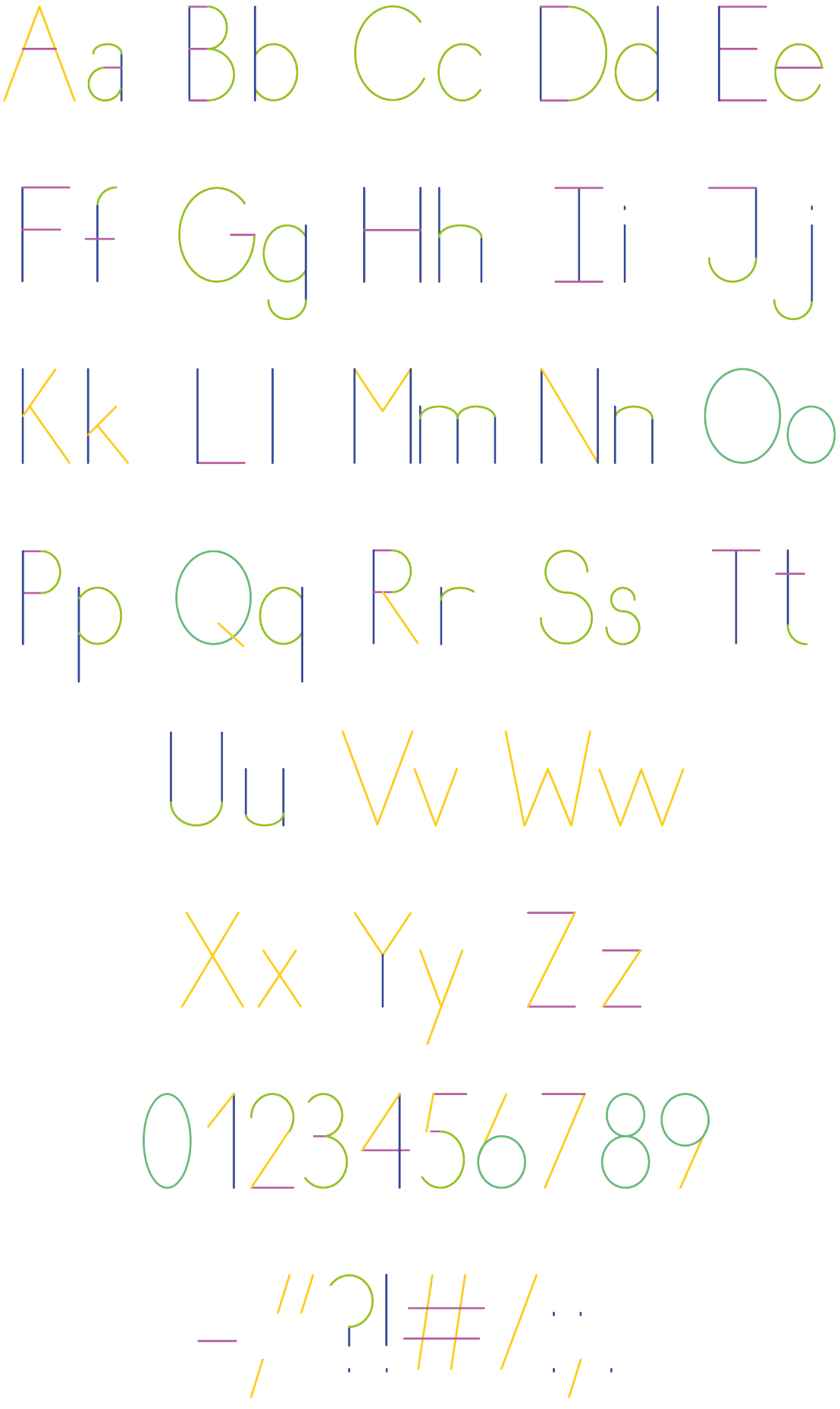
FIGURE 4.11.
Letter B and a in 60x60 grid (right).

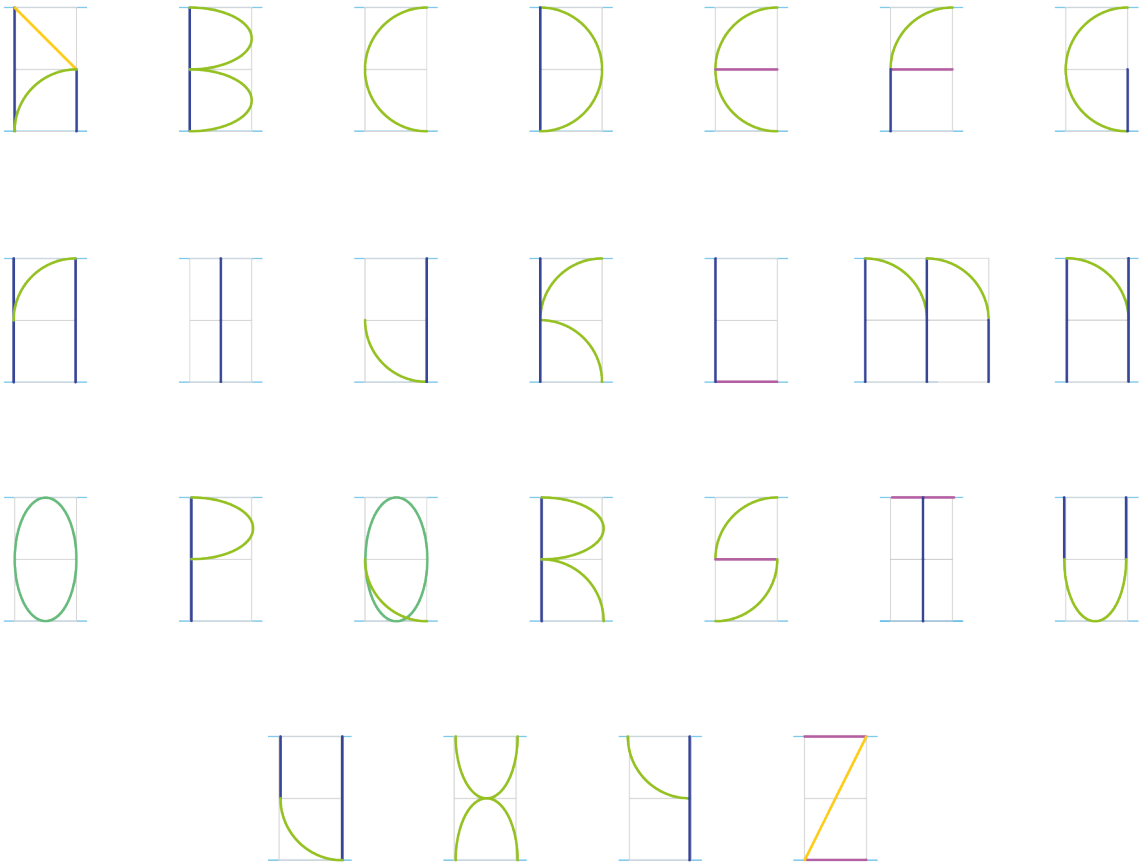
DEVELOPING OTHER TYPOGRAPHIC SKELETONS

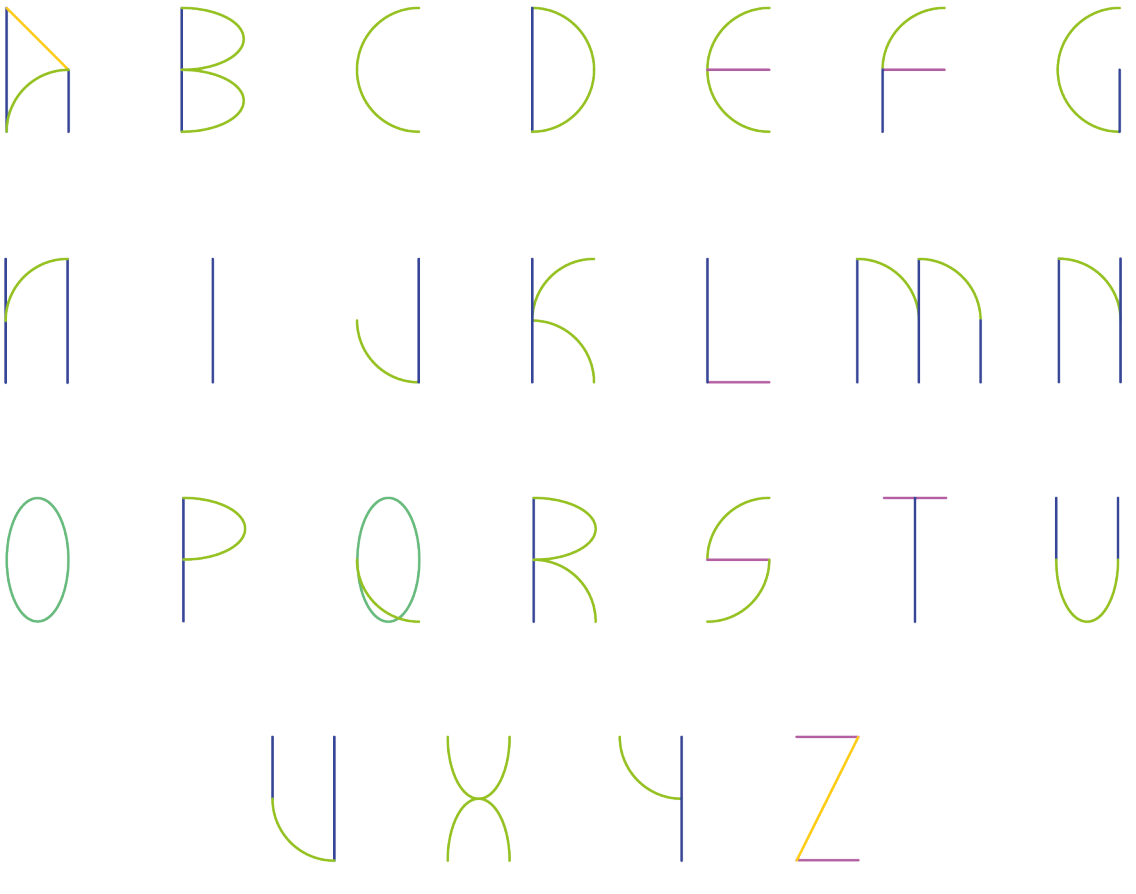
Throughout the development of this project, and besides the skeleton shown in Figure 4.12. (see next page), we also created two other skeletons, one more simple and the other more complex. The skeleton in Figure 4.13. uses a grid 4x4, and it is an all-caps skeleton. Contrary to the first monolinear skeleton, this one has a more abstract approach to type creation, and its construction was more similar to the traditional construction of modular typefaces. On the other hand, the skeleton in Figure 4.14. uses a grid similar to the first skeleton but has a serif-like module that stands out.

Note that these skeletons are not as complete as the first created, which contain lower and uppercase letters, symbols and numbers. The creation of these skeletons was not crucial to the project's development, unlike the skeleton detailed in the previous subsection. However, it allowed us to explore other possible outputs of the project, which are later discussed.

FIGURE 4.12.
First *skeleton* created with (left) and without the grid (right) (next page).


A a B b C c D d E e
F f G g H h I i J j
K k L l M m N n O o
P p Q q R r S s T t
U u V v W w
X x Y y Z z
0 1 2 3 4 5 6 7 8 9
_ , ' " ? ! # / : ;





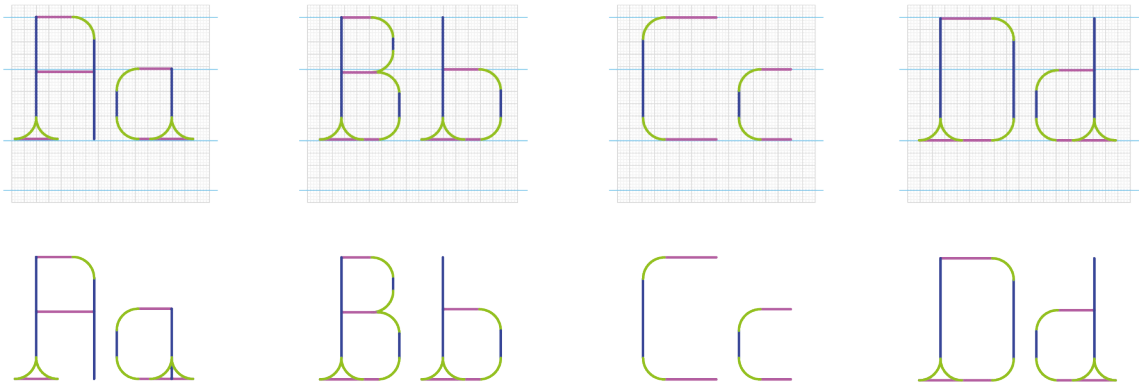


FIGURE 4.13.
Second all-caps skeleton created with and without the grid (left page).

FIGURE 4.14.
Last (serif-like) skeleton created with and without grid (top).

4.1.2. IMPLEMENTATION

In order to develop a web-based platform, we needed to choose what programming language we wanted for the project's core. Drawing letters into the web can happen in two rendering options: SVG (vector-based renderization) or Canvas2D (canvas-based renderization).

Various JavaScript libraries have either rendering options, but we focus on three of the most commonly used, *p5.js*[1], *paper.js*[2] and *Raphaël.js*[3]. The last two are vector base libraries, and the first is canvas based.

It made sense to draw the letter skeleton using a vector-based library since it allowed better manipulation of the individual modules. Additionally, it allowed for the exportation and manipulation of an SVG file. The *Raphaël.*js library had more documentation available than the paper.js library, making it the best choice to implement the base skeleton.

CREATION OF A SYNTAX

The first implementation step was to draw the monolinear typeface into a browser-based platform. To achieve this, we needed to find a way to program the letter by first reading its modules and later computationally drawing them. Our idea was to create a type of grammar/syntax with all characters detailed by their modules. This syntax would be written as a TXT file, and later it would be read and interpreted to create each letterform.

In order to correctly replicate the skeletons made, we had to take into consideration the metrics that were used in its manual version. These included the grid information (width and height and number of columns and rows) and, likewise, the anatomic measurements of the letter (ascender, x-height, baseline and descender). Therefore, the first information on the TXT file was precisely these values.

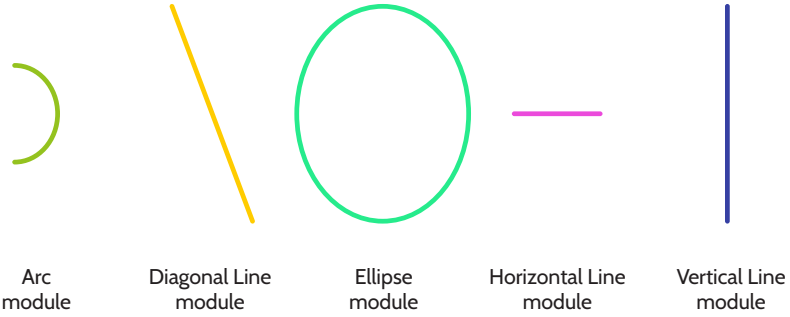
The next step was to analyse each letter, understand what modules could be identified, and develop a syntax for letter drawing. We had to analyse the chosen language and the type of values/information needed to create the module paths. Initially, we focused on upper case letters with only straight lines (A, E, F, H, I, K, L, M, N, T, V, W, X, Y, Z). Their modularity is more apparent than in lowercase letters making them faster to draw. Nonetheless, we found a need to iterate once more the skeleton and enhance the reusability of its modules (Figure 4.15).

[1] **p5.js**
https://p5js.org/

[2] **paper.js**
http://paperjs.org/

[3] **raphaël.js**
https://dmitrybaranovskiy.github.io/raphael/

FIGURE 4.15.
Names of modules types (primitives) present on *skeleton*.



The first version of the syntax had significant consideration of the *Raphaël*.js drawing functions, which referenced the SVG path string format[4]. For example, to draw a straight line *Raphaël*'s path command considers the start point as the origin. It then receives a parameter which indicates the path orientation (Horizontal *Lineto* (horizontal lines), Vertical *Lineto* (vertical lines) or *Lineto* (diagonal lines) (Figure 4.16.). Furthermore, to position the line in a specific grid coordinate, a Moveto command or a translation had to be used. Hence, the first version of the syntax used a more literal approach and was similar to the drawing process of the library (Figure 4.17).

FIGURE 4.16.
Raphaël's .path command (top).

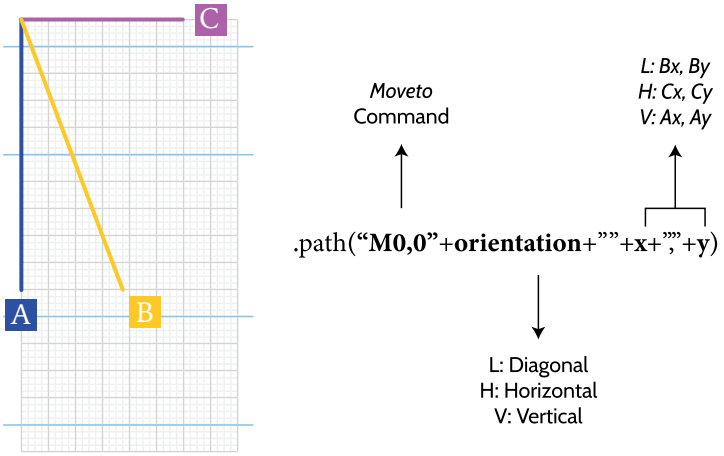
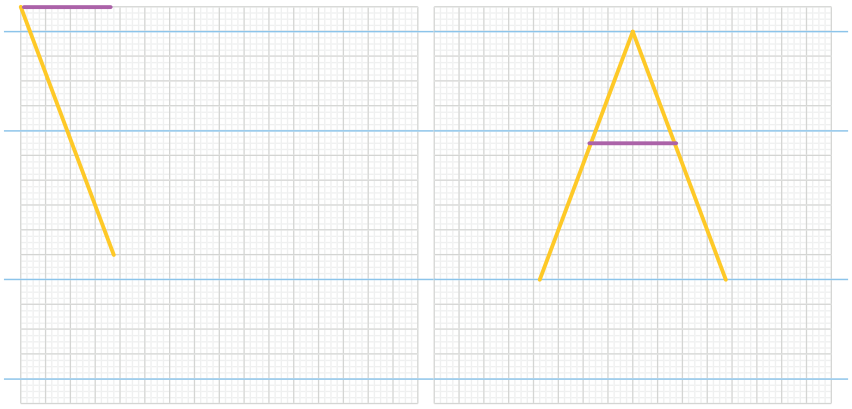


FIGURE 4.17.
Initial syntax for the uppercase A character, without (left) and with geometric transformations (right).



```
start_char_A
    module_line:L 15 40
    module_line:L 15 40
    module_line:H 14 0
end_char_A
```

```
start_char_A
    module_line:L 15 40 32 4
    module_line:L 15 40 17 4 s-1,1
    module_line:H 14 0 25 22
end_char_A
```

[4] **Paths SVG**
https://www.w3.org/TR/SVG/paths.html#PathData

Although it worked, we aimed to create a more general syntax that did not require previous knowledge of *Raphaël*'s language. Thus, we generalised and adapted it so that it could apply the information contained in the syntax to the library's language format.

The new syntax had three layers: characters (letters or symbols), modules (vertical line, horizontal line, etc) and primitives (line, arc and ellipse). The characters contain modules, and these, in turn, contain primitives. These concepts are better understood with the aid of Figure 4.18, which shows the detailed syntax for the 'L' character. The letter 'L' contains two modules: a vertical line (***module_vbar***) and a horizontal line (***module_hbar***). Each module contains a primitive line, the first horizontal and the last vertical (inside ***start_module [...] end_module_***).

As mentioned before, the typeface assembled was composed of identical modules; however, they suffered transformations, such as rotation, scale and translation. These transformations were detailed in the module reference inside the character specifications (***start_char_ [...] end_char_***).

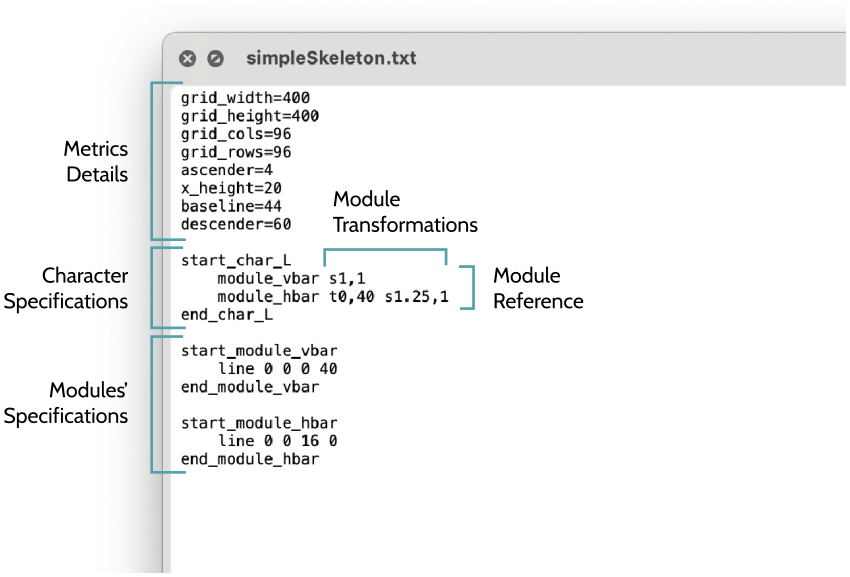


FIGURE 4.18.
Part of syntax with L character with module detail and transformations.

To draw each letter, we first had to add a keyboard input which received the letter/symbol typed by the user. Then, we created a recursive algorithm that reads all the information inside the TXT according to the glyph pressed.

This algorithm has three main functions, (I) ***read_char()*** — receives the key pressed name and the initial letter position (on the baseline). Loops through the content inside the character specifications part on the TXT (***start_char [...] end_char*** lines), and it sends each module or primitive name to the second function. (II) ***draw_part()*** — inspects if the content is a module or a primitive. If it is a module, the function runs again, recursively, to read the data inside the module specifications (***start_module [...] end_module_***). Otherwise, if the part is a primitive, the third function runs. (III) ***draw_primitive()*** — function draws the respective primitives with all the transformations defined.

For example, if the letter 'L' is pressed, the character specifications (i.e. its modules) are read. Then the module specifications are accessed (i.e. the module primitives), and finally, the primitives are drawn with all the transformations detailed in the module reference. This syntax allowed us to create letters composed of modules and primitives such as lines, ellipses or arcs that can suffer transformations so that they can be in the required position and dimension to assemble the various glyphs (Figure 4.19.).

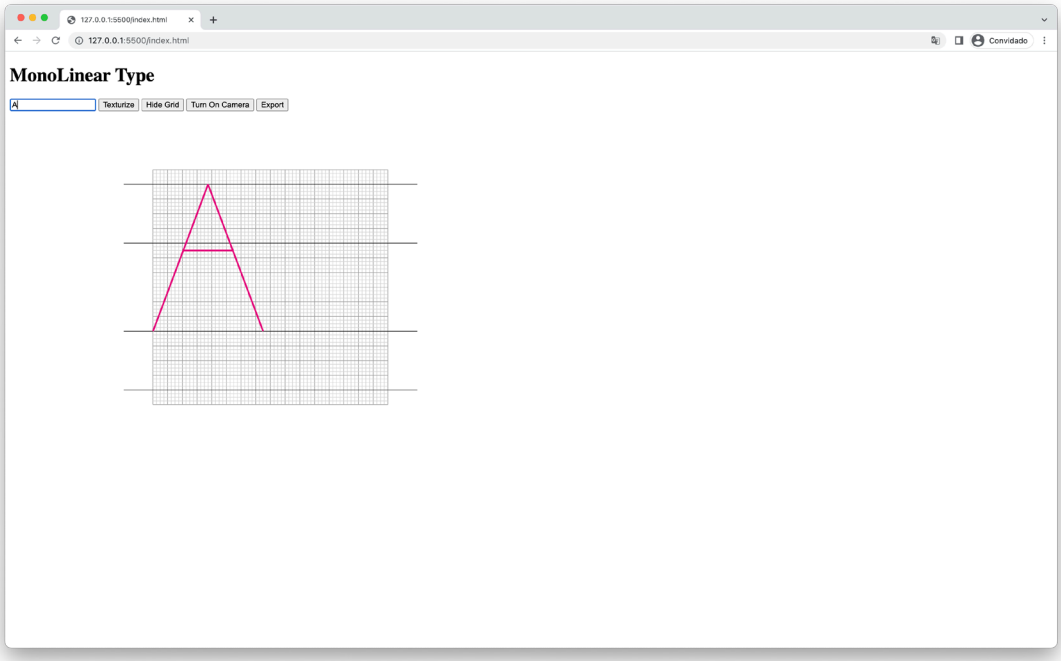


FIGURE 4.19.
System with *Raphaël*.js approach drawing the letter A.

A CHANGE OF PARADIGM

With the syntax implemented, the next step was to apply the texture to the skeleton and start exploring texture possibilities. Our initial idea was to read the SVG paths for the *skeleton* and apply the texture to them using the *Raphaël*.js library. However, this was soon dismissed because this library did not support cropping/clipping images and had limited possibilities for image manipulation. By this time, we had to take a step back and rethink our options.

We considered joining two different libraries, the *Raphaël*.js and the *p5*.js library. We tried to overlap the *Raphaël* canvas with the *p5*.js canvas; however, it was impossible to work with them while they overlapped each other. Hence, we tried to work with both libraries in parallel.

Despite seeming premature, we had to start thinking about how we wanted the user to interact with the web platform. Ideally, the user could write directly to the canvas where the letterforms appear, and if wanted, he could show the skeleton of the letterform. In addition, we needed to have options for image cropping.

We delve again into a discussion about the technologies used and what would be the best approach to solve the problems and options we were being faced with. We started exploring the possibility of adapting the

project to only use one library, in this case, *p5*.js. The only aspect that made us reluctant to discard the *Raphaël* portion of the project was its vector base characteristic and its ability to export the canvas as an SVG. In *p5*.js official documentation, there is no reference to an SVG renderer option; however, we found an SVG Runtime[5] for the *p5*'s API, which allowed for SVG exportation.

After some consideration and after accessing the pros and cons, we found that it was worth it to readjust the early implementation of the syntax and the reading/drawing algorithm to use the *p5*.js library. Note that the changes to be made were more significant on the drawing part of the algorithm, replacing the *Raphaël*.js primitive functions with the *p5*.js ones, such as ***line()***, ***ellipse()*** and ***arc()***. The part of reading the characters remained similar because they were made without the library. Therefore, this was the most viable approach because it did not require an outrageous amount of time, and did not devalue previously made work.

The first thing we needed to do was to create two graphic buffers with the native function ***createGraphics()*** from *p5*.js. This approach allowed us to create two different p5.rendered objects, one where the letterform would appear and the other with the source image and its sections.

Previously, when both *p5*.js and *Raphaël*.js were implemented, the letters were drawn with the texture in the ***draw()*** function of *p5*.js. This function continuously executes the lines of code inside, which made the project slower, even though the user could only type one letter at a time. Fortunately, applying the texturizing method outside the ***draw()*** function and only once in the buffer where the letterforms appeared was possible in this new approach and fixed the problem.

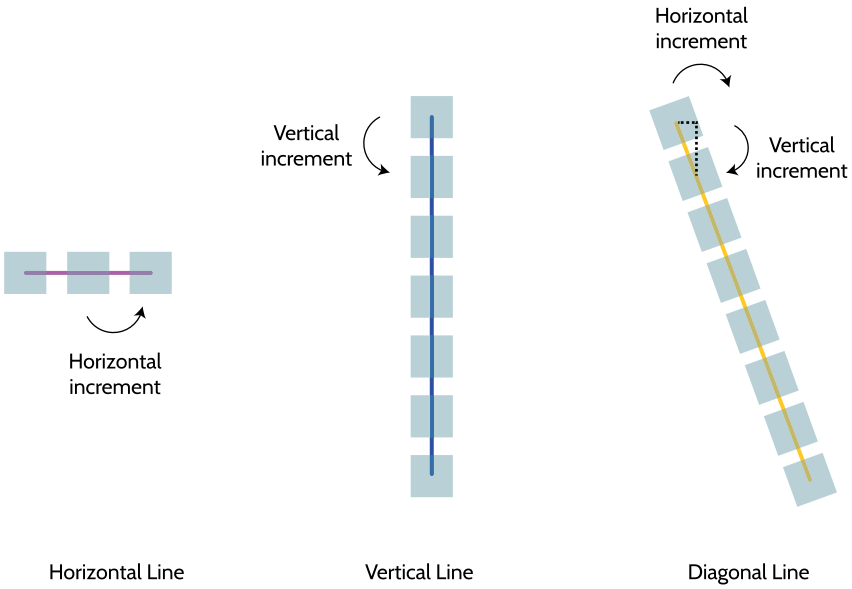
TEXTURIZING/FILLING METHOD

Although seeming like a step back, this paradigm change proved to be the best solution for the project. The similarity of *p5*.js and *Processing* allowed a logical comparison with the early experiment's method previously developed. We could identify the variables needed to create the new algorithm by analysing the function developed in the early experiment's method. Namely, the ***interval*** between the sections of the image, the ***start point (x,y)*** and the ***end point*** (x_1,y_1) of the module, the ***module type*** (arc, diagonal line, ellipse, horizontal line and vertical line), the ***length*** of the module and ***transformations*** needed (translation, scale and rotation). In addition, there were variables specific to curved modules, such as the ***start angle*** and ***end angle*** (arc module) and the ***width*** and ***height*** (ellipse module).

Applying the different sections of the image to the module paths is very straightforward in the case of straight lines. Given the ***interval*** value and the ***length***, we can calculate the number of images in a module segment. Consequently, the increment to distribute the sections along the module equals the length divided by the number of images. If the module type is a vertical line, then the increment affects only the y-axis coordinate; however, if horizontal, it affects the x-axis coordinate. Thus, in a diagonal line, the increment is added to both coordinates axis (Figure 4.20.).

[5] **SVG Runtime P5.js**
https://github.com/zenozeng/p5.js-svg

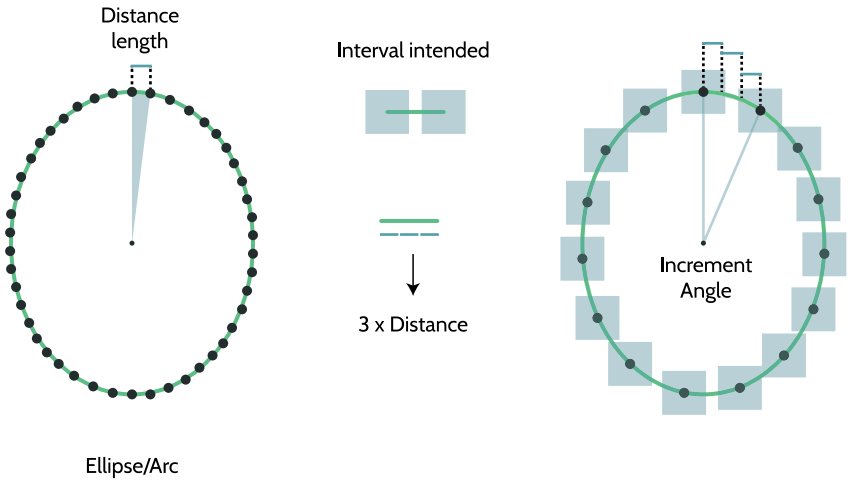
FIGURE 4.20.
Distributing/Incrementing image sections along straight lines modules.



Contrarily to straight lines, applying the section to an arc or ellipse is not as straightforward. First, calculating the length of an ellipse or arc is a more complex equation, especially in ellipses or arcs where the radius is not uniform. Secondly, the point where to draw the image section can only be calculated with trigonometry equations, and the increment is an angle value instead of a distance.

We had to make a point interpolation along the ellipse or arc to access its points. By dividing the ellipse or arc into one hundred points, we could calculate the distance between a consecutive pair of points. Then, we could calculate the number of points needed to skip to get the required interval and, thus, the angle increment needed (Figure 4.21.).

FIGURE 4.21.
Distributing/Incrementing image sections along Ellipse and Arc modules.



To have the different sections from the image, we needed to use the ***get()*** function from *p5*.js, which grabs a section of pixels from an image previously drawn on the canvas. Note that this required the image to be permanently displayed so its pixels could be copied. Later, this image section would be replicated and positioned in the different points that result from the previously described calculations.

Once this method was implemented, we could recreate the early experiments in a web-based platform. The same textures were developed, and even the live capture feature was implemented. However, live video is constantly updating, frame by frame, which requires the system to update the letter at the same frame rate as the video capture, making it too slow. Therefore, although it brought a level of dynamism to the letter, using live capture was inefficient and not viable for the project. It also compromised other crucial functionalities, such as the writing and drawing of the characters. Despite discarding this idea, we started creating more textures that could be applied to the letter skeleton.

TEXTURES

Implementing the filling method allowed us to start experimenting and creating different textures while using crops of images. Each texture gives typographic skeletons and glyphs a unique visual style. Furthermore, each filling approach contained a set of parameters that allowed us to manage various aspects of the texture. In this manner, we adjusted the variables to obtain various outcomes while in the same texture. It is important to note that by this stage, all this parameterisation was made internally and without the aid of UI (User Interface) components, like sliders or selectors. This approach allowed us to test and list a set of variables that impacted the end aspect of the letterforms.

A texture consists of a shape or a collection of shapes which create a *texture unit* that is later applied throughout the glyph skeleton by using the filling method. Different ouputs of the various textures combined with different set of parameters can be found in Appendix A.

BASIC

Initially developed to test the filling method, this first texture applies square crops of the source image into the glyph skeleton. Despite its simplicity, this texture received a set of parameters that could create a vast amount of different results. This texture eventually encouraged experimentation with other shapes of masks applied to the image sections, such as the circle and triangle.

CROSS STITCH

As the name implies, this texture was inspired by the cross stitch sewing technique. Two image sections were shrunk to create two rectangles that ultimately create a cross-like shape.

When applied to curved modules of the letter skeleton, this texture unveils a three-dimensional potential and visual dimension to the glyph. In addition, by using two different crops, there is a crossing among two different sections of the source image, making the letterform carry even more data of the image. However, due to the small details captured by the crops, this content is more abstract than the texture mentioned before.

BOOK STACK

Contrarily to the other textures, this texture uses all of the crops extracted from the source image and turns them into rectangular units. The section used in each texture unit is randomly selected, as well as its length. They are displayed in the letter skeleton, creating a chain of different sections that together, especially with a small interval value, remember a stack of book spines.

FIFTY FIFTY

This texture uses two different crops of the source image and applies them to the glyph skeleton. As the name suggests, the section used alongside the letter module changes in the middle. The more discrepant the crop of the source image, the more evident the texture becomes.

RANDOM SIZE AND OVERLAPPING

Both these textures use a similar approach. The first distributes the same crop image along the letter skeleton with random sizes, creating more horizontal or vertical texture units. The *Overlapping* texture uses the same technique of randomly choosing the crop image size, but differently from the *Random Size* Texture, it uses not one but four different sections. These textures add dimension to each glyph, specially the *Overlapping* texture, which explores and demonstrates a lot of colour and data from the source image.

EXPORTATION

Ideally, the letters would be exported as a TrueType (TTF) or an OpenType (OTF) font file. However, both types describe their glyph shapes by their outlines. OpenType provides various formats for colour fonts, one of which is the SVG table. Nevertheless, only vector graphics can be included, meaning that bitmapped images cannot be integrated (PeterCon, 2021). Therefore and unfortunately, trying to export the letterforms created as a font file is not yet possible.

Nonetheless, we still wanted to allow for the exportation of the output. A conceivable option was to export the output glyphs as an SVG. It allowed the output to be edited on any vector graphics software and expanded the already unlimited applications of this project output.

4.3. THE SYSTEM

This section consolidates and details the development and implementation process of the system (available at https://phototyper.dei.uc.pt/). Hence, this section delves into all the steps required to create and adapt the previous implementation into a system where the user can fully take advantage of the project. The section begins with a requirement specification stage (subsection 4.3.1.). Then, we demonstrate and explain the iterative process of developing and enhancing the user interface through the creation of wireframes (section 4.3.2.), which is followed by an interactive prototype (subsection 4.3.3.). Finally, we discuss and explain the implementation process, presenting the technologies employed, as well as the difficulties encountered and solutions found (subsection 4.3.4.).

4.3.1. USER INTERFACE CONCEPTUALISATION

By this stage of the project, we could begin conceptualising the user interface for the project. The next step demanded a requirement specification, where we analysed and outlined the essential parameters, interactions and features for the system and, finally, combined them with others already implemented. Part of this procedure was dedicated to fully understanding the parameters available for letter configuration and which ones could increase the output possibilities without creating unnecessary complexity to the UI. The following segments delve into the main aspects that need to be integrated into the user interface and what type of UI components could be used to fulfil their requirements.

SOURCE IMAGE

As mentioned, the texture applied to the letter comes from the crops of a source image. The source image is the base for customising the letter; different images create different outputs. Therefore, the user should be able to upload an image of his choice and change the different crops of the image, later used to fill the letters. In addition, the system should be prepared to receive horizontal or vertical images as input, becoming a fixed but flexible panel. This panel should contain the source image, the respective image crops (which could be modified) and an upload button.

CONFIGURATIONS

Regarding the configurations, we essentially had to look at all the variables/parameters which could be manually changed on the implemented code and understand how they could be adapted into a user-friendly UI component. The first parameters that came to mind were the skeleton and the texture type. A simple dropdown with all the options seemed viable since it did not occupy much space. Another parameter was the skeleton size, which could be added as a plus and minus button or as a slider.

Since the start, we aimed to allow the user to apply a specific texture and parameterisation to a module. Therefore, we needed to give the option to apply a set of configurations to a specific module or to every letter. If the user chooses to apply to a specific module, it activates a list of modules to choose from. Like the previous, a dropdown panel allowed this feature to be implemented. Note that adding a button was required to save and apply the configurations. In addition, a list of the modules and their saved configurations had to be accessible to the user.

By analysing the texture application algorithm, we could list other parameters that influenced the final aspect of a character, namely, the section size, the interval size and the offset value. The first parameter corresponds to the size of the image sections distributed throughout the letter, whereas the second value is the distance/interval between said sections. At last, the offset parameter adds a random offset value to the position of the image section along the skeleton. Initially, this value was made to be added to a specific texture, but we thought it could create exciting outputs if applied to the different textures. These three values should work best with sliders, allowing for a more intuitive manipulation by the end user.

Lastly, there was the section shape. This configuration started as three individual textures but created intricate outputs when applied to other pre-existing textures. The standard shape is a square, though the user could choose from other elementary shapes, such as a triangle and a circle. For this configuration, we decided to create a bullet checkbox.

WRITING AREA/LIVE PREVIEW

The interaction with the writing area/live preview should be straightforward, simple and clear. We wanted the user to write directly with the texturised letter. Therefore, we had to consider the interaction when the writing area was clicked, such as the writing cursor and other writing characteristics expected by the user, such as the Enter, Space and Return keys.

Additionally, if the letter did not update according to the parameter changes, then the configurations and the user actions would not be evident or effortless. Consequently, we knew that the letterform must directly react to all changes.

As mentioned previously, the letter is constructed using metrics and a base grid. These measurements created too much visual noise when writing, making it more confusing and cramped. However, it could be helpful for the user to show or hide those measurements and other details like the skeleton and the texture applied to the letter. A toggle button is the better UI component for this feature since it is commonly used for changing system settings or configurations, and it has two states, 0 and 1, in this case, show or hide.

OUTPUT LETTERS

This segment refers to the exporting options of the project. As mentioned, the letterforms could be exported as an SVG. However, after consideration, we thought it would be interesting also to allow exporting the letterform as a PNG file, thus, creating two ways for exportation, one more technical and editable and the other more direct and final.

4.3.2. WIREFRAMES

At this point, we had all the UI components we would need to create the platform but had not thought of the organisation and layout aspects of the interface. Hence, we began creating wireframes. The aim was to explore layout possibilities for the interface and understand how the functionalities could be displayed. These wireframes would serve as a foundation for creating the interactive prototype.

The interface has three main components: the preview/writing area, the source image and the configurations panel. The first corresponds to the canvas where the user would write, and the letterforms would be displayed. The second is a panel containing the source image, its sections, and the button to upload an image. In turn, the configurations panel houses all the parameters, features and configurations for letter customisation. There are also three other secondary panels, one for the export options, another containing the module's details and finally, and one containing the toggles for the writing area (Figure 4.22.).

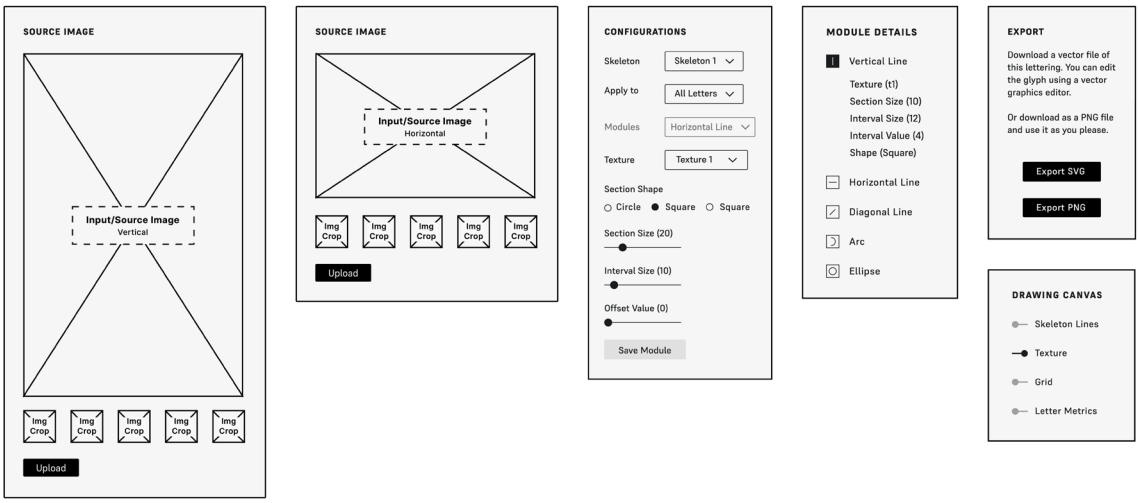


FIGURE 4.22.
Source image panel (vertical and horizontal) and the Configurations and the secondary panels: Module Details, Export and the Drawing Canvas panel (left to right).

The UI can be divided into two main areas: the preview/writing area and the panels' area where all the panels should be displayed (Figure 4.23.). When we started displaying them in the panel area, it quickly became clear that we needed more space for all the panels (Figure 4.24.). The solution found was to transform the secondary panels into closable panels, this way, they only occupied space if they were being used. Finally, we created the wireframes where we can see all the secondary panels closed with a vertical source image (Figure 4.25.) and with a horizontal image (Figure 4.26.) and the same, with all the panels opened (Figure 4.27. and Figure 4.28.).

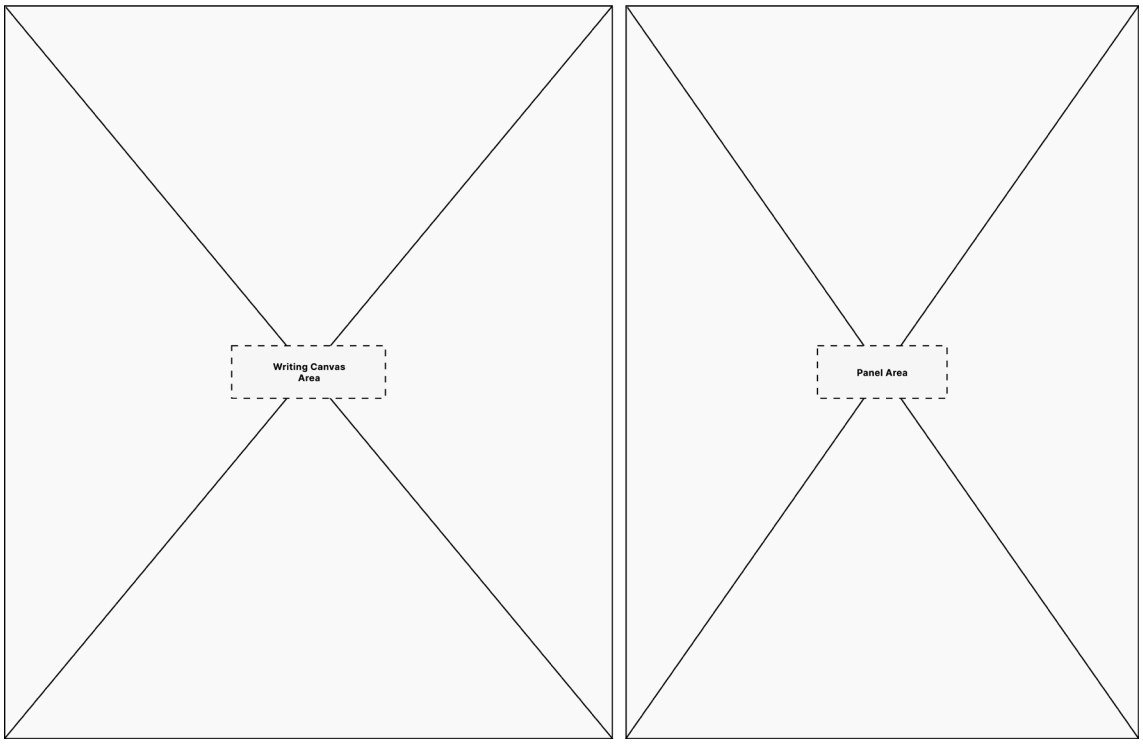


FIGURE 4.23.
Wireframe with closed secondary panels with a vertical source image.

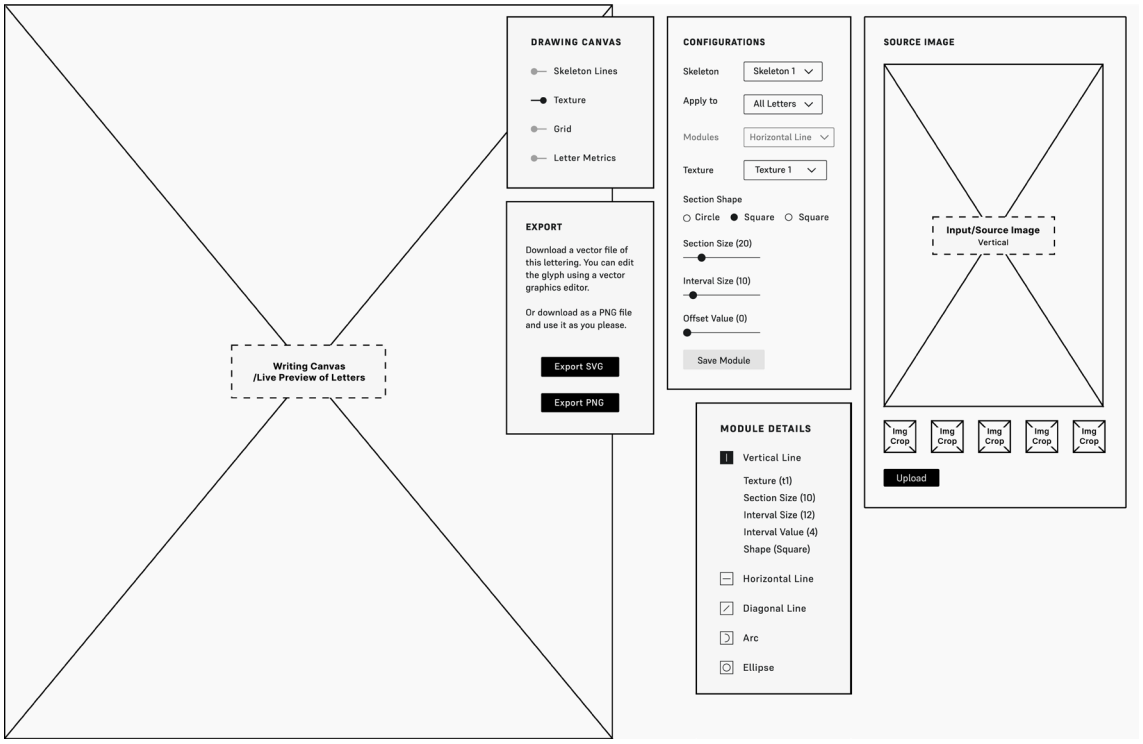


FIGURE 4.24.
Panels displayed onto Panel Area.

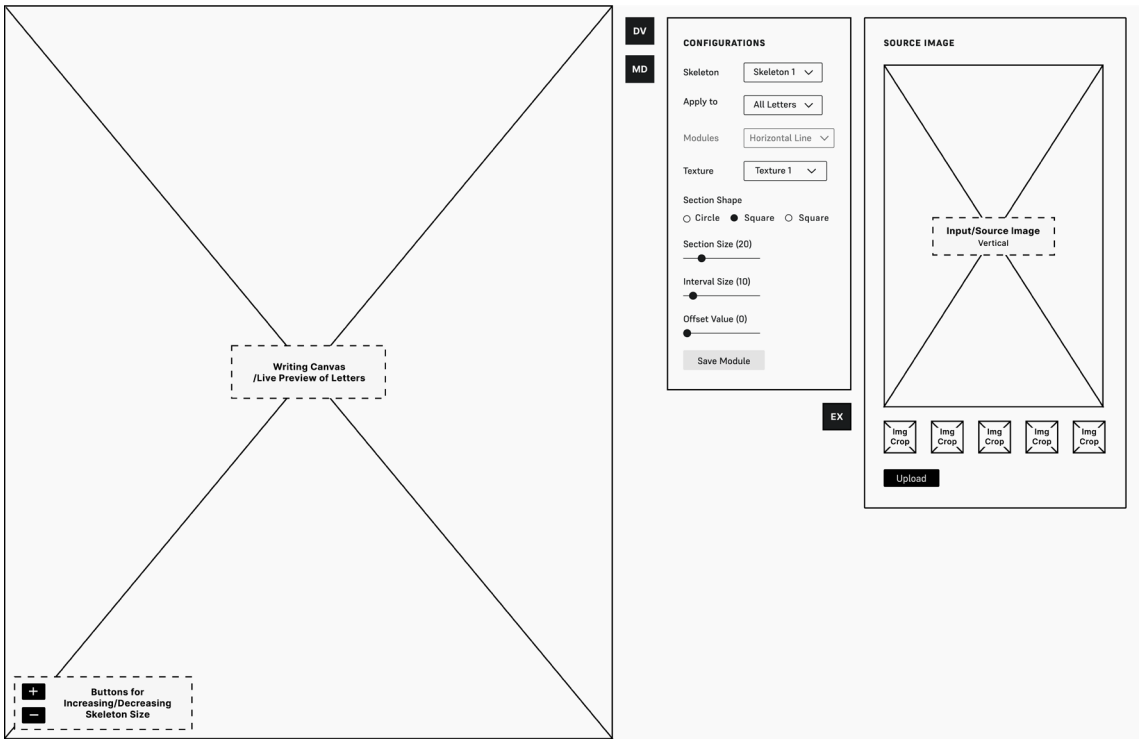


FIGURE 4.25.
Wireframe with closed secondary panels with a vertical source image.

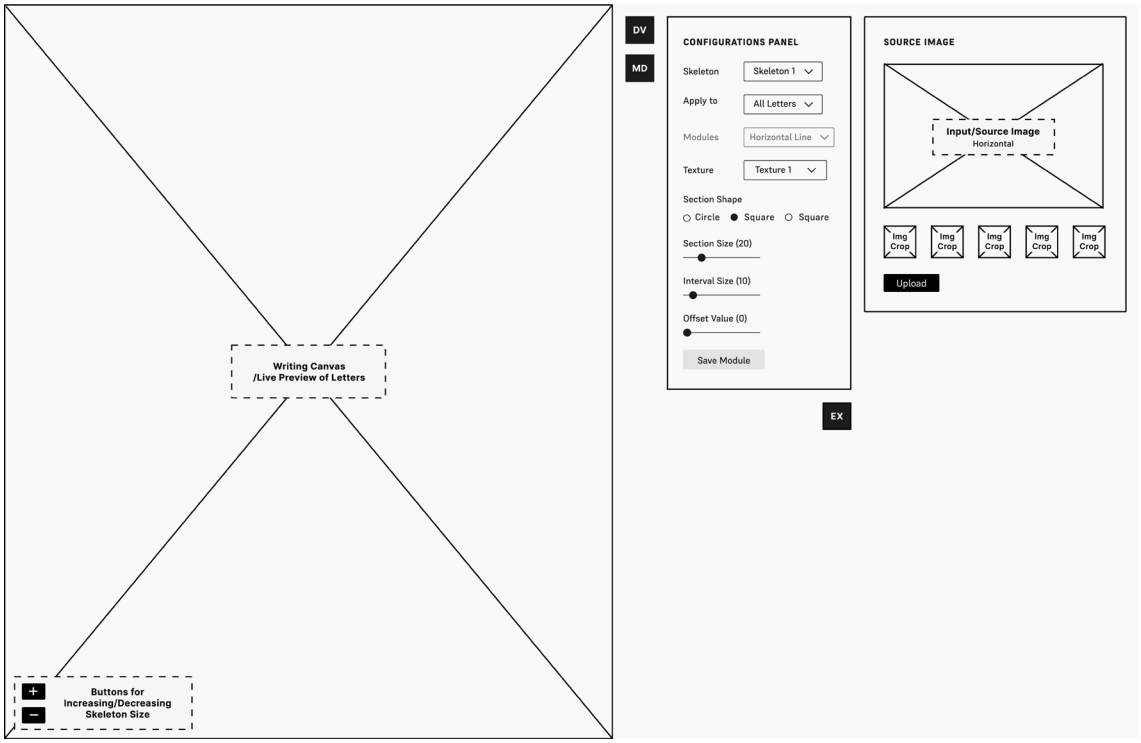
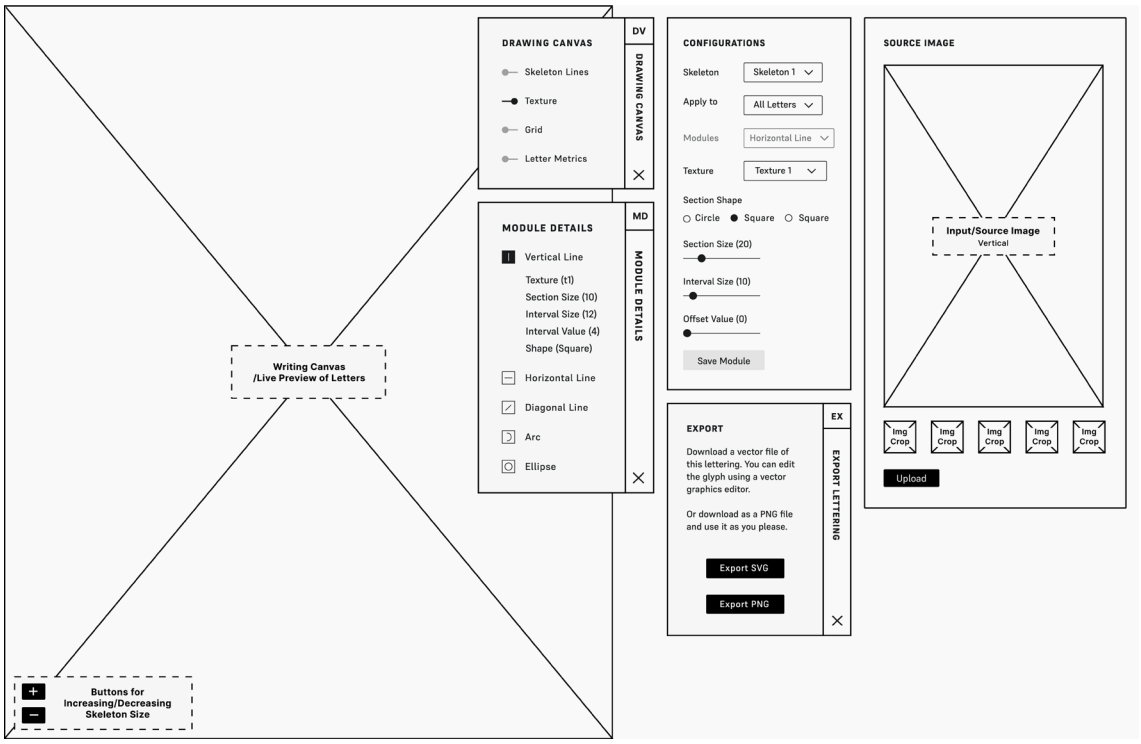


FIGURE 4.26.
Wireframe with closed secondary panels with a horizontal source image.



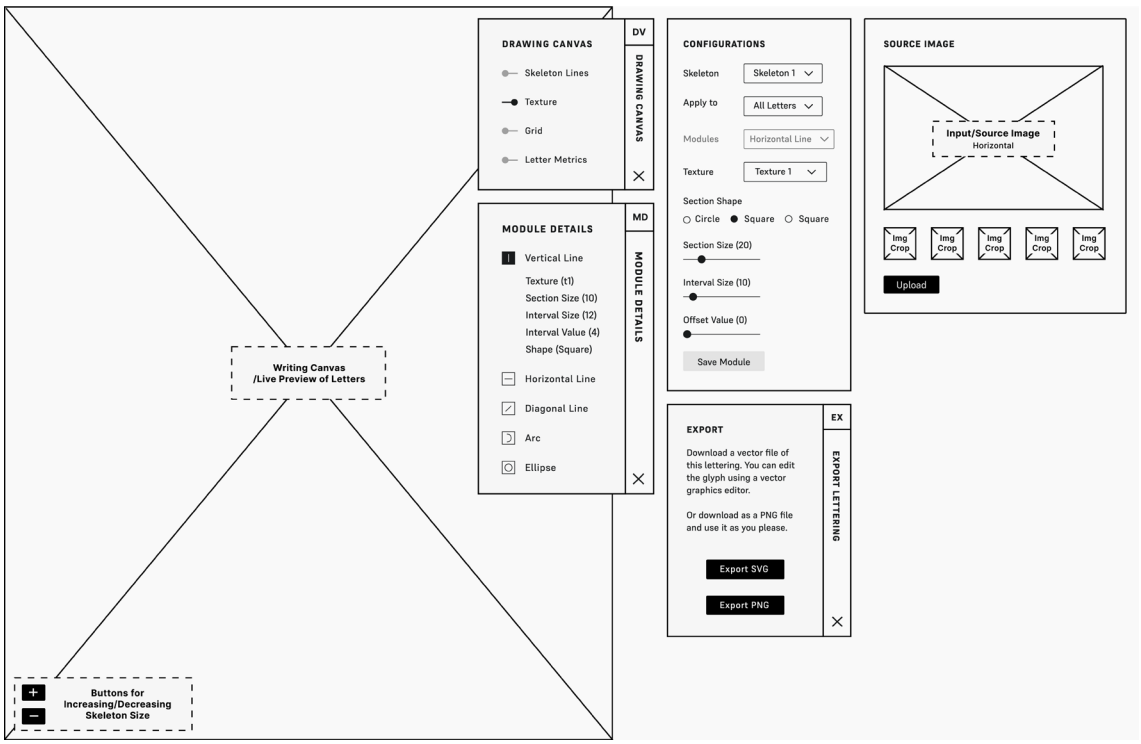


FIGURE 4.27.
Wireframe with closed panels and a vertical source image.

FIGURE 4.28.
Wireframe with opened panels and a horizontal source image.

WIREFRAMES REVIEW

After the wireframes were finished, we evaluated the layout, interactions, and general process to determine how the user interface should be enhanced in the UI's subsequent iteration. We first noticed that the best way to take advantage of the white space in the panel area was to focus on distributing the components on the vertical axis. Even with the layout created and with closable panels, the writing canvas was still partially

contaminated by the panels. We believed that decreasing the width of the configurations panel, for example, by putting the name above the respective dropdown, could open some horizontal space; however, these changes would consequently impact other aspects of the layout.

Another improvement that emerged upon analysing these wireframes was that showing the module details panel might only make sense if the option to apply to modules was activated. In addition, the next iteration of the UI has to incorporate other interface details, such as the navigation bar.

4.3.3. INTERACTIVE PROTOTYPE

After creating the wireframes and analysing them, we had a strong base from which to build the UI's final design and layout. At this point, we created an assortment of high-fidelity mockups and connected them to produce an interactive digital prototype. With this high-fidelity prototype, we aimed to begin materialising the UI's visual style and aesthetic while also obtaining a notion of the system's interactions. This process is simple since all the user actions are compiled on one page. Hence, we did not need to create many screens to make the interaction with the system apparent.

The following paragraphs demonstrate and discuss the prototype created, emphasising the key elements that have changed from the wireframes. The screens directly relevant to the text are shown in the following paragraphs, as in previous subsections. However, others screen iterated can be found in Appendix B. Besides the design effort made to create the user interface, the most evident differences from the wireframes are the layout and nomenclature of the various panels and the creation of a horizontal navigation bar.

When designing this prototype, we wanted it to be as candid and simple as possible, shifting the user's attention to the outputs created. As mentioned, the source image had to be permanently displayed, limiting the colour palette options for the UI. Furthermore, the infinite images that might be used and uploaded into the interface could create darker letterforms that did not work on a black background and vice versa. Hence, we chose two neutral colour palettes, which allowed the user to choose from a light and dark mode. The user could change the UI appearance on the navigation bar, which contained a light/dark mode switch, an about section and the project name. The about section works as an overlay which explains the project and its context to the user (Figure 4.29.).

Another improvement from the wireframes was the creation of icons for all the secondary panels and also, the icons for the toggle buttons on the Letter Canvas panel. In addition, and opposite the wireframes, the secondary panels had three states. The first was when the panel was closed, and only its icon was exposed, the second activated with a hovering interaction, which would open a tab with the panel name and finally, when clicked, the entire panel would be displayed (Figure 4.30.).

The first change in the layout was separating the Export Panel from the panel area. This panel is used with less frequency and is not related to letter customisation; thus, allocating it to the left corner of the screen spared more space in the panel area. In addition, and as mentioned in the

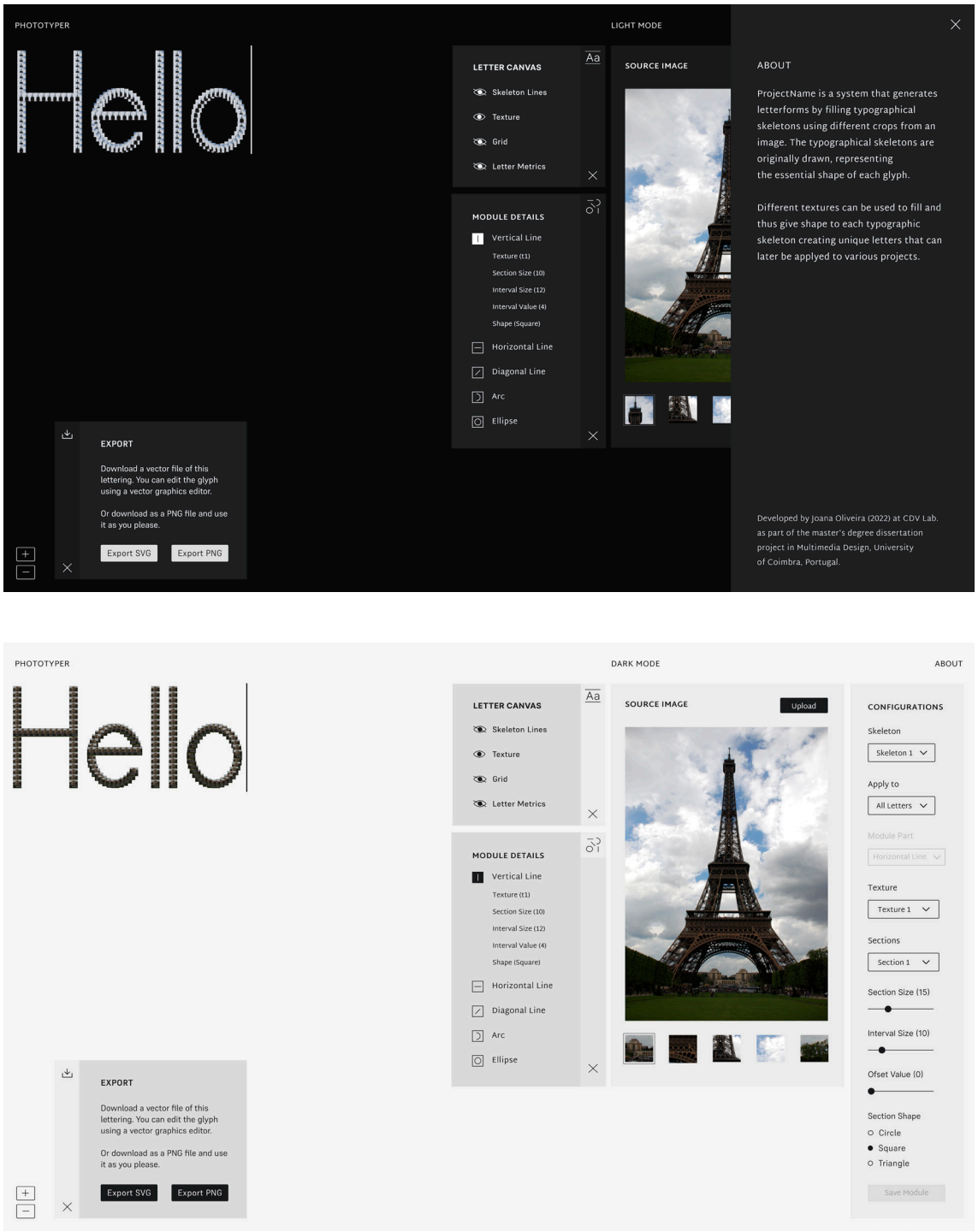


FIGURE 4.29.
Interface mockup with dark mode (About section opened) and light mode screens with opened panels.

wireframe discussion, we also decided to compact the configuration panel and make it a thinner and more vertical panel that covered the total height of the panel area. Therefore, we placed this panel on the furthest right, moving the source image panel to the centre. This panel also suffered some changes. First, the upload button was moved next to the panel name, and then, the image sections were signalised with a border if used.

Further, we thought about the interaction of selecting another image crop. First the user clicks on an image section already displayed underneath the image source. This action would show where the respective crop was located on the source image and after, while hovering over

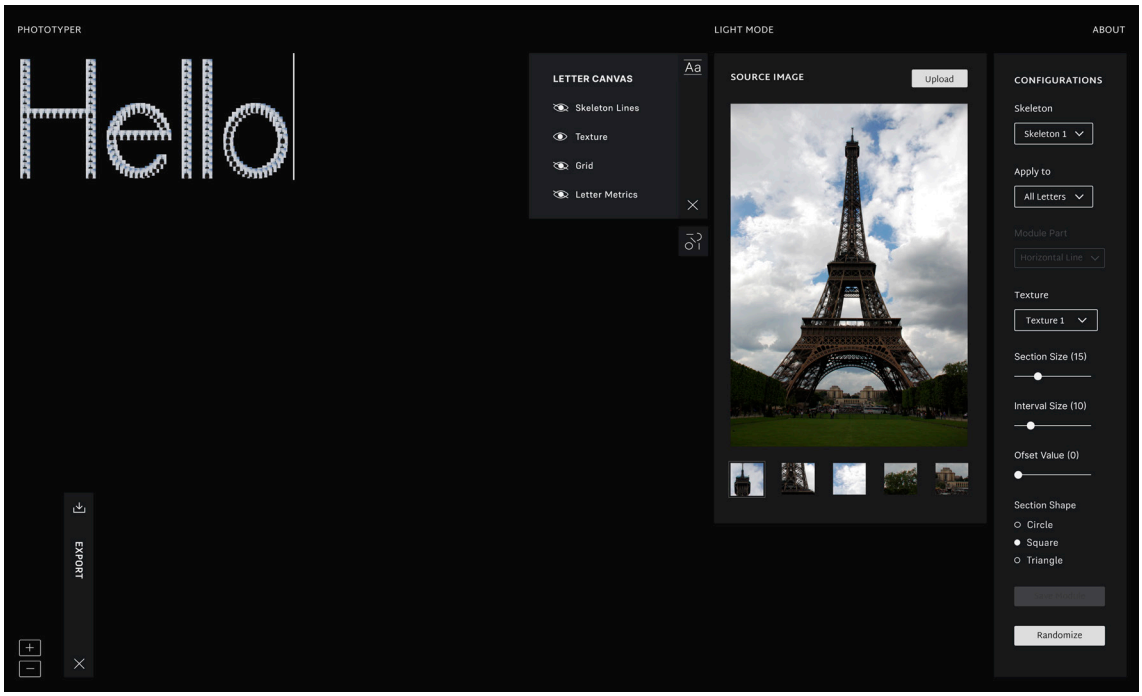


FIGURE 4.30.
Interface Mockup with Export panel in *hovering state*, closed Module Detail panel and opened Drawing Canvas panel.

the image, the user could choose another part of the image to be used. If it corresponded to the section being used by the selected texture (section with a border), the letterforms would update accordingly.

4.3.4. IMPLEMENTATION

As mentioned in subsection 4.2.2., the implementation used *p5*.js and Javascript as the core for the project. At this point, and in order to implement the user interface and other functionalities, we combined those languages with two others that constitute most websites, HTML and CSS. The first steps to implement this UI was to first correct some technicalities about the writing functionality in the webpage and then, distribute, position and implement all the panels according to the prototype created.

WRITING AREA/LIVE PREVIEW

One of the most crucial parts of the interface was the writing area, where the output letterforms would appear and where the user could write. Unlike the other elements, this area and the source image panel are created in a canvas with *p5.js*. Therefore, we had to integrate the texture application method mentioned in subsection 4.2.2. to allow the user to write directly onto the canvas. When writing, a user expects some behaviour from the system. First, it should draw a letter when a character key is pressed; secondly, a letter should disappear when the *Return* key is pushed; and finally, when the *Space* and *Enter* keys are pressed, it must create a space or a paragraph between letters. The first behaviour was already implemented however it was only possible to draw one letter at a time. Thus, we started by implementing a way for the user to write more than one letter.

By saving the furthest x coordinate value of a letter drawn, we could determine in which position to draw the following letter. Also, to create a space between letters, we had to add the space value intended to that variable. Similarly, to create a paragraph, we used the descender metric value, added the intended paragraph spacing, and saved that value as the y coordinate for the following letter. On a related note, to erase a letter, we had to save the keys pressed (what had been written) into an array. Then, once the *Return* key was pressed, the last element of the array was removed (the last key pressed), and all letters were drawn again.

Once these behaviours were implemented, writing with the intended typographic skeleton was possible. In addition, the system displays a placeholder text randomly selected from a previously curated collection of words. Furthermore, the placeholder is cleared when clicked, allowing the user to write directly without having to erase the initial phrase.

SOURCE IMAGE PANEL

This panel is also created on a canvas as the writing area. The first implementation improvement was making the canvas responsive to horizontal and vertical images. This refinement was crucial since the system allowed the users to upload images from their computers. Another implementation in this panel was the interaction explained in subsection 4.3.3., where the image sections could be selected and changed to another image crop.

In addition, to create more dynamism in the platform, we decided to choose four images and display them randomly on the platform once loaded. It created different letter outputs once the system was loaded, showcasing its potential to a user.

CONFIGURATIONS PANEL AND MODULE DETAIL PANEL

The implementation of this panel was relatively straightforward since it consisted of integrating a UI component (a dropdown, a slider or a bullet checkbox) to pre-existent variables. As mentioned, we wanted all the actions on this panel to affect the output letterforms directly. If the letters were being drawn inside the ***draw()*** function of *p5.js*, they would automatically change; however, as explained in subsection 4.2.2., this was not the case. Therefore, we had to develop a function ***updateLetter()*** which read the array that contained all the letters typed and drew them again once a configuration parameter was changed. In addition, we added another parameterisation that was not on the interactive prototype. If the selected texture used only one of the five image sections (*Basic* and *Random Size Texture*), then the user could select another section from the five available.

The other interaction which was also crucial in this panel was the application of configurations to a specific module. The *Modules* dropdown and *Save Module* button are deactivated once the system is loaded; however, this changes once the *Apply To* dropdown in the configurations panel swaps from *All letters* to *Modules*. In addition, the *Module Details* panel is also displayed, showing all the applied parameters to a module.

Once the *Apply To Modules* is selected, the user can select all the modules available in the skeleton from the dropdown box. Once the module is selected, all the forward alterations, namely, the texture, section, section size, interval size, offset value and section shape, are applied to the module in question. However, the module specifications are saved only when the *Save Module* button is clicked. Note that the user receives a feedback message to ensure the module is saved. As mentioned above, the user can also consult the module specification in the Module Detail panel.

LETTER CANVAS PANEL

The toggles inside this panel made changes to the writing area previously implemented. The first toggle on the panel shows or hides the skeleton selected. In the base implementation, all the skeleton modules were drawn with the same colour. However, in this final stage, we wanted to distinguish the different modules in the skeleton by drawing them with different colours. The next toggle would display or hide the grid in which the skeleton was created. Initially, in earlier implementations, the grid was made with lines, but due to the detail of its mesh, it clashed with the other elements. So we decided to turn the grid into points allowing the user to see the grid that served as a base for the skeleton.

However, even after these adjustments, we found that it still created visual clutter and was useless to the user because, most of the time, the glyphs were not positioned on the grid coordinates due to the tracking added to the letters. Therefore, we decided to remove this option but still implemented the metrics toggle, which allowed the user to see the anatomic measurements of the letter, namely, the *ascender, x-height, baseline* and *descender* lines. Lastly, the texture toggle shows or hides the texture in the letterforms created, thus allowing us to see the letters with only the skeleton, for example.

EXPORT PANEL AND SKELETON SIZE BUTTONS

The skeleton size button and the export panel are located side by side in the left lower corner of the writing area. As the name suggests, the first increases or decreases the skeleton's size by clicking the plus or minus button. The best way to implement this functionality was to decrease or increase the grid mesh used as the base for the skeleton selected. Further, we had to establish limits for a maximum and minimum size for the skeleton and then block the respective buttons so it is evident when the user has reached those limits. We call this button skeleton size because it only alters the size of the skeleton and not the respective texture applied to it.

The export lettering panel was implemented as a secondary panel with the three states referred to in the previous subsection. The inside of the panel contained the export as an SVG button, and we also implemented another button so it would be possible to export the letterforms as a PNG.

4.4. RESULTS AND APPLICATIONS

This section is dedicated to displaying letterforms created by the developed system and addressing possible uses for those outputs. One of the main goals of the web system is to allow graphic designers and other types of the target audience to explore and create letters able to convey meaning and carry information in their body and style. Thus, it made sense to explore possible applications where these outputs could be integrated and showcase some possible results created with the developed system.

The type of glyphs exported and created by the system may be a beneficial addition to design artefacts that require a more or less nuanced connection to a concept. In this way, the system and its letterforms may successfully contribute to designing visual materials whose source and base have a direct or rhetorical relationship to a specific image. This opportunity is vast due to the infinite concepts and realities that an image or photograph can capture, enabling a possible connection with the design concept.

Logically, some input images result in better outputs than others. When we first implemented the system, we used Wassily Kandinsky's oil canvas from 1911 as a source image, which, due to its colours and shapes, created intricate and colourful results. This first concept led us to create a set of design materials around a renowned artist, Pablo Picasso. Using his creations as source images, we developed different glyphs and, posteriorly, designed materials that directly related to his work (Figure 4.31.–4.36.). We created a set of materials that used, in total, seven paintings from the artist as input images to the system. These materials connect the artefact, the subject, and the intention behind the design project and concept. However, these images, like the Kandinsky one, are by themselves visually attractive and complex; thus, there was a need to test with more common images, such as photographs.

One of the first applications that came to mind was travel postcards. Typically, and especially travel-related postcards, use typefaces with images of the cities. Therefore, we designed simple postcards for different cities worldwide (Braga, Coimbra, Barcelona, London and New York) using images of those locations as inputs and exploring the texture options and parameterisation of the system (Figures 4.37.–4.39.).

Additionally, we developed a set of posters which explored different letter outputs developed with the system using arbitrary chosen source images (Figure 4.40.–4.42.). These posters explore the possible outputs in an unconnected and purely aesthetic way, contrasting to the other materials showcased above, which explored the connection of the letterforms with an inherent design concept (more detail in Appendix C).

FIGURE 4.31.
Bag design with letterforms created with the system by using several Picasso paintings as source images.

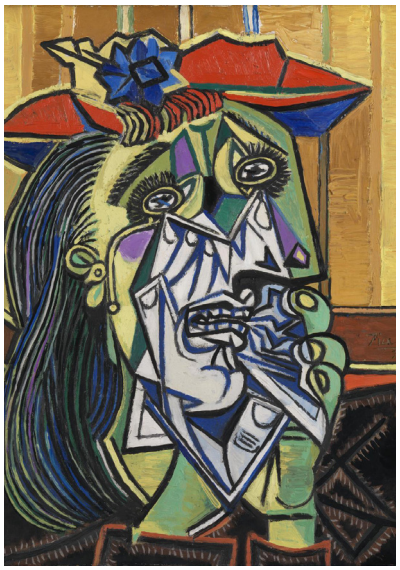
FIGURE 4.32.
Postcard (left) with letterforms created using the *Vieux guitariste aveugle* painting of Picasso as source image (right).

FIGURE 4.33.

Postcard (left) with letterforms created using the *Femme aux cheveux jaunes* painting of Picasso as source image (right).

FIGURE 4.34.

Postcard (left) with letterforms created using the *Femme en pleurs* painting of Picasso as source image (right).



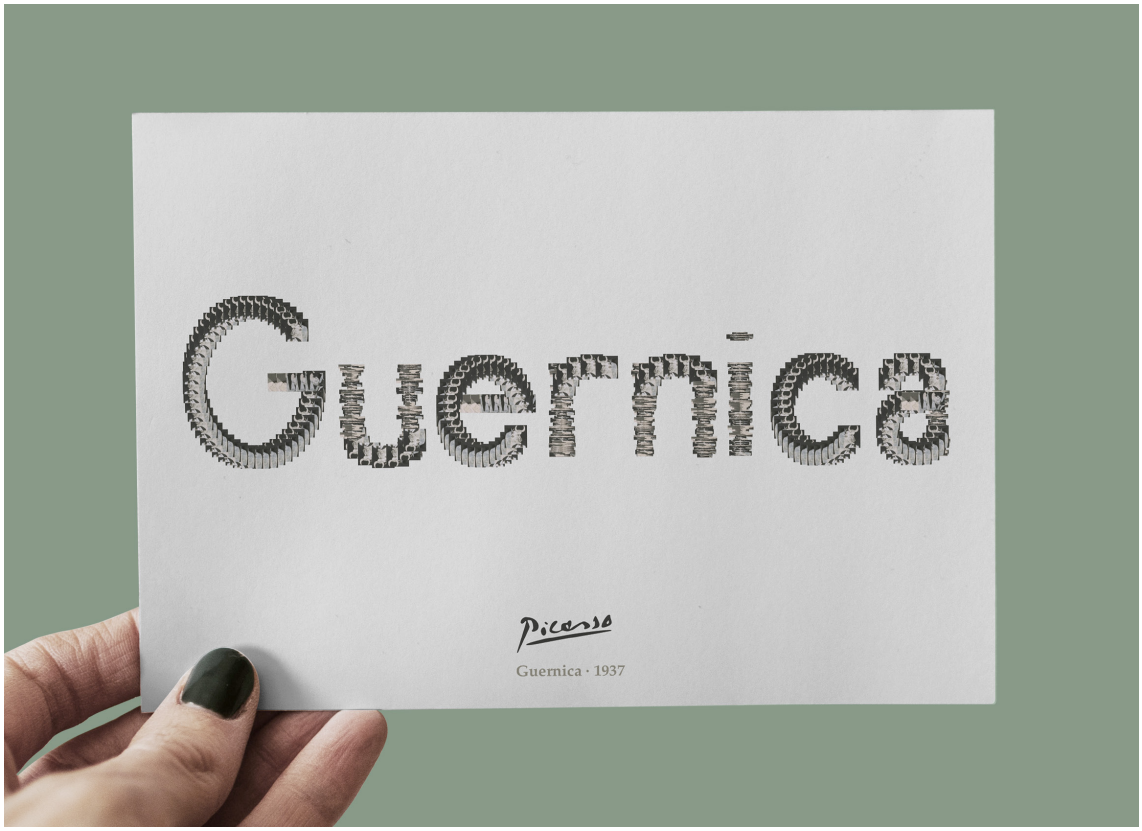


FIGURE 4.35.

Postcard (top) and t-shirt (middle) made with letterforms created using the *Guernica* painting of Picasso as source image (bottom).

FIGURE 4.36.
Postcard (top) with letterforms created using the *Les demoiselles d'Avignon* painting of Picasso as source image (bottom).

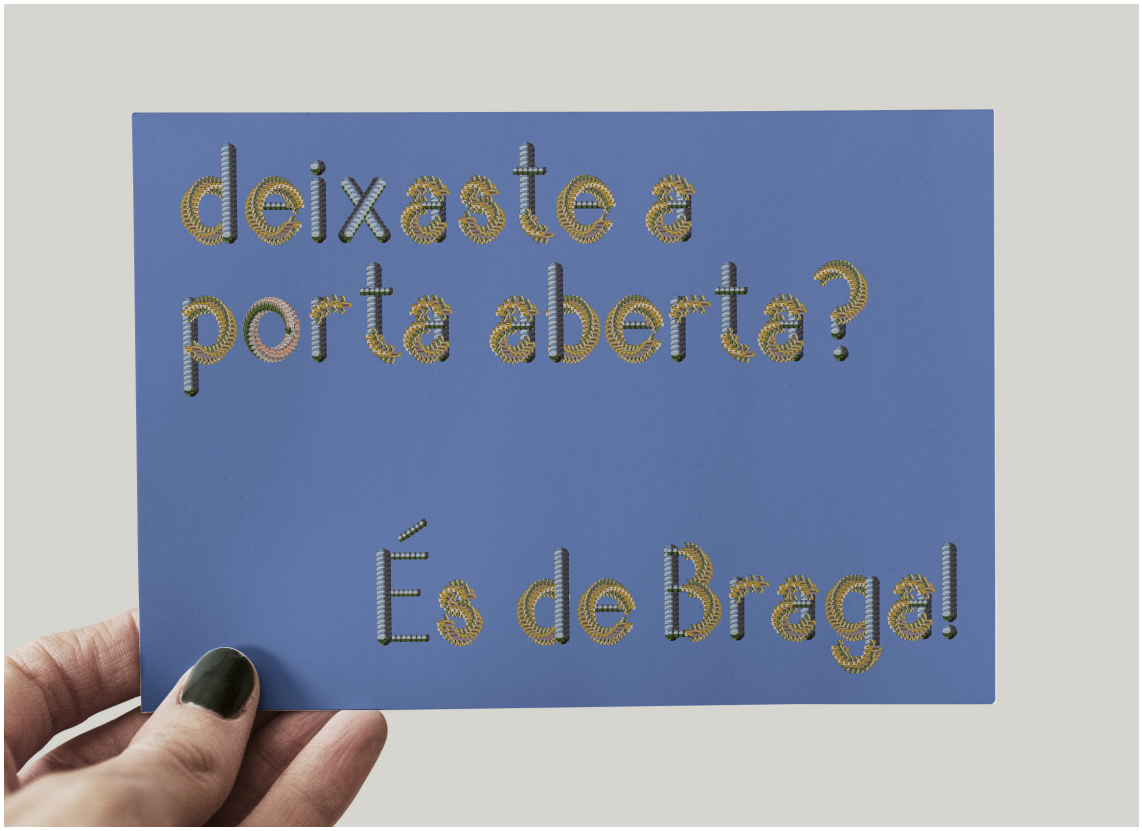


FIGURE 4.37.
Postcard with expression associated with Braga, Portugal. Letterforms created using an image of the respective city.

FIGURE 4.38.
Postcard for Coimbra, Portugal. Letterforms created using an image of the respective city.

FIGURE 4.39.

Postcard for Barcelona, London and New York city. Letterforms created using an image of the respective cities.

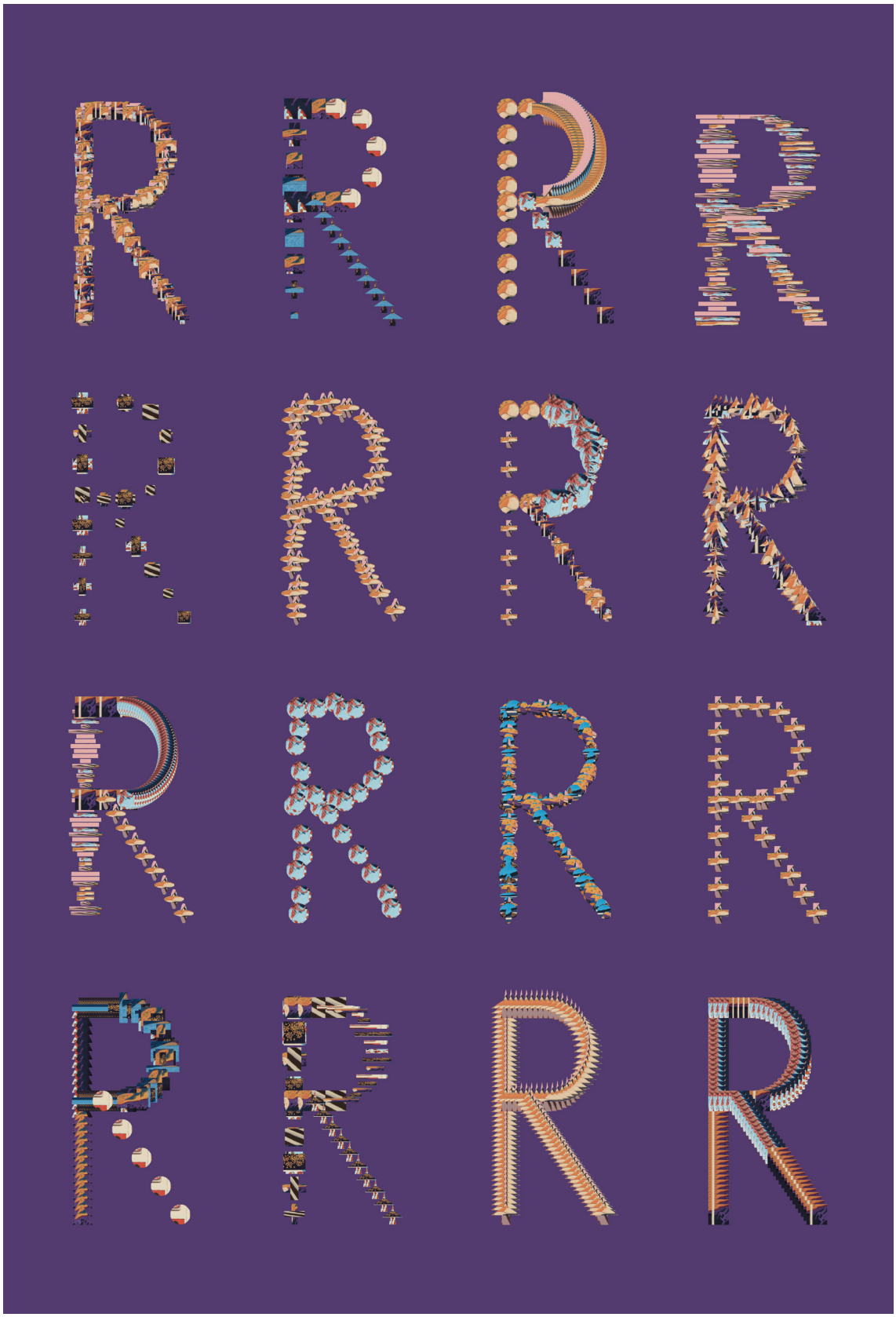


FIGURE 4.40.
Experimentation with letter R, with several textures and combinations.

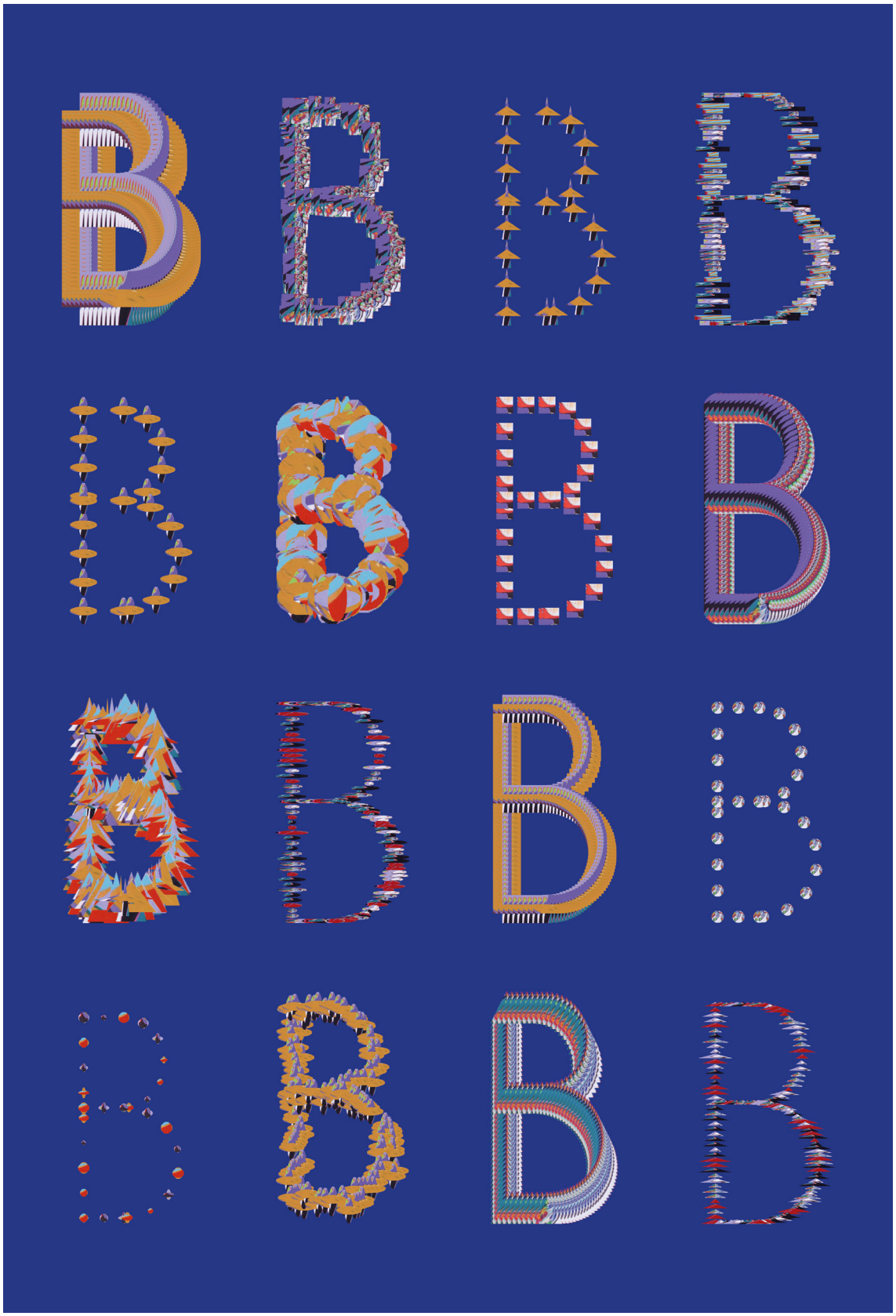


FIGURE 4.41.
Experimentation with letter B, with several textures and combinations.

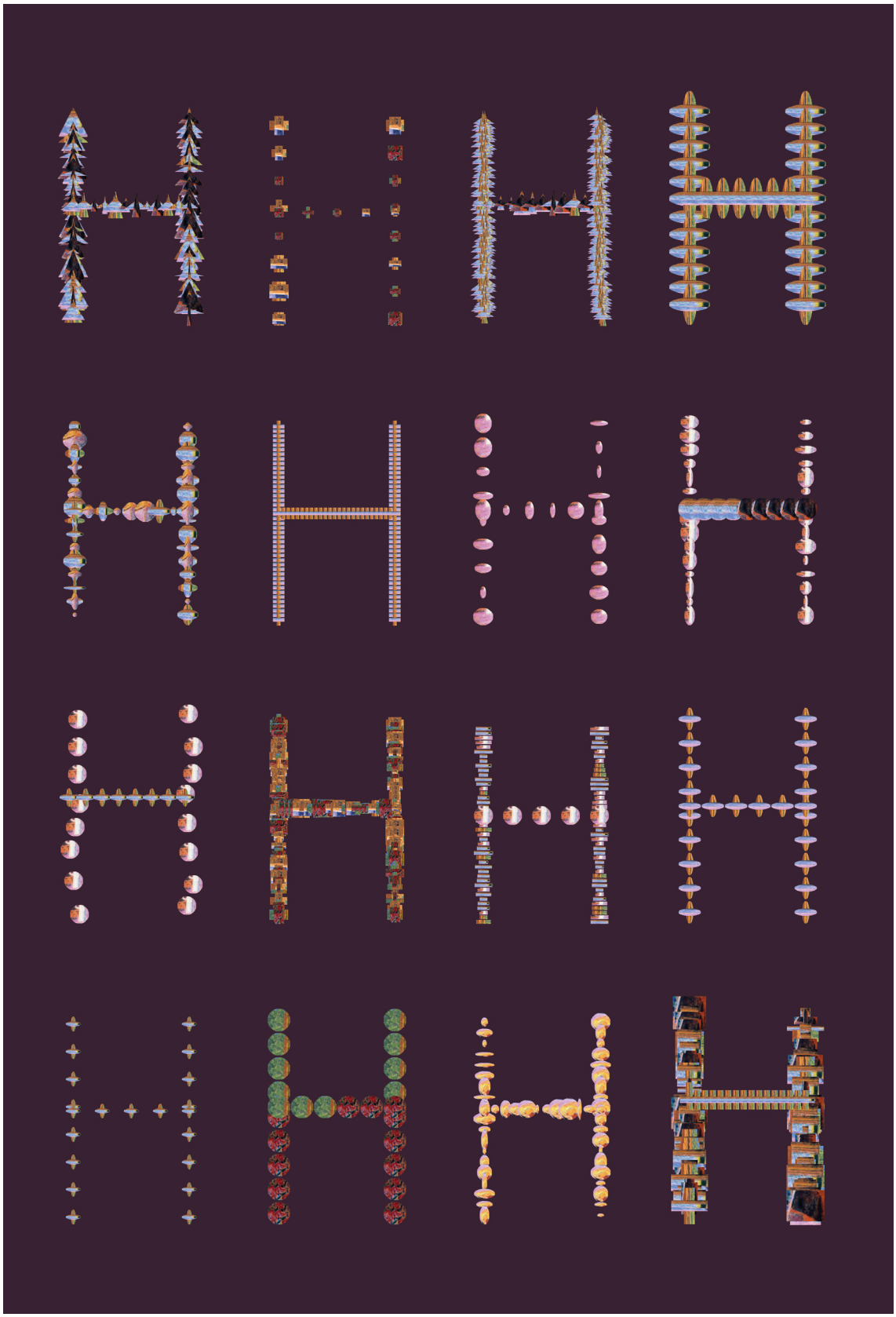


FIGURE 4.42.
Experimentation with letter H, with several textures and combinations.

5. CONCLUSION

We began this dissertation with a thorough investigation of typography. As humans, the importance of typography can be disguised with all the external noise. In contrast, as designers, its presence is crucial and can bring another level of meaning and significance to our work.

To better grasp and understand all the concepts and notions surrounding typography, it was necessary to create a theoretical foundation. This investigation started with an overview of its historical context, anatomic terminology, and functionalities. Later, research was done about the capacity of typography to be multimodal, how it relates to other mediums, such as images, and how they can also be a semiotic mode. We compiled and reviewed related work involving the objectives of this dissertation, more specifically, type development (computationally and manually). Numerous notions were mentioned, such as modular and conceptual typography, dynamic letterforms and type systems.

With this research in mind, we began conceptualising this dissertation's practical component, which revolved around developing a web-based system capable of creating letterforms filled and texturised with other mediums besides colour. Exploring the possible relation between letters and images led us to some initial experimentations that marked this practical component's course. These experiments included dragging a section of an image through the skeletal body of a letter. This approach concluded that the best way to draw and fill a letter was to look at them as a set of adjustable modules created with the same base grid and metrics. In addition, it also helped to clarify and establish some filling/texturising possibilities for the letter's body.

These different experiments led us to develop a drawing and texturising method that became the project's core. This method's sole purpose is to read, draw and fill/texturise the letter base structure (*skeleton*). This skeleton was previously manually drawn and later turned into a syntax the algorithm could read. After being computationally drawn, each skeleton is filled with various textures or filling techniques that use cropped sections from a source image. This system accepts different parameters, which impact the visual appearance of each letterform. In addition, the system was prepared to apply specific parameters and specifications to the individual components of the selected *skeleton*.

Adjacently, this method served as the base for the user-focused system, available at https://phototyper.dei.uc.pt/. By analysing the essential parameters already internally used in the former method, we could outline all the interactions and features of this web-based system. Part of this procedure was dedicated to fully understanding the parameters available for letter configuration and which ones could increase the output possibilities without creating unnecessary complexity to the user interface. The system allows users to type and create customisable letterforms, texturised with an image section from a chosen source image, via a public webpage. The resulting letters may then be exported as an image file (PNG) or as a vector file (SVG) which can be used and edited using a vector graphics editor.

Finally, the created system can generate texturised letters with any arbitrary image. Because of this absence of constraints, together with the remaining set of parameters, we can essentially state that, in accordance

with our goals, we are operating with a highly customisable and practically infinite variety of outputs. In addition, when exploring possible uses for those outputs, we could match our objectives further, proving that using images integrated with type can create and enhance a possible connection with the design concept or brief.

Finally, as the work covered within the scope of this dissertation comes to a close, we record a list of improvements, extra features, and general ideas for future work that we did not have time to explore or execute throughout development. To begin with, the immediate improvement that needs to be addressed is the absence of support for all major browsers. Similarly, even if the fundamental issues with lower resolution screens have been addressed, the user experience may benefit from a more responsive structure. In addition, the system still has some optimisation and performance problems with executing some parameter values. Concerning added features, we would like to allow the user to export and later import a previously created set of parameters that led to the creation of specific letters. If needed, this would allow the user to access the same letter texture and parameters.

Further, it could be interesting for the project exposure to allow users to save created letters in a public gallery incorporated inside the web platform. Other features that could be later incorporated are related to the texturising aspect of the letters. We think there is still room for developing more textures and filling techniques. In addition, we would like to incorporate artificial intelligence to connect the writing words with the source images making the texture automatically adapt to what was written. Finally, and as mentioned in the early experiments of the practical component of this dissertation, we believe in the interactive and animation potential of the system by having them react to their external environment through live video capture, for example, creating an opportunity for developing an interactive installation using the outputs letters.

6. REFERENCES

Aghabayan, T. (n.d.). *Elien: Gestalten mit Code*, FH Mainz. Retrieved 27 December 2021, from http://generative-typografie.de/generativetypografie/elien/

Amado, P., & Silva, C. (2011). *Anatomia Tipográfica in Veloso*, A.; Dias, N.; Martins, O.; Amado, P. "II Encontro de Tipografia: Livro de Atas". Aveiro: Edição eletrónica do II Encontro de Tipografia, Departamento de Comunicação e Arte da Universidade de Aveiro.

Antonelli, P., & Carmody, K. (2011). *Standard Deviations: Types and Families in Contemporary Design | MoMA*. Retrieved 27 December 2021, from The Museum of Modern Art website: https://www.moma.org/calendar/exhibitions/1138

Bargues, C. (2016, April 28). *Dada optophonetic*. Retrieved 23 December 2021, from Memento website: http://www.diptyqueparis-memento.com/en/dada-optophonetic/

Barthes, R., & Heath, S. (1977). *Image, music, text: Essays* (13. [Dr.]). London: Fontana.
Bil'ak, P. (2008). Typotheque: History font family. Retrieved 28 December 2021, from https://www.typotheque.com/fonts/history

Bil'ak, P. (2010, March 9). *Typotheque: The history of History by Peter Bil'ak*. Retrieved 28 December 2021, from https://www.typotheque.com/articles/the_history_of_history

Bringhurst, R. (2004). The elements of typographic style (3rd ed). Point Roberts, WA: Hartley & Marks, Publishers.

Butler, L. (2015, August 1). *Generative Sans*. Retrieved 4 August 2022, from Medium website: https://medium.com/@leonbutler/generative-sans-1560aa91a90a

Changizi, M. A. (2010). *The vision revolution: How the latest research overturns everything we thought we knew about human vision*. Dallas: Benbella Books.

Cheng, K. (2020). *Designing type (Second edition)*. New Haven: Yale University Press.

Deck, B. (2011). Barry Deck. *Template Gothic. 1990 | MoMA*. Retrieved 27 December 2021, from The Museum of Modern Art website: https://www.moma.org/collection/works/139319

Design Council UK. (2015, March 17). *What is the framework for innovation? Design Council's evolved Double Diamond*. Retrieved 1 January 2022, from Design Council website: https://www.designcouncil.org.uk/news-opinion/what-framework-innovation-design-councils-evolved-double-diamond

Dressman, M. (2019). *Multimodality and Language Learning*. In M. Dressman & R. W. Sadler (Eds.), The Handbook of Informal Language Learning (1st ed., pp. 39–55). Wiley. https://doi.org/10.1002/9781119472384.ch3

FF3300. (2019). *Progetto di comunicazione per Nòva—2019*. Retrieved 28 December 2021, from FF3300 website: https://www.ff3300.com/en/works/nova

FontShop. (n.d.a). *FF Beowolf Font*. Retrieved 27 December 2021, from FontShop website: https://www.fontshop.com/families/ff-beowolf

FontShop. (n.d.b). *FF Fudoni Font*. Retrieved 27 December 2021, from FontShop website: https://www.fontshop.com/families/ff-fudoni

Grrr.nl. (n.d.). *Wim Crouwel: Mr. Gridnik*. Retrieved 27 December 2022, from https://www.stedelijk.nl/en/exhibitions/wim-crouwel

Hamamoto, C. (2012, December 27). *FUSE 1–20*. Retrieved 3 August 2022, from Typographica website: https://typographica.org/typography-books/fuse-1-20/

Hanzer, E., & Zia, F. (n.d.). *Phase*. Retrieved 27 December 2021, from Elias Hanzer website: https://www.eliashanzer.com/phase/

Holland-Cunz, N. (n.d.). *Broken Grid: Gestalten mit Code, FH Mainz*. Retrieved 27 December 2021, from http://generative-typografie.de/generativetypografie/broken-grid/

Jung, I.-H. (n.d.). PDE Lubanoise: Gestalten mit Code, FH Mainz. Retrieved 27 December 2021, from http://generative-typografie.de/generativetypografie/pde-lubanoise/

Kersten, S. (n.d.). *Pong: Gestalten mit Code,* FH Mainz. Retrieved 27 December 2021, from http://generative-typografie.de/generativetypografie/pong/

Klein, D. (n.d.). *Blast: Gestalten mit Code,* FH Mainz. Retrieved 27 December 2021, from http://generative-typografie.de/generativetypografie/blast/

Lehni, J. (2012, November 15). *Scriptographer.org—News*. Retrieved 27 December 2021, from https://scriptographer.org/

Lewis, J. E., & Nadeau, B. (2010). *Post PostScript please. Digital Creativity*, 21(1), 18–29. https://doi.org/10.1080/14626261003654632

Lupton, E. (2010). *Thinking with type: A critical guide for designers, writers, editors, & students (2nd rev. and expanded ed)*. New York: Princeton Architectural Press.

Maçãs, C., Palma, D., & Rebelo, A. (2019). *typEm: A generative typeface that represents the emotion of the text.* Proceedings of the 9th International Conference on Digital and Interactive Arts, 1–10. Braga Portugal: ACM. https://doi.org/10.1145/3359852.3359874

Makela, P. S. (2011). P. Scott Makela. *Dead History. 1990 | MoMA.* Retrieved 27 December 2021, from The Museum of Modern Art website: https://www.moma.org/collection/works/139317

Marxer, R. (n.d.). *Geomerative*. Retrieved 27 December 2021, from http://ricardmarxer.com/geomerative/

McNeil, P. (2017). *The visual history of type*. London: Laurence King Publishing.

Meggs, P. B., & Purvis, A. W. (2016). *Meggs' history of graphic design (Sixth edition).* Hoboken, New Jersey: Wiley.

Mr. Keedy. (1993). *Emigre: Essays—Graphic designers probably won't read this.* Retrieved 7 March 2022, from https://www.emigre.com/Essays/Emigre/Graphicdesignersprobablywontreadthisbut

MuirMcNeil. (n.d.a). *About – MuirMcNeil*. Retrieved 27 December 2021, from https://muirmcneil.com/about/

MuirMcNeil. (n.d.d). *TwoBit Type System – MuirMcNeil*. Retrieved 27 December 2021, from https://muirmcneil.com/project/twobit-type-system/?section=about

MuirMcNeil. (n.d.e). *TwoPlus Type System – MuirMcNeil*. Retrieved 27 December 2021, from https://muirmcneil.com/project/twoplus/?section=about

MuirMcNeil. (n.d.f). *TwoPoint Backgroung—MuirMcNeil*. Retrieved 27 December 2021, from https://muirmcneil.com/project/twopoint/?section=about

Pape, P., & Florian, J. (n.d.). *Generative Typografie: Gestalten mit Code, FH Mainz*. Retrieved 27 December 2021, from http://generative-typografie.de/generativetypografie/about/

PeterCon. (2021, September 12). *SVG - Scalable vector graphics table (OpenType 1.9)—Typography*. Retrieved 11 August 2022, from https://docs.microsoft.com/en-us/typography/opentype/spec/svg

Phillips, K. (2015, June 18). *Quick Design History: Wolfgang Weingart #tbt*. Retrieved 9 March 2022, from Shillington Design Blog website: https://www.shillingtoneducation.com/blog/wolfgang-weingart-tbt/

PrintMag. (2012, June 8). *The Fuse Box: Faces of a Typographic Revolution*. Retrieved 3 August 2022, from PRINT Magazine website: https://www.printmag.com/design-books/the-fuse-box-faces-of-a-typographic-revolution/

Puckey, J. (2006, August 19). *Scriptographer.org—Tile Tool. Retrieved 27 December 2021*, from https://scriptographer.org/scripts/interactive-tools/tile-tool/comments#replies

Puckey, J. (2007). *Jonathan Puckey—Special Box*. Retrieved 27 December 2021, from https://jonathanpuckey.com/projects/special-box/

Reigel, A., & Müller, M. (n.d.). *Modulator | metaflop*. Retrieved 27 December 2021, from https://www.metaflop.com/modulator

Reimann, L. (n.d.). *Zwirn: Gestalten mit Code, FH Mainz*. Retrieved 27 December 2021, from http://generative-typografie.de/generativetypografie/zwirn/

Reinheimer, I. (n.d.). *Irratio: Gestalten mit Code, FH Mainz*. Retrieved 27 December 2021, from http://generative-typografie.de/generativetypografie/irratio/

Samara, T. (2011). *Typography workbook: A real-world guide to using type in graphic design*. Gloucester, Mass.: Rockport Publishers. Retrieved from https://search.ebscohost.com/login.aspx?direct=true&scope=site&db=nlebk&db=nlabk&AN=585192

Tschense, T. (n.d.). *Bastard: Gestalten mit Code, FH Mainz*. Retrieved 27 December 2021, from http://generative-typografie.de/generativetypografie/bastard/

Van Blokland, E., & Van Rossum, J. (2011). *Erik van Blokland, Just van Rossum. FF Beowolf.* 1990 | MoMA. Retrieved 27 December 2021, from The Museum of Modern Art website: https://www.moma.org/collection/works/139326

Van Leeuwen, T. (2005). *Introducing social semiotics.* London ; New York: Routledge.

van Leeuwen, T. (2006). T*owards a semiotics of typography. Information Design Journal, 14(2), 139–155.* https://doi.org/10.1075/idj.14.2.06lee

Venezky, M. (n.d.). *Speak Premiere Issue.* Retrieved 3 August 2022, from Martin Venezky website: https://www.martinvenezky.com/content/speak-issue-1-20

Willen, B., & Strals, N. (2009). *Lettering & type: Creating letters and designing typefaces (1st ed).* New York: Princeton Architectural Press.

XYZ2018 Lab. (2021). *GTL [Python]*. Retrieved from https://github.com/bbtgnn/GTL (Original work published 2019)

APPENDIX A
– TEXTURE OUTPUTS

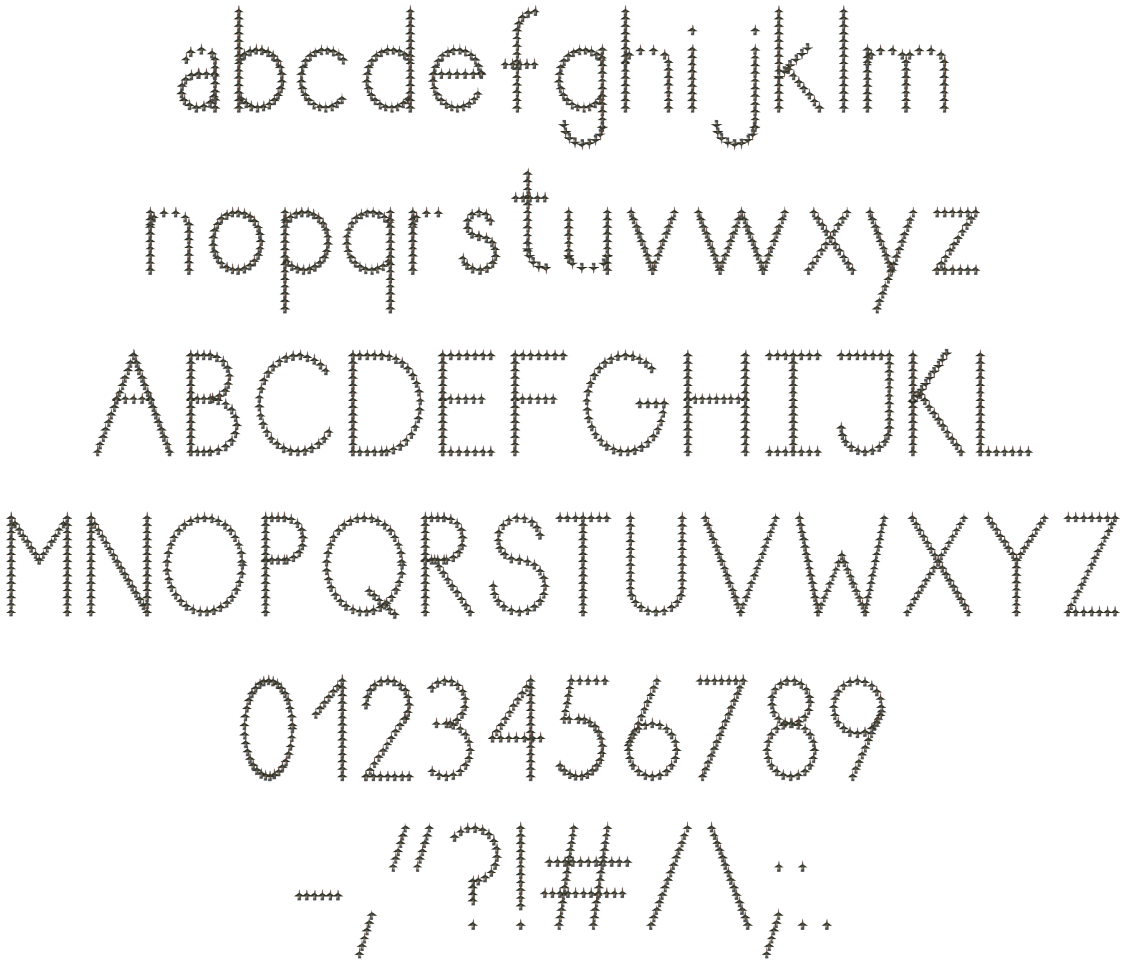
BASIC TEXTURE — SQUARE SHAPE

BASIC TEXTURE — CIRCLE SHAPE

BASIC TEXTURE — TRIANGLE SHAPE

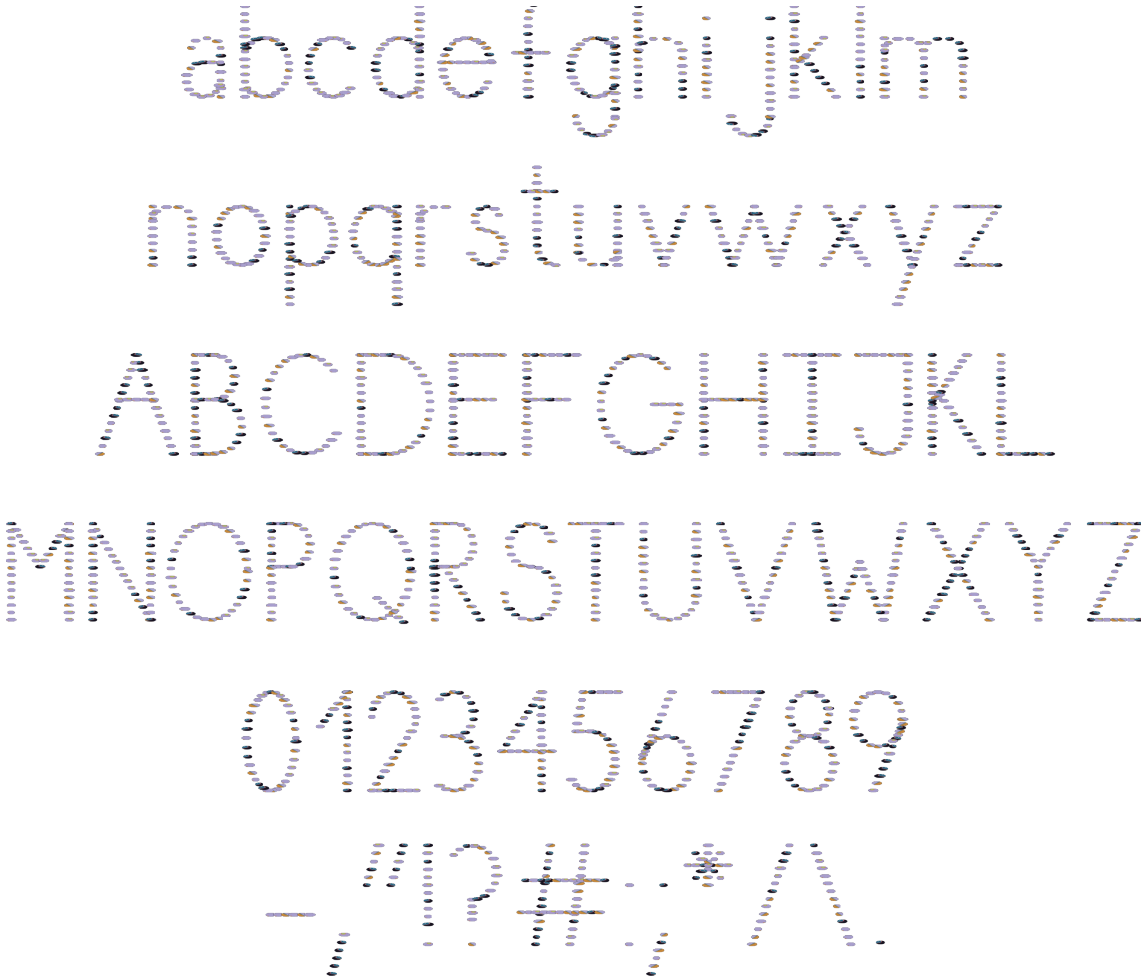
CROSS STITCH TEXTURE — SQUARE SHAPE

BASIC TEXTURE — CIRCLE SHAPE

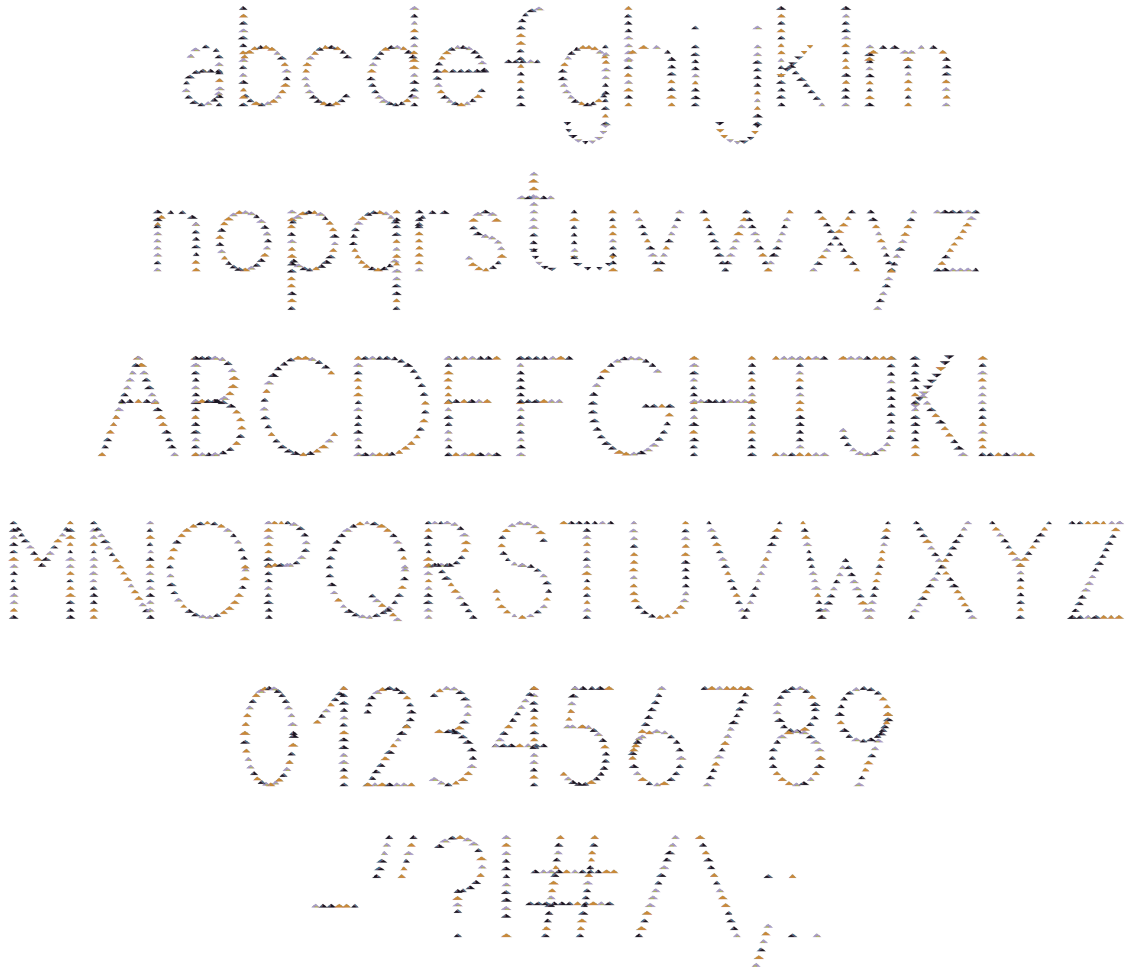


BASIC TEXTURE — TRIANGLE SHAPE

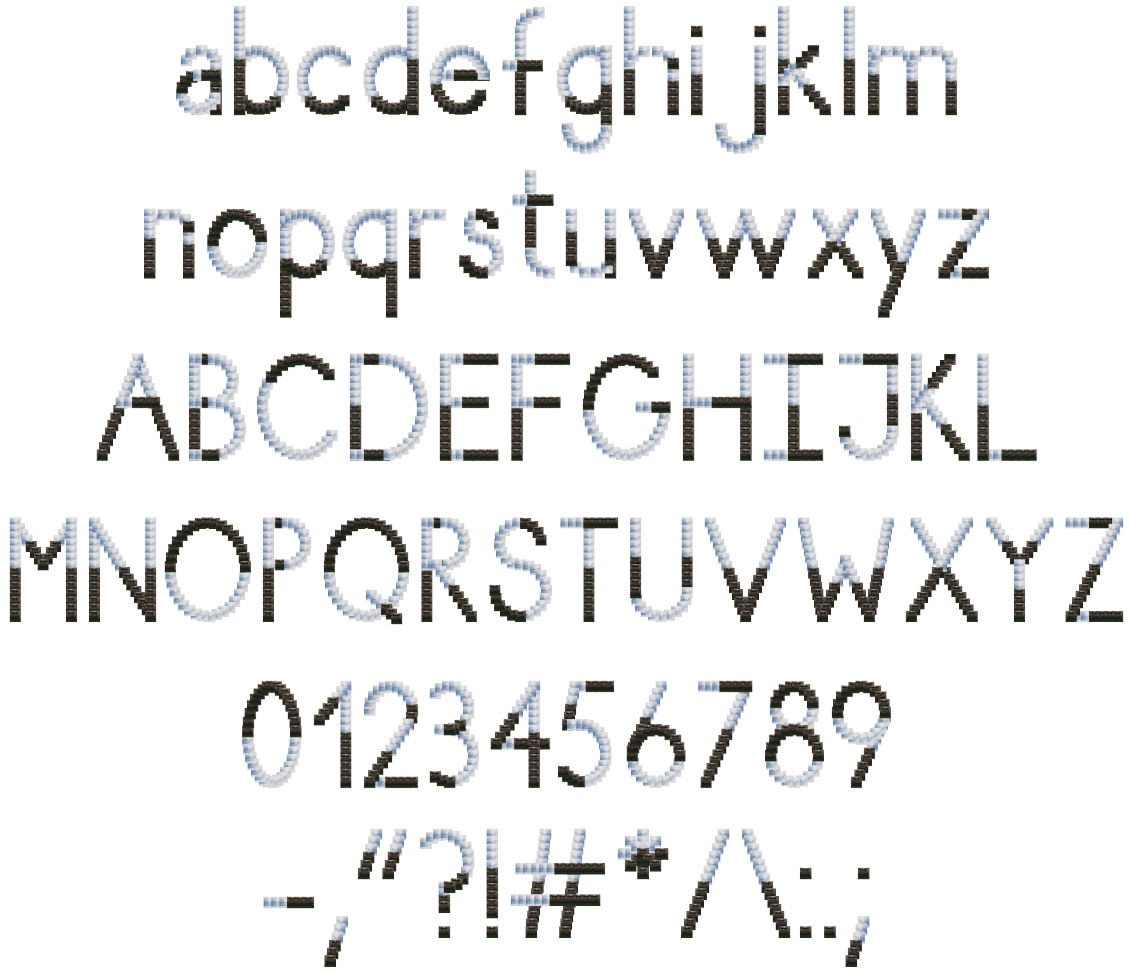
BOOK STACK TEXTURE — SQUARE SHAPE



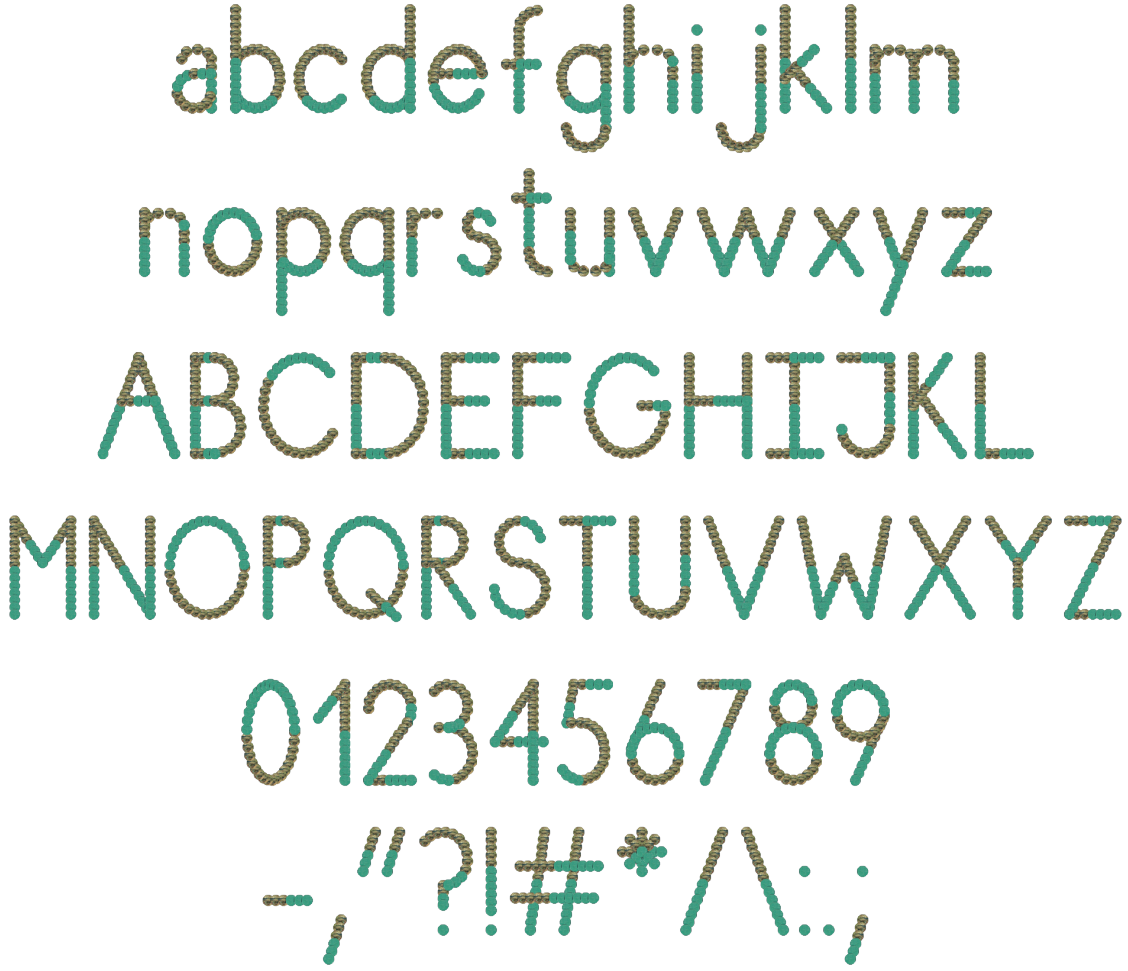
BOOK STACK TEXTURE — CIRCLE SHAPE



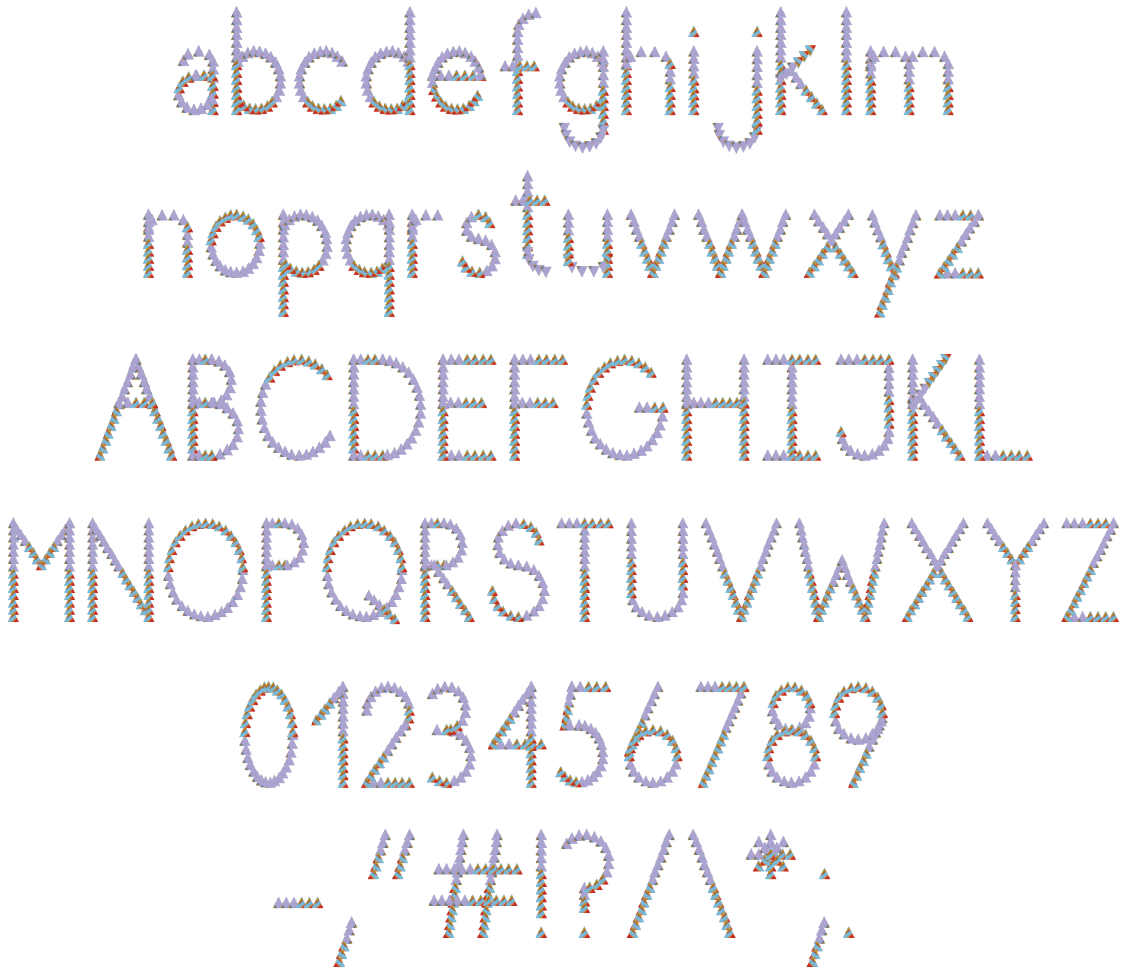
BOOK STACK TEXTURE — TRIANGLE SHAPE



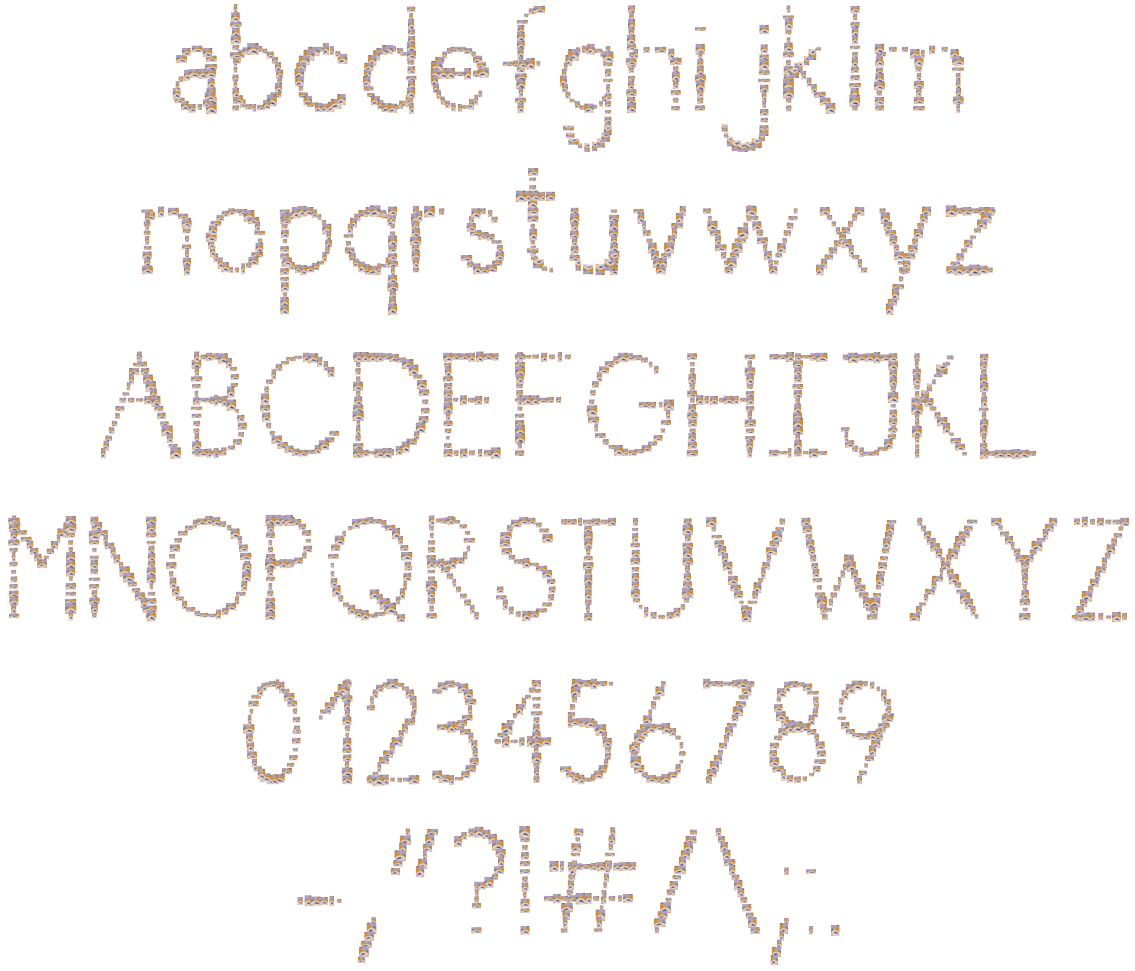
FIFTY FIFTY TEXTURE — SQUARE SHAPE



FIFTY FIFTY TEXTURE — CIRCLE SHAPE

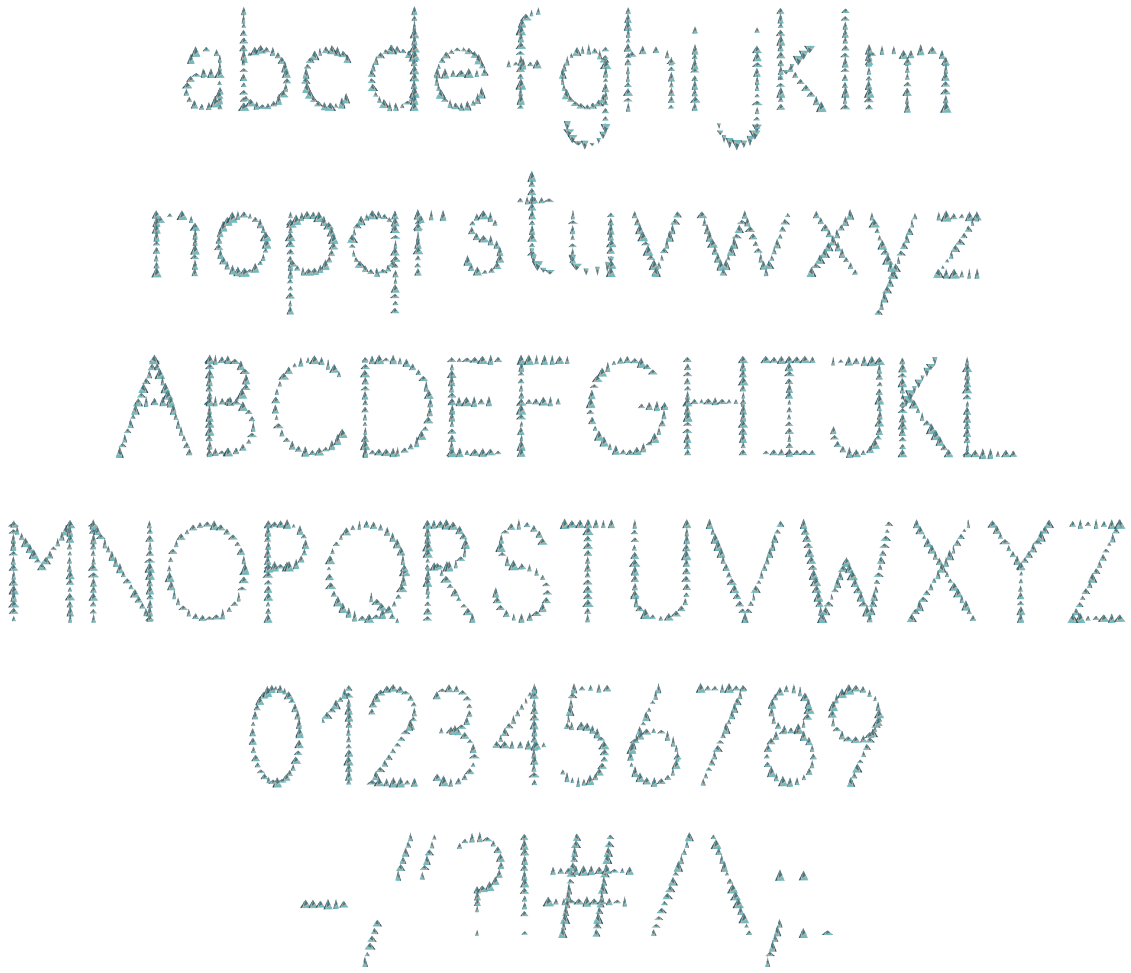


FIFTY FIFTY TEXTURE — TRIANGLE SHAPE

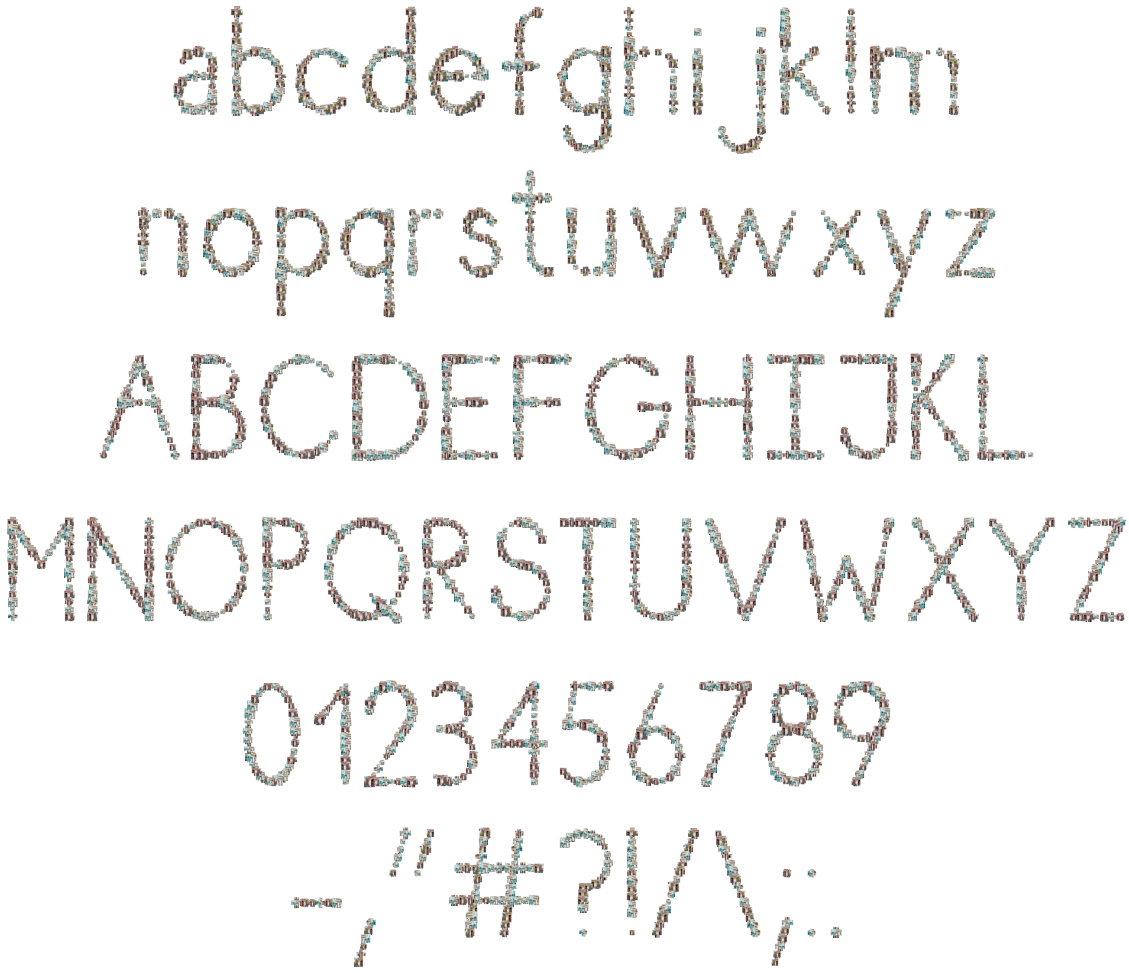


RANDOM SIZE TEXTURE — SQUARE SHAPE

RANDOM SIZE TEXTURE — CIRCLE SHAPE

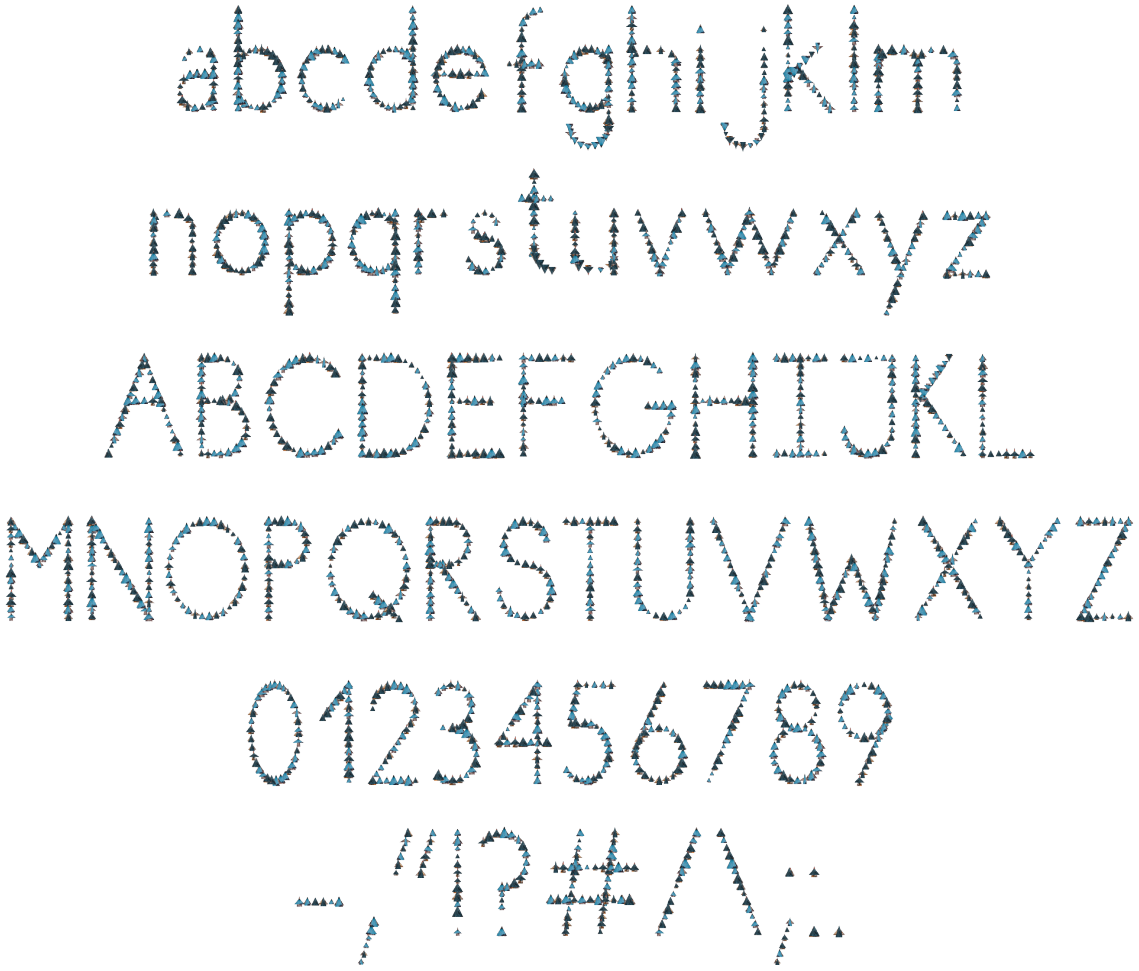


RANDOM SIZE TEXTURE — TRIANGLE SHAPE



OVERLAPPING TEXTURE — SQUARE SHAPE

OVERLAPPING TEXTURE — CIRCLE SHAPE

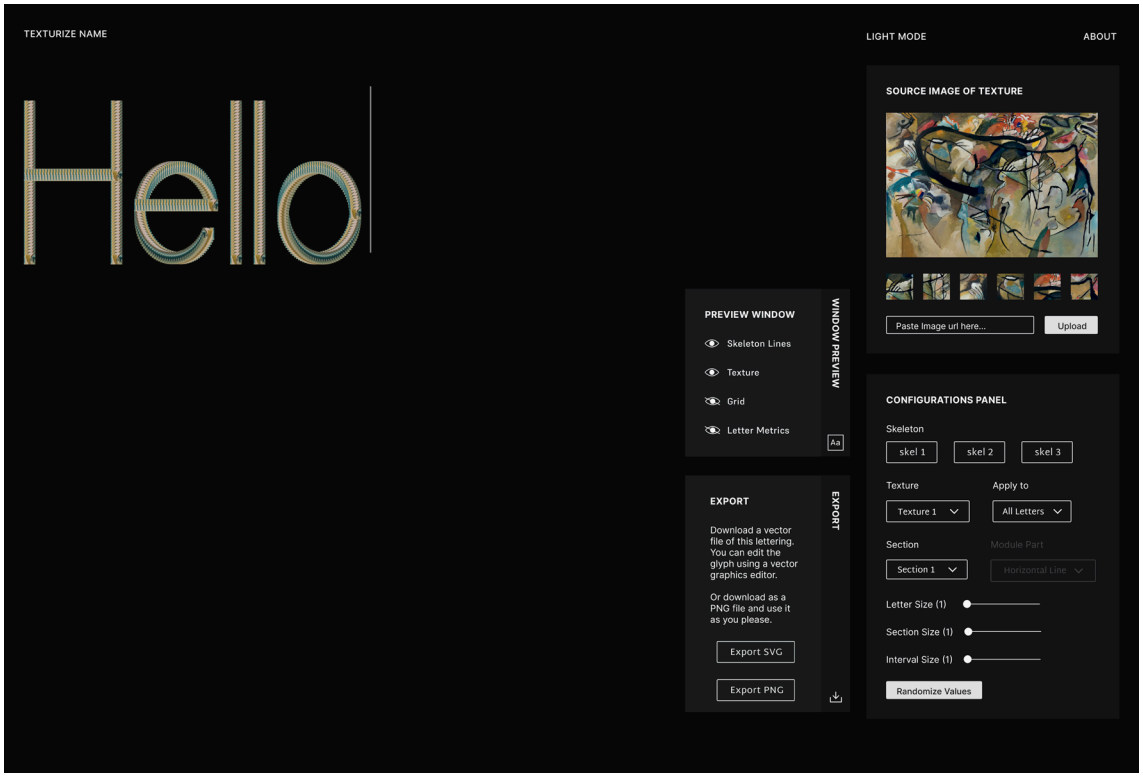


OVERLAPPING TEXTURE — TRIANGLE SHAPE

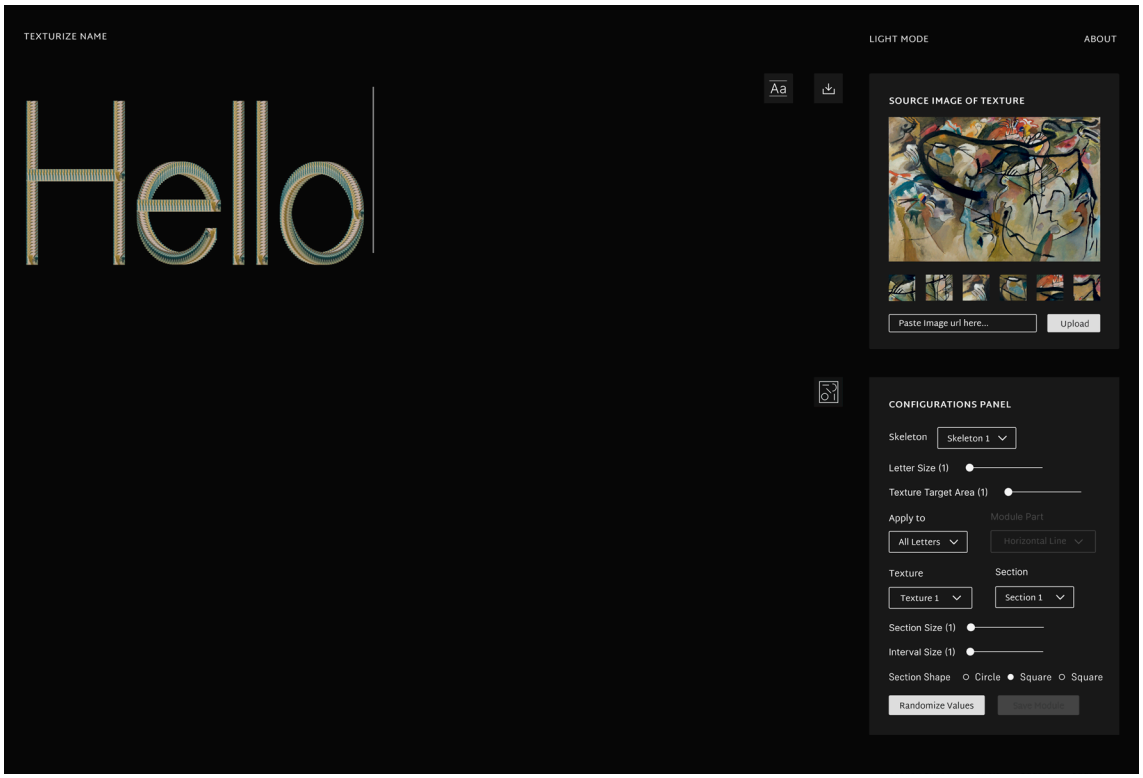
APPENDIX B

– INTERACTIVE PROTOTYPE

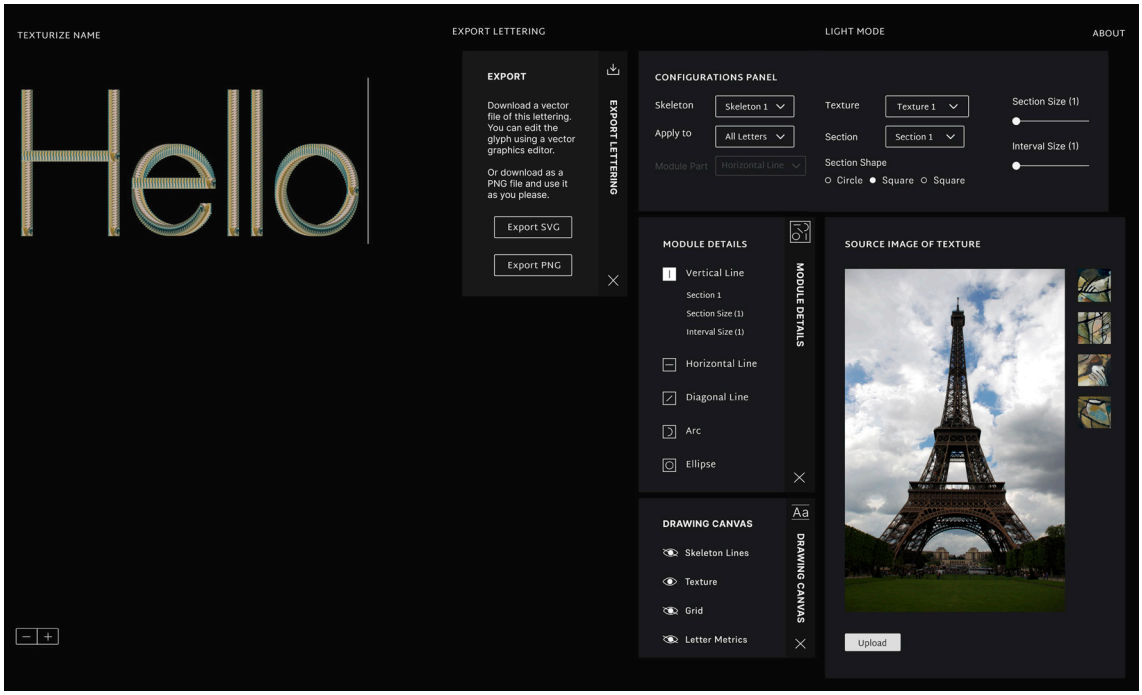
SCREENS OF FIRST ITERATION OF THE PROTOTYPE



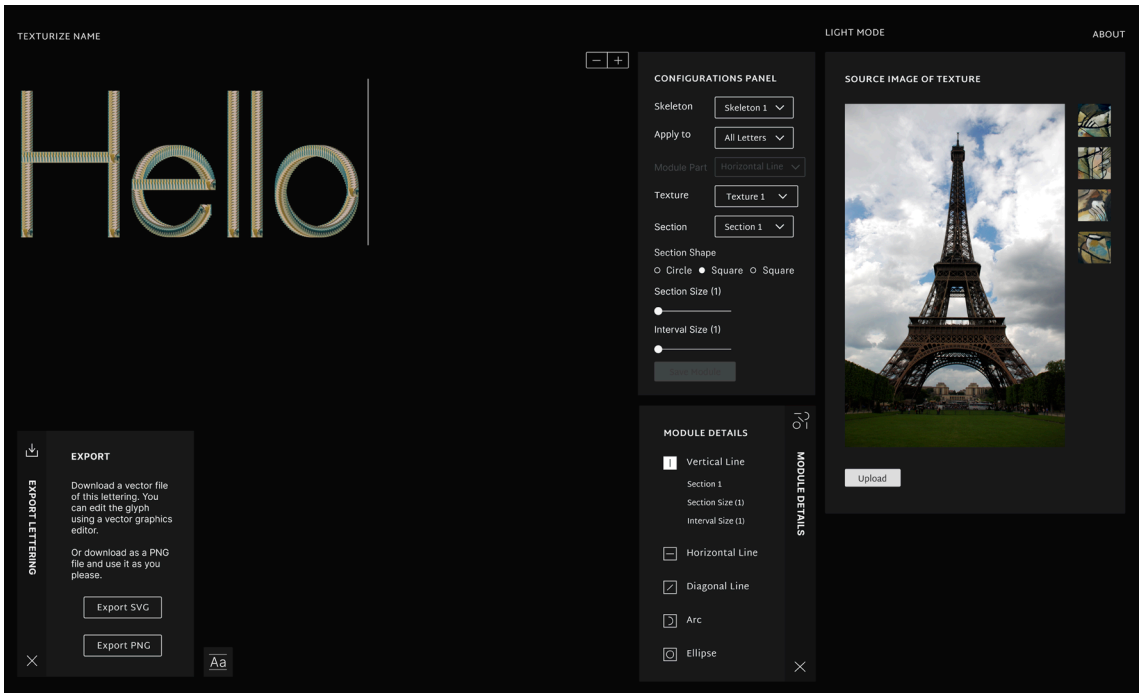
SCREEN B.1.



SCREEN B.2.



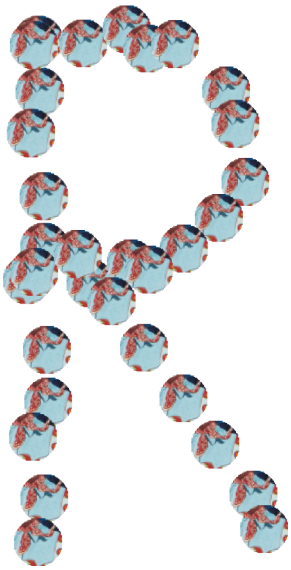
SCREEN B.1.

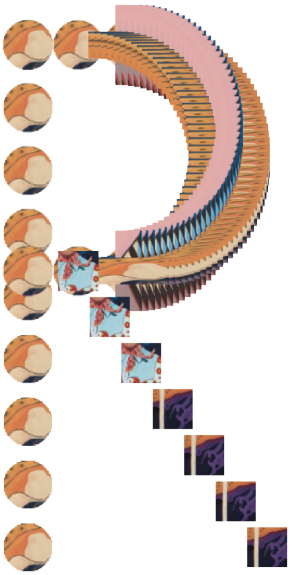


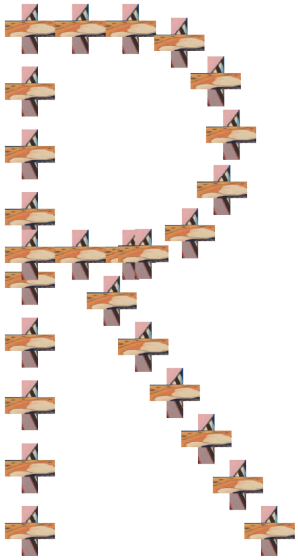
SCREEN B.2.

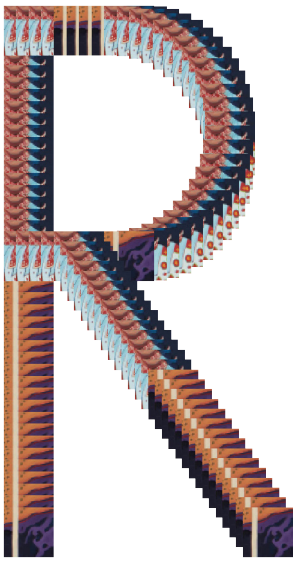
APPENDIX C – LETTERS

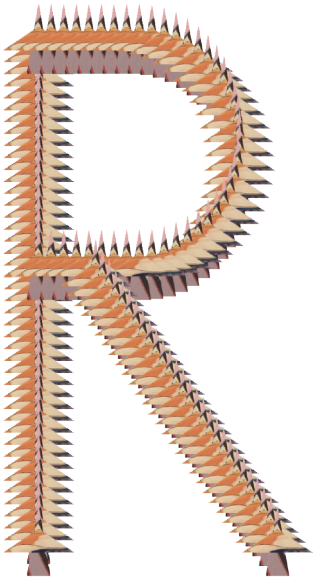
R'S FROM FIGURE 4.40.

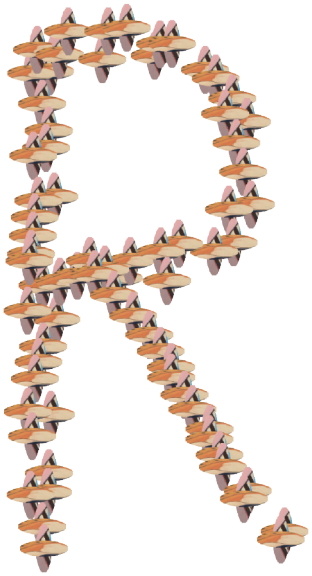


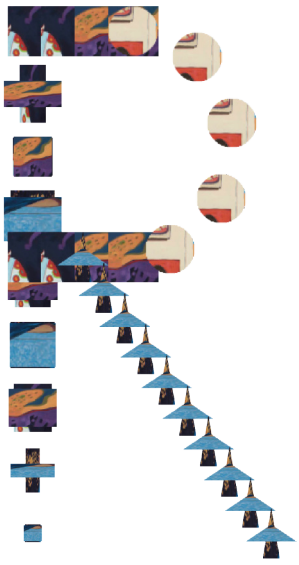


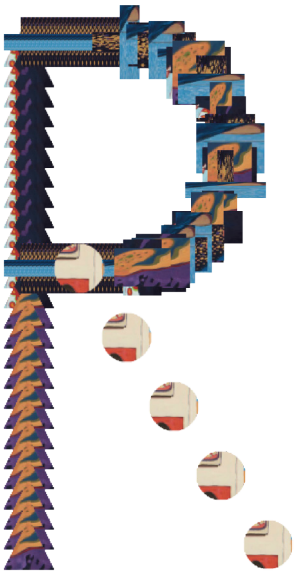




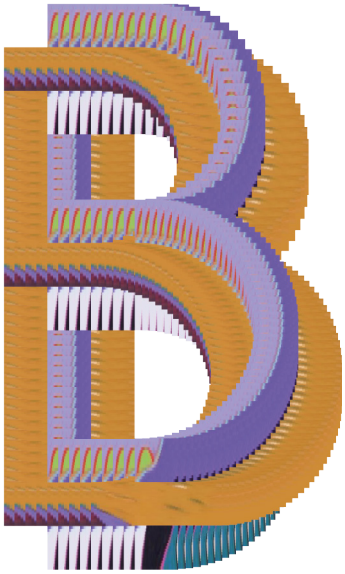


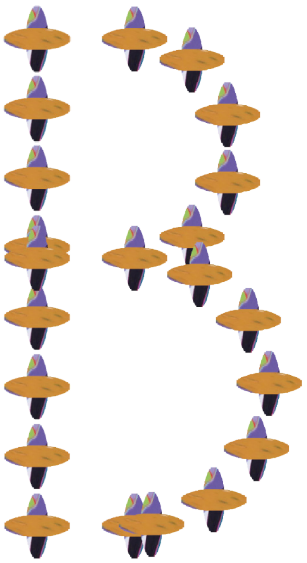


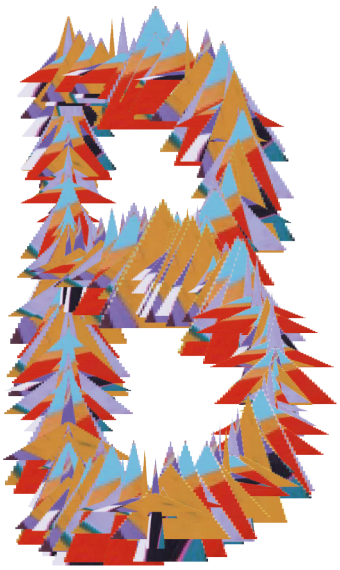


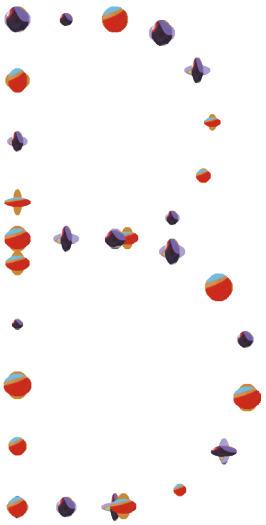


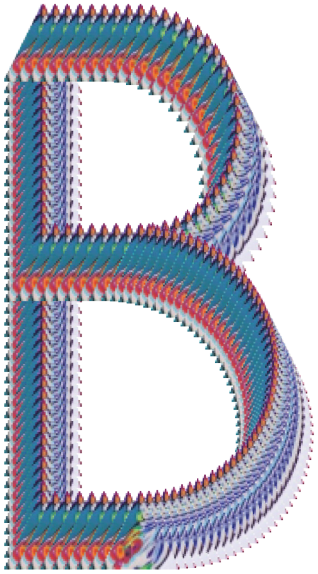
B'S FROM FIGURE 4.41.

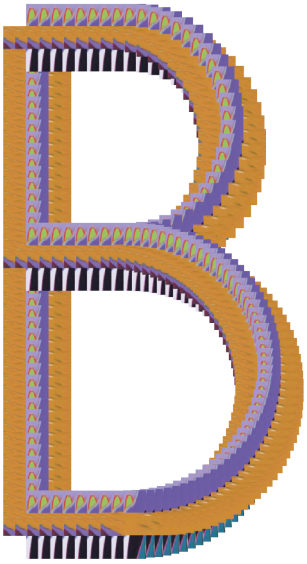


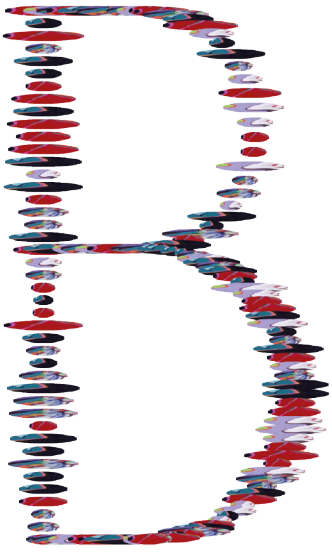


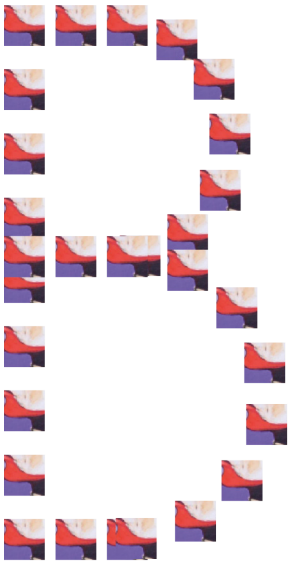


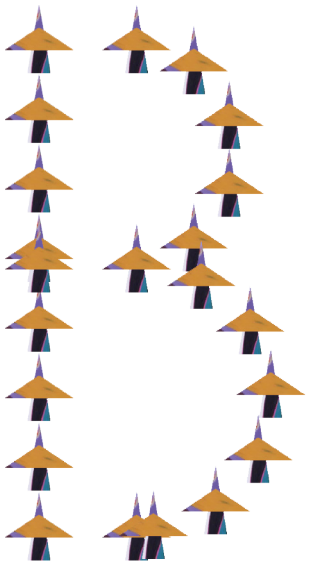


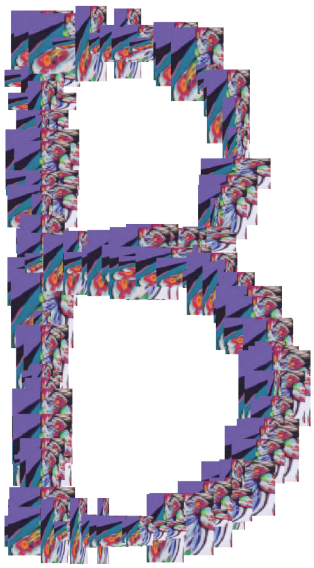


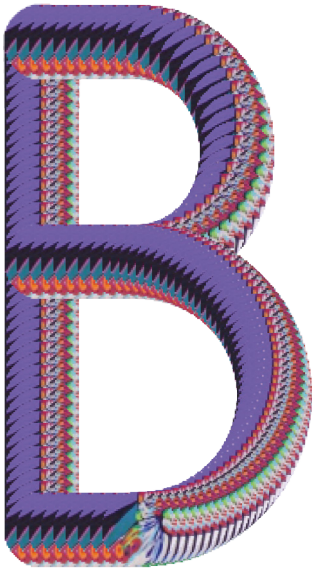


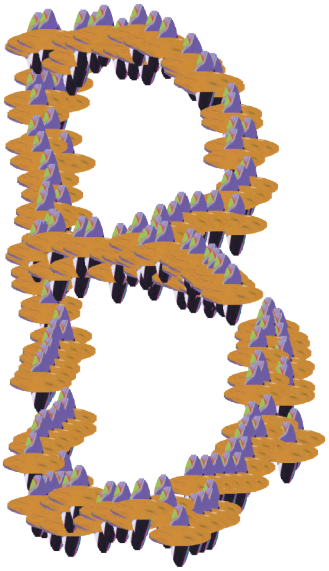


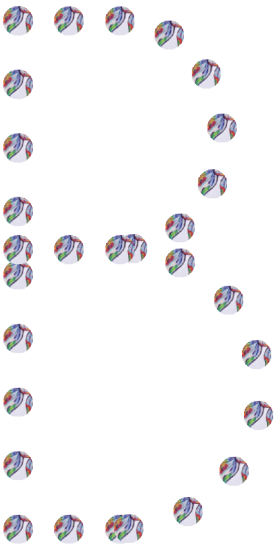


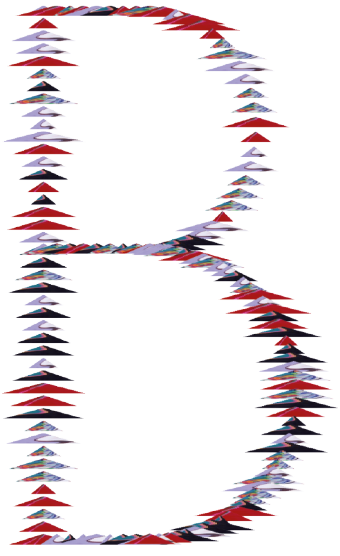












H'S FROM FIGURE 4.42.