

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

MATHEUS BRAGA RHODEN

**Assisted Typography: Streamlining Type
Design with Artificial Intelligence**

Work presented in partial fulfillment
of the requirements for the degree of
Bachelor in Computer Science

Advisor: Prof^a. Carla Maria Dal Sasso Freitas

Porto Alegre
April 2023

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Reitor: Prof. Carlos André Bulhões Mendes

Vice-Reitora: Prof^a. Patricia Helena Lucas Pranke

Pró-Reitor de Graduação: Prof^a. Cíntia Inês Boll

Diretora do Instituto de Informática: Prof^a. Carla Maria Dal Sasso Freitas

Coordenador do Curso de Ciência de Computação: Prof. Marcelo Walter

Bibliotecário-chefe do Instituto de Informática: Alexsander Borges Ribeiro

ABSTRACT

Type design is the craft of creating fonts, which are an essential part of all graphic communication. Type designers are busier than ever creating fonts for different applications, media, and audiences. Technology plays an important role in this process, and there are many tools that make type designers' lives much easier. Still, creating fonts can take months. On the other hand, there are exciting advances in generative AI for fonts. The aim of this work is to integrate these recent advances into the workflow of a type designer. To achieve this kind of integration, we propose Glyph-o-matic, a font editor plugin that leverages existing machine learning techniques to generate suggestions of letter forms. To evaluate its efficiency, we have conducted experiments with type designers and collected their feedback. The results we obtained show that, although Glyph-o-matic and its underlying machine learning model are not mature enough to be put to practical use, there is great potential for further collaboration between type designers and computer scientists.

Keywords: Type design. generative AI. user experience

RESUMO

Design de tipos é a atividade de criar fontes, as quais são uma parte essencial de toda comunicação gráfica. Designers de tipos estão mais ocupados do que nunca criando fontes para diferentes aplicações, mídias e públicos. A tecnologia desempenha um papel importante nesse processo, e existem muitas ferramentas que facilitam muito a vida dos designers de tipos. Ainda assim, criar fontes pode levar meses. Por outro lado, existem avanços animadores em IA generativa para fontes. O objetivo deste trabalho é integrar estes avanços recentes no fluxo de trabalho do designer de tipos. Para alcançar este tipo de integração, propomos o Glyph-o-matic, um plugin para editor de fontes que utiliza métodos de aprendizado de máquina existentes para gerar sugestões de formas de letras. Para avaliar sua eficiência, conduzimos experimentos com designers de tipos e coletamos suas opiniões. Os resultados que obtivemos mostram que, apesar do fato que o Glyph-o-matic e seu modelo de aprendizado de máquina subjacente não serem maduros o suficiente para serem colocados em uso prático, existe grande potencial para mais colaboração entre designers de tipos e cientistas da computação.

Palavras-chave: design de tipos. IA generativa. experiência do usuário.

LIST OF FIGURES

Figure 1.1 Diversity in type design.	8
Figure 2.1 Common shapes across glyphs.	12
Figure 2.2 Variations of the letter "A" across different typefaces.	13
Figure 2.3 Type design workflow and its four main phases: (1) conceptual, (2) design, (3) production and (4) post-production.	17
Figure 3.1 A typical font editor's overview screen where the type designer interacts with the font as a whole	18
Figure 3.2 A typical font editor's vector drawing screen where the type designer interacts with glyphs individually	19
Figure 3.3 The <i>Show stem thickness</i> plugin assists the type designer by showing how thick the hovered stem is.	20
Figure 3.4 The <i>Speed punk</i> plugin assists the type designer by enhancing Glyphs 3's curvature visualization.	20
Figure 3.5 A Zero-Shot Text-to-Image Generation example in which a GAN generates images of " <i>a tapir made of accordion</i> ".	20
Figure 3.6 MC-GAN can generate all of the majuscules in the same style of a reference subset, in this case: E, H, K, R and S.	21
Figure 3.7 Example of vector fonts generated by DeepVecFont given a few reference glyphs marked in red.	21
Figure 3.8 Example that illustrates the DeepVecFont limitations has when handling fonts with thin shapes.	22
Figure 4.1 Glyph-o-matic fits into the architecture of Glyphs 3 through the Plugin Manager.	25
Figure 4.2 Sequence diagram of the type design workflow showing two alternative scenarios: With Glyph-o-matic and Without Glyph-o-matic.	26
Figure 4.3 Screenshots of how the user can interact with Glyph-o-matic to generate glyphs based on their original drawings.	27
Figure 5.1 Pie chart: familiarity with Glyphs 3.	29
Figure 5.2 Pie chart: familiarity with plugins for Glyphs 3.	29
Figure 5.3 Bar chart: ease of use of the plugin.	30
Figure 5.4 Bar chart: usefulness of the suggestions generated by the plugin.	30
Figure 5.5 Bar chart: confidence about using the plugin.	31
Figure 5.6 Bar chart: required knowledge to use the plugin.	31
Figure 5.7 Bar chart: willingness to use the plugin again.	31

LIST OF TABLES

Table 1.1 Top 5 most commonly used writing systems.	9
Table 3.1 Results of the user study conducted to evaluate fonts generated by Deep-VecFont.	22

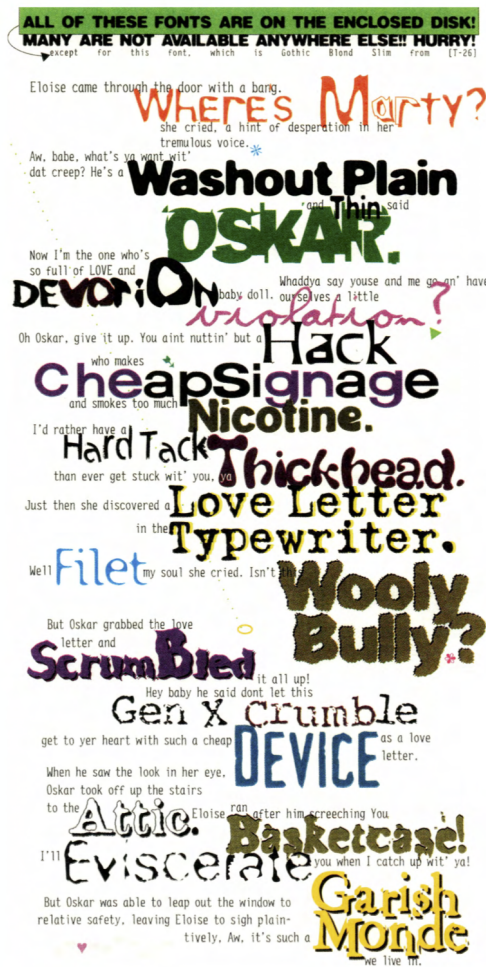
SUMMARY

1 INTRODUCTION	8
1.1 Objective and Scope	9
1.2 Structure of the work	10
2 BACKGROUND	11
2.1 The morphology of letters	11
2.2 Type quality assurance	13
2.3 The type design workflow	14
2.3.1 The concept stage.....	15
2.3.2 The design stage.....	15
2.3.3 The production stage.....	16
2.3.4 The post-production stage.....	16
3 RELATED WORK	18
3.1 Font editors	18
3.2 Artificial intelligence and type design	19
4 GLYPH-O-MATIC	23
4.1 Development	23
4.2 Architecture	24
4.3 Interaction flow	25
5 EVALUATION	28
5.1 Method	28
5.2 Task	28
5.3 Results	29
6 CONCLUSION AND FUTURE WORK	32
REFERENCES	33
APPENDIX A — QUESTIONNAIRE	35

1 INTRODUCTION

According to Unger (2018), *"of all designed objects, letters are probably the most pervasive, very familiar yet amazingly diverse in their appearance"*. Figure 1.1 illustrates such a diversity showing experimental fonts.

Figure 1.1 – Diversity in type design.



Source: Unger (2018).

Even though there are thousands of fonts, new ones are created everyday. Nowadays, designers must keep up with fast-changing trends by gathering inspiration from the world around them and creating fonts that fit our ever-changing needs.

One might ask: why create new fonts? In that sense, Henestrosa, Meseguer e Scaglione (2020) states that *"What happens in other similar activities: why make a new song, or a new novel, or a new movie? Luckily there are people that make new attempts and the results are remarkable. We would have lost a great deal if no one had made rock*

music after the Beatles."

Also, recent advances in technology have increased the popularity and accessibility of font creation (CHENG, 2020), and in recent years, this area has gotten the attention of people from other fields of study (TRACY, 2003). Computer scientists, for example, are using technology not only to streamline the creation of fonts, but to automate parts of the process. Such technological advances are likely to be useful for designers.

This work aims to bridge the gap between designers and promising technological advancements that have so far mostly been explored from an academic perspective.

1.1 Objective and Scope

The objective of this work is to provide designers with a tool that allows them to create fonts more efficiently. More specifically, it aims to optimize the designer's workflow by enhancing the software they already use with artificial intelligence.

The scope of this work includes the design, development and evaluation of a plugin - Glyph-o-matic - that automates specific tasks of the process of creating fonts. This work does not present new methods for design automation. Instead, it explores the usage of existing methods as tools for the designer without disrupting their workflow nor limiting their creativity.

We decided to focus our work on the Latin script, even though many notable writing systems have been used around the world. One of the reasons for this choice is that nearly 70% of the world's population use the Latin alphabet according to Vaughan (2020) (Table 1.1), which makes it the most popular writing system. Other than that, scripts such as the Chinese one are much broader and more complex than the Latin script, which makes it overly challenging to provide a solution that would cover both and possibly others.

Table 1.1 – Top 5 most commonly used writing systems.

<i>Script</i>	<i>Users</i>
Latin	5.6 billion
Chinese	1.3 billion
Arabic	660 million
Devanagari	600 million
Bengali	300 million

Source: (VAUGHAN, 2020).

Moreover, this work will focus on text fonts, which are fundamentally different from display fonts. According to Unger (2018), text fonts are meant for sustained reading

of large amounts of text in small sizes (e.g., books). On the other hand, display fonts are meant for small amounts of text in large sizes (e.g., publicity). This work will focus on text fonts because the process of designing them relies heavily on convention, which seems to be easier to model with artificial intelligence.

1.2 Structure of the work

This work is organized as follows. Chapter 2 summarizes the type design workflow, providing a background on the topic. Chapter 3 presents related work. Chapter 4 describes the design, development, and the proposed plugin itself, i.e., its architecture and features. Chapter 5 analyzes the results obtained with the use of the proposed plugin by an expert user. Finally, chapter 6 concludes this work and reflects on opportunities for future work.

2 BACKGROUND

This chapter summarizes the process of designing type. More specifically, it starts by describing how the letters of the Latin alphabet are constructed. Then, it describes how type designers assess the quality of their work. Finally, it establishes the steps involved in designing type.

In order to facilitate reading, at first, we introduce definitions for terms that are used throughout this work. The following list is a subset of the glossary presented by Unger (2018).

- **Font:** a font is a complete set of characters in one single style or weight: for instance, all the characters in the light roman version of a particular typeface. In the days of metal type, it was a complete set of characters for a particular variant cast at a specific size.
- **Glyph:** the particular version of a character. The basic shape of the A, for example, appears in many different guises as glyphs. Type designers make glyphs.
- **Type:** this was the term used for metal letters to be composed for letterpress printing. It referred to them collectively or generally, as in typesetting or a typewriter. It is now used also for digitally made letters.
- **Type design:** the design of typographic letterforms and other signs; also referred to as typeface design. The fact that 'type' is the object of design implies that letters per se are not the purpose, but rather unified groups of letters and other typographic characters (fonts and groups of fonts).
- **Typeface:** a set of fonts of the same or related design.

2.1 The morphology of letters

The Latin alphabet is largely based on three geometric shapes (CHENG, 2020):

- Circles (O/o, C/c, e, etc.)
- Triangles (A, V/v, W/w, etc.)
- Squares (N/n, H/h, E, etc.)

Therefore, the type designer can start by designing a key character of each geometric group. Then, it is possible to derive many of the remaining designs based on

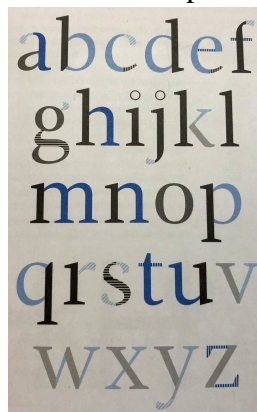
morphological similarity, which saves time and effort. (HENESTROSA; MESEGUER; SCAGLIONE, 2020)

In practice, many type designers begin with 'n', 'o', and 'v' because these letters correspond to basic geometric shapes and provide most of the information required to derive other letters. For instance, letters such as 'h', 'm', 'u', and 'r' can be derived from an 'n' with relative ease. Moreover, the letter 'a' can be a useful choice since it is the letter that contributes the most to the distinctiveness of the typeface. Aside from these conventional choices, the order varies largely according to the type designer's preference (HENESTROSA; MESEGUER; SCAGLIONE, 2020).

One of the benefits of deriving letters from others is that it helps the type designer create a cohesive typeface. In that regard, Parker (n.d. apud UNGER, 2018) states that the aim of the type designer is *"to create a beautiful group of letters and not a group of beautiful letters"*. Moreover, considering the roots that typography has on calligraphy, Cheng (2020) argues that all glyphs of a typeface should appear to be made by the same instrument. In other words, consistency is a key factor for type design quality.

In practice, the type designer achieves consistency by repeating common elements across glyphs (HENESTROSA; MESEGUER; SCAGLIONE, 2020). Figure 2.1 illustrates how shapes are similar across glyphs, such as verticals, diagonals and bowls. Most characters can be built by repeating and sensibly modifying such fundamental strokes.

Figure 2.1 – Common shapes across glyphs.

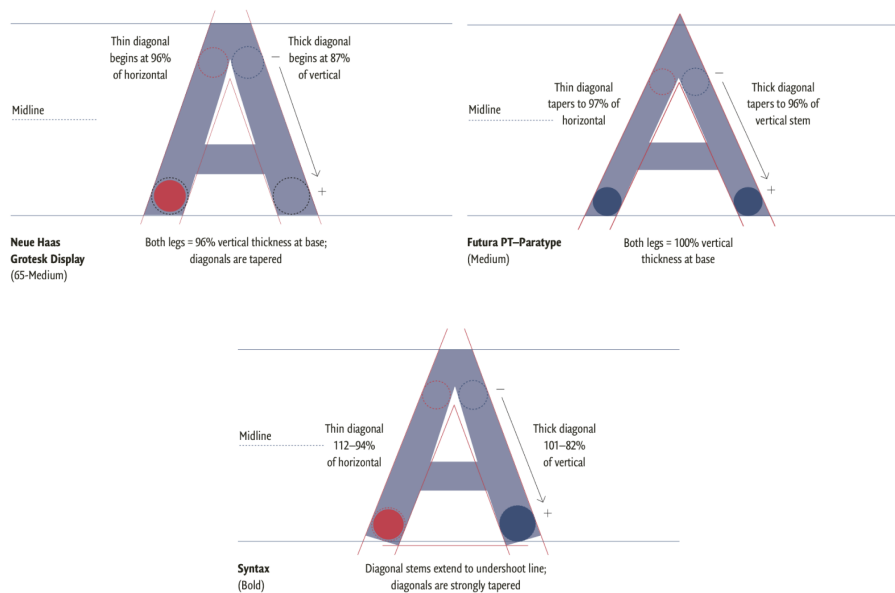


Source: Henestrosa, Meseguer e Scaglione (2020).

On the other hand, as stated by Unger (2018), *"[...] a type design cannot be based solely on convention, which serves instead as a flexible boundary you can respect, push, or climb over"*. Therefore, it is up to the type designer to balance distinctiveness and readability. The designer might want to design easily distinguishable fonts or improve their readability by changing spacing, for example.

Tracy (2003) adds that a typeface's identity is built on top of the structural features of letters. For example, a type designer may conform to structural convention and still produce a distinctive typeface by experimenting with different combinations of stroke weight, contrast, stress, and serif shapes. Figure 2.2 illustrates how such combinations are explored by three different typefaces.

Figure 2.2 – Variations of the letter "A" across different typefaces.



Source: Cheng (2020).

It is also important to consider that many fonts are part of font families, in which the same overall design is replicated across different weights (e.g., light, medium and bold). Cheng (2020) argued that, in the past, type designers had to draw each weight of a font family manually, which used to be a burdensome process. Nowadays, font editors can interpolate two or more variants and generate one or more intermediates. These variants are referred to as "masters", which are available in most modern font editors. Also, multiple masters have become essential to the development of font families with many variants (HENESTROSA; MESEGUER; SCAGLIONE, 2020).

2.2 Type quality assurance

The quality of a typeface can be assessed based on four criteria (UNGER, 2018):

- Functionality: how well it fulfills its main purpose;
- Social quality: how widely legible it is within its audience;
- Creativity: how aesthetically original it is;
- Expressivity: how well it conveys the desired emotive response.

When drawing letters, the type designer constantly evaluates how well they fit these criteria. According to Henestrosa, Meseguer e Scaglione (2020), *"Testing the fonts can only be done by composing words and not alphabetical sequences - 'a,b,c', etc. - as letters only acquire meaning through words, which at the same time allow us to assess and evaluate the relationships between characters"*. Cheng (2020) adds that a test word contains all the different forms in the Latin alphabet. Although they are arbitrary, examples of commonly used test words include *Adhesion*, *Hamburgerfontsv* and *Handgloves*. As the design progresses, it is useful to expand testing to larger portions of text, which allows for more coverage of combinations of letterforms.

Although computer programs are suitable for arithmetic calculations, the designer's eye is still the best source of aesthetic judgement (TRACY, 2003) and, as stated by Henestrosa, Meseguer e Scaglione (2020), *"it is justified to think that mathematics plays a lesser part than a critical eye"*.

For example, type designers perform subtle optical adjustments to make their letterforms more aesthetically pleasant (CHENG, 2020). However, there seem to be no reliable ways to calculate such adjustments, which instead are made arbitrarily by type designers based on their optical judgement (UNGER, 2018).

2.3 The type design workflow

According to Unger (2018), *"Type changed from tangible objects into immaterial images, and the time to create and produce a type design has shrunk drastically"*. However, designing a typeface family can still require months or even years of intensive effort depending on the number of variants (e.g., black, italic, thin, etc). The type designer relies on constant proofing and correcting to avoid introducing changes at advanced stages, which tend to be time-consuming (HENESTROSA; MESEGUER; SCAGLIONE, 2020).

As software advanced, the type designer has acquired attributions that otherwise would fall to specialized colleagues. Such attributions include, for example, preparing fonts for various applications (e.g., print or web) and extending character sets for vari-

ous languages. Still, type designers do collaborate with other graphic designers and the interaction between these actors can become complicated when the subjective expressive quality of the typeface is discussed (UNGER, 2018).

The process of type design can be divided into four consecutive stages (Figure 2.3): concept, design, production and post-production, which will be briefly described in the next sections.

2.3.1 The concept stage

In the concept stage, the type designer starts by devising a design brief that, as stated by Cheng (2020), should answer at least the following questions:

- What is the intended function of the typeface?
- In what media will it be used?
- What languages are needed?
- What personality should the typeface convey?
- What design characteristics are needed?
- What typographic style is desired?

Based on the guidelines defined in the design brief, the type designer can proceed to sketch out ideas. Although they are free to sketch on paper or on screen, the drawing needs to move into digital form at some point (HENESTROSA; MESEGUER; SCAGLIONE, 2020). Experienced type designer tend to prefer to sketch on screen directly (CHENG, 2020).

2.3.2 The design stage

In the design stage, the type designer proceeds to draw all core glyphs, such as majuscules, minuscules, numbers, and symbols. To do so, they use their initial sketches to derive most core glyphs. Nevertheless, there are glyphs that need to be drawn from scratch. At this point, the type designer has to be extra mindful of the fact that the next stages of the workflow rely on the quality of these core glyphs.

Also, in the case of a typeface family with multiple variants, at least two of them are created during the design stage. Having two variants enable the type designer to as-

sess what kind of problems they might have when using very thick or very thin shapes, for example. Moreover having two variants is the bare minimum to perform interpolation/extrapolation of other variants.

2.3.3 The production stage

In the production stage, the type designer proceeds to draw secondary glyphs, such as ordinals, small capitals, and superscripts. This is accomplished mostly by copying and sensibly modifying the core glyphs that were created in the design stage. For example, small capitals can be achieved with relative ease by downscaling existing regular capitals.

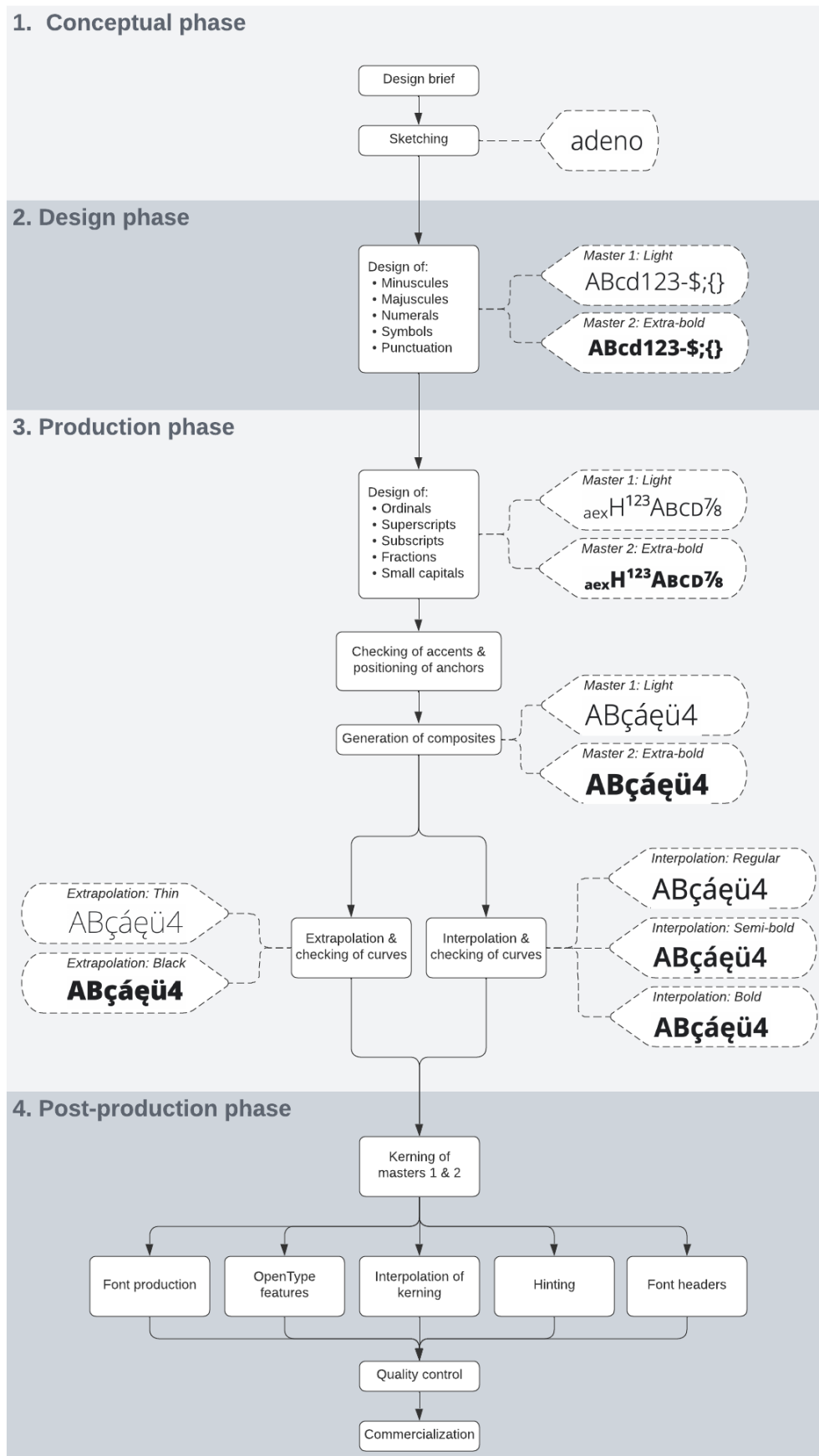
Moreover, at this point accents are designed and anchored to all glyphs to which they can be applied. The dimension of this task depends a lot on which languages are to be supported by the typeface.

Also, in the case of a typeface family with multiple variants, all remaining variants are generated via interpolation or extrapolation. It is worth noticing that the results of interpolation and extrapolation are not perfect. For example, an extrapolation with extremely thick strokes might cause the gap inside an uppercase *A* to disappear. Therefore, it is up to the type designer to review them and perform adjustments

2.3.4 The post-production stage

Finally, in the post-production stage, the type designer proceeds to perform more technical finishing touches such as kerning (i.e., customized spacing between pairs of characters), hinting (i.e., optimization for specific screen resolutions), and OpenType features (e.g., ligatures that combine glyphs like "ff" or "fi").

Figure 2.3 – Type design workflow and its four main phases: (1) conceptual, (2) design, (3) production and (4) post-production.



Source: adapted from Henestrosa, Meseguer e Scaglione (2020).

3 RELATED WORK

Over the years, type design has become a sophisticated technique that relies heavily on software. Technology has enabled a single designer with appropriate software to carry out tasks that used to require a large team.

According to Unger (2018), *"type designs carry all kinds of messages on top of their primary function, which is to be legible"*. Since technology has enabled type designers to do so much in so little time, the focus of developing a typeface has shifted to the abstract quality of type.

In this chapter, we briefly survey software and academic works that are related to our work.

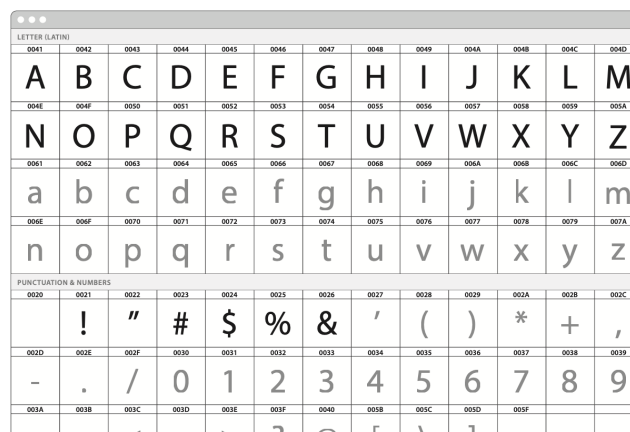
3.1 Font editors

A font editor is a computer program that type designers use to carry out most of their tasks. As illustrated in Figure 3.1, a font editor typically displays a matrix of glyphs, which are to be drawn progressively.

The type designer draws in the font editor itself using tools that are similar to those in other vector drawing programs (e.g., Adobe Illustrator), but are optimized for drawing glyphs. Figure 3.2 exemplifies a typical vector drawing tool.

Glyphs 3 (GLYPHS, 2023) and FontLab (FONTLAB, 2023) are among the best known and most popular commercial font editors. These programs have built-on fea-

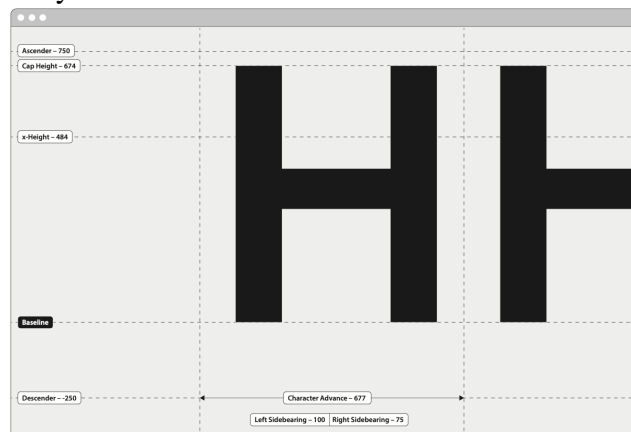
Figure 3.1 – A typical font editor’s overview screen where the type designer interacts with the font as a whole



LETTER (LATIN)												
0041	0042	0043	0044	0045	0046	0047	0048	0049	004A	004B	004C	004D
A	B	C	D	E	F	G	H	I	J	K	L	M
004E	004F	0050	0051	0052	0053	0054	0055	0056	0057	0058	0059	005A
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0061	0062	0063	0064	0065	0066	0067	0068	0069	006A	006B	006C	006D
a	b	c	d	e	f	g	h	i	j	k	l	m
006E	006F	0070	0071	0072	0073	0074	0075	0076	0077	0078	0079	007A
n	o	p	q	r	s	t	u	v	w	x	y	z
PUNCTUATION & NUMBERS												
0020	0021	0022	0023	0024	0025	0026	0027	0028	0029	002A	002B	002C
	!	"	#	\$	%	&	'	()	*	+	,
002D	002E	002F	0030	0031	0032	0033	0034	0035	0036	0037	0038	0039
-	.	/	0	1	2	3	4	5	6	7	8	9
003A	003B	003C	003D	003E	003F	0040	005B	005C	005D	005F		
.	.	/	-	\	?	@	[\				

Source: Cheng (2020)

Figure 3.2 – A typical font editor’s vector drawing screen where the type designer interacts with glyphs individually



Source: Cheng (2020)

tures that automate or semi-automate part of the type designer’s tasks that are less unpredictable, such as letter spacing. Both can be extended with plugins and Python scripts. Generally speaking, although FontLab is older than Glyphs 3 - they were first released in 1993 and 2011, respectively - the latter has a more mature community in terms of documentation, SDKs and support.

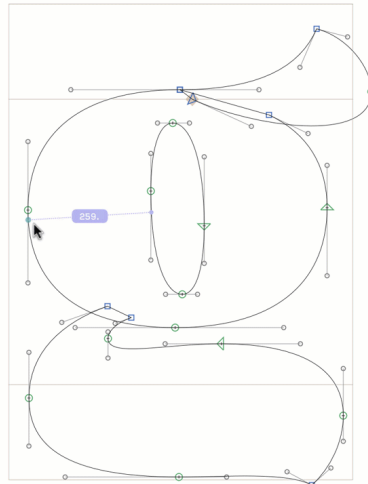
Therefore, we have decided to use Glyphs 3 as the font editor in which our plugin is integrated, as detailed in Chapter 4. At the time of writing of this work, Glyphs 3 has hundreds of plugins written by a large community. Among the most popular, we find:

- **Show stem thickness**: a tool shows how thick is the stem in a pointed place - illustrated in Fig. 3.3 (BUCHNER, 2023);
- **Speed punk**: a curvature illustration plug-in - illustrated in Fig. 3.4 (YANONE, 2023);
- **Kern on**: a semi-automated kerning assistant (AHRENS, 2023).

3.2 Artificial intelligence and type design

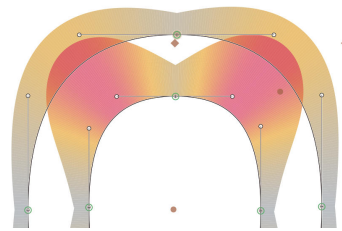
Lately, there have been significant advances in AI image generation, most of which based on Generative Adversarial Networks (GANs). Introduced by Goodfellow et al. (2014), GANs are based on the interaction between a generative model and a discriminative model. Figure 3.5 illustrates how the method proposed in Ramesh et al. (2021) - which originated DALL-E, one of the most popular commercial models of this kind - is able to generate images of "a tapir made of accordion".

Figure 3.3 – The *Show stem thickness* plugin assists the type designer by showing how thick the hovered stem is.



Source: (BUCHNER, 2023).

Figure 3.4 – The *Speed punk* plugin assists the type designer by enhancing Glyphs 3's curvature visualization.



Source: (YANONE, 2023).

Figure 3.5 – A Zero-Shot Text-to-Image Generation example in which a GAN generates images of "a tapir made of accordion".



Source: (RAMESH et al., 2021).

Figure 3.6 – MC-GAN can generate all of the majuscules in the same style of a reference subset, in this case: E, H, K, R and S.

Ground Truth	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
T-Effect:0.18	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
MC-GAN:0.82	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Source: (AZADI et al., 2017).

Figure 3.7 – Example of vector fonts generated by DeepVecFont given a few reference glyphs marked in red.

DeepVecFont (w/o refinement)	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <i>a b c d e f g h i j k l m n o p q r s t u v w x y z</i>
DeepVecFont (w/ refinement)	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <i>a b c d e f g h i j k l m n o p q r s t u v w x y z</i>
Ground Truth	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <i>a b c d e f g h i j k l m n o p q r s t u v w x y z</i>
DeepVecFont (w/o refinement)	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <i>a b c d e f g h i j k l m n o p q r s t u v w x y z</i>
DeepVecFont (w/ refinement)	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <i>a b c d e f g h i j k l m n o p q r s t u v w x y z</i>
Ground Truth	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <i>a b c d e f g h i j k l m n o p q r s t u v w x y z</i>

Source: (WANG; LIAN, 2021).

Azadi et al. (2017) introduced MC-GAN, a GAN-based model that is capable of generating all of the uppercase characters of the Latin alphabet based on a subset of glyphs, as shown in Fig. 3.6. Other works present similar methods for more complex writing systems, such as Chinese (PARK et al., 2020; ZENG et al., 2020; TANG et al., 2022) and Korean (CHA et al., 2020). Although visually pleasant, the results generated by these models are images, which cannot be easily consumed for practical usage (e.g., editing and distribution).

DeepVecFont, proposed by Wang e Lian (2021), is a novel method for automatic font generation based on deep learning that works with vector fonts (i.e., points, lines and curves) rather than images. The key idea of DeepVecFont is to exploit the dual-modality information (i.e., raster images and vector outlines) of vector fonts. As exemplified in Figure 3.7, its results are comparable to human-designed glyphs. However, this technique has limitations related to highly stylized fonts, as shown in Figure 3.8.

The evaluation of DeepVecFont included a user study that consisted of a Turing test in which respondents (both designers and non-designers) had to distinguish between generated fonts and human-designed fonts. Subsequently, they were asked to rate (1-5)

Figure 3.8 – Example that illustrates the DeepVecFont limitations has when handling fonts with thin shapes.



Source: (WANG; LIAN, 2021).

the generated fonts in terms of quality and style consistency. The results are shown in Table 3.1.

Table 3.1 – Results of the user study conducted to evaluate fonts generated by DeepVecFont.

<i>User Group</i>	<i>Precision</i>	<i>Recall</i>	<i>Accuracy</i>	<i>Quality</i>	<i>Consistency</i>
Designers	83.7%	41.2%	66.5%	3.4	4.5
Non-designers	67.1%	24.3%	55.6%	4.6	4.8
Average	77.0%	34.4%	62.1%	3.9	4.6

Source: (WANG; LIAN, 2021).

Although these advances are exciting, most of the mentioned studies do not offer much insight on how type designers can benefit from them. Therefore, the aim of this work is not to propose a technique that is alternative to them. Instead, this work uses one of them - more specifically, DeepVecFont (see Chapter 4) - as part of a plugin for type designers to use and evaluate how helpful AI-generated glyphs are within their workflow.

4 GLYPH-O-MATIC

Based on what we learned about typography and the process of type design through the works of Unger (2018), Henestrosa, Meseguer e Scaglione (2020), and Cheng (2020) (see Chapters 2 and 3), we found out that we could make a contribution to the type designers' community by developing Glyph-o-matic, which is a plugin for Glyphs 3 . Therefore, the methodology for this work consists of the development of Glyph-o-matic and its evaluation with users, which is reported in Chapter 5.

4.1 Development

First, we met with a type designer whom we already knew. We approached him with information about our work, and he agreed to collaborate with us. Together, we gathered requirements that would fit the scope of this work. We presented the idea of using artificial intelligence to generate glyphs based on a few original designs, and he thought that could be a useful tool in the initial stages of a type design project. Therefore, we established the following use case for our plugin from the perspective of the type designer: *Given I have drawn a few glyphs, when I apply the plugin using my original designs and targeting glyphs I would like to draw next, then I would like suggested letterforms to be generated for the target glyphs.*

The feature that our plugin introduces to Glyphs 3 is the ability to select input glyphs (i.e., glyphs drawn by the type designer) and output glyphs (i.e., glyphs to be generated by Glyph-o-matic) so that the underlying DeepVecFont model can infer on the provided data.

Glyph-o-matic is written in the Python programming language and powered by DeepVecFont (WANG; LIAN, 2021). As detailed in section 3.2, DeepVecFont generates glyph images based on a few original designs provided by the type designer. The author of DeepVecFont has made the source code available along with instructions on how to train and test the model. For the sake of convenience, they also provided a pre-trained model which saved us the time and effort of having to train DeepVecFont against a large font database.

As for the plugin development, Glyphs 3 provides an SDK with templates for the following categories of plugins (GERNER, 2016):

- **File format:** exporting the new font formats;
- **Filter without dialog:** change the font through a Filter menu or upon export;
- **Filter with dialog:** same as above, with the use of a user interface dialog;
- **General plug-in:** no special purpose;
- **Palette:** add a palette view to the side bar;
- **Reporter:** draw into the Edit view and Preview to illustrate features of the new glyphs;
- **Select Tool:** enhance the Select Tool with new features.

Glyph-o-matic is based on the *Filter with dialog* template because it changes the font based on a user interface component. As recommended in the Glyphs 3's SDK documentation, we use Cocoa for developing the dialog. Cocoa is Apple's own application environment. Combined with Xcode, which is Apple's integrated development environment for macOS, Cocoa makes it easy to create a well-factored, full-featured application (COCOA, 2013). To build the user interface component itself, we use Interface Builder (IB) (XCODER, 2011), which is a visual design tool that is part of Xcode. According to the user documentation, this tool allows the developer to assemble windows, views, controls, menus, and other elements from a library of configurable objects.

4.2 Architecture

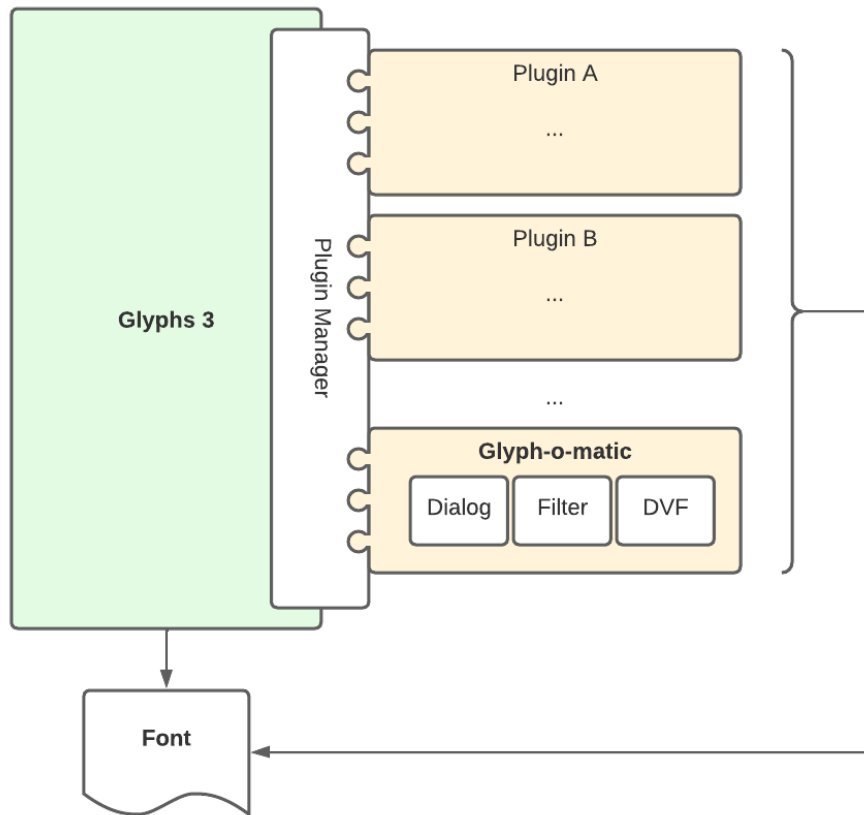
As illustrated in Figure 4.1, Glyph-o-matic integrates into Glyphs 3 through the Plugin Manager, just like any other plugin of this software.

Moreover, Glyph-o-matic is composed of three building blocks:

- **Dialog:** a user interface component that allows the user to configure and run the automated glyph generation;
- **DeepVecFont:** a pre-trained model that is able to generate glyphs based on examples provided by the user (see Section 3.2);
- **Filter:** a middleware component that invokes DeepVecFont based on user input provided in the Dialog component.

We proposed this division to ensure we have decoupled components. For example, we are able to replace DeepVecFont with an alternative model in the future without modifying the other components of the plugin.

Figure 4.1 – Glyph-o-matic fits into the architecture of Glyphs 3 through the Plugin Manager.



4.3 Interaction flow

The diagram shown in Figure 4.2 illustrates the flow of events in a scenario where a type designer is tasked with drawing all the minuscules for a new typeface. We have chosen this scenario because it is common for type designers to focus on one case at a time (i.e., either lowercase or uppercase) and because the minuscules tend to be more challenging to design than the majuscules.

In this scenario, at first (top two sequences in the diagram) the user draws a few initial glyphs.

The alternative that includes Glyph-o-matic depicts how the type designer would interact with the plugin. With the incomplete set of glyphs, and based on the letters 'n', 'o', and 'v', the user invokes Glyph-o-matic, which generates all the remaining minuscules. After that, the user refines the generated glyphs. Using Glyphs 3, the user can adjust the generated glyphs if needed. Figure 4.3 shows what each of these steps look like from our users' perspective.

On the other hand, the alternative that does not include Glyph-o-matic corresponds

Figure 4.2 – Sequence diagram of the type design workflow showing two alternative scenarios: With Glyph-o-matic and Without Glyph-o-matic.

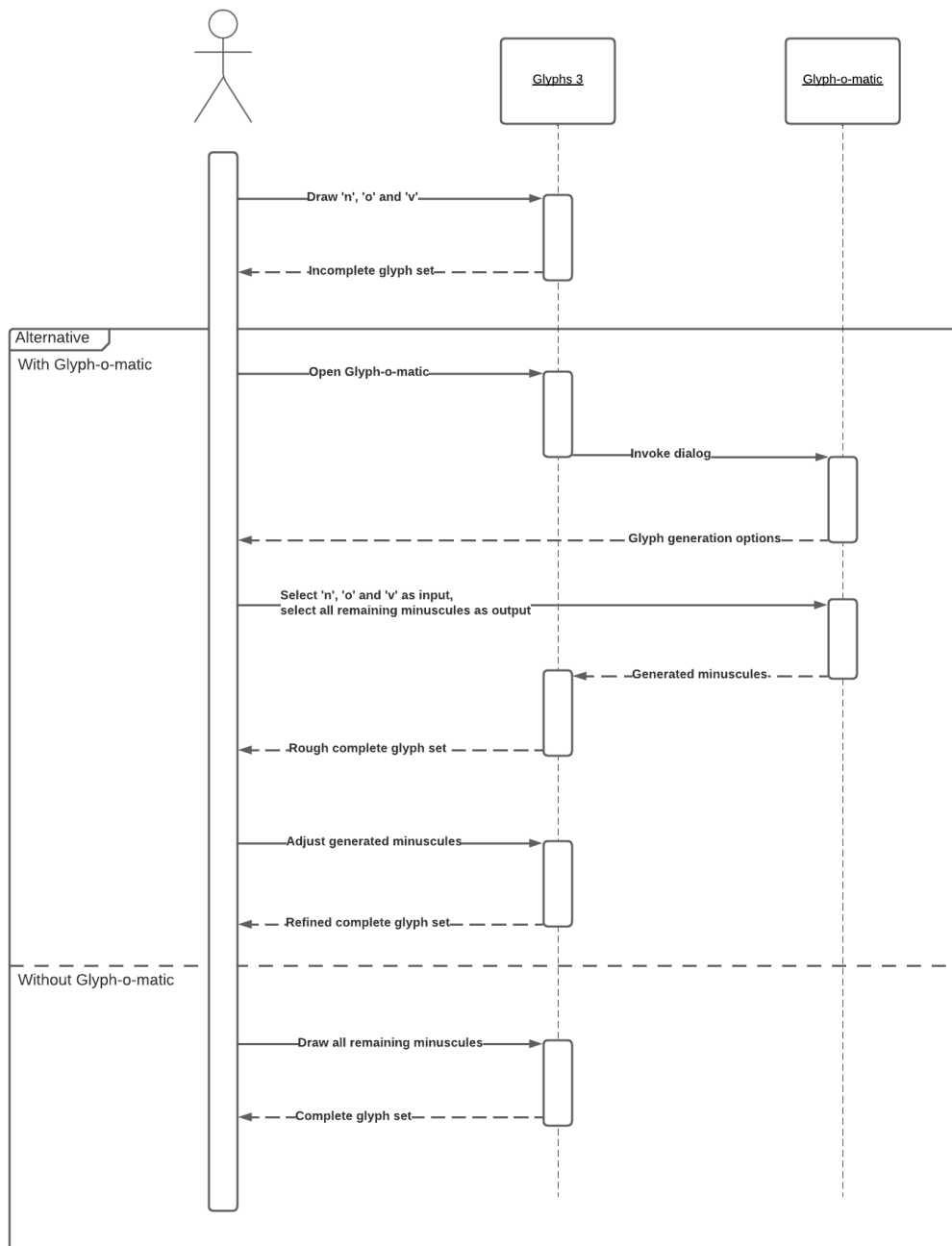
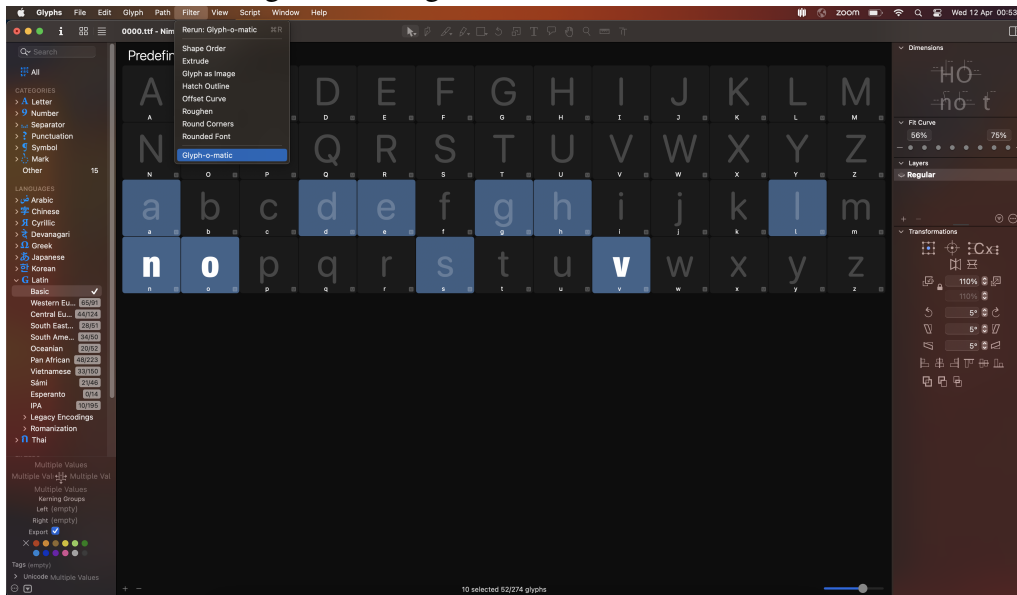


Figure 4.3 – Screenshots of how the user can interact with Glyph-o-matic to generate glyphs based on their original drawings.



to a simpler, yet more burdensome scenario. In this case, the type designer draws all remaining glyphs manually.

Our work's hypothesis is that type designers will prefer the overhead introduced by interacting with Glyph-o-matic and adjusting its results over manually drawing a large number of glyphs. This hypothesis was evaluated through an experiment where an expert user employed Glyph-o-matic to design a new font (see Chapter 5).

5 EVALUATION

This chapter describes the method used to evaluate our plugin with expert users, the task we proposed them in our experiment, and the results we obtained.

5.1 Method

Since this work revolves around the type design workflow, we decided to invite type designers as participants of a remote asynchronous experiment. We invited one type designer with whom we had contact, and he forwarded the invitation to his four coworkers and to other type designers in their network. The participants were asked to try out Glyph-o-matic and provide feedback through a questionnaire. Appendix A contains the questionnaire the type designers answered as a result of their experimentation with Glyph-o-matic.

The questionnaire contains: (1) a term of consent agreed by the participant; (2) questions about the respondent's profile; (3) the download link to the plugin and a sample font; (4) a step-by-step tutorial on how to complete the proposed task (see 5.2) ; (5) rating scales to evaluate how useful the plugin was to perform the task.

5.2 Task

To evaluate the usage of our plugin, we proposed our users the following task:

- Load a provided font file - which contains only the 'n', 'o' and 'v' glyphs - into Glyphs 3;
- Use Glyph-o-matic to generate all other minuscules required to compose the word *handgloves*;
- Adjust the generated glyphs until they judge them to be good enough for initial validation .

We have decided to provide an initial incomplete font to simplify the experiment. Since Glyph-o-matic requires a few original designs to use as reference, requesting our users to draw them would increase the complexity of the task without benefit to our work. Also, we have chosen the word *handgloves* because, as explained in Section 2.2, it is one

Figure 5.1 – Pie chart: familiarity with Glyphs 3.

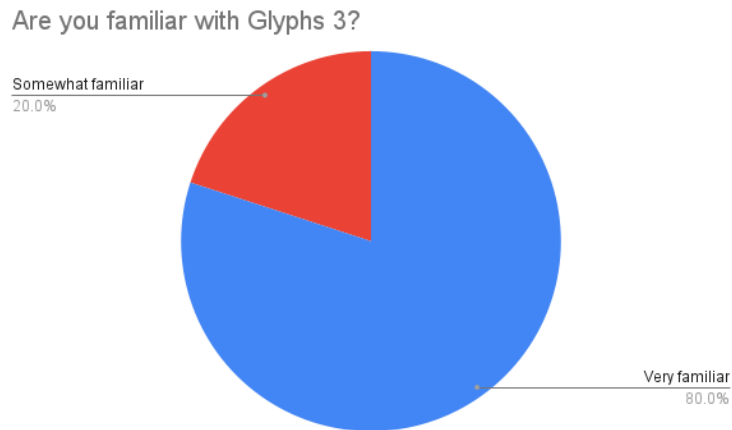
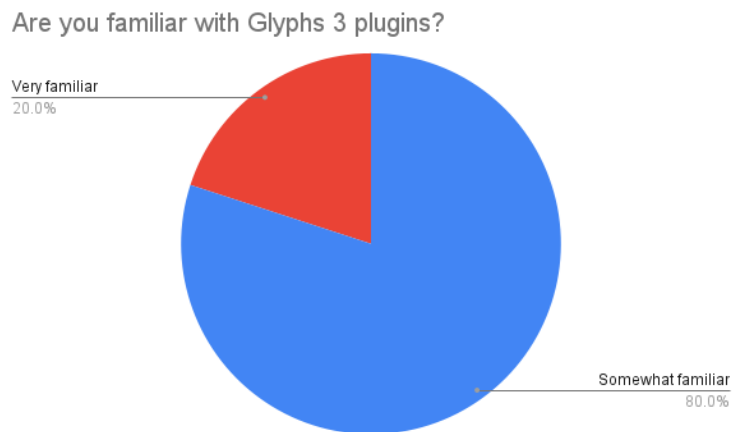


Figure 5.2 – Pie chart: familiarity with plugins for Glyphs 3.



of the most commonly used test words in type design.

5.3 Results

Our questionnaire had a total of five responses. As expected, given the audience of the invitation to this experiment, all of the respondents have advanced knowledge in type design.

As shown in Figures 5.1 and 5.2, most respondents are familiar with Glyphs 3 and its plugins. Therefore, our experiment was not hindered by respondents having to learn how to use the tools upon which Glyph-o-matic was built.

Figure 5.3 illustrates that most respondents found the plugin easy to use. They also stated that they found it easy to generate glyphs using the plugin, even though they would not have found it intuitive without the instructions we gave them.

Figure 5.3 – Bar chart: ease of use of the plugin.

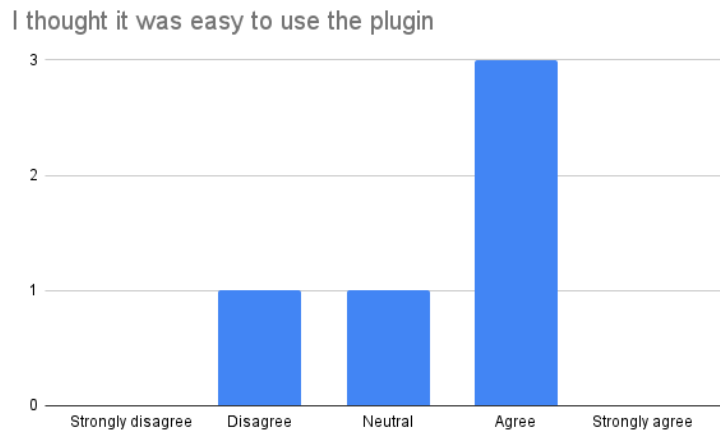
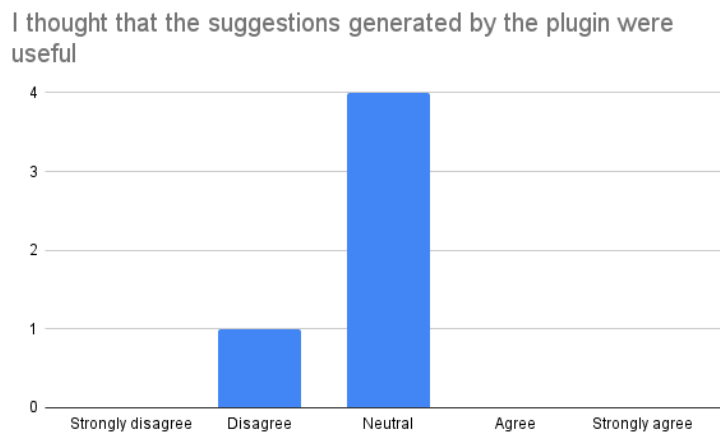


Figure 5.4 – Bar chart: usefulness of the suggestions generated by the plugin.



As far as the usefulness of the suggestions generated by the plugin is concerned, most respondents found them insufficient, as shown in Figure 5.4. Feedback in this regard indicated that the generated glyph images were not very well aligned with the guidelines of the font editor, which frequently required manual repositioning.

As shown in Figure 5.5, most respondents did not feel confident about using the plugin. According to their comments, they felt frustrated because they could not find a logical explanation for the suggestions that Glyph-o-matic generated.

Figure 5.6 indicates that most respondents did not have to learn many things before they could use the plugin. This is consistent with the fact that they already knew how to use Glyphs 3, and that our plugin was fairly simple to use for the proposed experiment.

Finally, as shown in Figure 5.7, most respondents are willing to use the plugin again. Therefore, although the plugin does not provide wondrous suggestions in its current state, our respondents are hopeful about its potential in the future.

Figure 5.5 – Bar chart: confidence about using the plugin.

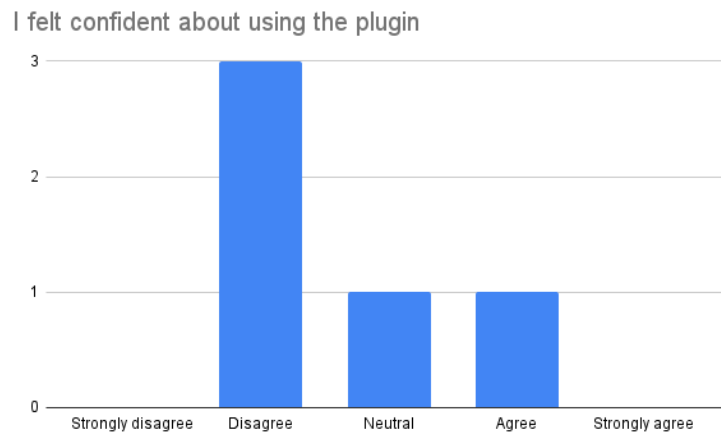


Figure 5.6 – Bar chart: required knowledge to use the plugin.

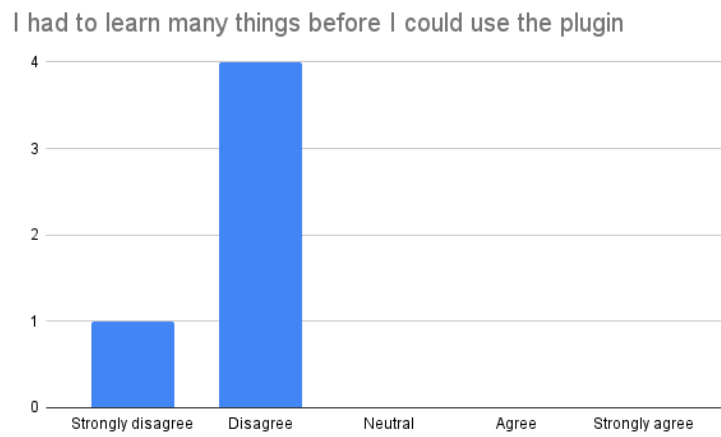
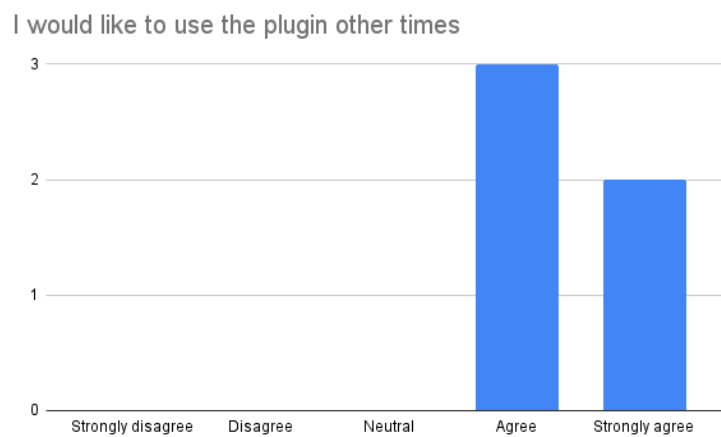


Figure 5.7 – Bar chart: willingness to use the plugin again.



6 CONCLUSION AND FUTURE WORK

This work explored the role that technology has in enabling type designers to create fonts efficiently. In that same sense, we have built a plugin for Glyphs 3 called Glyph-o-matic. Our plugin generates suggestions of letter forms using DeepVecFont, a model for font generation. Then, we conducted experiments with type designers to test the effectiveness of our plugin in helping them carry out a simple task. Based on the obtained results, we conclude that the recent advances in automated font generation are promising tools for type designers. Moreover, we could identify that there is plenty of room for improvement both for our plugin and for the state of the art of AI-powered font generation.

Nonetheless, we could observe that type designers are very technology savvy in general, making it easier for us computer scientists to collaborate with them. There are many tasks in the type design workflow that are still very much manual and error-prone. On the other hand, font editors provide plenty of tooling for their community to build extensions, plugins and scripts. Therefore, we believe that this work can be seen as a valid step towards bridging the gap between type designers and recent advances in artificial intelligence.

As opportunity for future work, we suggest further iterations with type designers to improve the user experience of Glyph-o-matic. For example, it would be possible to allow the user to tweak parameters of the underlying inference process that can yield results faster and/or of higher quality. Moreover, a new version of the model we used (DeepVecFont) was recently presented by Wang et al. (2023). This poses an opportunity to evaluate how type designers can benefit from its results - which are supposedly better than the ones we used.

REFERENCES

- AHRENS, T. 2023. Disponível em: <<https://kern-on.com/>>.
- AZADI, S. et al. **Multi-Content GAN for Few-Shot Font Style Transfer**. arXiv, 2017. Disponível em: <<https://arxiv.org/abs/1712.00516>>.
- BUCHNER, R. 2023. Disponível em: <<https://github.com/RafalBuchner/StemThickness>>.
- CHA, J. et al. **Few-shot Compositional Font Generation with Dual Memory**. arXiv, 2020. Disponível em: <<https://arxiv.org/abs/2005.10510>>.
- CHENG, K. **Designing Type**. 2. ed. New Haven, CT: Yale University Press, 2020.
- COCOA. Apple, 2013. Disponível em: <<https://developer.apple.com/library/archive/documentation/Cocoa/Conceptual/CocoaFundamentals/WhatIsCocoa/WhatIsCocoa.html>>.
- FONTLAB. 2023. Disponível em: <<https://www.fontlab.com>>.
- GERNER, J. **Writing plug-ins**. Glyphs, 2016. Disponível em: <<https://glyphsapp.com/learn/plugins>>.
- GLYPHS. 2023. Disponível em: <<https://glyphsapp.com/>>.
- GOODFELLOW, I. J. et al. **Generative Adversarial Networks**. arXiv, 2014. Disponível em: <<https://arxiv.org/abs/1406.2661>>.
- HENESTROSA, C.; MESEGUER, L.; SCAGLIONE, J. **Como criar tipos: do esboço à tela**. 2. ed. [S.l.]: Estereográfica, 2020.
- PARK, S. et al. **Few-shot Font Generation with Localized Style Representations and Factorization**. arXiv, 2020. Disponível em: <<https://arxiv.org/abs/2009.11042>>.
- RAMESH, A. et al. **Zero-Shot Text-to-Image Generation**. 2021.
- TANG, L. et al. **Few-Shot Font Generation by Learning Fine-Grained Local Styles**. arXiv, 2022. Disponível em: <<https://arxiv.org/abs/2205.09965>>.
- TRACY, W. **Letters of Credit: A View of Type Design**. Jaffrey, NH: David R. Godine, 2003.
- UNGER, G. **Theory of Type Design**. Rotterdam, Netherlands: NAI, 2018.
- VAUGHAN, D. **The world's 5 most commonly used writing systems**. Encyclopædia Britannica, inc., 2020. Disponível em: <<https://www.britannica.com/list/the-worlds-5-most-commonly-used-writing-systems>>.
- WANG, Y.; LIAN, Z. **DeepVecFont: Synthesizing High-quality Vector Fonts via Dual-modality Learning**. arXiv, 2021. Disponível em: <<https://arxiv.org/abs/2110.06688>>.
- WANG, Y. et al. **DeepVecFont-v2: Exploiting Transformers to Synthesize Vector Fonts with Higher Quality**. 2023.

XCODE. Apple, 2011. Disponível em: <<https://developer.apple.com/library/archive/documentation/IDEs/Conceptual/Xcode4TransitionGuide/InterfaceBuilder/InterfaceBuilder.html>>.

YANONE. 2023. Disponível em: <<https://github.com/yanone/speedpunk>>.

ZENG, J. et al. Strokegan: Reducing mode collapse in chinese font generation via stroke encoding. arXiv, 2020. Disponível em: <<https://arxiv.org/abs/2012.08687>>.

APPENDIX A — QUESTIONNAIRE

Questionário de avaliação do Glyph-o-matic

Este estudo está sendo realizado por Matheus Rhoden, aluno de graduação do curso de Ciência da Computação da UFRGS, sob orientação da professora Carla Dal Sasso Freitas.

Termo de Consentimento Livre e Esclarecido

Você está sendo convidado(a) a participar de uma pesquisa sobre um plugin para o Glyphs 3 chamado Glyph-o-matic. Este termo tem o propósito de explicar os objetivos do estudo, os procedimentos, os riscos e como serão conduzidos os testes. Pedimos que o leia atentamente e esclareça todas as dúvidas antes de consentir na sua participação. Ressaltamos que não são solicitados dados que possam identificar você. Os dados coletados serão utilizados unicamente para a pesquisa de TCC e serão eliminados após a conclusão e publicação do trabalho.

Este estudo tem como objetivo avaliar o uso do Glyph-o-matic como assistente a type designers que utilizam o Glyphs 3 para criar fontes. Mais especificamente, este estudo visa avaliar se glifos gerados por inteligência artificial podem ser aproveitados por type designers para poupar tempo.

Procedimentos: É recomendado que o participante tenha conhecimento avançado de type design e esteja familiarizado com o uso de Glyphs 3. Ademais, é desejável que o participante tenha experiência com uso de plugins de Glyphs 3.

Riscos e benefícios: O presente estudo poderá demorar cerca de 60 minutos para ser concluído. Portanto, é recomendado que você tenha no mínimo esse tempo disponível para poder realizar o estudo completo. O benefício para o participante é a oportunidade de poder contribuir para uma pesquisa sobre a aplicação da inteligência artificial no fluxo de trabalho de type designers, o que poderá resultar em melhoria de futuras aplicações desses métodos.

Você consente em participar do estudo? *

Sim

Não

Responda as perguntas abaixo para que seu perfil fique associado com suas respostas. Ratificamos que não será necessário se identificar e que os dados, portanto, serão anônimos.

Qual é o seu nível de conhecimento de type design? *

- Baixo (Eu conheço os conceitos básicos de type design e entendo seus diferentes termos)
- Intermediário (Trabalhei em alguns projetos de type design)
- Alto (Eu trabalho na área de type design)

Você possui familiaridade com Glyphs 3? *

- Sim (Possuo muita experiência com Glyphs 3)
- Sim (Possuo alguma experiência com Glyphs 3)
- Não (Não possuo experiência com Glyphs 3)

Você já utilizou plugins de Glyphs 3? *

- Sim (Já utilizei muitos plugins de Glyphs 3)
- Sim (Já utilizei alguns plugins de Glyphs 3)
- Não (Nunca utilizei plugins de Glyphs 3)

Preparação

- Faça download do Glyph-o-matic e instale o plugin conforme as instruções: <https://drive.google.com/drive/folders/1nTrntw0JDCTmFq->

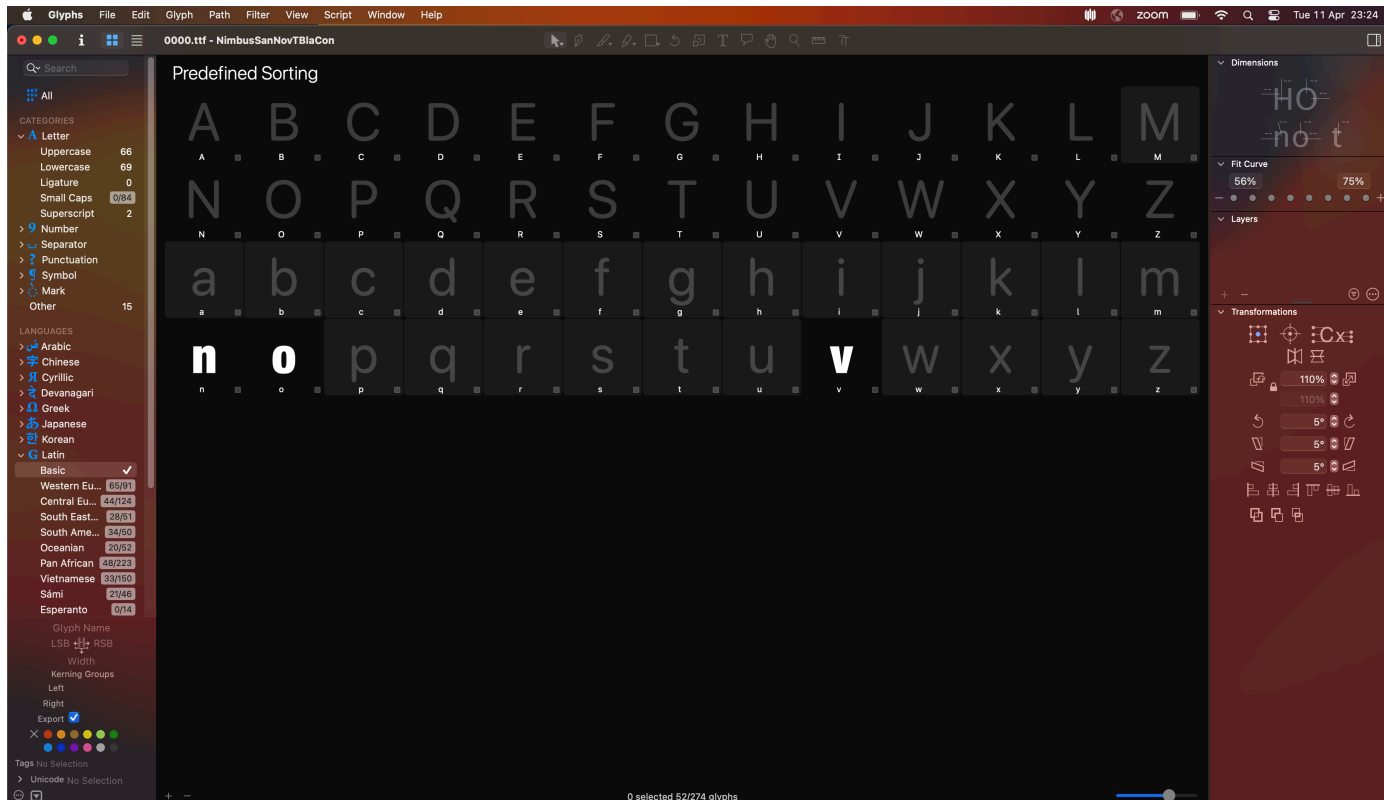
[00UGdsPD_We_YA5GD?usp=share_link](https://drive.google.com/file/d/16iA5oJf9af3yAbREJVAvx60agLUh4ZPp/view?usp=share_link)

- Faça download da fonte que será utilizada no experimento: https://drive.google.com/file/d/16iA5oJf9af3yAbREJVAvx60agLUh4ZPp/view?usp=share_link

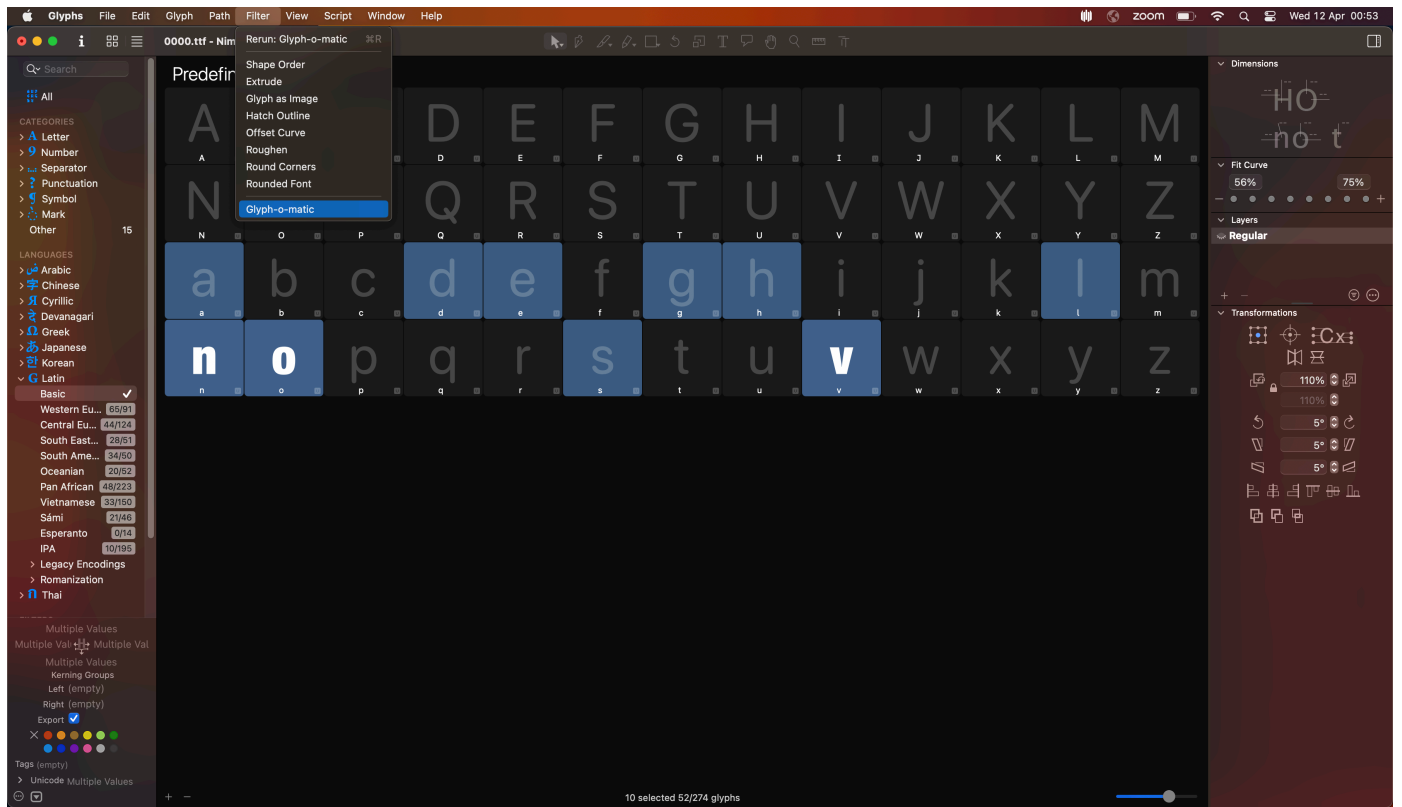
Experimento

- Este experimento consiste em desenhar todas as letras necessárias para escrever a palavra teste *handgloves*.
- Como ponto de partida, a fonte disponibilizada na etapa de preparação contém três dessas letras: *n*, *o* e *v*.
- Sendo assim, sua tarefa consiste em gerar as demais letras da palavra *handgloves* com o auxílio do nosso plugin, o Glyph-o-matic

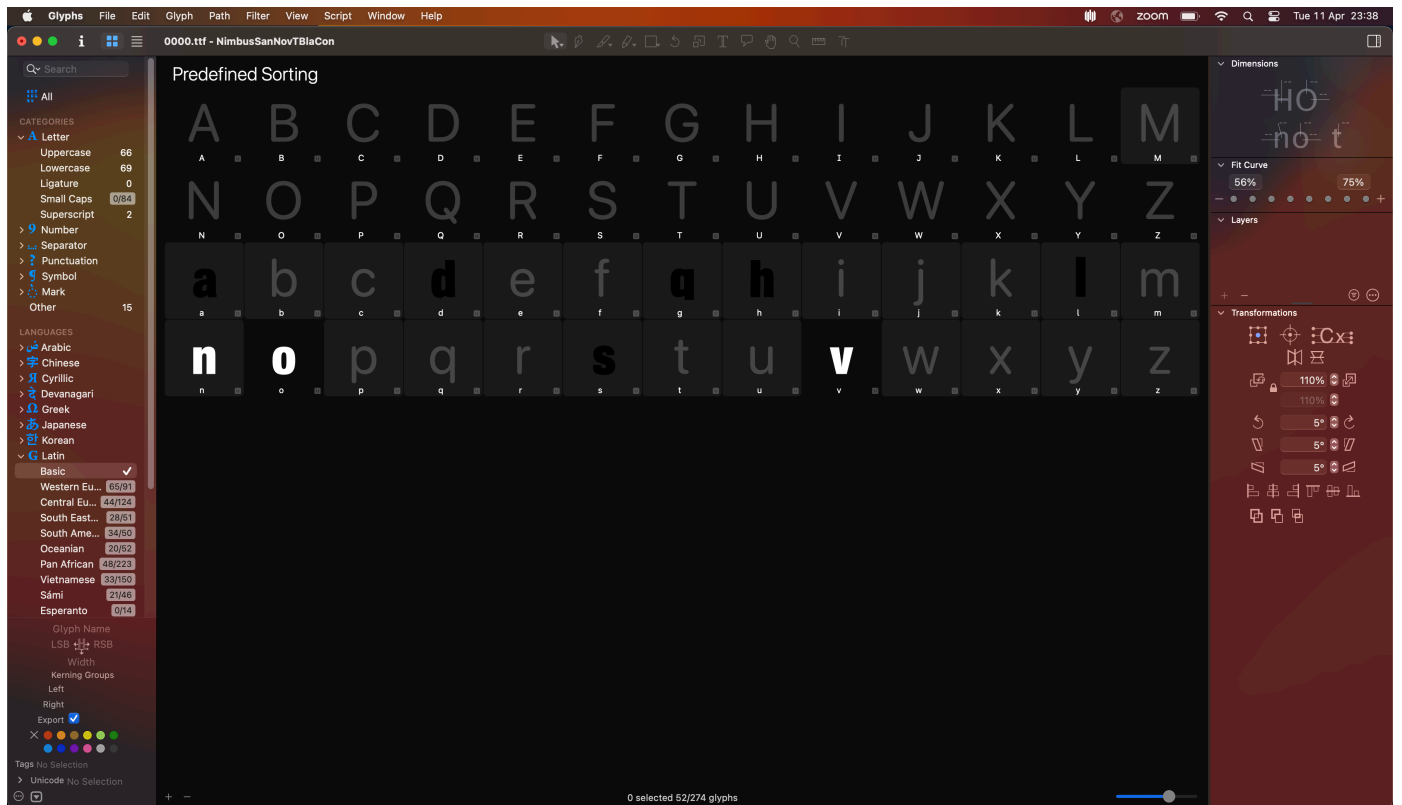
Abra o Glyphs 3 e importe a fonte baixada na etapa de preparação



Selecione todas as letras da palavra *handgloves* e clique em *Glyph-o-matic* na aba de Filtros do Glyphs 3



Feito isso, o plugin haverá gerado sugestões de desenhos para as demais letras



Com base nas sugestões geradas no passo anterior, desenhe as letras restantes para compor a palavra *handgloves*

Tenha em mente que seu objetivo é criar um design consistente com as letras *n*, *o* e *v* disponibilizadas. Como são geradas com inteligência artificial, as sugestões geradas pelo plugin não necessariamente serão de grande ajuda para isso. Sendo assim, sugerimos que você realize os ajustes necessários e, na etapa seguinte, avalie quão produtivo foi utilizar o plugin.

Avaliação geral da usabilidade

Responda as seguintes perguntas sobre como foi usar o plugin *

	Discordo plenamente	Discordo	Neutro	Concordo	Concordo plenamente
Eu achei fácil utilizar o plugin	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
Eu acho que as sugestões geradas pelo plugin são úteis	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
Eu me senti confiante ao usar o plugin	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Eu precisei aprender várias coisas novas antes de conseguir usar o plugin	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Eu gostaria de utilizar esse plugin outras vezes	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>

Se você tiver críticas e/ou sugestões, escreva aqui:

.....

Muito obrigado por participar do nosso estudo!

This content is neither created nor endorsed by Google.

Google Forms