

```

    font = ImageFont.truetype(font_path, font_size)
    text_width, text_height = draw.textsize(text, font=font)
    x = (width - text_width) // 2
    y = (height - text_height) // 2

    if effects:
        for effect in effects:
            if effect == "glitch":
                draw_glitch_effect(draw, text, font, x, y)
            elif effect == "wave":
                draw_wave_effect(draw, text, font, x, y)
            elif effect == "distort":
                draw_distorted_text(draw, text, font, x, y)
        else:
            draw.text((x, y), text, font=font, fill="black")

def draw_glitch_effect(draw, text, font, x, y):
    for offset in range(-5, 6, 2):
        color = random.choice(["red", "blue", "green"])
        draw.text((x + offset, y), text, font=font, fill=color)

def draw_wave_effect(draw, text, font, x, y):
    spacing = 10
    for i, char in enumerate(text):
        offset = int(10 * random.sin(i / spacing))
        draw.text((x + i * 40, y + offset), char, font=font,
fill="black")

def draw_distorted_text(draw, text, font, x, y):
    for i in range(len(text)):
        jitter_x = random.randint(-3, 3)
        jitter_y = random.randint(-3, 3)
        draw.text((x + i * 40 + jitter_x, y + jitter_y), text[i],
font=font, fill="black")

if __name__ == "__main__":
    text_to_display = " L'évolution du
dessin de caractères créatif par
ordinateur, dès les expérimentations
du ACG MIT Media Lab, jusqu'à nos
jours. "

    output_file = "text_design_output.png"
    applied_effects = ["glitch", "wave", "distort"]

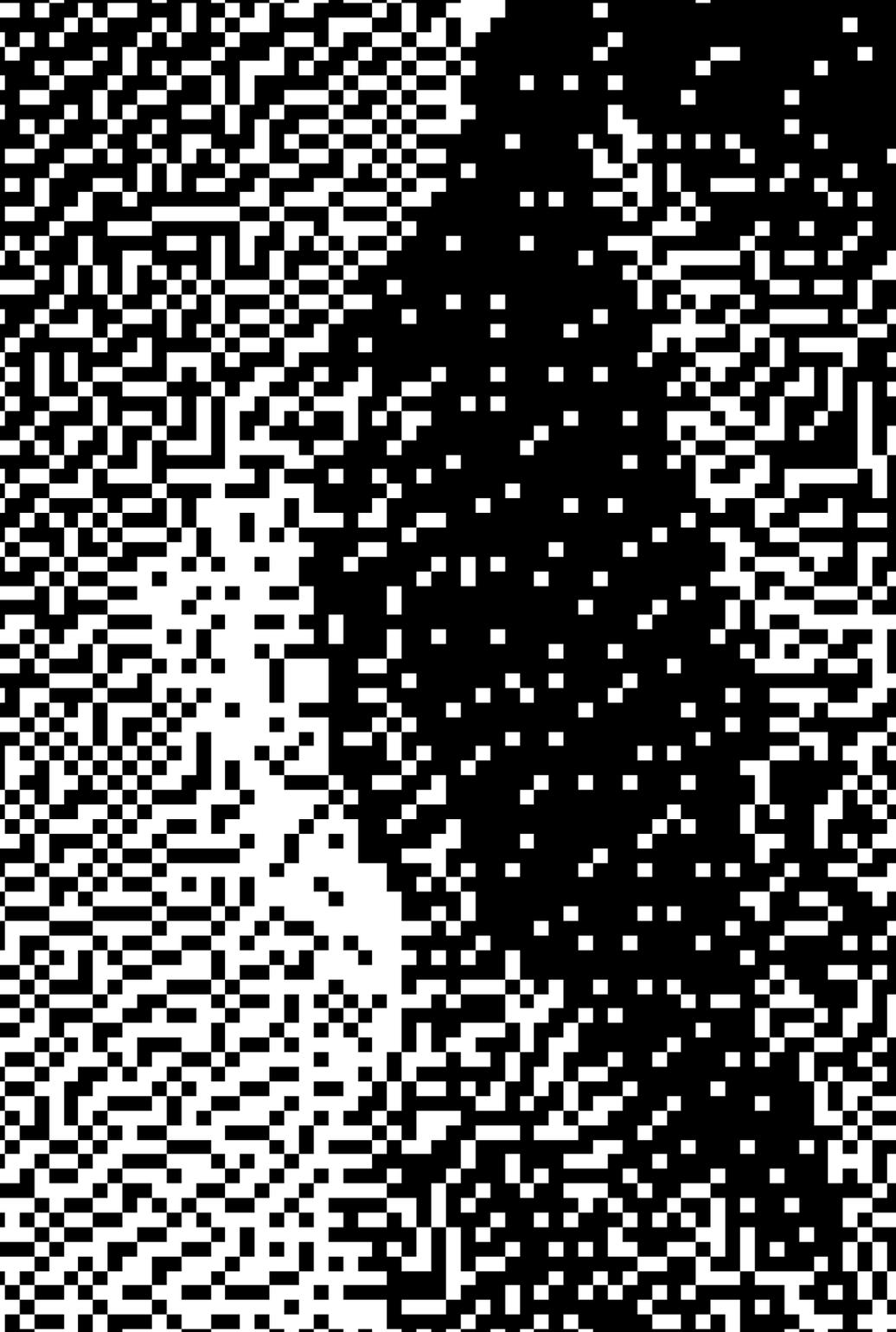
    generate_text_image(text_to_display, font_file, output_file,
effects=applied_effects)

```


Émilie Roudaut

L'évolution du dessin de caractères créatif par ordinateur,

**dès les expérimentations
du ACG MIT Media Lab jusqu'à
nos jours.**



```
print(  
«Sommaire»  
)
```

Introduction 10

Un peu de contexte et de définitions 13

**La genèse du type design au
ACG MIT Media LAB** 17

Reactive Books 19

Design by Numbers 21

Type me Type me not 22

Alphabet Zoo 24

**L'héritage de ces expérimentations
et leurs dépassements** 27

Creative Coding as a School of Thoughts 27

How could virtual reality change typography? 28

Weird Type 29

Scriptographer 31

Type & Form 32

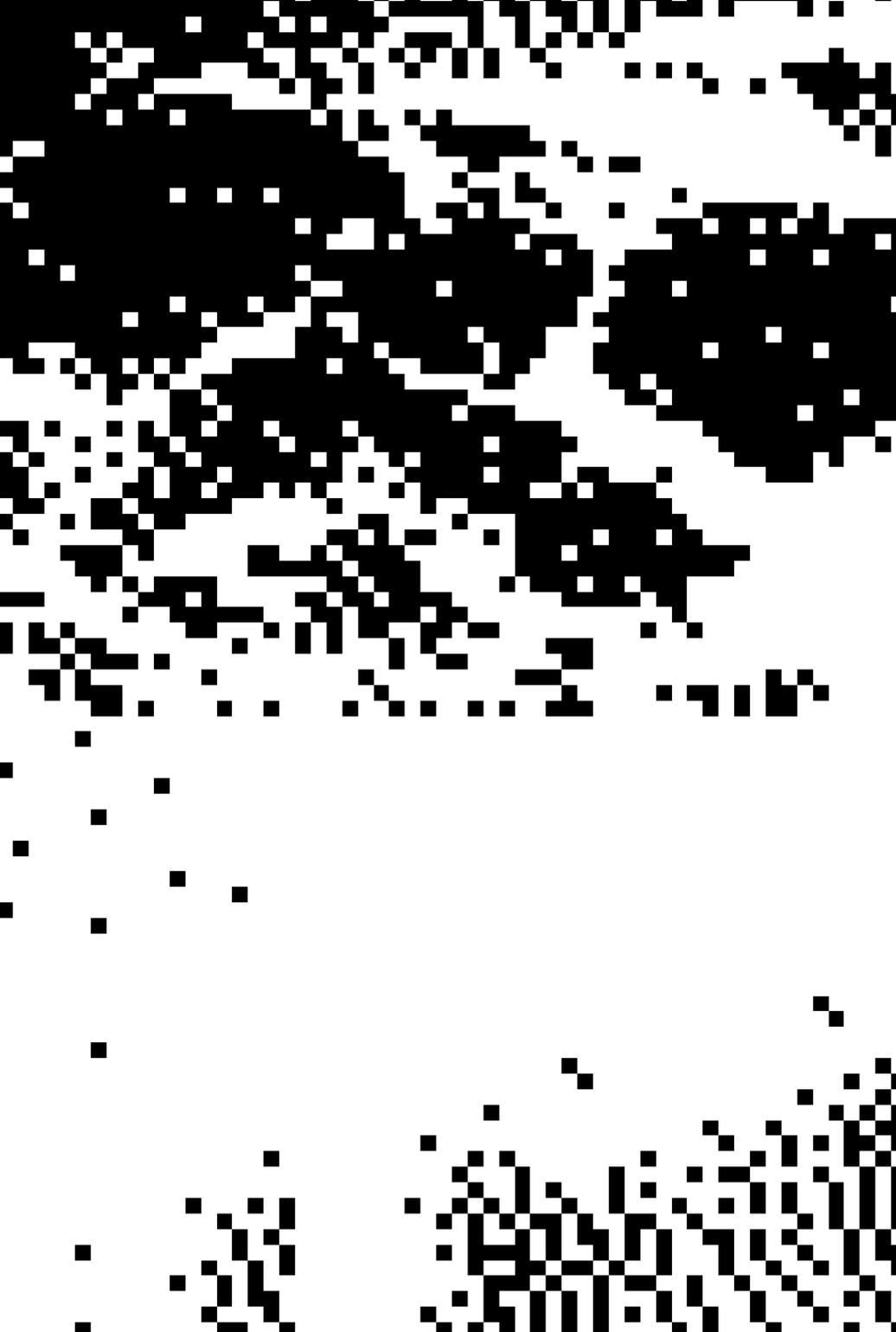
Schrofer's illegible alphabet 33

Conclusion	36
Annexes	39
Entretien	54
Glossaire	71
Références	76
Bibliographie	76
Sitographie	78
Vidéos et Podcasts	79
Remerciements	81

“Electronics will soon force its claims upon letterforms, and let us hope it will liberate us from the dust of the past.”

Hermann Zapf, 1968 [a]

[a] *Citation de Hermann Zapf provenant de Letterform Design Systems by Lynn Ruggles, Page 1.*



```
print(  
«Introduction»  
)
```

13|

Depuis mes premières explorations dans l'univers captivant des objets graphiques, chaque lettre de l'alphabet m'a semblé posséder une identité propre, un langage visuel distinct. Mon intérêt pour la typographie a rapidement évolué au-delà de la simple compréhension des caractères en tant qu'éléments linguistiques pour devenir une fascination profonde pour leur capacité à communiquer des messages, à susciter des émotions et à témoigner de l'envie commune de communiquer. Les affiches, et autres supports visuels qui ont orné mes murs depuis l'enfance ont constitué une source inépuisable d'inspiration, propulsant ma réflexion vers l'évolution du dessin de caractères créatif sur ordinateur. Au fil de mes études, j'ai été plongé dans le monde complexe de la typographie numérique. Chaque lettre, désormais façonnée par la technologie et l'ordinateur, devenait une toile malléable pour exprimer des nuances esthétiques. L'exploration des possibilités offertes par l'informatique pour modeler et altérer ces caractères est devenue le cœur de ma démarche artistique. Parallèlement, une autre dimension de ma fascination a émergé, centrée autour de la manipulation informatique, du creative coding ainsi que des langages de programmation. Ces éléments, ont capturé mon attention en tant que sources riches d'inspiration créative, me poussant à explorer les frontières subtiles entre une esthétique intentionnelle et les altérations fortuites résultant de l'interaction complexe entre l'homme et la machine. Dans cette quête de compréhension des origines de ces possibilités créatives, mes investigations m'ont naturellement dirigé vers les travaux précurseurs du Aesthetics and Computation Group (ACG) menés au Media Lab du MIT dans les années 80 et 90. Comment ces pionniers ont-ils contribué à façonner le paysage actuel du dessin de caractère créatif sur ordinateur? Quel impact leurs premières expérimentations ont-elles eu sur la trajectoire de cet art émergent? Et surtout, comment le dessin de caractère, rendu possible grâce à des modèles computationnels, a-t-il évolué depuis les premières étapes du ACG jusqu'à nos jours?

C'est dans cette perspective que ce document se propose d'explorer en profondeur cette évolution, offrant une analyse approfondie de l'histoire du dessin de caractère créatif sur ordinateur depuis les années 80 jusqu'à nos jours. Pour répondre à ces interrogations, je débiterai par établir le contexte technologique et artistique de l'époque, définissant les termes et les concepts qui ont nourri ces premières explorations. Ensuite, je me plongerai dans l'examen détaillé de projets emblématiques menés au sein du Media Lab Aesthetics Computation Group (ACG) du MIT, mettant en lumière les avancées novatrices qui ont marqué cette époque. Enfin, je clôturerai cette exploration en présentant un échantillon représentatif de ce que les artistes contemporains produisent aujourd'hui, soulignant ainsi l'héritage toujours vivant de ces premières expérimentations du ACG et la manière dont elles continuent à influencer la création typographique contemporaine.



```
print(  
«Un peu de contexte et de définitions»  
)
```

Les années 80 et 90 ont été une période charnière dans l'histoire de l'informatique mondiale, marquant l'avènement de technologies qui ont redéfini la manière dont nous interagissons avec le numérique. Dans le contexte mondial, ces décennies ont été témoins de la prolifération des ordinateurs personnels, du développement des interfaces utilisateur graphiques (GUI), et de l'émergence d'innovations qui ont ouvert la voie à l'ère numérique actuelle. Le marché de l'informatique personnelle a connu une explosion dans les années 80, avec l'introduction de machines emblématiques telles que l'IBM PC [b] et le Macintosh d'Apple [c].



[b] L'IBM PC

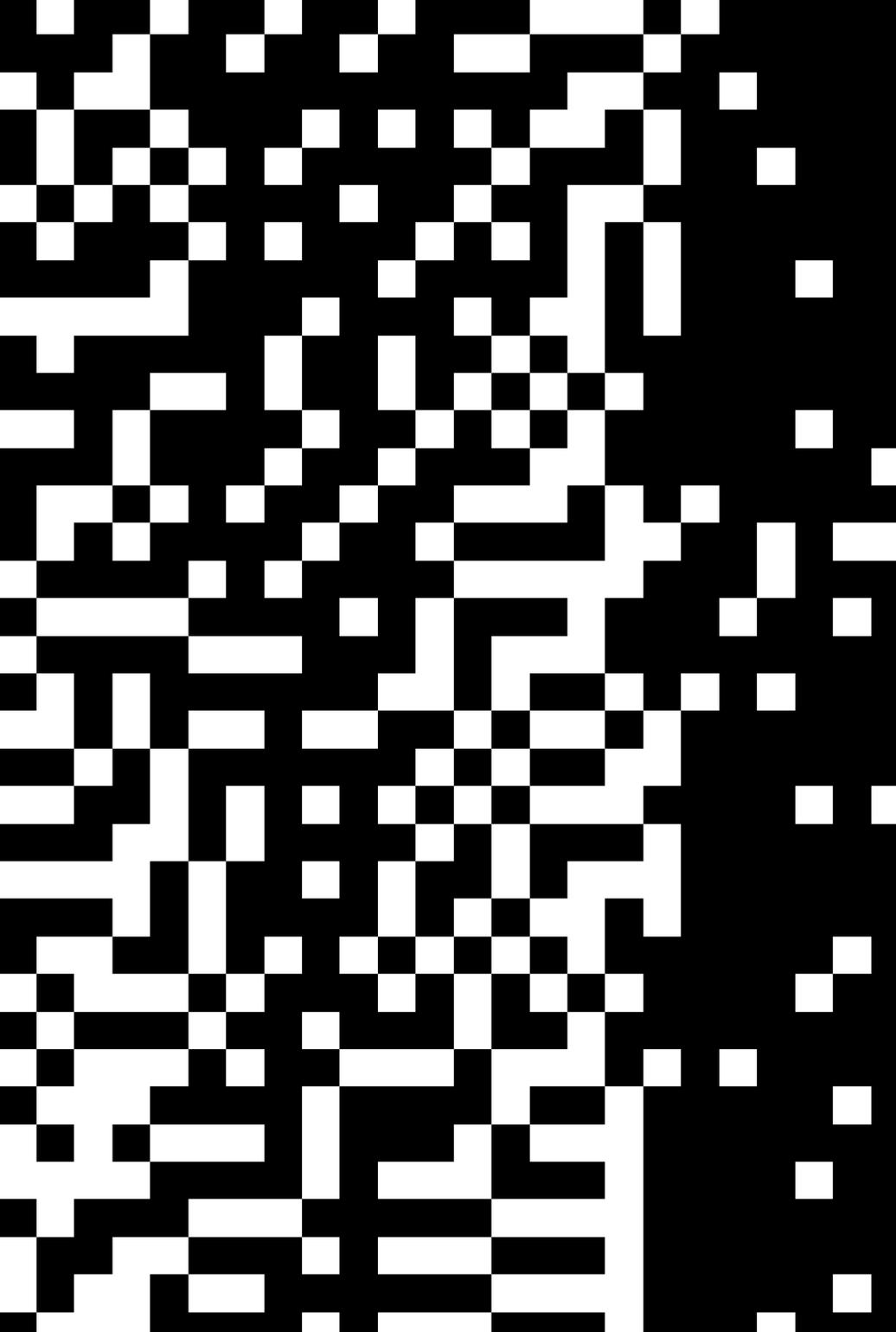


[c] Le Macintosh d'Apple

Ces ordinateurs ont démocratisé l'accès à la technologie, transformant les machines autrefois réservées aux experts en outils accessibles à un public plus large. L'interface graphique révolutionnaire du Macintosh, lancée en 1984, a ouvert la voie à une nouvelle ère d'interactions visuelles, remplaçant les interfaces en ligne de commande par des icônes, des fenêtres et une souris. Cela a considérablement amélioré l'expérience utilisateur et a marqué le début de la démocratisation de la créativité numérique. Dans ce contexte, le MIT Media Lab (Media Lab du Massachusetts Institute of Technology, Cambridge, États-Unis) a joué un rôle crucial dans la fusion de l'art, de la science et de la technologie. Fondé en 1985, le Media Lab a été le berceau de l'innovation numérique, explorant les frontières de la créativité computationnelle et de la recherche interdisciplinaire. Au sein du Media Lab, le Aesthetics and Computation Group (ACG) a été formé en 1989, dirigé par des figures visionnaires telles que John Maeda. Le groupe s'est consacré à l'exploration de l'intersection entre l'esthétique, la créativité et l'informatique.

- 17| Les travaux du ACG au MIT ont inclus des projets novateurs dans le domaine de la typographie générative, de la création artistique numérique et de l'exploration des relations entre le texte et l'image. Les expérimentations du groupe ont souvent impliqué l'utilisation d'algorithmes pour générer des formes typographiques innovantes, repoussant les limites de la créativité numérique. Ces travaux ont contribué à élargir la conception traditionnelle de la typographie et ont ouvert de nouvelles perspectives sur la manière dont la technologie peut être intégrée dans le processus créatif. Dans cet environnement dynamique, le ACG a collaboré avec des artistes, des designers et des chercheurs, favorisant des échanges interdisciplinaires qui ont enrichi la créativité numérique. Ces expérimentations ont non seulement eu un impact au niveau académique, mais ont également influencé le paysage plus vaste de l'art numérique et du design interactif.

En résumé, les années 80 et 90 ont été une époque où l'informatique personnelle s'est répandue, transformant la manière dont nous interagissons avec la technologie. Dans ce contexte, le MIT Media Lab et le ACG ont émergé comme des pionniers qui ont repoussé les frontières de la créativité computationnelle, contribuant à façonner le paysage numérique contemporain. Ces décennies ont jeté les bases de l'ère numérique dans laquelle nous vivons aujourd'hui, où la technologie et la créativité convergent de manière inédite.



```
print(  
«La genèse du type design au ACG MIT  
Media LAB»  
)
```

La genèse du type design au sein du Aesthetics and Computation Group (ACG) du MIT est un récit qui nous plonge au cœur de l'intersection entre l'art, la technologie et la pensée computationnelle. Fondé en 1989 par John Maeda, le ACG s'est imposé comme un laboratoire novateur au sein du Media Lab, explorant les frontières de l'esthétique et de la computation. La créativité générée au sein de ce groupe a laissé une empreinte indélébile sur le monde du type design, introduisant des concepts novateurs et des approches qui ont façonné la discipline de manière significative. John Maeda, en tant que fondateur et figure clé du ACG, a joué un rôle central dans l'émergence du type design au sein du groupe. Sa vision audacieuse de fusionner l'esthétique avec la computation a jeté les bases de projets révolutionnaires qui ont redéfini la manière dont nous percevons et concevons les caractères typographiques. Maeda, avec son expertise en arts visuels et en programmation informatique, a créé un terrain fertile pour l'innovation dans le domaine du type design.

|20

En collaboration avec John Maeda, plusieurs figures ont joué des rôles clés au sein du ACG. Les deux autres personnalités principales que nous pouvons nommer sont Muriel Cooper et Peter Cho. En effet, Muriel Cooper, d'abord professeur de Maeda et fondatrice du Visual Language Workshop [d], a été une figure emblématique au sein du ACG.

[d] *Le Visible Language Workshop (VLW) accueillait le cours «Messages and Means», qui initiait les étudiants à la conception graphique en leur donnant accès aux méthodes de production et en les encourageant à rechercher de nouvelles formes de communication graphique. Dirigé par Muriel Cooper, et intégré dans le riche environnement technologique du MIT, le VLW était idéalement placé pour innover au sein de ces nouvelles formes graphiques et technologiques.*

Son travail a aidé une exploration approfondie de la relation entre le langage visuel et la technologie. Le Visual Language Workshop, en collaboration avec le ACG, a été un creuset d'idées novatrices qui ont nourri le type design en intégrant des concepts visuels, linguistiques et computationnels. D'un autre côté, Peter Cho, a également contribué de manière significative à l'évolution du type design au sein du groupe. Son engagement dans la recherche sur l'interaction entre l'humain et la machine a apporté une perspective unique à la conception typographique. Les expérimentations de Cho ont souvent mis en lumière la manière dont la technologie peut être utilisée de manière interactive pour influencer la création de caractères, inaugurant ainsi de nouvelles avenues pour le type design. Les premières expériences du groupe portaient le plus souvent sur des formes géométriques simples, des motifs et des combinaisons de couleurs. Le ACG propose des expérimentations qui permettent d'élargir la perception de la typographie en intégrant des principes computationnels et en explorant de nouvelles formes d'expression visuelle. Lors de mes recherches, j'ai relevé 6 types d'expérimentations: la typographie générative, l'interaction homme-machine, la programmation créative, l'exploration de la relation texte-image, l'utilisation de l'ordinateur comme outil numérique et enfin, l'intérêt de l'esthétique dans le processus computationnel. Tous ces tests débutent dans les années 80/90, ils témoignent d'une exploration audacieuses intégrant la programmation et l'esthétique pour repenser la typographie. Ils donnent à voir l'interaction entre la conception graphique et l'informatique, tout en influençant par la suite le champ de la typographie numérique. La typographie générative est abordée par la ACG grâce à des programmes informatiques, des algorithmes qui ont été développés pour générer des formes.

Afin de mieux comprendre les expérimentations menées au sein du groupe, nous allons évoquer quelques projets.

```
print(  
«Reactive Books»  
)
```

Le projet *Reactive Books* a été l'une des premières incursions du ACG dans le domaine du type design. Initié par John Maeda, ce projet explorait la réactivité des caractères typographiques aux interactions de l'utilisateur. Les lettres répondaient aux mouvements et aux actions de l'utilisateur, créant une expérience dynamique et interactive. *Reactive Books* a ouvert la voie à la conception de caractères qui transcendent le statique pour devenir des entités réactives et engageantes. Sur son site John Maeda explique le projet de la manière suivante : [e] « Lorsque j'ai commencé en 1992 à créer des graphiques interactifs, ou réactifs comme je les appelais, il y avait beaucoup de contenu basé sur des CD-ROM qui semblaient passer à côté de l'intérêt des médias computationnels. Avec l'éditeur de médias numériques Digitalogue, j'ai créé quatre livres (le cinquième n'a jamais été imprimé) qui se concentraient sur différents aspects de l'ordinateur en relation avec le média visuel. Le premier s'intitulait *The Reactive Square* et a été publié en 1994. Il s'agit de 10 carrés qui réagissent à l'entrée d'un microphone. Ensuite, en 1995, j'ai créé *Flying Letters*, qui utilisait la souris pour manipuler des marionnettes typographiques. J'ai ensuite créé une série de 12 horloges numériques en 1996, intitulée *12 o'clocks*, qui jouait sur des comportements graphiques simples basés sur le temps. En 1998, j'ai publié *Tap, Type, Write*, un hommage à la machine à écrire en noir et blanc. Une pièce intitulée *Mirror Mirror*, qui utilisait la vidéo comme principal moyen d'interaction, devait être publiée en 1999, mais Digitalogue a fermé ses portes en 2000 en raison de la maladie de l'éditeur fondateur. »

| 22

[e] Lien vers l'explication du projet sur le site de John Maeda.
<https://maedastudio.com/the-five-reactive-books/>



Reactive Books par John Maeda

https://www.youtube.com/watch?v=nA_UTUvC4h8

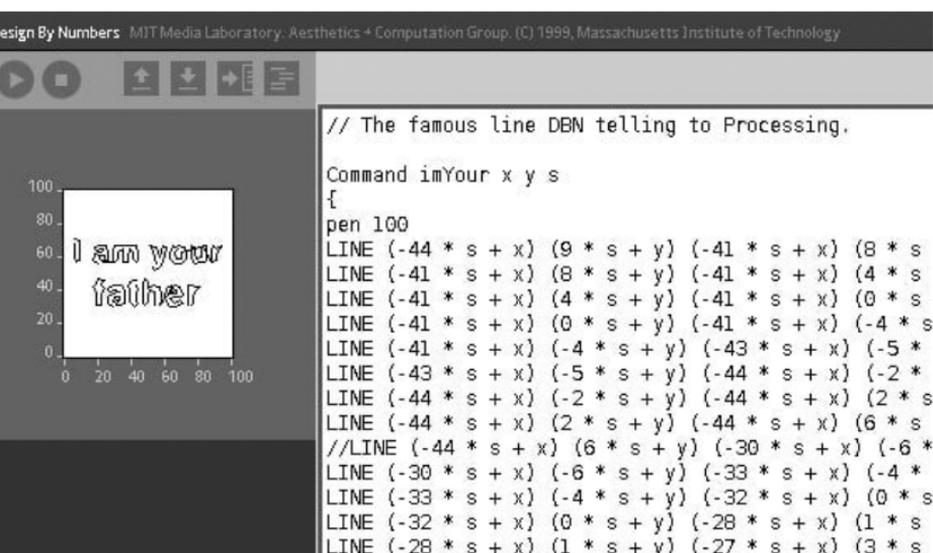
23|

Cette approche expérimentale représente une rupture avec la passivité de la lecture traditionnelle et rapproche le design graphique de la programmation. En intégrant de l'interactivité dans son projet Maeda redéfinit le rôle du designer graphique. Ce dernier devient un créateur d'expériences interactives, capable de contrôler non seulement la forme visuelle mais aussi le comportement des éléments. La place de l'utilisateur est elle aussi modifiée. Il évolue de lecteur à participant actif. Il modifie lui même son expérience de lecture en la rendant unique avec par exemple les actions de cliquer, glisser, modifier les éléments de *Reactive Books*. En somme, ce premier projet présenté démontre d'une volonté de s'ouvrir vers un design graphique et typographique axé sur l'expérience.

```
print(  
«Design By Numbers»  
)
```

Design by Numbers, un autre projet marquant du ACG, a également été dirigé par John Maeda. Ce projet visait à démocratiser la création typographique artistique en utilisant un langage de programmation visuel simple. En intégrant des principes de design graphique et d'algorithmes computationnels, Design by Numbers a initié une approche accessible et ludique du type design, permettant à un large public d'explorer la création de caractères sans barrières techniques complexes.

Design By Numbers MIT Media Laboratory, Aesthetics + Computation Group, (C) 1999, Massachusetts Institute of Technology



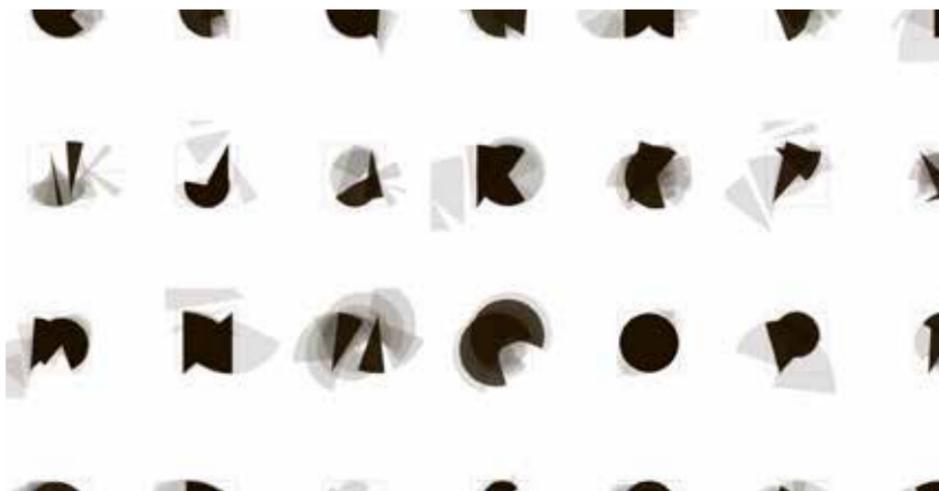
```
// The famous line DBN telling to Processing.  
  
Command imYour x y s  
{  
  pen 100  
  LINE (-44 * s + x) (9 * s + y) (-41 * s + x) (8 * s  
  LINE (-41 * s + x) (8 * s + y) (-41 * s + x) (4 * s  
  LINE (-41 * s + x) (4 * s + y) (-41 * s + x) (0 * s  
  LINE (-41 * s + x) (0 * s + y) (-41 * s + x) (-4 * s  
  LINE (-41 * s + x) (-4 * s + y) (-43 * s + x) (-5 *  
  LINE (-43 * s + x) (-5 * s + y) (-44 * s + x) (-2 *  
  LINE (-44 * s + x) (-2 * s + y) (-44 * s + x) (2 * s  
  LINE (-44 * s + x) (2 * s + y) (-44 * s + x) (6 * s  
  //LINE (-44 * s + x) (6 * s + y) (-30 * s + x) (-6 *  
  LINE (-30 * s + x) (-6 * s + y) (-33 * s + x) (-4 *  
  LINE (-33 * s + x) (-4 * s + y) (-32 * s + x) (0 * s  
  LINE (-32 * s + x) (0 * s + y) (-28 * s + x) (1 * s  
  LINE (-28 * s + x) (1 * s + y) (-27 * s + x) (3 * s
```

Design By Numbers par John Maeda
<https://vimeo.com/875751048>

C'est une nouvelle génération de designers qui a été encouragé à explorer le potentiel créatif du code grâce à ce projet. Maeda et son projet *Design by Numbers* ont pu avoir une influence pédagogique sur le design computationnel. En effet, il a fournit un outil éducatif accessible, permettant aux étudiants d'expérimenter les bases du codage sans les complexités d'un langage de programmation classique.

```
print(  
«Type me Type me not»  
)
```

Type me Type me not a continué cette exploration en poussant les limites de la réactivité dans la typographie. Ce projet, mené par Peter Cho, sous la direction de John Maeda, examinait comment les caractères pouvaient être modifiés en temps réel en réponse aux entrées de l'utilisateur. L'accent mis sur l'interaction et la transformation dynamique des caractères a inspiré des avancées significatives dans le domaine de la typographie expérimentale.



25|

Type me Type me not, Partie A par Peter Cho

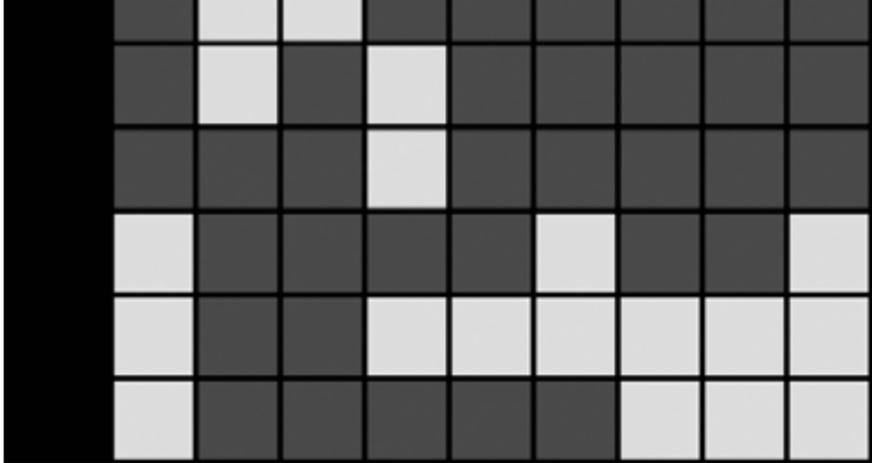
Dans cette première partie du projet, Cho propose un système typographique dans lequel chaque caractères est composé de formes géométriques de base. Il explore comment les lettres peuvent être déconstruites et reconstruites grâce à des règles géométriques utilisant des formes simples (lignes, cercles, rectangles...). Cette partie est basée sur les questions de flexibilité, de géométrie ainsi que de typographies variables. Peter Cho met donc en lumière l'ossature de la typographie et comment elle peut être modifiée et appréhendée.

La deuxième partie du projet est sûrement la plus expérimentale. Chaque lettre devient un objet animés et expressif réagissant aux stimuli externes comme le son ou les interactions de l'utilisateur. Cette partie dépasse la typographie statique, elle s'adapte à l'environnement dans lequel elle évolue. Dans le domaine du design computationnel, en intégrant des polices de caractères comme celle-ci, la typographie peut devenir un médium d'expression visuelle, permettant de nouvelles perspectives pour les designers d'animation, de motion ou typographes expérimentaux.



Type me Type me not, Partie B par Peter Cho

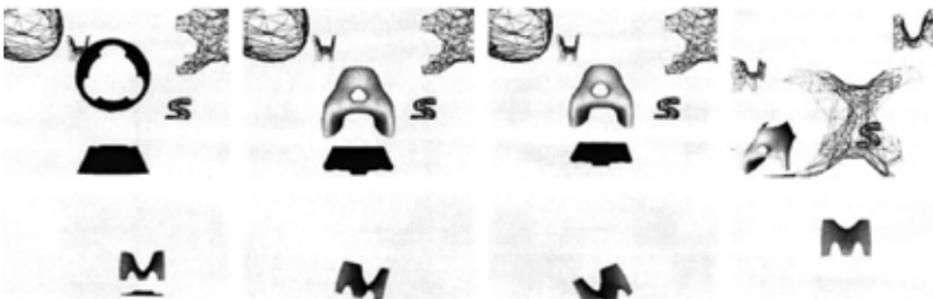
Dans cette dernière partie du projet, Cho conçoit chaque lettre à partir d'un ensemble de modules disposés sur une grille. Les lettres ne sont plus des formes géométriques continues, mais des compositions de blocs qui se combinent pour former les caractères. Chaque module est une unité qui peut être réarrangée, redimensionnée, et réorganisée, permettant de multiples configurations pour chaque lettre. Cette troisième partie du projet aborde donc l'aspect modulaire et combinatoire de la typographie. La grille confère une structure et permet une création rapide et flexible.



Type me Type me not, Partie C par Peter Cho

```
print(  
«Alphabet Zoo»  
)
```

27| *Alphabet Zoo*, également mené par Peter Cho au sein du ACG, était une aventure conceptuelle dans laquelle les lettres étaient transformées en créatures animées. Ce projet a exploré la personnalité et le caractère «psychologique» des caractères typographiques, élargissant notre compréhension du type design au-delà de son aspect statique. En introduisant une dimension ludique, *Alphabet Zoo* a influencé la manière dont les designers perçoivent et interagissent avec les caractères typographiques. Dans son projet, Cho dote chaque lettres de mouvements et de transformations visuelles qui semblent exprimer une personnalité unique.



Alphabet Zoo par Peter Cho
<https://vimeo.com/917910033>

Chaque caractère est indépendant, ce qui rappelle les typographies expressives, où les caractères ne sont pas simplement des éléments textuels, mais des symboles visuels personnifiés. Cette première approche au type design animé a également contribué au développement des typographies variables et des polices de caractères dynamiques réagissant à des paramètres interactifs.

En regardant vers l'avenir, l'influence de la genèse du type design au sein du ACG résonne dans les pratiques actuelles de nombreux designers et artistes qui continuent d'explorer les frontières de la typographie numérique. Les concepts de réactivité, d'interaction et de créativité computationnelle, initialement introduits par le ACG, demeurent des éléments essentiels dans le paysage dynamique du type design contemporain.

Ainsi, la genèse du type design au sein du ACG continue de nourrir une communauté créative engagée dans une exploration incessante de l'esthétique, de la technologie et de la signification derrière chaque caractère typographique.



```
print(  
«L'héritage de ces expérimentations  
et leurs dépassements»  
)
```

L'influence des expérimentations du Aesthetics and Computation Group (ACG) du MIT dans le domaine du dessin de caractères créatif sur ordinateur s'étend de manière significative jusqu'à nos jours, jouant un rôle crucial dans la redéfinition des pratiques artistiques et de conception numériques. Ces expérimentations ont transcédé leur époque, inspirant une génération d'artistes, de designers et de programmeurs qui ont continué à explorer les possibilités infinies offertes par l'alliance de l'esthétique et de la computation. Pour comprendre pleinement cette influence, examinons certains des acteurs clés contemporains qui ont été profondément marqués par les travaux du ACG.

```
print(  
«Creative Coding as a School of Thoughts»  
)
```

Tim Rodenbröker, dans sa thèse *Creative Coding as a School of Thoughts*, explore la notion de codage créatif en tant que paradigme de pensée. Il examine comment les expérimentations du ACG ont contribué à établir une fondation pour cette école de pensée, soulignant comment la programmation créative n'est pas simplement une compétence technique, mais plutôt une approche artistique et conceptuelle. Les idées explorées par le ACG ont influencé la façon dont les créateurs contemporains perçoivent et abordent la création numérique, plaçant l'accent sur la pensée computationnelle comme une extension naturelle du processus créatif. Contrairement aux méthodes de design traditionnelles, le creative coding encourage une approche interactive et expérimentale, où les erreurs, les accidents et les imprévus peuvent mener à de nouvelles découvertes. La thèse aborde comment le creative coding transforme le design computationnel en une discipline culturelle et artistique. En apprenant à coder, les designers peuvent contrôler et manipuler les algorithmes et les données, ce qui les libère des

limites imposées par les outils standards. Malgré que le lien avec notre sujet puisse paraître éloigné, les questions et les idées que Tim Rodenbröcker soulève dans sa thèse sont en direct lien et descendance avec les questionnement du ACG autour du type design computationnel.



31 |

Creative Coding as a School of Thoughts par Tim Rodenbröcker
<https://timrodenbroecker.de/creative-coding-as-a-school-of-thought/>

```
print(  
«How could virtual reality change  
typography?»  
)
```

How could virtual reality change typography? écrit par Eric Shrijver en 2014 est une référence notable dans le domaine du dessin de caractères virtuel. Cette exploration approfondie examine comment les technologies numériques ont redéfini notre compréhension de la typographie. Les expérimentations du ACG, avec leur penchant pour la typographie générative et la manipulation informatique, ont contribué à ouvrir la voie à des recherches telles que cet essai. Ce texte met en lumière comment la typographie peut transcender

les limites du matériel, du réel et devenir une entité virtuelle, façonnée par les nouvelles dimensions offertes par le numérique. Dans cet écrit, on comprend que le type design computationnel peut permettre une expansion de la typographie jusqu'au monde virtuel. Les interactions des utilisateurs permettent de « toucher » ou « d'explorer » de manière immersive la typographie dans un espace tridimensionnel.



*How could virtual reality change typography? par Erik Shrijver
<https://kabk.github.io/go-theses-17-Remco-Blom-Parallax/>*

```
print(  
«Weird Type»  
)
```

Zach Lieberman, en tant que pionnier du code créatif, a été profondément influencé par les expérimentations du ACG. Ses œuvres explorent l'intersection entre la technologie, le mouvement et la créativité. Les travaux du ACG, en particulier dans le domaine de la typographie générative, ont inspiré Lieberman à repousser les frontières de l'interaction entre l'utilisateur et la machine. Ses projets, tels que ceux qui réagissent au mouvement du corps pour générer des formes typographiques dynamiques, sont des exemples concrets de l'impact durable des expérimentations du ACG. Par exemple, Zach Lieberman en collaboration avec l'artiste

Molmol Kuo a développé une application nommée *Weird Type* en 2018. L'application permet de manipuler une ligne de texte (typographie fournie par la maison de design suisse Dinamo) en appliquant l'un des six filtres qui transforment la typographie en un objet physique dans l'espace. L'utilisateur a la possibilité de casser et faire exploser la typographie, la transformer en ruban, la fusionner avec une photo ou la laisser immobile alors que le monde autour d'elle bouge.

Bien qu'elle soit récente, l'application *Weird Type* perpétue l'héritage du Visible Language Workshop de Muriel Cooper, un groupe du Media Lab du MIT qui, dans les années 1990, expérimentait les premières versions de la typographie en 3D. À l'époque, les étudiants de Cooper utilisaient les premiers logiciels d'infographie pour détacher les caractères de leur grille en 2D. Ils faisaient flotter les mots dans l'espace, leur donnant ainsi une impression de profondeur et de mouvement. À l'époque, ces expériences donnaient un aperçu de l'avenir de l'édition numérique. Aujourd'hui, ces mêmes concepts deviennent enfin plus courants.

33|



Weird Type par Zach Lieberman et Molmol Kuo
<https://www.youtube.com/watch?v=UzNCGqE9MhE>

```
print(  
«Scriptographer»  
)
```

Jürg Lehni est un autre artiste-programmeur dont le travail est imprégné par les principes du ACG. Son exploration des modèles computationnels pour dessiner des caractères typographiques s'inscrit dans la lignée des expérimentations du ACG sur la typographie générative. Lehni a élaboré des outils créatifs tels que *Scriptographer* qui permettent une interaction directe avec le code pour générer des formes typographiques variées. Il s'agit d'un plugin pour Adobe Illustrator qui permet d'introduire des scripts de programmation dans le logiciel, transformant cet outil de dessin vectoriel en une plateforme de design computationnel. En utilisant JavaScript, *Scriptographer* permet aux utilisateurs de créer des scripts pour automatiser des tâches, générer des formes, manipuler des objets, et expérimenter avec des créations graphiques qui seraient complexes ou impossibles à réaliser manuellement. Son travail illustre comment les idées initiales du ACG ont évolué et ont été mises en pratique de manière innovante. L'outil *Scriptographer* a notamment été utilisé pour produire une série de poster pour l'artiste suisse Philippe Decrauzat.

| 34



Interface du plugin *Scriptographer* par Jürg Lehni

```
print(  
«Type & Form»  
)
```

35|

Karsten Schmidt (aussi connu sous le pseudonyme Toxi), incarne la fusion entre l'art, la programmation et le design génératif. Son engagement dans la création de logiciels artistiques reflète les principes du ACG, où la programmation devient un moyen d'expression artistique. Les expérimentations du ACG ont servi de bases pour des approches telles que celles de Schmidt, où la créativité trouve son expression à travers le code et la génération algorithmique de formes typographiques. Pour exemple, nous pouvons donner le projet *Type & Form*, réalisé en 2008. C'est un projet d'exploration typographique générative qui utilise le code pour créer des formes de lettres dynamiques et abstraites. Dans *Type & Form*, Schmidt utilise le code pour générer des variations typographiques en temps réel, démontrant comment des caractères peuvent être non seulement statiques, mais aussi devenir des entités vivantes et évolutives. Les formes de lettres sont générées à partir de règles et de paramètres programmés, permettant de créer des itérations infinies. Cette approche computationnelle permet de contrôler des aspects tels que la courbure, la densité, la taille et même le mouvement des lettres. Karsten Schmidt montre alors à la communauté du design computationnel, une approche de design où les créateurs utilisent le code non seulement comme un outil technique, mais aussi comme un médium créatif.



Type & Form par Karsten Schmidt

```
print(  
«Schrofer's illegible alphabet»  
)
```

Maurice Meilleur, avec son travail autour de *Schrofer's illegible alphabet* (l'alphabet illisible de Schrofer) datant de 2018, explore comment les systèmes contraints peuvent être appliqués à la typographie numérique. En reprenant les contraintes de Schrofer et en les réutilisant dans un contexte numérique Maurice Meilleur inscrit son projet dans la lignée des expérimentations du ACG sur les contraintes et les règles dans la création typographique. L'idée que les règles peuvent être des moteurs créatifs plutôt que des limitations est un héritage direct des premières expérimentations du ACG.



Schrofer's illegible alphabet par Maurice Meilleur

Avoid Software, Burrowlab et Constraints Systems sont autant d'autres noms qui reflètent l'influence élargie des expérimentations du ACG. Ces collectifs et studios ont intégré les principes de la créativité computationnelle dans leurs pratiques, explorant la typographie numérique sous des angles divers. Des collaborations interdisciplinaires, des installations interactives aux expérimentations algorithmiques, ces entités ont contribué à repousser les frontières de la typographie créative.

L'influence des expérimentations du ACG est omniprésente dans le tissu de la création contemporaine. Les artistes-programmeurs, designers génératifs et collectifs créatifs d'aujourd'hui sont les héritiers directs des idées novatrices explorées par le ACG au MIT. Les principes de la typographie générative, de l'interaction utilisateur-machine et de la créativité computationnelle sont devenus des éléments fondamentaux de la boîte à outils créative de nombreux professionnels.

37 | Les expérimentations du ACG au MIT ont donc eu un impact profond et durable sur le domaine du dessin de caractères créatif sur ordinateur. Les idées initiales du ACG ont évolué pour devenir des piliers fondamentaux de la créativité computationnelle moderne. Des artistes et des créateurs tels que Tim Rodenbröker, Eric Shrijver, Zach Lieberman, Jürg Lehni, Karsten Schmidt, Maurice Meilleur, Avoid Software, Burrowlab et Constraints Systems ont tous été façonnés par les principes explorés par le ACG. Aujourd'hui, l'héritage du ACG se manifeste dans une multitude de projets innovants et de pratiques artistiques qui continuent de repousser les frontières de la typographie numérique et de la créativité computationnelle.



```
print(  
«Conclusion»  
)
```

La trajectoire du dessin de caractères créatif sur ordinateur depuis les premiers travaux du Aesthetics and Computation Group (ACG) au MIT dans les années 80 jusqu'à aujourd'hui est fascinante et témoigne d'une évolution remarquable. Les pionniers du ACG ont tracé les contours d'une nouvelle ère où l'informatique et l'esthétique convergent pour redéfinir la typographie numérique. Dans les premières expérimentations du groupe, nous avons vu l'émergence de la typographie générative, explorant comment des algorithmes pouvaient être exploités pour créer des caractères et des compositions typographiques novatrices. Ces travaux ont ouvert des portes créatives inédites, donnant naissance à une diversité de formes et de styles qui transcendent les conventions traditionnelles.

39| Au fil des décennies, cette exploration esthétique s'est approfondie, influençant le paysage du design graphique et de la typographie numérique. Les artistes et les créateurs contemporains ont intégré les enseignements du ACG dans leurs pratiques, repoussant encore plus loin les limites de la créativité typographique sur ordinateur. La démocratisation des outils de conception et l'avènement de logiciels de pointe ont également contribué à l'évolution du dessin de caractères. Les concepteurs disposent aujourd'hui d'une panoplie d'outils puissants, leur permettant de donner vie à des idées qui auraient semblé futuristes à l'époque du ACG. L'interaction utilisateur-machine a atteint de nouveaux sommets, permettant une personnalisation accrue des polices de caractères et une expérience immersive pour les utilisateurs.

Les collaborations interdisciplinaires entre artistes, programmeurs et designers ont également marqué cette évolution. Les frontières entre le code, l'art et le design continuent de s'estomper, créant un terrain fertile pour l'innovation typographique. Des artistes contemporains explorent la relation complexe entre le texte et l'image, fusionnant ces deux éléments

pour créer des compositions visuelles évocatrices et expressives. En conclusion, le dessin de caractères créatif sur ordinateur a parcouru un long chemin depuis les premières expérimentations du ACG au MIT. Des fondements jetés par ces pionniers, émerge une réalité où la typographie numérique est devenue une forme d'art à part entière. Les travaux novateurs du ACG continuent d'inspirer la communauté créative, témoignant d'une évolution constante où la technologie et l'esthétique s'entrelacent pour redéfinir les normes du design typographique dans notre ère numérique.

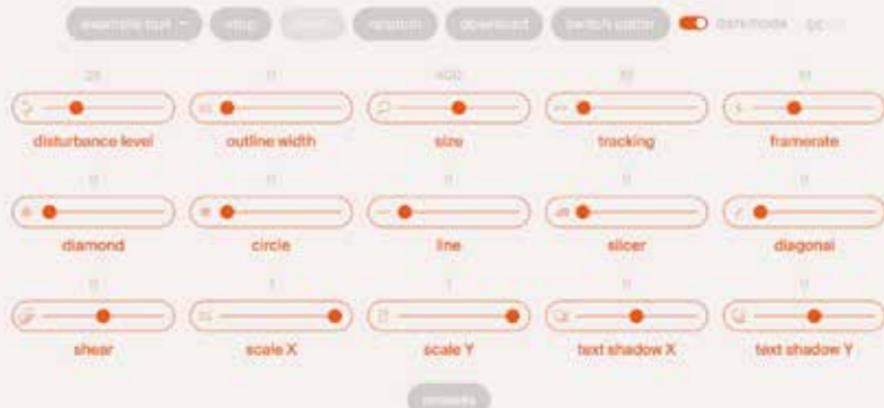
Alors que nous réfléchissons à cette remarquable évolution du dessin de caractères créatif sur ordinateur, il est tout aussi captivant d'anticiper son avenir, particulièrement dans le contexte en constante mutation de la réalité augmentée (AR) et virtuelle (VR). Ces nouvelles frontières technologiques ouvrent des horizons inexplorés pour le type design, offrant des possibilités de création immersive et interactive. Dans le monde de la réalité augmentée, les caractères typographiques peuvent transcender les écrans traditionnels pour s'intégrer dans notre environnement physique, créant des expériences visuelles inédites. De même, la réalité virtuelle offre un terrain de jeu infini où les utilisateurs peuvent interagir avec des caractères tridimensionnels, explorant ainsi une dimension spatiale et cinétique du type design. Les expérimentations à venir dans le domaine du type design en AR et VR promettent de repousser les limites de la créativité, plaçant le spectateur au cœur d'une expérience typographique immersive et engageante. Ainsi, la trajectoire fascinante du dessin de caractères s'inscrit désormais dans une ère où la convergence de l'informatique, de l'esthétique et des réalités augmentée et virtuelle façonne un nouveau chapitre passionnant pour le type design.



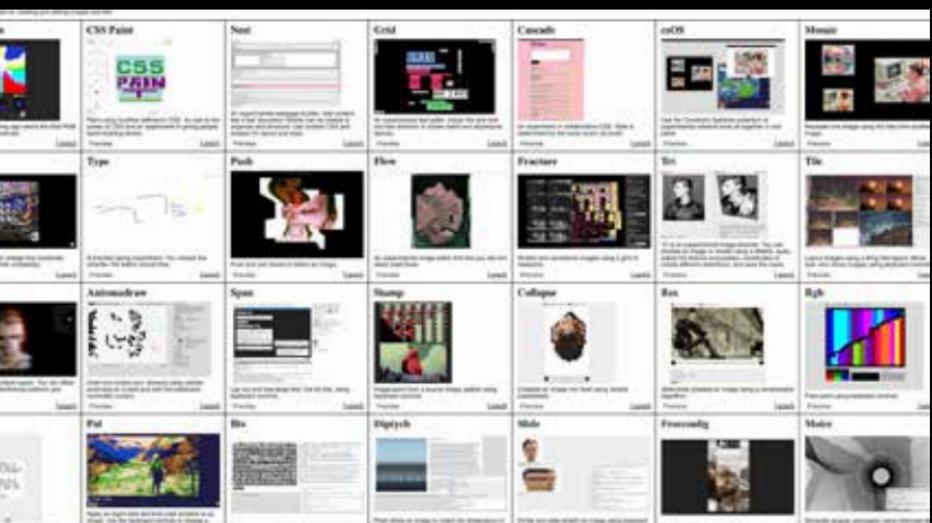
```
print(  
«Annexes»  
)
```



Interface Burrowlab

The word "STOER" is displayed in a white, pixelated, blocky font against a black background.

Interface Stoer Editor



Interface Constraint Systems

Rune



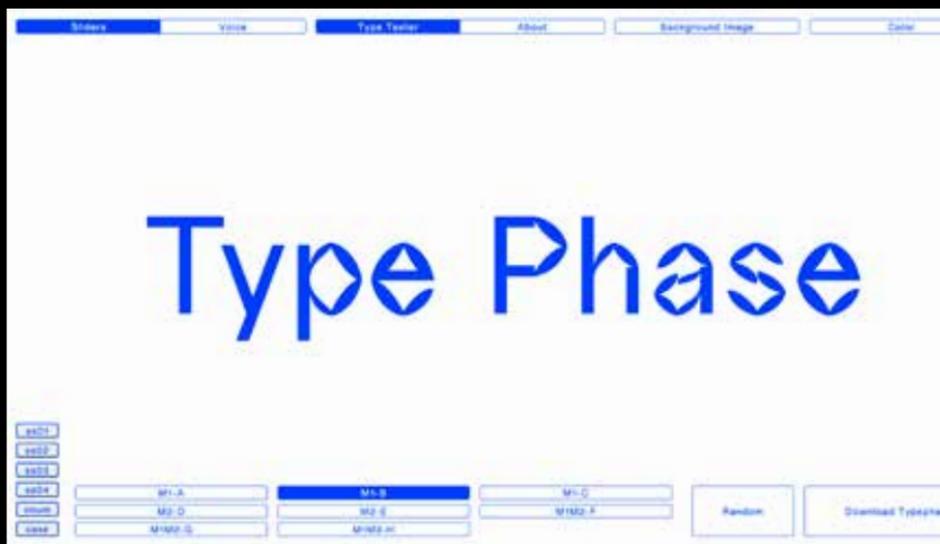
```
var r = new Rune({
  container: "#canvas",
  width: 800,
  height: 500
});

// Create a new Rune.Font object
var f = new Rune.Font("../Roboto-Regular.ttf");

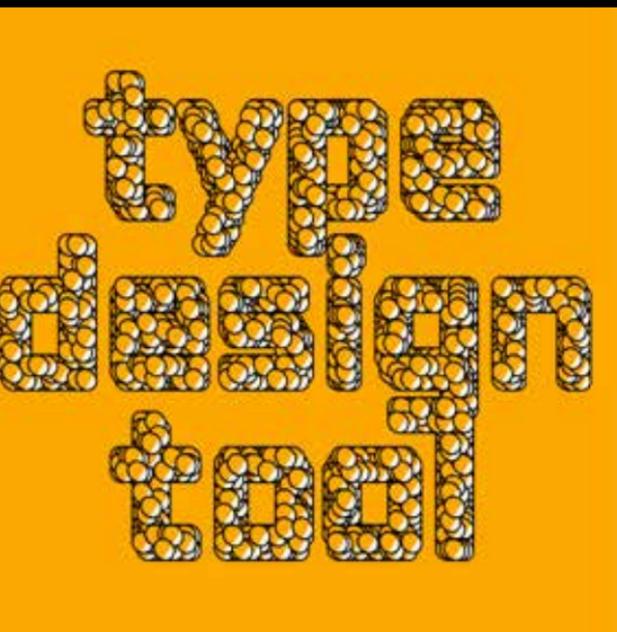
// load() will actually load the font file and call the function
// when the loading is done.
f.load(function(err) {
```

45 |

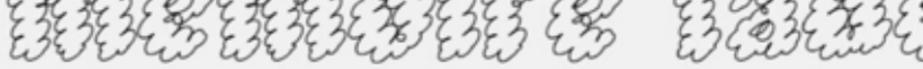
Interface du plugin Rune.font



Interface Phase



Interface RASTR Tool



10 kelloe²

hoquettes

10 kelloe² hoquettes

1. Spectral (coeff = 5, rot = 45, stroke = 10)
2. Compagnon Roman (coeff = 1, rot = 0, stroke = 10)
3. Bienchen SAS (coeff = 2, rot = 0, stroke = 20)

Test typographique avec l'outil 2cool4school développé par Avoid Software



Exemples de caractères dessinés sur NaN Generative Fonts



Interface Munken

simulierung züchtung GenLabio GenoTyp Laktose milch

Stammbaum

10 ahg ahg ahg ahg

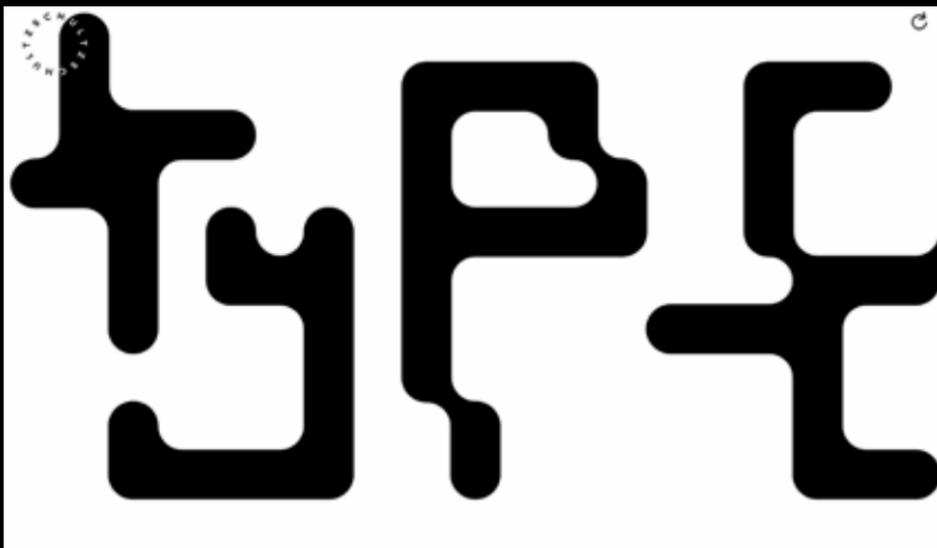
11 ahg ahg ahg ahg ahg

12 ahg ahg

Phänotyp Name P1_5 Stam P2_3 + P3_4

ahgeroimkep

Exemples de caractères dessinés en reprenant les codes de la génétique par Michael Schmitz sur GenoTyp



Interface Schultz Schultz



Exemples de caractères dessinés sur Fontstruct

A catalog of scripts that make the UFO workflow a little easier. Maintained by [Lucas Edmondson](#). Set in [Type](#) by David Jonathan Ross.

Search ...

TAGS

account glyphs accents anchors
 auto reference layout vtl surfaces
 user corners CORNER brackets

drawing [asciitext](#)

essentials information management

history glyph glyphs [glyph](#)

window [gms](#) guides

interpolation [math](#)

mastering [italics](#)

[mechanic](#) [metrics](#)

multiple fonts [operator](#)

overlap [paths](#) grid [preview](#)

radius [rotator](#) [rotator](#) [rotator](#)

save [search](#) [SPACE](#)

center [spacing](#) [size](#)

testing [verifying](#) [weird](#)

name

WurstSchreiber

by Alexandre Saumier Demers

This is easily the most useful extension one could develop. "WurstSchreiber is a tool for drawing segments as sausages in RoboFont."



Extention WurstSchreiber disponible sur Robofont



```
print(  
«Entretien»  
)
```

Interview de Maurice Meilleur (video-call 27/09/2024)

Émilie Roudaut

Hi, first of all thank you so much for agreeing to do an interview with me. Maybe you could start by presenting yourself?

Bonjour, tout d'abord merci beaucoup d'avoir accepté de faire cette interview avec moi. Vous pouvez peut-être commencer par vous présenter?

Maurice Meilleur

57| *Oh sure, I can talk about myself! Well I came to design in a kind of a weird way. So I actually started college as a piano performance major and then moved to political science and ended up getting a PHD in political theory in 2004 and then came to design from there. So while I was working at the university of Illinois I decided I would rather be a graphic designer and it kind of took my benefits and started it, and started teaching it. That's sort of like the quick bio but, I was born in Alaska and raised in Anchorage and then my family moved to Washington (state) in 1975. I went to school in eastern Washington and got my undergraduate from Washington city university in 1991 I think. I was also an exchange student in Germany, in Bonn from 1985 to 1986, so I've been visiting Europe for a long time. I went back a few times. I've been going more recently, a couple times every year because I study Jurriaan Schrofer. I've been visiting archives and talking to people that knew him... That seems very scattershot but yeah that's about it.*

Bien sûr, je peux parler de moi! Je suis rentré dans le monde du design d'une manière un peu bizarre. J'ai commencé mes études à l'université en tant que pianiste, puis je me suis orienté vers les sciences politiques

et j'ai obtenu un doctorat en théorie politique en 2004. Alors que je travaillais au sein de l'université d'Illinois, j'ai décidé que je préférais être graphiste et j'ai donc commencé à enseigner le graphisme. Pour un peu de contexte, je suis né en Alaska et j'ai grandi à Anchorage, puis ma famille a déménagé dans l'État de Washington en 1975. Je suis allé à l'école dans l'est de l'État de Washington et j'ai obtenu mon diplôme de premier cycle à l'université de la ville de Washington en 1991. J'ai également participé à un échange étudiant en Allemagne, à Bonn, de 1985 à 1986, ça fait donc quelques temps que je visite l'Europe. J'y suis allé plus récemment, quelques fois par an, parce que j'étudie Jurriaan Schrofer. J'ai visité des archives et parlé à des gens qui l'ont connu... Cela semble très dispersé, mais oui, c'est à peu près tout.

ÉR

Thank you so much for presenting yourself! As I told you in my emails, my thesis is entitled "The evolution of creative computer type design - From the ACG's Media Lab experimentations to today" and in my second part I'm speaking about you as kind of a descendant of these experiments. And so I was wondering what's your relationship with typography/type design?

|58

Merci beaucoup de vous être présenté! Comme je vous l'ai dit dans mes emails, mon mémoire s'intitule «L'évolution du dessin de caractères créatif par ordinateur - Des expérimentations du ACG au MIT Media Lab jusqu'à aujourd'hui» et dans ma deuxième partie, je parle de vous comme une sorte de descendant de ces expérimentations. Je me demandais donc quelle était votre relation avec la typographie/le dessin de caractères?

MM

So type was the reason that I got into graphic design in the first place. It was the first thing I noticed even though I was doing my PHD work. You know it's weird

how just noticing something small can just send you down, like basically creating an entire life for you. I think the first time I really noticed type was in the course of my studies, I ran across *The New Left Review* which is a journal published in London. I noticed the type in the journal and I'd never seen it before, it wasn't on my computer. And so I had lots of questions about it. And this was in 1996/1997, which is when a lot of design was just starting to move online. People were just discovering the web and starting to put a lot of work, there were a lot of discussions. And so there were boards like *Speak Up*, which is now defunct, but used to be like a big sort of design blog. Lots of designers and lots of type people were really active in conversations on that board. And so fortunately I had a place where I could ask questions and actually get answers from a lot of the people that were online. So that kind of opened all the doors for me. I became super interested in type. I started licensing typefaces, buying books about type and reading about it. And I actually - at the time, I mean this is a great story. I guess it was spring of 2004, *Fontshop International*, which at the time had offices in Berlin. And then there was an office in San Francisco. And the San Francisco office ran a competition because they wanted to re-keyword all of their typefaces. So they had a competition online where you could create an account and sign up, log in, and they would show you a typeface. And then they would ask you a series of questions about it. You would answer them. And then they would take your answers and log them along with other people's answers and put them in their database. And that was how they would keyword them. And for every 50 of those that you did, you would get a license for a free weight of whatever typeface you wanted. So I said to myself, I want all the licenses for *Scala* and for *Meta*. So I did 800 typefaces. But the other thing was that if you did 50, not only did you get the free weight, but you got put into a drawing for a free trip to Berlin and then to Brussels to meet *Erich Spiekermann* and go to *Typo Berlin* and then to go meet *Rudi Haraus* and *Fritz Mayers* in Brussels. And then they would drive you to like *Ghent* and show you the office and whatever. I won the contest. And I wasn't even a designer. And I remember that *Rudi Haraus* asked me when

we were at dinner, do you think that you'd ever, you know, you're so into type, do you think you would ever like design a typeface or get into design? I'm like, oh, no, no, no, I'm a political theorist. But anyway, here's where I wound up. So basically type is what got me into design. It's the center of my practice and it's the center of what I research. And it's what I teach. I teach advanced typography here at Iowa State every semester for students.

C'est donc grâce à la typographie que j'ai commencé à travailler dans le graphisme. C'est la première chose que j'ai remarquée alors que j'étais en train de faire mon doctorat. Vous savez, c'est étrange de voir à quel point le simple fait de remarquer quelque chose d'anodin peut vous faire sombrer, comme si vous vous étiez créé une vie entière. Je pense que la première fois que j'ai vraiment remarqué la typographie, c'était au cours de mes études. Je suis tombée sur la revue *The New Left Review*, une revue publiée à Londres. J'ai remarqué que la police de caractère m'était inconnue, elle n'était pas sur mon ordinateur. Je me suis donc posé beaucoup de questions à ce sujet. C'était en 1996/1997, à l'époque où une grande partie du design commençait à être mise en ligne. Les gens découvraient le web et commençaient à y travailler, il y avait beaucoup de discussions. Il y avait donc des forums comme *Speak Up*, qui n'existe plus aujourd'hui, mais qui était une sorte de grand blog sur le design. Beaucoup de designers et de typographes participaient activement aux conversations sur ce forum. J'avais un endroit où je pouvais poser des questions et obtenir des réponses de la part d'un grand nombre de personnes en ligne. Cela m'a ouvert toutes les portes. J'ai commencé à m'intéresser de près à la typographie. J'ai commencé à acheter des licences de caractères, à acheter des livres sur les caractères et à lire sur le sujet. Et j'ai en fait, à l'époque, c'est une belle histoire. Je crois que c'était au printemps 2004, *Fontshop International*, qui à l'époque avait des bureaux à Berlin. Il y avait aussi un bureau à San Francisco. Le bureau de San Francisco a organisé un concours parce qu'il voulait redéfinir les mots-clés de tous ses caractères. Ils ont donc organisé un concours en ligne où vous pouviez créer un compte,

vous inscrire, vous connecter, et ils vous montraient une police de caractères, et vous posaient une série de questions à son sujet. Vous deviez y répondre. Ensuite, ils prenaient vos réponses et les enregistraient avec les réponses d'autres personnes et les mettaient dans leur base de données. C'est ainsi qu'ils attribuaient des mots-clés. Et pour chaque 50 réponses, vous obteniez une licence pour un graisse gratuite de la police de caractères que vous vouliez. Je me suis donc dit que je voulais toutes les licences pour Scala et Meta. J'ai donc répondu à 800 polices de caractères. Mais l'autre chose, c'est que si vous en faisiez 50, non seulement vous obteniez une graisse gratuite, mais vous participiez à un tirage au sort pour un voyage gratuit à Berlin, puis à Bruxelles, pour rencontrer Erich Spiekermann et aller à Typo Berlin, puis pour rencontrer Rudi Haraus et Fritz Mayers à Bruxelles. Ensuite, ils vous conduisaient à Gand et vous montraient le bureau et tout le reste. J'ai gagné le concours. Et je n'étais même pas designer. Je me souviens que Rudi Haraus m'a demandé, lors d'un dîner, «penses-tu que tu pourrais un jour dessiner une police de caractères ou te lancer dans le design?» Je me suis dit, oh, non, non, non, je suis un théoricien politique. Quoi qu'il en soit, voilà où j'en suis arrivé. En fait, c'est la typographie qui m'a amené au design. C'est le centre de ma pratique et le centre de mes recherches. Et c'est ce que j'enseigne. J'enseigne la typographie avancée ici, à Iowa State, tous les semestres pour les étudiants.

ÉR

Indeed that's a great story! And how much does technology interfere with your type work? Do you use technology in the sense of not only Glyphs or FontLab but do you kind of modify type with the new technologies, like by programming for exemple?

C'est en effet une belle histoire! Et donc dans quelle mesure la technologie interfère-t-elle avec votre travail typographique? Utilisez-vous la technologie dans le sens où vous n'utilisez pas seulement Glyphs

ou FontLab mais modifiez vous les caractères avec les nouvelles technologies, comme la programmation par exemple?

MM

Yeah. So my practice is very much code-based - it was not originally. Code was something that I came slowly to. Like when I was doing my MFA (master of fine arts) in Illinois, I had two semesters of interactive design. But it didn't really stick with me because I didn't know what to do with it, right? And it was like HTML, CSS, and JavaScript. And I'm like I don't know what I'm going to do with this. But then I took Judith Van Rossum's Python and Drawbot workshop at Typographics in New York City in 2016. And it was right then, that I was starting to realize what I was super interested in studying and kind of writing about in terms of my research. And that was kind of when things clicked for me. So basically since 2016, code has been really central to my practice, writing Python and working in Drawbot specifically. Both creatively and also as a research tool. So the way that I study Jurriaan Schrofer's work is by basically rebuilding it with code. I mean he was obviously not a coder himself. There were a few places where he interacted with computers, but all of his stuff was by hand. But I've basically been recreating it to learn more about it with code. So yeah, it's central to my practice.

|62

Ma pratique est très largement basée sur le code, ce n'était pas le cas à l'origine. Le code est quelque chose qui m'est venu lentement. Par exemple, lorsque j'ai fait mon MFA (master d'arts) dans l'Illinois, j'ai suivi deux semestres de design interactif. Mais je n'ai pas vraiment accroché parce que je ne savais pas quoi en faire. C'était du HTML, du CSS et du JavaScript. Mais ensuite, j'ai suivi l'atelier Python et Drawbot de Judith Van Rossum à Typographics à New York en 2016. Et c'est à ce moment-là que j'ai commencé à réaliser ce que j'avais envie d'étudier et d'écrire dans le cadre de mes recherches. Et c'est à ce moment-là que les choses se sont enclenchées pour moi. Depuis 2016,

le code est au cœur de ma pratique, j'écris en Python et je travaille avec Drawbot. À la fois de manière créative et en tant qu'outil de recherche. Par exemple, la façon dont j'étudie le travail de Jurriaan Schrofer consiste à le reconstruire avec du code, il n'était évidemment pas un codeur lui-même. Il y avait quelques moments où il interagissait avec les ordinateurs, mais tout son travail était fait à la main. Mais je l'ai essentiellement recréé pour en apprendre davantage sur lui avec du code. Alors oui, c'est un élément central de ma pratique.

ÉR

In that context of code and programming did you know about the ACG Media Lab and all their experimentations around type and interactivity with type?

Dans ce contexte de code et de programmation, connaissiez-vous le Media Lab de l'ACG et toutes ses expérimentations autour de la typographie et de l'interactivité avec la typographie?

63|

MM

Not around type. I knew of people that were involved with MIT's Media Lab. So I'm familiar with Muriel Cooper and her work and her history. And I know who John Maeda is and Ben Fry and Casey Reyes. So those names I knew. I never really specifically associated them with typography. But just knew that they were very much involved in creative coding and didn't really even think of the ACG as such. I just kind of thought Media Lab. And I knew Processing a lot. And it's starting to creep back into my practice. Like I taught P5.js last spring to graduate students. But I kind of knew them. But not so much the ACG and not so much for the typography. More just contemporary currents in creative coding can be traced back very much to that group and to what they contributed.

Pas dans ce domaine. Je connaissais des personnes impliquées dans le Media Lab du MIT. Je connais donc Muriel Cooper, son travail et son histoire. Je connais également John Maeda, Ben Fry et Casey Reyes. Je connaissais donc ces noms. Je ne les ai jamais associés spécifiquement à la typographie. Je savais simplement qu'ils étaient très impliqués dans le codage créatif et je ne pensais même pas à l'ACG en tant que tel. J'ai juste pensé au Media Lab. Et je connaissais beaucoup Processing. Et cela commence à revenir dans ma pratique. Au printemps dernier, j'ai enseigné P5.js à des étudiants diplômés. Je savais seulement que les courants contemporains du codage créatif peuvent être attribués à ce groupe et à ce qu'ils ont apporté.

ÉR

I actually discovered them during my research. I was interested in how can we modify the letters with code. I started with Processing, and then I started to question how did they do it before Processing and the creative coding we know.

En fait, je les ai découverts au cours de mes recherches. Je me suis intéressée à la manière dont nous pouvions modifier les lettres avec un code. J'ai commencé par Processing, puis je me suis demandé comment ils procédaient avant Processing et le codage créatif que nous connaissons.

MM

But you know, you've probably run across this before. But one of the things that I've noticed about Processing. Well, I mean now, at the time, I didn't really kind of understand this. But it became more important as I started to use more code in my work. Is that Processing as an environment and as a library of code it doesn't actually work very well with type, right? It's not a very sophisticated way of working with type. I tried to modify letters and stuff with Processing. And then I realized it was not made for

it. And so one of the things that really attracted me to working in Python and specifically working in Python in DrawBot is that DrawBot was created by people at the KABK (Royal Academy of fine arts in Den Haag, Netherlands) who were, you know, basically teaching the Type Media Masters of Type Design program. And they incorporated into DrawBot a library called Font Tools. Which is basically like the Python library that you use in applications like RoboFont to manipulate, to work with type and to manipulate outlines. You know, and that's not an accident, obviously, because they were trying to teach their students like, okay, here's the basics of coding in Python because now you can go on and you can write scripts and glyphs or, you can basically create your own tools.

65 | Mais vous savez, vous avez probablement déjà rencontré cela auparavant. Mais l'une des choses que j'ai remarquées à propos de Processing. Eh bien, je veux dire maintenant, à l'époque, je n'ai pas vraiment compris cela. Mais c'est devenu plus important lorsque j'ai commencé à utiliser plus de code dans mon travail. C'est que Processing, en tant qu'environnement et bibliothèque de code, ne fonctionne pas très bien avec le type design, n'est-ce pas? Ce n'est pas une façon très sophistiquée de travailler avec les caractères. J'ai essayé de modifier des lettres et d'autres choses avec Processing. Puis j'ai réalisé qu'il n'était pas fait pour cela. C'est ainsi, qu'une des choses qui m'ont vraiment attiré dans le travail en Python, et plus particulièrement dans DrawBot, c'est que DrawBot a été créé par des gens de la KABK (Académie royale des beaux-arts de La Haye, Pays-Bas) qui enseignaient le programme de maîtrise en dessin de caractères de Type Media. Ils ont intégré à DrawBot une bibliothèque appelée Font Tools. Il s'agit en fait de la bibliothèque Python que vous utilisez dans des applications comme RoboFont pour manipuler, travailler avec des caractères et manipuler des contours. Vous savez, et ce n'est pas un accident, ils essayaient d'enseigner à leurs étudiants une chose comme, ok, voici les bases du codage en Python, maintenant vous pouvez aller de l'avant et vous pouvez écrire des scripts, dessiner des glyphes et créer vos propres outils.

ÉR

So you spoke about the use of technology in your work and in your typographic work. And an other question I had was, can you tell me a bit more about your project Animated Martens letters that you did in 2018?

Vous avez donc parlé de l'utilisation de la technologie dans votre travail et dans votre travail typographique. Et j'avais une autre question, pouvez-vous m'en dire un peu plus sur votre projet Animated Martens letters que vous avez réalisé en 2018?

MM

I don't know if you're familiar with Karel Marten's monoprints, but I just remember this was around the time that I was really kind of discovering, or this comes out of like me noticing those monoprints and really being fascinated by the work that he was doing with them. And me just basically trying to think, what if I took shapes like the ones he had and just imagine constructing letters out of them? And it was a project that I always meant to come back to at some point, I think I got to T. Around the time that I was animating these, was about the time that I had to go back on the job market. I was trying to get out of Appalachia. And so I never did come back to it, but what I really liked about, what I like about Martens' work generally, is his mentality towards the colors and shapes. He was just really fascinated with things like how color interacts with itself and how shapes interact with each other. And just trying to explore the possibilities of those things. And I tried to bring that into this project and just imagine, like any other kind of constructed script project I'm trying to make, trying to discover some of the things that letters could be, by using sort of a very limited set of forms. So what are some of the possibilities for those and how can I make them more interesting to look at? So I guess that in many ways, this project was just another avenue for thinking about typographic, basically thinking about type as form, right? And just saying, you know, like leave off maybe for a moment, the fact that

it represents a character in a writing system, just what are these forms like? You know, how can they be broken down and recombined and explored?

Je ne sais pas si vous connaissez les monotypes de Karel Marten, mais je me souviens que c'est à peu près à l'époque où je l'ai découvert, ou plutôt où j'ai remarqué ces monotypes et où j'ai été fasciné par le travail qu'il faisait avec eux. Et j'ai essayé de me dire e et si je prenais des formes comme celles qu'il avait et que j'imaginai construire des lettres à partir d'elles? C'est un projet auquel j'ai toujours voulu revenir à un moment donné, je pense que je suis arrivé jusqu'à T. C'est également à peu près au moment où j'ai dû retourner sur le marché du travail. J'essayais de quitter les Appalaches. Je n'y suis donc jamais revenu, mais ce que j'ai vraiment aimé, et ce que j'aime dans le travail de Martens en général, c'est sa mentalité à l'égard des couleurs et des formes. Il était vraiment fasciné par la façon dont les couleurs interagissent entre elles et dont les formes interagissent les unes avec les autres. Il essayait d'explorer les possibilités de ces relations. J'ai essayé d'intégrer cela dans mon projet et d'imaginer, comme dans tout autre projet que 67 | j'essaie de réaliser, de découvrir toutes les choses que les lettres pourraient être, en utilisant une sorte d'ensemble très limité de formes. Quelles sont donc les possibilités de ces lettres et comment puis-je les rendre plus intéressantes à regarder? Je suppose donc qu'à bien des égards, ce projet n'était qu'une autre façon de penser à la typographie, de penser à la typographie en tant que forme, n'est-ce pas? Et juste laisser de côté peut-être pour un moment, le fait qu'il représente un caractère dans un système d'écriture, juste à quoi ressemblent ces formes? Comment peut-on les décomposer, les recombinaison et les explorer?

ÉR

I didn't know his work at all, but it sounds fascinating! These are also questions that interest me and are in line with the questions I'm asking myself in this thesis. How can type design go further? By the way,

my last two questions are in this context, how do you see the future of type design and how would you define the evolution of type design?

Je ne connaissais pas du tout son travail, mais ça a l'air fascinant! Ce sont aussi des questions qui m'intéressent et qui rejoignent celles que je me pose dans mon mémoire. Comment la création de caractères peut-elle aller plus loin? D'ailleurs, mes deux dernières questions s'inscrivent dans ce contexte comment voyez-vous l'avenir de la création typographique et comment définiriez-vous l'évolution de la création typographique?

MM

Yeah, these are great questions actually! Why don't we start with the evolution thing first? So one of the things that struck me when I read that question in your email is I thought, well, you know, like all new technologies, as digital type emerged, one of the first priorities for the people that were involved in that, were trying to basically reproduce what existing technologies were already doing. They're trying to solve problems in existing typographic technology. But they're trying to do it in a way that's recognizable. They're trying to do in a way that basically reproduces what people already expect from type. And you still see that a lot in the innovations that have happened since. So if you think, okay, depending on when you wanna date digital typography, it either goes back to the 60s with bitmaps or, you can think about, moving into like digital fonts that we would recognize them, like the first Icarus fonts and then PostScripts in the 80s and that kind of thing. In all of those stages, the priorities are basically let's try to solve a problem, but still keep things, keep doing things the way we're already doing them. And so you're really focused on things like, staying true to how you imagine letter forms look like, how do you reproduce things with the new technology? How do you do things that you couldn't do? That comes along to me kind of later, that people are thinking about questions of what would computational

type look like, but they're not becoming really prominent in the field until like years after.

69 | So I think about Donald Knuth, and Metaphon in the late 70's and early 80's, he's laying the groundwork for later parametric type design, even though he doesn't know it yet, or he kind of sees it, but the rest of the field is not really there yet. They're not really thinking about him in that context. And so they're evaluating it against these old technologies. And so what I've sort of seen as the history of type is a move away over time from reproducing what older technologies do. Because I think now we're getting to the point where people are starting to say, what can digital type do sort of intrinsically, like that nothing else can do, right? And you're still seeing a lot of people doing conventional things, but I think you're starting now to see people really starting to explore. Like, wow, there's like stuff that you can do. Like one of the things that variable type does, the open type variable format. It's start to breach, I think of, you know, what used to be a very longstanding wall between like, the design of type on the one side and the context in which it's applied on the other. That for a very long time in typography, at least in Western Europe, from Gutenberg up until like the mid 20th century. You've got the instructions for how to draw letter forms over here stored somehow. They're in metal or in wood or film or whatever. And you've got the places where you apply them on the other. And I think one of the things that open type variable does is it starts to break that barrier. Where information from context starts to affect the set of instructions for how type is drawn. And I think that's a real promising development, in typography. And I would like to see that, like my preference, is I think, on the one hand, I would like to see the conventional stuff continue. I mean, because one of the other things that's happened is that type tech has gotten more sophisticated. More and more writing systems now can be dealt with as better as digital type. So Arabic is a really good case in point where when it's in metal, it's like an imitation, like a very distant imitation of what actual written Arabic looks like, because you're stuck in rectangular boxes, stuck next

to each other. When you've got this sort of writing system, which is very calligraphic and all the letters have different forms depending on the position. And then there's like this sort of upturned baseline. And now we have digital type tech that allows us to do that. That allows you to set type properly that way. That's great. And I really appreciate that. But I would like to see more formal experimentation. I would like to see more people starting to question that boundary between the two. Between like context and instructions and play around with that. And I think those were the two. Is that it? Yeah, evolution and future. I think that's what I wish for the future. There you go. We went from one right into the other.

Oui, ce sont d'excellentes questions en fait! Pourquoi ne pas commencer par la question de l'évolution? L'une des choses qui m'a frappée lorsque j'ai lu cette question dans votre email, c'est que je me suis dit que, comme toutes les nouvelles technologies, lorsque la typographie numérique est apparue, l'une des premières priorités des personnes impliquées dans ce domaine a été d'essayer de reproduire ce que les technologies existantes faisaient déjà. Ils essaient de résoudre les problèmes de la technologie typographique existante. Mais ils essaient de le faire d'une manière reconnaissable. Ils essaient de reproduire ce que les gens attendent déjà de la typographie. Et c'est ce que l'on constate encore souvent dans les innovations qui ont eu lieu depuis. Ainsi, selon la date à laquelle vous voulez dater la typographie numérique, elle remonte soit aux années 60 avec les bitmaps, soit à l'évolution vers des polices numériques que nous pourrions reconnaître, comme les premières polices Icarus, puis PostScripts dans les années 80 et ce genre de choses. À tous ces stades, les priorités sont essentiellement d'essayer de résoudre un problème, tout en conservant les choses, en continuant à faire les choses de la manière dont nous les faisons déjà. Vous vous concentrez donc sur des aspects tels que la fidélité à l'image que vous vous faites des lettres, la manière de reproduire les choses avec la nouvelle technologie, la manière de faire des choses que vous ne pouviez pas faire. Comment faire des choses que vous ne pouviez pas faire? Pour moi,

ce n'est que des années plus tard que l'on commence à s'intéresser à cette question.

Je pense donc à Donald Knuth et à Metaphon à la fin des années 70 et au début des années 80. Il jette les bases de la conception paramétrique ultérieure, même s'il ne le sait pas encore, mais le reste du secteur n'en est pas encore là. Ils ne pensent pas vraiment à lui dans ce contexte. Ils l'évaluent par rapport à ces anciennes technologies. Ce que j'ai vu dans l'histoire de la typographie, c'est qu'au fil du temps, on s'est éloigné de la reproduction des anciennes technologies.

71 | Je pense que nous arrivons maintenant à un point où les gens commencent à se demander ce que la typographie numérique peut faire intrinsèquement, ce que rien d'autre ne peut faire, n'est-ce pas? Il y a encore beaucoup de gens qui font des choses conventionnelles, mais je pense que l'on commence à voir des gens qui commencent vraiment à explorer. Il y a des choses que l'on peut faire. Par exemple, l'une des choses que fait le type variable, le format variable de type ouvert. Il commence à briser, je pense à, vous savez, ce qui était un mur très ancien entre la conception de la typographie d'un côté et le contexte dans lequel elle est appliquée de l'autre. C'est ce qui s'est passé pendant très longtemps en typographie, du moins en Europe occidentale, depuis Gutenberg jusqu'au milieu du 20e siècle. Les instructions sur la façon de dessiner les formes de lettres sont stockées ici d'une manière ou d'une autre. Elles sont en métal, en bois, en film ou autre. Et vous avez les endroits où vous les appliquez sur l'autre. Et je pense que l'une des choses que fait la variable de type, c'est qu'elle commence à briser cette barrière. L'information provenant du contexte commence à affecter l'ensemble des instructions relatives à la manière dont les caractères sont dessinés. Je pense qu'il s'agit là d'une évolution prometteuse en matière de typographie. Et j'aimerais voir cela, je pense que, d'une part, j'aimerais voir les choses conventionnelles se poursuivre. Je veux dire, parce que l'une des autres choses qui s'est produite, c'est que la technologie de la typographie est devenue plus sophistiquée. De plus en plus de systèmes d'écriture peuvent être traités aussi bien

que les caractères numériques. L'arabe est un très bon exemple, lorsqu'il est en métal, c'est une imitation, une imitation très éloignée de ce à quoi ressemble l'arabe écrit, parce que vous êtes coincé dans des boîtes rectangulaires, collées l'une à côté de l'autre. Le système d'écriture est très calligraphique et toutes les lettres ont des formes différentes en fonction de leur position. Et puis il y a cette sorte de ligne de base renversée. Aujourd'hui, nous disposons d'une technologie de typographie numérique qui nous permet de faire cela. Cela vous permet de définir le type de manière appropriée. C'est très bien. Et j'apprécie vraiment cela. Mais j'aimerais voir une expérimentation plus formelle. J'aimerais voir plus de gens commencer à remettre en question la frontière entre les deux. Entre le contexte et les instructions, et jouer avec cela. Et je pense que ce sont les deux réponses à vos questions. Oui, l'évolution et l'avenir. Je pense que c'est ce que je souhaite pour l'avenir. Voilà. Nous sommes passés de l'un à l'autre.

ÉR

|72

*We'll see if you desires for the future come true!
Anyway, these were all my questions for you. Thanks
you so much for anwsering them and for giving me some
of your time. you've been really easy to talk to.
If you want I'll send you a copy of my thesis once
it's finished!*

Nous verrons bien si vos désirs pour l'avenir se réalisent! En tout cas, c'était toutes les questions que j'avais à vous poser. Merci beaucoup d'y avoir répondu et de m'avoir accordé un peu de votre temps. Si vous le souhaitez, je vous enverrai une copie de mon mémoire une fois qu'il sera terminé!

MM

*No problem. Happy to help. Let me know if you need
anything else. And of course, I would love to read it!
Take care and good luck!*

Pas de problème. Je suis heureux de pouvoir vous aider.
Faites-moi savoir si vous avez besoin d'autre chose.
Et bien sûr, je serais ravi de le lire! Prenez soin
de vous et bonne chance!

ÉR

Have a nice day, bye ;)

Passez une bonne journée, à bientôt ;)



```
print(  
«Glossaire»  
)
```

Algorithme

Un algorithme est une suite finie et non ambiguë d'instructions et d'opérations permettant de résoudre une classe de problèmes.

Augmented Reality

Augmented reality (AR), réalité augmentée en français, consiste en la superposition de la réalité et d'éléments (sons, images 2D, 3D, vidéos, etc.) calculés par un système informatique en temps réel. Elle désigne souvent les différentes méthodes qui permettent d'incruster de façon réaliste des objets virtuels dans une séquence d'images. Elle s'applique aussi bien à la perception visuelle (superposition d'images virtuelles aux images réelles) qu'aux perceptions proprioceptives comme les perceptions tactiles ou auditives.

75|

Creative Coding

Code créatif en français. Consiste à utiliser la programmation informatique comme moyen d'expression artistique.

Dessin de caractères

Le dessin de caractères (Type design en anglais) est l'art et le processus de conception des caractères. Il s'agit de dessiner chaque forme de lettre en utilisant un style cohérent.

Kinetic Typography

Kinetic Typography (typographie cinétique en français) le nom technique de « texte en mouvement » est une technique d'animation qui mélange le mouvement et le texte pour exprimer des idées au moyen de l'animation vidéo. Ce texte est présenté au fil du temps de manière à transmettre ou évoquer une idée ou une émotion particulière. En classification, la typographie cinétique est une forme de typographie temporelle (typographie qui se présente au fil du temps).

MetaFont

MetaFont est un langage de programmation permettant de générer des fontes de caractères. Il fut créé par le mathématicien et informaticien Donald Knuth en 1977 en même temps que TeX, un logiciel de composition de documents depuis utilisé par une majeure partie de la communauté scientifique internationale.

Modèle Computationnel

| 76

Un modèle computationnel, est un modèle qui fait des calculs à partir d'informations données en entrée, et qui donne en sortie un résultat numérique.

PostScript

PostScript (souvent abrégé en PS) est un langage de description de page et un langage de programmation basé sur une pile. Il est le plus couramment utilisé dans le domaine de l'édition électronique et de la publication assistée par ordinateur, mais en tant que langage de programmation complet de Turing, il peut également être utilisé à de nombreuses autres fins. PostScript a été créé chez Adobe Systems par John Warnock, Charles Geschke, Doug Brotz, Ed Taft et Bill Paxton de 1982 à 1984. La version la plus récente, PostScript 3, est sortie en 1997.

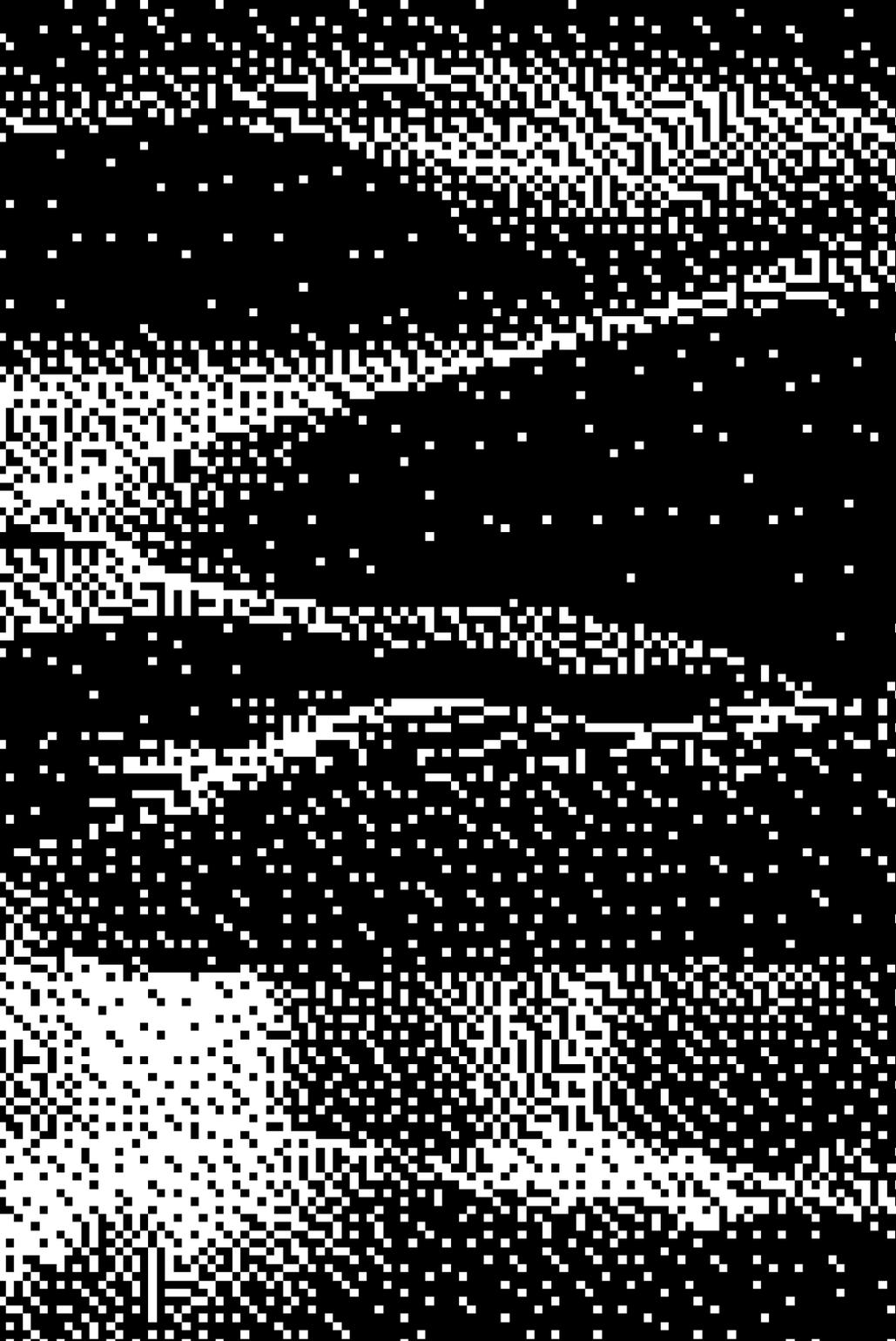
Temporal Typography

La temporal typography (typographie temporelle) est une sorte de typographie qui semble se déplacer ou changer d'apparence au fil du temps. Elle apparaît normalement dans les médias basés sur l'écran, et en particulier dans les séquences de titre, les identités de station de télévision, et la publicité. Semblable à la typographie cinétique, la typographie temporelle porte le sceau du temps mais n'est pas reléguée au mouvement ou aux médias temporels. Les formes temporelles ont la capacité de se manifester de manière statique et cinétique, ainsi que physique et numérique, et ne peuvent donc pas être évaluées par les mêmes facteurs fonctionnels de la typographie traditionnelle : la lisibilité. L'audience et la perception sont élevées, et les conditions formelles et expérientielles sont redéfinies au fur et à mesure que les formes passent d'un état à l'autre.

Virtual Reality

77|

Virtual Reality (VR), ou réalité virtuelle renvoie typiquement à une technologie informatique qui simule la présence physique d'un utilisateur dans un environnement artificiellement généré par des logiciels. La réalité virtuelle crée un environnement avec lequel l'utilisateur peut interagir. La réalité virtuelle reproduit donc artificiellement une expérience sensorielle, qui peut inclure la vue, le toucher, l'ouïe et l'odorat (visuelle, sonore ou haptique).



```
print(  
«Références»  
)
```

Bibliographie

BELLANTONI, Jeff. WOOLMAN, Matt. *Moving Type, Designing for Time and Space*, Angleterre : RotoVision, 2000.

BRUCKMAN, Amy. RESNICK, Mitchel. *Virtual Professional Community : Results from the MedaiMOO Project*, États-Unis : Presented at the Third International Conference on Cyberspace in Austin, 1993.

CARTER, Matthew. *Typography and Current Technologies*, Design Quarterly No.148, États-Unis : Walker Art Center and MIT, 1990.

CHO, Peter. *Pliant Type : Development and Temporal Manipulation of Expressive, Malleable Typography*, Bachelor's thesis, États-Unis : Massachusetts Institute of Technology, 1997.

79| **CHO, Peter.** *Computational Models for Expressive Dimensional typography*, Thesis, États-Unis : Massachusetts Institute of Technology, 1997.

COOPER, Muriel. *Computers and Design*, Design Quarterly No.142, Cambridge : MIT Press, 1989.

ISHIZAKI, Suguru. *Typographic Performance : Design Solutions as Emergent Behaviors of Active Agents*, PHD thesis proposal, États-Unis : Massachusetts Institute of Technology, 1994.

KNOTH, Christoph. *Computed Type Design*, Master's thesis, Suisse : École cantonale d'art de Lausanne, 2011.

KNUTH, Donald. *Computer Modern Typefaces*, États-Unis : Addison Wesley Professional, 1986.

KRAM, Reed. *System Models for Performance*, Master Thesis Proposal, États-Unis : Massachusetts Institute of Technology, 1997.

MAEDA, John. *Design by Numbers*, États-Unis : MIT Press, 2001.

MILLER, Abbott. *Dimensional Typography*, États-Unis : Princeton Architectural Press, 1996.

MILLS, Carol. WELDON, Linda. *Reading Text from Computer Screens*, États-Unis : ACM Computing Surveys, Vol 19. No.19, 1987.

POTTER, Mary. *New Methods in Reading Comprehension Research, Chapitre Rapid Serial Visual Presentation RSVP : A Method for Studying Language Processing*, Angleterre : Routledge Revivals, 1984.

TERESTCHENKO ROCHA, Ivan. *A Metaphysical and Linguistic Approach to Type Design and Typography*, Thesis, États-Unis : Rochester Institute of Technology, 1993.

SHAMIR, Ariel. RAPPOPORT, Ari. *Feature-Based Design of Fonts using Constraints*, Israël : Institute of Computer Science, 1998.

| 80

SMALL, David. *Expressive Typography: High Quality Dynamic and Responsive Typography in the Electronic Environment*, Master Thesis, États-Unis : Massachusetts Institute of Technology, 1990.

WONG, Yin Yin. *Temporal Typography : Characterization of Time-Varying Typographic Forms*, Master Thesis, États-Unis : Massachusetts Institute of Technology, 1995.

Sitographie

RODENBRÖKER, Tim. *Creative Coding as a school of thought*. TRCC. 22 juin 2022.
<https://timrodenbroeker.de/creative-coding-as-a-school-of-thought/> [Consulté le 18 décembre 2023].

MAEDA, John. *The law of simplicity*. Maeda Studio.
<http://lawsofsimplicity.com/>
[Consulté le 18 décembre 2023].

Design by Numbers. 2008. Wikipédia. https://en.wikipedia.org/wiki/Design_By_Numbers/
[Consulté le 22 décembre 2023].

CHO, Peter. *Type Me, Type Me Not*. <https://pcho.net/typemenot/> [Consulté le 26 décembre 2023].
The aesthetics + computation group. ACG. <https://acg.media.mit.edu/projects/> [Consulté le 26 décembre 2023].

MAEDA, John. *Reactive Books 1994-1999*. 2005.
Maeda Studio. http://maedastudio.com/2004/rbooks2kindexphp?this=reactive_books
[Consulté le 04 janvier 2024].

81|

Interface Graphique. 2004. Wikipédia. https://fr.wikipedia.org/wiki/Interface_graphique/
[Consulté le 08 janvier 2024].

MEILLEUR, Maurice. *Motion*. Maurice Meilleur.
https://mauricemeilleur.net/motion_index
[Consulté le 12 janvier 2024].

Vidéos et Podcasts

MAEDA, John. *Reactive Books (1995 to 1999).*

2015. Disponible sur <https://vimeo.com/124707805>

[Vidéo consultée le 01 janvier 2024].

MAEDA, John. *A collection of work from the MIT Media Lab Aesthetics + Computation Group (1998-2000).*

2013. Disponible sur <https://vimeo.com//71996582>

[Vidéo consultée le 04 janvier 2024].

MARTEL, Frédéric. *L'Alphabet numérique - MIT Media*

Lab. 2015. Disponible sur <https://www.radiofrance.fr/franceculture/podcasts/l-alphabet-numerique/mit-media-lab-2124520> [Podcast consulté le 10 janvier 2024].



```
print(  
«Remerciements»  
)
```

Je remercie Alexandra Aïn pour ses conseils et ses relectures attentives. Je remercie Maurice Meilleur d'avoir pris le temps de répondre à mes questions dans une interview. Je remercie Julien Bidoret qui m'a aidé dans la mise en page web et la mise en ligne de mon mémoire. Enfin, je remercie Solange Caillon pour son aide dans la conception de cet ouvrage.



*Ouvrage écrit et conçu par Émilie Roudaut
dans le cadre du DNSEP option Nouveaux Médias.
Imprimé sur les presses de l'École Supérieure
d'Art et de Design des Pyrénées.*

*Intérieur de l'ouvrage imprimé sur du papier
Digital Color Printing Clairefontaine White 160g.
Couverture de l'ouvrage imprimée sur du papier
Digital Color Printing Clairefontaine White 300g.*

*Composé avec la police de caractères Hack,
dessinée par Chris Simpkins.*

2025






```

    font = ImageFont.truetype(font_path, font_size)
    text_width, text_height = draw.textsize(text, font=font)
    x = (width - text_width) // 2
    y = (height - text_height) // 2

    if effects:
        for effect in effects:
            if effect == "glitch":
                draw_glitch_effect(draw, text, font, x, y)
            elif effect == "wave":
                draw_wave_effect(draw, text, font, x, y)
            elif effect == "distort":
                draw_distorted_text(draw, text, font, x, y)
        else:
            draw.text((x, y), text, font=font, fill="black")

def draw_glitch_effect(draw, text, font, x, y):
    for offset in range(-5, 6, 2):
        color = random.choice(["red", "blue", "green"])
        draw.text((x + offset, y), text, font=font, fill=color)

def draw_wave_effect(draw, text, font, x, y):
    spacing = 10
    for i, char in enumerate(text):
        offset = int(10 * random.sin(i / spacing))
        draw.text((x + i * 40, y + offset), char, font=font,
fill="black")

def draw_distorted_text(draw, text, font, x, y):
    for i in range(len(text)):
        jitter_x = random.randint(-3, 3)
        jitter_y = random.randint(-3, 3)
        draw.text((x + i * 40 + jitter_x, y + jitter_y), text[i],
font=font, fill="black")

if __name__ == "__main__":
    text_to_display = " L'évolution du
dessin de caractères créatif par
ordinateur, dès les expérimentations
du ACG MIT Media Lab, jusqu'à nos
jours. "

    output_file = "text_design_output.png"
    applied_effects = ["glitch", "wave", "distort"]

    generate_text_image(text_to_display, font_file, output_file,
effects=applied_effects)

```